Universidad Nacional de La Pampa, Facultad de Ingeniería

PROYECTO FINAL DE GRADO DE INGENIERÍA EN SISTEMAS

"Diseño y automatización de pruebas de sistema utilizando una estrategia de pruebas de software basada en escenarios y consciente de la situación"

Autor: Fernandez Reale Leonel Cristian Andres.

Director: Becker, Pablo.

Co-Director: Olsina, Luis.

Presentado: General Pico, La Pampa. 2023 – Aprobado el día 23/03/2023.

Jurados: Lafuente Guillermo, Rivera Belen, Filippi Jose. **Filiación**: Facultad Ingeniería UNLPam.

Resumen: la prueba de software es una etapa importante en el ciclo de desarrollo del software ya que permite identificar defectos y alcanzar una estabilidad en el producto. En la actualidad las aplicaciones se han vuelto muy complejas debido a que se relacionan con otras aplicaciones o servicios. En consecuencia, es difícil realizar diseños de pruebas de una forma eficaz. Por lo tanto, para poder llevar a cabo dicha actividad es recomendable seguir una estrategia de testing que cuente con una serie de acciones para conseguir buenos resultados. Además, sería deseable que tal estrategia esté integrada por un marco conceptual, especificaciones de procesos y métodos.

Por otro lado, una práctica recomendada en el área de testing es la automatización de las pruebas dado que facilitan la realización de pruebas de regresión, evitando los costos de ejecutarlas de forma manual ante las actualizaciones del software.

Particularmente, en este proyecto se utiliza una estrategia de pruebas basada en escenarios y consciente de la situación sobre una aplicación web de una inmobiliaria con el fin de diseñar casos de pruebas para las distintas situaciones y, finalmente, automatizarlas por medio de un framework de pruebas llamado Cypress.

Palabras clave: Pruebas, Estrategia, Framework de prueba, Aplicación web.

Abstract: software testing is an important stage during the software development cycle since it enables to identify defects and achieve stability in the product. Nowadays, applications have become very complex because they are related to other applications or services. Consequently, it is difficult to produce testing designs in an effective way. Hence, to carry out this activity it is advisable to follow a testing strategy that has a series of actions to achieve good results. Besides, it would be of paramount importance that such a strategy could be integrated by a conceptual framework, specifications of process and methods.

On the other hand, a recommended practice in the testing area is the tests automation which facilitates the realization of regression testing, avoiding the costs of running them manually in the event of software updates.

Particularly, this project applies a scenario-based and situation-aware test strategy on a real estate agency web application in order to design test cases for different situations and, finally, automate them through a testing framework called Cypress.

Key Words: Testing, Strategy, Testing framework, Web application.

Proyecto Final

Diseño y automatización de pruebas de sistema utilizando una estrategia de pruebas de software basada en escenarios y consciente de la situación

4 de noviembre de 2022

Carrera: Ingeniería en Sistemas

Plan: 2017

Legajo: 5065

Estudiante: Fernandez Reale, Leonel Cristian Andres

Director: Becker, Pablo

Co-Director: Olsina, Luis

Asistente: Tebes, Guido

Agradecimientos

En primer lugar, quiero agradecer a mis viejos, mis hermanos, mi novia y el resto de mi familia que estuvieron presentes brindándome su apoyo incondicional en todo momento. Gracias por ser mis pilares y haber estado en aquellos momentos que tanto los necesité.

A la Universidad Nacional de La Pampa que me brindó los recursos disponibles para poder realizar la carrera. También, agradecer a todos los profesores que ofrecieron sus conocimientos y tiempo. En especial al Dr. Pablo Becker, Dr. Luis Olsina y el Ing. Guido Tebes quienes me entregaron recursos y experiencia para poder concluir con el proyecto final y a la Dra. Belen Rivera, mi tutora de la práctica profesional supervisada.

Quiero agradecer a mis compañeros de la Agencia de Investigación Científica, quienes fueron mi primer acercamiento laboral, me ofrecieron el tiempo y espacio para poder avanzar en el proyecto y me enseñaron gran parte de mi experiencia laboral.

Muchas gracias a la inmobiliaria que me permitió hacer uso de su sistema web para poder aplicar los conceptos en un caso real.

Finalmente, agradecer a mis amigos y compañeros de la facultad que hicieron que todo sea más fácil. Tantos proyectos, parciales y momentos compartidos con unos buenos mates para olvidarnos de las responsabilidades por un rato.

¡A todos ustedes, les estaré eternamente agradecido!

Resumen

La prueba de software o testing de software es una etapa importante en el ciclo de vida del software ya que permite identificar defectos y alcanzar una estabilidad en el producto. En la actualidad las aplicaciones se han vuelto muy complejas debido a que no funcionan de manera aislada sino que se relacionan con otras aplicaciones o servicios. En consecuencia, es difícil realizar diseños de pruebas de una forma eficaz. Por lo tanto, para poder llevar a cabo dicha actividad es recomendable seguir una estrategia de testing que cuente con una serie de acciones establecidas que ayuden a tomar decisiones y conseguir los mejores resultados posibles. Además, sería deseable que tal estrategia esté integrada por un marco conceptual, especificaciones de procesos y métodos. Si cumple con las tres capacidades mencionadas entonces será posible mantener un vocabulario común acorde al dominio, especificar cuáles son las tareas a realizar y cómo se llevarán a cabo a través de métodos.

Por otro lado, en el área de testing, una práctica recomendada es la automatización de las pruebas dado que facilitan la realización de pruebas de regresión, evitando los costos de ejecutarlas de forma manual una y otra vez ante las actualizaciones del software.

Particularmente, en este proyecto se utiliza una estrategia de pruebas basada en escenarios y consciente de la situación sobre una aplicación web de una inmobiliaria con el fin de diseñar casos de pruebas para las distintas situaciones y, finalmente, automatizarlas por medio de un framework de pruebas llamado Cypress.

Índice

1. Introducción

Son varias las razones por las que una aplicación web, y el software en general, puede contener errores o no conformidades. Algunos ejemplos son: una mala interpretación de los requisitos, un diseño incorrecto, no seguir buenas prácticas de programación, la falta de experiencia con el lenguaje de programación utilizado, entre otras. Si los errores son encontrados en etapas finales, probablemente ocurran desvíos en el plan del proyecto de desarrollo provocando gastos imprevistos, pérdida de confianza en el producto por parte del usuario, pérdida de clientes o patrocinadores, entre otras consecuencias no deseables. Por ende, es importante producir artefactos de calidad y para poder lograr dicho objetivo surge el Aseguramiento de la Calidad de Software (SQA, por sus siglas en inglés de Software Quality Assurance).

El SQA está compuesto por un conjunto de procesos, métodos, herramientas y estrategias que permiten gestionar la calidad de un producto de software durante su desarrollo. El área incluye un rango amplio de actividades y prácticas, por ejemplo [1]:

- Uso de estándares: asegura que los estándares adoptados se sigan correctamente y los productos de trabajos se apeguen a estos.
- Revisiones y auditorías: las revisiones técnicas son actividades realizadas por los ingenieros para controlar la calidad y su objetivo es detectar errores. Las auditorías son un tipo de revisión que tiene como objetivo garantizar que se sigan los lineamientos de calidad en el trabajo de la Ingeniería de Software.
- Pruebas de software: asegura que las pruebas se planean de una forma apropiada y eficaz, con la finalidad de que la probabilidad de detectar errores sea máxima.
- Colección y análisis de los errores: en esta actividad se reúnen y analizan errores y datos sobre los defectos para poder entender cómo se producen y de qué forma solucionarlos.

- ❖ Gestión de la seguridad: el SQA debe garantizar la seguridad del software, utilizando procesos y tecnologías apropiadas. Además, se debe evaluar el efecto de las fallas del software y realizar las acciones necesarias para disminuir el riesgo.
- ❖ Gestión de riesgos: asegura que las actividades de gestión de riesgos se realicen de forma apropiada y que se establezcan planes de contingencia.

Como se puede apreciar en la lista anterior, las pruebas de software (de ahora en adelante, testing) [2],[3],[4] son parte del SQA y es una de las etapas más importantes en el ciclo de vida del software debido a que ayuda a contribuir en la detección de errores y, por lo tanto, en la mejora de la calidad de los productos de software. El testing es un elemento de un tema más amplio, conocido como verificación y validación (V & V) y abarca muchas actividades de aseguramiento de la calidad del software. La verificación es el conjunto de tareas que garantizan que el software realiza correctamente una función específica. Por otro lado, la validación es un conjunto de tareas que aseguran que el software que se construye sigue los requerimientos del cliente.

1.1 Problema

Como se mencionó anteriormente, una de las actividades a realizar con el fin de asegurar la calidad de un producto son las pruebas. Al momento de diseñarlas surgen preguntas como: ¿qué pasos realizar? ¿cómo realizar pruebas eficaces? ¿cómo hacer para que el equipo de testing trabaje de la misma forma? Por otro lado, si las pruebas no son automatizadas, se ven obligados a realizar un trabajo manual para ejecutar cada caso de prueba. En consecuencia, provoca la pérdida de tiempo y recursos humanos que podrían estar abocados a otras tareas.

1.2 Motivación

Toda organización que desee asegurar la calidad de sus productos software/web y alcanzar sus metas de negocio debería seguir una estrategia o curso de acción bien definido que indique claramente el camino a seguir para

poder cumplir los objetivos planteados. Una estrategia bien definida debería estar integrada por un marco conceptual, especificaciones de procesos y métodos [5]. Si la estrategia cumple con estas tres capacidades, tendrá como ventaja mantener un vocabulario común acorde al dominio, especificar cuáles son las tareas a realizar y cómo se llevarán a cabo a través de métodos.

Teniendo en cuenta lo anterior, en este trabajo final se pretende aplicar una estrategia de testing llamada Situation-aware Scenario-based Testing Strategy (SaST Strategy) [6], la cual integra las tres capacidades ya mencionadas, a una aplicación web, en desarrollo, de una inmobiliaria de la localidad de Santa Rosa. Esta estrategia da soporte al diseño e implementación de casos de prueba basados en escenarios y conscientes de la situación. Notar que las pruebas basadas en escenarios son un tipo de técnica de caja negra que utiliza como insumo especificaciones de escenarios o situaciones para el diseño de las pruebas [3].

Finalmente, es importante destacar que la automatización de las pruebas es un aspecto fundamental en el proceso de desarrollo de software. Contar con la automatización de las pruebas facilita la realización de pruebas de regresión, evitando los costos de ejecutar las mismas pruebas de forma manual una y otra vez ante las actualizaciones del software. Por lo tanto, además de diseñar pruebas de sistema utilizando SaST, en este trabajo se automatizarán las misma con Cypress [7].

1.3 Objetivos

Los objetivos generales son:

- Probar que las funcionalidades de la app estén libres de errores haciendo uso de una estrategia basada en escenarios y consciente de la situación para diseñar pruebas de sistema.
- Automatizar las pruebas para facilitar las mismas luego de realizar mejoras a la aplicación.

Respecto al primer objetivo general, tenemos los siguiente objetivos específicos:

- ❖ Estudiar y comprender las tres capacidades de la estrategia SaST.
- ❖ Diseñar y documentar un conjunto de casos de pruebas a nivel de sistema usando la estrategia SaST.

Por último, los objetivos específicos para el segundo objetivo general son:

- ❖ Estudiar y comprender Cypress, herramienta que da soporte a la automatización de pruebas.
- ❖ Usar Cypress para automatizar las pruebas diseñadas y documentadas con SaST.

1.4 Estructura de la tesis

El proyecto final está estructurado en seis capítulos. El Capítulo 1 explica el problema a resolver, la motivación y los objetivos generales juntos a sus correspondientes objetivos específicos.

Luego, en el contenido del Capítulo 2 se pueden encontrar los tipos y métodos de pruebas, el marco conceptual, el método y la especificación del proceso de la estrategia SaST y define el framework de Cypress junto a sus funcionalidades y diferencias con las demás herramientas.

El Capítulo 3 explica los pasos que se realizaron para la aplicación de la estrategia SaST y define los artefactos generados por las actividades. También, describe la forma en que se automatizaron las pruebas.

Respecto al Capítulo 4, se pueden identificar cuatro secciones principales. Las primeras tres explican la aplicación de la estrategia SaST a las funcionalidades "iniciar sesión", "nueva publicación" y "contactar inmobiliaria por correo electrónico". También se comenta en cada caso la automatización de las pruebas. Es importante mencionar que la estrategia también se aplicó para probar la funcionalidad "modificar publicación", sin embargo los artefactos generados para este caso se decidió incluirlos en un anexo (ver Anexo E), ya que los mismos son similares a los de otras funcionalidades presentados en este capítulo. En la última sección se analizan los resultados de las pruebas realizadas.

El Capítulo 5 contiene información sobre trabajos relacionados. La finalidad del mismo es explicarlos de forma resumida y, luego, compararlos con la estrategia SaST.

En cuanto al Capítulo 6, este comenta las conclusiones y resultados, las apreciaciones personales y los trabajos pendientes a realizar en el futuro.

2. Marco teórico de la estrategia SaST y Cypress

Una organización debe utilizar una estrategia para poder asegurar la calidad de sus productos software/web, alcanzar las metas de negocio y lograr los objetivos planteados. En este trabajo se hace uso de SaST con el objetivo de probar la aplicación "Inmobiliaria X" desde un enfoque de pruebas de sistema para verificar su correcto funcionamiento de acuerdo a los requisitos funcionales.
SaST es una estrategia bien definida ya que se encuentra integrada por un marco conceptual (la ontología TestTDO), especificaciones de proceso (denominado SaSTPro) y método (denominado SaSTMe).

Por otro lado, como se ha mencionado anteriormente, automatizar las pruebas es muy importante. Cypress es un framework que ofrece herramientas para automatizar pruebas de end-to-end.

A continuación se explicarán brevemente los tipos de pruebas que se pueden encontrar en el área de software. Luego, se comentará la estrategia SaST teniendo en cuenta sus tres pilares (marco conceptual, proceso y método). Por último, se presenta información sobre Cypress, la herramienta escogida para automatizar los casos de prueba diseñados con la estrategia SaST.

2.1. Tipos de pruebas

En el área de testing existe una variedad de tipos de pruebas que se las puede clasificar principalmente en pruebas funcionales y no funcionales. Desde el punto de vista funcional se pueden encontrar:

- Pruebas unitarias: se realizan a una unidad o componente individual para probar su correctitud. Normalmente, es desarrollado por el programador en la fase de desarrollo. Se considera unidad a un método, función, procedimiento o objeto.

- Pruebas de integración: se aplica a dos o más módulos que se agrupan y son probadas como un todo. Su propósito es detectar fallas en la comunicación entre ellos.
- Prueba de sistema: verifica la correctitud del sistema contra los requerimientos especificados. A diferencia de los anteriores, en este caso se utiliza el sistema entero, es decir, todas las distintas partes de la aplicación interactuando entre sí como en el mundo real en un entorno de pruebas.
- Prueba de aceptación: es un tipo de prueba que sucede en la última etapa antes de enviar el sistema a producción. El cliente valida que las características y funcionalidades definidas funcionan correctamente y como esperaba.

Además de tener estos tipos de pruebas funcionales también se pueden encontrar métodos de pruebas que proveen eficacia al momento de realizarla. Dos tipos de métodos ampliamente conocidos en el área de las pruebas son:

- Caja blanca: el código interno de la aplicación de software, diseño y estructura son examinados para verificar el flujo de datos de entrada a salida. Normalmente, es utilizado en etapas de prueba unitaria pero también puede ser utilizada para pruebas de integración o sistemas.
- Caja negra: este método se enfoca en las funcionalidades de la aplicación sin tener conocimiento del diseño o estructura interna de la aplicación de software. Usualmente, es utilizado para las pruebas de integración y sistema, aunque puede ser utilizada para pruebas unitarias.

Respecto a las pruebas no funcionales, está integrada por:

- Pruebas de seguridad: la finalidad es verificar la seguridad del software ante amenazas internas o externas.
- Pruebas de rendimiento: este tipo de prueba se encarga de analizar los tiempos de respuesta y la estabilidad ante la presencia de mucha carga.
- Pruebas de usabilidad: consisten en probar una aplicación desde la perspectiva del usuario para verificar la apariencia y la facilidad de uso.

- ❖ Pruebas de compatibilidad: es un tipo de prueba que permite validar cómo se comporta el software en diferentes ambientes como servidores web, hardware y entornos de red.

Particularmente, las pruebas a realizar considerando la estrategia SaST se enmarcan dentro de las pruebas de sistemas, un tipo de prueba funcional, haciendo uso de la metodología de caja negra. A continuación se describen detalles de la estrategia SaST.

2.2. Marco conceptual de la estrategia SaST

La estrategia SaST está semánticamente soportada por un conjunto de ontologías que pertenecen a una arquitectura ontológica denominada FCD-OntoArch [8]. Como se ilustra en la Fig. 1, esta se encuentra formada por las capas *foundational, core, domain e instance*. A su vez, la capa domain está dividida en las subcapas top-domain y low-domain. Las ontologías se pueden relacionar con las del mismo nivel, a excepción de ThingFO [9] dado que en el nivel foundational se encuentra sola. Además, las ontologías de niveles inferiores pueden ser enriquecidas semánticamente por otras de una capa superior.

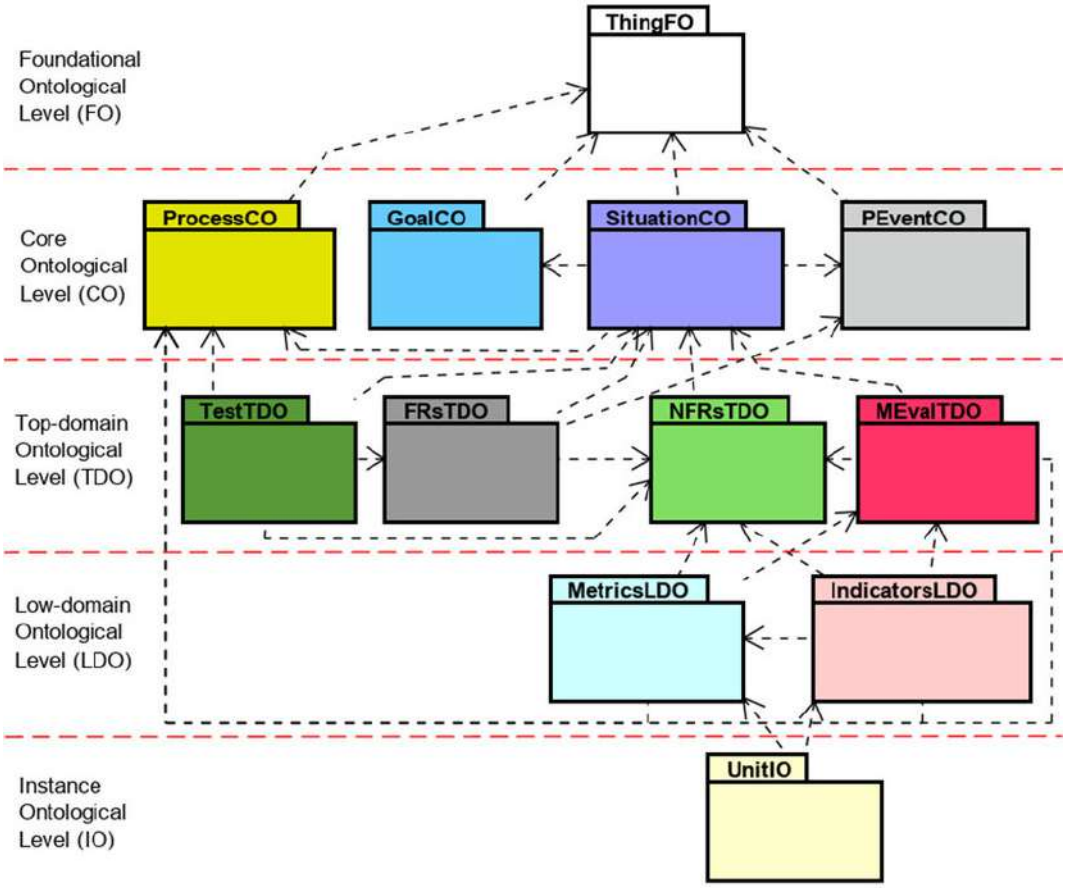


Fig.1. Arquitectura ontológica FCD-OntoArch.

La estrategia SaST se encuentra enriquecida semánticamente por la ontología de pruebas de software denominada TestTDO [10], perteneciente a la capa top-domain. Esta provee los términos a la estrategia para las especificaciones de procesos y métodos.

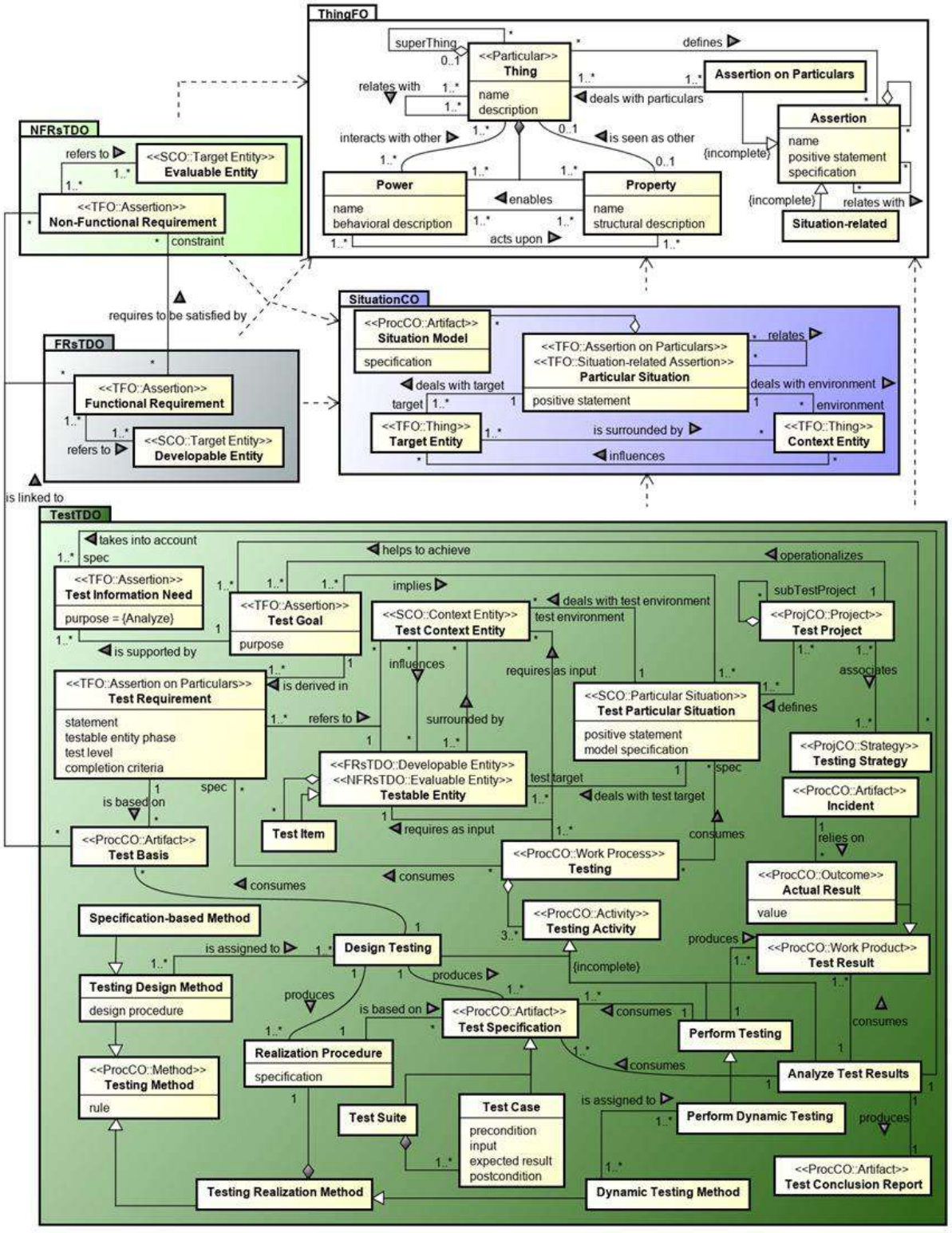


Fig. 2. Fragmento de algunas ontologías de FCD-OntoArch. Tener en cuenta que se usa TFO para representar Thing Foundational Ontology (ThingFO), SCO para Situation Core Ontology (SituationCO), ProcCO para Process Core Ontology, ProjCO para Project Core Ontology, NFRsTDO para Non-Functional Requirements Top-Domain Ontology y FRsTDO para Functional Requirements Top-Domain Ontology.

Respecto a TestTDO, la Fig. 2 muestra un fragmento de la misma. Las definiciones de los términos, propiedades, relaciones no taxonómicas y, además, las especificaciones de axiomas están documentadas en https://arxiv.org/abs/2104.09232. La ontología mencionada enriquece a la estrategia SaST con los siguientes términos. Un Proceso de Prueba (Testing Process) está compuesta al menos por tres Actividades de Prueba (Testing Activities): Diseñar Prueba (Design Testing), Ejecutar la Prueba (Perform Testing) y Analizar los Resultados de prueba (Analyze Test Results). Se pueden agregar otras actividades al proceso, por ejemplo configurar el entorno de prueba.

La actividad Diseñar la Prueba tiene la finalidad de producir Especificaciones de la Prueba (Test Specifications) y los Procedimientos de Realización (Realization Procedures). Las Especificaciones de Prueba pueden ser diseñadas mediante el uso de Bases de Prueba (Test Basis), Requisitos de la Prueba (Test Requirements) y Situaciones Particulares de Prueba (Test Particular Situations).

Las Bases de Prueba pueden estar compuestas por especificaciones de requisitos, código fuente, etc. Los Requisitos de Prueba son enriquecidos por el término aserciones sobre particulares (Assertion on Particulars) de la ontología ThingFO (ver enlace para más detalles sobre la ontología https://arxiv.org/abs/2107.09129) y establecen lo que se debe validar o verificar de un Objeto de Prueba (Test Object). Por último, la Situación Particular de la Prueba es enriquecida por Situación Particular (Particular Situation) de la ontología SituationCO (ver enlace para más detalle sobre la ontología http://arxiv.org/abs/2107.10083) y asocia la Entidad Verificable (Testable Entity) con algunas Entidades de Contexto de la Prueba (test environment, si corresponde). Puede ser especificada mediante escenarios de casos de usos, diagramas de comunicaciones u otro tipo de representación que resulte de utilidad para el diseño de la Especificación de la Prueba. Además, dependiendo de la Situación Particular de Prueba, una Entidad Verificable puede tener semántica de Entidad Desarrollable (Developable Entity), si se realizan pruebas funcionales, o Entidad Evaluable (Evaluable Entity), si se realizan pruebas no funcionales.

Una actividad Diseño de Prueba tiene asignado un Método de Diseño de Prueba (Testing Design Method). Un Método basado en Especificaciones (Specification-based Method), también conocido como método de caja negra, es un tipo de Método de Diseño de Prueba y un tipo más específico es un Método de Prueba basado en Escenarios (Scenario-based Testing Method). Tener en cuenta que un Método de Diseño de Pruebas tiene un procedimiento de diseño que tiene la finalidad de ordenar un conjunto de instrucciones u operaciones, que especifican cómo se debe realizar una subactividad o tarea de la actividad Diseñar las Pruebas y, además, tiene reglas de diseño.

Un tipo de Especificación de Prueba es un Caso de Prueba (Test Case). Este artefacto se utiliza en la actividad Realizar Pruebas Dinámicas y está compuesto por precondiciones (preconditions), entradas (inputs), resultados esperados (expected result) y postcondiciones (postconditions). Una precondición es una restricción que debe evaluarse como verdadera antes de que la entrada del Caso de Prueba se use en la actividad Ejecutar las Pruebas. En cambio, una postcondición es una restricción que debe evaluarse como verdadera después de que se utilizó la entrada del caso de prueba y se obtuvo el resultado real en una actividad de realización de pruebas.

2.3. Método de la estrategia SaST

Esta sección está destinada a describir el método que utiliza la estrategia SaST llamado Situation-aware Scenario-based Testing Method (SaSTMe, Método de Prueba basado en Escenarios y consciente de la Situación), debido a que considera la especificación del modelo de situación. SaSTMe ofrece soporte al diseño y documentación de los casos de prueba, ya que permite obtener información importante, por ejemplo identificar las entidades objetivos y de contexto. Es importante aclarar que el método está basado en especificaciones y solo sirve para realizar pruebas dinámicas. Para ilustrar y describir el soporte semántico en la estrategia se utilizaran términos de ontologías mencionadas en el marco conceptual con palabras capitalizadas, las propiedades con cursiva y las relaciones no taxonómicas con subrayado.

En la Tabla 1 puede visualizarse la plantilla del método SaSTMe. Para comenzar, hay que tener en claro que la plantilla tiene como objetivo capturar información importante de la Situación Particular de Prueba, por lo que los diseñadores de prueba deben documentar su *declaración positiva* y su *especificación de modelo* (Situation Model). Ambos pueden especificarse en lenguaje natural pero la situación de modelo es más valiosa si se utiliza un diagrama estandarizado, por ejemplo, diagramas de secuencia o diagramas de comunicación de UML, entre otros. Cabe destacar que la Situación Particular de Prueba puede ser diseñada por los diseñadores de prueba o previamente elaborada, en lenguaje natural o especificada formalmente, en alguna fase del ciclo de desarrollo del software por otras partes interesadas.

Tabla 1. Plantilla del Método de Prueba Basado en Escenarios y Consciente de la Situación (SaSTMe).

Test Particular Situation: *-positive statement*: -(Particular) **Situation Model** *specification*: *-Conditions: (not mandatory)* *-preconditions:* *-postconditions:*
Testable Entities (and **Test Items**, if applicable)**:** *indicate what are the test targets*
Test Context Entities: (not mandatory) If exists Test Context Entities: **-Influences:** describe how the test context entities influence the test targets.
Test Requirement:
Test Basis:
Test Cases: *-input*: *-expected result*: *-preconditions*: *-postconditions*:

Luego de que la Situación Particular de Prueba fue definida, el siguiente paso es analizarla con la finalidad de identificar las Entidades Verificables, los Elementos de prueba (Test Items) si corresponde y, por último en caso de existir, las Entidades de Contexto de Prueba (Test Context Entities) que las influyen (influences). Además,

la plantilla permite capturar los Requisitos de Prueba y las información de Bases de Prueba que puede ser utilizada para especificar las Situaciones Particulares de Prueba y diseñar los *resultados esperados* en los Casos de Prueba. Un ejemplo de información de Base de prueba puede ser la especificación de los requisitos funcionales o no funcionales.

Por último, utilizando toda la información incluida en la plantilla los encargados de diseñar las pruebas elaboran los Casos de Prueba. Es importante señalar que las Situaciones Particulares de Prueba y las Entidades de Contexto de Prueba asociadas pueden influir en las *entradas*, las *condiciones* o ambas cosas de los Casos de Prueba. Por lo tanto, los diseñadores de pruebas deben analizar los Modelos de Situaciones Particulares para diseñar las *entradas* y/o *condiciones* de los Casos de Prueba, que luego serán útiles para verificar o validar efectivamente la Entidad Verificable con las Entidades de Contexto de Prueba circundantes (surrounded by).

2.4. Especificación del proceso de la estrategia SaST

A continuación, se describe el proceso denominado Situation-aware Scenario-based Testing Process (Proceso de Prueba basado en Escenarios y Consciente de la Situación, SaSTPro), modelado en la Fig. 3 haciendo uso del lenguaje SPEM (Sofware & Systems Process Engineering Metamodel). La especificación considera dos de las cuatro perspectivas de modelado propuesta por Curtis [11]. Una de las perspectivas utilizadas es la *funcional* que tiene la finalidad de describir qué actividades deben llevarse a cabo y qué flujo de artefactos es necesario para poder ejecutar las actividades y tareas. La otra perspectiva utilizada es la *de comportamiento* que tiene el objetivo de especificar cuándo se deben ejecutar las actividades, es decir, muestra las secuencias, paralelismos y/o iteraciones. Es importante remarcar el uso de las perspectivas de modelado de procesos dado que permite fortalecer las especificaciones de procesos facilitando su comprensión y comunicación, en consecuencia, promueve la consistencia y repetibilidad.

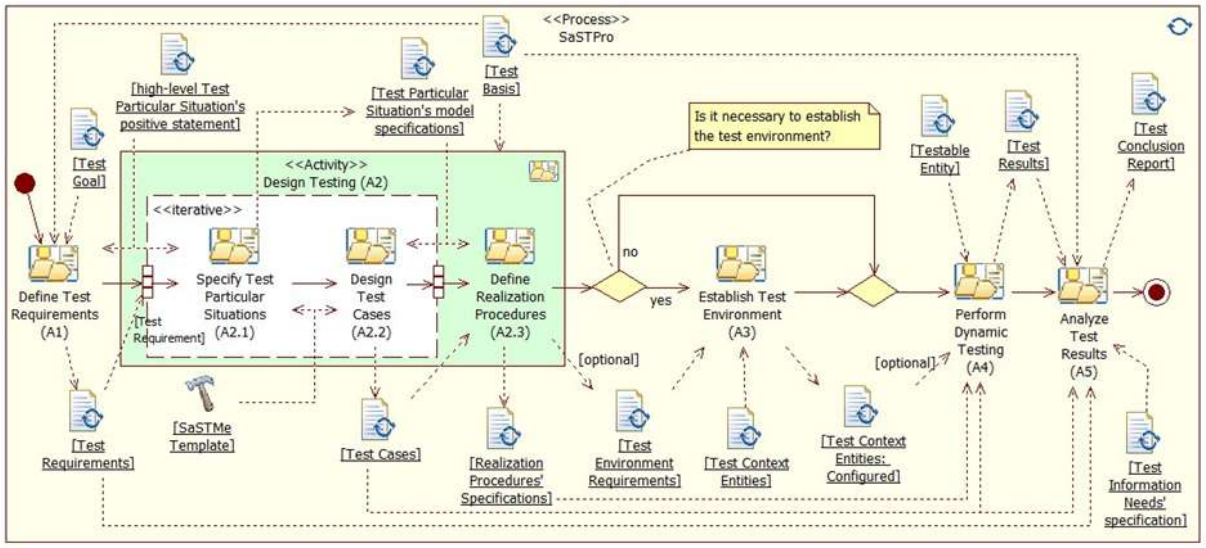


Fig. 3. Especificación de Situation-aware Scenario-based Testing Process (SaSTPro) con perspectiva funcional y de comportamiento.

En esta sección se describirán las actividades principales de SaSTPro y los artefactos que éstas consumen y producen. La semántica de los términos de procesos se encuentran en la ontología Process Core Ontology (ProcessCO).

Respecto al proceso de la Fig. 3, la primera actividad a llevar a cabo es "Definir los Requisitos de Prueba (Define Test Requirements, A1)". Esta actividad consume los siguientes documentos: Objetivo de la Prueba (Test Goal) y *declaración positiva* de la Situación Particular de Prueba (Test Particular Situation's *positive statement*), tanto a nivel de de Proyecto de Prueba (Test Project) como de Base de Prueba. El objetivo de A1 es producir los Requisitos de Prueba que establece que debe ser verificado y/o validado de un Objeto de Prueba (Test Object) o Entidad Verificable.

La siguiente actividad es Diseñar la Prueba (Design Testing, A2) y está compuesta por tres subactividades. La primera es Especificar la Situación Particular de Prueba (Specify Test Particular Situations, A2.1), donde los diseñadores de las pruebas elaboran los Modelos de Situación, por ejemplo el modelo de escenario de caso de uso o un diagrama de comunicación utilizando las Bases de Prueba. Los Modelos de Situación deben estar alineados con los Requisitos de Prueba y la *declaración positiva* de la Situación Particular de la Prueba. Además, asocian la Entidad Verificable y, en caso de existir, las Entidades de Contexto (Context Entities),

que deben identificarse y registrarse. Finalmente, toda la información producida se registra de acuerdo con la especificación de la plantilla SaSTMe comentada en la Sección 2.2.

Una vez finalizado A2.1, comienza la actividad Diseñar los Casos de Pruebas (Design Test Cases, A2.2). Los Casos de Pruebas son diseñados partiendo de la Situación Particular de Prueba y las Bases de Prueba. Un Caso de Prueba está compuesto por las *entradas (inputs)*, precondiciones (preconditions), *postcondiciones (postconditions) y el resultado esperado (expected result)*. Para cada Caso de Prueba, estos elementos serán diseñados analizando los Modelos de las Situaciones y serán registrados utilizando la plantilla SaSTMe. Es importante aclarar que, tal como se muestra en la Fig. 3, las actividades A2.1 y A2.2 se realizan de forma iterativa para cada Requisito de Prueba.

Luego, se debe realizar la subactividad Definir los Procedimientos de Realización (Define Realization Procedures, A2.3). Una vez en esta actividad, la plantilla ya debe estar completa debido a que se utiliza para producir las Especificaciones de Procedimientos de Realización (Realization Procedures' *specifications*). Por ejemplo, cuando se llevan a cabo Pruebas Dinámicas (Dynamic Testing) que implican la ejecución de los Casos de Prueba, la especificación de un Procedimiento de Realización (Realization Procedure' specification) puede ser una secuencia de Casos de Pruebas que se ejecutan en un cierto orden y cualquier acción asociada que pueda ser necesaria para establecer las condiciones previas y posteriores.

Una vez finalizada la tarea A2.3 continúa la actividad opcional Establecer Entorno de Prueba (Establish Test Environment, A3). Esta actividad utiliza los Requisitos del Entorno de Prueba (Test Environment Requirements) obtenidos de A2.3 teniendo en cuenta la plantilla SaSTMe debido a que esta contiene información sobre las Entidades de Contexto de Prueba como hardware, instrumentación, simuladores, herramientas de software y otros elementos de apoyo necesarios para realizar una prueba. Es importante aclarar que A3 es una actividad opcional.

La siguiente actividad es Realizar Pruebas Dinámicas (Perform Dynamic Testing, A4). En este momento los Procedimientos de Realización son ejecutados utilizando la Entidad Verificable y, en caso de existir, las Entidades de Contexto de Prueba. Las especificaciones de los Casos de Pruebas también pueden ser utilizados, por ejemplo en una prueba manual. Al finalizar la tarea, A4 produce los Resultados de Prueba (Test Results), los cuales pueden ser Resultados reales (Actual Result) y/o Incidentes (Incidents).

Por último, se lleva a cabo la actividad Analizar Resultados de Prueba (Analyze Test Results, A5). En A5, los Resultados de la Prueba se analizan con el fin de determinar si se cumplen los Requisitos de la Prueba y si se alcanzan las Metas de Necesidad de Información de la Prueba (Test Information Need Goals). Este tipo de objetivos provee información de utilidad para poder decidir si se logró el Objetivo de la Prueba y luego especificar el análisis en el Informe de Conclusión de la Prueba (Test Conclusion Report).

Para cerrar, es importante señalar que las actividades de SaSTPro se muestran de manera secuencial para mantener su legibilidad, salvo las subactividades A2.1 y A2.2 que iteran por cada Requisito de Prueba. Al momento de aplicar la estrategia pueden tomarse otras tareas como iterativas, por ejemplo, si algún requisito de prueba no se cumple debido a que hay que tener en cuenta más Situaciones Particulares de Prueba, las actividades A2-A5 también pueden realizarse de forma iterativa.

2.5. Cypress

2.5.1 ¿Qué es Cypress?

Cypress [7] es un framework de pruebas que consta de una aplicación gratuita de código abierto y un servicio de tablero para registrar pruebas. El lenguaje que utiliza es javascript y permite realizar pruebas end-to-end, integración y unitarias a cualquier cosa que se ejecute en un navegador.

Cypress es utilizado por: Paypal, Walt Dysney Studios, DHL, HashiCorp, AirTable y otras grandes empresas.

2.5.2. Funcionalidades

La herramienta tiene funcionalidades importantes que otras no ofrecen. Por ejemplo:

- Viaje en el tiempo: toma capturas de pantallas durante la ejecución de las pruebas.
- Depuración: permite leer errores, seguir una pila de ejecución y usar herramientas de desarrolladores. De esa forma evita adivinar la causa de un determinado error.
- Espera automática: provee un mecanismo de espera automática con el fin de evitar el uso de tiempos de esperas.
- Spies, stubs y clocks: facilita la verificación de funciones, respuestas de servidores y temporizadores.
- Control de tráfico de red: permite controlar de una forma fácil la red creando stubs y probar una determinada funcionalidad sin involucrar servidores.
- Resultados consistentes: la arquitectura de Cypress no utiliza Selenium y WebDriver. Selenium es un framework de pruebas que utiliza WebDriver, es decir, una API que se encarga de realizar peticiones HTTP para simular las interacciones de usuarios en un navegador. Por lo tanto, Cypress provee herramientas de pruebas rápidas, consistentes y confiables.
- Capturas y videos: la herramienta permite ver capturas de pantalla en el momento en que falla una prueba o realiza videos de toda la ejecución de las pruebas.
- Prueba entre navegadores: permite ejecutar prueba localmente dentro de la familia de navegadores de firefox y chrome.

2.5.3. Diferencias con otros frameworks

- No utiliza Selenium: la mayoría de los framework para realizar pruebas de end-to-end utilizan Selenium. Cypress utiliza una arquitectura distinta, es decir, se ejecuta en el mismo ciclo de ejecución que la aplicación. Por otro lado, Selenium, ejecuta comandos remotos a través de la red.

- Solo se enfoca en realizar end-to-end: no es una herramienta que se enfoca en realizar pruebas en distintas partes de la aplicación, por ejemplo back-end. Cypress sólo hace foco en realizar pruebas de end-to-end por lo que intenta proveer los medios para facilitar las pruebas.
- Funciona en cualquier framework front-end y sitio web: la arquitectura de Cypress es compatible con cualquier framework front-end, por ejemplo Angular, React, Vue, etc. El framework tiene la capacidad de probar cualquier cosa que se ejecute en un navegador.
- Las pruebas son escritas solamente en Javascript: si bien puede compilar a JavaScript desde cualquier otro idioma, en última instancia, el código de prueba se ejecuta dentro del navegador. No hay enlaces de lenguaje o de controlador, solo hay JavaScript.
- Cypress es todo en uno: realizar una prueba de end-to-end necesita de muchas herramientas. Cypress provee todas las herramientas sin necesidad de instalar algo por separado.
- Cypress es para desarrolladores e ingenieros de control de calidad: uno de los objetivos es hacer realidad el desarrollo basado en pruebas para las pruebas de end-to-end. Cypress se ha diseñado para que las pruebas y el desarrollo puedan realizarse simultáneamente. Puede desarrollar rápido mientras conduce todo el proceso de desarrollo con pruebas porque la aplicación aún tiene acceso a las herramientas de desarrollo y los cambios se reflejan en tiempo real.
- Ejecución rápida: ofrece servicio de tablero, paralelización y equilibrio de carga automatizado que potencian las velocidades de prueba.

3. Metodología

En este capítulo se describe cómo se resolvieron las etapas del proyecto final. En primer lugar, se realizó una lectura y estudio de los artículos "Specifying and Analyzing a Software Testing Ontology at the Top-Domain Ontological Level" [12] y "A Situation-aware Scenario-based Testing Strategy" [6]. En resumen, la primera lectura especifica y analiza la ontología denominada TestTDO, ver Fig. 2. Esta surge posterior a un análisis de un grupo de ontologías relacionadas a las pruebas. El estudio de la ontología TestTDO permitió comprender la terminología del dominio de Testing. En cuanto al segundo documento, este describe la estrategia SaST, el método que utiliza y cómo es su proceso para llevarla a cabo. Estudiar estos aspectos de la estrategia SaST fue fundamental para poder realizar las pruebas de la aplicación web de una inmobiliaria.

Una vez comprendidas la ontología y la estrategia, se seleccionaron las funcionalidades a probar. Seleccionar las funcionalidades permitió realizar un relevamiento y obtener información pertinente para las bases de pruebas. Por ejemplo, en la Sección 4.1.1 se pueden identificar como bases de pruebas el diagrama de casos de usos, la historia de usuario y el wireframe correspondiente a la funcionalidad "iniciar sesión" (ver Fig. 4, Tabla 2, Fig. 5, respectivamente).

Es importante mencionar que si bien el proceso de la Fig. 3 presenta sus actividades principales de manera secuencial, en este trabajo se realizaron las actividades A2, A3 y A4 de manera iterativa por cada funcionalidad con la finalidad de facilitar su documentación. Luego, la actividad A5 fue realizada una única vez considerando los resultados de todas las iteraciones con el objetivo de realizar un único reporte de conclusión de la prueba.

Además, se tomó la decisión de explicar la estrategia desde un caso simple a un caso complejo. Por ejemplo, en la funcionalidad "iniciar sesión" se tomó como ejemplo el caso más simple. Luego, en "nueva publicación" se seleccionó un escenario un poco más complejo, donde participan las validaciones de los

formularios. Por último, en "contactar inmobiliaria por correo electrónico" se eligió una situación de mayor complejidad, donde participa el servidor de correos electrónicos como entidad de contexto de prueba.

Una vez obtenida la información para las bases de pruebas se elaboraron los requisitos de prueba. Por ejemplo, en la Sección 4.1.2 se puede ver cómo se especifica el requisito de prueba para la funcionalidad "iniciar sesión" (ver Tabla 3). Cada especificación de requisito de prueba está formado por:

- Label (Etiqueta): permite identificar el requisito de prueba de forma única.
- Statement (Declaración): una declaración explícita del requisito de prueba que debe ser cumplida. Suele ser escrita en lenguaje natural o de alto nivel.
- Testable Entity Phase (Fase de la entidad verificable): permite identificar en qué etapa se encuentra la entidad a verificar. Por ejemplo: inicio, desarrollo, despliegue, operación y mantenimiento.
- Test Level (Nivel de prueba): representa el tipo de prueba a realizar teniendo en cuenta la declaración del requisito de prueba.
- Completion Criteria (Criterios de finalización): conjuntos de condiciones que permiten identificar si un requisito de prueba será considerado completo.
- Refers to Testable Entity (Entidad a verificar): permite identificar la entidad a verificar.
- Refers to Test Context Entities (Entidades de Contexto de Prueba): permite identificar las entidades de contexto de prueba.

Notar que todos los campos indicados anteriormente surgen de la ontología de TestTDO comentada en el Capítulo 2.

Posteriormente, se debe producir la situación particular de prueba con sus casos de prueba asociados. En la sección 4.1.3 se explica que a través de la subactividad A2.1 (ver Fig. 3) se especifican las situaciones particulares de prueba

para la funcionalidad "iniciar sesión" (ver Tablas 4, 5 y 6). La situación particular de prueba está compuesta por:

- Positive statement (Declaración positiva): una declaración explícita de una situación particular de prueba a definir, qué puede hacer referencia a una situación dinámica o estática.
- Particular situation model specification (Especificación de modelo de la situación particular de prueba): representa un artefacto que especifica y modela una situación particular de prueba en un lenguaje determinado. Respecto al lenguaje puede ser formal, semiformal e informal.
- Conditions (Condiciones): conjunto de restricciones, que en caso de existir, deben cumplirse antes y/o después de la ejecución de la situación particular de prueba.

Después de definir las situación particular de prueba se definen los casos de prueba, para ello se debe tener en cuenta que un caso de prueba está formado por:

- Conditions (Condiciones): conjunto de restricciones, que en caso de existir, deben cumplirse antes y/o después de ejecutar el caso de prueba.
- Inputs (Entradas): valores o acciones que se utilizan en la ejecución del caso de prueba.
- Expected result (Resultado esperado): resultado que se espera luego de ejecutar la prueba con una determinada entrada de valores o acciones.

Notar que especificar la situación particular de prueba y definir los casos de pruebas son tareas iterativas hasta cumplir con todas los requisitos a probar. En la Sección 4.1.3 se expresa como la subactividad A2.2 (ver Fig. 3) produce los casos de pruebas para la funcionalidad seleccionada (ver Tablas 7, 8 y 9).

Luego de diseñar los casos de pruebas, se avanza hacia la automatización de las pruebas utilizando Cypress. Para poder comenzar esta etapa se realizó la lectura de la documentación de la página oficial y otras que dan soporte a las dudas de

los programadores, y la visualización de videos tutoriales de Cypress. Luego, se utilizó una estructura de carpetas definida de la siguiente forma:

- Fixtures: contiene los conjuntos de datos de entrada que son consumidos para ejecutar los casos de prueba de las distintas situaciones particulares de pruebas.
- Integration/ "Nombre de funcionalidad" (por ejemplo: Integration/Login): cada carpeta con el nombre de su funcionalidad contiene las distintas situaciones particulares de prueba. Cada archivo dentro de cada situación contiene los códigos de automatización de los casos de pruebas correspondientes.
- Support: esta carpeta contiene las funciones que brindan soporte a las distintas funcionalidades a probar.

Cabe destacar que se utilizó el sistema control de versiones de GIT para trabajar en el proyecto. Al comenzar con la programación, se agregó una cierta cantidad de tareas que hacen referencia a las funcionalidades a probar. Luego, cada tarea tiene subtareas. Para las subtareas se utilizó una convención de nombres de ramas. Estas estaban definidas de la siguiente manera: "test/(letra inicial del proyecto)-(número de issue de la funcionalidad en el tablero de GitLab)-(número de issue de la subtarea)". Suponiendo una funcionalidad con el número de issue 1 y una subtarea correspondiente a dicha funcionalidad con el número de issue 2, entonces el nombre de la rama es "test/t-1-2". Por otro lado, para solucionar un error se utiliza la siguiente nomenclatura "bugfix/(letra inicial del proyecto)-(número de issue de la funcionalidad en el tablero de GitLab)-(número de issue del error)". Finalizando la Sección 4.1.3 se pueden identificar las especificaciones de los procedimientos de realización automatizadas con Cypress y los requisitos de entornos que son necesarios para la realización de las pruebas en la funcionalidad "iniciar sesión" (ver Fig. 6 y Tabla 10).

Posteriormente, se configuraron los requisitos de pruebas obtenidos. En la Sección 4.1.4, se puede visualizar en una captura de pantalla como fue

configurado el requisito de prueba de la funcionalidad “iniciar sesión” (ver Tabla 11).

Una vez realizados los pasos anteriores, se está en condición para ejecutar las pruebas. En la Sección 4.1.5 se documenta la ejecución de los procedimientos de realización, o en otras palabras, las pruebas automatizadas a través de la herramienta de Cypress (ver Fig. 10).

Finalmente, en la Sección 4.4 se realiza el análisis de los resultados de las pruebas para las funcionalidades seleccionadas. El objetivo de unificar esta actividad es obtener solo un reporte de conclusión de las pruebas.

4. Aplicación de la estrategia SaST y automatización de las pruebas

En este capítulo se hará uso de la estrategia SaST, y una vez diseñados los casos de pruebas, serán automatizados con el framework de Cypress. Se seguirá el proceso especificado en la Fig 3 teniendo en cuenta la secuencia mencionada en el capítulo anterior con el fin de facilitar su aplicación e identificar los artefactos de cada funcionalidad.

La entidad que se probó fue una aplicación web para una inmobiliaria de la ciudad de Santa Rosa (La Pampa). Es importante mencionar que la aplicación probada no es la versión que se encuentra actualmente en producción, sino una versión que al inicio de este trabajo estaba en desarrollo.

Para llevar a cabo la aplicación de la estrategia se eligieron cuatro funcionalidades de la aplicación web de la inmobiliaria para realizar la etapa de testing:

- Iniciar sesión
- Nueva publicación
- Modificar publicación
- Contactar a la inmobiliaria por correo electrónico

Tal como se muestra en la Fig. 3, la primera actividad es "definir los requisitos de prueba" (A1) y tiene como entrada el objetivo de la prueba (Test Goal), la declaración positiva de la situación particular de prueba de alto nivel (high-level Test Particular Situation's positive statement) y las bases de pruebas. El objetivo de la prueba es: "*probar la aplicación Inmobiliaria X desde un enfoque de pruebas de sistema para verificar su correcto funcionamiento de acuerdo a los requisitos funcionales*". Por otro lado, la declaración positiva de la situación particular de prueba de alto nivel es "*un usuario administrador tiene acceso a las funcionalidades de iniciar sesión, nueva publicación, y modificar publicación.*

Respecto a los usuarios clientes, pueden contactar a la inmobiliaria por medio de la funcionalidad de enviar un correo electrónico. En todos los casos se espera que las funcionalidades estén libres de error".

A continuación se describe la ejecución de las actividades A1-A4 para las funcionalidades "Iniciar sesión", "Nueva publicación" y "Contactar a la inmobiliaria por correo electrónico". Los artefactos generados para probar la funcionalidad "Modificar publicación" se encuentran en el Anexo E.

Cabe destacar que las bases de pruebas deben ser definidas previo a la actividad A1 pero se decidió crear una sección en cada funcionalidad con el objetivo de definirlas por separado y así ayudar al lector a identificar más fácilmente qué bases de prueba se corresponden con cada funcionalidad.

4.1. Iniciar sesión

4.1.1. Bases de pruebas para la funcionalidad iniciar sesión

Para definir las bases de pruebas se realizó un análisis que permitiera identificar artefactos relevantes para la prueba de la funcionalidad iniciar sesión. Como resultado, se identificaron los siguientes documentos:

- Diagrama de casos de usos (Fig. 4): permite identificar las funcionalidades a probar.
- Wireframe de iniciar sesión (Fig. 5): ilustra el formulario para la funcionalidad iniciar sesión.
- Historia de usuario de la funcionalidad de iniciar sesión (Tabla 2): describe lo que espera el usuario y los criterios de aceptación de la funcionalidad.

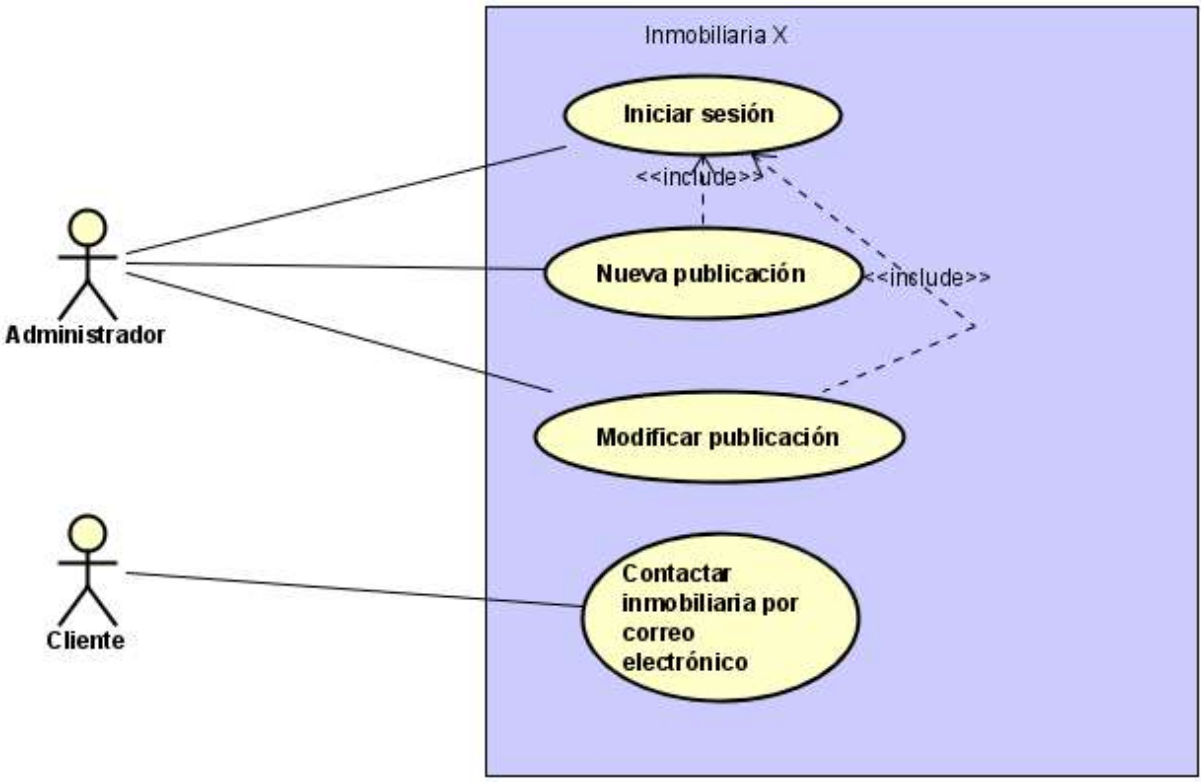


Fig. 4. Diagrama de casos de uso de las funcionalidades a probar.

Fig. 5. Wireframe de la pantalla iniciar sesión.

Tabla 2. Historia de usuario de la funcionalidad iniciar sesión.

Historia de usuario	
Número: 1	**Usuario:** administrador
Nombre de historia: iniciar sesión	
Prioridad en negocio: alta	**Riesgo en desarrollo:** alta
Puntos estimados: 12	**Iteración asignada:** 1
Descripción: quiero poder acceder a mi cuenta y datos privados, a través de un nombre de usuario y contraseña, con el fin de protegerlos y garantizar la confidencialidad de los mismos.	
Criterios de aceptación: ● Solo se accede a la pantalla de administración cuando se cargan el nombre de usuario y contraseña correspondientes al administrador. En caso contrario se muestra un mensaje de error.	

4.1.2 A1 Definir los requisitos de prueba

Según la Fig. 3, la actividad A1 utiliza los siguientes artefactos:

- ❖ Objetivo de la prueba (definido al inicio del Capítulo 4).
- ❖ Declaración positiva de la situación particular de prueba de alto nivel (definida al inicio del Capítulo 4).
- ❖ Bases de pruebas (definidas en la Sección 4.1.1).

Al finalizar la actividad se obtiene el requisito de prueba (Tabla 3).

Tabla 3. Requisitos de prueba de la funcionalidad iniciar sesión.

Label: TR-System-IniciarSesion.
Statement: se requiere verificar la correctitud de la funcionalidad que permite iniciar sesión al usuario administrador.
Testable Entity Phase: desarrollo.
Test Level: sistema.
Completion Criteria: se debe desarrollar al menos un caso de prueba para cada una de las siguientes situaciones:

- El usuario inicia sesión de forma satisfactoria, ingresando correctamente su usuario y contraseña.
- El usuario ingresa incorrectamente algún o algunos de los datos de inicio de sesión, o deja algún campo sin completar, y, por lo tanto, no puede iniciar sesión correctamente. Además, el usuario debe poder visualizar mensajes de error adecuados.
- El usuario intenta iniciar sesión pero no logra hacerlo dado que el usuario y/o contraseña son incorrectos. Luego, corrige el error y logra iniciar sesión satisfactoriamente.

Refers to Testable Entity: Inmobiliaria X.
Refers to Test Context Entities: Base de datos de pruebas.

Al observar la Tabla 3, el primer campo a completar es *label*. Para ello se utilizó *TR-System-IniciarSesion* como nombre para identificar el requisito de prueba de forma única. Según la *situación particular de prueba de alto nivel* se espera que la funcionalidad esté libre de errores por lo que el campo *statement* es: "*se requiere verificar la correctitud de la funcionalidad que permite iniciar sesión al usuario administrador*". Luego, para completar el campo *testable entity phase* se debe tener presente la fase en la que se encuentra la entidad a verificar, en este caso está en desarrollo. El siguiente campo es *test level*, dado que el objetivo de la prueba establece que se deben realizar pruebas de sistemas, entonces el nivel de prueba del requisito es de sistemas. Respecto a *completion criteria*, debe tener presente los criterios para considerar que la prueba fue completa. Notar que se debe diseñar al menos un caso de prueba para cada situación que especifique el requisito, con el objetivo de obtener un 100% de cobertura de los escenarios. Finalmente, tenemos los campos *refers to testable entity* y *refers to test context entities*. Como fue mencionado, la aplicación a probar es *Inmobiliaria X* y, como entidad de contexto de prueba de la funcionalidad iniciar sesión, se identificó a la *base de datos de pruebas*.

4.1.3. A2 Diseñar las pruebas

La actividad A2 está compuesta por tres actividades: A2.1 especificar las situaciones particulares de prueba, A2.2 diseñar los casos de pruebas y, por último, definir los procedimientos de realización. Tener en cuenta que A2.1 y A2.2 son actividades iterativas que hacen uso de la plantilla SaSTMe, es decir, cada situación particular de prueba debe ser especificada y, luego, diseñar los casos de pruebas correspondientes. Finalmente, A2.3 indica la forma en que deben realizarse los casos de pruebas, por ejemplo en el proyecto actual los casos de pruebas fueron automatizados con Cypress.

Respecto a los artefactos, la actividad especificar las situaciones particulares de prueba consume:

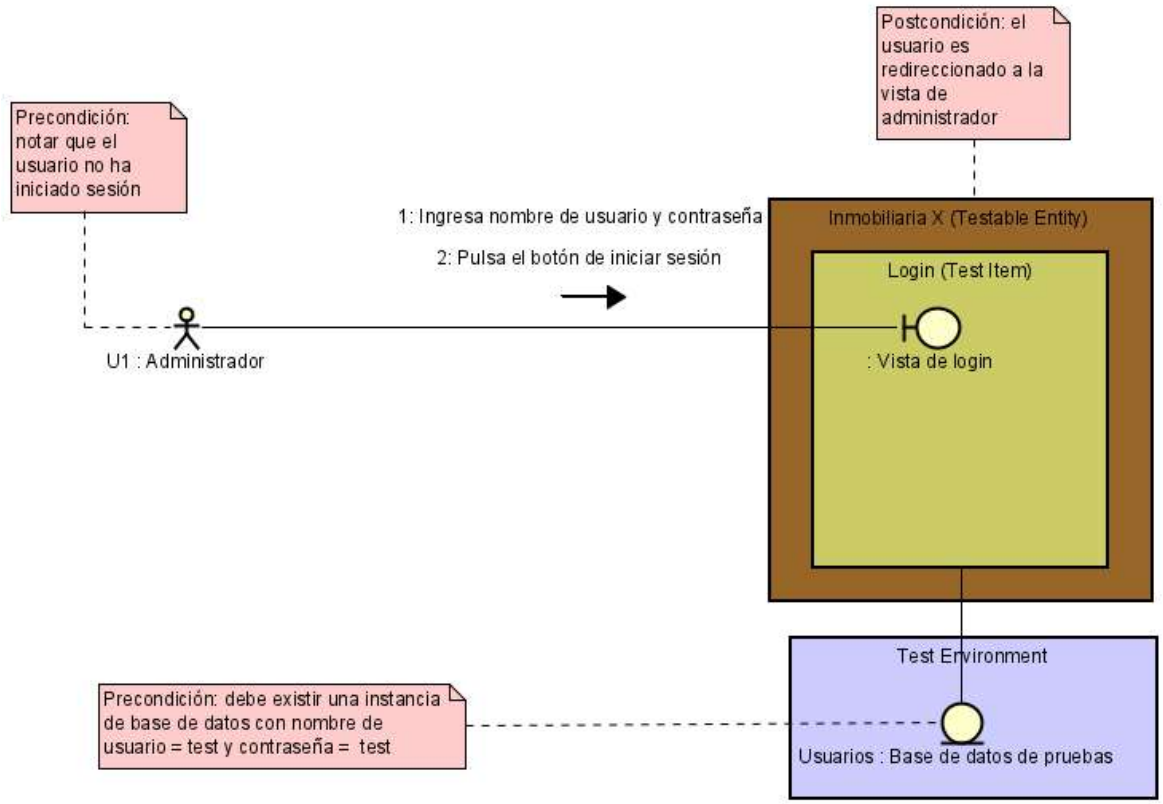
- Declaración positiva de la situación particular de prueba de alto nivel (definida al inicio del Capítulo 4).
- Requisito de prueba de iniciar sesión (Tabla 3).
- Plantilla de SaSTMe (Tabla 1).

El resultado es generar las especificaciones del modelo de prueba de las situaciones particulares de prueba. En este caso se generaron tres situaciones:

- Situación particular de prueba #1 (TPS#1, Tabla 4).
- Situación particular de prueba #2 (TPS#2, Tabla 5).
- Situación particular de prueba #3 (TPS#3, Tabla 6)

Tabla 4. Especificación de la TPS#1 de la funcionalidad iniciar sesión.

Test Particular Situation: TPS#1 -positive statement: El usuario administrador comienza en la pantalla de iniciar sesión. Luego, sin estar autenticado, el usuario ingresa el nombre de usuario y contraseña correctos. Por último, presiona el botón de iniciar sesión y logra ingresar satisfactoriamente, por lo que es redireccionado a la vista del administrador. - (Particular) Situation Model Specification:



-Conditions:

-preconditions: existe una instancia en la base de datos de prueba con los datos nombre de usuario = “test” y contraseña = “test”. Además, el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: el usuario logró autenticarse y se encuentra en la vista del administrador.

Testable Entity: Inmobiliaria X.

Test Item: funcionalidad iniciar sesión.

Test Target: funcionalidad iniciar sesión.

Test Context Entities: base de datos de pruebas.

-influences: cuando el usuario presiona iniciar sesión, la funcionalidad de login realiza una consulta a la base de datos para verificar la existencia de los datos ingresados, es decir, del nombre de usuario y su correspondiente contraseña.

Test Requirement: TR-System-IniciarSesion (Tabla 3).

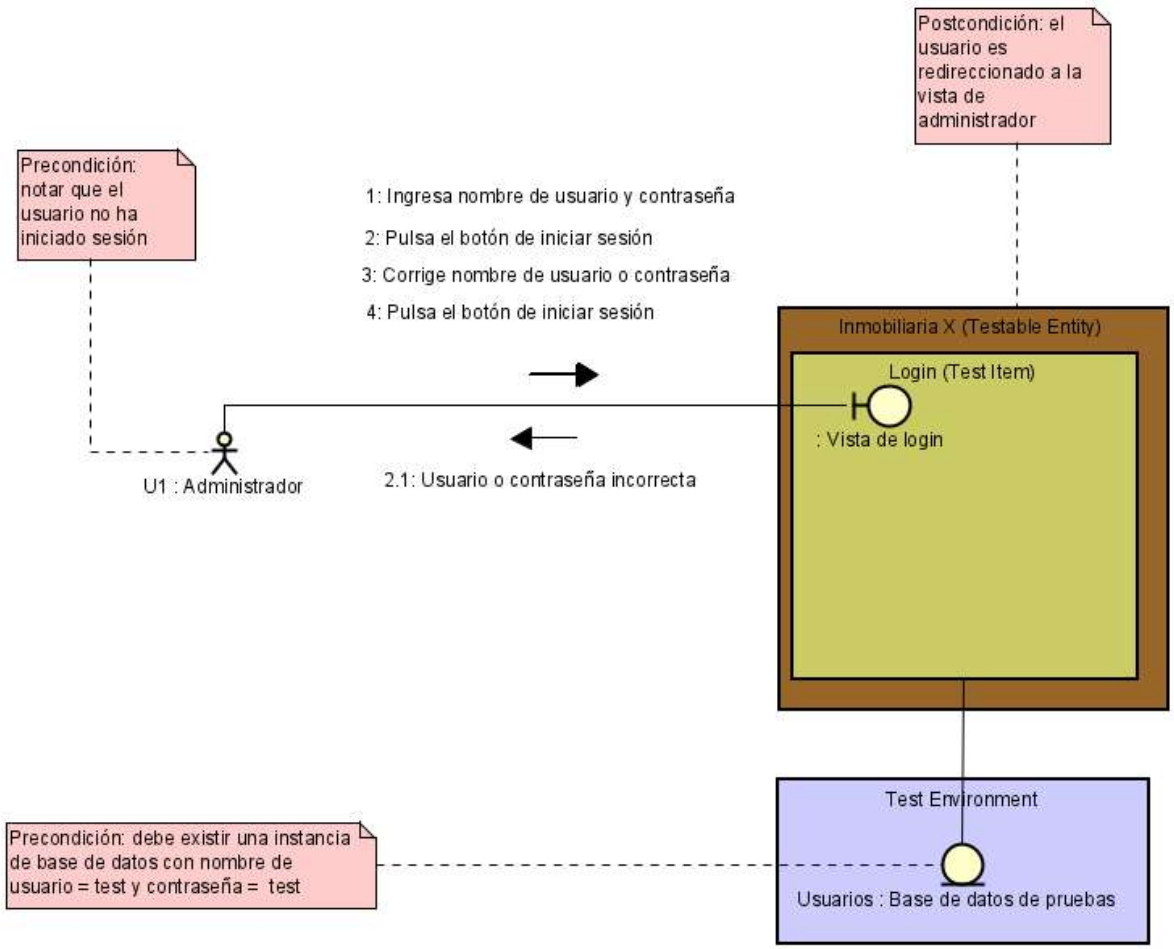
Test Basis:

- Diagrama de casos de uso (Fig. 4).
- Wireframe de funcionalidad iniciar sesión (Fig. 5).
- Historia de usuario de la funcionalidad iniciar sesión (Tabla 2).

Tabla 5. Especificación de la TPS#2 de la funcionalidad iniciar sesión.

Test Particular Situation:
TPS#2
-positive statement:
El usuario administrador comienza en la pantalla de iniciar sesión. Luego, sin estar autenticado, realiza la funcionalidad de iniciar sesión con datos incorrectos. Luego, corrige e intenta de nuevo pero utilizando los datos correctos de usuario y contraseña. Finalmente, el usuario logra autenticarse correctamente.

- (Particular) Situation Model Specification:



-Conditions:
-preconditions: existe una instancia en la base de datos de prueba con los datos nombre de usuario = “test” y contraseña = “test”. Además, el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.
-postconditions: el usuario logró autenticarse y se encuentra en la vista del administrador.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad iniciar sesión.
Test target: funcionalidad iniciar sesión.

Test Context Entities: base de datos de pruebas.

-influences: cuando el usuario presiona iniciar sesión la funcionalidad de login realiza una

consulta a la base de datos para verificar la existencia del nombre de usuario y su correspondiente contraseña.

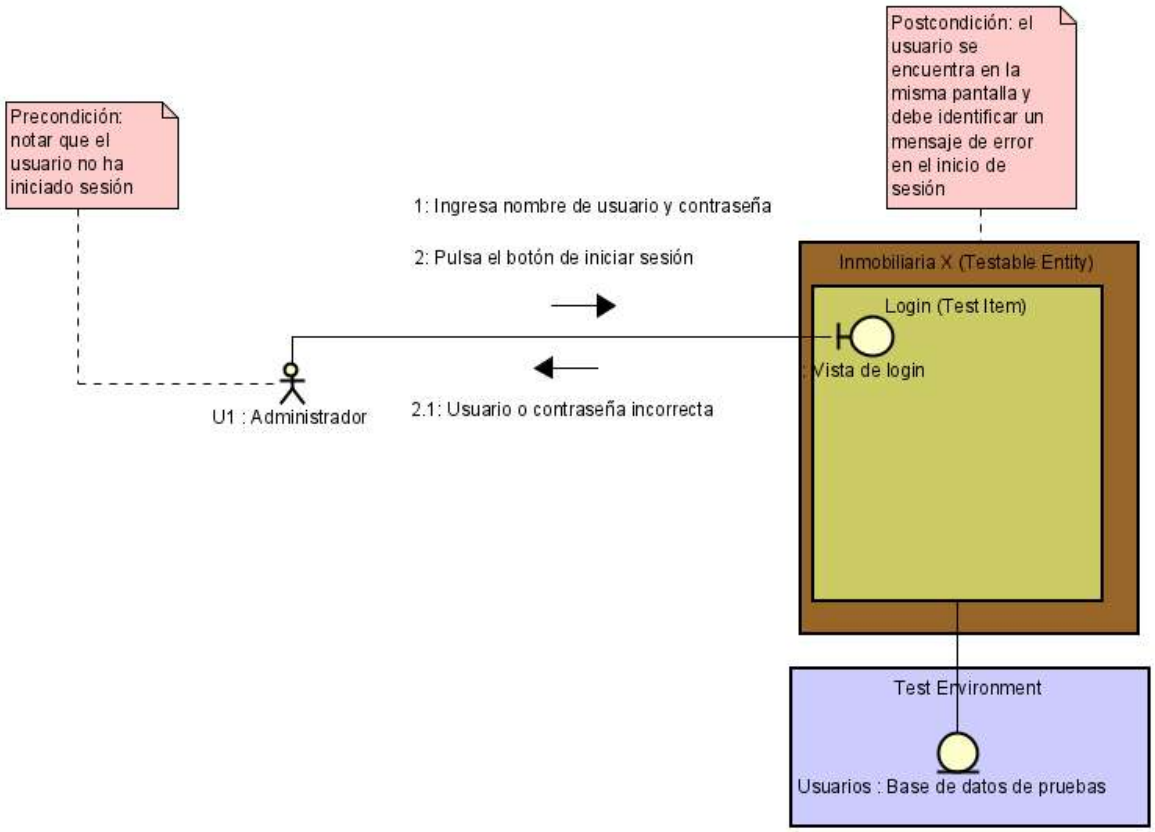
Test Requirement: TR-System-IniciarSesion (Tabla 3).

Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de funcionalidad iniciar sesión (Fig. 5).
- Historia de usuario de funcionalidad iniciar sesión (Tabla 2).

Tabla 6. Especificación de la TPS#3 de la funcionalidad iniciar sesión.

Test Particular Situation:
TPS#3
-positive statement:
El usuario administrador comienza en la pantalla de iniciar sesión. Luego, sin estar autenticado, ingresa usuario y contraseña. Luego, realiza la funcionalidad de iniciar sesión y no logra ingresar debido a un usuario o contraseña incorrecta.

-(Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.
-postconditions: el usuario se encuentra en la misma pantalla y debe identificar un mensaje de error en el inicio de sesión.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad iniciar sesión.
Test target: funcionalidad iniciar sesión.

Test Context Entities: base de datos de pruebas. -influences: cuando el usuario presiona iniciar sesión la funcionalidad de login realiza una consulta a la base de datos para verificar la existencia del nombre de usuario y su correspondiente contraseña.
Test Requirement: TR-System-IniciarSesion (Tabla 3).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de funcionalidad iniciar sesión (Fig. 5). ● Historia de usuario de funcionalidad iniciar sesión (Tabla 2).

Es importante mencionar que SaST utiliza una forma de diseñar las situaciones particulares de pruebas que facilita su comprensión. Primero hay que tener en cuenta una situación simple y satisfactoria. Luego, diseñar otra situación satisfactoria agregando complejidad. Por último, considerar otra situación no satisfactoria.

En este caso, se puede observar como caso simple y satisfactorio a TPS#1. El usuario inicia sesión sin problemas ingresando el nombre de usuario y contraseña. Respecto a TPS#2, se clasifica como un caso complejo y satisfactorio debido a que el usuario no logra iniciar sesión por un error en los datos, luego corrige y logra cumplir con lo esperado. Por otra lado, considerando el caso de error se encuentra TPS#3 donde el usuario no logra iniciar sesión y el sistema indica un error en el nombre de usuario o contraseña.

La situación seleccionada para ejemplificar la actividad A2.1 es la simple y satisfactoria (ver Tabla 4). Por lo tanto, teniendo en cuenta lo establecido en la declaración positiva de la situación particular de prueba de alto nivel (ver comienzo del Capítulo 4) y considerando el requisito de prueba (Tabla 2), se va a especificar el primer criterio "*El usuario inicia sesión de forma satisfactoria, ingresando correctamente su usuario y contraseña*".

Al observar la Tabla 4, el campo *positive statement* se completó de la siguiente forma: "*el usuario administrador comienza en la pantalla de iniciar sesión. Luego, sin estar autenticado, el usuario ingresa el nombre de usuario y contraseña correctos. Por último, presiona el botón de iniciar sesión y logra ingresar*

satisfactoriamente, por lo que es redireccionado a la vista del administrador". La idea es que la situación a probar esté relacionada al criterio mencionado anteriormente.

Respecto al campo *situation model specification*, se utilizó una herramienta para confeccionar un diagrama de comunicación dado que facilita la comprensión a la hora de diseñar los casos de pruebas. Este describe las condiciones a cumplir, la interacción del usuario con el sistema, y además, cómo se relaciona el sistema con las entidades de contexto de prueba.

Por último, se deben aclarar cuales son las *preconditions* y *postconditions* de la situación. En este caso podemos identificar como precondición la existencia de una base de datos de prueba con un usuario "test" y contraseña "test". Por otra parte, la postcondición es la autenticación satisfactoria y redirección del usuario a la vista de administrador.

La siguiente subactividad es diseñar los casos de pruebas. Esta utiliza:

- Especificaciones del modelo de las situaciones particulares de prueba
  - TPS#1 (Tabla 4).
  - TPS#2 (Tabla 5).
  - TPS#3 (Tabla 6).
- Plantilla de SaSTMe (Tabla 1).

El resultado de la actividad son los casos de pruebas correspondientes a las situaciones particulares de prueba especificadas en la actividad anterior:

- Casos de pruebas para TPS#1 (Tabla 7).
- Casos de pruebas para TPS#2 (Tabla 8).
- Casos de pruebas para TPS#3 (Tabla 9).

Tabla 7. Casos de pruebas de la TPS#1 de la funcionalidad iniciar sesión.

Test Cases:

TC#1.1
-input:
nombre de usuario = “test”,
contraseña = “test”.

-expected result: el usuario inició sesión satisfactoriamente.

-preconditions: existe una instancia en la base de datos de prueba con los datos nombre de usuario = “test” y contraseña = “test”. Además, el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: el usuario logró autenticarse y se encuentra en la vista del administrador.

Tabla 8. Casos de pruebas de la TPS#2 de la funcionalidad iniciar sesión.

Test Cases:

TC#2.1
-input:
nombre de usuario = “test1”,
contraseña = “test1”

-expected result: usuario no logra iniciar sesión satisfactoriamente.

-preconditions: existe una instancia en la base de datos de prueba con los datos nombre de usuario = “test” y contraseña = “test”. Además, el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: se muestra un mensaje de error donde dice “Usuario o contraseña incorrecta”.

TC#2.2
-input:
nombre de usuario = “test”,
contraseña = “test”

-expected result: usuario inició sesión satisfactoriamente.

-preconditions: se debe cumplir TC# 2.1 y debe existir una instancia en la base de datos de prueba con los datos nombre de usuario = “test” y contraseña = “test”.

-postconditions: el usuario logró autenticarse y se encuentra en la vista del administrador.

Tabla 9. Casos de pruebas de la TPS#3 de la funcionalidad iniciar sesión.

Test Cases:

TC#3.1
-input:
  nombre de usuario = “test1”,
  contraseña = “test1”.

-expected result: usuario no logra iniciar sesión.

-preconditions: el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: el usuario se encuentra en la misma pantalla y debe identificar un mensaje de error que diga “Usuario o contraseña incorrecta”.

TC#3.2
-input:
  nombre de usuario = “test”,
  contraseña = “”

-expected result: usuario no logra iniciar sesión.

-preconditions: el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: el usuario se encuentra en la misma pantalla y debe identificar un mensaje de error que diga “Ingrese una contraseña”.

TC#3.3
-input:
  nombre de usuario = “”,
  contraseña = “test”

-expected result: usuario no logra iniciar sesión.

-preconditions: el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.

-postconditions: el usuario se encuentra en la misma pantalla y debe identificar un mensaje de error que diga “Ingrese un usuario”.

TC#3.4
-input:
  nombre de usuario = “”,
  contraseña = “”

-expected result: usuario no logra iniciar sesión.

-preconditions: el usuario se encuentra en la pantalla de iniciar sesión y no se ha autenticado.
-postconditions: el usuario se encuentra en la misma pantalla y debe identificar dos mensajes de errores que digan “Ingrese un usuario” e “Ingrese una contraseña”.

Notar las diferencias de los casos de pruebas para las distintas situaciones. Por ejemplo, TPS#1 solo tiene un caso de prueba donde el usuario ingresa los datos necesarios para iniciar sesión y cumple con las condiciones establecidas. Sin embargo, TPS#2 es más complejo ya que utiliza casos de pruebas como precondiciones, es decir, antes de ejecutar TC#2.2 se debe cumplir TC#2.1. Por otro lado, TPS#3 prueba los distintos casos donde un usuario puede fallar al realizar la funcionalidad.

Retomando el ejemplo de TPS#1, se diseñarán los casos de pruebas asociados (ver Tabla 7). En este caso, dado a que es simple, solo se necesita de un caso de prueba. El campo *input* debe contener los datos de entrada que realizará el usuario para poder cumplir con la *declaración positiva de la situación particular de prueba*. Como se puede observar en la Tabla 7, se debe ingresar el usuario "test" y la contraseña "test".

En cuanto a condiciones se clasifican en dos. Estas pueden ser opcionales, es decir, un caso de prueba no necesariamente debe tener una *precondition* y/o *postcondition*. Para este caso, como precondición se necesita que exista una base de datos con un usuario "test" y contraseña "test", y además, que el usuario no esté autenticado. Por otro lado, la postcondición se espera que el usuario se haya autenticado satisfactoriamente y esté en la vista del administrador.

El campo restante es el *expected result*. Este debe describir lo que se espera luego de ejecutar las pruebas con un conjunto de entradas. En el caso de prueba de ejemplo se espera que el usuario haya iniciado sesión satisfactoriamente.

Según la Tabla 1, la especificación de la situación particular de prueba y los casos de pruebas correspondientes deben estar en la misma tabla. No obstante, en las actividades anteriores, fueron separadas con la finalidad de explicar de forma clara los artefactos resultantes.

Finalmente, la subactividad definir los procedimientos de realización, hace uso de los siguientes artefactos:

- ❖ Especificaciones de los modelos de las situaciones particulares de pruebas.
  - ➢ TPS#1 (Tabla 4).
  - ➢ TPS#2 (Tabla 5).
  - ➢ TPS#3 (Tabla 6).
- ❖ Casos de pruebas.
  - ➢ Casos de pruebas de la TPS#1 (Tabla 7).
  - ➢ Casos de pruebas de la TPS#2 (Tabla 8).
  - ➢ Casos de pruebas de la TPS#3 (Tabla 9).

El objetivo de A2.3 es producir:

- ❖ Especificaciones de los procedimientos de realización.
  - ➢ Automatización de casos de pruebas correspondiente a TPS#1 (Fig. 6).
  - ➢ Automatización de casos de pruebas correspondiente a TPS#2 (ver Fig. B1 en Anexo B.1).
  - ➢ Automatización de casos de pruebas correspondiente a TPS#3 (ver Fig. B2 y B3 en Anexo B.1).
- ❖ Requisitos del entorno de prueba (Tabla 10).

```
describe('TPS#1', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('login').then((dataSet) => this.dataSet = dataSet);
    cy.visit('/login');
    //Precondicion: debe estar en la pantalla de iniciar sesion y aun no se ha autenticado.
    cy.url().should('contain', 'login');
  })

  it('Usuario ingresa los datos correctamente', function () {
    //Act (Actuar)
    cy.login(this.dataSet[0].nombreUsuario, this.dataSet[0].contrasena);

    //Assert (Afirmar)
    //Postcondicion: debe estar en la pantalla del administrador.
    cy.url().should('contain', 'listarPublicaciones');
  })
})
```

Fig. 6. Especificación de procedimiento de realización de TPS#1 de la funcionalidad de iniciar sesión.

Tabla 10. Requisitos del entorno de prueba para la funcionalidad iniciar sesión.

Entidad de contexto de prueba	Requisitos del entorno de prueba
Base de datos de prueba.	Debe existir una instancia en la base de datos donde el nombre de usuario es “test” y la contraseña es “test”.

Cabe destacar que previo a realizar las automatizaciones, se programaron funciones que dan soporte a todas las funcionalidades a probar. Por ejemplo, cargas de formularios (ver Fig. 7) y aserciones (ver Fig. 8) y otras que pueden encontrarse en el Anexo A. Al crear estas funciones se obtiene la ventaja de evitar código repetido y reutilizarlas en donde sea necesario, minimizando el tiempo de programación.

```
export function cargarFormularioUbicacion(ubicacion) {
  cy.selectInput('#provincia', ubicacion.provincia);
  cy.selectInput('#localidad', ubicacion.localidad);
  cy.invokeInput('#direccion', ubicacion.direccion);
  cy.invokeInput('#iframe', ubicacion.iframe);
}

export function cargarFormularioPropiedad(propiedad) {
  cy.selectInput('#tipoPropiedad', propiedad.tipoPropiedad);
  cy.typeInput('#ambientes', propiedad.ambientes);
  cy.typeInput('#dormitorios', propiedad.dormitorios);
  cy.typeInput('#banos', propiedad.banos);
  cy.typeInput('#garaje', propiedad.garajes);
  cy.typeInput('#servicioSanitarioMunicipal', propiedad.servicioSanitarioMunicipal);
  cy.typeInput('#expensas', propiedad.expensas);
  cy.selectInput('#antiguedad', propiedad.antiguedad);
  cy.typeInput('#superficieTotal', propiedad.superficieTotal);
  cy.typeInput('#superficieCubierta', propiedad.superficieCubierta);
  cy.typeInput('#medidas', propiedad.medidas);
  cy.selectInput('#unidadMedidas', propiedad.unidadMedidas);
}

export function cargarFormularioServicios(servicios) {
  quitarServicios();
  if (servicios) {
    cy.get('#servicios label').each(servicioLabel => {
      cy.wrap(servicioLabel).invoke('text').then(text => {
        if (servicios.includes(text))
          cy.wrap(servicioLabel).siblings('input').check();
        else
          cy.wrap(servicioLabel).siblings('input').uncheck();
      });
    });
  }
}
```

Fig. 7. Funciones de soporte para las situaciones particulares de prueba.

```
export function asercionesUbicacion(ubicacion) {
  cy.get('#direccion').should('contain', `${ubicacion.direccion}, ${ubicacion.localidad}, ${ubicacion.provincia}`);
  if (ubicacion.iframe)
    cy.get('#iframe').should('exist');
  else
    cy.get('#iframe').should('not.exist');
}

export function asercionesPropiedad(propiedad) {
  let tipoSuperficie = propiedad.tipoSuperficie ? ` ${propiedad.tipoSuperficie}` : '';
  cy.get('#tipo-propiedad').should('contain', propiedad.tipoPropiedad);
  cy.get('#ambientes').should('contain', propiedad.ambientes ? propiedad.ambientes : '---');
  cy.get('#dormitorios').should('contain', propiedad.dormitorios ? propiedad.dormitorios : '---');
  cy.get('#banos').should('contain', propiedad.banos ? propiedad.banos : '---');
  cy.get('#garajes').should('contain', propiedad.garajes ? propiedad.garajes : '---');
  cy.get('#servicio-sanitario-municipal')
    .should('contain', propiedad.servicioSanitarioMunicipal ? propiedad.servicioSanitarioMunicipal : '---');
  cy.get('#expensas').should('contain', propiedad.expensas ? propiedad.expensas : '---');
  cy.get('#antiguedad').should('contain', propiedad.antiguedad ? propiedad.antiguedad : '---');
  cy.get('#superficie-total')
    .should('contain', propiedad.superficieTotal ? propiedad.superficieTotal + tipoSuperficie : '---');
  cy.get('#superficie-cubierta')
    .should('contain', propiedad.superficieCubierta ? propiedad.superficieCubierta + tipoSuperficie : '---');
  cy.get('#medidas').should('contain', propiedad.medidas ? propiedad.medidas : '---');
}

export function asercionesServicios(servicios) {
  if (servicios) {
    servicios.forEach(servicio => {
      cy.get('#servicios-descripcion-iframe')
        .contains(servicio)
        .should('contain', servicio);
    })
  } else
    cy.get('#servicios').should('not.exist');
}

export function asercionesPublicacion(publicacion) {
  cy.get('#titulo').should('contain', publicacion.titulo);
  cy.get('#tipo-propiedad-operacion')
    .should('contain', publicacion.tipoOperacion == "Comprar" ? "Venta" : "Alquiler");
```

Fig. 8. Funciones de soporte para las situaciones particulares de prueba.

Luego, se analizaron los datos de entradas documentados en los casos de pruebas con el propósito de crear un conjunto de datos de entradas en un archivo con extensión *.json* (ver Fig. 9). Por lo tanto, solo se necesita levantarlo y utilizarlo en los casos de pruebas automatizados.

Fig. 9. Conjunto de entradas para las situaciones particulares de prueba de iniciar sesión.

Después, se codifican los casos de pruebas de las situaciones particulares de prueba teniendo presente las *precondiciones* y *postcondiciones*, las *entradas de datos* y el *resultado esperado*. Como se mencionó anteriormente, el diagrama de comunicación brinda mejor compresión y permite identificar una secuencia de pasos a programar. Además, es importante destacar el uso de las tres A: Arrange, Act y Assert. En la sección de Arrange, se prepara el caso de prueba para llevar a cabo la etapa de Act. En la porción de código de Act se deben realizar las acciones y, luego, en Assert se realizan las aserciones que debe cumplir el caso de prueba al finalizar.

Por ejemplo, en la Fig. 6 se puede visualizar la prueba automatizada correspondiente a TPS#1. En la porción de código de Arrange se obtienen los datos de entradas de la funcionalidad iniciar sesión y se cumple con la precondición de "*estar ubicado en la pantalla de iniciar sesión*". Luego, en Act se realizan las acciones para llevar a cabo la funcionalidad de iniciar sesión con los datos de entradas correspondiente al caso de prueba. En la sección Assert se realizan las

aserciones de la postcondición y el resultado esperado. En este caso, ambos verifican que el usuario se encuentra en la pantalla del administrador.

Por otro lado, como requisito del entorno prueba (ver Tabla 10), se necesita configurar la base de datos de prueba con una instancia de usuario que cumpla con los valores nombre de usuario "test" y contraseña "test". De este modo es posible realizar la funcionalidad satisfactoriamente.

4.1.4. A3 Establecer entorno de prueba

Teniendo presente la Fig. 3, la actividad A3 es opcional y solo se realiza en caso de ser necesario establecer un entorno de prueba. Para poder llevarla a cabo utiliza:

- Entidades de contexto de prueba (Tabla 10).
- Requisitos del entorno de prueba (Tabla 10).

El resultado de la actividad es la configuración de las entidades de contexto de prueba (Tabla 11).

Tabla 11. Entidad de contexto de prueba configurada para la funcionalidad de iniciar sesión.

	IdPerfil	NombrePerfil	Contrasena	IdRol	IdPersona
1	2	test	098f6bcd4621d373cade4e832627b4f6	1	1

En la funcionalidad de iniciar sesión se puede identificar como entidad de contexto de prueba a una base de datos de pruebas y que debe cumplir con sus respectivos requisitos de prueba mencionados en la sección anterior. En este caso, el requisito de prueba es que en la base de datos exista una instancia con un nombre de usuario "test" (ver columna NombrePerfil en Tabla 11) y contraseña "test" (notar que en la Tabla 11 la contraseña "test" se encuentra cifrada por cuestiones de seguridad). Para poder realizar la configuración se utilizó al administrador de base de datos de Sql Server.

4.1.5. A4 Realizar pruebas dinámicas

La actividad actual, realizar las pruebas dinámicas, utiliza:

- Entidades de contexto de prueba configuradas (Tabla 11).
- Especificaciones de procedimientos de realización.
  - Automatización de TPS#1 (Fig. 6).
  - Automatización de TPS#2 (ver Fig. B1 en Anexo B.1).
  - Automatización de TPS#3 (ver Fig. B2 y B3 en Anexo B.1).
- Entidad verificable, en este caso la aplicación web de la inmobiliaria.

Notar que la Fig. 3 ilustra que la actividad A4 también consume los casos de pruebas. Normalmente, se utilizan para casos de ejecución manual pero en este caso ya se encuentran automatizados. Al finalizar la actividad se obtienen los resultados de las pruebas:

- Resultado de la prueba de TPS#1 (Fig. 10).
- Resultado de la prueba de TPS#2 (ver Fig. B4 en Anexo B.2).
- Resultado de la prueba de TPS#3 (ver Fig. B5 en Anexo B.2).

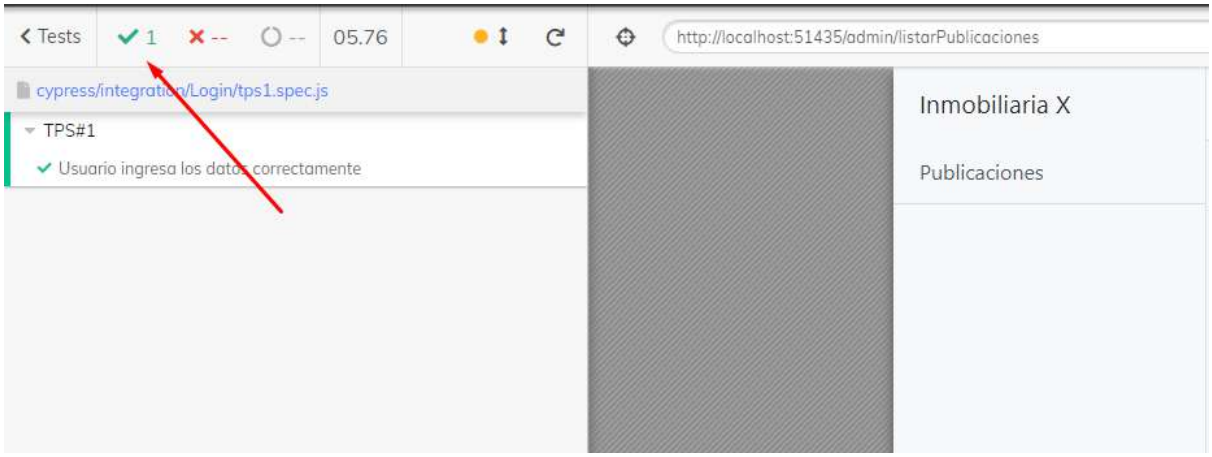


Fig. 10. Captura de ejecución de TPS#1 de la funcionalidad de iniciar sesión.

Respecto a la ejecución, la aplicación debe estar en un entorno de prueba y la entidad de contexto de prueba tiene que estar configurada correctamente. Luego, al tener los códigos automatizados de los casos de pruebas, o como indica la Fig. 3, las especificaciones de procedimientos de realización, se ejecuta la línea de comando "*npx cypress open*". Inmediatamente, aparece una interfaz de Cypress y comienza la ejecución de las pruebas. Finalmente, se obtienen las capturas de los resultados de las tres situaciones automatizadas. Notar que la flecha

de color rojo indica el resultado de la ejecución de TPS#1 y debajo se encuentran los casos de pruebas correspondientes.

4.2. Nueva publicación

4.2.1. Bases de pruebas para la funcionalidad nueva publicación

Previo a definir las bases de pruebas se realizó un análisis para identificar artefactos relevantes para la prueba de esta funcionalidad. Particularmente, se va a utilizar:

- Diagrama de casos de uso (Fig. 4).
- Historia de usuario de la funcionalidad nueva publicación (Tabla 12).
- Validaciones de los formularios (Tabla 13): describe las validaciones para los distintos formularios perteneciente a la pantalla de nueva publicación.
- Wireframes de los pasos del formulario.
  - Wireframe del esqueleto del formulario por paso (ver Fig. C1 en Anexo C.1).
  - Wireframe del paso de ubicación (Fig. 11).
  - Wireframe del paso de propiedad (ver Fig. C2 en Anexo C.1).
  - Wireframe del paso de servicio (ver Fig. C3 en Anexo C.1).
  - Wireframe del paso de publicación (ver Fig. C4 Anexo C.1).
  - Wireframe del paso de fotos (ver Fig. C5 Anexo C.1).
- DER de publicación (Fig. 12): describe las tablas de la base de datos que utiliza la funcionalidad nueva publicación.

Tabla 12. Historia de usuario de funcionalidad modificar publicación.

Historia de usuario	
Número: 2	**Usuario:** administrador
Nombre de historia: nueva publicación	
Prioridad en negocio: alta	**Riesgo en desarrollo:** medio
Puntos estimados: 20	**Iteración asignada:** 1
Descripción: quiero poder cargar nuevos inmuebles para que los clientes sepan qué es lo que hay en el mercado.	
Observaciones: para poder realizar correctamente la funcionalidad deben cumplirse ciertas validaciones en los formularios (ver Tabla 13).	
Criterios de aceptación: ● Solo es posible cargar una nueva publicación cuando se cumplen las validaciones de los formularios. En caso contrario, el sistema debe emitir un mensaje de error y no debe permitir avanzar al siguiente paso.	

Tabla 13. Tabla de validaciones de formularios de publicación.

Paso	Campo
Ubicación	● Provincia ○ Campo obligatorio. ● Localidad ○ Campo obligatorio. ● Dirección ○ Campo obligatorio. ○ Cantidad máxima de caracteres igual a 50.
Propiedad	● Tipo de propiedad ○ Campo obligatorio. ● Medidas ○ Cantidad máxima de caracteres igual a 50.
Publicación	● Título ○ Campo obligatorio. ○ Cantidad máxima de caracteres igual a 100. ● Tipo de operación ○ Campo obligatorio. ● Precio ○ Campo obligatorio. ○ Precio máximo de 9223372036854778. ● Tipo de moneda ○ Campo obligatorio.

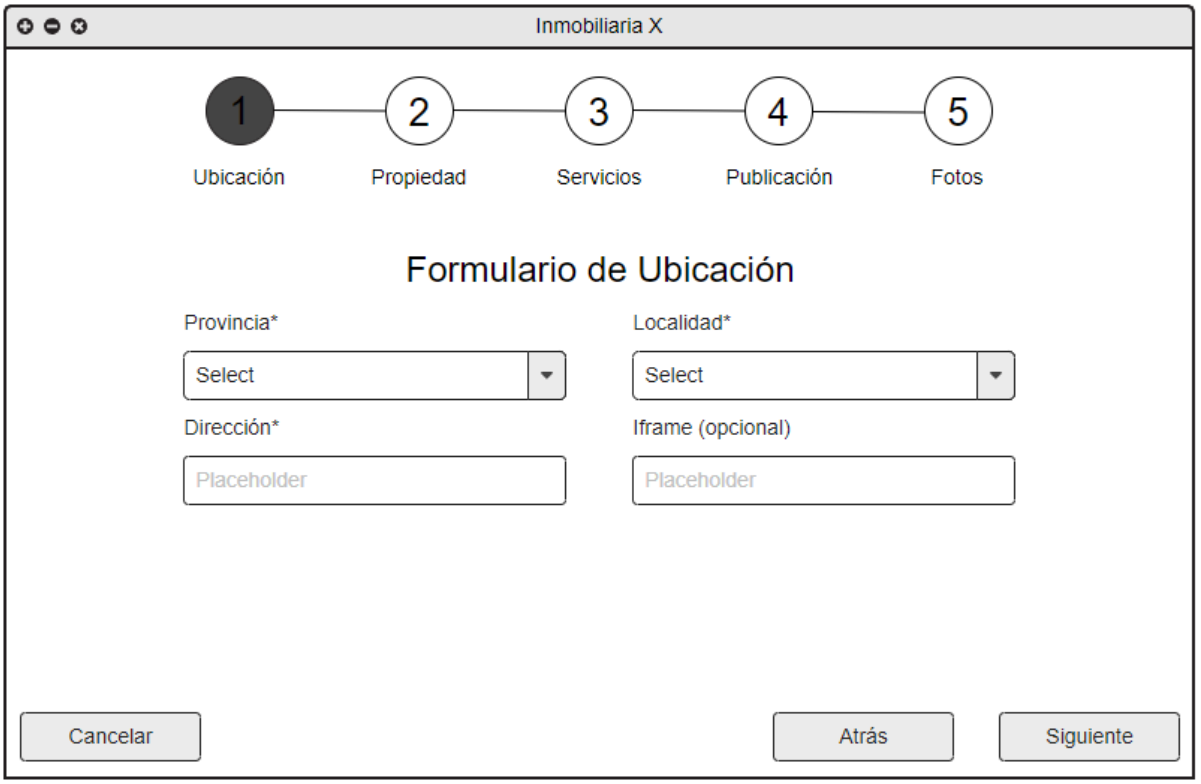


Fig. 11. Wireframe del formulario de ubicación.

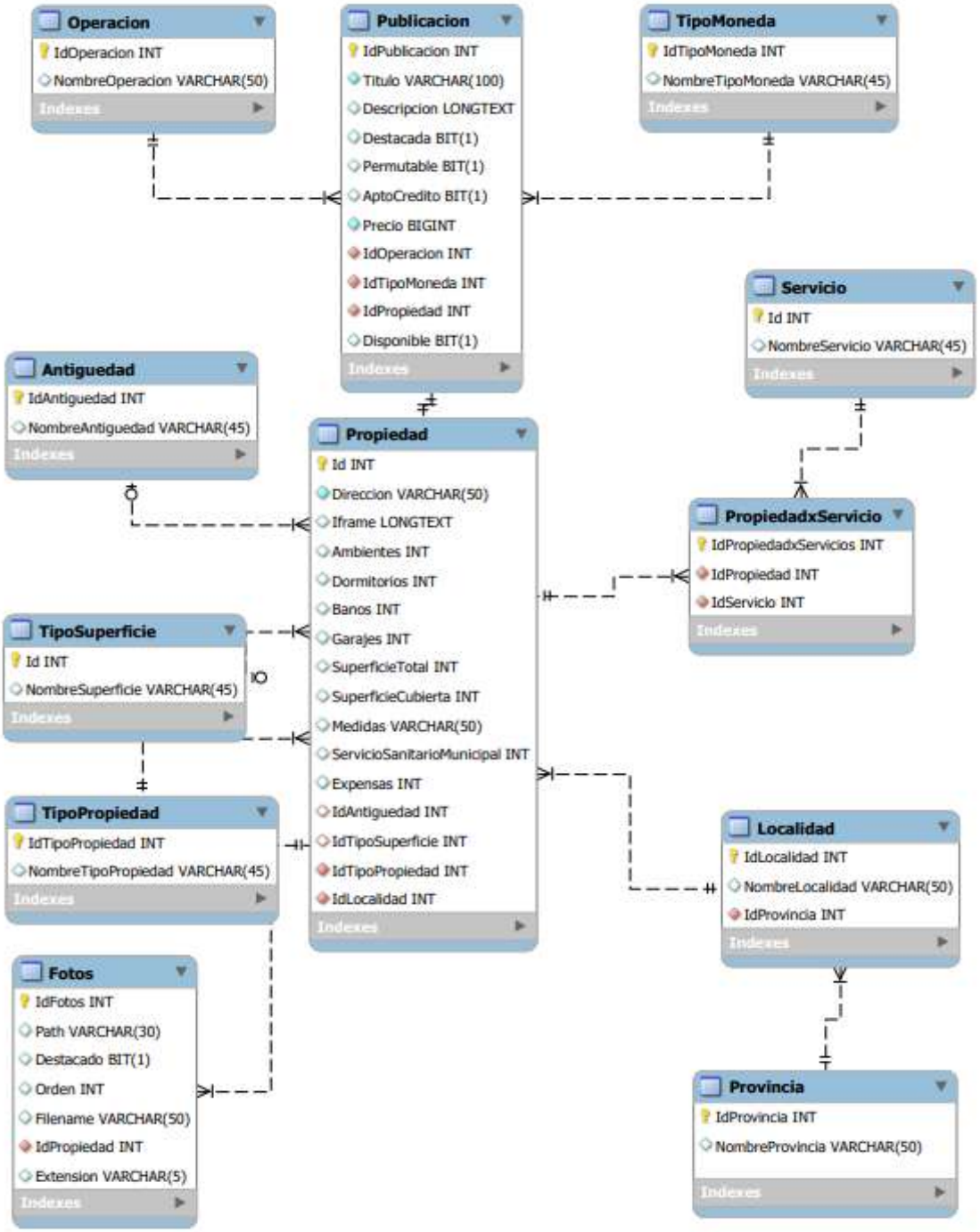


Fig. 12. DER de publicación.

Notar la relación que existe entre los wireframes, el DER de la publicación y la tabla de validaciones. Por ejemplo, el formulario de ubicación tiene cuatro campos y están relacionados con los de la base de datos. El campo *provincia* es una clave foránea en la tabla *Localidad* y su objetivo es, al seleccionar una provincia, filtrar por las localidades de esa provincia. Por otro lado, el campo *localidad* es una clave foránea que se encuentra en la tabla *Propiedad*. El campo *direccion*, se

encuentra en la tabla *Propiedad* y se puede identificar que admite hasta 50 caracteres. Notar que los campos nombrados anteriormente son obligatorios. Por último, el campo *iframe*, es un campo opcional de texto libre que tiene como finalidad almacenar un mapa generado por Google Maps.

En el Anexo C.1 se pueden encontrar los wireframes de los pasos restantes. Es importante tener en cuenta estas validaciones para probar su correctitud.

4.2.2 A1 Definir los requisitos de prueba

Para realizar la actividad A1, en la funcionalidad "Nueva publicación", las entradas fueron las siguientes:

- Objetivo de la prueba (definido al inicio del Capítulo 4).
- Declaración positiva de la situación particular de prueba de alto nivel (definida al inicio del Capítulo 4).
- Bases de prueba (definidas en la Sección 4.2.1).

Como resultado de A1, se generaron los requisitos de prueba que se muestran en la Tabla 14.

Tabla 14. Requisitos de prueba de la funcionalidad nueva publicación.

Label: TR-System-NuevaPublicacion.
Statement: se requiere verificar la correctitud de la funcionalidad que permite cargar una nueva publicación.
Testable Entity Phase: desarrollo.
Test level: sistema.
Completion Criteria: se debe desarrollar al menos un caso de prueba para cada una de las siguientes situaciones:

- El usuario administrador carga una nueva publicación satisfactoriamente sólo considerando los campos obligatorios.
- El usuario administrador carga una publicación satisfactoriamente rellenando todos los campos disponibles en los distintos formularios.
- El usuario administrador está cargando una nueva publicación y, en un determinado paso, decide volver a uno anterior para modificar algún campo. Luego, avanza a los siguientes pasos y carga la publicación satisfactoriamente.
- El usuario no cumple con las validaciones de los formularios y no logra avanzar al siguiente paso. Finalmente, cancela la operación y ninguno de los datos se guarda.
- El usuario no cumple con la validación de un formulario, corrige y avanza hasta cargar la publicación exitosamente.

Refers to Testable Entity: Inmobiliaria X.
Refers to Test Context Entities: Base de datos de pruebas.

El proceso para completar el requisito es similar a los requisitos de prueba de la funcionalidad de iniciar sesión (Tabla 3). Es importante resaltar el contenido del campo *completion criteria* de los requisitos de prueba generados por la actividad actual. Hay un criterio simple donde el usuario administrador solo completa los campos obligatorios y logra cargar una publicación. Luego, hay otros criterios donde aplican más complejidad, por ejemplo rellenar todos los formularios, volver a un paso anterior o cancelar la funcionalidad. Además, se deben tener en cuenta las validaciones para evitar un caso de error.

4.2.3. A2 Diseñar las pruebas

En la actividad A2.1, especificar las situaciones particulares de prueba, se deben consumir los siguientes artefactos:

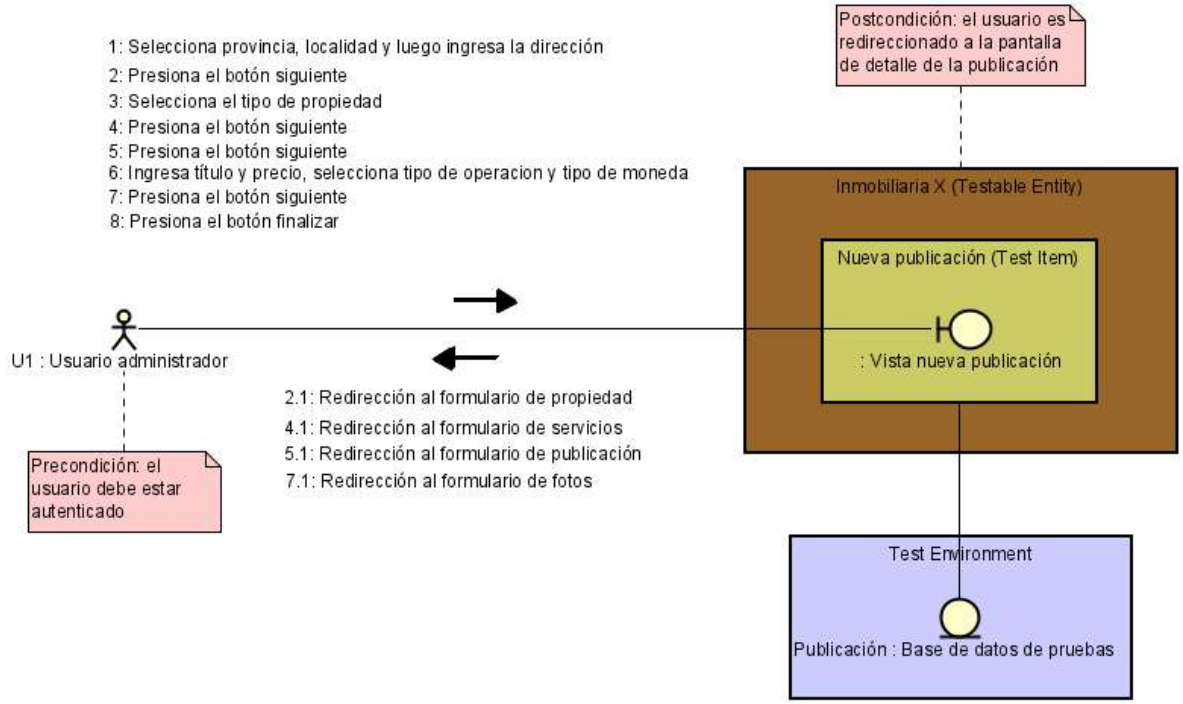
- Declaración positiva de la situación particular de prueba de alto nivel (definida al inicio del Capítulo 4).
- Requisito de prueba de la funcionalidad nueva publicación (Tabla 14).
- Plantilla de SaSTMe (Tabla 1).

Los artefactos generados por la actividad son las especificaciones del modelo de las situaciones particulares de prueba:

- Situación particular de prueba #1 (TPS#1, Tabla 15).
- Situación particular de prueba #2 (TPS#2, Tabla 16).
- Situación particular de prueba #3 (TPS#3, Tabla 17).
- Situación particular de prueba #4 (TPS#4, Tabla 18).
- Situación particular de prueba #5 (TPS#5, Tabla 19).

Tabla 15. Especificación de la TPS#1 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#1
-positive statement:
El usuario administrador carga una publicación satisfactoriamente ingresando solamente los campos obligatorios.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.
-postconditions: los datos de la publicación se encuentran almacenados correctamente en la base de datos y el usuario es redireccionado a la pantalla de detalle de la publicación.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad nueva publicación.
Test Target: funcionalidad nueva publicación.

Test Context Entities: base de datos de pruebas.

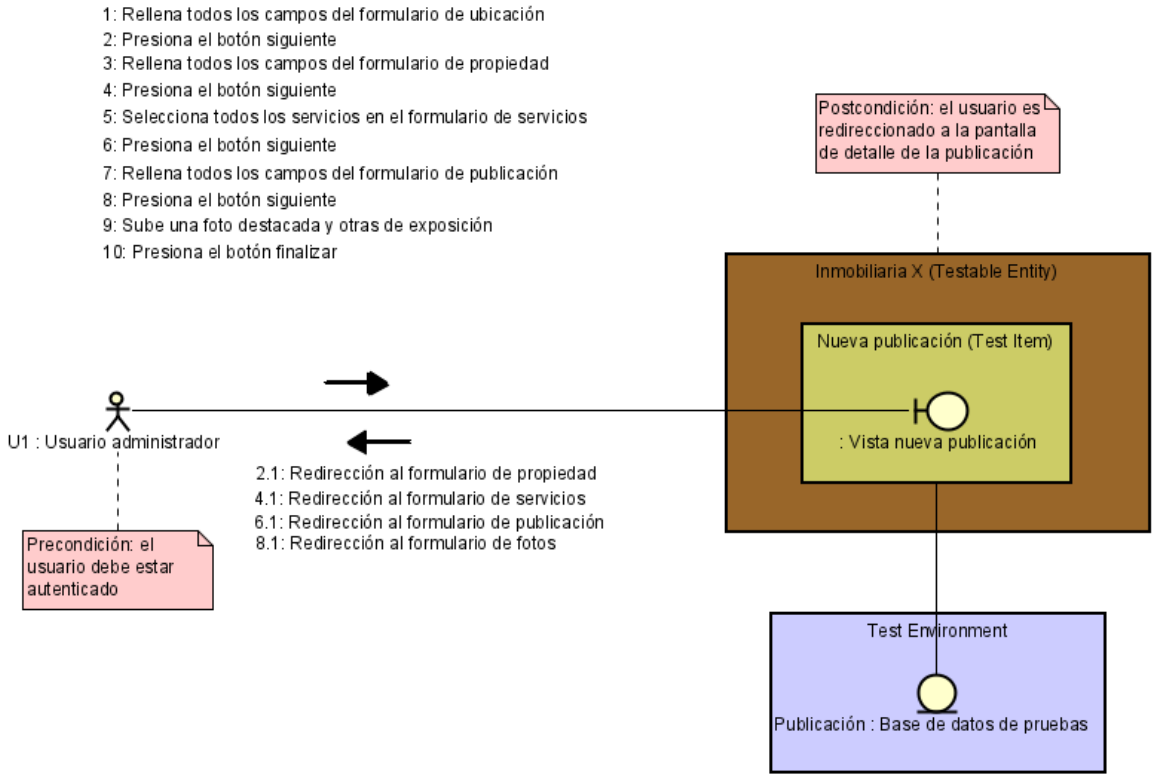
-influences: la base de datos de pruebas contiene datos de referencias que luego son utilizados en la carga y es donde se guardará la publicación al finalizar.

Test Requirement: TR-System-NuevaPublicacion (Tabla 14).

Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de nueva publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad nueva publicación (Tabla 12).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).

Tabla 16. Especificación de la TPS#2 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#2
-positive statement:
El usuario administrador carga una publicación satisfactoriamente ingresando valores en todos los campos disponibles de los formularios.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.
-postconditions: los datos de la publicación se encuentran almacenados correctamente en la base de datos y el usuario es redireccionado a la pantalla de detalle de la publicación.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad nueva publicación.
Test Target: funcionalidad nueva publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene datos de referencias que luego son utilizados en la carga y es donde se guardará la publicación al finalizar.

Test Requirement: TR-System-NuevaPublicacion (Tabla 14).

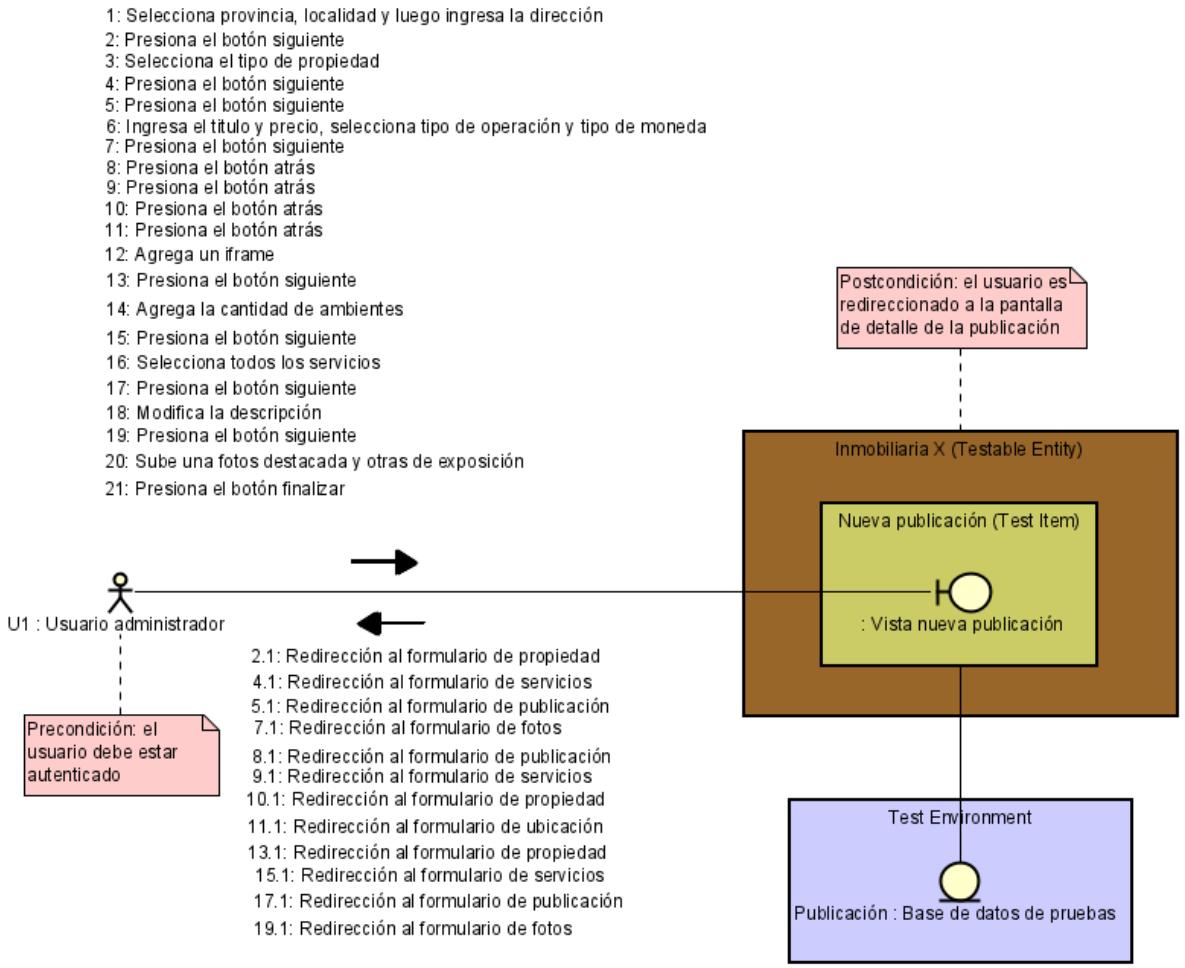
Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de nueva publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad nueva publicación (Tabla 12).

- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).

Tabla 17. Especificación de la TPS#3 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#3
-positive statement:
El usuario administrador está cargando una nueva publicación y en determinado formulario decide volver a un formulario previo para modificar algún campo. Luego, avanza a los siguientes formularios, completando los campos y finalmente se guarda la publicación satisfactoriamente.

- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario debe estar autenticado y debe estar en el formulario de ubicación de la pantalla de nueva publicación.
-postconditions: los datos de la publicación se encuentran almacenados correctamente en la base de datos y el usuario es redireccionado a la pantalla de detalle de la publicación.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad nueva publicación.
Test Target: funcionalidad nueva publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene datos de referencias que luego son utilizados en la carga y es donde se guardará la publicación al finalizar.

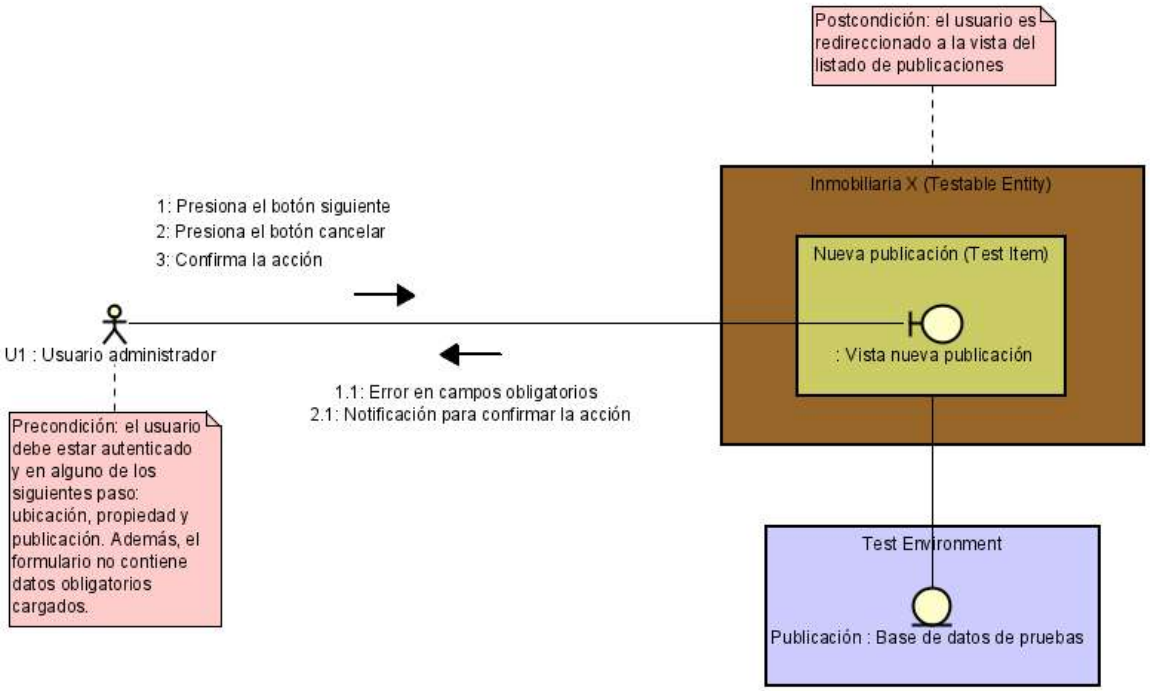
Test Requirement: TR-System-NuevaPublicacion (Tabla 14).

Test Basis:

- Diagrama de casos de uso (Fig. 4).
- Wireframe de nueva publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad nueva publicación (Tabla 12).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).

Tabla 18. Especificación de la TPS#4 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#4
-positive statement:
El usuario administrador se encuentra en un cierto paso que contiene validaciones en el formulario de nueva publicación (es decir, en el paso ubicación, propiedad o publicación) con todos los campos incompletos y no puede avanzar debido a que no cumple con la validación de los campos obligatorios. Finalmente, cancela la operación.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario debe estar autenticado, debe estar en la pantalla de nueva publicación y en alguno de los siguientes formularios: ubicación, propiedad o publicación. Además, el formulario no contiene datos obligatorios cargados.
-postconditions: el usuario administrador no logra cargar la nueva publicación debido a que no cumple con las validaciones del formulario. Luego, al cancelar la operación es redireccionado a la vista del listado de publicaciones.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad nueva publicación.
Test Target: funcionalidad nueva publicación.

Test Context Entities: base de datos de pruebas.

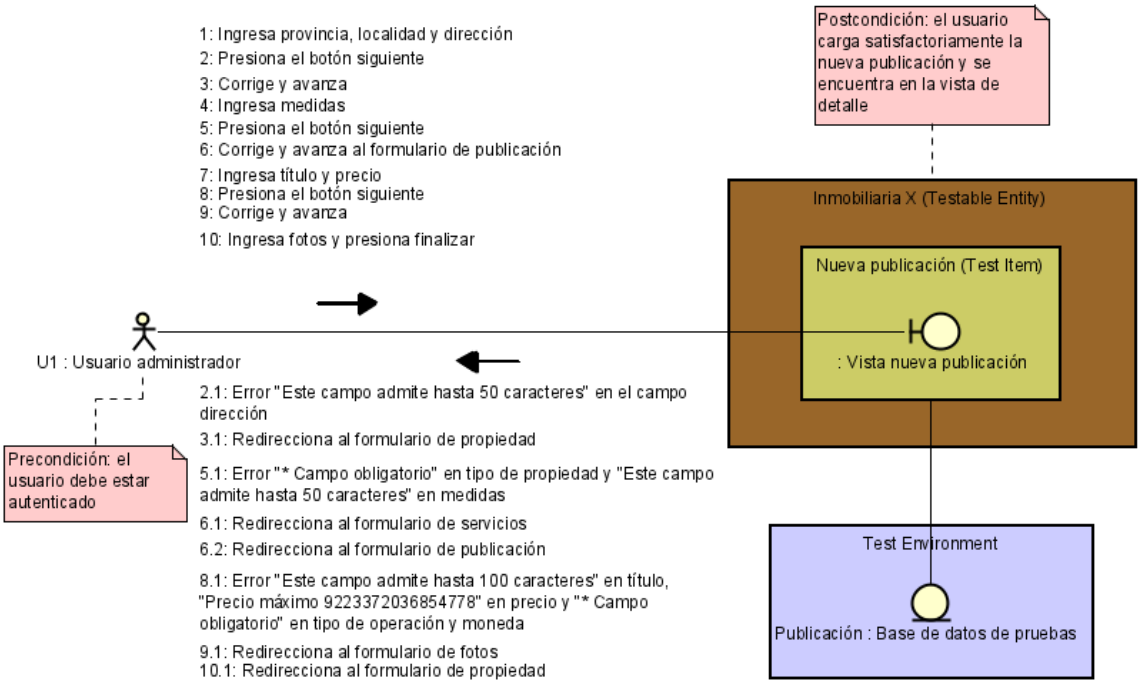
-influences: la base de datos de pruebas contiene datos de referencias que luego son utilizados en la carga y es donde se guardará la publicación al finalizar.

Test Requirement: TR-System-NuevaPublicacion (Tabla 14).

Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de nueva publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad nueva publicación (Tabla 12).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).

Tabla 19. Especificación de la TPS#5 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#5
-positive statement:
El usuario administrador se encuentra en la pantalla de nueva publicación y no cumple con las validaciones de cierto formulario (ubicación, propiedad, publicación). Luego, corrige los errores y finaliza la carga satisfactoriamente.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario debe estar autenticado y debe estar en el formulario de ubicación de la pantalla de nueva publicación.
-postconditions: los datos de la publicación se encuentran almacenados correctamente en la base de datos y el usuario es redireccionado a la pantalla de detalle de la

publicación.
Testable Entity: Inmobiliaria X. **Test Item:** funcionalidad nueva publicación. Test Target: funcionalidad nueva publicación.
Test Context Entities: base de datos de pruebas. -influences: la base de datos de pruebas contiene datos de referencias que luego son utilizados en la carga y es donde se guardará la publicación al finalizar.
Test Requirement: TR-System-NuevaPublicacion (Tabla 14).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de nueva publicación (Fig. 11 y Anexo C.1). ● Historia de usuario de la funcionalidad nueva publicación (Tabla 12). ● DER de publicación (Fig. 12). ● Validaciones de formulario de publicación (Tabla 13).

Como se mencionó en la Sección 4.1.2, los criterios de finalización (ver Tabla 14) deben tener al menos un caso de prueba para poder cubrir las distintas situaciones. TPS#1 cubre el criterio "*el usuario administrador carga una nueva publicación satisfactoriamente sólo considerando los campos obligatorios*", TPS#2 hace referencia a "*el usuario administrador carga una publicación satisfactoriamente rellenando todos los campos disponibles en los distintos formularios*", TPS#3 prueba el criterio "*el usuario administrador está cargando una nueva publicación y, en un determinado paso, decide volver a uno anterior para modificar algún campo. Luego, avanza a los siguientes pasos y carga la publicación satisfactoriamente*". Por otro lado, TPS#4 y TPS#5 prueban las validaciones pero cada una tiene su diferencia. Respecto a TPS#4, hace referencia a "*el usuario no cumple con las validaciones de los formularios y no logra avanzar al siguiente paso. Finalmente, cancela la operación y ninguno de los datos se guarda*". Por otro lado, TPS#5 cubre "el usuario no cumple con la validación de un formulario, corrige y avanza hasta cargar la publicación exitosamente". Por lo tanto, todos los criterios fueron cubiertos en las situaciones.

Respecto a las situaciones, TPS#1 es el caso simple y satisfactorio de la funcionalidad de nueva publicación, es decir, el usuario solo completa los datos obligatorios. Luego, el resto de las situaciones a probar tienen una complejidad un

poco mayor. Por ejemplo, completar todos los campos existentes, volver un paso atrás, probar las validaciones de los campos o cancelar la carga de la publicación.

Respecto a la actividad A2.2, se debe consumir:

- Especificaciones de los modelos de las situaciones particulares de prueba.
  - TPS#1 (Tabla 15).
  - TPS#2 (Tabla 16).
  - TPS#3 (Tabla 17).
  - TPS#4 (Tabla 18).
  - TPS#5 (Tabla 19).
- Plantilla de SaSTMe (Tabla 1).

Al finalizar se obtienen como resultado:

- Casos de pruebas para TPS#1 (Tabla 20).
- Casos de pruebas para TPS#2 (Tabla 21).
- Casos de pruebas para TPS#3 (Tabla 22).
- Casos de pruebas para TPS#4 (Tabla 23).
- Casos de pruebas para TPS#5 (Tabla 24).

Tabla 20. Casos de pruebas de la TPS#1 de la funcionalidad nueva publicación.

Test Cases:

TC#1.1
-input:
direccion: “Calle 108 esq 7”,
localidad: “General Pico”,
tipoOperacion: “Alquilar”,
provincia: “La Pampa”,
tipoMoneda: “$”,
tipoPropiedad: “Departamento”,
precio: 80000,
titulo: “Test FR#2 TPS#1 TC#1”

-expected result: los datos que el usuario ingreso se encuentran cargados en la base de datos.

-preconditions: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.

-postconditions: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de la publicación.

Tabla 21. Casos de pruebas de la TPS#2 de la funcionalidad nueva publicación.

Test Cases:

TC#2.1

-input:

direccion: “Calle 108 esq 7”,
localidad: “General Pico”,
provincia: “La Pampa”,
iframe: “<iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.3630185366687!2d-63.77326528515634!3d-35.66806242209237!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95c363aee37d8599%3A0x548ef9fbf8aa69b5!2sFacultad%20de%20Ingenier%C3%ADa%20-%20UNLPam!5e0!3m2!1ses-419!2sar!4v1641503144668!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe>”,
tipoPropiedad: “Casa”,
ambientes: 3,
dormitorios: 3,
banos: 3,
garaje: 3,
servicioSanitarioMunicipal: 1225,
expensas: 2222,
antiguedad: “A estrenar”,
superficieTotal: 200,
superficieCubierta: 150,
medidas: “10x20”,
unidadMedidas: “m²”,
titulo: “Test FR#2 TPS#2 TC#1”,
tipoOperacion: “Comprar”,
tipoMoneda: “$”,
precio: 0,
descripcion: “descripcion Test FR#2 TPS#2 TC#1”,
permuta: true,
destacada: true,
credito:true,
fotoDestacada: “../pruebas fotos/d1.jpeg”,
fotosExposicion: [“../pruebas fotos/h1.jpeg”,
“../pruebas fotos/h2.jpeg”,
“../pruebas fotos/h3.jpeg”,
“../pruebas fotos/h4.jpeg”,
“../pruebas fotos/h5.jpeg”,
“../pruebas fotos/h6.jpeg”,
“../pruebas fotos/h7.jpeg”,
“../pruebas fotos/h8.jpeg”],
servicios: [“Agua corriente”, “Luz eléctrica”, “Cloacas”, “Gas natural”, “Pavimento”]

-expected result: los datos que el usuario ingreso se encuentran cargados en la base de datos.

-preconditions: el usuario debe estar autenticado y debe estar en la pantalla de nueva

publicación.

-postconditions: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de la publicación.

Tabla 22. Casos de pruebas de la TPS#3 de la funcionalidad nueva publicación.

Test Cases:

TC#3.1
-input:
direccion: “Calle 108 esq 7”,
localidad: “General Pico”,
iframe: “<iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.3630185366687!2d-63.77326528515634!3d-35.66806242209237!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95c363aee37d8599%3A0x548ef9fbf8aa69b5!2sFacultad%20de%20Ingenier%C3%ADa%20-%20UNLPam!5e0!3m2!1ses-419!2sar!4v1641503144668!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe>”,
provincia: “La Pampa”,
tipoOperacion: “Alquilar”,
ambientes: 1,
tipoMoneda: “U$S”,
tipoPropiedad: “Departamento”,
precio: 25000,
titulo: “Test FR#2 TPS#3 1”,
descripcion: “descripcion para FR#2 TPS#3”,
fotoDestacada: “../pruebas fotos/d1.jpeg”,
fotosExposicion: [“../pruebas fotos/d1.jpeg”,
“../pruebas fotos/h1.jpeg”,
“../pruebas fotos/h2.jpeg”,
“../pruebas fotos/h3.jpeg”,
“../pruebas fotos/h4.jpeg”,
“../pruebas fotos/h5.jpeg”,
“../pruebas fotos/h6.jpeg”,
“../pruebas fotos/h7.jpeg”,
“../pruebas fotos/h8.jpeg”],
servicios: [“Agua corriente”, “Luz eléctrica”, “Cloacas”, “Gas natural”, “Pavimento”]

-expected result: los datos que el usuario ingreso se encuentran cargados en la base de datos.

-preconditions: el usuario debe estar autenticado y debe estar en el formulario de ubicación de la pantalla de nueva publicación.

-postconditions: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de la publicación.

Tabla 23. Casos de pruebas de la TPS#4 de la funcionalidad nueva publicación.

Test Cases:

TC#4.1
-input:
presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema no carga la publicación en la base de datos debido a que no cumple con las validaciones del formulario de ubicación.

-preconditions:el usuario debe estar autenticado y estar en el formulario de ubicación de la pantalla de nueva publicación y no debe contener ningún dato obligatorio cargado, es decir, “inicia con el formulario vacío”.

-postconditions: el usuario administrador se encuentra en el mismo paso con mensajes de error “* Campo obligatorio”.

TC#4.2
-input: presiona el botón cancelar.

-expected result: el sistema redirecciona al usuario a la vista de listado de publicaciones.

-preconditions: se debe cumplir TC#4.1.

TC#4.3
-input:
Presiona el botón siguiente en el formulario de propiedad.

-expected result: el sistema no carga la publicación en la base de datos debido a que no cumple con las validaciones del formulario de propiedad.

-preconditions: el usuario debe estar autenticado y debe estar en el formulario de propiedad de la pantalla de nueva publicación y no debe contener ningún dato obligatorio cargado, es decir, “inicia con el formulario vacío”.

-postconditions: el usuario administrador se encuentra en el mismo paso con mensajes de error “* Campo obligatorio”.

TC#4.4
-input:
presiona el botón cancelar.

-expected result: el sistema redirecciona al usuario a la vista de listado de publicaciones.

-preconditions: se debe cumplir TC#4.3.

TC#4.5
-input:
Presiona el botón siguiente en el formulario de publicación.

-expected result: el sistema no carga la publicación en la base de datos debido a que no cumple con las validaciones del formulario de publicación.

-preconditions: el usuario debe estar autenticado y debe estar en el formulario de publicación de la pantalla de nueva publicación y no debe contener ningún dato obligatorio cargado, es decir, “inicia con el formulario vacío”.

-postconditions: el usuario administrador se encuentra en el mismo paso con mensajes de error “* Campo obligatorio”.

TC#4.6
-input: presiona el botón cancelar.

-expected result: el sistema redirecciona al usuario a la vista de listado de publicaciones.

-preconditions: se debe cumplir TC#4.5.

Tabla 24. Casos de pruebas de la TPS#5 de la funcionalidad nueva publicación.

Test Cases:

TC#5.1
-input:
direccion: “Calle 108 esq 7 Calle 108 esq 7 Calle 108 esq 7 Calle 108 esq 7”,
localidad: “Barrios”,
provincia: “Jujuy”
Presiona siguiente en el formulario de ubicación.
.
-expected result: usuario administrador no logra avanzar al formulario de propiedad de la nueva publicación.

-preconditions: el usuario debe estar autenticado y debe estar en el formulario de ubicación de la pantalla de nueva publicación.

-postconditions: el usuario administrador identifica un mensaje de error “Este campo admite hasta 50 caracteres” en el campo de dirección.

TC#5.2
-input:
direccion: “Calle 108 esq 7”
Presiona siguiente en el formulario de ubicación.

-expected result: usuario administrador logra avanzar al formulario de propiedad de la nueva publicación.

-preconditions: se debe cumplir el TC#5.1.

TC#5.3
-input:
medidas: “12345678901234567890123 4 x 12345678901234567890123 4”
Presiona siguiente en el formulario de propiedad.

-expected result: usuario administrador no logra avanzar al formulario de servicios de la nueva publicación.

-preconditions: se debe cumplir el TC#5.2.
-postconditions: el usuario administrador identifica mensajes de error “* Campo

obligatorio” en el campo de tipo de propiedad y “Este campo admite hasta 50 caracteres” en medidas.

TC#5.4
-input:
  tipoPropiedad: “Departamento”,
  medidas: “12345 x 12345”,
Presiona siguiente en el formulario de propiedad y servicios.

-expected result: usuario administrador logra avanzar al formulario de publicación de la nueva publicación.

-preconditions: se debe cumplir el TC#5.3.

TC#5.5
-input:
  titulo: “Test FR#2 TPS#5Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5”,
  precio: 99999999999999999
Presiona siguiente en el formulario de publicación.

-expected result: usuario administrador no logra avanzar al formulario de fotos de la nueva publicación.

-preconditions: se debe cumplir el TC#5.4.

-postconditions: el usuario administrador identifica mensajes de error “Este campo admite hasta 100 caracteres” en el campo de título, “Precio máximo 9223372036854778” en precio y “* Campo obligatorio” en tipo de operación y moneda.

TC#5.6
-input:
  titulo: “Test FR#2 TPS#5”,
  tipoOperacion: “Alquilar”,
  tipoMoneda: “$”,
  precio: 99999,
  fotoDestacada: “../pruebas fotos/d1.jpeg”,
  fotosExposicion: [“../pruebas fotos/h1.jpeg”,
    “../pruebas fotos/h2.jpeg”,
    “../pruebas fotos/h3.jpeg”,
    “../pruebas fotos/h4.jpeg”,
    “../pruebas fotos/h5.jpeg”,
    “../pruebas fotos/h6.jpeg”,
    “../pruebas fotos/h7.jpeg”,
    “../pruebas fotos/h8.jpeg”]
Presiona siguiente en el formulario de publicación. Luego, presiona finalizar en el formulario de fotos.

-expected result: todos los datos ingresados en los casos de prueba TC#5.1, TC#5.2, TC#5.4 y TC#5.6 se encuentran en la base de datos.

-preconditions: se debe cumplir el TC#5.5.
-postconditions: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de la publicación.

Es importante mencionar la diferencia entre los casos de pruebas. Por ejemplo en las tres primeras situaciones solo se identifica un único caso de prueba. Por otro lado, en TPS#4 y TPS#5, se puede apreciar la dependencia entre casos de pruebas para poder identificar acciones como cancelar la operación, validaciones o correcciones para poder cumplir con las validaciones.

A continuación, en la actividad A2.3 se necesitan los siguientes artefactos:

- Especificaciones de los modelos de las situaciones particulares de pruebas.
  - TPS#1 (Tabla 15).
  - TPS#2 (Tabla 16).
  - TPS#3 (Tabla 17).
  - TPS#4 (Tabla 18).
  - TPS#5 (Tabla 19).
- Casos de pruebas.
  - Casos de pruebas de la TPS#1 (Tabla 20).
  - Casos de pruebas de la TPS#2 (Tabla 21).
  - Casos de pruebas de la TPS#3 (Tabla 22).
  - Casos de pruebas de la TPS#4 (Tabla 23).
  - Casos de pruebas de la TPS#5 (Tabla 24).

Al finalizar la actividad se obtienen:

- Especificaciones de los procedimientos de realización.
  - Automatización de casos de pruebas correspondiente a TPS#1 (ver Fig. C10 y C11 en Anexo C.3).
  - Automatización de casos de pruebas correspondiente a TPS#2 (ver Fig. C12 y C13 en Anexo C.3).
  - Automatización de casos de pruebas correspondiente a TPS#3 (ver Fig. C14 y C15 en Anexo C.3).
  - Automatización de casos de pruebas correspondiente a TPS#4 (ver Fig. C16 y C17 en Anexo C.3).

➢ Automatización de casos de pruebas correspondiente a TPS#5 (Fig. 13, 14 y 15).

❖ Requisitos del entorno de prueba (Tabla 25).

```
describe('TPS#5', () => {
  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('nueva-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion:  el usuario debe estar autenticado y debe estar en el
     formulario de ubicación de la pantalla de nueva publicación.  */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.get('#nueva-publicacion').click();
    cy.url().should('contain', 'nuevaPublicacion');
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
  })

  it('No cumple con las validaciones, corrige y avanza', function () {
    //Act (Actuar)
    //form ubicacion - paso 1
    cargarFormularioUbicacion(this.publicaciones[5].ubicacion);
    cy.get('#boton-siguiente-web').click()

    /* Postcondicion: el usuario administrador identifica un mensaje de error "Este campo
    admite hasta 50 caracteres" en el campo de dirección */
    cy.get('#direccion-length').should('contain', "Este campo admite hasta 50 caracteres");

    //Corrige paso 1, avanza y no cumple con la validacion del paso 2
    cargarFormularioUbicacion(this.publicaciones[6].ubicacion);
    cy.get('#boton-siguiente-web').click()

    /* Precondicion: usuario administrador logra avanzar al formulario de propiedad de la
    nueva publicación */
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');

    cargarFormularioPropiedad(this.publicaciones[6].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true });

    /* Postcondicion: el usuario administrador identifica mensajes de error "* Campo obligatorio"
     en el campo de tipo de propiedad y "Este campo admite hasta 50 caracteres" en medidas */
    cy.get('#tipoPropiedad-requerido').should('contain', "* Campo obligatorio");
    cy.get('#medidas-length').should('contain', "Este campo admite hasta 50 caracteres");

    //Corrige paso 2, avanza al formulario de servicios
    cargarFormularioPropiedad(this.publicaciones[7].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true });

    //Avanza a publicacion
    cy.get('#boton-siguiente-web').click({ force: true });
```

Fig. 13. Especificación de procedimiento de realización de TPS#5 de la funcionalidad de nueva publicación.

```
/* Precondicion: usuario administrador logra avanzar al formulario de publicación de la nueva
 publicación. */
cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');

cargarFormularioPublicacion(this.publicaciones[7].publicacion);
cy.get('#boton-siguiente-web').click({ force: true });

/* Postcondicion: el usuario administrador identifica mensajes de error "Este campo admite hasta
 100 caracteres" en el campo de titulo, "Precio máximo 9223372036854778" en precio y "* Campo
 obligatorio" en tipo de operación y moneda */
cy.get('#titulo-length').should('contain', "Este campo admite hasta 100 caracteres");
cy.get('#precio-maximo').should('contain', "Precio máximo 9223372036854778");
cy.get('#tipoOperacion-requerido').should('contain', "* Campo obligatorio");
cy.get('#tipoMoneda-requerido').should('contain', "* Campo obligatorio");

//Corrige paso 4, avanza y carga el formulario de fotos
cargarFormularioPublicacion(this.publicaciones[8].publicacion);
cy.get('#boton-siguiente-web').click({ force: true });

cargarFormularioFotos(this.publicaciones[8].fotos);
cy.get('#boton-finalizar-web').click();

//Assert
/* Postcondicion: el usuario se encuentra en la pantalla del detalle de la publicación y puede
 ver el detalle de la publicación. */
cy.url().should('contain', 'detallePublicacion')
  .then(() => {
    cy.wait(2000);
    /* Se crea un objeto porque es un caso particular donde se usan datos de la publicacion "5" y
    "6" del dataset */
    let ubicacion = {
      provincia: this.publicaciones[5].ubicacion.provincia,
      localidad: this.publicaciones[5].ubicacion.localidad,
      direccion: this.publicaciones[6].ubicacion.direccion
    }
```

Fig. 14. Especificación de procedimiento de realización de TPS#5 de la funcionalidad de nueva publicación.

```
    asercionesUbicacion(ubicacion);
    asercionesPropiedad(this.publicaciones[7].propiedad);
    asercionesServicios(this.publicaciones[7].servicios);
    asercionesPublicacion(this.publicaciones[8].publicacion);
    asercionesFotos(this.publicaciones[8].fotos);

    this.publicaciones[8].ubicacion = ubicacion;
    this.publicaciones[8].propiedad = this.publicaciones[7].propiedad;
    this.publicaciones[8].servicios = this.publicaciones[7].servicios;
    cy.url()
      .then(url => {
        let idPublicacion = url.split('/').pop();
        /* Resultado esperado: los datos que el usuario ingreso se encuentran cargados en la base de datos. */
        asercionesBD(this.publicaciones[8], idPublicacion);
      });
  });
});
})
```

Fig. 15. Especificación de procedimiento de realización de TPS#5 de la funcionalidad de nueva publicación.

Tabla 25. Requisitos del entorno de prueba para la funcionalidad nueva publicación.

Entidad de contexto de prueba	**Requisitos del entorno de prueba**
Base de datos de prueba.	Debe tener contenidos en las tablas de Antiguedad, Localidad, Operacion, Provincia, Servicio, TipoMoneda, TipoPropiedad y TipoSuperficie.

Antes de comenzar con la automatización de las pruebas, tener presente que al igual que la funcionalidad iniciar sesión, se creó un conjunto de datos de

entradas. En el Anexo C.2 se muestra el código con los datos de entrada utilizados para las pruebas.

Se va a utilizar como ejemplo la automatización de TPS#5 (Fig. 13, 14 y 15), ya que es una situación compleja que comprueba la validaciones de los formularios. Esta situación verifica que si el usuario se encuentra en un determinado paso y no cumple con alguna validación entonces no puede avanzar. Además, debe permitir corregir los campos inválidos y continuar con la carga de la publicación (ver Tabla 19 y 24).

Al inicio del caso de prueba automatizado de TPS#5 (ver Fig. 13), se puede identificar una función denominada "*beforeEach*". Dicha función tiene la finalidad de ejecutarse antes de que inicie cada caso de prueba y, en este caso, se utiliza para ejecutar la sección Arrange. En la sección mencionada se obtienen los datos de entrada relacionados a la funcionalidad y se cumple con la precondición: "el usuario debe estar autenticado y debe estar en el formulario de ubicación de la pantalla de nueva publicación". Notar que la función "*should*" sirve para validar que una aserción se cumpla.

En la sección Act se puede visualizar como los distintos casos de pruebas están relacionados por medio de precondiciones. Cabe destacar que los casos de pruebas confeccionados en las tablas se automatizaron en un único caso de prueba. El propósito fue relacionar la dependencia entre cada uno, ya que la versión actual de Cypress no soporta como precondición a un caso de prueba.

Continuando con la codificación, comienza con la carga del formulario de ubicación, luego presiona el botón siguiente y surgen mensajes de validación en el formulario. Notar que esto es el caso de prueba #1, o TC#5.1 de la Tabla 24. Después, se puede visualizar el TC#5.2, donde se corrige el error en el campo de dirección y avanza al siguiente formulario. La misma secuencia se repite para los formularios de propiedad y publicación.

Respecto a la sección de Assert (ver Fig. 14 y 15, y TC#5.6 de la Tabla 24), se puede observar que por un lado realiza la validación de la postcondición: "*el*

usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de la publicación", mientras que por otro lado se valida el resultado esperado: "*todos los datos ingresados en los casos de prueba TC#5.1, TC#5.2, TC#5.4 y TC#5.6 se encuentran en la base de datos*". De esta manera finaliza la automatización de TPS#5.

Finalmente, es importante configurar la base de datos de pruebas con los datos requeridos en la Tabla 25, dado a que son utilizados para almacenar información y seleccionar datos ya existentes en la base de datos. Por ejemplo, los tipos de propiedad.

4.2.4. A3 Establecer entorno de prueba

Para iniciar con A3, se necesita consumir los siguientes artefactos:

- Entidades de contexto de prueba (Tabla 25).
- Requisitos del entorno de prueba (Tabla 25).

El resultado de la actividad es la base de datos de pruebas configurada con los requisitos mencionados anteriormente.

	IdLocalidad	NombreLocalidad	IdProvincia
1	1	25 de Mayo	1
2	2	3 de febrero	1
3	3	Adolfo Alsina	1
4	4	Adolfo Gonzáles Cháves	1
5	5	Aguas Verdes	1
6	6	Alberti	1
7	7	Arrecifes	1
8	8	Ayacucho	1
9	9	Azul	1
10	10	Bahía Blanca	1
11	11	Balcarce	1
12	12	Baradero	1
13	13	Benito Juárez	1
14	14	Berisso	1
15	15	Bolívar	1
16	16	Bragado	1
17	17	Brandsen	1
18	18	Campana	1
19	19	Cañuelas	1

Fig. 16. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

Para poder realizar la ejecución de las pruebas se necesita configurar la base de datos de prueba. En la Fig. 16 y en el Anexo C.4 se encuentran las capturas de pantallas de la entidad de contexto de prueba configurada para la funcionalidad nueva publicación. Para ello, se utilizó el administrador de bases de datos de SQL Server.

En estas capturas se pueden identificar dos tipos de tablas. Por un lado, tablas donde contienen información necesaria para los pasos del formulario. Por ejemplo, los servicios, antigüedades, provincias, entre otras. Por otro lado, hay tablas que almacenan la información de la propiedad y publicación.

4.2.5. A4 Realizar pruebas dinámicas

La actividad A4 consume:

- Entidades de contexto de prueba configuradas (ver Fig. 16 y Anexo C.4).
- Especificaciones de procedimientos de realización:
  - Automatización de TPS#1 (ver Fig. C10 y C11 en Anexo C.3).
  - Automatización de TPS#2 (ver Fig. C12 y C13 en Anexo C.3).
  - Automatización de TPS#3 (ver Fig. C14 y C15 en Anexo C.3).
  - Automatización de TPS#4 (ver Fig. C16 y C17 en Anexo C.3).
  - Automatización de TPS#5 (Fig. 13, 14 y 15).
- Entidad verificable, es decir, la aplicación web de la inmobiliaria.

Los artefactos producidos en la actividad son los resultados de las pruebas:

- Resultado de la prueba de TPS#1 (ver Fig. C26 y C27 en Anexo C.5).
- Resultado de la prueba de TPS#2 (ver Fig. C28 en Anexo C.5).
- Resultado de la prueba de TPS#3 (ver Fig. C29 en Anexo C.5).
- Resultado de la prueba de TPS#4 (ver Fig. C30 en Anexo C.5).
- Resultado de la prueba de TPS#5 (Fig. 17 y 18).

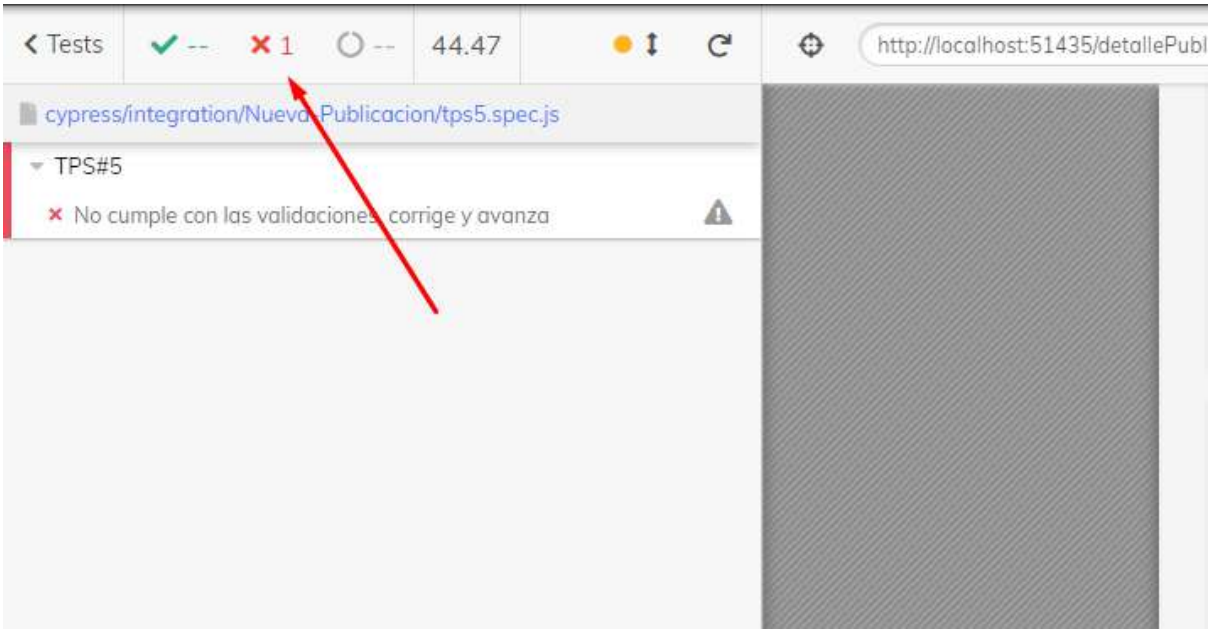


Fig. 17. Captura de pantalla de la ejecución de TPS#5 de la funcionalidad nueva publicación.

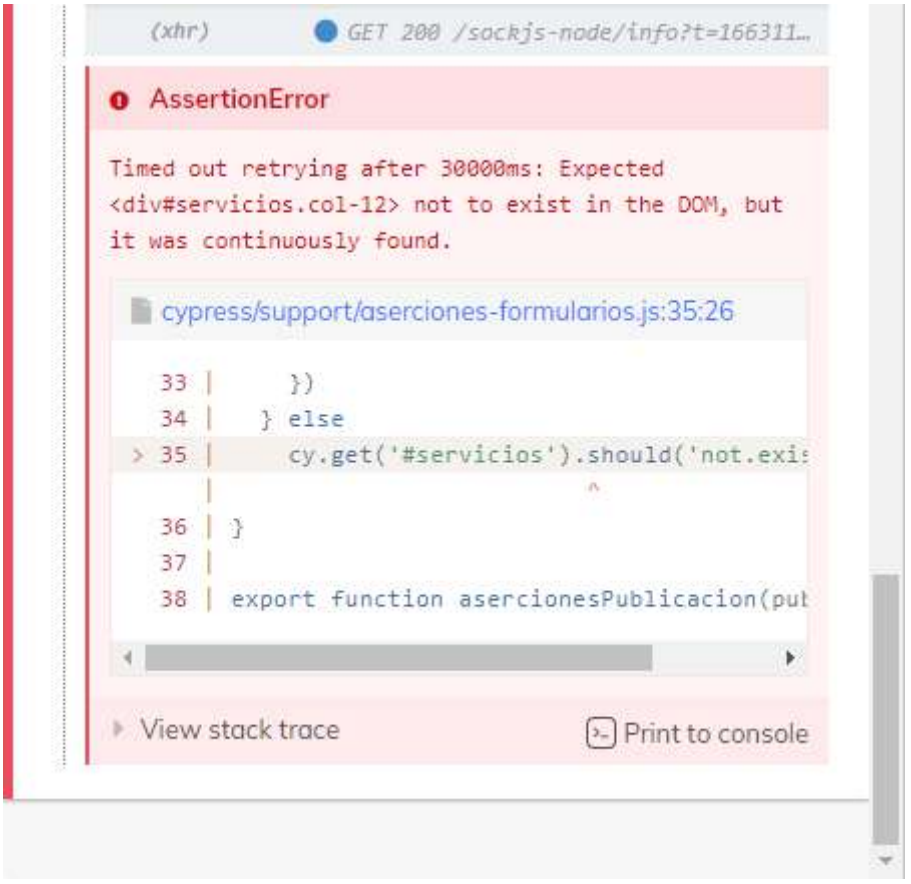


Fig. 18. Captura de pantalla del error detectado en la traza de la ejecución de TPS#5 de la funcionalidad nueva publicación.

En la Fig. 17 se puede visualizar que la ejecución del caso de prueba correspondiente a la TPS#5 (ver Fig. 13, 14 y 15) falló. Como se mencionó anteriormente, los casos de pruebas para la situación actual fueron codificados en un solo caso de prueba debido a que Cypress no acepta como precondición a otro caso de prueba. La desventaja de este enfoque es que no permite identificar exactamente qué caso de prueba de la Tabla 24 fue el que falló. Sin embargo, un beneficio de utilizarlo es que ofrece una traza de la ejecución de la prueba (ver Fig. 18). En dicha traza se puede observar el siguiente error: "*Expected <div#servicios.col-12> not to exist in the DOM, but it was continuously found*". Esto significa que no se esperaba que el elemento "*div*", que contiene los servicios en el detalle de publicación, exista en el DOM ya que en el momento de realizar la carga de la publicación no se ingresaron servicios. Por lo tanto, ya se tiene información para la corrección, es decir, puede que esté fallando el módulo del formulario de servicios o en la pantalla de detalle de publicación está mostrando los servicios cuando no debería.

Respecto a los resultados de prueba de las demás automatizaciones se pueden encontrar en el Anexo C.5. A modo de resumen, en TPS#1 y TPS#5 se detectaron errores, mientras que las otras situaciones cumplieron con lo esperado.

4.3. Contactar inmobiliaria por correo electrónico

4.3.1. Bases de pruebas para la funcionalidad contactar inmobiliaria por correo electrónico

Respecto a las bases de pruebas de la funcionalidad contactar inmobiliaria por correo electrónico, se consideraron:

- Diagrama de casos de uso (Fig. 4).
- Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26).
- Wireframe del formulario de contacto (Fig. 19).
- Validaciones del formulario (Tabla 27).

Tabla 26. Historia de usuario de funcionalidad contactar inmobiliaria por correo electrónico.

Historia de usuario	
Número: 5	**Usuario:** cliente
Nombre de historia: contactar inmobiliaria por correo electrónico	
Prioridad en negocio: alta	**Riesgo en desarrollo:** bajo
Puntos estimados: 6	**Iteración asignada:** 2
Programador responsable: Leonel Fernandez	
Descripción: quiero poder enviar un correo electrónico a la inmobiliaria con la información/consulta.	
Observaciones: para poder realizar correctamente la funcionalidad deben cumplirse ciertas validaciones en el formulario (ver Tabla 27).	
Criterios de aceptación: ● Solo es posible comunicarse con la inmobiliaria por medio de un correo electrónico cuando se cumplen las validaciones de los formularios. En caso contrario, el sistema debe emitir un mensaje de error y no debe permitir enviar el correo electrónico.	

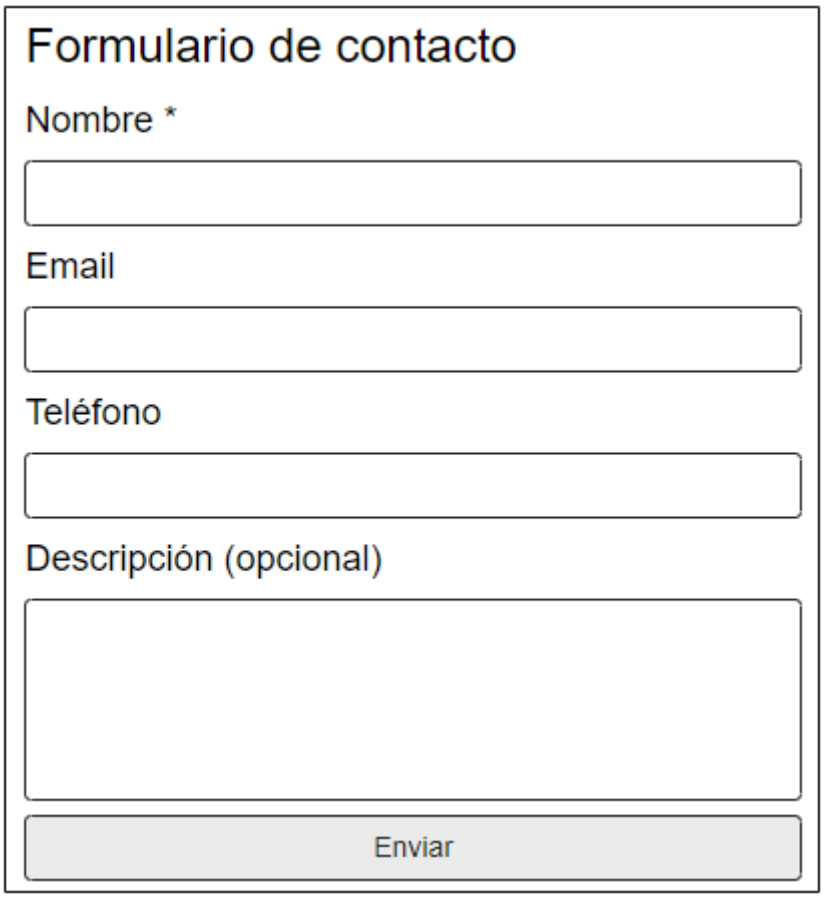


Fig. 19. Wireframe del formulario de contacto.

Tabla 27. Tabla de validaciones del formulario de contacto.

Formulario	Campo
Contacto	● Nombre ○ Campo obligatorio. ● E-mail ○ Debe tener formato de correo electrónico ○ Obligatorio en caso que el usuario no ingrese el número de teléfono. ● Teléfono ○ Debe tener formato de número de teléfono. ○ Obligatorio en caso que el usuario no ingrese el correo electrónico.

Según la historia de usuario, se espera que la funcionalidad permita enviar un correo electrónico al administrador de la inmobiliaria. Sin embargo, deben cumplirse ciertas validaciones. Por ejemplo, al observar la Tabla 27 se puede ver que los tres primeros campos de la inmobiliaria tienen validaciones como campos obligatorios y deben cumplir con ciertos formatos. Cabe destacar que la descripción no es obligatoria debido a un requisito del administrador de la inmobiliaria, con la finalidad de facilitar la funcionalidad al cliente en caso de que le sea más fácil ser contactado por otro medio.

4.3.2 A1 Definir los requisitos de prueba

La actividad actual consume:

- Objetivo de la prueba (definido al inicio del Capítulo 4).
- Declaración positiva de la situación particular de prueba de alto nivel (definido al inicio del Capítulo 4).
- Bases de pruebas (definidas en la Sección 4.3.1).

Finalmente, se elabora el requisito de la prueba (Tabla 28).

Tabla 28. Requisitos de prueba de la funcionalidad contactar inmobiliaria por correo electrónico.

Label: TR-System-ContactarInmobiliaria.
Statement: se requiere verificar la correctitud de la funcionalidad que permite al cliente contactar a la inmobiliaria por medio de un correo electrónico.
Testable Entity Phase: desarrollo.
Test level: sistema.
Completion Criteria: se debe desarrollar al menos un caso de prueba para cada una de las siguientes situaciones:

- El usuario cliente contacta a la inmobiliaria satisfactoriamente sólo considerando los campos obligatorios (ver Tabla 27).
- El usuario cliente contacta a la inmobiliaria satisfactoriamente rellenando todos los campos disponibles en el formulario.
- El usuario no cumple con la validación de un campo y no puede realizar su consulta. El sistema muestra un mensaje de error al respecto (ver Tabla 27).
- El usuario no cumple con la validación de un campo, corrige y logra contactarse con la inmobiliaria exitosamente (ver Tabla 27).
- El usuario intenta contactar a la inmobiliaria y no lo logra debido a que el servidor de correos no funciona correctamente. El sistema debe advertir al usuario que ocurrió un error.

Refers to Testable Entity: Inmobiliaria X.
Refers to Test Context Entities: servidor de correos electrónicos y correos de pruebas.

Al observar la Tabla 28, se pueden identificar cinco criterios en *completion criteria*. Al igual que "Nueva publicación" se debe verificar la correctitud teniendo en cuenta casos solo con campos obligatorios, todos los campos y validaciones del formulario. Respecto a las validaciones, el formulario debe permitir corregir los campos incorrectos. Otro escenario importante es verificar el estado de la aplicación en caso de que falle el servidor de correo electrónico.

4.3.3. A2 Diseñar las pruebas

Considerando el proceso de la Fig. 3, se debe realizar la actividad A2.1. En esta actividad se debe utilizar:

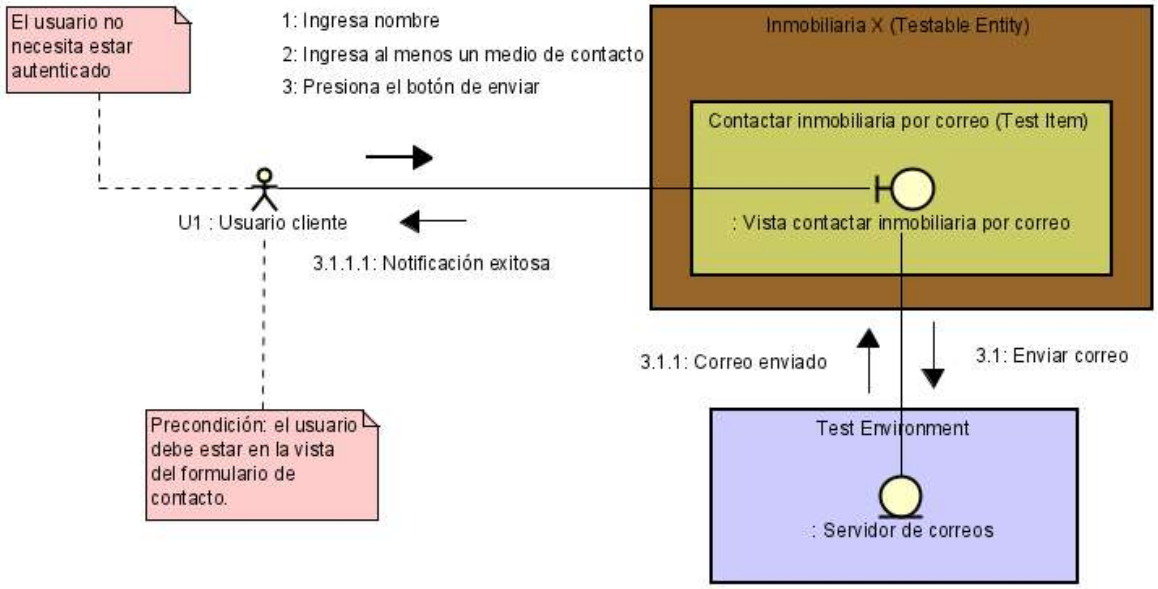
- Declaración positiva de la situación particular de prueba de alto nivel (establecido al inicio del Capítulo 4).
- Requisitos de prueba de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 28).
- Plantilla de SaSTMe (Tabla 1).

Al finalizar la actividad se obtienen las especificaciones del modelo de las situaciones particulares de prueba:

- Situación particular de prueba #1 (TPS#1, Tabla 29).
- Situación particular de prueba #2 (TPS#2, Tabla 30).
- Situación particular de prueba #3 (TPS#3, Tabla 31).
- Situación particular de prueba #4 (TPS#4, Tabla 32).
- Situación particular de prueba #5 (TPS#5, Tabla 33).

Tabla 29. Especificación de la TPS#1 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#1
-positive statement:
El usuario cliente se contacta con la inmobiliaria satisfactoriamente ingresando solo los campos obligatorios, es decir, nombre y, al menos, un medio de contacto.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario cliente debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con un correo emisor y otro receptor.
-postconditions: el correo electrónico fue enviado exitosamente al administrador de la inmobiliaria con los datos ingresados por el usuario. Además, el usuario puede identificar una notificación advirtiendo que la operación se realizó correctamente.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad contactar inmobiliaria por correo.
Test Target: funcionalidad contactar inmobiliaria por correo.

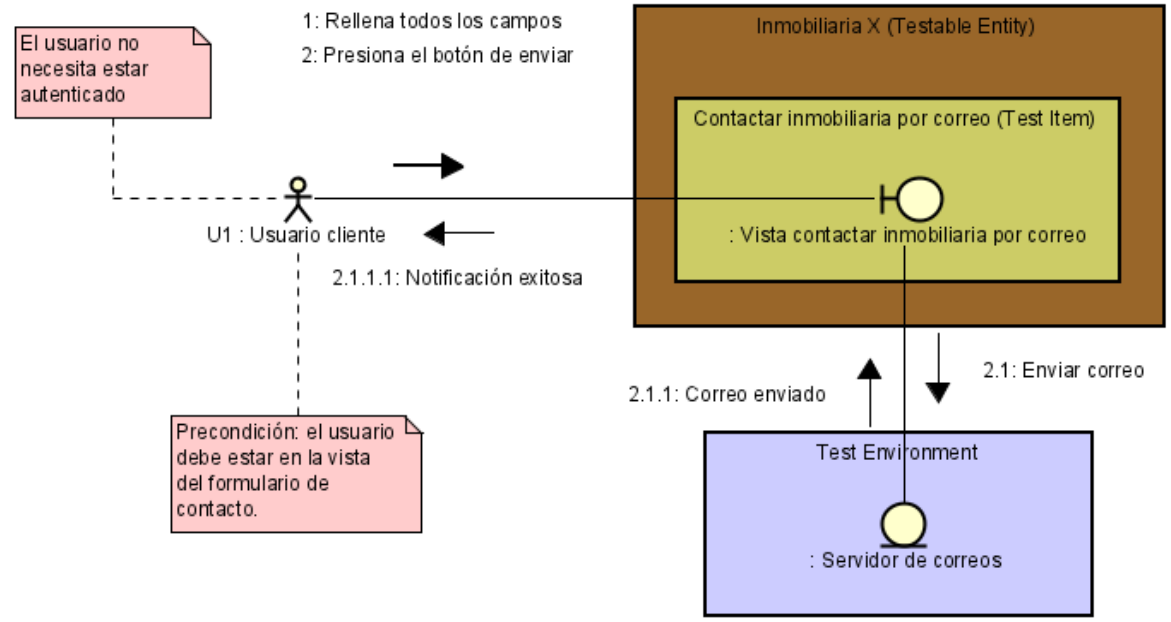
Test Context Entities: servidor de correos electrónicos. Tener en cuenta que este servidor debe tener configurado un correo emisor y otro receptor para realizar las pruebas.

-influences: el servidor de correo envía un correo electrónico al administrador de la inmobiliaria y notifica al sistema de la inmobiliaria que este correo fue enviado.

Test Requirement: TR-System-ContactarInmobiliaria (Tabla 28).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de contactar inmobiliaria por correo (Fig. 19). ● Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26). ● Tabla de validaciones del formulario de contacto (Tabla 27).

Tabla 30. Especificación de la TPS#2 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#2
-positive statement:
El usuario cliente se contacta con la inmobiliaria satisfactoriamente rellenando todos los campos disponibles del formulario.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario cliente debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con un correo emisor y otro receptor.
-postconditions: el correo electrónico fue enviado exitosamente al administrador de la inmobiliaria con los datos ingresados por el usuario y puede identificar una notificación advirtiendo que la operación se realizó correctamente.

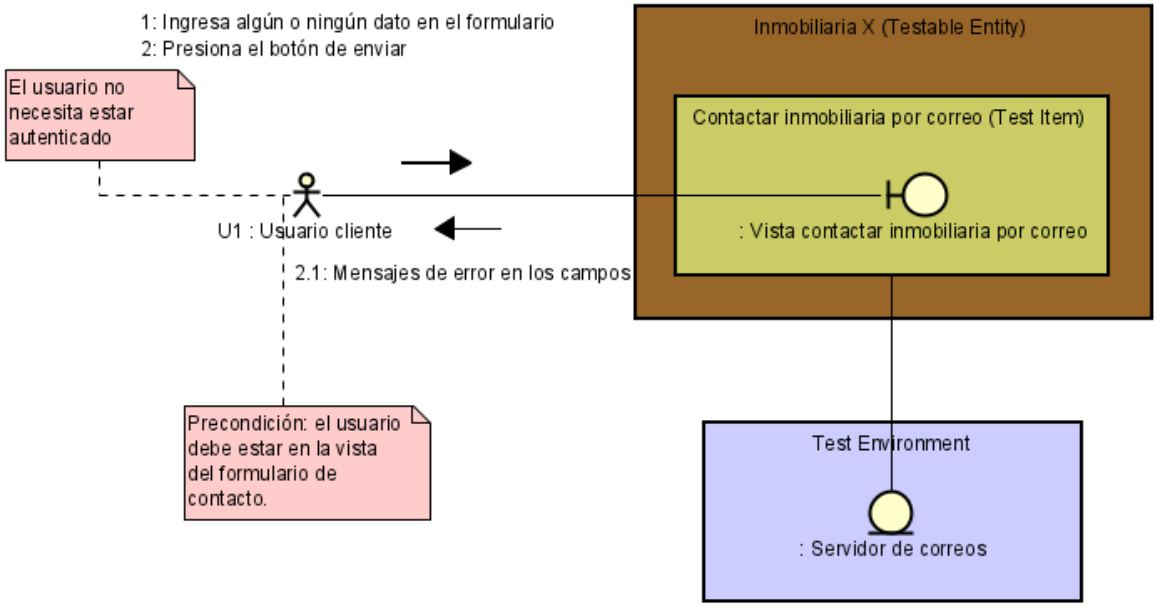
Testable Entity: Inmobiliaria X.
Test Item: funcionalidad contactar inmobiliaria por correo.
Test Target: funcionalidad contactar inmobiliaria por correo.

Test Context Entities: servidor de correos electrónicos. Tener en cuenta que este servidor debe tener configurado un correo emisor y otro receptor para realizar las pruebas.
-influences: el servidor de correo envía un correo electrónico al administrador de la inmobiliaria y notifica al sistema de la inmobiliaria que este correo fue enviado.

Test Requirement: TR-System-ContactarInmobiliaria (Tabla 28).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de contactar inmobiliaria por correo (Fig. 19). ● Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26). ● Tabla de validaciones del formulario de contacto (Tabla 27).

Tabla 31. Especificación de la TPS#3 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#3
-positive statement:
El usuario cliente no logra contactarse con la inmobiliaria debido a que no cumple con la validación del formulario.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario cliente debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con un correo emisor y otro receptor.
-postconditions: el usuario cliente no logró contactarse satisfactoriamente con la inmobiliaria y puede identificar mensajes de error en las entradas de formularios.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad contactar inmobiliaria por correo.
Test Target: funcionalidad contactar inmobiliaria por correo.

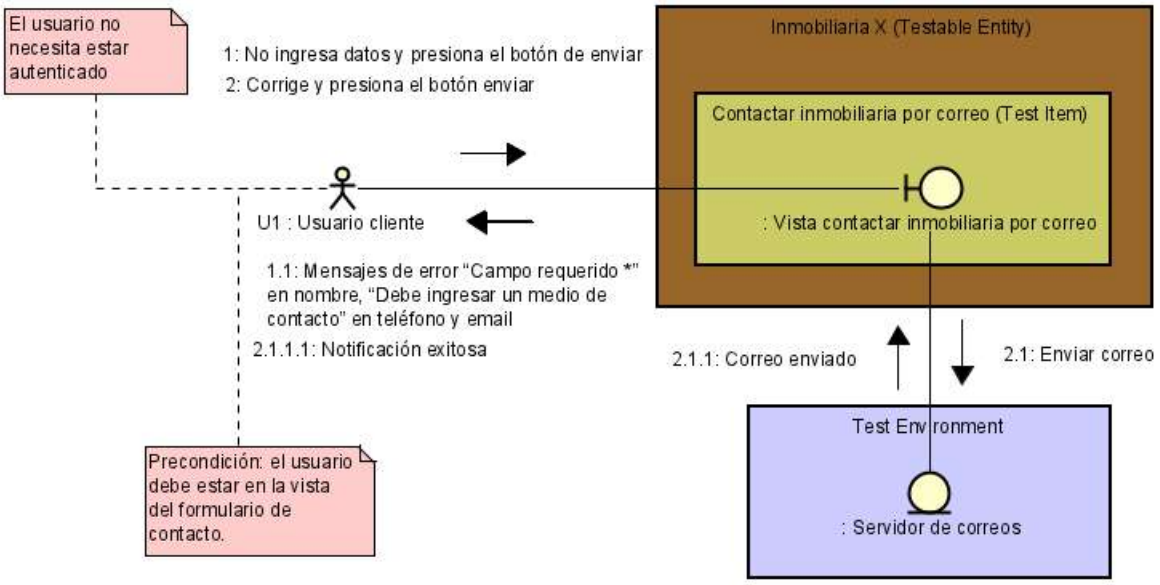
Test Context Entities: servidor de correos electrónicos. Tener en cuenta que este servidor debe tener configurado un correo emisor y otro receptor para realizar las pruebas.

-influences: el servidor de correo envía un correo electrónico al administrador de la inmobiliaria y notifica al sistema de la inmobiliaria que este correo fue enviado.

Test Requirement: TR-System-ContactarInmobiliaria (Tabla 28).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de contactar inmobiliaria por correo (Fig. 19). ● Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26). ● Tabla de validaciones del formulario de contacto (Tabla 27).

Tabla 32. Especificación de la TPS#4 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#4
-positive statement:
El usuario cliente no cumple con la validación del formulario, corrige y logra contactarse exitosamente con la inmobiliaria.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario cliente debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con un correo emisor y otro receptor.
-postconditions: el correo electrónico fue enviado exitosamente al administrador de la inmobiliaria con los datos ingresados por el usuario y puede identificar una notificación advirtiendo que la operación se realizó correctamente.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad contactar inmobiliaria por correo.
Test Target: funcionalidad contactar inmobiliaria por correo.

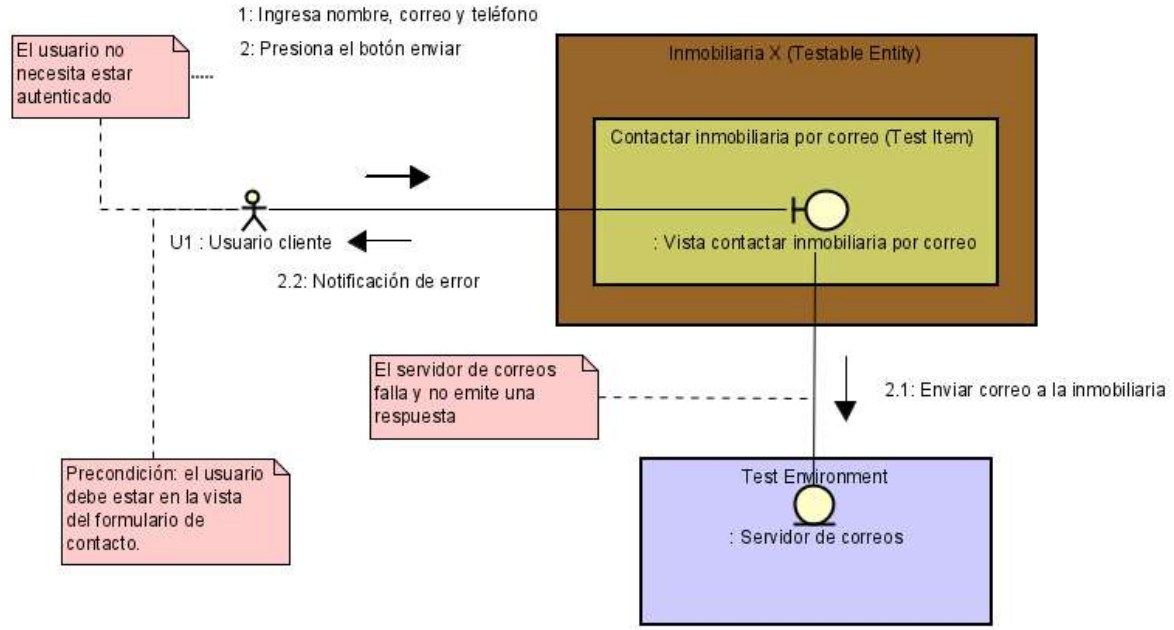
Test Context Entities: servidor de correos electrónicos. Tener en cuenta que este servidor debe tener configurado un correo emisor y otro receptor para realizar las pruebas.

-influences: el servidor de correo envía un correo electrónico al administrador de la inmobiliaria y notifica al sistema de la inmobiliaria que este correo fue enviado.

Test Requirement: TR-System-ContactarInmobiliaria (Tabla 28).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de contactar inmobiliaria por correo (Fig. 19). ● Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26). ● Tabla de validaciones del formulario de contacto (Tabla 27).

Tabla 33. Especificación de la TPS#5 de la funcionalidad nueva publicación.

Test Particular Situation:
TPS#5
-positive statement:
El servidor de correos falla en el momento que el usuario intenta contactarse con la inmobiliaria por medio de un correo electrónico.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: el usuario cliente debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con un correo emisor y otro receptor. Este no responde a las solicitudes que realiza el sistema debido a un problema propio del servidor, ajeno a la aplicación.
-postconditions: el usuario cliente no logra contactarse con la inmobiliaria y puede identificar una notificación advirtiendo que la operación no pudo realizarse.

Testable Entity: Inmobiliaria X
Test Item: funcionalidad contactar inmobiliaria por correo.
Test Target: funcionalidad contactar inmobiliaria por correo.

Test Context Entities: servidor de correos electrónicos. Tener en cuenta que este servidor debe tener configurado un correo emisor y otro receptor para realizar las pruebas.

-influences: el servidor de correo envía un correo electrónico al administrador de la inmobiliaria y notifica al sistema de la inmobiliaria que este correo fue enviado.

Test Requirement: TR-System-ContactarInmobiliaria (Tabla 28).
Test Basis: ● Diagrama de casos de uso (Fig. 4). ● Wireframe de contactar inmobiliaria por correo (Fig. 19). ● Historia de usuario de la funcionalidad contactar inmobiliaria por correo electrónico (Tabla 26). ● Tabla de validaciones del formulario de contacto (Tabla 27).

Una vez confeccionadas las especificaciones de las distintas situaciones se puede observar que se cubrieron todos los criterios de finalización de la tabla requisitos de prueba de la funcionalidad (ver Tabla 28). TPS#1 cubre el criterio "*El usuario cliente contacta a la inmobiliaria satisfactoriamente sólo considerando los campos obligatorios*". Luego, TPS#2 hace referencia a "*El usuario cliente contacta a la inmobiliaria satisfactoriamente rellenando todos los campos disponibles en los distintos formularios*". TPS#3 y TPS#4, están relacionados a los criterios de las validaciones, es decir, el que no cumple con la validación del formulario y, el otro, verificar que puedan corregirse los datos ingresados. Por otro lado, se encuentra TPS#5, que tiene como finalidad verificar el último criterio: "*El usuario intenta contactar a la inmobiliaria y no lo logra debido a que el servidor de correos no funciona correctamente. El sistema debe advertir al usuario que ocurrió un error*".

Notar que TPS#5 es una situación interesante, dado que se comprueba el estado de la aplicación en caso de que la entidad de contexto de prueba falle. En este caso, la entidad de contexto de prueba es un servidor de correos. Tener presente esta situación ya que se tomará como ejemplo para las siguientes secciones.

La siguiente subactividad de A2 es diseñar los casos de pruebas (A2.2). Para poder comenzar, se necesitan:

- ❖ Especificaciones del modelo de las situaciones particulares de prueba:
  - ➢ TPS#1 (Tabla 29).
  - ➢ TPS#2 (Tabla 30).
  - ➢ TPS#3 (Tabla 31).
  - ➢ TPS#4 (Tabla 32).

- ➢ TPS#5 (Tabla 33).

- ❖ Plantilla de SaSTMe (Tabla 1).

Al finalizar se obtienen los casos de pruebas de las distintas situaciones de pruebas:

- ❖ Casos de pruebas para TPS#1 (Tabla 34).
- ❖ Casos de pruebas para TPS#2 (Tabla 35).
- ❖ Casos de pruebas para TPS#3 (Tabla 36).
- ❖ Casos de pruebas para TPS#4 (Tabla 37).
- ❖ Casos de pruebas para TPS#5 (Tabla 38).

Tabla 34. Casos de pruebas de la TPS#1 de la funcionalidad contactar inmobiliaria por correo electrónico.

Test Cases:

TC#1.1
-input:
nombre: “Test FR#4 TPS#1 TC#1”,
correo: “test@gmail.com”
Presiona el botón enviar.

-expected result: el sistema emitió un correo a la inmobiliaria con los datos suministrados por el cliente.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación se realizó correctamente.

TC#1.2
-input:
nombre: “Test FR#4 TPS#1 TC#2”,
telefono: “12345678”
Presiona el botón enviar.

-expected result: el sistema emitió un correo a la inmobiliaria con los datos suministrados por el cliente.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además,tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación se realizó correctamente.

TC#1.3
-input:
nombre: “Test FR#4 TPS#1 TC#3”,
telefono: “12345678”,
correo: “test@gmail.com”
Presiona el botón enviar.

-expected result: el sistema emitió un correo a la inmobiliaria con los datos suministrados por el cliente.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación se realizó correctamente.

Tabla 35. Casos de pruebas de la TPS#2 de la funcionalidad contactar inmobiliaria por correo electrónico.

Test Cases:

TC#2.1
-input:
nombre: “Test FR#4 TPS#2 TC#1”,
correo: “test@gmail.com”,
telefono: “12345678”,
descripcion: “una descripcion Test FR#4 TPS#2 TC#1”
Presiona el botón enviar

-expected result: el sistema emitió un correo a la inmobiliaria con los datos suministrados por el cliente.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación se realizó correctamente.

Tabla 36. Casos de pruebas de la TPS#3 de la funcionalidad contactar inmobiliaria por correo electrónico.

Test Cases:

TC#3.1
-input:
presiona el botón enviar (sin cargar ningún campo).

-expected result: el sistema no permite enviar el correo ya que el usuario no cumple con las validaciones del formulario de contacto.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario puede identificar mensajes de error “Campo requerido *” en nombre, “Debe ingresar un medio de contacto” en teléfono y en email.

TC#3.2
-input:
nombre: “Test FR#4 TPS#3 TC#2”,
correo: “test.com”,
telefono: “123.45678”
Presiona el botón enviar.

-expected result: el sistema no permite enviar el correo ya que el usuario no cumple con las validaciones del formulario de contacto.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente no logró contactarse satisfactoriamente con la inmobiliaria y puede identificar mensajes de error “No corresponde a un correo electrónico” en email y “Debe estar formado por dígitos, "-", "+" y "()"” en teléfono.

Tabla 37. Casos de pruebas de la TPS#4 de la funcionalidad contactar inmobiliaria por correo electrónico.

Test Cases:

TC#4.1
-input: presiona el botón enviar (sin cargar ningún campo).

-expected result: el sistema no permite enviar el correo ya que el usuario no cumple con las validaciones del formulario de contacto.

-preconditions: el usuario debe estar en la vista del formulario de contacto. Además, tiene que haber un servidor de correos electrónicos configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar mensajes de error “Campo requerido *” en nombre, “Debe ingresar un medio de contacto” en teléfono y en email.

TC#4.2
-input:
nombre: “Test FR#4 TPS#4 TC#2”,
correo: “test@hotmail.com”
Presiona el botón enviar.

-expected result: el sistema emitió un correo a la inmobiliaria con los datos suministrados por el cliente.

-preconditions:TC#4.1.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación se realizó correctamente.

Tabla 38. Casos de pruebas de la TPS#5 de la funcionalidad contactar inmobiliaria por correo electrónico.

Test Cases:

TC#5.1
-input:
nombre: “Test FR#4 TPS#5 TC#1”,
correo: “test@gmail.com”,
telefono: “12345678”
Presiona el botón enviar.

-expected result: el sistema no logra enviar el correo electrónico con los datos suministrados por el cliente debido a que el servidor de correos falló.

-preconditions: el usuario debe estar en la vista del formulario de contacto. El servidor de correos no responde a las solicitudes que realiza el sistema y tiene que estar configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com.

-postconditions: el usuario cliente puede identificar una notificación advirtiendo que la operación no pudo realizarse.

La tabla relacionada a los casos de prueba de TPS#1 (ver Tabla 34) contiene tres casos de prueba. La idea es verificar que la funcionalidad permite enviar el correo electrónico solo con datos obligatorios. Por ejemplo, agregar nombre y correo electrónico, agregar nombre y teléfono y, por último, agregar nombre, correo electrónico y teléfono.

Respecto a TPS#2, solo se confeccionó un caso de prueba (ver Tabla 35) para poder verificar la situación. Se espera que la funcionalidad envíe un correo electrónico al ingresar el nombre, correo electrónico, teléfono y descripción.

Luego, en TPS#3 se verificó que se cumplan las validaciones del formulario (ver Tabla 36). La idea de estas validaciones es que el administrador de la inmobiliaria tenga un medio de contacto y pueda identificar a la persona para poder contactarla. Además, hay validaciones de formato del correo electrónico y teléfono ingresado.

En los casos de prueba de TPS#4 (ver Tabla 37), se puede observar que verifican que el formulario permita corregir los datos luego de que el usuario cliente ingrese un dato inválido. En TC#4.1, el usuario intenta contactarse sin ingresar datos por lo que el sistema notifica los errores. Luego, TC#4.2 utiliza como precondición a

TC#4.1 e ingresa nombre y correo electrónico. Finalmente, logra contactarse satisfactoriamente con la inmobiliaria.

Por último, en el caso de prueba de TPS#5 (ver Tabla 38) comprueba el estado de la aplicación en caso de que el servidor de correos electrónicos falle. Es un caso de prueba importante, dado que se espera que el sistema sea capaz de notificar al usuario cliente que hubo un error al contactar a la inmobiliaria.

Para comenzar con la actividad A2.3 se deben consumir:

- Especificaciones de los modelos de las situaciones particulares de pruebas:
  - TPS#1 (Tabla 29).
  - TPS#2 (Tabla 30).
  - TPS#3 (Tabla 31).
  - TPS#4 (Tabla 32).
  - TPS#5 (Tabla 33).
- Casos de pruebas:
  - Casos de pruebas para TPS#1 (Tabla 34).
  - Casos de pruebas para TPS#2 (Tabla 35).
  - Casos de pruebas para TPS#3 (Tabla 36).
  - Casos de pruebas para TPS#4 (Tabla 37).
  - Casos de pruebas para TPS#5 (Tabla 38).

Los artefactos resultantes de la actividad actual son:

- Especificaciones de los procedimientos de realización.
  - Automatización de casos de pruebas correspondiente a TPS#1 (ver Fig. D2 y D3 en Anexo D.2).
  - Automatización de casos de pruebas correspondiente a TPS#2 (ver Fig. D4 en Anexo D.2).
  - Automatización de casos de pruebas correspondiente a TPS#3 (ver Fig. D5 en Anexo D.2).

➢ Automatización de casos de pruebas correspondiente a TPS#4 (ver Fig. D6 en Anexo D.2).

➢ Automatización de casos de pruebas correspondiente a TPS#5 (Fig. 20).

❖ Requisitos del entorno de prueba (Tabla 39).

```
describe('TPS#5', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('contactar-inmobiliaria').then((dataSet) => this.dataSet = dataSet);
    cy.intercept(
      'POST',
      '/api/ContactoMail/SendMail',
      { statusCode: 500 }
    ).as('reqError')
    /* Precondicion: el usuario debe estar en la vista del formulario de contacto.
     El servidor de correos no responde a las solicitudes que realiza el sistema
     y tiene que estar configurado con los siguientes correos: contactotest@test.com y
    inmobiliariatest@test.com */
    cy.visit('/contacto');
    cy.get('#titulo-contacto').should('contain', 'Formulario de Contacto');
  })

  it('Usuario contacta a la inmobiliaria y el servidor de correos falla', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[6]);
    cy.get("#enviar").click()

    cy.wait('@reqError')
      .then(interception => {
        /* Resultado esperado: el sistema no logra enviar el correo electrónico con los
        datos suministrados por el cliente debido a que el servidor de correos falló. */
        expect(interception.response.statusCode).to.equal(500);
      });

    //Assert (Afirmar)
    /* Postcondicion: el usuario cliente puede identificar una notificación advirtiendo
     que la operación no pudo realizarse. */
    cy.get(".swal2-error").should("exist");
  })
})
```

Fig. 20. Especificación de procedimiento de realización de TPS#5 de la funcionalidad contactar inmobiliaria por correo electrónico.

Tabla 39. Requisitos del entorno de prueba para la funcionalidad contactar inmobiliaria por correo electrónico.

Entidad de contexto de prueba	**Requisitos del entorno de prueba**
Servidor de correos	Debe tener configurado dos correos electrónicos: contactotest@test.com e inmobiliariatest@test.com.

Tener en cuenta que, como las demás funcionalidades, se creó un conjunto de datos de entradas que serán de utilidad posteriormente en las automatizaciones de los casos de pruebas. Este se puede visualizar en la Fig. D1 del Anexo D.1.

Como se mencionó anteriormente, se automatizaron cinco situaciones de pruebas. En esta sección se va a hacer foco en el código de TPS#5 (Fig. 20) debido a que es un caso especial donde hay que provocar que la entidad de contexto, en este caso el servidor de correos, falle. De este modo, se puede verificar la respuesta del sistema ante esta situación. Por otro lado, los códigos de las situaciones particulares restantes pueden visualizarse en el Anexo D.2.

Al observar la implementación de TPS#5, se puede ver que en la porción de código de Arrange se obtienen los datos de entradas correspondientes. Posteriormente, se ejecuta la función "*intercept*". Notar la importancia de esta función ya que nos permite simular una respuesta del servidor de correos. En este caso, se crea un interceptor que al identificar una solicitud de tipo "*POST*" y con la URL "*/api/ContactoMail/SendMail*" debe responder "*statusCode: 500*". Luego, al interceptor se le da el siguiente alias "*reqError*".

Antes de comenzar a interactuar con la funcionalidad, se debe cumplir con la precondición "*el usuario debe estar en la vista del formulario de contacto. El servidor de correos no responde a las solicitudes que realiza el sistema y tiene que estar configurado con los siguientes correos: contactotest@test.com e inmobiliariatest@test.com*". Para cumplir que el usuario se encuentre en el formulario de contacto en el código se realiza una aserción con la URL "*/contacto*" y, además, debe existir el título "*Formulario de contacto*". Respecto a la precondición relacionada al servidor de correos, se debe configurar la entidad de contexto de prueba con los requisitos mencionados en la Tabla 39.

Una vez finalizada la etapa de preparación, se puede avanzar a la porción Act. En esta sección solo se ejecutan dos acciones. Primero, se realiza la carga del formulario con los datos de entrada correspondiente y, luego, se presiona el botón enviar.

En la etapa Assert, se verifica que el resultado esperado y la precondición se cumplan. El caso de prueba menciona como resultado esperado lo siguiente: "el sistema no logra enviar el correo electrónico con los datos suministrados por el cliente debido a que el servidor de correos falló". Por lo tanto, se debe hacer uso

del interceptor creado al comienzo de la automatización. Para ello se ejecuta "*cy.wait(reqError)*" que tiene la finalidad de escuchar las solicitudes que realiza la aplicación y al encontrar que se envió el correo electrónico entonces verifica que el resultado sea un error 500.

Respecto a la postcondición, la implementación comprueba que sea visible una notificación de error. De esta manera logra verificar la postcondición "*el usuario cliente puede identificar una notificación advirtiendo que la operación no pudo realizarse*" y finaliza el caso de prueba.

4.3.4. A3 Establecer entorno de prueba

Para comenzar A3, se necesitan los siguientes artefactos:

- Entidades de contexto de prueba (Tabla 39).
- Requisitos del entorno de prueba (Tabla 39).

El resultado de la actividad es configurar el servidor de correos con dos correos electrónicos. El correo contactotest@test.com es el encargado de enviar un correo electrónico con los datos suministrados por el cliente. Mientras que el otro, es el receptor del correo. Notar que no se agregaron capturas de pantallas con la entidad de contexto de prueba configurada debido a la confidencialidad.

4.3.5. A4 Realizar pruebas dinámicas

La actividad A4 utiliza los siguientes artefactos:

- Entidades de contexto de prueba configuradas (configurada en la Sección 4.3.4)
- Especificaciones de procedimientos de realización:
  - Automatización de TPS#1 (ver Fig. D2 y D3 en Anexo D.2).
  - Automatización de TPS#2 (ver Fig. D4 en Anexo D.2).
  - Automatización de TPS#3 (ver Fig. D5 en Anexo D.2).
  - Automatización de TPS#4 (ver Fig. D6 en Anexo D.2).
  - Automatización de TPS#5 (Fig. 20).
- Entidad verificable, es decir, la aplicación web de la inmobiliaria.

Al finalizar la actividad se obtienen los resultados de las pruebas:

- Resultado de la prueba de TPS#1 (ver Fig. D7 en Anexo D.3).
- Resultado de la prueba de TPS#2 (ver Fig. D8 en Anexo D.3).
- Resultado de la prueba de TPS#3 (ver Fig. D9 en Anexo D.3).
- Resultado de la prueba de TPS#4 (ver Fig. D10 en Anexo D.3).
- Resultado de la prueba de TPS#5 (Fig. 21).

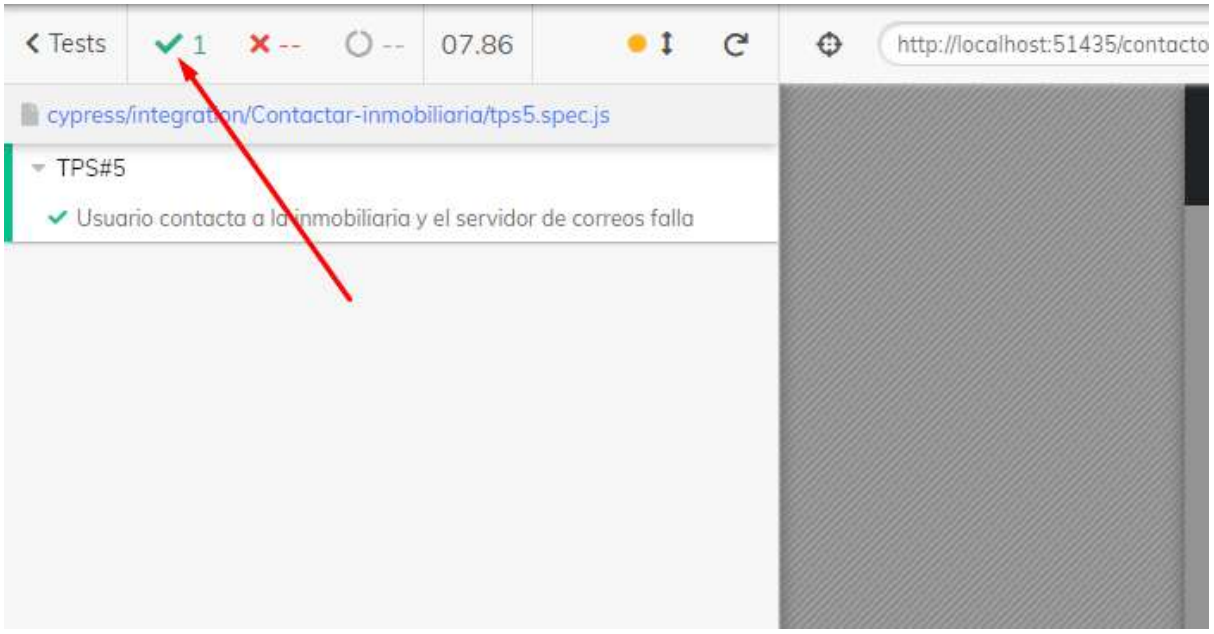


Fig. 21. Captura de pantalla de la ejecución de TPS#5 de la funcionalidad contactar inmobiliaria por correo electrónico.

Respecto al resultado de las ejecuciones de las distintas situaciones de prueba (ver Anexo D.3 y Fig. 21) se puede observar que todas fueron satisfactorias. En la Fig. 21 se puede identificar un caso de prueba y una flecha roja que indica el resultado de la ejecución de la situación particular de prueba. De esta forma finaliza la actividad A4 de la funcionalidad contactar inmobiliaria por correo electrónico.

4.4. A5 Analizar los resultados de las pruebas

Según la Fig. 3, la actividad A5 es la última del proceso. Los artefactos necesarios para comenzar son:

- Bases de pruebas:
  - Bases de pruebas de iniciar sesión (ver Sección 4.1.1).

- ➢ Bases de pruebas de nueva publicación (ver Sección 4.2.1).
- ➢ Bases de pruebas de contactar inmobiliaria por correo electrónico (ver Sección 4.3.1).
- ➢ Bases de pruebas de modificar publicación (ver Anexo E.1).

❖ Requisitos de pruebas:

- ➢ Requisito de prueba de iniciar sesión (Tabla 3).
- ➢ Requisito de prueba de nueva publicación (Tabla 14).
- ➢ Requisito de prueba de contactar inmobiliaria (Tabla 28).
- ➢ Requisito de prueba de iniciar sesión (ver Tabla E3 en Anexo E.2).

❖ Casos de pruebas:

- ➢ Casos de prueba de iniciar sesión (ver Sección 4.1.3).
- ➢ Casos de prueba de nueva publicación (ver Sección 4.2.3).
- ➢ Casos de prueba de contactar inmobiliaria por correo electrónico (ver Sección 4.3.3).
- ➢ Casos de prueba de modificar publicación (ver Anexo E.3).

❖ Resultados de pruebas:

- ➢ Resultados de pruebas de iniciar sesión (ver Sección 4.1.5).
- ➢ Resultados de pruebas de nueva publicación (ver Sección 4.2.5).
- ➢ Resultados de pruebas de contactar inmobiliaria por correo electrónico (ver Sección 4.3.5).
- ➢ Resultados de pruebas de modificar publicación (ver Anexo E.8).

❖ Especificación de las metas de necesidad de información de la prueba (Tabla 40).

Tabla 40. Especificación de las metas de necesidad de información de la prueba.

Label: test-inmobiliaria
Statement: analizar la correctitud de las funcionalidades seleccionadas de la aplicación web de la inmobiliaria
Purpose: analizar

La actividad genera el reporte de conclusión de la prueba en la Tabla 41.

Tabla 41. Reporte de la conclusión de la prueba.

Name: RCP#1
Version: 1.0
Description: ● Funcionalidad iniciar sesión: TPS#1, TPS#2 y TPS#3 pasaron satisfactoriamente las pruebas. ● Funcionalidad nueva publicación: TPS#2, TPS#3 y TPS#4 pasaron satisfactoriamente las pruebas. Por otro lado, TPS#1 y TPS#5 no lograron pasar debido a un error en la lógica de la aplicación. Finalmente, se solucionaron y se ejecutaron nuevamente de forma satisfactoria. ● Funcionalidad contactar inmobiliaria por correo electrónico: TPS#1, TPS#2, TPS#3, TPS#4 y TPS#5 pasaron satisfactoriamente. ● Funcionalidad modificar publicación: TPS#1, TPS#2, TPS#3, TPS#4 y TPS#5 pasaron satisfactoriamente las pruebas. Para concluir, se verificó la correctitud de todas las funcionalidades seleccionadas. Además, se realizó la cobertura del 100% de los criterios de finalización de los requisitos de pruebas. Respecto a los resultados de pruebas, aproximadamente el 88% de las situaciones particulares de pruebas pasaron satisfactoriamente y, después de solucionar los errores, la correctitud de las situaciones particulares de prueba es 100%.

Para realizar el reporte se debe tener presente la especificación de las metas de necesidad de información de la prueba (ver Tabla 40). Esta establece: "*analizar la correctitud de las funcionalidades seleccionadas de la aplicación web de la inmobiliaria*".

Al observar los casos de usos de la Fig. 4 contenida en las bases de pruebas, se verificó la correctitud al 100% de las funcionalidades. Además, como se mencionó en los capítulos de la funcionalidades, se cubrieron todos los criterios de finalización de requisitos de pruebas con al menos un caso de prueba, obteniendo un 100% de cobertura.

Respecto a los resultados esperados de los casos de pruebas con los resultados actuales, o en otras palabras el resultado de las pruebas automatizadas, se detectó que dos situaciones de prueba no cumplieron con lo esperado. Por lo tanto, el 88% de las situaciones particulares de pruebas fueron satisfactorias. Sin embargo, se realizó la corrección de los errores detectados (ver Anexo C.6). En consecuencia, el 100% de las pruebas pasaron exitosamente.

Finalmente, con la información obtenida anteriormente se puede concluir que todos los requisitos de pruebas cumplieron con el objetivo de la prueba.

5. Trabajos relacionados

Existen diferentes propuestas que buscan realizar testing considerando el contexto. A continuación se comentan algunas que se encuentran en la literatura y se indica en qué se diferencia SaST de las mismas.

Un trabajo relacionado reciente está documentado en [13]. Los autores proponen utilizar CATS#, una evolución de la técnica de diseño CATS (Context-Aware Test Suite). Su técnica está destinada a dar soporte a los ingenieros de software con la especificación de los casos de pruebas conscientes del contexto. Al utilizar esta técnica, los casos de prueba pueden capturar variables de contexto y condiciones variables. Sin embargo, su enfoque no captura explícitamente cuáles son las entidades de contexto como lo hace la estrategia SaST, sino que solo captura algunas propiedades de las entidades de contexto. Además, CATS# no incluye una base conceptual.

En [14] se presenta un enfoque basado en pruebas de escenario usado formalmente para validar los diagramas de clases UML. Ellos proponen un conjunto de pasos, es decir el proceso, especificado en lenguaje natural. También, definen tres tipos de escenarios: consistente, inconsistente y de clasificación. En resumen, la idea propuesta es traducir el diagrama de clases UML a OWL y verificar su consistencia. Luego, si el modelo es consistente, el siguiente paso es producir y verificar escenarios que representen los requisitos del negocio. En esta investigación, el autor utiliza ontologías, pero solamente para propósito de verificación y no para enriquecimiento de semántica de conceptos de prueba como en el caso de la estrategia SaST. Además, esta metodología es solamente aplicada a actividades de pruebas estáticas.

Otros trabajos como [15] y [16], discuten enfoques para probar sistemas de software consciente del contexto. Tener en cuenta que la estrategia SaST es más amplia, en consecuencia, puede ser aplicada a estos tipos de sistemas de software y a otros también. Asimismo, todos ellos concuerdan con la siguiente definición de

contexto: "Cualquier información que pueda ser utilizada para caracterizar la situación de entidades (es decir, ya sea una persona, lugar u objeto) que se consideren relevantes para la interacción entre un usuario y una aplicación, incluida el usuario y la propia aplicación" [17]. Esta definición es similar a la definición utilizada en la estrategia SaST. Desde el punto de vista semántico de SituationCO, las entidades objetivo rodeadas e influenciadas por entidades de contexto en una situación particular tienen la semántica de contexto.

Según ISO 29119-2 [4], el proceso de pruebas dinámicas debe tener cuatro actividades principales: diseño e implementación de pruebas, configuración y mantenimiento del entorno de pruebas, ejecución de las pruebas e informes de incidentes de pruebas. Por otro lado, ISTQB [18] también considera cuatro actividades principales: análisis de prueba, diseño de prueba, implementación de prueba y ejecución de prueba. Como se observa en la Fig. 3, SaSTPro considera las actividades mencionadas y, además, agrega una subactividad llamada "Especificar situaciones particulares de prueba (A2.1)" que no está incluida en las propuestas citadas.

Por último, se encuentra la tesis de grado realizada por el Ing. Marcos Eleicegui [19], donde aplica la estrategia de prueba SaST a una aplicación que se encarga de gestionar contratos inteligentes y pagos digitales con el fin de diseñar casos de pruebas para las distintas situaciones. Es importante mencionar que dicha tesina está limitada sólo al diseño de las pruebas, y no a su automatización, ejecución y análisis, como se realiza en el presente trabajo.

6. Conclusiones

6.1 Conclusiones y resultados

Para concluir, la estrategia SaST se aplicó a una aplicación web de una inmobiliaria que permite a un administrador poder gestionar publicaciones para que los clientes puedan visualizar los inmuebles disponibles para alquilar y comprar. El objetivo era probar la aplicación para verificar el correcto funcionamiento de las funcionalidades de iniciar sesión, nueva publicación, modificar publicación y contactar inmobiliaria por correo electrónico. Al finalizar, el resultado que se obtuvo fue lo esperado a excepción de la funcionalidad de nueva publicación donde se pudo identificar un error en el paso de servicios. Por lo tanto, se encontró la solución y se ejecutaron nuevamente las pruebas automatizadas obteniendo un resultado satisfactorio.

6.2 Apreciaciones personales

Considero que SaST es una estrategia que se encuentra bien definida por un marco teórico integrado por ontologías y definiciones de términos, un método simple definido a través de una plantilla y un proceso con las actividades a seguir identificando los artefactos generados paso a paso. Esta permitió realizar la verificación de la correctitud de un conjunto de funcionalidades desde un nivel de sistema de forma rápida y sencilla, permitiendo identificar ya sea la aplicación a probar como las entidades de contexto.

Por otro lado, se detectó que TestTDO no hace diferencia entre las definiciones de resultado esperado y postcondición, como por ejemplo el glosario del ISTQB [18]. Cabe destacar que no es algo que perjudique a la estrategia, solo puede provocar incertidumbres al momento de diseñar el estado posterior de la aplicación luego de ejecutar un caso de prueba.

6.3. Trabajos futuros

Como trabajo futuro, se va a aplicar la estrategia SaST al actual sistema en un entorno de prueba, debido a que cuando se comenzó el presente proyecto la aplicación web estaba en desarrollo, y por lo tanto la versión actual que se encuentra en producción no es la misma que se utilizó para el caso de prueba presentado en esta tesina.

Además, se pretende aplicar las prácticas de CI/CD (integración continua y entrega continua) que permiten automatizar el proceso de construcción, pruebas y despliegue. De esta forma, se podrá aumentar la detección de defectos en etapas tempranas y realizar entregas del producto con mayor frecuencia. Por último, se está analizando utilizar la funcionalidad de paralelización de pruebas que ofrece el framework de Cypress con el objetivo de disminuir el tiempo del proceso de ejecución de las pruebas. Como resultado de aplicar estas prácticas se busca disminuir el tiempo de trabajo manual en la prueba del software ante los cambios en las funcionalidades y aumentar la calidad del producto.

Bibliografía

[1] R.S. Pressman, Software engineering. A practitioner's approach , 7th ed. New York: McGraw-Hill, 2010.

[2] Software and systems engineering – Software Testing – Part 1: Concepts and definitions, ISO/IEC/IEEE 29119-1, 2013.

[3] Software and systems engineering – Software Testing – Part 4: Test techniques, ISO/IEC/IEEE 29119-4, 2015.

[4] Software and systems engineering – Software Testing – Part 2: Test processes, ISO/IEC/IEEE 29119-2, 2013.

[5] L. Olsina and P. Becker, "Family of strategies for different evaluation purposes," in XX CIbSE' 17, Curran Associates, 2017, pp. 221–234.

[6] G. Tebes, D. Peppino, P. Becker, and L. Olsina, "A Situation-aware Scenario-based Testing Strategy," in SCCC 2021 - 40th International Conference of the Chilean Computer Science Society, to appear in IEEE Xplore, (held virtually, Nov. 2021), 2021, pp. 1–8.

[7] JavaScript End to End Testing Framework | cypress.io testing tools. Accessed: Feb. 2, 2022. [Online]. Available: https://cypress.io/

[8] L. Olsina, "Analyzing the Usefulness of ThingFO as a Foundational Ontology for Sciences," in Proceedings of ASSE'20, Argentine Symposium on Software Engineering, 49 JAIIO, 2020, pp. 172–191.

[9] L. Olsina, "Applicability of a Foundational Ontology to Semantically Enrich the Core and Domain Ontologies," in 13th International Conference on Knowledge Engineering and Ontology Development (KEOD), IC3K, pp. 111–119, 2021.

[10] G. Tebes, L. Olsina, D. Peppino, and P. Becker, "TestTDO: A Top-Domain Software Testing Ontology," in XXIII CIbSE' 20, Curran Associates, 2020, pp. 364–377.

[11] B. Curtis, M. Kellner, and J. Over, "Process Modeling," Communication of ACM, vol. 35, no. 9, pp. 75–90, 1992, [Online]. Available: https://doi.org/10.1145/130994.130998.

[12] G. Tebes, L. Olsina, D. Peppino, and P. Becker, "Specifying and Analyzing a Software Testing Ontology at the Top-Domain Ontological Level," Journal of Computer Science & Technology, vol. 21, pp. 126-145, 2021.

[13] A. C. De Souza Doreste and G. H. Travassos, "Towards Supporting the Specification of Context-Aware Software System Test Cases," in XXIII CIbSE' 20, Curran Associates, 2020, pp. 356–363.

[14] H. F. Harmse, K. Britz, A. Gerber, and D. Moodley, "Scenario testing using formal ontologies," in CEUR Workshop Proceedings, 2014, vol. 1301, [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84916230683&partnerID=40&md5=729d79762557cd6ad4bf631688206579.

[15] M. A. Mehmood and M. N. A. Khan, "An Automated Functional Testing Framework for Context-Aware Applications," IEEE Access, vol. 6, pp. 46568–46583, Aug. 2018, doi: 10.1109/ACCESS.2018.2865213.

[16] Z. Wang, S. Elbaum, and D. S. Rosenblum, "Automated Generation of Context-Aware Tests," in 29th International Conference on Software Engineering (ICSE'07), 2007, pp. 406–415, doi: 10.1109/ICSE.2007.18.

[17] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a Better Understanding of Context and Context- Awareness," in Gellersen HW. (eds) Handheld and Ubiquitous Computing, 1999, doi: https://doi.org/10.1007/3-540-48157-5_29.

[18] ISTQB, "International Software Testing Qualifications Board, Certified Tester – Foundation Level Syllabus." 2018, [Online]. Available: https://www.istqb.org/downloads/category/2-foundation-level-documents.html.

[19] Eleicegui, M. (2022). Aplicación práctica de SaST, una estrategia de prueba basada en escenarios y consciente del contexto [Tesis de grado no publicada]. Universidad Nacional de La Pampa.

Anexo A: Funciones de soporte para las situaciones particulares de prueba

```
Cypress.Commands.add('login', (nombreUsuario, contrasena) => {
  //login
  if (nombreUsuario)
    cy.get('#username').clear().type(nombreUsuario);
  if (contrasena)
    cy.get('#password').clear().type(contrasena);

  cy.get('#boton-login').click()
});

Cypress.Commands.add('selectInput', (selectorId, value) => {
  if (value)
    cy.get(selectorId).select(value);
});

Cypress.Commands.add('typeInput', (selectorId, value) => {
  if (value || value == 0)
    cy.get(selectorId).clear().type(value);
});

Cypress.Commands.add('invokeInput', (selectorId, value) => {
  if (value)
    cy.get(selectorId).clear().invoke('val', value).trigger('input');
});

Cypress.Commands.add('checkInput', (selectorId, value) => {
  if (value)
    cy.get(selectorId).check();
  else if (value == false)  //Evitar caso undefined
    cy.get(selectorId).uncheck();
});


Cypress.Commands.add('waitForRequest', (url) => {
  cy.intercept(url).as('request');
  cy.wait('@request');
});
```

Fig. A1. Funciones de soporte para las situaciones particulares de prueba.

```
Cypress.Commands.add('disableUncaughtException', () => {
  Cypress.on('uncaught:exception', (err, runnable) => {
    return false
  })
})

Cypress.Commands.add('typeLogin', () => {
  cy.visit('/login')
  //login
  cy.get('#username').type('test')
  cy.get('#password').type('test')
  cy.get('#button-login').click()
})
```

Fig. A2. Funciones de soporte para las situaciones particulares de prueba.

```
export function cargarFormularioPublicacion(publicacion) {
  cy.typeInput('#titulo', publicacion.titulo);
  cy.selectInput('#operacion', publicacion.tipoOperacion);
  cy.typeInput('#precio', publicacion.precio);
  cy.selectInput('#tipoMoneda', publicacion.tipoMoneda);
  cy.checkInput('#permuta', publicacion.permuta);
  cy.checkInput('#destacada', publicacion.destacada);
  cy.checkInput('#credito', publicacion.credito);
  cy.invokeInput('#descripcion', publicacion.descripcion);
}

export function cargarFormularioFotos(fotos) {
  //Caso para eliminar fotos en la pantalla de modificar
  eliminarFotos();
  if (fotos.fotoDestacada)
    cy.get('#fotoDestacada').attachFile(fotos.fotoDestacada);
  if (fotos.fotosExposicion)
    fotos.fotosExposicion.forEach(filePath => cy.get('#foto').attachFile(filePath));
}

export function cargarFormularioContacto(datos) {
  cy.typeInput('#nombre', datos.nombre);
  cy.typeInput('#correo', datos.correo);
  cy.typeInput('#telefono', datos.telefono);
  cy.invokeInput('#descripcion', datos.descripcion);
}

function quitarServicios() {
  cy.get('#servicios input')
    .each($checkbox => cy.wrap($checkbox).uncheck());
}

function eliminarFotos() {
  cy.get('body').then($body => {
    if ($body.find('.card-img-overlay button').length) {
      cy.get('.card-img-overlay button').each(btn => {
        cy.wrap(btn).click({ force: true });
      });
    }
  })
}
```

Fig. A3. Funciones de soporte para las situaciones particulares de prueba.

```
  cy.get('#precio').should('contain', `${publicacion.tipoMoneda + publicacion.precio}`);
  cy.get('#tipo-propiedad-operacion').should('contain', publicacion.permuta ? 'permuta' : '');
  if (publicacion.descripcion)
    cy.get('#descripcion').should('contain', publicacion.descripcion);
}

/* La libreria clona dos fotos de exposicion para
el carousel */
export function asercionesFotos(fotos) {
  if (fotos && fotos.fotosExposicion && fotos.fotosExposicion.length) {
    cy.get('#image-gallery')
      .children()
      .should('have.length', fotos.fotosExposicion.length + 2);
  } else
    cy.get('.lSSlideOuter').should('not.exist');
}

export function asercionesContacto() {
  cy.get(".swal2-success").should("exist")
}
```

Fig. A4. Funciones de soporte para las situaciones particulares de prueba.

```
export function cancelarPublicacion() {
  cy.get('#boton-cancelar').click({ force: true });
  cy.get('.swal2-confirm').click({ force: true });
}

export function asercionNotificacionSatisfactoria() {
  cy.get(".swal2-success").should("exist")
}

export function aceptarNotificacion() {
  cy.get(".swal2-confirm").click()
}
```

Fig. A5. Funciones de soporte para las situaciones particulares de prueba.

```
export function asercionesBD(publicacion,id) {
  cy.request(`api/FullPublicacionView/${id}`).as('publicacionBD');
  cy.request(`api/PropiedadxServicio/fields1?value0=${id}`).as('serviciosPublicacionBD');
  cy.request(`api/Foto/fields1?value0=${id}`).as('fotosPublicacionBD');

  cy.get('@publicacionBD').then((publicacionBD) => {
    asercionesUbicacionBD(publicacion.ubicacion, publicacionBD.body);
    asercionesPropiedadBD(publicacion.propiedad, publicacionBD.body);
    asercionesPublicacionBD(publicacion.publicacion, publicacionBD.body);
  })

  cy.get('@serviciosPublicacionBD').then((serviciosPublicacionBD) => {
    asercionesServiciosBD(publicacion.servicios, serviciosPublicacionBD.body);
  })

  cy.get('@fotosPublicacionBD').then((fotosPublicacionBD) => {
    asercionesFotosBD(publicacion.fotos, fotosPublicacionBD.body);
  })
}

export function asercionesUbicacionBD(ubicacion, publicacionBD) {
  if (ubicacion.provincia)
    expect(publicacionBD.nombreProvincia).to.equal(ubicacion.provincia);
  if(ubicacion.localidad)
    expect(publicacionBD.nombreLocalidad).to.equal(ubicacion.localidad);
  if(ubicacion.direccion)
    expect(publicacionBD.direccion).to.equal(ubicacion.direccion);
  if (ubicacion.iframe)
    expect(publicacionBD.iframe).to.equal(ubicacion.iframe);
}
```

Fig. A6. Funciones de soporte para las situaciones particulares de prueba.

```
export function asercionesPropiedadBD(propiedad, publicacionBD) {
  if (propiedad.tipoPropiedad)
    expect(publicacionBD.nombreTipoPropiedad).to.equal(propiedad.tipoPropiedad);
  if (propiedad.ambientes)
    expect(publicacionBD.ambientes).to.equal(propiedad.ambientes);
  if (propiedad.dormitorios)
    expect(publicacionBD.dormitorios).to.equal(propiedad.dormitorios);
  if (propiedad.banos)
    expect(publicacionBD.banos).to.equal(propiedad.banos);
  if (propiedad.garajes)
    expect(publicacionBD.garaje).to.equal(propiedad.garajes);
  if (propiedad.servicioSanitarioMunicipal)
    expect(publicacionBD.servicioSanitarioMunicipal).to.equal(propiedad.servicioSanitarioMunicipal);
  if (propiedad.expensas)
    expect(publicacionBD.expensas).to.equal(propiedad.expensas);
  if (propiedad.nombreAntiguedad)
    expect(publicacionBD.nombreAntiguedad).to.equal(propiedad.nombreAntiguedad);
  if (propiedad.superficieTotal)
    expect(publicacionBD.superficieTotal).to.equal(propiedad.superficieTotal);
  if (propiedad.superficieCubierta)
    expect(publicacionBD.superficieCubierta).to.equal(propiedad.superficieCubierta);
  if (propiedad.medidas)
    expect(publicacionBD.medida).to.equal(propiedad.medidas);
  if (propiedad.unidadMedidas)
    expect(publicacionBD.nombreSuperficie).to.equal(propiedad.unidadMedidas);
}

export function asercionesServiciosBD(servicios, serviciosPublicacionBD) {
  if (servicios) {
    let serviciosBD = serviciosPublicacionBD.map(s => s.idServicioNavigation.nombreServicio);
    //Chequeo de servicios dataset a los servicios de BD
    servicios.forEach(servicio => {
      expect(servicio).to.be.oneOf(serviciosBD);
    });

    //Chequeo de servicios de BD a los servicios de dataset
    serviciosBD.forEach(servicioBD => {
      expect(servicioBD).to.be.oneOf(servicios);
    });
  }
  else if (servicios == [])
    expect(serviciosPublicacionBD.length).to.be.equal(0);
}
```

Fig. A7. Funciones de soporte para las situaciones particulares de prueba.

```
export function asercionesPublicacionBD(publicacion, publicacionBD) {
  if (publicacion.titulo)
    expect(publicacionBD.titulo).to.equal(publicacion.titulo);
  if (publicacion.tipoOperacion)
    expect(publicacionBD.nombreOperacion).to.equal(publicacion.tipoOperacion);
  if (publicacion.precio)
    expect(publicacionBD.precio).to.equal(publicacion.precio);
  if (publicacion.tipoMoneda)
    expect(publicacionBD.nombreTipoMoneda).to.equal(publicacion.tipoMoneda);
  if (publicacion.disponible)
    expect(publicacionBD.disponible).to.equal(publicacion.disponible);
  if (publicacion.destacada)
    expect(publicacionBD.destacada).to.equal(publicacion.destacada);
  if (publicacion.credito)
    expect(publicacionBD.aptoCredito).to.equal(publicacion.credito);
  if (publicacion.permuta)
    expect(publicacionBD.permutable).to.equal(publicacion.permuta);
  if (publicacion.descripcion)
    expect(publicacionBD.descripcion).to.equal(publicacion.descripcion);
}

export function asercionesFotosBD(fotos, fotosPublicacionBD) {
  if (fotos) {
    let cantFotos = fotos.fotosExposicion.length + (fotos.fotoDestacada ? 1 : 0);
    expect(fotosPublicacionBD.length).to.equal(cantFotos);
  }
}
```

Fig. A8. Funciones de soporte para las situaciones particulares de prueba.

```
export function asercionesDatosEnviados(request, datos) {
  cy.wait(request)
    .then(interception => {
      let body = interception.request.body;
      expect(body.clientMail).to.equal(datos.correo || null);
      expect(body.clientName).to.equal(datos.nombre);
      expect(body.clientPhone).to.equal(datos.telefono || null);
      expect(body.description).to.equal(datos.descripcion || null);
    });
}
```

Fig. A9. Funciones de soporte para las situaciones particulares de prueba.

Anexo B: Iniciar sesión

Anexo B.1: Especificaciones de procedimientos de realización

```
describe('TPS#2', () => {
  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('login').then((dataSet) => this.dataSet = dataSet);
    cy.visit('/login');
    //Precondicion: debe estar en la pantalla de iniciar sesion y aun no se ha autenticado.
    cy.url().should('contain', 'login');
  })

  it('Usuario ingresa los datos de forma incorrecta, corrige y logra ingresar', function () {
    //Act (Actuar)
    //login intento #1
    cy.login(this.dataSet[1].nombreUsuario, this.dataSet[1].contrasena);
    //Postcondicion: se muestra un mensaje de error donde dice  "Usuario o contraseña incorrecta"
    cy.get('#datos-incorrectos').should('contain', 'Usuario o contraseña incorrecta');

    //login intento #2
    cy.login(this.dataSet[0].nombreUsuario, this.dataSet[0].contrasena);

    //Assert (Afirmar)
    //Postcondicion: debe estar en la pantalla del administrador.
    cy.url().should('contain', 'listarPublicaciones');
  })
})
```

Fig. B1. Especificación de procedimiento de realización de TPS#2 de la funcionalidad de iniciar sesión.

```
describe('TPS#3', () => {
  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('login').then((dataSet) => this.dataSet = dataSet);
    cy.visit('/login');
    //Precondicion: debe estar en la pantalla de iniciar sesion y aun no se ha autenticado.
    cy.url().should('contain', 'login');
  })

  it('Usuario ingresa usuario y/o contraseña incorrecta', function () {
    //Act (Actuar)
    cy.login(this.dataSet[1].nombreUsuario, this.dataSet[1].contrasena);

    //Assert (Afirmar)
    /*Postcondicion: el usuario se encuentra en la misma pantalla y
    debe identificar un mensaje de error que diga "Usuario o contraseña incorrecta".*/
    cy.url().should('contain', 'login');
    cy.get('#datos-incorrectos').should('contain', 'Usuario o contraseña incorrecta');
  })

  it('Usuario ingresa usuario y no ingresa contraseña', function() {
    //Act (Actuar)
    cy.login(this.dataSet[2].nombreUsuario, this.dataSet[2].contrasena);

    //Assert (Afirmar)
    /*Postcondicion: el usuario se encuentra en la misma pantalla y
     debe identificar un mensaje de error que diga "Ingrese una contraseña".*/
    cy.url().should('contain', 'login');
    cy.get('#sin-contrasena').should('contain', 'Ingrese una contraseña');
  })

  it('Usuario ingresa contraseña y  no ingresa usuario', function() {
    //Act (Actuar)
    cy.login(this.dataSet[3].nombreUsuario, this.dataSet[3].contrasena);

    //Assert (Afirmar)
    /*Postcondicion: el usuario se encuentra en la misma pantalla y
     debe identificar un mensaje de error que diga "Ingrese un usuario".*/
    cy.url().should('contain', 'login');
    cy.get('#sin-usuario').should('contain', 'Ingrese un usuario');
  })
```

Fig. B2. Especificación de procedimiento de realización de TPS#3 de la funcionalidad de iniciar sesión.

```
  it('Usuario no ingresa usuario y contraseña', function() {
    //Act (Actuar)
    cy.login(this.dataSet[4].nombreUsuario, this.dataSet[4].contrasena);

    //Assert (Afirmar)
    /*Postcondicion: el usuario se encuentra en la misma pantalla y
     debe identificar dos mensajes de errores que digan "Ingrese un usuario" e "Ingrese una contraseña".*/
    cy.url().should('contain', 'login');
    cy.get('#sin-usuario').should('contain', 'Ingrese un usuario');
    cy.get('#sin-contrasena').should('contain', 'Ingrese una contraseña');
  })
})
```

Fig. B3. Especificación de procedimiento de realización de TPS#3 de la funcionalidad de iniciar sesión.

Anexo B.2: Resultados de las pruebas

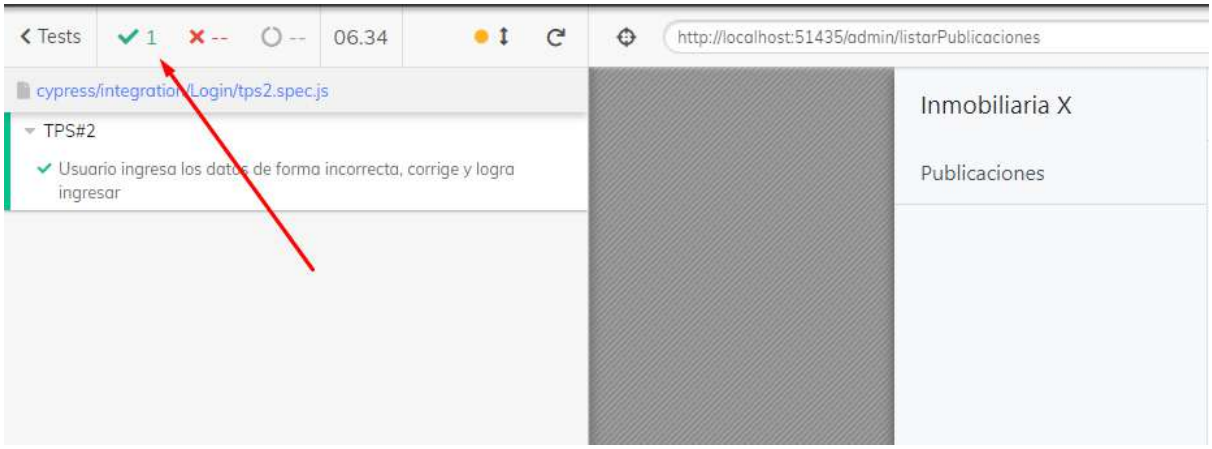


Fig. B4. Captura de ejecución de TPS#2 de la funcionalidad de iniciar sesión.

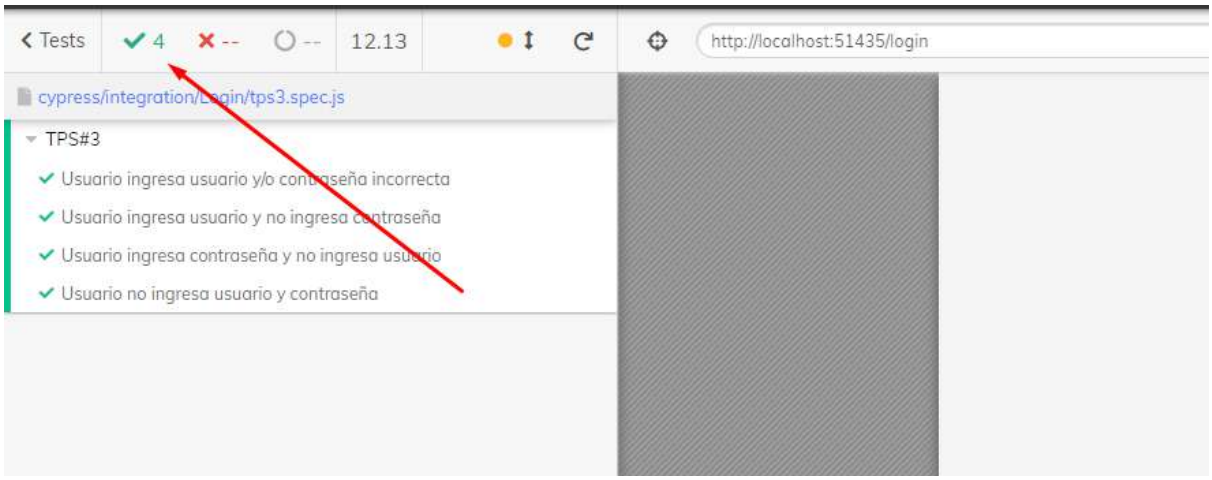


Fig. B5. Captura de ejecución de TPS#3 de la funcionalidad de iniciar sesión.

Anexo C: Nueva publicación

Anexo C.1: Bases de pruebas

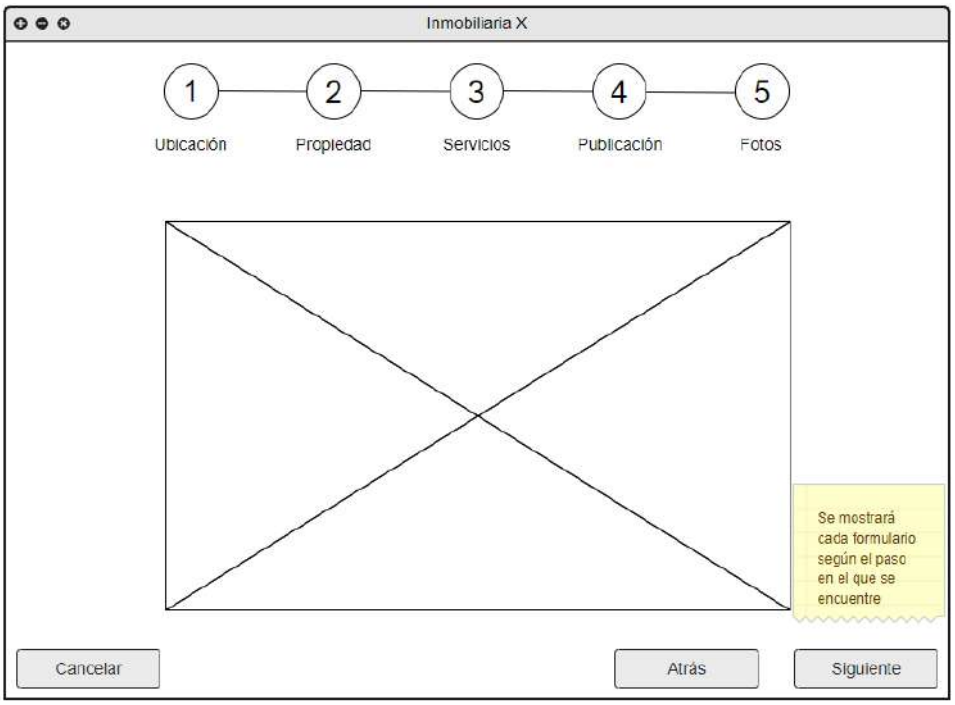


Fig. C1. Wireframe del formulario de pasos.

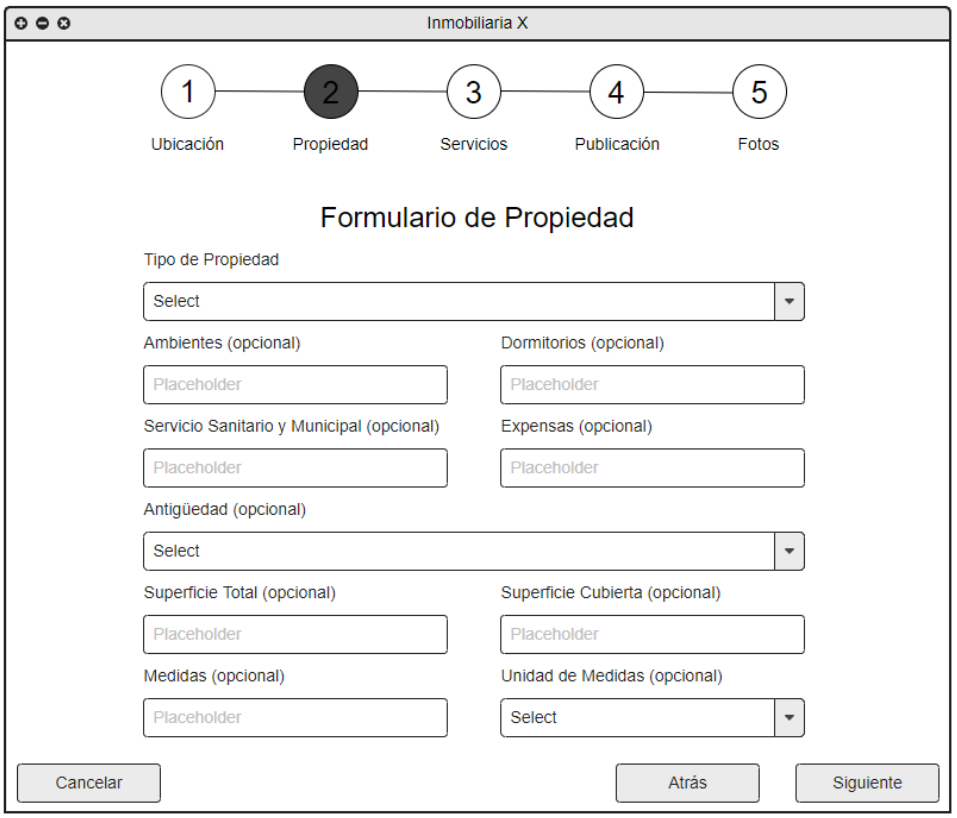


Fig. C2. Wireframe del formulario de propiedad.

Fig. C3. Wireframe del formulario de servicios.

Fig. C4. Wireframe del formulario de publicación.

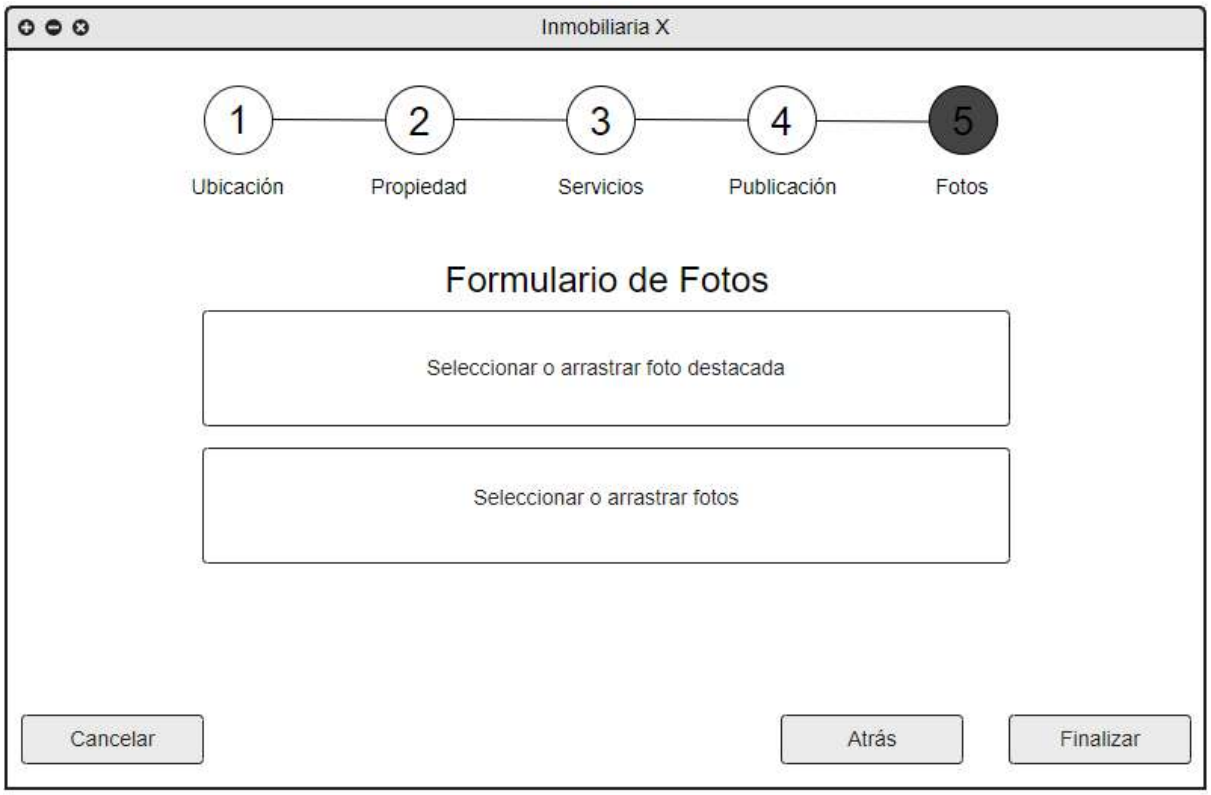


Fig. C5. Wireframe del formulario de fotos.

Anexo C.2: Conjunto de datos de entradas para las situaciones particulares de prueba


```
[
  {
    "ubicacion": {
      "direccion": "Calle 108 esq 7",
      "localidad": "General Pico",
      "provincia": "La Pampa"
    },
    "propiedad": {
      "tipoPropiedad": "Departamento"
    },
    "publicacion": {
      "tipoOperacion": "Alquilar",
      "tipoMoneda": "$",
      "precio": 80000,
      "titulo": "Test FR#2 TPS#1 TC#1"
    }
  },
  {
    "ubicacion": {
      "direccion": "Calle 108 esq 7",
      "localidad": "General Pico",
      "provincia": "La Pampa",
      "iframe": "<iframe src=\"https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.3630185366687!2d-63.7732652
    },
    "propiedad": {
      "tipoPropiedad": "Casa",
      "ambientes": 3,
      "dormitorios": 3,
      "banos": 3,
      "garajes": 3,
      "servicioSanitarioMunicipal": 1225,
      "expensas": 2222,
      "antiguedad": "A estrenar",
      "superficieTotal": 200,
      "superficieCubierta": 150,
      "medidas": "10x20",
      "unidadMedidas": "m²"
    },
    "servicios": [ "Agua corriente", "Cloacas", "Pavimento", "Luz eléctrica", "Gas natural" ],
    "publicacion": {
      "titulo": "Test FR#2 TPS#2 1",
      "tipoOperacion": "Comprar",
```


Fig. C6. Conjunto de entradas para las situaciones particulares de prueba de nueva publicación.

```
    "tipoMoneda": "$",
    "precio": 0,
    "descripcion": "descripcion descripcion descripcion descripcion descripcion descripcion descripcion ",
    "permuta": true,
    "destacada": true,
    "credito": true
  },
  "fotos": {
    "fotoDestacada": "../pruebas fotos/d1.jpeg",
    "fotosExposicion": [
      "../pruebas fotos/h1.jpeg",
      "../pruebas fotos/h2.jpeg",
      "../pruebas fotos/h3.jpeg",
      "../pruebas fotos/h4.jpeg",
      "../pruebas fotos/h5.jpeg",
      "../pruebas fotos/h6.jpeg",
      "../pruebas fotos/h7.jpeg",
      "../pruebas fotos/h8.jpeg"
    ]
  }
},
{
  "ubicacion": {
    "direccion": "Calle 108 esq 7",
    "localidad": "General Pico",
    "iframe": "<iframe src=\"https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.3630185366687!2d-63.773265
    "provincia": "La Pampa"
  },
  "propiedad": {
    "ambientes": 1,
    "tipoPropiedad": "Departamento"
  },
  "servicios": [ "Agua corriente", "Cloacas", "Pavimento", "Luz eléctrica", "Gas natural" ],
  "publicacion": {
    "tipoOperacion": "Alquilar",
    "tipoMoneda": "U$S",
    "precio": 25000,
    "titulo": "Test FR#2 TPS#3 1",
    "descripcion": "descripcion para FR#2 TPS#3"
  },
  "fotos": {
    "fotoDestacada": "../pruebas fotos/d1.jpeg",
```


Fig. C7. Conjunto de datos de entrada para las situaciones particulares de prueba de nueva publicación.

```
      "fotosExposicion": [
        "../pruebas fotos/h1.jpeg",
        "../pruebas fotos/h2.jpeg",
        "../pruebas fotos/h3.jpeg",
        "../pruebas fotos/h4.jpeg",
        "../pruebas fotos/h5.jpeg",
        "../pruebas fotos/h6.jpeg",
        "../pruebas fotos/h7.jpeg",
        "../pruebas fotos/h8.jpeg"
      ]
    }
  },
  {
    "ubicacion": {
      "provincia": "La Rioja",
      "localidad": "Independencia",
      "direccion": "Calle 108 esq 7"
    }
  },
  {
    "ubicacion": {
      "provincia": "Misiones",
      "localidad": "Candelaria",
      "direccion": "Calle 108 esq 7"
    },
    "propiedad": {
      "tipoPropiedad": "Campo"
    }
  },
  {
    "ubicacion": {
      "provincia": "Jujuy",
      "localidad": "Barrios",
      "direccion": "Calle 108 esq 7 Calle 108 esq 7 Calle 108 esq 7 Calle 108 esq 7"
    }
  },
  {
    "ubicacion": {
      "direccion": "Calle 108 esq 7"
    },
    "propiedad": {
      "medidas": "12345678901234567890123 4 x 12345678901234567890123 4"
    }
```


Fig. C8. Conjunto de datos de entrada para las situaciones particulares de prueba de nueva publicación.


```
  },
  {
    "propiedad": {
      "tipoPropiedad": "Departamento",
      "medidas": "12345 x 12345"
    },
    "publicacion": {
      "titulo": "Test FR#2 TPS#5Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test FR#2 TPS#5 Test
      "precio": 99999999999999999
    }
  },
  {
    "publicacion": {
      "titulo": "Test FR#2 TPS#5",
      "tipoOperacion": "Alquilar",
      "tipoMoneda": "$",
      "precio": 99999999999
    },
    "fotos": {
      "fotoDestacada": "../pruebas fotos/d1.jpeg",
      "fotosExposicion": [
        "../pruebas fotos/h1.jpeg",
        "../pruebas fotos/h2.jpeg",
        "../pruebas fotos/h3.jpeg",
        "../pruebas fotos/h4.jpeg"
      ]
    }
  }
]
```


Fig. C9. Conjunto de datos de entrada para las situaciones particulares de prueba de nueva publicación.

Anexo C.3: Especificaciones de procedimientos de realización

```
describe('TPS#1', () => {

  beforeEach(function() {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('nueva-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    //Precondicion: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.
    cy.visit('/login');
    cy.login('test', 'test');
    cy.get('#nueva-publicacion').click();
    cy.url().should('contain', 'nuevaPublicacion');
  })

  it('Carga con campos obligatorios', function() {
    //Act (Actuar)
    //form ubicacion - paso 1
    cargarFormularioUbicacion(this.publicaciones[0].ubicacion);
    cy.get('#boton-siguiente-web').click()

    //form propiedad - paso 2
    cargarFormularioPropiedad(this.publicaciones[0].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form servicios propiedad - paso 3
    cy.get('#boton-siguiente-web').click({ force: true })

    //form publicacion - paso 4
    cargarFormularioPublicacion(this.publicaciones[0].publicacion);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form fotos - paso 5
    cy.get('#boton-finalizar-web').click()

    //Assert (Afirmar)
    /* Postcondicion: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle de
    la publicación.*/
    cy.url().should('contain', 'detallePublicacion')
      .then(() => {
        cy.wait(2000);
        asercionesUbicacion(this.publicaciones[0].ubicacion);
```

Fig. C10. Especificación de procedimiento de realización de TPS#1 de la funcionalidad nueva publicación.

```
        asercionesPropiedad(this.publicaciones[0].propiedad);
        asercionesServicios(this.publicaciones[0].servicios);
        asercionesPublicacion(this.publicaciones[0].publicacion);
        asercionesFotos(this.publicaciones[0].fotos);

        cy.url()
          .then(url => {
            let idPublicacion = url.split('/').pop();
            /* Resultado esperado: los datos que el usuario ingreso se encuentran cargados en la base de datos. */
            asercionesBD(this.publicaciones[0], idPublicacion);
          });
      });
  });
})
```

Fig. C11. Especificación de procedimiento de realización de TPS#1 de la funcionalidad nueva publicación.

```
describe('TPS#2', () => {
  beforeEach(function() {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('nueva-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    //Precondicion: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.
    cy.visit('/login');
    cy.login('test', 'test');
    cy.get('#nueva-publicacion').click();
    cy.url().should('contain', 'nuevaPublicacion');
  })

  it('Carga con todos los campos', function () {

    //Act (Actuar)
    //form ubicacion - paso 1
    cargarFormularioUbicacion(this.publicaciones[1].ubicacion);
    cy.get('#boton-siguiente-web').click()

    //form propiedad - paso 2
    cargarFormularioPropiedad(this.publicaciones[1].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form servicios propiedad - paso 3
    cargarFormularioServicios(this.publicaciones[1].servicios);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form publicacion - paso 4
    cargarFormularioPublicacion(this.publicaciones[1].publicacion);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form fotos - paso 5
    cargarFormularioFotos(this.publicaciones[1].fotos);
    cy.get('#boton-finalizar-web').click();

    //Assert (Afirmar)
    /* Postcondicion: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle
    de la publicación. */
    cy.url().should('contain', 'detallePublicacion')
      .then(() => {
        cy.wait(2000);
        asercionesUbicacion(this.publicaciones[1].ubicacion);
        asercionesPropiedad(this.publicaciones[1].propiedad);
        asercionesServicios(this.publicaciones[1].servicios);
        asercionesPublicacion(this.publicaciones[1].publicacion);
        asercionesFotos(this.publicaciones[1].fotos);
```

Fig. C12. Especificación de procedimiento de realización de TPS#2 de la funcionalidad nueva publicación.

```
        cy.url()
          .then(url => {
            let idPublicacion = url.split('/').pop();
            /* Resultado esperado: los datos que el usuario ingreso se encuentran cargados en la base de datos. */
            asercionesBD(this.publicaciones[1], idPublicacion);
          });
      });
  });
})
```

Fig. C13. Especificación de procedimiento de realización de TPS#2 de la funcionalidad nueva publicación.

```
describe('TPS#3', () => {
  beforeEach(function() {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('nueva-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    //Precondicion: el usuario debe estar autenticado y debe estar en la pantalla de nueva publicación.
    cy.visit('/login');
    cy.login('test', 'test');
    cy.get('#nueva-publicacion').click();
    cy.url().should('contain', 'nuevaPublicacion');
  })

  it('Carga de formulario volviendo a paso anteriores', function() {
    //Act (Actuar)
    //form ubicacion - paso 1
    cy.get('#provincia').select(this.publicaciones[2].ubicacion.provincia)
    cy.get('#localidad').select(this.publicaciones[2].ubicacion.localidad)
    cy.get('#direccion').type(this.publicaciones[2].ubicacion.direccion)
    cy.get('#boton-siguiente-web').click()

    //form propiedad - paso 2
    cy.get('#tipoPropiedad').select(this.publicaciones[2].propiedad.tipoPropiedad)
    cy.get('#boton-siguiente-web').click({ force: true })

    //form servicios propiedad - paso 3
    cy.get('#boton-siguiente-web').click({ force: true })

    //form publicacion - paso 4
    cy.get('#titulo').type(this.publicaciones[2].publicacion.titulo)
    cy.get('#operacion').select(this.publicaciones[2].publicacion.tipoOperacion)
    cy.get('#precio').type(this.publicaciones[2].publicacion.precio)
    cy.get('#tipoMoneda').select(this.publicaciones[2].publicacion.tipoMoneda)
    cy.get('#boton-siguiente-web').click({ force: true })

    //Volver a paso 1
    cy.get('#boton-atras-web').click({ force: true })
    cy.get('#boton-atras-web').click({ force: true })
    cy.get('#boton-atras-web').click({ force: true })
    cy.get('#boton-atras-web').click({ force: true })

    //form ubicacion - paso 1
    cy.get('#iframe').invoke('val', this.publicaciones[2].ubicacion.iframe).trigger('input')
    cy.get('#boton-siguiente-web').click()
```

Fig. C14. Especificación de procedimiento de realización de TPS#3 de la funcionalidad de nueva publicación.

```
    //form propiedad - paso 2
    cy.get('#ambientes').type(this.publicaciones[2].propiedad.ambientes)
    cy.get('#boton-siguiente-web').click({ force: true })

    //form servicios propiedad - paso 3
    cargarFormularioServicios(this.publicaciones[2].servicios);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form publicacion - paso 4
    cy.get('#descripcion').type(this.publicaciones[2].publicacion.descripcion)
    cy.get('#boton-siguiente-web').click({ force: true })

    //form fotos - paso 5
    cargarFormularioFotos(this.publicaciones[2].fotos);
    cy.get('#boton-finalizar-web').click();

    //Assert (Afirmar)
    /* Postcondicion: el usuario se encuentra en la pantalla del detalle de la publicación y puede ver el detalle
    de la publicación.*/
    cy.url().should('contain', 'detallePublicacion')
      .then(() => {
        cy.wait(2000);
        asercionesUbicacion(this.publicaciones[2].ubicacion);
        asercionesPropiedad(this.publicaciones[2].propiedad);
        asercionesServicios(this.publicaciones[2].servicios);
        asercionesPublicacion(this.publicaciones[2].publicacion);
        asercionesFotos(this.publicaciones[2].fotos);

        cy.url()
          .then(url => {
            let idPublicacion = url.split('/').pop();
            /* Resultado esperado: los datos que el usuario ingreso se encuentran cargados en la base de datos. */
            asercionesBD(this.publicaciones[2], idPublicacion);
          });
      });

  });
})
```

Fig. C15. Especificación de procedimiento de realización de TPS#3 de la funcionalidad de nueva publicación.

```
describe('TPS#4', () => {
  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('nueva-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: el usuario debe estar autenticado y debe estar
      en la pantalla de nueva publicación.
    */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.get('#nueva-publicacion').click();
    cy.url().should('contain', 'nuevaPublicacion');
  })

  it('No cumple con la validacion del paso 1', function() {
    /* Precondicion: debe estar en el formulario de ubicación y
     no debe contener ningún dato obligatorio cargado, es decir,
     "inicia con el formulario vacio".
    */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');

    //Act (Actuar)
    //form ubicacion - paso 1
    cy.get('#boton-siguiente-web').click()

    //Assert (Afirmar)
    /*Postcondicion: el usuario administrador se encuentra en el mismo paso
    con mensajes de error "* Campo obligatorio". */
    cy.get('#provincia-requerido').should('contain', "* Campo obligatorio");
    cy.get('#localidad-requerido').should('contain', "* Campo obligatorio");
    cy.get('#direccion-requerido').should('contain', "* Campo obligatorio");
  });

  it('No cumple con la validacion del paso 2', function() {
    /* Precondicion: debe estar en el formulario de propiedad y
     no debe contener ningún dato obligatorio cargado, es decir,
     "inicia con el formulario vacio".
    */
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');
    //Act (Actuar)
    //form ubicacion - paso 1
    cargarFormularioUbicacion(this.publicaciones[3].ubicacion);
    cy.get('#boton-siguiente-web').click()

    //form propiedad - paso 2
    cy.get('#boton-siguiente-web').click({ force: true })
```

Fig. C16. Especificación de procedimiento de realización de TPS#4 de la funcionalidad de nueva publicación.

```
    //Assert (Afirmar)
    /*Postcondicion: el usuario administrador se encuentra en el mismo paso
    con mensajes de error "* Campo obligatorio". */
    cy.get('#tipoPropiedad-requerido').should('contain', "* Campo obligatorio");
  });

  it('No cumple con la validacion del paso 4', function() {
    /* Precondicion: debe estar en el formulario de publicacion y
     no debe contener ningún dato obligatorio cargado, es decir,
     "inicia con el formulario vacio".
    */
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');

    //Act (Actuar)

    //form ubicacion - paso 1
    cargarFormularioUbicacion(this.publicaciones[4].ubicacion);
    cy.get('#boton-siguiente-web').click()

    //form propiedad - paso 2
    cargarFormularioPropiedad(this.publicaciones[4].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true })

    //form servicios propiedad - paso 3
    cy.get('#boton-siguiente-web').click({ force: true })

    //form publicacion - paso 4
    cy.get('#boton-siguiente-web').click({ force: true })

    //Assert (Afirmar)
    /*Postcondicion: el usuario administrador se encuentra en el mismo paso
    con mensajes de error "* Campo obligatorio". */
    cy.get('#titulo-requerido').should('contain', "* Campo obligatorio");
    cy.get('#tipoOperacion-requerido').should('contain', "* Campo obligatorio");
    cy.get('#tipoMoneda-requerido').should('contain', "* Campo obligatorio");
    cy.get('#precio-requerido').should('contain', "* Campo obligatorio");
  });

  afterEach(function () {
    /* Postcondicion: el sistema redirecciona al usuario a la vista de listado
     de publicaciones. */
    cancelarPublicacion();
    cy.url().should('contain', 'listarPublicaciones');
  })
})
```

Fig. C17. Especificación de procedimiento de realización de TPS#4 de la funcionalidad de nueva publicación.

Anexo C.4: Entidad de contexto de prueba configurada

	IdAntiguedad	NombreAntiguedad
1	1	A estrenar
2	2	Menor a 10 años
3	3	Entre 10 y 25 años
4	4	Entre 25 y 50 años
5	5	Mayor a 50 años

Fig. C18. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

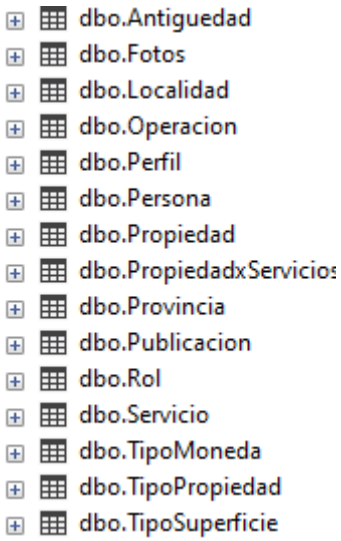


Fig. C19. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdOperacion	NombreOperacion
1	1	Comprar
2	2	Alquilar

Fig. C20. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdProvincia	NombreProvincia
1	1	Buenos Aires
2	2	Buenos Aires-GE
3	3	Capital Federal
4	4	Catamarca
5	5	Chaco
6	6	Chubut
7	7	Córdoba
8	8	Corrientes
9	9	Entre Ríos
10	10	Formosa
11	11	Jujuy
12	12	La Pampa
13	13	La Rioja
14	14	Mendoza
15	15	Misiones
16	16	Neuquén
17	17	Río Negro
18	18	Salta
19	19	San Juan

Fig. C21. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdServicio	NombreServicio
1	1	Agua corriente
2	2	Luz eléctrica
3	3	Cloacas
4	4	Gas natural
5	5	Pavimento

Fig. C22. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdTipoMoneda	NombreTipoMoneda
1	1	$
2	2	U$S

Fig. C23. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdTipoPropiedad	NombreTipoPropiedad
1	1	Casa
2	2	Departamento
3	3	Quinta
4	4	Terreno
5	5	Campo
6	6	Chacra
7	7	Local comercial
8	8	Oficina
9	9	Galpón

Fig. C24. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación.

	IdTipoSuperficie	NombreSuperficie
1	1	m^2
2	2	ha

Fig. C25. Entidad de contexto de prueba configurada para la funcionalidad de nueva publicación

Anexo C.5: Resultados de pruebas

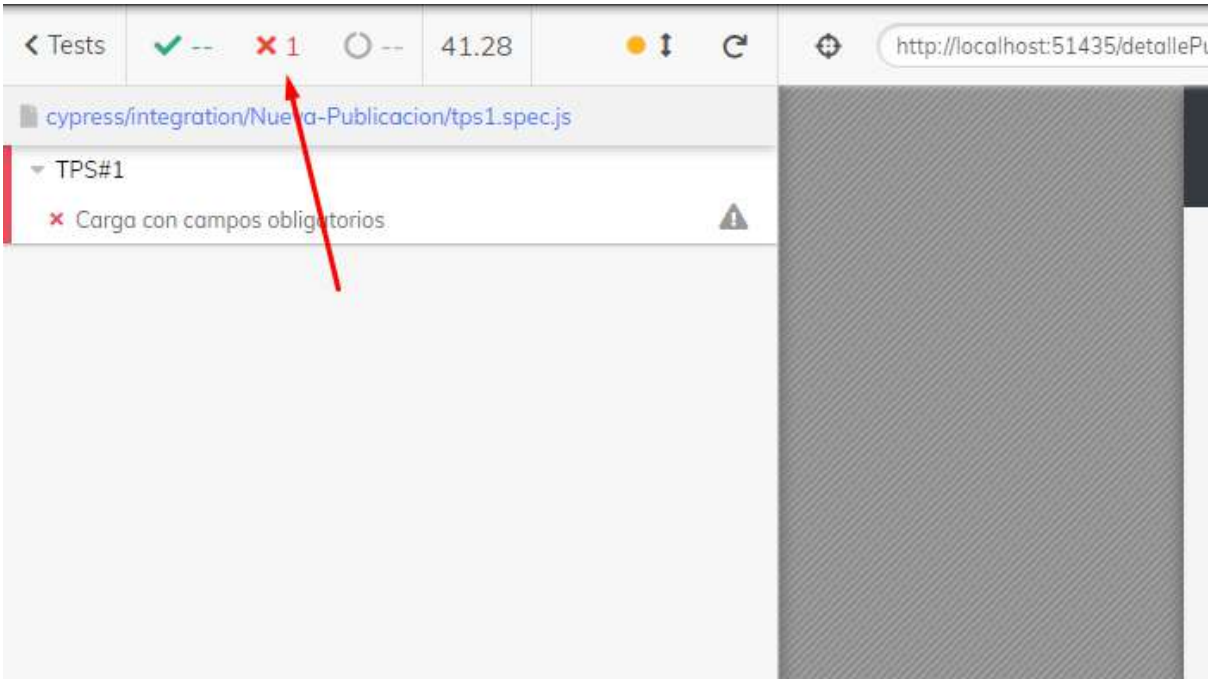


Fig. C26. Captura de pantalla de la ejecución de TPS#1 de la funcionalidad nueva publicación.

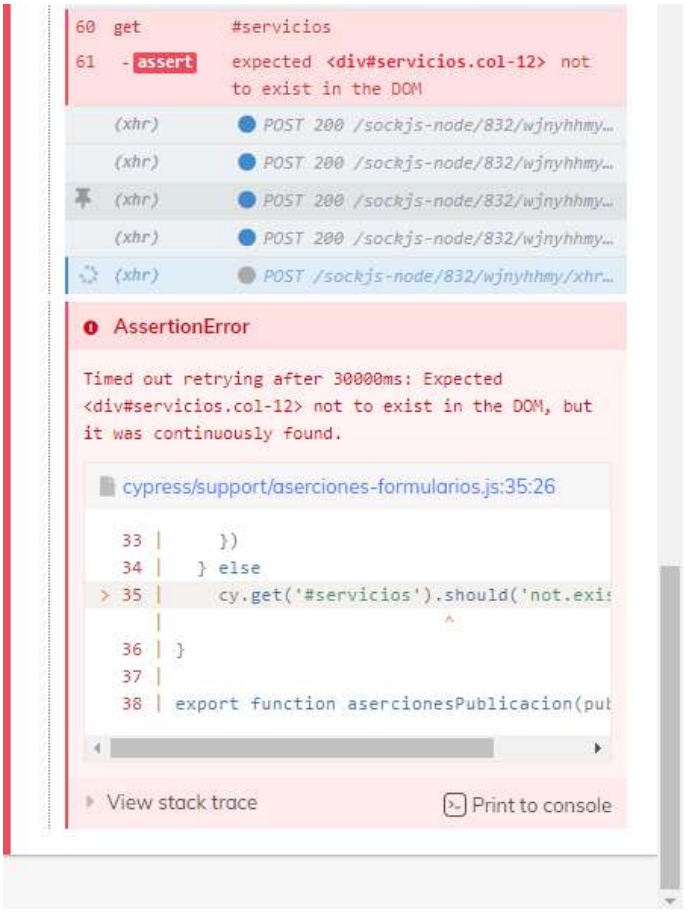


Fig. C27. Captura de pantalla del error detectado en la traza de la ejecución de TPS#1 de la funcionalidad nueva publicación.

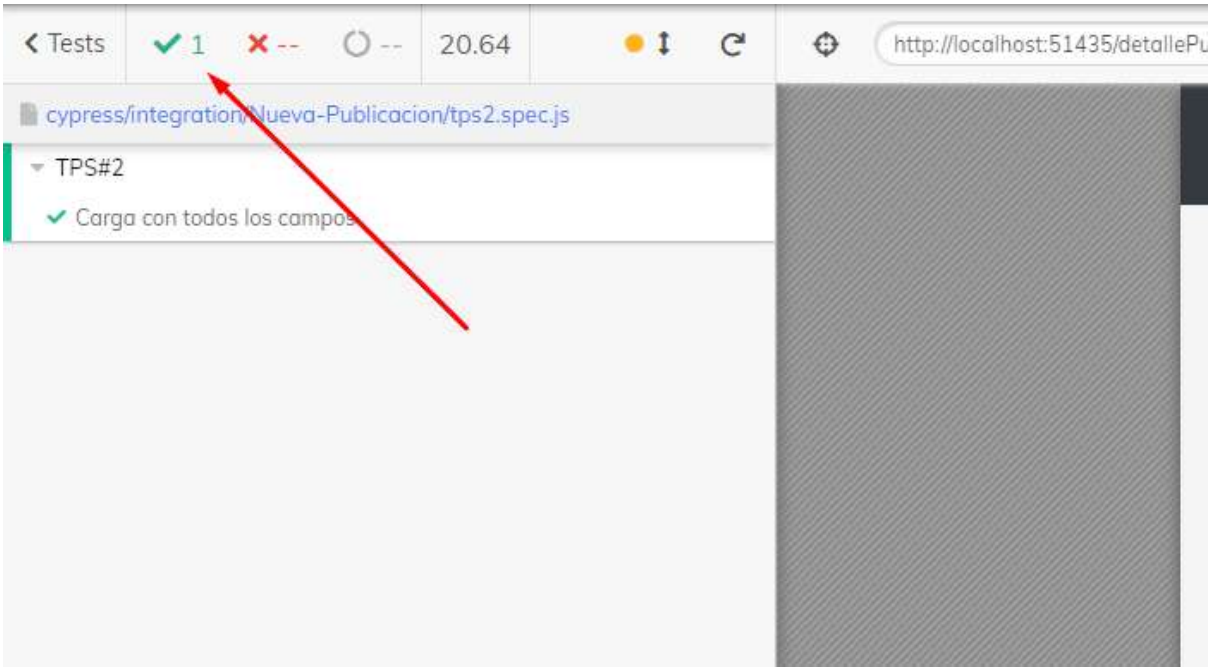


Fig. C28. Captura de pantalla de la ejecución de TPS#2 de la funcionalidad nueva publicación.

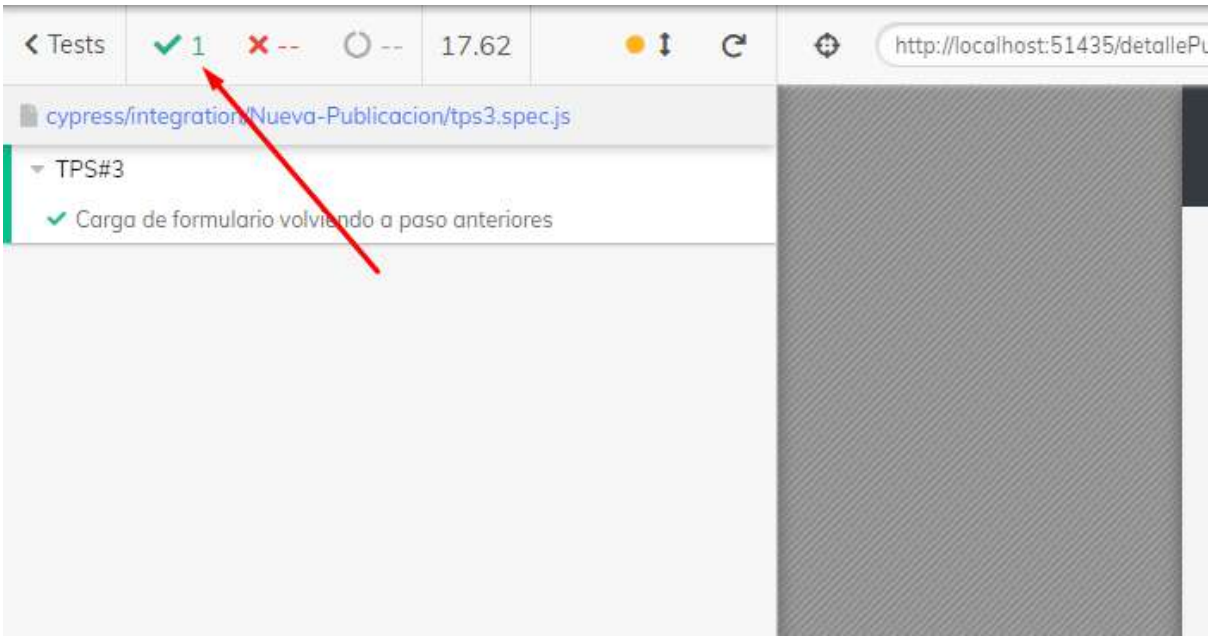


Fig. C29. Captura de pantalla de la ejecución de TPS#3 de la funcionalidad nueva publicación.

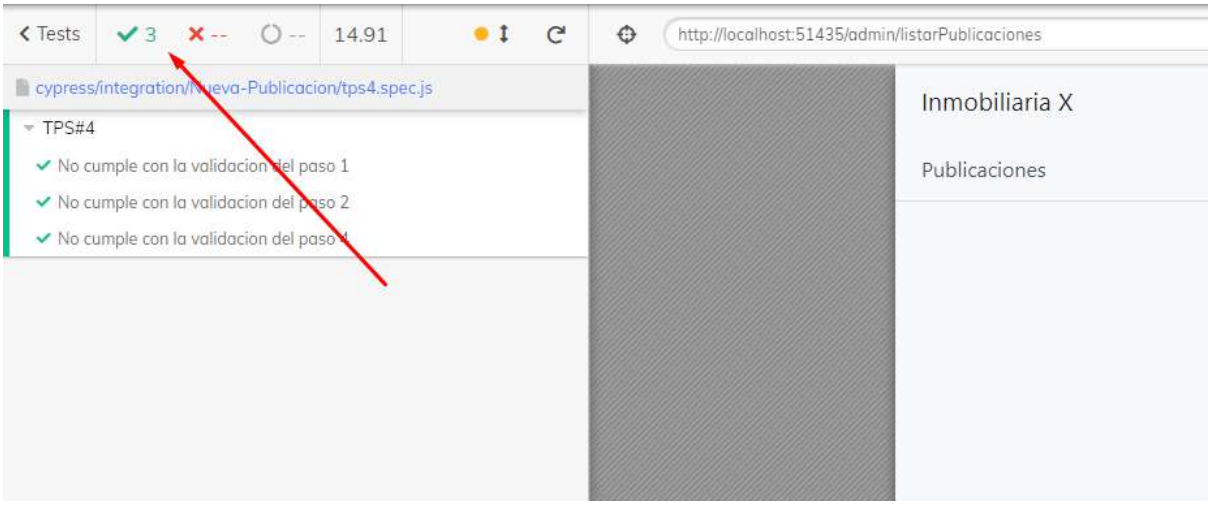


Fig. C30. Captura de pantalla de la ejecución de TPS#4 de la funcionalidad nueva publicación.

Anexo C.6 : Resultados de prueba de TPS#1 y TPS#5 después de la corrección

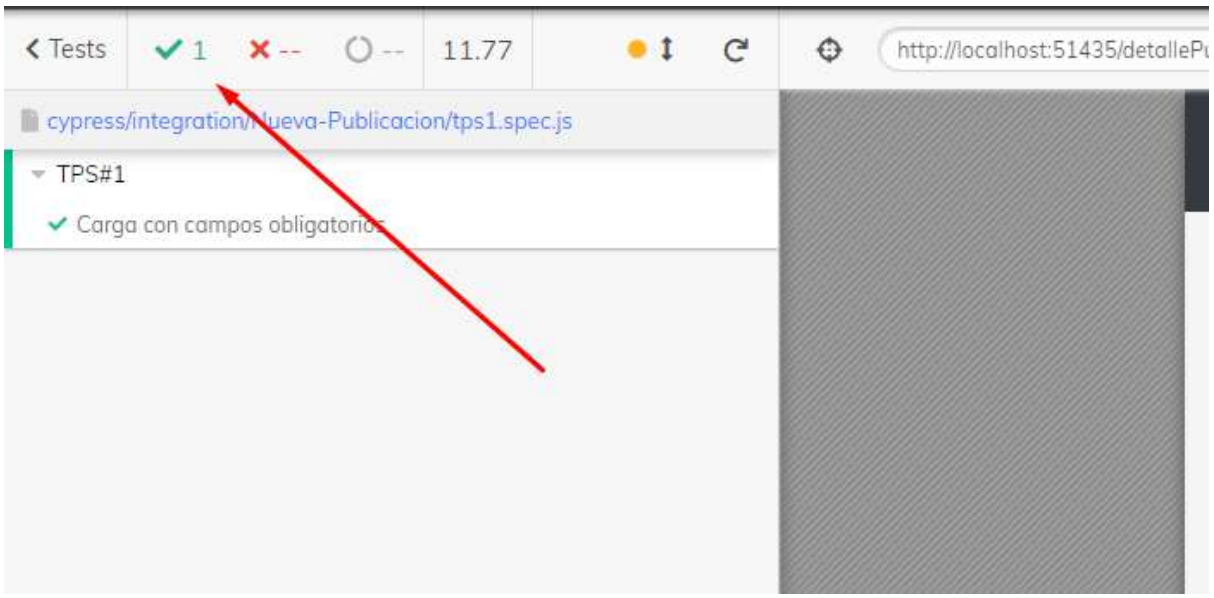


Fig. C31. Captura de pantalla de la ejecución de TPS#1 de la funcionalidad nueva publicación con el error solucionado.

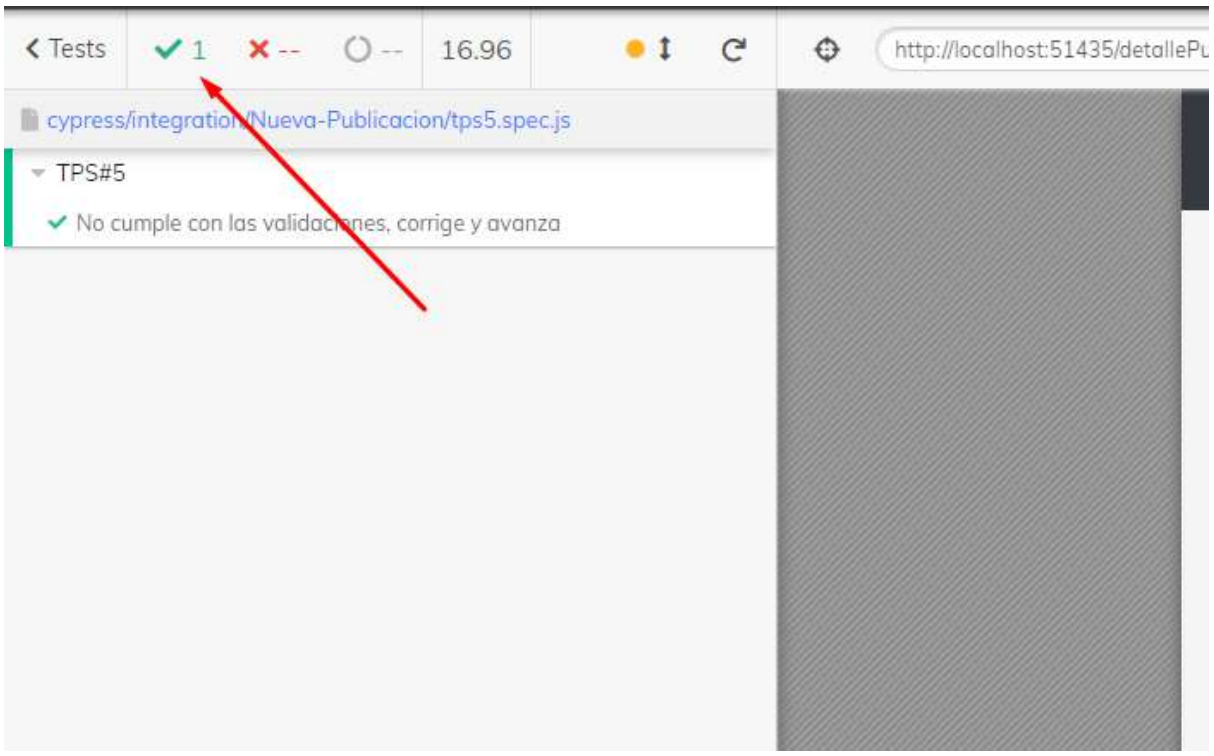
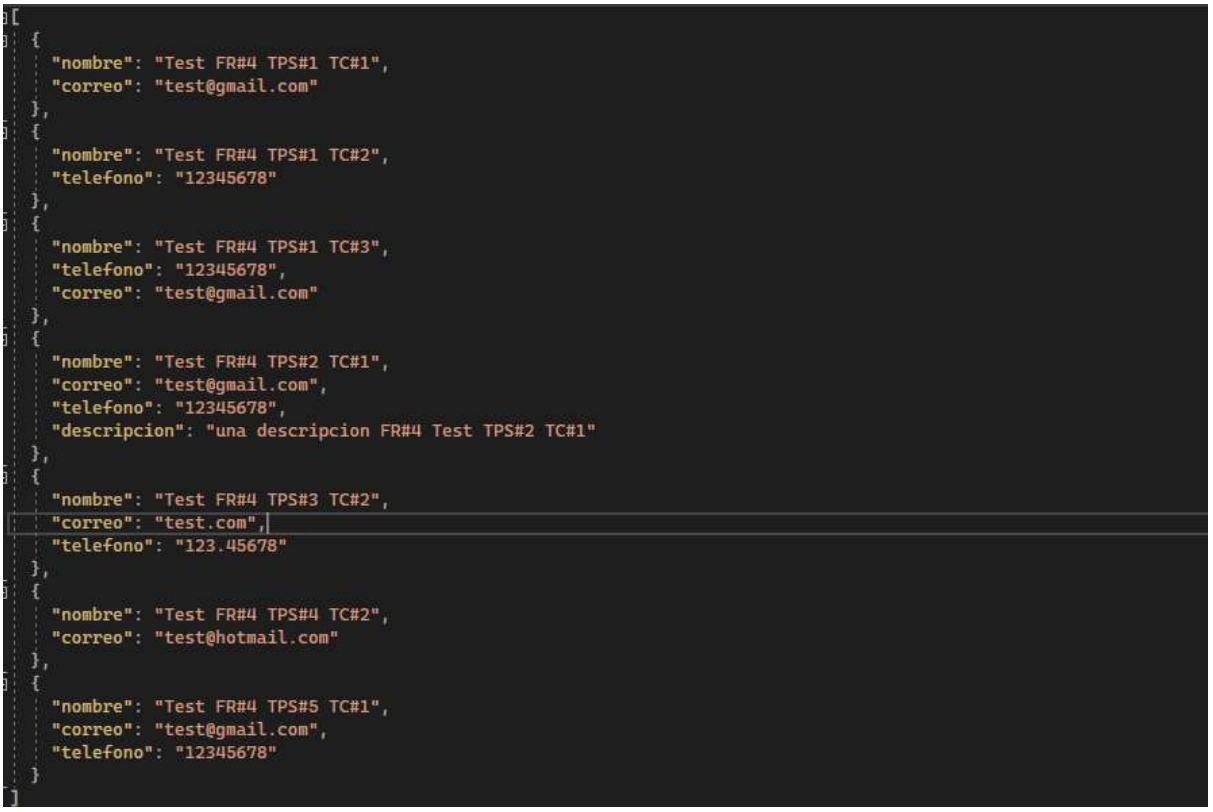


Fig. C32. Captura de pantalla de la ejecución de TPS#5 de la funcionalidad nueva publicación con el error solucionado.

Anexo D: Contactar inmobiliaria por correo electrónico

Anexo D.1: Conjunto de datos de entradas para las situaciones particulares de prueba



```
[
  {
    "nombre": "Test FR#4 TPS#1 TC#1",
    "correo": "test@gmail.com"
  },
  {
    "nombre": "Test FR#4 TPS#1 TC#2",
    "telefono": "12345678"
  },
  {
    "nombre": "Test FR#4 TPS#1 TC#3",
    "telefono": "12345678",
    "correo": "test@gmail.com"
  },
  {
    "nombre": "Test FR#4 TPS#2 TC#1",
    "correo": "test@gmail.com",
    "telefono": "12345678",
    "descripcion": "una descripcion FR#4 Test TPS#2 TC#1"
  },
  {
    "nombre": "Test FR#4 TPS#3 TC#2",
    "correo": "test.com",
    "telefono": "123.45678"
  },
  {
    "nombre": "Test FR#4 TPS#4 TC#2",
    "correo": "test@hotmail.com"
  },
  {
    "nombre": "Test FR#4 TPS#5 TC#1",
    "correo": "test@gmail.com",
    "telefono": "12345678"
  }
]
```

Fig. D1. Conjunto de datos de entrada para las situaciones particulares de prueba de contactar inmobiliaria por correo electrónico.

Anexo D.2: Especificaciones de procedimientos de realización

```
describe('TPS#1', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('contactar-inmobiliaria').then((dataSet) => this.dataSet = dataSet);
    cy.intercept('api/ContactoMail/SendMail').as('datosEnviados');
    /* el usuario debe estar en la vista del formulario de contacto. Además, tiene
     que haber un servidor de correos electrónicos configurado con los siguientes
     correos: contactotest@test.com y inmobiliariatest@test.com */
    cy.visit('/contacto');
    cy.get('#titulo-contacto').should('contain','Formulario de Contacto');
  })

  it('Usuario contacta a la inmobiliaria con nombre y correo', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[0]);

    //Assert (Afirmar)
    cy.get("#enviar").click();

    /* Resultado esperado: el sistema emitió un correo a la inmobiliaria con
     los datos suministrados por el cliente. */
    asercionesDatosEnviados('@datosEnviados', this.dataSet[0]);
  })

  it('Usuario contacta a la inmobiliaria con nombre y telefono', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[1]);

    //Assert (Afirmar)
    cy.get("#enviar").click();

    /* Resultado esperado: el sistema emitió un correo a la inmobiliaria con
     los datos suministrados por el cliente. */
    asercionesDatosEnviados('@datosEnviados', this.dataSet[1]);
  })

  it('Usuario contacta a la inmobiliaria con nombre, correo y telefono', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[2]);
```

Fig. D2. Especificación de procedimiento de realización de TPS#1 de la funcionalidad contactar inmobiliaria por correo electrónico.

```
    //Assert (Afirmar)
    cy.get("#enviar").click();

    /* Resultado esperado: el sistema emitió un correo a la inmobiliaria con
     los datos suministrados por el cliente. */
    asercionesDatosEnviados('@datosEnviados', this.dataSet[2]);
  })

  afterEach(function () {
    /* Postcondicion: el usuario cliente puede identificar una notificación advirtiendo que la
    operación se realizó correctamente. */
    asercionesContacto();
  })
})
```

Fig. D3. Especificación de procedimiento de realización de TPS#1 de la funcionalidad contactar inmobiliaria por correo electrónico.

```
describe('TPS#2', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('contactar-inmobiliaria').then((dataSet) => this.dataSet = dataSet);
    cy.intercept('api/ContactoMail/SendMail').as('datosEnviados');
    /* el usuario debe estar en la vista del formulario de contacto. Además, tiene
     que haber un servidor de correos electrónicos configurado con los siguientes
     correos: contactotest@test.com y inmobiliariatest@test.com */
    cy.visit('/contacto');
    cy.get('#titulo-contacto').should('contain', 'Formulario de Contacto');
  })

  it('Usuario contacta a la inmobiliaria con todos los datos desde seccion contacto', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[3]);
    cy.get("#enviar").click()

    //Assert (Afirmar)
    /* Resultado esperado: el sistema emitió un correo a la inmobiliaria con
     los datos suministrados por el cliente. */
    asercionesDatosEnviados('@datosEnviados', this.dataSet[3]);
  })

  afterEach(function () {
    /* Postcondicion: el usuario cliente puede identificar una notificación advirtiendo que la
    operación se realizó correctamente. */
    asercionesContacto();
  })
})
```

Fig. D4. Especificación de procedimiento de realización de TPS#2 de la funcionalidad contactar inmobiliaria por correo electrónico.

```
describe('TPS#3', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('contactar-inmobiliaria').then((dataSet) => this.dataSet = dataSet);
    /* el usuario debe estar en la vista del formulario de contacto. Además, tiene
     que haber un servidor de correos electrónicos configurado con los siguientes
     correos: contactotest@test.com y inmobiliariatest@test.com */
    cy.visit('/contacto');
    cy.get('#titulo-contacto').should('contain', 'Formulario de Contacto');
  })

  it('Usuario no ingresa valores, formulario no valido', function() {
    //Act (Actuar)
    cy.get("#enviar").click()

    //Assert (Afirmar)
    /* Postcondicion: el usuario puede identificar mensajes de error “Campo requerido *”
     en nombre, “Debe ingresar un medio de contacto” en teléfono y en email. */
    cy.get("#nombre-obligatorio").should("contain", "Campo requerido *")
    cy.get("#sin-contacto").should("contain", "Debe ingresar un medio de contacto")
  })

  it('Usuario ingresa valores pero no cumple con los formatos, formulario no valido', function() {
    //Act (Actuar)
    cargarFormularioContacto(this.dataSet[4]);
    cy.get("#enviar").click();

    //Assert (Afirmar)
    /* Postcondicion: el usuario cliente no logró contactarse satisfactoriamente con la inmobiliaria
     y puede identificar mensajes de error “No corresponde a un correo electrónico” en email y “Debe
     estar formado por dígitos, "-", "+" y "()"” en teléfono. */
    cy.get("#correo-invalido").should("contain", "No corresponde a un correo electrónico");
    cy.get("#telefono-invalido").should("contain", 'Debe estar formado por dígitos, "-", "+" y "()"');
  })
})
```

Fig. D5. Especificación de procedimiento de realización de TPS#3 de la funcionalidad contactar inmobiliaria por correo electrónico.

```
describe('TPS#4', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    cy.fixture('contactar-inmobiliaria').then((dataSet) => this.dataSet = dataSet);
    cy.intercept('api/ContactoMail/SendMail').as('datosEnviados');
    /* el usuario debe estar en la vista del formulario de contacto. Además, tiene
     que haber un servidor de correos electrónicos configurado con los siguientes
     correos: contactotest@test.com y inmobiliariatest@test.com */
    cy.visit('/contacto');
    cy.get('#titulo-contacto').should('contain', 'Formulario de Contacto');
  })

  it('Usuario no cumple con las validaciones, corrige y contacta a la inmobiliaria', function() {
    //Act (Actuar)
    cy.get("#enviar").click();

    //Assert (Afirmar)
    /* Postcondicion: el usuario cliente puede identificar mensajes de error "Campo requerido *"
     en nombre, "Debe ingresar un medio de contacto" en teléfono y en email. */
    cy.get("#nombre-obligatorio").should("contain", "Campo requerido *");
    cy.get("#sin-contacto").should("contain", "Debe ingresar un medio de contacto");

    cargarFormularioContacto(this.dataSet[5]);
    cy.get("#enviar").click();

    //Assert (Afirmar)
    /* Resultado esperado: el sistema emitió un correo a la inmobiliaria con
     los datos suministrados por el cliente. */
    asercionesDatosEnviados('@datosEnviados', this.dataSet[5]);
    /* Postcondicion:  el usuario cliente puede identificar una notificación advirtiendo
     que la operación se realizó correctamente.*/
    asercionesContacto();
  })
})
```

Fig. D6. Especificación de procedimiento de realización de TPS#4 de la funcionalidad contactar inmobiliaria por correo electrónico.

Anexo D.3: Resultados de pruebas

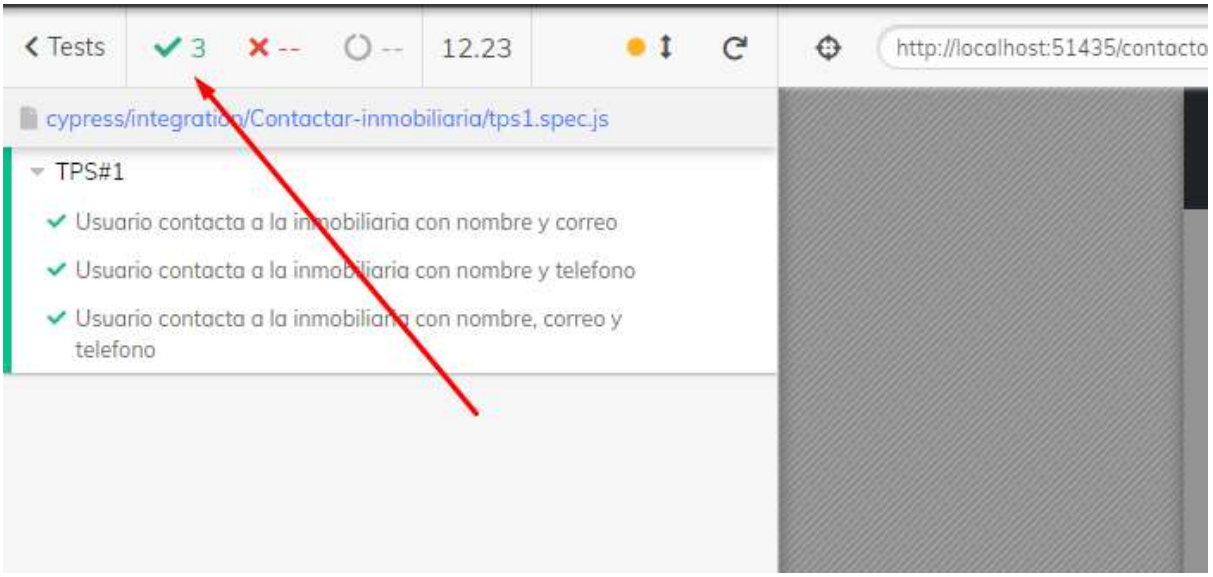


Fig. D7. Captura de pantalla de la ejecución de TPS#1 de la funcionalidad contactar inmobiliaria por correo electrónico.

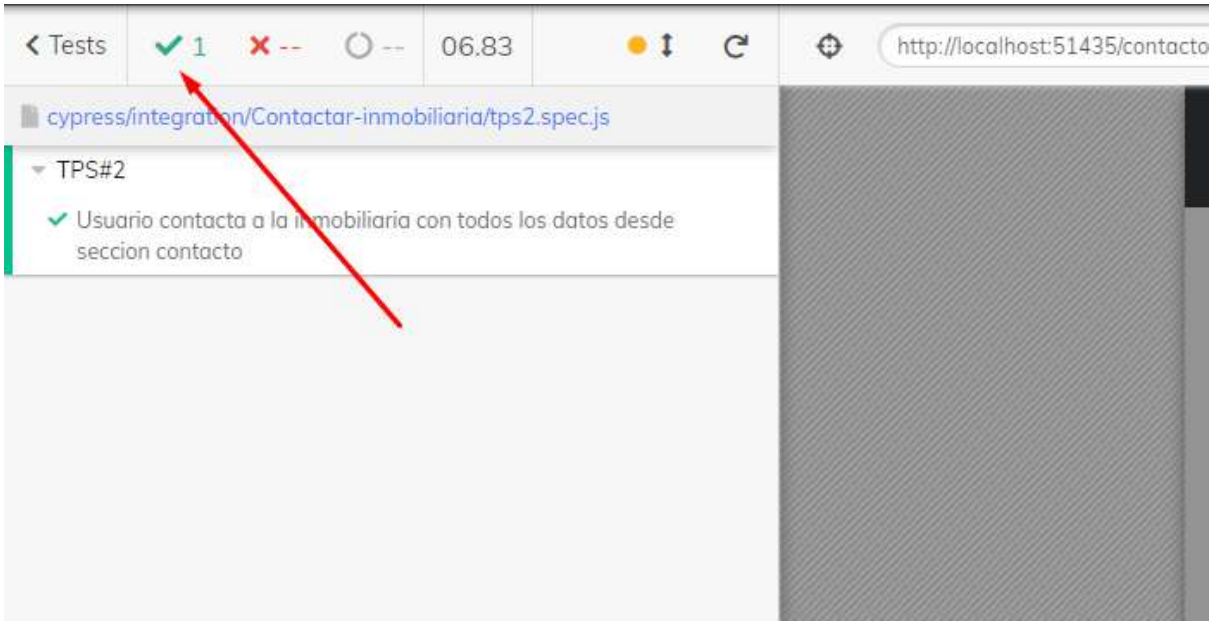


Fig. D8. Captura de pantalla de la ejecución de TPS#2 de la funcionalidad contactar inmobiliaria por correo electrónico.

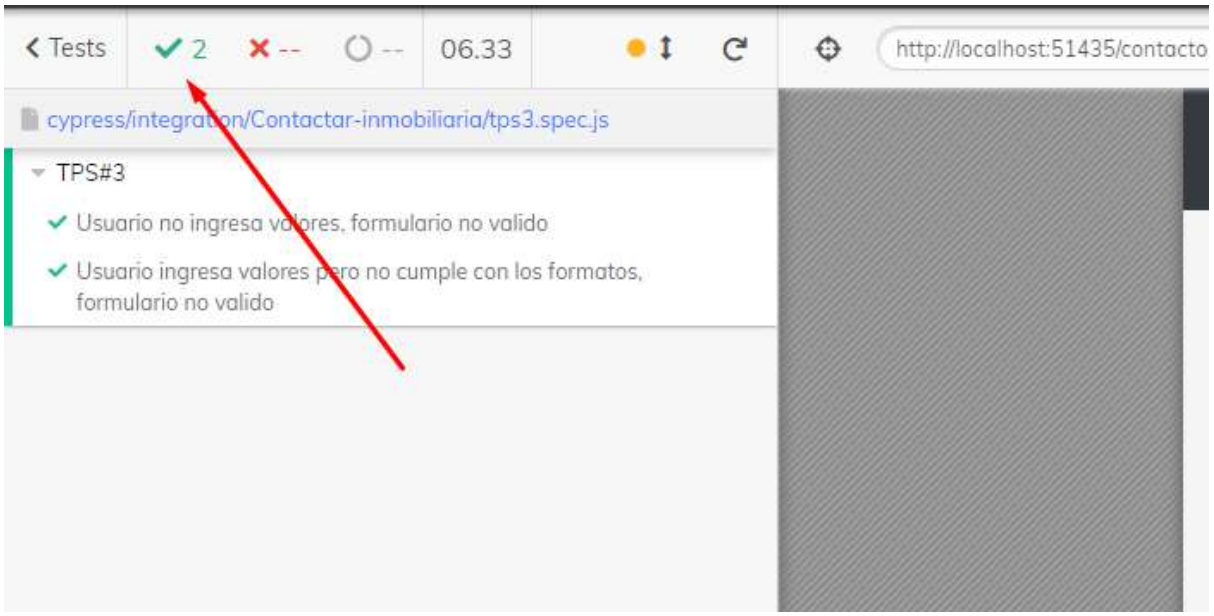


Fig. D9. Captura de pantalla de la ejecución de TPS#3 de la funcionalidad contactar inmobiliaria por correo electrónico.

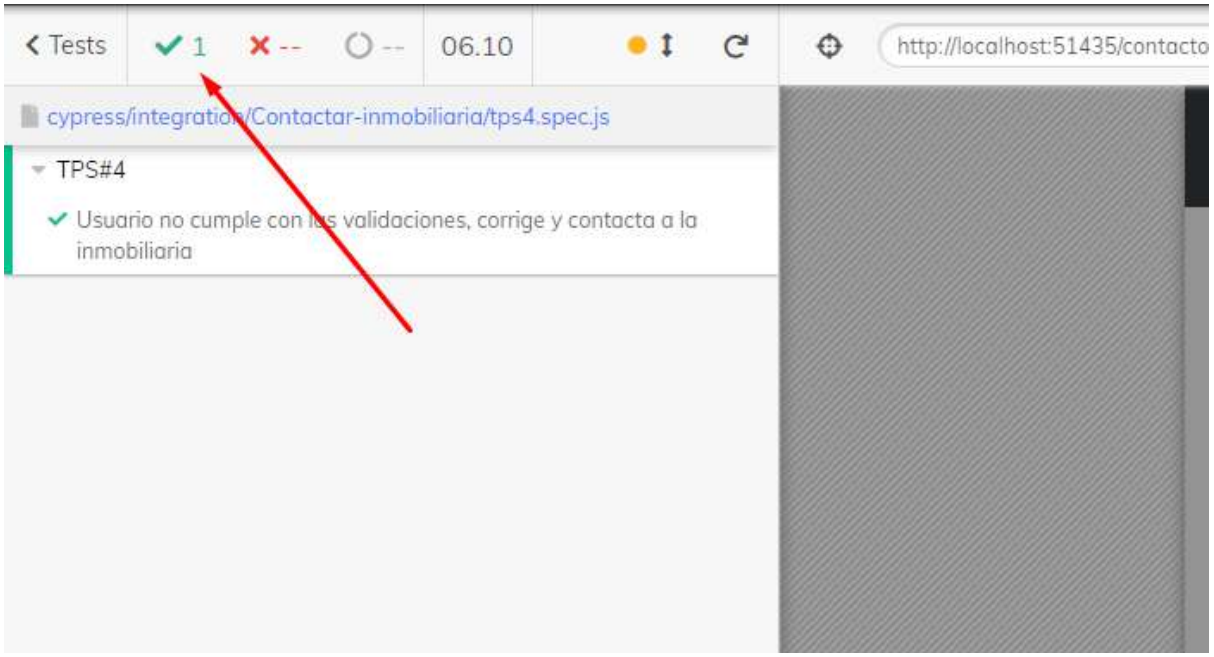


Fig. D10. Captura de pantalla de la ejecución de TPS#4 de la funcionalidad contactar inmobiliaria por correo electrónico.

Anexo E: Modificar publicación

Anexo E.1: Bases de pruebas

Tabla E1. Historia de usuario de funcionalidad modificar publicación.

Historia de usuario	
Número: 3	**Usuario:** administrador
Nombre de historia: modificar publicación	
Prioridad en negocio: alta	**Riesgo en desarrollo:** medio
Puntos estimados: 10	**Iteración asignada:** 2
Descripción: quiero poder modificar una publicación para mantenerla actualizada.	
Observaciones: para poder realizar correctamente la funcionalidad deben cumplirse ciertas validaciones (ver Tabla 13) en los formularios.	
Criterios de aceptación: ● Solo es posible modificar una publicación cuando se cumplen las validaciones de los formularios. En caso contrario, el sistema debe emitir un mensaje de error y no debe permitir avanzar al siguiente paso.	

Tabla E2. Tabla de publicaciones preexistentes en la base de datos de prueba.

Publicación 1
Id: 1
Datos de ubicación
Provincia: “La Pampa”
Localidad: “General Pico”
Dirección: “calle 110 esquina 7”
Iframe:
Datos de propiedad
Tipo de propiedad: “Departamento”
Ambientes: 2
Dormitorios: 1
Baños: 1
Garajes:
Servicio sanitario y municipal:
Expensas:
Antigüedad:
Superficie total:
Superficie cubierta:
Medidas:
Unidad de medidas:
Datos de servicios
Servicios: [“Agua corriente”]
Datos de publicación:
Título: “Publicación para TPS#1”
Tipo de operación: “Comprar”
Precio: 25000
Tipo de moneda: “U$S”
Disponible: True

Destacada: False
Acepta permuta: True
Acepta crédito: False
Descripción: “Una descripción para la publicación de TPS#1”
Datos de fotos
Foto destacada: “../pruebas fotos/d1.jpeg”
Fotos de exposición: [“../pruebas fotos/h1.jpeg”, “../pruebas fotos/h2.jpeg”, “../pruebas fotos/h3.jpeg”, “../pruebas fotos/h4.jpeg”, “../pruebas fotos/h5.jpeg”, “../pruebas fotos/h6.jpeg”, “../pruebas fotos/h7.jpeg”, “../pruebas fotos/h8.jpeg”]
Publicación 2
Id: 2
Datos de ubicación
Provincia: “Córdoba”
Localidad: “La Cumbrecita”
Dirección: “calle 108 esquina 9”
Iframe: “<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d202.5853881251418!2d-63.769722947351354!3d-35.667983821529 84!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95c363aee37d8599%3A0x548ef9fbf8aa69b5!2sFacultad%20de%20Ingenier%C3%ADa%20-%20UNLPam!5e0!3m2!1ses-419!2sar!4v1650491546387!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>”
Datos de propiedad
Tipo de propiedad: “Oficina”
Ambientes: 2
Dormitorios:
Baños: 1
Garajes:
Servicio sanitario y municipal: 2000
Expensas: 5000
Antigüedad: “A estrenar”
Superficie total: 300

Superficie cubierta: 280
Medidas: 30 x 10
Unidad de medidas: m²
Datos de servicios
Servicios: [“Agua corriente”, “Cloacas”, “Luz eléctrica”]
Datos de publicación:
Título: “Publicación para TPS#2”
Tipo de operación: “Alquilar”
Precio: 25000
Tipo de moneda: “$”
Disponible: True
Destacada: False
Acepta permuta: False
Acepta crédito: False
Descripción: “Una descripción para la publicación de TPS#2”
Datos de fotos
Foto destacada: “../pruebas fotos/d2.jpeg”
Fotos de exposición: [“../pruebas fotos/h9.jpeg”, “../pruebas fotos/h10.jpeg”]
Publicación 3
Id: 3
Datos de ubicación
Provincia: “Córdoba”
Localidad: “Alta Gracia”
Dirección: “calle 110 esquina 9”
Iframe:
Datos de propiedad
Tipo de propiedad: “Terreno”
Ambientes:
Dormitorios:

Baños:
Garajes:
Servicio sanitario y municipal: 5000
Expensas:
Antigüedad: “A estrenar”
Superficie total: 500
Superficie cubierta:
Medidas: 10 x 50
Unidad de medidas: m²
Datos de servicios
Servicios: [“Agua corriente”, “Cloacas”, “Luz eléctrica”]
Datos de publicación:
Título: “Publicación para TPS#3”
Tipo de operación: “Comprar”
Precio: 25000
Tipo de moneda: “U$S”
Disponible: True
Destacada: False
Acepta permuta: True
Acepta crédito: True
Descripción: “Una descripción para la publicación de TPS#3”
Datos de fotos
Foto destacada: “../pruebas fotos/d3.jpeg”
Fotos de exposición: [“../pruebas fotos/h1.jpeg”, “../pruebas fotos/h2.jpeg”, “../pruebas fotos/h3.jpeg”, “../pruebas fotos/h4.jpeg”]
Publicación 4
Id: 4
Datos de ubicación
Provincia: “Buenos Aires”

Localidad: “Bahía Blanca”
Dirección: “calle 110 esquina 7”
Iframe: “<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3286.325773174226!2d-58.451963584771875!3d-34.54530618047464!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95bcb43ae6018ddf%3A0x3d7f60a75bfa308a!2sEstadio%20Monumental%20Antonio%20Vespucio%20Liberti!5e0!3m2!1ses-419!2sar!4v1650492830512!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>”
Datos de propiedad
Tipo de propiedad: “Casa”
Ambientes: 3
Dormitorios: 2
Baños: 1
Garajes: 1
Servicio sanitario y municipal: 2000
Expensas: 2000
Antigüedad: “Menor a 10 años”
Superficie total: 250
Superficie cubierta: 200
Medidas: 10 x 25
Unidad de medidas: m²
Datos de servicios
Servicios: [“Agua corriente”, “Cloacas”, “Gas natural”, “Luz eléctrica”]
Datos de publicación:
Título: “Publicación para TPS#4”
Tipo de operación: “Alquilar”
Precio: 40000
Tipo de moneda: “$”
Disponible: True
Destacada: True

Acepta permuta: False
Acepta crédito: False
Descripción: “Una descripción para la publicación de TPS#4”
Datos de fotos
Foto destacada: “../pruebas fotos/d4.jpeg”
Fotos de exposición: [“../pruebas fotos/h5.jpeg”, “../pruebas fotos/h6.jpeg”, “../pruebas fotos/h7.jpeg”, “../pruebas fotos/h8.jpeg”]
Publicación 5
Id: 5
Datos de ubicación
Provincia: “La Pampa”
Localidad: “General Pico”
Dirección: “calle 9 esquina 16”
Iframe:
Datos de propiedad
Tipo de propiedad: “Casa”
Ambientes: 4
Dormitorios: 2
Baños: 1
Garajes: 1
Servicio sanitario y municipal: 2000
Expensas: 2000
Antigüedad: “Menor a 10 años”
Superficie total: 250
Superficie cubierta: 200
Medidas: 10 x 25
Unidad de medidas: m²
Datos de servicios
Servicios: [“Agua corriente”, “Cloacas”, “Gas natural”, “Luz eléctrica”]

Datos de publicación:
Título: “Publicación para TPS#5”
Tipo de operación: “Alquilar”
Precio: 45000
Tipo de moneda: “$”
Disponible: True
Destacada: True
Acepta permuta: False
Acepta crédito: False
Descripción: “Una descripción para la publicación de TPS#5”
Datos de fotos
Foto destacada: “../pruebas fotos/d5.jpeg”
Fotos de exposición: [“../pruebas fotos/h9.jpeg”, “../pruebas fotos/h10.jpeg”]

Anexo E.2: Requisitos de prueba

Tabla E3. Requisitos de prueba de la funcionalidad modificar publicación.

Label: TR-System-ModificarPublicacion.
Statement: se requiere verificar la correctitud de la funcionalidad que permite modificar una publicación existente.
Testable Entity Phase: desarrollo.
Test level: sistema
Completion Criteria: se debe desarrollar al menos un caso de prueba para cada una de las siguientes situaciones:

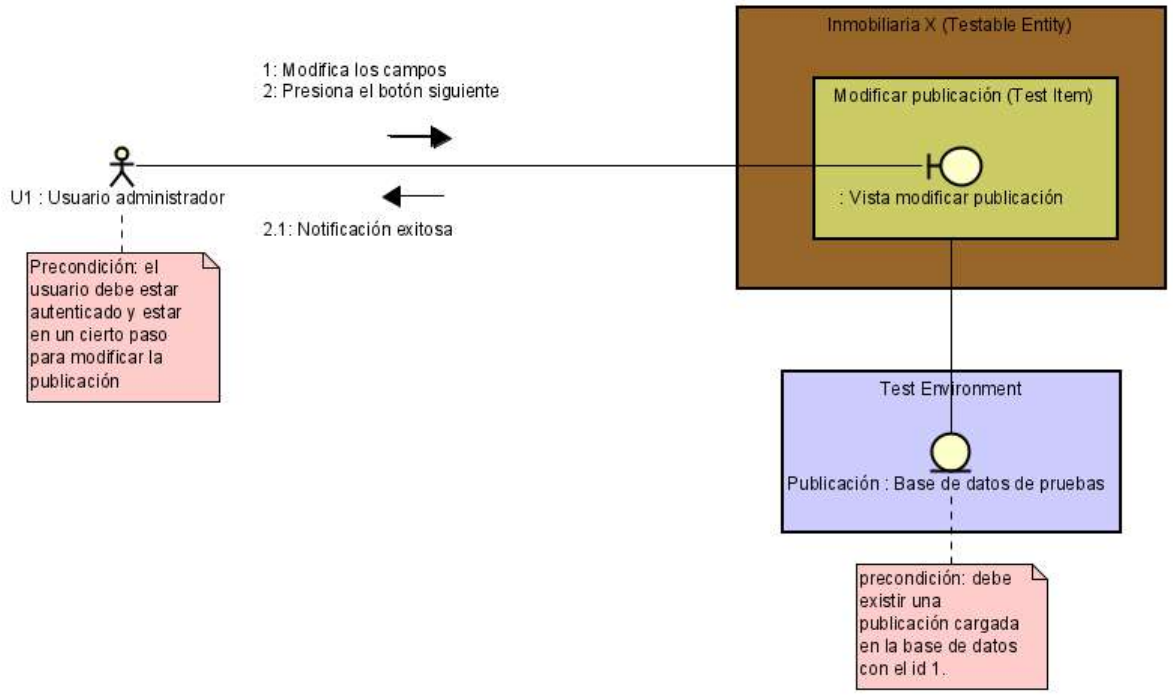
- El usuario administrador solo modifica satisfactoriamente los datos que aparecen en el formulario de un paso determinado (notar que un paso puede ser ubicación, propiedad, servicios, publicación, fotos).
- El usuario administrador realiza modificaciones en todos los pasos del formulario de forma satisfactoria.
- El usuario administrador modifica una publicación y, antes de finalizar, decide retroceder a un paso anterior para modificar otros campos. Luego, estando en ese paso anterior, no logra satisfacer las validaciones del formulario y ,finalmente, cancela la operación.
- El usuario, en un cierto paso, modifica la información e intenta avanzar al paso siguiente pero no cumple alguna validación de formulario. Luego, visualiza su error y lo corrige, y modifica la publicación exitosamente.
- El usuario se encuentra en un determinado paso de la publicación y lo modifica exitosamente y avanza al siguiente paso. Luego, retrocede y modifica correctamente el mismo paso anterior del formulario.

Refers to Testable Entity: Inmobiliaria X.
Refers to Test Context Entities: Base de datos de pruebas.

Anexo E.3: Especificaciones de las situaciones particulares de prueba y casos de pruebas

Tabla E4. Especificación de la TPS#1 de la funcionalidad iniciar sesión junto a los casos de pruebas.

Test Particular Situation:
TPS#1
-positive statement:
El usuario administrador se encuentra en un determinado paso (ubicación, propiedad, servicios, publicación, fotos), lo modifica, avanza y debe identificar una notificación que le indique que su operación se realizó con éxito.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en un cierto paso (ubicación, propiedad, servicios, publicación, fotos) para modificar la publicación de la publicación anteriormente mencionada.
-postconditions: los datos de la publicación fueron modificados correctamente en la base de datos.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad modificar publicación.
Test Target: funcionalidad modificar publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene las publicaciones ya registradas que se van a modificar, y a su vez es donde se guardarán los cambios.

Test Requirement: TR-System-ModificarPublicacion (Tabla E3).

Test Basis:

- Diagrama de casos de uso (Fig. 4).
- Wireframe de modificar publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad modificar publicación (Tabla E1).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).
- Tabla de publicaciones cargadas en base de datos (Tabla E2).

Test Cases:

TC#1.1
-input:
direccion: “Domicilio modificado”,
localidad: “General Alvear”,
provincia: “Mendoza”,
iframe: “<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d51235.979605250046!2d-64.32483082089848!3d-36.6203992!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95c2cd07f50bb34f%3A0x65387ab5a09745f1!2sUniversidad%20Nacional%20De%20La%20Pampa!5e0!3m2!1ses-419!2sar!4v1642173005142!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe>”
Presiona el botón siguiente en el formulario de ubicación.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.
.
-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y debe estar en el paso de ubicación en la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla de modificar publicación.

TC#1.2
-input:
tipoPropiedad: “Oficina”,
ambientes: 2,
dormitorios: 2,
banos: 2,
garaje: 2,
servicioSanitarioMunicipal: 1111,
expensas: 1111,
antiguedad: “Mayor a 50 años”,
superficieTotal: 300,
superficieCubierta: 250,
medidas: “10x30”,
unidadMedidas: “m²”
Presiona el botón siguiente en el formulario de propiedad.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver

Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y debe estar en el paso de propiedad en la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de servicios en la pantalla de modificar publicación.

TC#1.3
-input:
servicios: [“Gas natural”]
Presiona el botón siguiente en el formulario servicios

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y debe estar en el paso de servicios en la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de publicación en la pantalla de modificar publicación.

TC#1.4
-input:
titulo: “Test FR#3 TPS#1 1 Modificado”,
tipoOperacion: “Comprar”,
tipoMoneda: “$”,
precio: 0,
descripcion: “Descripción Test FR#3 TPS#1 1 Modificada”
Presiona el botón siguiente en el formulario de publicación.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y debe estar en el paso de publicación en la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de fotos en la pantalla de modificar publicación.

TC#1.5
-input:
fotoDestacada: “../pruebas fotos/d1.jpeg”,
fotosExposicion: [“../pruebas fotos/h1.jpeg”],
Presiona el botón finalizar en el formulario de fotos.

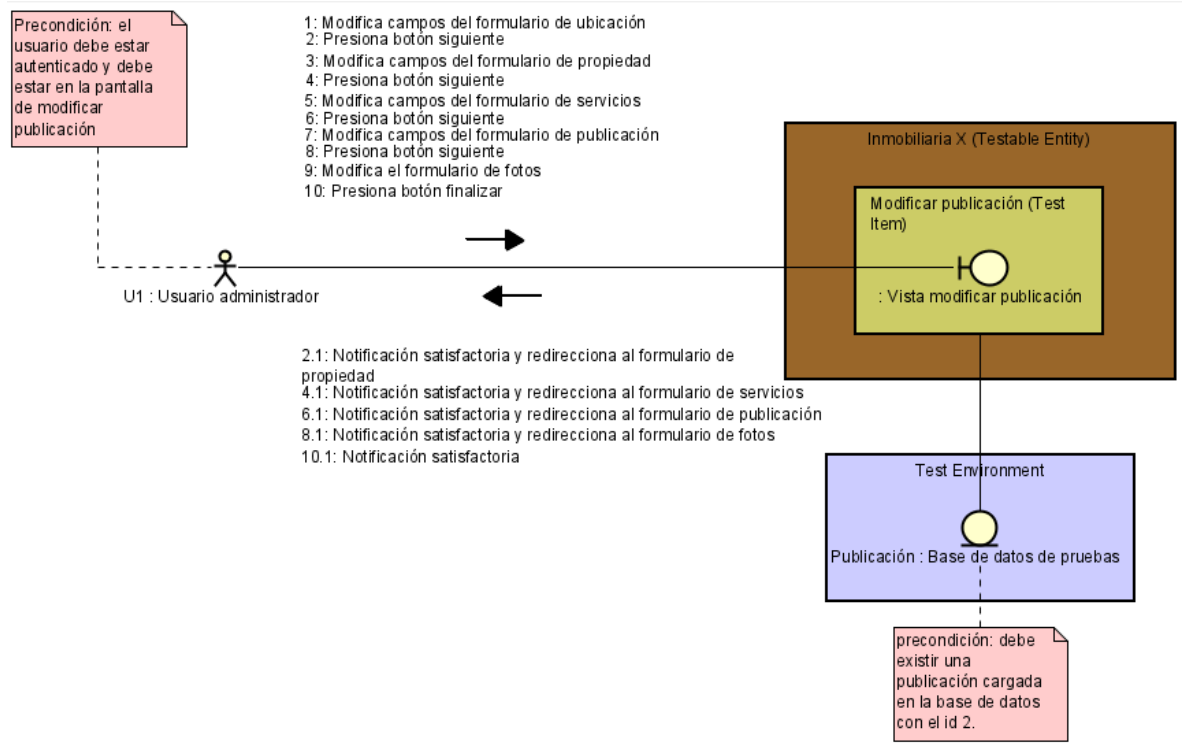
-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 1 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y debe estar en el paso de fotos en la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente.

Tabla E5. Especificación de la TPS#2 de la funcionalidad iniciar sesión junto a los casos de pruebas.

Test Particular Situation:
TPS#2
-positive statement:
El usuario administrador se encuentra en la pantalla de modificar publicación, realiza cambios en todos los formularios y finaliza satisfactoriamente la operación de modificación.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: debe existir una publicación cargada en la base de datos con id 2 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en un cierto paso (ubicación, propiedad, servicios, publicación, fotos) para modificar la publicación anteriormente mencionada.
-postconditions: los datos de la publicación fueron modificados correctamente en la base de datos.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad modificar publicación.
Test Target: funcionalidad modificar publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene las publicaciones ya registradas que se van a modificar, y a su vez es donde se guardarán los cambios.

Test Requirement: TR-System-ModificarPublicacion (Tabla E3).

Test Basis:

- Diagrama de casos de uso (Fig. 4).
- Wireframe de modificar publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad modificar publicación (Tabla E1).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).
- Tabla de publicaciones cargadas en base de datos (Tabla E2).

Test Cases:

TC#2.1
-input:
provincia: “Córdoba”,
localidad: “Almafuerte”,
direccion: “Test FR#3 TC#2.1 modificado”
Presiona el botón siguiente en el formulario de ubicación.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 2 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en el paso de ubicación para modificar la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla de modificar publicación.

TC#2.2
-input:
tipoPropiedad: “Quinta”,
ambientes: 3,
dormitorios: 3,
banos: 3,
garajes: 3,
servicioSanitarioMunicipal: 3000,
expensas: 3000,
antiguedad: “A estrenar”,
superficieTotal: 200,
superficieCubierta: 180,
medidas: “10 X 20”,
unidadMedidas: “m²”
Presiona el botón siguiente en el formulario de propiedad.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: se debe cumplir TC#2.1.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de servicios en la pantalla de modificar publicación.

TC#2.3
-input:
  servicios: [“Agua corriente”]
Presiona el botón siguiente en el formulario de servicios.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: se debe cumplir TC#2.2.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de publicación en la pantalla de modificar publicación.

TC#2.4
-input:
  titulo: “Test FR#3 TPS#2 TC#2.4 modificado”,
  tipoOperacion: “Comprar”,
  precio: “35555”,
  tipoMoneda: “U$S”,
  disponible: true,
  destacada: true,
  aceptaPermuta: true,
  aceptaCredito: true,
  descripcion: “Descripcion FR#3 TPS#2 TC#2.4 modificada”
Presiona el botón siguiente en el formulario de publicación.

-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: se debe cumplir TC#2.3.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de fotos en la pantalla de modificar publicación.

TC#2.5
-input:
  fotoDestacada: “../pruebas fotos/d1.jpeg”,
  fotosExposicion: [“../pruebas fotos/h1.jpeg”]
Presiona el botón finalizar en el formulario de fotos.

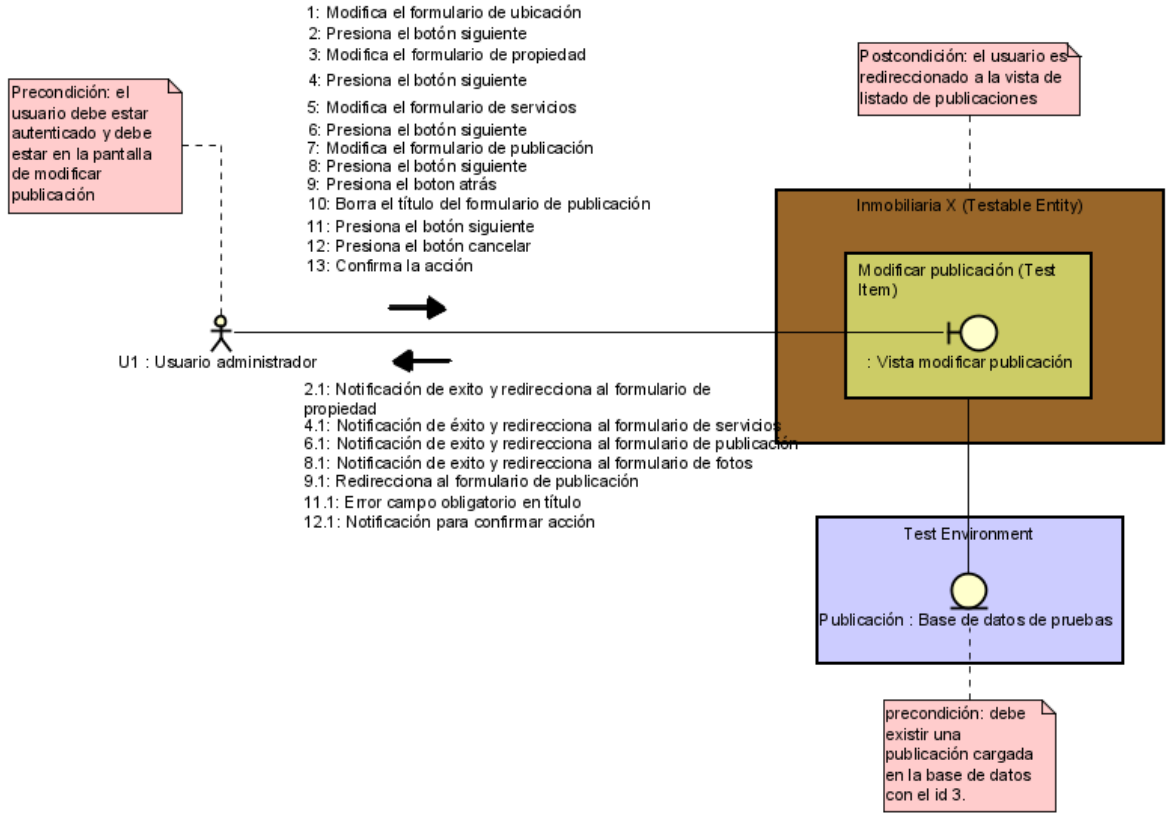
-expected result: los datos que el usuario modificó se encuentran cargados en la base de datos.

-preconditions: se debe cumplir TC#2.4.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente.

Tabla E6. Especificación de la TPS#3 de la funcionalidad iniciar sesión junto a los casos de pruebas.

Test Particular Situation:
TPS#3
-positive statement:
El usuario administrador modifica una publicación y antes de finalizar la operación decide retroceder a un paso anterior para modificar otros campos. Luego, estando en ese paso anterior, no logra satisfacer las validaciones del formulario y, finalmente, cancela la operación. Notar que, al cancelar la operación, los cambios realizados en los pasos anteriores se realizan, no pasando lo mismo con los últimos cambios realizados en el paso donde se cancela la operación.
- (Particular) Situation Model Specification:



-Conditions:
-preconditions: debe existir una publicación cargada en la base de datos con id 3 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en la pantalla de modificar publicación de la publicación anteriormente mencionada.
-postconditions: el usuario administrador es redireccionado a la vista de publicaciones.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad modificar publicación.
Test Target: funcionalidad modificar publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene las publicaciones ya registradas que se van a modificar, y a su vez es donde se guardarán los cambios.

Test Requirement: TR-System-ModificarPublicacion (Tabla E3).

Test Basis:

- Diagrama de casos de uso (Fig. 4).
- Wireframe de modificar publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad modificar publicación (Tabla E1).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).
- Tabla de publicaciones cargadas en base de datos (Tabla E2).

Test Cases:

TC#3.1
-input:
  direccion: “Test FR#3 TPS#3 TC#3.1 modificado”.
Presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 3 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en el paso de ubicación de la pantalla de modificar publicación de la publicación anteriormente mencionada.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla de modificar publicación.

TC#3.2
-input:
  dormitorios: 5
presiona el botón siguiente en el formulario de propiedad.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: se debe cumplir TC#3.1.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de servicios en la pantalla de modificar publicación.

TC#3.3
-input:
  servicios: [“Gas natural”]
Presiona el botón siguiente en el formulario de servicios.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: se debe cumplir TC#3.2.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de publicación en la pantalla de modificar publicación.

TC#3.4
-input:
  precio: 26000

Presiona el botón siguiente en el formulario de publicación.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: se debe cumplir TC#3.3.

-postconditions: el sistema emite una notificación indicando que la operación se realizó exitosamente y se encuentra en el paso de fotos en la pantalla de modificar publicación.

TC#3.5
-input:
Presiona el botón atrás.

-expected result: el sistema redirecciona al usuario al formulario de publicación.

-preconditions: se debe cumplir TC#3.4.

TC#3.6
-input:
titulo: “Test FR#3 TPS#3 TC#3.6 modificado Test FR#3 TPS#3 TC#3.6 modificado Test FR#3 TPS#3 TC#3.6 modificado”
Presiona el botón siguiente en el formulario de publicación.

-expected result: el sistema no carga esta modificación de la publicación en la base de datos debido a que no cumple con las validaciones del formulario de publicación.

-preconditions: se debe cumplir TC#3.5.

-postconditions: el usuario administrador se encuentra en el mismo paso con mensajes de error “Este campo admite hasta 100 caracteres”.

TC#3.7
-input:
Presiona el botón cancelar.

-expected result: el sistema emite una notificación indicando al usuario que confirme la acción de cancelar la operación.

-preconditions: se debe cumplir TC#3.6.

TC#3.8
-input:
Presiona el botón “Si, cancelar”.

-expected result: el sistema cancela la operación actual, es decir, no guarda los últimos cambios realizados en el paso de publicación.

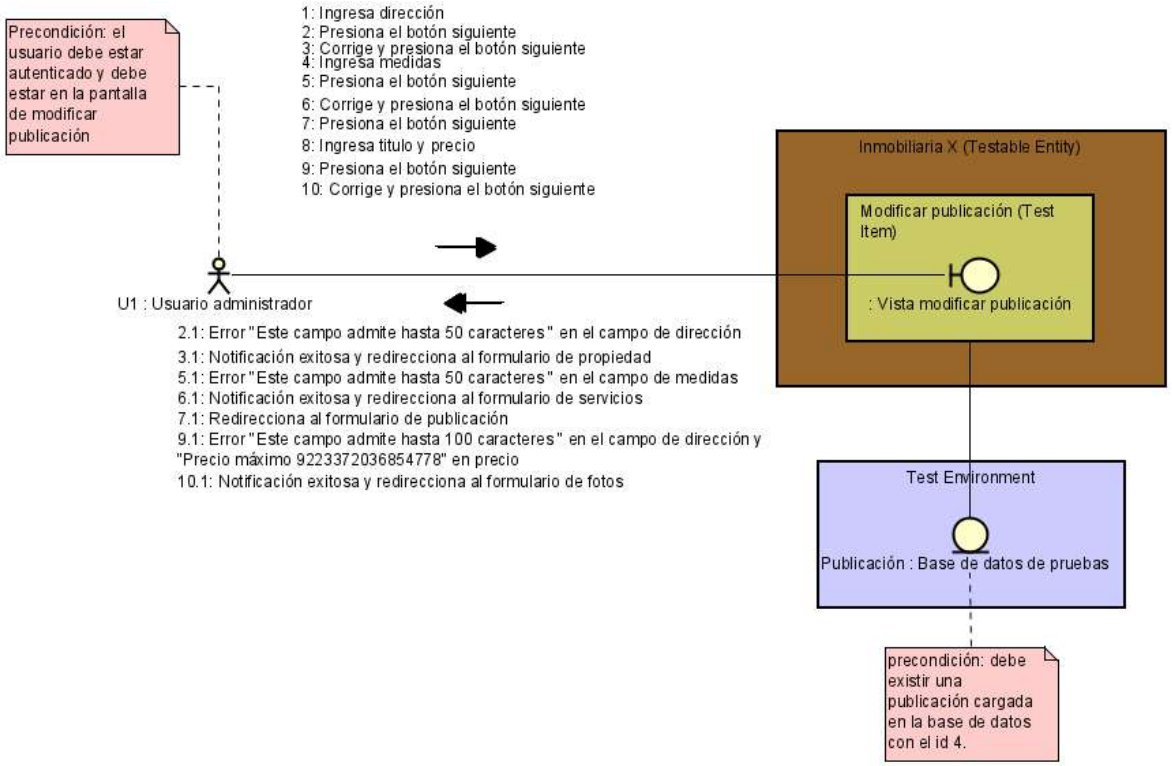
-preconditions: se debe cumplir TC#3.7.

-postconditions: el usuario es redireccionado a la vista de publicaciones.

Tabla E7. Especificación de la TPS#4 de la funcionalidad iniciar sesión junto a los casos de pruebas.

Test Particular Situation:
TPS#4
-positive statement:
El usuario, en un cierto paso (ubicación, propiedad o publicación), modifica la información e intenta avanzar al paso siguiente pero no cumple alguna validación de formulario. Luego, visualiza su error y lo corrige, y modifica la publicación exitosamente.

- (Particular) Situation Model Specification:



-Conditions:
-preconditions: debe existir una publicación cargada en la base de datos con id 4 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en el paso de ubicación para modificar la publicación anteriormente mencionada.
-postconditions: los cambios de la publicación se encuentran almacenados correctamente en la base de datos.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad modificar publicación.
Test Target: funcionalidad modificar publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene las publicaciones ya registradas que se van a modificar, y a su vez es donde se guardarán los cambios.

Test Requirement: TR-System-ModificarPublicacion (Tabla E3).

Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de modificar publicación (Fig. 11 y Anexo C.1).

- Historia de usuario de la funcionalidad modificar publicación (Tabla E1).
- DER de publicación (Fig. 12).
- Validaciones de formulario de publicación (Tabla 13).
- Tabla de publicaciones cargadas en base de datos (Tabla E2).

Test Cases:
TC#4.1
-input:
direccion: “Test FR#3 TPS#4 TC#4.1 Modificado Test FR#3 TPS#4 TC#4.1 Modificado”.
Presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema no modifica la publicación en la base de datos debido a que el usuario no cumple con las validaciones del formulario.

-preconditions: debe existir una publicación cargada en la base de datos con id 4 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en el paso de ubicación para modificar la publicación anteriormente mencionada.

-postconditions: el usuario administrador identifica un mensaje de error “Este campo admite hasta 50 caracteres” en el campo de dirección.

TC#4.2
-input:
direccion: “Test FR#3 TPS#4 TC#4.2 modificado”.
Presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: se debe cumplir TC#4.1.

-postconditions: el sistema muestra una notificación indicando que la operación se realizó exitosamente y redirecciona al formulario de propiedad.

TC#4.3
-input:
medidas: “12345678901234567890123 4 x 12345678901234567890123 4”
Presiona el botón siguiente en el formulario de propiedad.

-expected result: el sistema no modifica la publicación en la base de datos debido a que el usuario no cumple con las validaciones del formulario.

-preconditions: se debe cumplir TC#4.2.

-postconditions: el usuario administrador identifica un mensaje de error “Este campo admite hasta 50 caracteres” en el campo medidas.

TC#4.4
-input:
medidas: “12345 x 12345”
Presiona el botón siguiente en el formulario de propiedad.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: se debe cumplir TC#4.3.

-postconditions: el sistema muestra una notificación indicando que la operación se realizó exitosamente y redirecciona al formulario de servicios.

TC#4.5
-input:
Presiona el botón siguiente en el formulario de servicios.

-expected result: el sistema redirecciona al usuario al formulario de publicación.

-preconditions: se debe cumplir TC#4.4.

TC#4.6
-input:
titulo: “Test FR#3 TPS#4 TC#4.5 Modificado Test FR#3 TPS#4 TC#4.5 Modificado Test FR#3 TPS#4 TC#4.5 Modificado”,
precio: 99999999999999999

Presiona el botón siguiente en el formulario de publicación.

-expected result: el sistema no modifica la publicación en la base de datos debido a que el usuario no cumple con las validaciones del formulario.

-preconditions: se debe cumplir TC#4.5.

-postconditions: el usuario administrador identifica los mensajes de error “Este campo admite hasta 100 caracteres” en el campo de título, y “Precio máximo 9223372036854778” en precio.

TC#4.7
-input:
titulo: “Test FR#3 TPS#4 TC#4.5 Modificado”,
precio: 99999
Presiona el botón siguiente en el formulario de publicación.

-expected result: el sistema modifica la publicación en la base de datos.

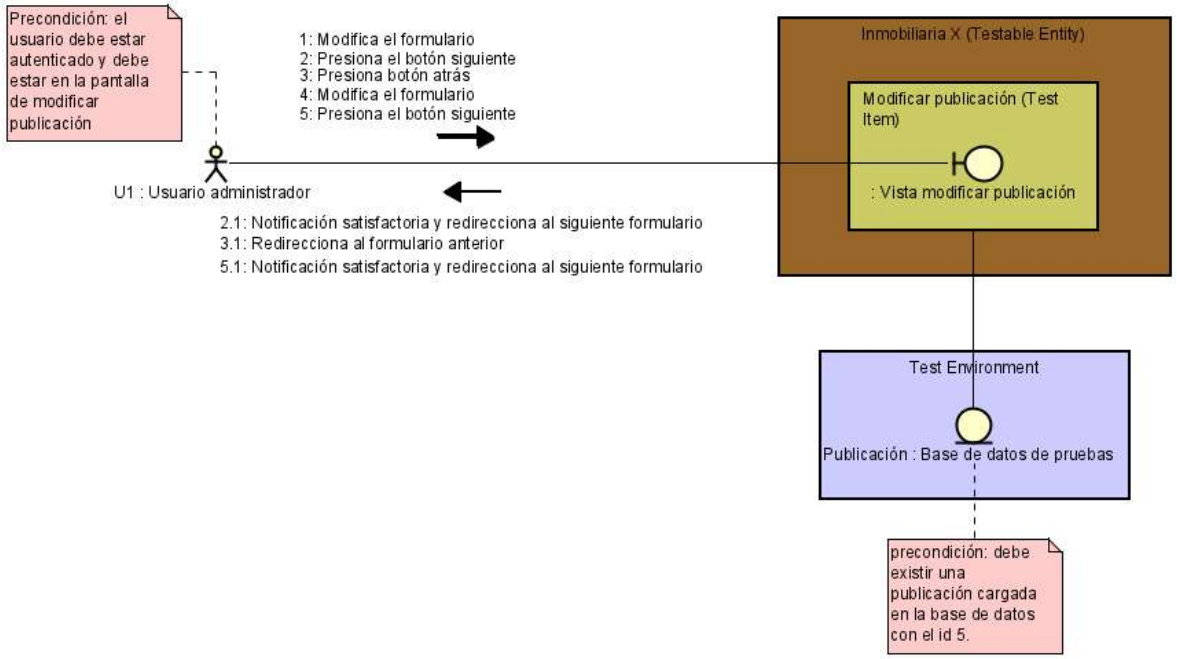
-preconditions: se debe cumplir TC#4.6.

-postconditions: el sistema muestra una notificación indicando que la operación se realizó exitosamente y redirecciona al formulario de fotos.

Tabla E8. Especificación de la TPS#5 de la funcionalidad iniciar sesión junto a los casos de pruebas.

Test Particular Situation:
TPS#5
-positive statement:
El usuario se encuentra en un determinado paso de la publicación y la modifica exitosamente y avanza al siguiente paso. Luego, retrocede y vuelve a modificar correctamente el mismo paso anterior del formulario.

- (Particular) Situation Model Specification:



-Conditions:
-preconditions: debe existir una publicación cargada en la base de datos con id 5 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en un cierto paso (ubicación, propiedad, servicios, publicación) para modificar la publicación anteriormente mencionada.
-postconditions: los cambios de la publicación se encuentran almacenados correctamente en la base de datos.

Testable Entity: Inmobiliaria X.
Test Item: funcionalidad modificar publicación.
Test Target: funcionalidad modificar publicación.

Test Context Entities: base de datos de pruebas.

-influences: la base de datos de pruebas contiene las publicaciones ya registradas que se van a modificar, y a su vez es donde se guardarán los cambios.

Test Requirement: TR-System-ModificarPublicacion (Tabla E3).

Test Basis:
- Diagrama de casos de uso (Fig. 4).
- Wireframe de modificar publicación (Fig. 11 y Anexo C.1).
- Historia de usuario de la funcionalidad modificar publicación (Tabla E1).
- DER de publicación (Fig. 12).

- Validaciones de formulario de publicación (Tabla 13).
- Tabla de publicaciones cargadas en base de datos (Tabla E2).

Test Cases:
TC#5.1
-input:
direccion: “Test FR#3 TPS#5 TC#5.1 modificado”
Presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: debe existir una publicación cargada en la base de datos con id 5 (ver Tabla E2 en Anexo E.1). Además, el usuario debe estar autenticado y estar en el paso de ubicación para modificar la publicación anteriormente mencionada.

-postconditions: el sistema muestra una notificación indicando que la operación se realizó exitosamente y redirecciona al formulario de propiedad.

TC#5.2
-input:
Presiona el botón atrás en el formulario de propiedad.

-expected result: el sistema redirecciona al usuario al formulario de ubicación.

-preconditions: se debe cumplir TC#5.1

TC#5.3
-input:
iframe: “<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.741460074764!2d-63.7593793847418 84!3d-35.658740980199504!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x95c37d59e452f9b1%3A0x79116d9fbb9064b2!2sPlaza%20San%20Mart%C3%ADn!5e0!3m2!1ses-419!2sar!4v1651056981875!5m2!1ses-419!2sar" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>”
Presiona el botón siguiente en el formulario de ubicación.

-expected result: el sistema modifica la publicación en la base de datos.

-preconditions: TC#5.2.

-postconditions: el sistema muestra una notificación indicando que la operación se realizó exitosamente y redirecciona al formulario de propiedad.

Anexo E.4: Conjuntos de datos de entradas para las situaciones particulares de prueba

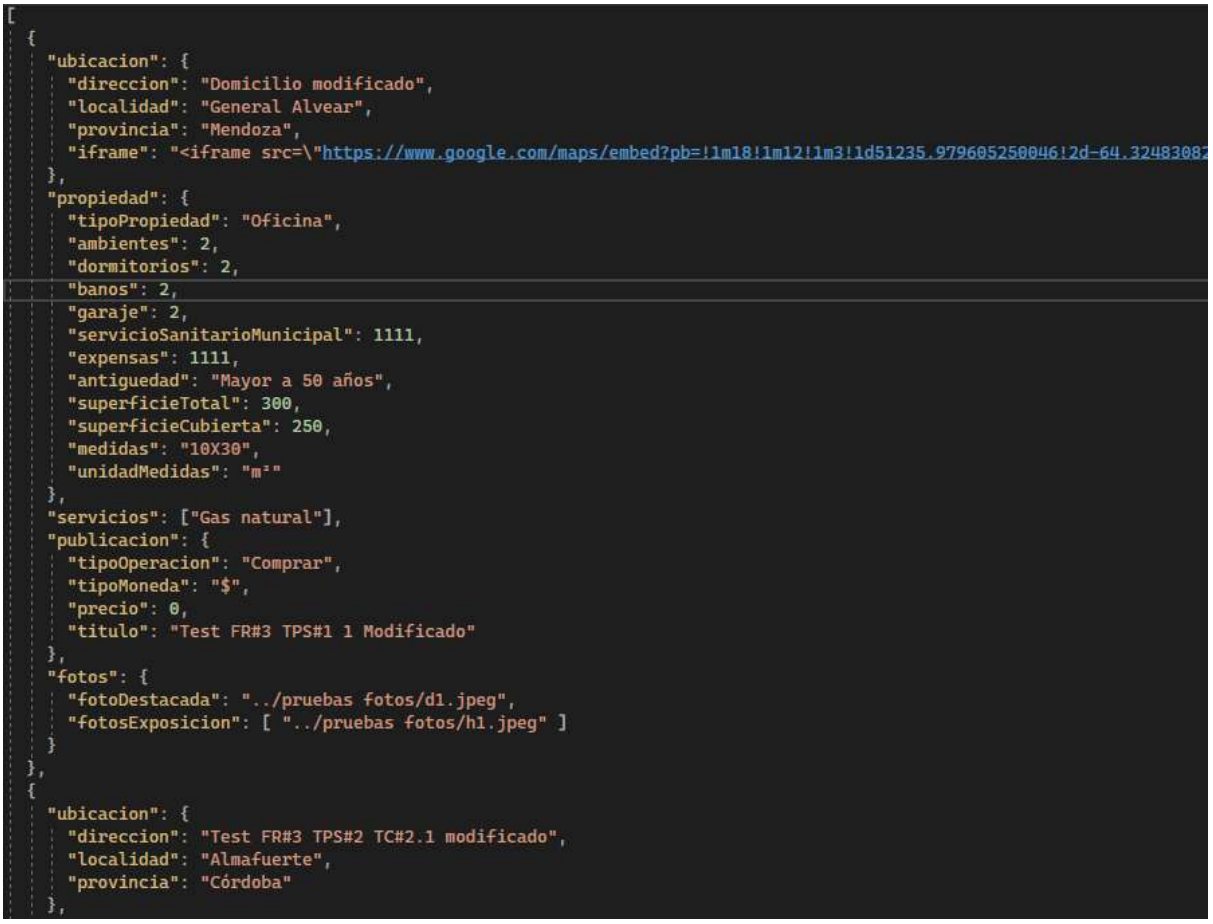


Fig. E9. Conjunto de datos de entrada para las situaciones particulares de prueba de modificar publicación.

```
  "propiedad": {
    "tipoPropiedad": "Quinta",
    "ambientes": 3,
    "dormitorios": 3,
    "banos": 3,
    "garajes": 3,
    "servicioSanitarioMunicipal": 3000,
    "expensas": 3000,
    "antiguedad": "A estrenar",
    "superficieTotal": 200,
    "superficieCubierta": 180,
    "medidas": "10x20",
    "unidadMedidas": "m²"
  },
  "servicios": ["Agua corriente"],
  "publicacion": {
    "titulo": "Test FR#3 TPS#2 TC#2.1 modificado",
    "tipoOperacion": "Comprar",
    "tipoMoneda": "U$S",
    "precio": 35555,
    "descripcion": "Descripcion FR#3 TPS#2 TC#2.4 modificada",
    "disponible": true,
    "permuta": true,
    "destacada": true,
    "credito": true
  },
  "fotos": {
    "fotoDestacada": "../pruebas fotos/d1.jpeg",
    "fotosExposicion": ["../pruebas fotos/h1.jpeg"]
  }
},
{
  "ubicacion": {
    "direccion": "Test FR#3 TPS#3 TC#3.1 modificado"
  },
  "propiedad": {
    "dormitorios": 5
  },
  "servicios": ["Gas natural"],
  "publicacion": {
    "precio": 26000,
    "titulo": "Test FR#3 TPS#3 TC#3.6 modificado Test FR#3 TPS#3 TC#3.6 modificado Test FR#3 TPS#3 TC#3.6 modificado"
  }
```


Fig. E10. Conjunto de datos de entrada para las situaciones particulares de prueba de modificar publicación.


```
},
{
  "ubicacion": {
    "direccion": "Test FR#3 TPS#4 TC#4.1 Modificado Test FR#3 TPS#4 TC#4.1 Modificado"
  },
  "ubicacionCorregida": {
    "direccion": "Test FR#3 TPS#4 TC#4.2 modificado"
  },
  "propiedad": {
    "medidas": "12345678901234567890123 4 X 123456789012345678901234"
  },
  "propiedadCorregida": {
    "medidas": "12345 X 12345"
  },
  "publicacion": {
    "titulo": "Test FR#3 TPS#4 TC#4.5 Modificado Test FR#3 TPS#4 TC#4.5 Modificado Test FR#3 TPS#4 TC#4.5 Modificado",
    "precio": 99999999999999999
  },
  "publicacionCorregida": {
    "titulo": "Test FR#3 TPS#4 TC#4.5 modificado",
    "precio": 99999
  }
},
{
  "ubicacion": {
    "iframe": "<iframe src=\"https//www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3241.741460074764!2d-63.7593793847418
    "direccion": "Test FR#3 TPS#5 TC#5.1 modificado"
  }
}
]
```


Fig. E11. Conjunto de datos de entrada para las situaciones particulares de prueba de modificar publicación.

Anexo E.5: Especificaciones de procedimientos de realización

```
describe('TPS#1', () => {
  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    this.idPublicacion = 1;
    cy.fixture('modificar-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: debe existir una publicación cargada en la base de datos
    con id 1(ver tabla de publicaciones cargadas en base de datos).Además,
    el usuario debe estar autenticado */
    cy.intercept('api/FullPublicacionView').as('publicacion');
    cy.visit('/login');
    cy.login('test', 'test');
    cy.wait(2000);
    cy.visit(`admin/modificarPublicacion/${this.idPublicacion}`);
    cy.url().should('contain', 'modificarPublicacion');
    //Esperamos el request de la publicacion
    cy.wait('@publicacion');
    //Esperamos que se carguen los valores en los formularios
    cy.wait(2000);
  })

  it("Modifcar ubicacion", function () {
    /* Precondicion: debe estar en el formulario de ubicacion */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
    cargarFormularioUbicacion(this.publicaciones[0].ubicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(this.publicaciones[0].ubicacion, publicacionBD.body);
    })
  })
```

Fig. E12. Especificación de procedimiento de realización de TPS#1 de la funcionalidad modificar publicación.

```
it("Modifcar propiedad", function () {
  cy.get('#boton-siguiente-web').click();
  /* Precondicion: debe estar en el formulario de propiedad */
  cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');
  cargarFormularioPropiedad(this.publicaciones[0].propiedad);
  cy.get('#boton-siguiente-web').click({ force: true });
  /* Postcondicion:  el sistema emite una notificación indicando que la operación
   se realizó exitosamente y se encuentra en el paso de servicios en la pantalla
   de modificar publicación */
  asercionNotificacionSatisfactoria();
  cy.get('#titulo-servicios').should('contain', 'Formulario de Servicios');
  /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
   en la base de datos.*/
  cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
  cy.get('@publicacionBD').then((publicacionBD) => {
    asercionesPropiedadBD(this.publicaciones[0].propiedad, publicacionBD.body);
  })
})

it("Modifcar servicios", function () {
  cy.get('#boton-siguiente-web').click();
  cy.get('#boton-siguiente-web').click({ force: true });
  /* Precondicion: debe estar en el formulario de servicios */
  cy.get('#titulo-servicios').should('contain', 'Formulario de Servicios');
  cargarFormularioServicios(this.publicaciones[0].servicios);
  cy.get('#boton-siguiente-web').click({ force: true });
  /* Postcondicion:  el sistema emite una notificación indicando que la operación
   se realizó exitosamente y se encuentra en el paso de publicacion en la pantalla
   de modificar publicación */
  asercionNotificacionSatisfactoria();
  cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
  /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
   en la base de datos.*/
  cy.request(`api/PropiedadxServicio/fields1?value0=${this.idPublicacion}`).as('serviciosPublicacionBD');
  cy.get('@serviciosPublicacionBD').then((serviciosPublicacionBD) => {
    asercionesServiciosBD(this.publicaciones[0].servicios, serviciosPublicacionBD.body);
  })
})
```

Fig. E13. Especificación de procedimiento de realización de TPS#1 de la funcionalidad modificar publicación.

```
it("Modifcar publicacion", function () {
    cy.get('#boton-siguiente-web').click();
    cy.get('#boton-siguiente-web').click({ force: true });
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Precondicion: debe estar en el formulario de publicacion */
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
    cargarFormularioPublicacion(this.publicaciones[0].publicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de fotos en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    cy.get('#titulo-fotos').should('contain', 'Formulario de Fotos');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPublicacionBD(this.publicaciones[0].publicacion, publicacionBD.body);
    })
  })

  it("Modificar fotos", function () {
    cy.get('#boton-siguiente-web').click();
    cy.get('#boton-siguiente-web').click({ force: true });
    cy.get('#boton-siguiente-web').click({ force: true });
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Precondicion: debe estar en el formulario de publicacion */
    cy.get('#titulo-fotos').should('contain', 'Formulario de Fotos');
    cargarFormularioFotos(this.publicaciones[0].fotos);
    cy.get('#boton-finalizar-web').click();
    /* Postcondicion:  el sistema emite una notificación indicando que la operación se
     realizó exitosamente y redirecciona a la pantalla del detalle de la publicación */
    asercionNotificacionSatisfactoria();
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/Foto/fields1?value0=${this.idPublicacion}`).as('fotosPublicacionBD');
    cy.get('@fotosPublicacionBD').then((fotosPublicacionBD) => {
      asercionesFotosBD(this.publicaciones[0].fotos, fotosPublicacionBD.body);
    })
  })
})
```

Fig. E14. Especificación de procedimiento de realización de TPS#1 de la funcionalidad modificar publicación.

```
describe('TPS#2', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    this.idPublicacion = 2;
    cy.fixture('modificar-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: debe existir una publicación cargada en la base de datos
    con id 2 (ver tabla de publicaciones cargadas en base de datos).Además,
    el usuario debe estar autenticado */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.intercept('api/FullPublicacionView').as('publicacion');
    cy.visit(`admin/modificarPublicacion/${this.idPublicacion}`);
    cy.url().should('contain', 'modificarPublicacion');
    //Esperamos el request de la publicacion
    cy.wait('@publicacion');
    //Esperamos que se carguen los valores en los formularios
    cy.wait(2000);
  })

  it("Modifica todos los pasos de la publicacion", function () {
    /* Precondicion: debe estar en el formulario de ubicacion */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
    cargarFormularioUbicacion(this.publicaciones[1].ubicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(this.publicaciones[1].ubicacion, publicacionBD.body);
    })
```

Fig. E15. Especificación de procedimiento de realización de TPS#2 de la funcionalidad modificar publicación.

```
    cargarFormularioPropiedad(this.publicaciones[1].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de servicios en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-servicios').should('contain', 'Formulario de Servicios');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPropiedadBD(this.publicaciones[1].propiedad, publicacionBD.body);
    })

    cargarFormularioServicios(this.publicaciones[1].servicios);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de publicacion en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/PropiedadxServicio/fields1?value0=${this.idPublicacion}`).as('serviciosPublicacionBD');
    cy.get('@serviciosPublicacionBD').then((serviciosPublicacionBD) => {
      asercionesServiciosBD(this.publicaciones[1].servicios, serviciosPublicacionBD.body);
    })

    cargarFormularioPublicacion(this.publicaciones[1].publicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de fotos en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-fotos').should('contain', 'Formulario de Fotos');
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
```

Fig. E16. Especificación de procedimiento de realización de TPS#2 de la funcionalidad modificar publicación.

```
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPublicacionBD(this.publicaciones[1].publicacion, publicacionBD.body);
    })

    cargarFormularioFotos(this.publicaciones[1].fotos);
    cy.get('#boton-finalizar-web').click();
    /* Postcondicion:  el sistema emite una notificación indicando que la operación se
     realizó exitosamente y redirecciona a la pantalla del detalle de la publicación */
    asercionNotificacionSatisfactoria();
    /* Resultado esperado: los datos que el usuario modificó se encuentran cargados
     en la base de datos.*/
    cy.request(`api/Foto/fields1?value0=${this.idPublicacion}`).as('fotosPublicacionBD');
    cy.get('@fotosPublicacionBD').then((fotosPublicacionBD) => {
      asercionesFotosBD(this.publicaciones[1].fotos, fotosPublicacionBD.body);
    })
  })
})
```

Fig. E17. Especificación de procedimiento de realización de TPS#2 de la funcionalidad modificar publicación.

```
describe('TPS#3', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    this.idPublicacion = 3;
    cy.fixture('modificar-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: debe existir una publicación cargada en la base de datos
    con id 3 (ver tabla de publicaciones cargadas en base de datos).Además,
    el usuario debe estar autenticado */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.intercept('api/FullPublicacionView').as('publicacion');
    cy.visit(`admin/modificarPublicacion/${this.idPublicacion}`);
    cy.url().should('contain', 'modificarPublicacion');
    //Esperamos el request de la publicacion
    cy.wait('@publicacion');
    //Esperamos que se carguen los valores en los formularios
    cy.wait(2000);
  })

  it("Modifica campos, retrocede y cancela cambios", function () {
    /* Precondicion: debe estar en el formulario de ubicacion */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
    cargarFormularioUbicacion(this.publicaciones[2].ubicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de propiedad en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');
    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(this.publicaciones[2].ubicacion, publicacionBD.body);
    })
```

Fig. E18. Especificación de procedimiento de realización de TPS#3 de la funcionalidad modificar publicación.

```
    cargarFormularioPropiedad(this.publicaciones[2].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de servicios en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-servicios').should('contain', 'Formulario de Servicios');
    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPropiedadBD(this.publicaciones[2].propiedad, publicacionBD.body);
    })

    cargarFormularioServicios(this.publicaciones[2].servicios);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de publicacion en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/PropiedadxServicio/fields1?value0=${this.idPublicacion}`).as('serviciosPublicacionBD');
    cy.get('@serviciosPublicacionBD').then((serviciosPublicacionBD) => {
      asercionesServiciosBD(this.publicaciones[2].servicios, serviciosPublicacionBD.body);
    })

    let publicacion1 = {
      precio: this.publicaciones[2].publicacion.precio
    }
    cargarFormularioPublicacion(publicacion1);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el sistema emite una notificación indicando que la operación
     se realizó exitosamente y se encuentra en el paso de fotos en la pantalla
     de modificar publicación */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-fotos').should('contain', 'Formulario de Fotos');
    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
```

Fig. E19. Especificación de procedimiento de realización de TPS#3 de la funcionalidad modificar publicación.

```
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPublicacionBD(publicacion1, publicacionBD.body);
    })

    cy.get('#boton-atras-web').click({ force: true })
    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');

    let publicacion2 = {
      titulo: this.publicaciones[2].publicacion.titulo
    }
    cargarFormularioPublicacion(publicacion2);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion: el usuario administrador se encuentra en el mismo paso con
     mensajes de error "* Campo obligatorio". */
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
    cy.get('#titulo-length').should('contain', "Este campo admite hasta 100 caracteres");

    cancelarPublicacion();
    /* Postcondicion: el usuario es redireccionado a la vista de publicaciones */
    cy.url().should('contain', 'listarPublicaciones');

    /* Resultado esperado: el sistema cancela la operación actual, es decir, no guarda
     los últimos cambios realizados en el paso de publicación. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      expect(publicacionBD.body.titulo).to.not.equal(publicacion2.titulo);
    })
  })
})
```

Fig. E20. Especificación de procedimiento de realización de TPS#3 de la funcionalidad modificar publicación.

```
describe('TPS#4', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    this.idPublicacion = 4;
    cy.fixture('modificar-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: debe existir una publicación cargada en la base de datos
    con id 4 (ver tabla de publicaciones cargadas en base de datos).Además,
    el usuario debe estar autenticado */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.intercept('api/FullPublicacionView').as('publicacion');
    cy.visit(`admin/modificarPublicacion/${this.idPublicacion}`);
    cy.url().should('contain', 'modificarPublicacion');
    //Esperamos el request de la publicacion
    cy.wait('@publicacion');
    //Esperamos que se carguen los valores en los formularios
    cy.wait(2000);
  })

  it("Modifica, tiene errores, corrije y modifica satisfactoriamente", function () {
    /* Precondicion: debe estar en el formulario de ubicacion */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
    cargarFormularioUbicacion(this.publicaciones[3].ubicacion);
    cy.get('#boton-siguiente-web').click({ force: true });

    /* Postcondicion: el usuario administrador identifica un mensaje de error "Este campo
     admite hasta 50 caracteres" en el campo de dirección.*/
    cy.get('#direccion-length').should('contain', "Este campo admite hasta 50 caracteres");

    /* Resultado esperado: el sistema no modifica la publicación en la base de datos debido a
     que el usuario no cumple con las validaciones del formulario. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      expect(publicacionBD.body.direccion).to.not.equal(this.publicaciones[3].ubicacion.direccion);
    })

    cargarFormularioUbicacion(this.publicaciones[3].ubicacionCorregida);
    cy.get('#boton-siguiente-web').click({ force: true });
```

Fig. E21. Especificación de procedimiento de realización de TPS#4 de la funcionalidad modificar publicación.

```
    /* Postcondicion: el sistema muestra una notificación indicando que la operación se realizó
     exitosamente y redirecciona al formulario de propiedad.*/
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');

    /* Resultado esperado: el sistema modifica la publicación en la base de datos.*/
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(this.publicaciones[3].ubicacionCorregida, publicacionBD.body);
    })

    cargarFormularioPropiedad(this.publicaciones[3].propiedad);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion:  el usuario administrador identifica un mensaje de error "Este
     campo admite hasta 50 caracteres" en el campo medidas. */
    cy.get('#medidas-length').should('contain', "Este campo admite hasta 50 caracteres");

    /* Resultado esperado: el sistema no modifica la publicación en la base de datos debido
     a que el usuario no cumple con las validaciones del formulario. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      expect(publicacionBD.body.medida).to.not.equal(this.publicaciones[3].propiedad.medidas);
    })

    cargarFormularioPropiedad(this.publicaciones[3].propiedadCorregida);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion: el sistema muestra una notificación indicando que la operación se
     realizó exitosamente y redirecciona al formulario de servicios.*/
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-servicios').should('contain', 'Formulario de Servicios');

    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPropiedadBD(this.publicaciones[3].propiedadCorregida, publicacionBD.body)
    })

    cy.get('#boton-siguiente-web').click({ force: true });
    /* Resultado esperado: el sistema redirecciona al usuario al formulario de publicación. */
    cy.get('#titulo-publicacion').should('contain', 'Formulario de Publicación');
```

Fig. E22. Especificación de procedimiento de realización de TPS#4 de la funcionalidad modificar publicación.

```
    cargarFormularioPublicacion(this.publicaciones[3].publicacion);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion: el usuario administrador identifica los mensajes de error "Este campo
     admite hasta 100 caracteres" en el campo de título, y "Precio máximo 9223372036854778"
     en precio. */
    cy.get('#titulo-length').should('contain', "Este campo admite hasta 100 caracteres");
    cy.get('#precio-maximo').should('contain', "Precio máximo 9223372036854778");
    /* Resultado esperado: el sistema no modifica la publicación en la base de datos
     debido a que el usuario no cumple con las validaciones del formulario. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      expect(publicacionBD.body.titulo).to.not.equal(this.publicaciones[3].propiedad.titulo);
      expect(publicacionBD.body.precio).to.not.equal(this.publicaciones[3].propiedad.precio);
    })

    cargarFormularioPublicacion(this.publicaciones[3].publicacionCorregida);
    cy.get('#boton-siguiente-web').click({ force: true });
    /* Postcondicion: el sistema muestra una notificación indicando que la operación
    se realizó exitosamente y redirecciona al formulario de fotos. */
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-fotos').should('contain', 'Formulario de Fotos');
    /* Resultado esperado:  el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesPublicacionBD(this.publicaciones[3].publicacionCorregida, publicacionBD.body);
    })
  })
})
```

Fig. E23. Especificación de procedimiento de realización de TPS#4 de la funcionalidad modificar publicación.

```
describe('TPS#5', () => {

  beforeEach(function () {
    //Arrange (Preparar)
    cy.disableUncaughtException();
    this.idPublicacion = 5;
    cy.fixture('modificar-publicacion').then(publicaciones => {
      this.publicaciones = publicaciones;
    });
    /* Precondicion: debe existir una publicación cargada en la base de datos
    con id 5 (ver tabla de publicaciones cargadas en base de datos).Además,
    el usuario debe estar autenticado */
    cy.visit('/login');
    cy.login('test', 'test');
    cy.intercept('api/FullPublicacionView').as('publicacion');
    cy.visit(`admin/modificarPublicacion/${this.idPublicacion}`);
    cy.url().should('contain', 'modificarPublicacion');
    //Esperamos el request de la publicacion
    cy.wait('@publicacion');
    //Esperamos que se carguen los valores en los formularios
    cy.wait(2000);
  })

  it("Modifica, retrocede y modifica de nuevo", function () {
    let ubicacion1 = { direccion: this.publicaciones[4].ubicacion.direccion };
    let ubicacion2 = { iframe: this.publicaciones[4].ubicacion.iframe };

    /* Precondicion: debe estar en el formulario de ubicacion */
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');
    cargarFormularioUbicacion(ubicacion1);
    cy.get('#boton-siguiente-web').click({ force: true });

    /* Postcondicion: el sistema muestra una notificación indicando que la
     operación se realizó exitosamente y redirecciona al formulario de propiedad.*/
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');

    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(ubicacion1, publicacionBD.body);
    })
```

Fig. E24. Especificación de procedimiento de realización de TPS#5 de la funcionalidad modificar publicación.

```
    cy.get('#boton-atras-web').click({ force: true });
    /* Resultado esperado: el sistema redirecciona al usuario al formulario de ubicación.*/
    cy.get('#titulo-ubicacion').should('contain', 'Formulario de Ubicación');

    cargarFormularioUbicacion(ubicacion2);
    cy.get('#boton-siguiente-web').click({ force: true });

    /* Postcondicion: el sistema muestra una notificación indicando que la
     operación se realizó exitosamente y redirecciona al formulario de propiedad.*/
    asercionNotificacionSatisfactoria();
    aceptarNotificacion();
    cy.get('#titulo-propiedad').should('contain', 'Formulario de Propiedad');

    /* Resultado esperado: el sistema modifica la publicación en la base de datos. */
    cy.request(`api/FullPublicacionView/${this.idPublicacion}`).as('publicacionBD');
    cy.get('@publicacionBD').then((publicacionBD) => {
      asercionesUbicacionBD(ubicacion2, publicacionBD.body);
    })
  })
})
```

Fig. E25. Especificación de procedimiento de realización de TPS#5 de la funcionalidad modificar publicación.

Anexo E.6: Requisitos del entorno de prueba

Tabla E26. Requisitos del entorno de prueba para la funcionalidad nueva publicación.

Entidad de contexto de prueba	Requisitos del entorno de prueba
Base de datos de prueba.	Debe tener contenidos en las tablas de Antiguedad, Localidad, Operacion, Provincia, Servicio, TipoMoneda, TipoPropiedad y TipoSuperficie. Deben existir las cinco instancias de publicaciones mencionadas en la tabla 27.

Anexo E.7: Entidad de contexto de prueba configurada

	IdPublicacion	Titulo	IdPropiedad	Descripcion	IdOperacion	Destacada	Disponible	Permutable	IdTipoMoneda	AptoCredito	Precio
1	1	Publicación para TPS#1	1	Una descripción para la publicación de TPS#1	1	0	1	1	2	0	25000
2	2	Publicación para TPS#2	2	Una descripción para la publicación de TPS#2	2	0	1	0	1	0	25000
3	3	Publicación para TPS#3	3	Una descripción para la publicación de TPS#3	1	0	1	1	2	1	25000
4	4	Publicación para TPS#4	4	Una descripción para la publicación de TPS#4	2	1	1	0	1	0	40000
5	5	Publicación para TPS#5	5	Una descripción para la publicación de TPS#5	2	1	1	0	1	0	45000

Fig. E27. Entidad de contexto de prueba configurada para la funcionalidad modificar publicación.

Anexo E.8: Resultados de pruebas

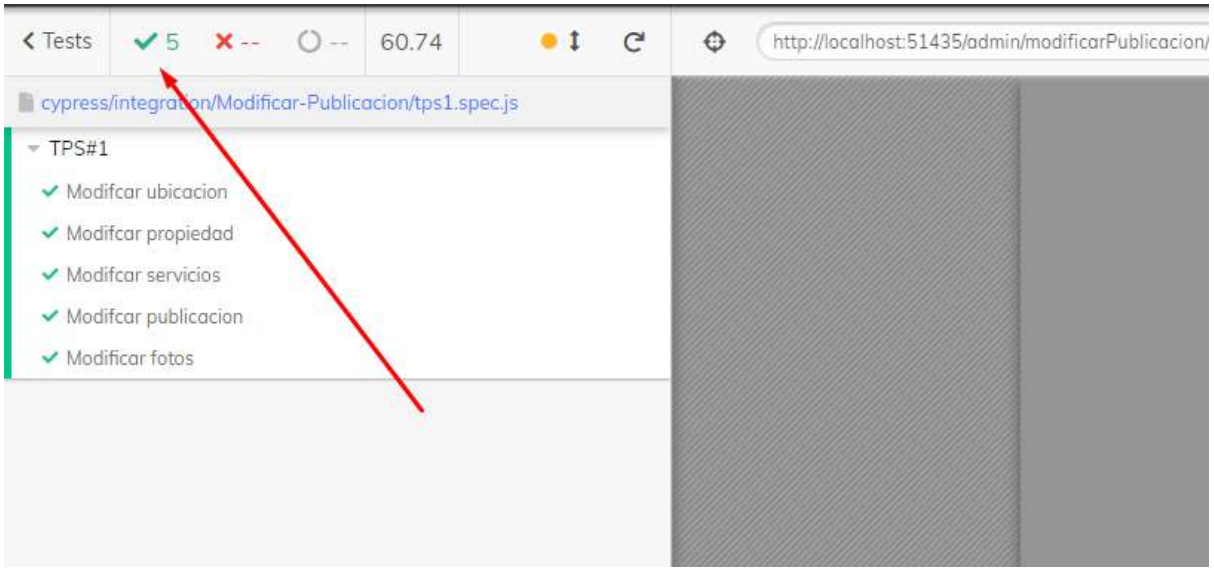


Fig. E28. Captura de pantalla de la ejecución de TPS#1 de la funcionalidad modificar publicación.

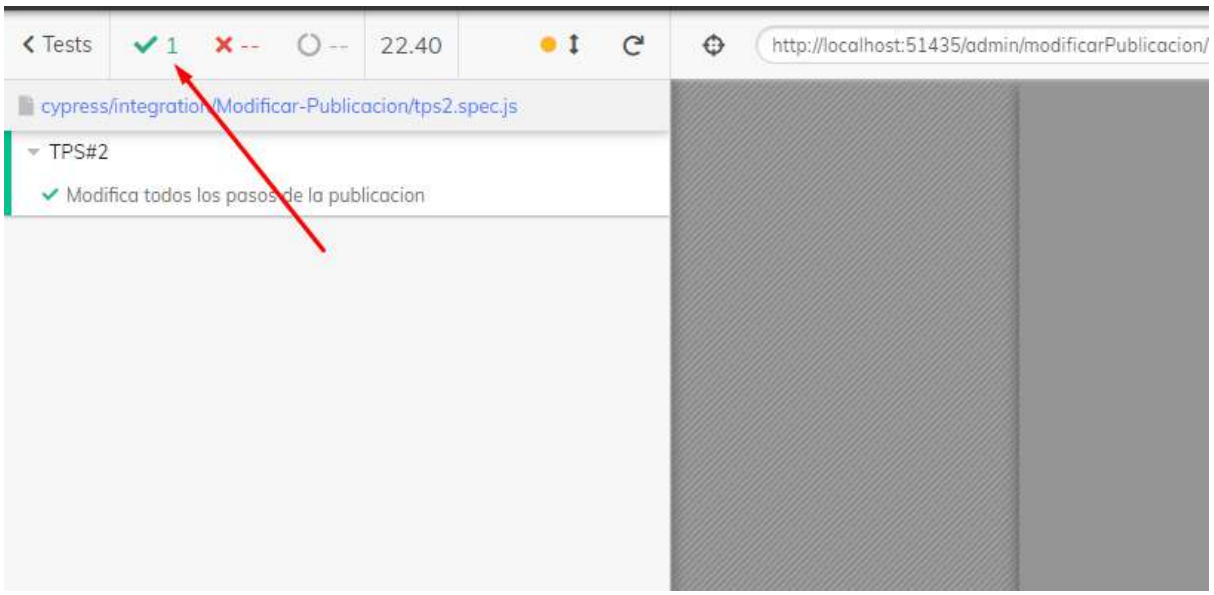


Fig. E29. Captura de pantalla de la ejecución de TPS#2 de la funcionalidad modificar publicación.

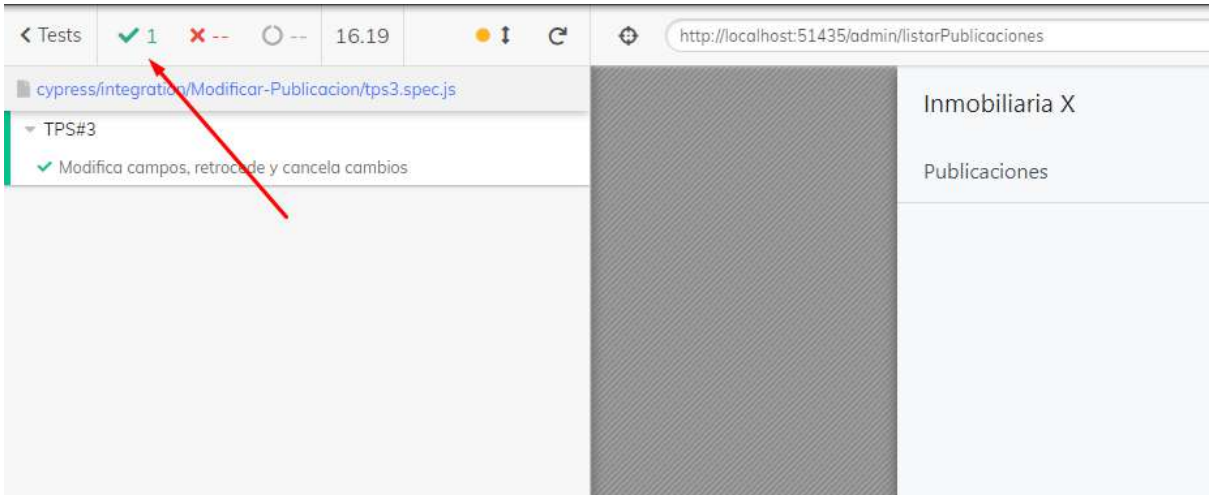


Fig. E30. Captura de pantalla de la ejecución de TPS#3 de la funcionalidad modificar publicación.

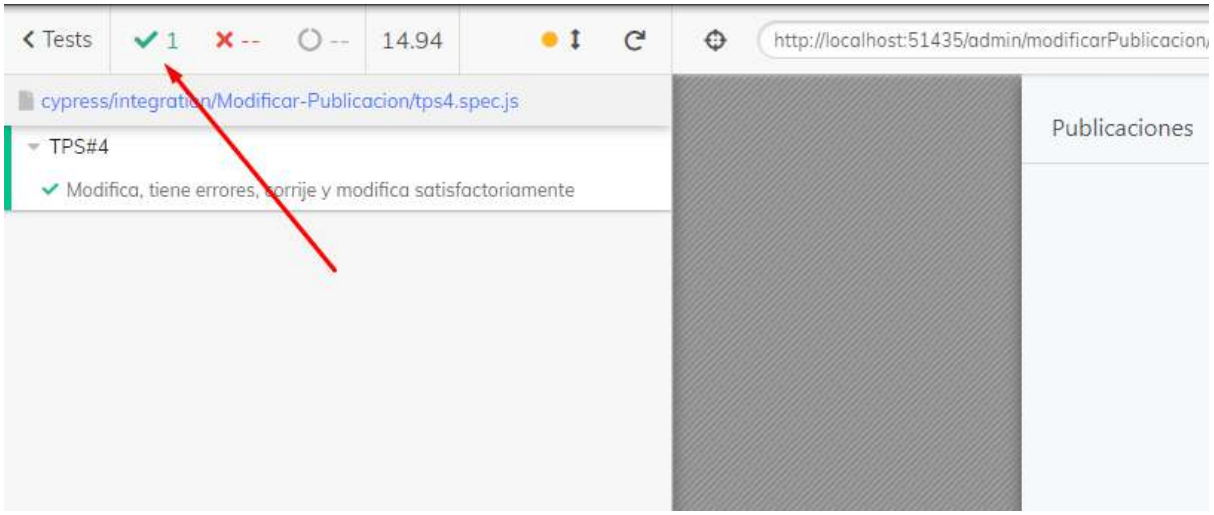


Fig. E31. Captura de pantalla de la ejecución de TPS#4 de la funcionalidad modificar publicación.

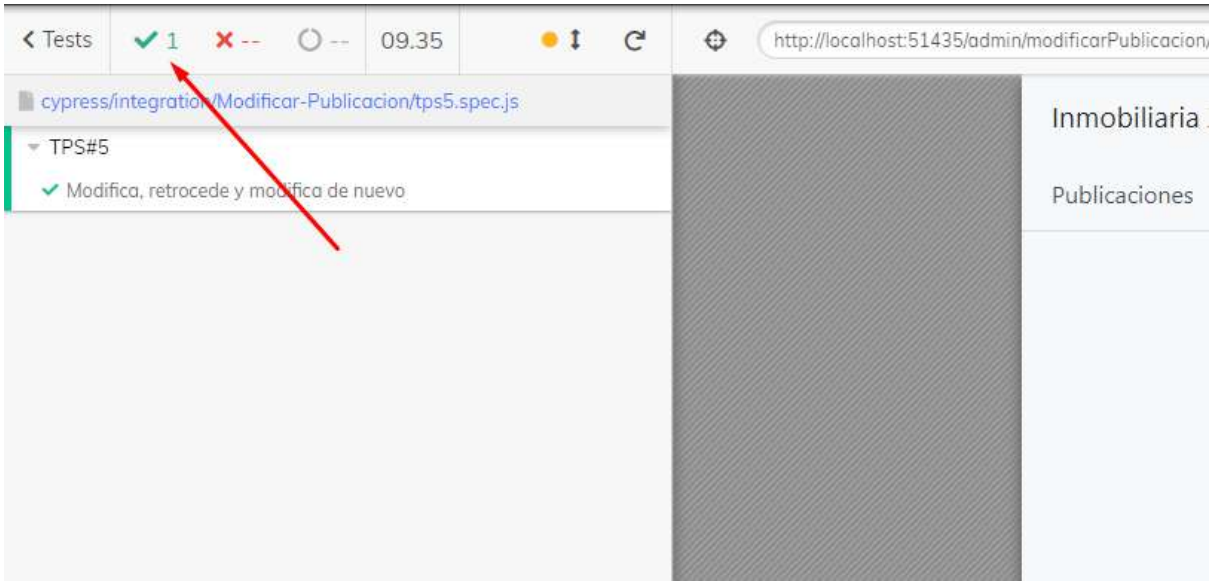


Fig. E32. Captura de pantalla de la ejecución de TPS#5 de la funcionalidad modificar publicación.