Diseño e implementación de pruebas para una aplicación en Angular

Mateo Castro Muñoz

Informe de práctica para optar al título de Ingeniero de Sistemas
otorgado por Universidad de Antioquia

Asesores
Deisy Loaiza Berrío, Ingeniera de Sistemas
León Darío Arango Amaya, Ingeniero de Sistemas

Universidad de Antioquia
Facultad de Ingeniería, Departamento de Ingeniería de Sistemas
Ingeniería de Sistemas
Medellín, Colombia
2024

Cita	(Castro Muñoz, 2024)
Referencia	[1] Castro Muñoz M. (2024). *Diseño e implementación de pruebas para una aplicación en Angular* [Semestre de Industria]. Universidad de Antioquia, Medellín.

Estilo APA 7 (2020)

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia
Sistema de Bibliotecas

Centro de Documentación Ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes.

Decano/Director: Julio Saldarriaga..

Jefe departamento: Diego José Botia Valderrama.

Tabla de contenido

Lista de figuras

Siglas, acrónimos y abreviaturas

AAA	Arrange, Act, Assert
Aprox.	Aproximadamente
IC	Integración contínua
S.	Segundos
TI	Tecnologías de la información
VSCode	Visual Studio Code

Resumen

Mis Aliados es un proyecto resultante de una alianza entre Bancolombia y SURA, cuyo objetivo es el de servir como intermediario para que las personas puedan contratar servicios ofertados por los trabajadores independientes de Colombia. Para esto, entre otros frentes, Mis Aliados cuenta con un aplicativo web responsive desarrollado en Angular, mediante el que los usuarios pueden buscar y contratar los servicios que necesiten. Se pretende desarrollar un plan de pruebas frontend para aumentar la cobertura en este frente y agregar documentación a sus componentes, haciendo uso de Storybook.

Palabras clave: cobertura, pruebas, Angular, Storybook.

Abstract

Mis Aliados is a project resulting from an alliance between Bancolombia and SURA, whose objective is to serve as an intermediary so that people can contract services offered by independent workers in Colombia. For this, among other modules, Mis Aliados has a responsive web application developed in Angular, through which users can search and hire the services they need. The aim is to develop a frontend testing plan to increase coverage on this module and add documentation to its components, using Storybook.

*Keywords***:** coverage, testing, Angular, Storybook.

Introducción

Una de las apuestas más grandes de las empresas en la actualidad es la de facilitar la adquisición de sus servicios, apuntando a su alta accesibilidad por medio de plataformas con la finalidad de satisfacer de manera sencilla las necesidades de sus clientes y de paso atraer nuevos interesados potenciales.

Con esta finalidad en mente, se torna de vital importancia mantener y garantizar la estabilidad de la plataforma con la que los usuarios interactúan, teniendo en cuenta que cuando se presenta un error, se pueden generar malas experiencias o, en los peores casos, la total inoperatividad de esta.

En este contexto toman protagonismo las pruebas. Dependiendo de la complejidad de la plataforma o proyecto en cuestión, estas pueden ser ejecutadas de manera manual previa a cada despliegue, sin embargo, esta deja de ser una opción viable cuando se trata de un proyecto robusto con grandes cantidades de usuarios y funcionalidades. En estos casos, la alternativa es la implementación de pruebas automatizadas integradas en el pipeline de release, donde son ejecutadas y se espera sean exitosas como requisito para un despliegue exitoso cada vez que se agreguen o se modifiquen los módulos existentes.

El proyecto en el que se realizaron las pruebas planteadas fue el resultado de una alianza entre Bancolombia y SURA llamado **Mis Aliados**, una plataforma cuya finalidad fue la de actuar como intermediario entre los trabajadores independientes de Colombia y usuarios o clientes quienes pudiesen requerir de sus servicios, que podían ser para el hogar como plomería, carpintería, pintura, entre otros, como para mascotas, negocio, etcétera.

Mis Aliados sufrió cambios considerables a lo largo de sus aproximados 7 años de operación, durante los que, desde luego, también se evolucionó en las tecnologías sobre las que se desarrolló el proyecto y en los módulos que lo conformaban, tales como la página web responsive para los usuarios, la página web correspondiente al módulo o panel de administración, aplicación móvil para los independientes prestadores de servicios, backend y servicios de nube.

En este proyecto se tuvo la finalidad desarrollar una serie de pruebas automatizadas que debían ejecutarse cada vez que se agregara o modificara algún módulo de la página web responsive (desarrollada en Angular) que proveía a los usuarios, mediante las que se propuso aumentar la cobertura de código y así, tener la posibilidad de desplegar los cambios teniendo la seguridad de que pasaron por sus respectivas pruebas previamente y de manera exitosa.

1 Objetivos

1.1 Objetivo general

Alcanzar cobertura de pruebas del 50% del proyecto mediante la implementación de un conjunto de pruebas automatizadas y documentación de componentes en los flujos y vistas principales de la página web en pro de mejorar en los procesos de calidad relacionados con los despliegues de todos los ambientes.

1.2 Objetivos específicos

- Generar documentación de componentes más relevantes del sitio web.
- Diseñar e implementar pruebas de humo para los componentes más relevantes del sitio web.
- Diseñar conjunto de pruebas unitarias y de interacción para los componentes más relevantes del sitio web.
- Implementar pruebas unitarias y de interacción según fase de diseño.
- Automatizar conjuntos de pruebas.

2 Marco teórico

Cuando se quiere desarrollar un producto de software, no solo es importante identificar las necesidades de sus usuarios potenciales, sino también garantizar que estos obtengan una buena experiencia, constatando que el producto funciona según lo esperado desde sus primeras versiones y durante las posteriores. Para esto se implementan las pruebas de software, conjunto en el que destacan las pruebas de humo, unitarias, visuales, de integración, entre otras.

Las pruebas de humo se centran en determinar si el código de los diferentes módulos se compila de manera satisfactoria. Si se puede garantizar la compilación del código, se asegura que las demás pruebas podrán ser llevadas a cabo, siendo este el conjunto de pruebas más básico. (Palamarchuk, 2022).

Las pruebas unitarias se encargan de probar las partes más pequeñas o atómicas de código de manera individual, tales como métodos o funciones, con el fin de verificar si funcionan según lo esperado. (GeeksforGeeks, 2023).

Las pruebas visuales son las empleadas para garantizar el correcto funcionamiento de los elementos de la interfaz de usuario con la que los usuarios interactúan. (Battat, 2022)

Las pruebas de integración tienen como objetivo validar la interacción entre distintos módulos o componentes de un sistema. A diferencia de las pruebas unitarias, que se centran en unidades individuales de código, las pruebas de integración se enfocan en detectar problemas que pueden surgir cuando estas unidades interactúan entre sí. Estos problemas pueden incluir errores en la transferencia de datos, conflictos en la funcionalidad combinada y fallos en las interfaces. Es esencial realizar estas pruebas para asegurarse de que el sistema funciona de manera cohesiva como una entidad unificada. (Bourque, Dupuis, Abran, Moore & Tripp, 1999).

La automatización de este tipo de pruebas ha tomado un papel fundamental en el ámbito de la calidad de software. Las pruebas automatizadas no solo garantizan que las nuevas características de una aplicación funcionen según lo previsto, sino que también las

antiguas sigan funcionando correctamente. Esto es crucial para sistemas como **Mis Aliados**, que ha evolucionado a lo largo del tiempo y que integra diversos módulos en su evolución.

Las prácticas de IC se han consolidado en el mundo del desarrollo como una estrategia esencial para garantizar la entrega de código de calidad. A través de la IC, cada cambio en el código es automáticamente probado y validado, permitiendo a los equipos de desarrollo identificar y solucionar problemas con mayor rapidez. Al integrar pruebas automatizadas en este proceso, las empresas pueden asegurarse de que las actualizaciones o nuevas características no rompan la funcionalidad existente. (Rehkopf).

La cobertura de pruebas es una métrica que permite asegurar que sus pruebas pueden cubrir la mayoría de las funcionalidades del producto. Ayuda a monitorear la calidad de las pruebas y da una idea a los testers de si es necesario crear pruebas que cubran áreas que faltan o no están validadas. (Palamarchuk, 2022).

Angular es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página (SPA). Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.

Storybook es una herramienta diseñada para probar el frontend de una aplicación, por medio de la instanciación de componentes o páginas de una aplicación web de manera aislada. Es usada también para desarrollar y generar la documentación de la interfaz de usuario, es gratuita y de código abierto.

Storybook hace uso de las denominadas historias para llevar a cabo las pruebas de cada componente, siendo cada una de estas, una instancia o situación específica que requiere ser testeada.

3 Metodología

Para llevar a cabo la implementación de las pruebas y el cumplimiento de los objetivos se hará uso de Storybook, herramienta de testing que cuenta con una extensa documentación consignada en la página oficial, que servirá de apoyo durante cada fase de este proyecto.

Como primer paso, es necesario integrar Storybook al frontend de Mis Aliados. Esta integración, como el uso de la propia herramienta fueron diseñados para ser de baja complejidad y fácil adaptación por parte del tester, sin embargo, cabe resaltar la actual complejidad y robustez del proyecto en cuestión (Mis Aliados), ya que por las tecnologías que integra y los módulos que lo componen, la adaptación de una nueva tecnología puede llegar a ser desafiante de cara a las configuraciones necesarias para el correcto funcionamiento de Storybook.

Una vez integrado al frontend de Mis Aliados, se dará paso a la fase de experimentación, llevando a cabo ejemplos de documentación de componentes simples con la finalidad de ir incrementando de manera gradual el entendimiento de la herramienta.

Teniendo en cuenta que entre los objetivos de este proyecto se encuentra el de alcanzar una cobertura de pruebas del 50%, previo a comenzar a agregar la documentación al frontend, es necesario aplicar un filtro con el fin de identificar sus componentes y flujos más relevantes, a los que posteriormente se les aplicarán las pruebas. Una vez identificados, deberán identificarse las dependencias para cada uno de estos, ya que la finalidad de Storybook es la de renderizar cada componente de manera aislada al resto de la aplicación. Posteriormente, se dará comienzo a la generación de la documentación y pruebas dummies de los componentes más sencillos y que hayan sido los últimos desarrollados y agregados al frontend, así se garantiza que se tiene un contexto claro sobre el funcionamiento del componente.

El siguiente paso será el de implementar pruebas de humo en los componentes, con la finalidad de garantizar que tanto estos de manera independiente como las configuraciones y documentaciones iniciales consignadas en las historias de Storybook compilen de manera correcta, y así, dar paso a los demás conjuntos de pruebas.

Una vez definidas y ejecutadas de manera exitosa las pruebas de humo se diseñarán e implementarán cada una de las pruebas visuales, que corresponden cada una a una historia que buscará simular el comportamiento del componente en cuestión en cierta situación específica deseada. A su vez, cada una de estas historias tendrá asociadas pruebas de interacción (así denominadas en Storybook) cuya finalidad es la de corroborar de manera automática que los componentes y su lógica respondan de manera correcta a la situación planteada, simulando interacciones de usuarios tales como clics, llenados de formularios, scrolls, etcétera.

Habiendo implementado este plan de pruebas, finalmente se espera alcanzar la cobertura del 50% del código y poder automatizar su ejecución mediante un pipeline para certificar que cada una de las pruebas diseñadas será ejecutada como paso previo y requisito para cada despliegue en cualquiera de los ambientes de Mis Aliados.

4 Resultados

Los resultados obtenidos para este trabajo se pueden desglosar en 4 partes:

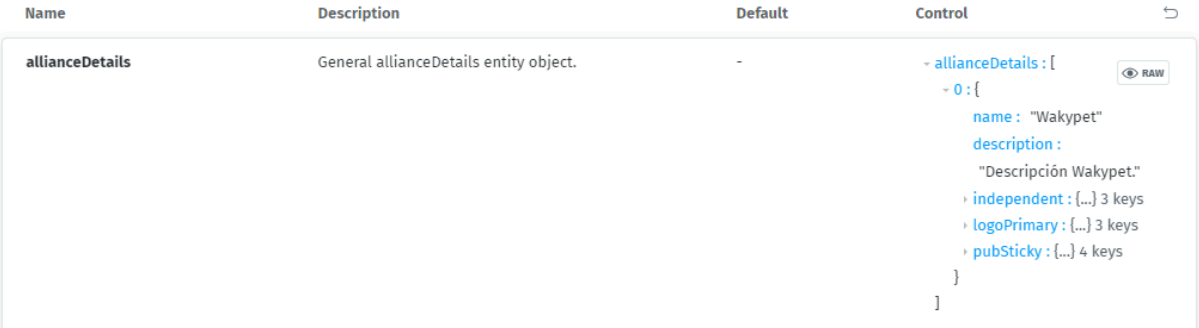
- Documentación
- Pruebas de humo
- Pruebas unitarias y de interacción
- Automatización y cobertura de pruebas

4.1 Documentación

La generación de la documentación corresponde al primer paso luego de haber identificado los componentes principales del proyecto. Esta no es más que una instancia de cada componente generada a partir de información ficticia, cuya única finalidad es la de obtener una vista previa, con la posibilidad de modificar de manera reactiva cada campo de la información recibida por el componente en cuestión.

Figura 1

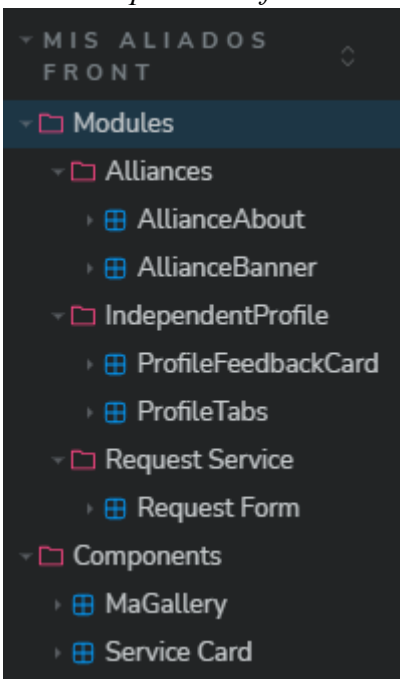
Controles para modificar datos en un componente



En la **Figura 1** se observan los controles para la documentación de la instancia de uno de los componentes, mediante los que es posible modificar la información que éste mostrará en pantalla. Cada instancia cuenta con sus propios controles, los cuales reciben diferentes tipos de datos u objetos, según sea el caso.

Figura 2

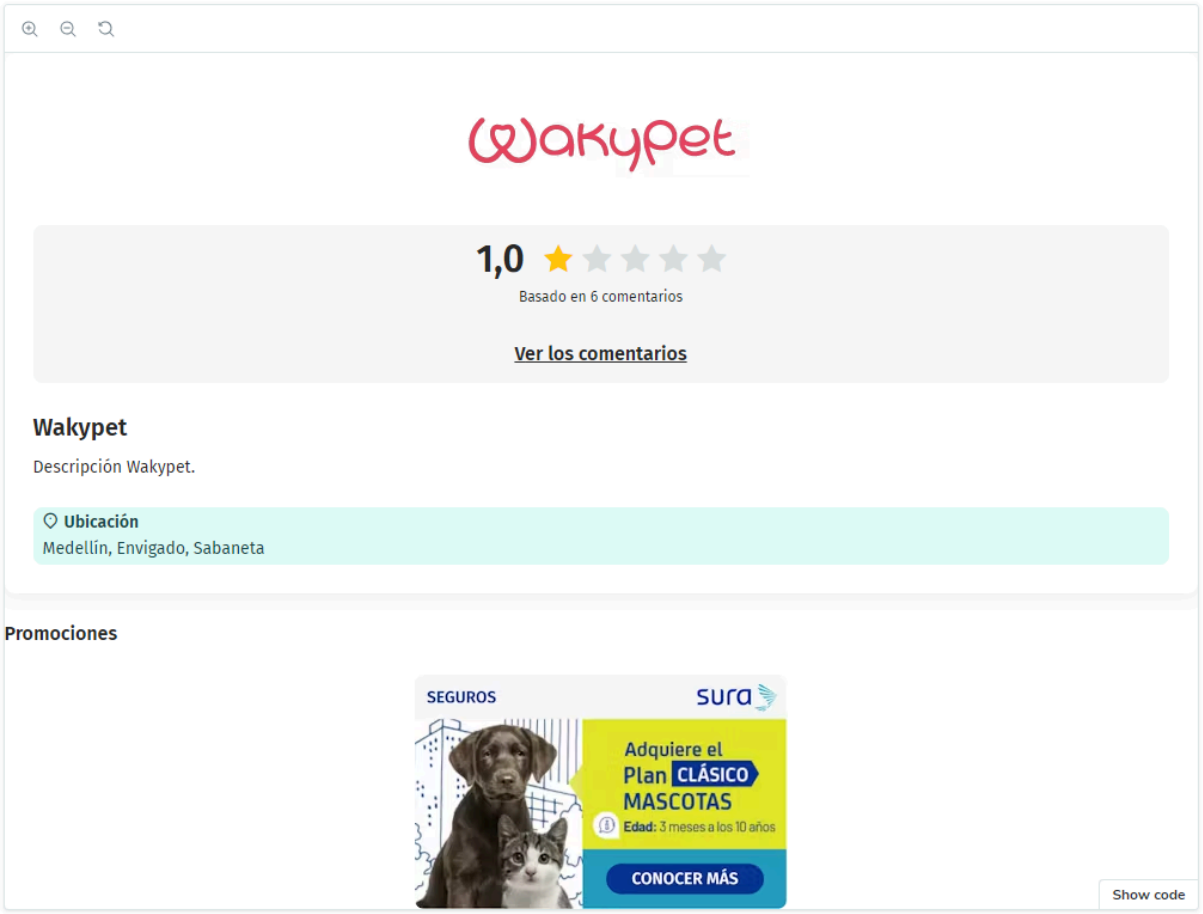
Controles para modificar datos en un componente



En la **Figura 2** pueden verse los componentes identificados como principales o más relevantes para el flujo general de **Mis Aliados**, para los que se desarrollaron las respectivas pruebas. Algunos de estos están separados acorde al módulo de la aplicación al que pertenecen, y otros, considerados transversales o compartidos, se encapsulan en un directorio general llamado *Components*. A continuación se presenta la vista previa de la documentación para cada uno de estos, si aplica.

Figura 3

Documentación componente AllianceAbout

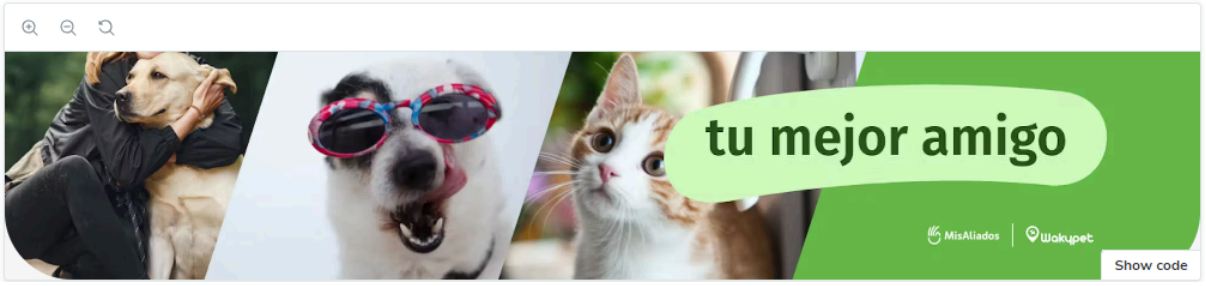


Este componente se usa en la vista principal de las alianzas[1] y su objetivo no es otro que el de brindar información general de la misma, tal como un logo principal, calificación según usuarios, nombre, descripción, entre otros campos como los que se pueden ver en la **Figura 1**.

[1] Son empresas con las que Mis Aliados establece una alianza, con la finalidad de ofrecer sus servicios mediante la plataforma.

Figura 4

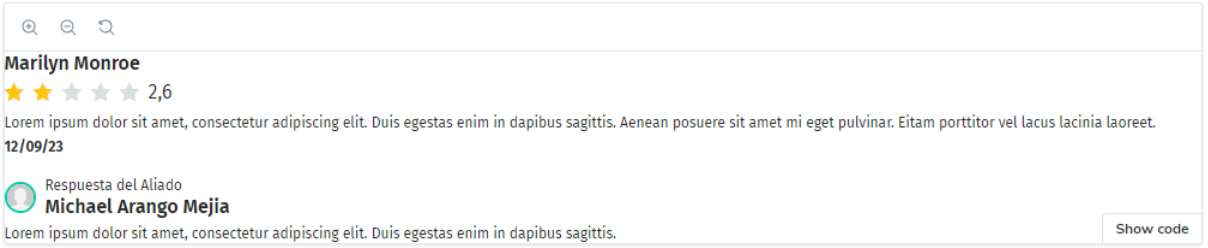
Documentación componente AllianceBanner



Este componente es utilizado en la vista principal de una alianza y es usado como banner publicitario.

Figura 5

Documentación componente ProfileFeedbackCard



Este componente tiene la finalidad de mostrar los comentarios y/o valoraciones de los usuarios respecto a un aliado o alianza de la plataforma, con posibilidad de mostrar también una respuesta.

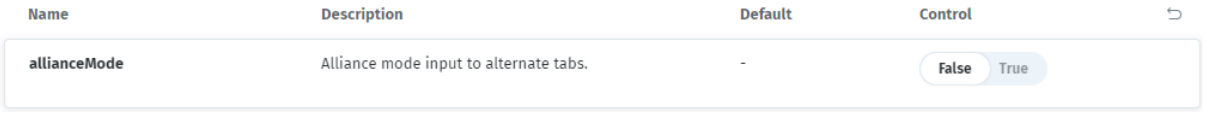
Figura 6

Documentación componente ProfileTabs

Este componente es usado en el perfil del aliado o alianza y su funcionalidad es la de alternar entre diferentes vistas o apartados como los que se observan en la **Figura 6**. Éste recibe también un parámetro que determina si será usado en el perfil de una alianza o no, ya que las vistas varían en base a esto.

Figura 7

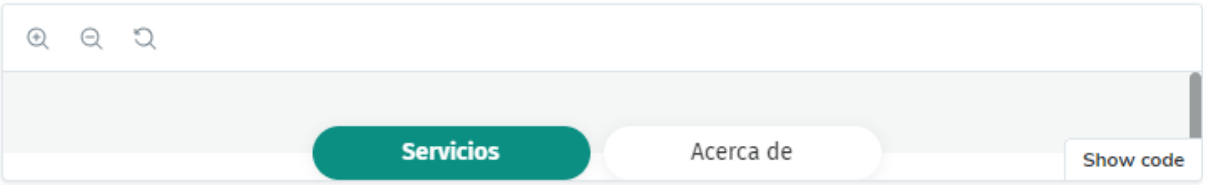
Controles componente ProfileTabs



El estado por defecto del componente *ProfileTabs* para el caso en que el control *allianceMode* es *false* corresponde al enseñado en la **Figura 6**. El caso contrario se muestra a continuación.

Figura 8

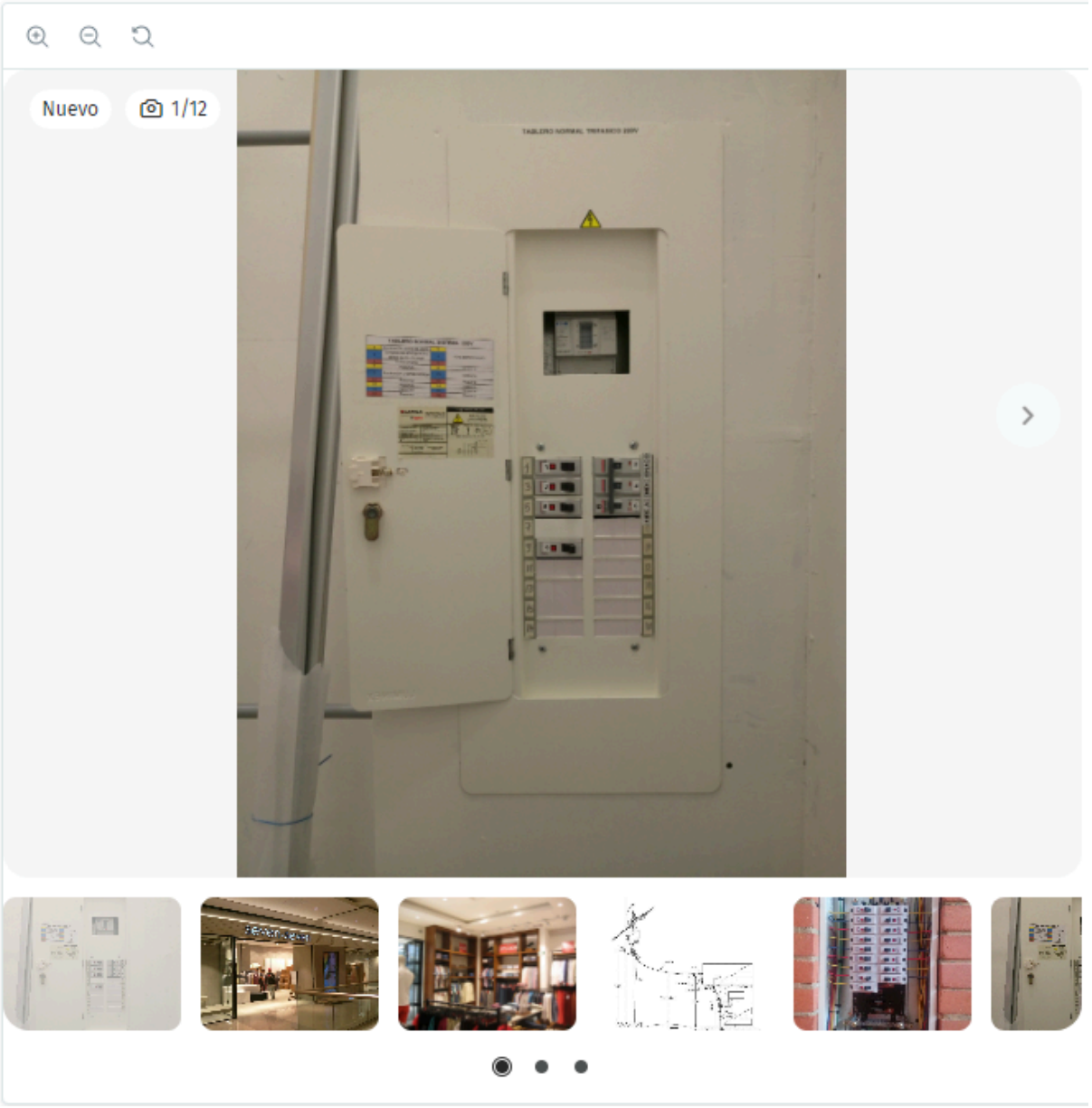
Componente ProfileTabs con *allianceMode = true*



Como se puede observar, nuestro componente se renderiza de manera diferente. Alternar entre estos estados de manera reactiva es en este caso una de las ventajas que nos ofrece Storybook, dándonos la posibilidad de interactuar con el componente y verificar su correcto funcionamiento dependiendo de sus controles o entradas.

Figura 9

Documentación componente MaGallery



Este componente es utilizado en el apartado *Galería de trabajos* que se puede ver en la **Figura 6**, con la finalidad de permitir al usuario visualizar fotos o imágenes referentes a los trabajos realizados por los aliados.

Figura 10

Documentación componente ServiceCard

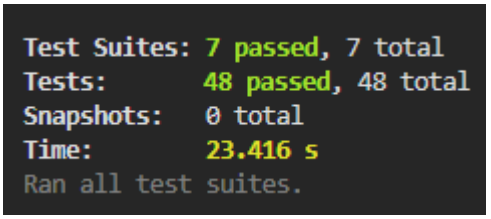
Este componente es probablemente el más usado y uno de los más transversales en el módulo web de **Mis Aliados**. Se renderiza en múltiples vistas, desde la propia landing principal hasta landings de promociones y perfiles de aliados/alianzas. Entre sus objetivos se encuentran el de brindar información del servicio en cuestión al usuario, visitar el perfil del aliado que lo ofrece, visitar la landing de detalle del servicio y contratarlo/adquirirlo.

4.2 Pruebas de humo

Posterior a generar la documentación para los componentes en cuestión, es posible llevar a cabo la ejecución de las denominadas pruebas de humo con el objetivo de compilar el código de cada instancia y verificar que no ocurra ningún error. Este es un paso fundamental teniendo en cuenta que la correcta compilación de las instancias es requerida para la posterior ejecución de las pruebas unitarias y de interacción.

Figura 11

Consola en VSCode mostrando los resultados de la ejecución de las pruebas de humo



La ejecución de estas pruebas arroja como resultado la correcta compilación de todas las instancias, en un tiempo de aprox. 23 s.

4.3 Pruebas unitarias y de interacción

La documentación de Storybook expuesta anteriormente para cada componente enseña una vista previa del mismo de manera predeterminada. Esta vista previa corresponde a la primera instancia desarrollada del componente en cuestión, que a su vez es una prueba unitaria.

Lo anterior quiere decir que, las vistas previas expuestas en la sección de **Documentación** son en sí una prueba unitaria con la diferencia de que allí se ofrece la posibilidad de alterar la información de entrada que requiere cada componente.

Las pruebas de interacción en Storybook son conjuntos de pruebas que se ejecutan de manera automática y paso a paso para una prueba unitaria, cuya finalidad es la de verificar que un componente se comporte de la manera esperada cuando se tiene la interacción de un usuario, así como constatar que sus elementos más relevantes funcionen de acuerdo a la información recibida por sus entradas.

Con esto en mente, a continuación se muestran las pruebas unitarias y de interacción para la prueba unitaria más relevante de cada componente, omitiendo las pruebas unitarias predeterminadas anteriormente expuestas y las pruebas de interacción que pueden considerarse repetitivas o menos aportantes.

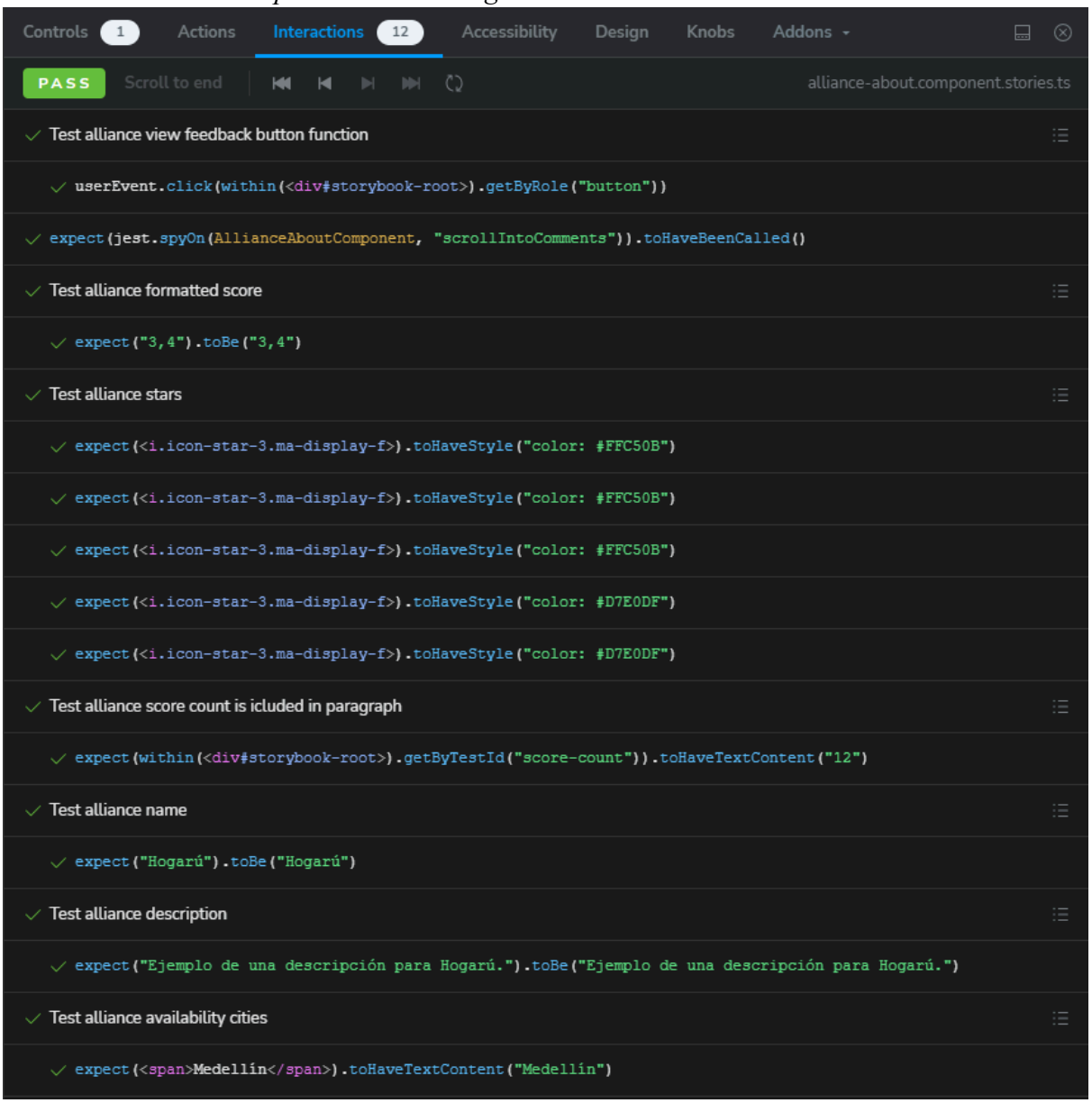
Figura 12

Prueba unitaria Hogarú en componente AllianceAbout

Esta es una prueba unitaria del componente *AllianceAbout* que se enseña en la **Figura 3**, donde se verifica que este refleje los cambios esperados cuando recibe información diferente por sus entradas.

Figura 13

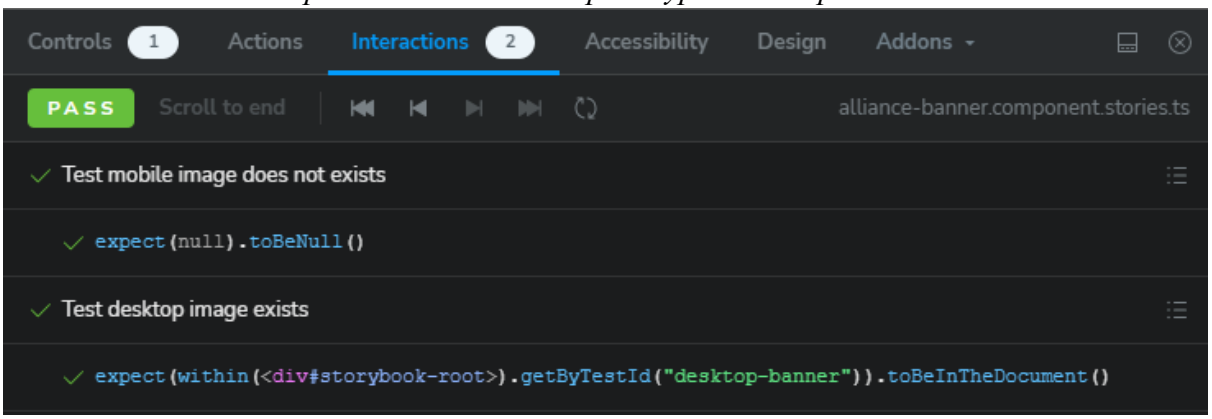
Pruebas de interacción para instancia Hogarú



En este conjunto de pruebas de interacción se comprueban principalmente 7 comportamientos correspondientes a las líneas de texto en color blanco; primero se simula la interacción de un usuario haciendo click en el botón *Ver los comentarios*, esperando que este haga el llamado correspondiente a la función *scrollIntoComments*, encargada de hacer scroll hasta la sección de comentarios. Luego, se prueba que la puntuación de la alianza se esté mostrando en el formato adecuado, que los iconos de estrella que se muestran en color amarillo correspondan a dicha puntuación, que la cantidad de comentarios se esté mostrando en el sitio esperado, que el componente esté mostrando el nombre y la descripción de la alianza correctamente, y por último, que se muestren las ciudades de disponibilidad esperadas.

Figura 14

Pruebas de interacción para instancia Desktop Wakypet en componente AllianceBanner

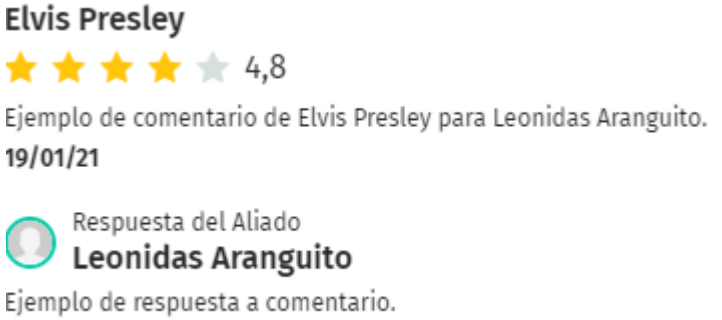


Estas son las pruebas de interacción para la instancia enseñada en la **Figura 4** cuyo componente es *AllianceBanner*. Este componente tiene la funcionalidad de alternar de manera reactiva entre dos imágenes de diferente tamaño, de acuerdo a la resolución en la que esté siendo renderizada la vista.

Teniendo en cuenta este comportamiento y que estas pruebas de interacción corresponden a una prueba unitaria en tamaño de escritorio, lo que se quiere comprobar es básicamente que la imagen de tamaño pequeño (dispositivo móvil) no esté siendo renderizada pero la de escritorio sí.

Figura 15

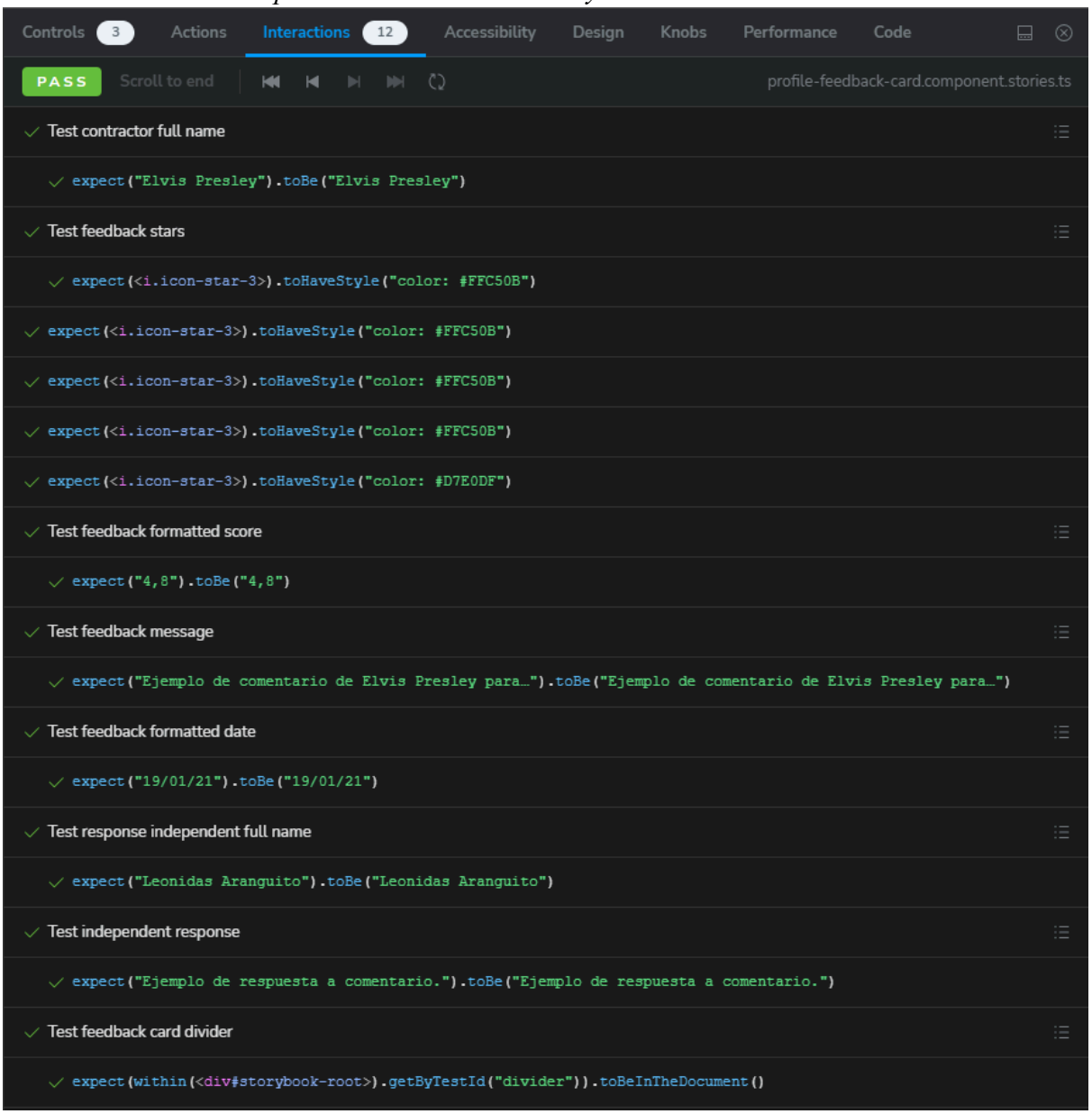
Prueba unitaria Elvis Presley en componente ProfileFeedbackCard



Esta es una prueba unitaria del componente *ProfileFeedbackCard* que se enseña en la **Figura 5**.

Figura 16

Pruebas de interacción para instancia Elvis Presley

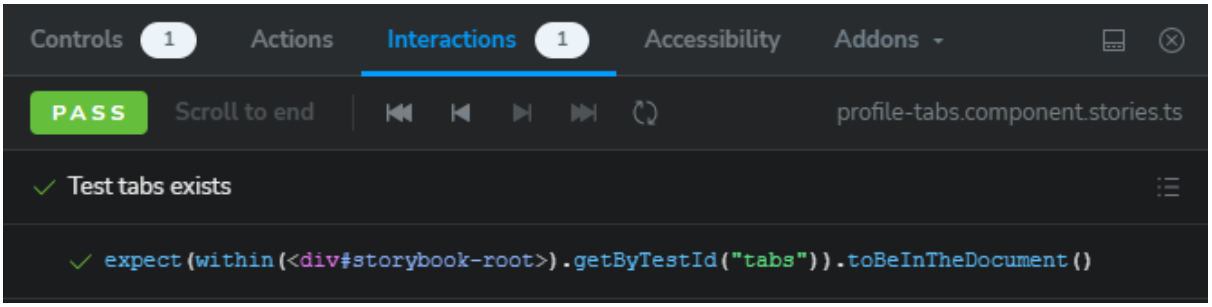


En este conjunto de pruebas de interacción se prueba el comportamiento de los elementos principales del componente, tales como el nombre del usuario que publica la reseña, el color de los iconos de estrella de acuerdo a la calificación del usuario, el formato de la calificación, el texto del comentario del usuario, el formato de la fecha de la reseña, el nombre del aliado en la respuesta a la reseña, el texto de la respuesta del aliado y por último se verifica que el separador esté siendo renderizado para el caso en que hayan más reseñas por mostrar abajo.

Figura 17
Prueba unitaria Mobile Alliance Mode en componente ProfileTabs

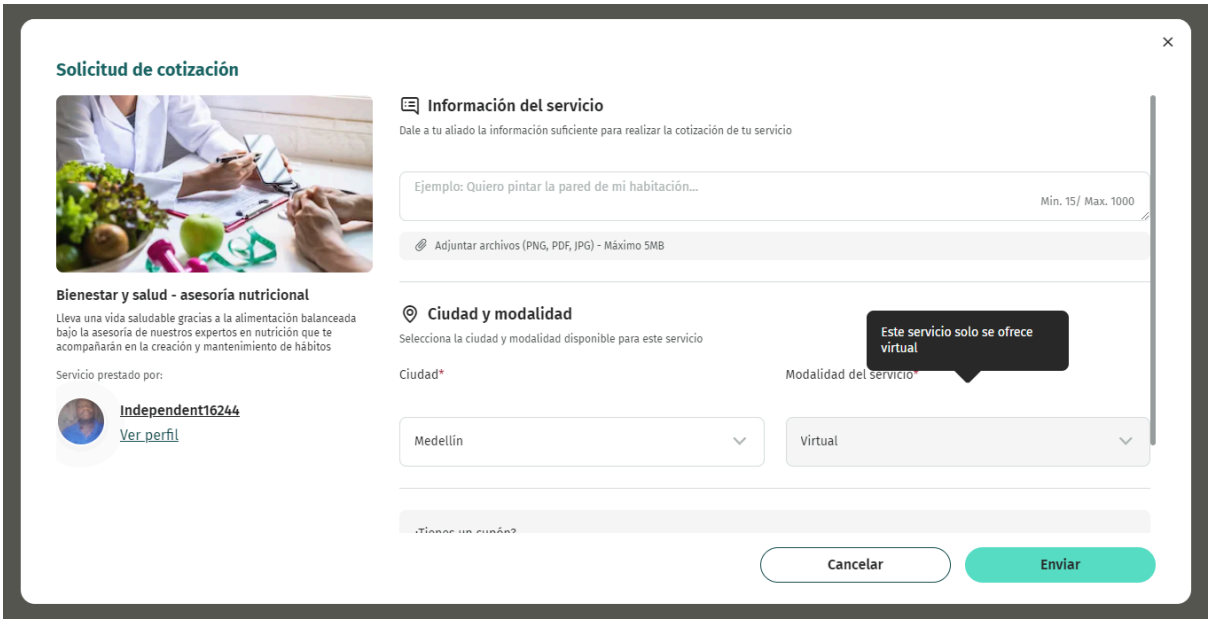
Esta es una prueba unitaria del componente *ProfileTabs* que se enseña en la **Figura 6**.

Figura 18
Pruebas de interacción para instancia Mobile Alliance Mode



Teniendo en cuenta que el componente *ProfileTabs* en su modalidad *AllianceMode* fue diseñado para ser renderizado únicamente en resoluciones de escritorio, ésta es la única verificación que se hace en estas pruebas de interacción.

Figura 19
Prueba unitaria One To One Single Modality en componente RequestForm



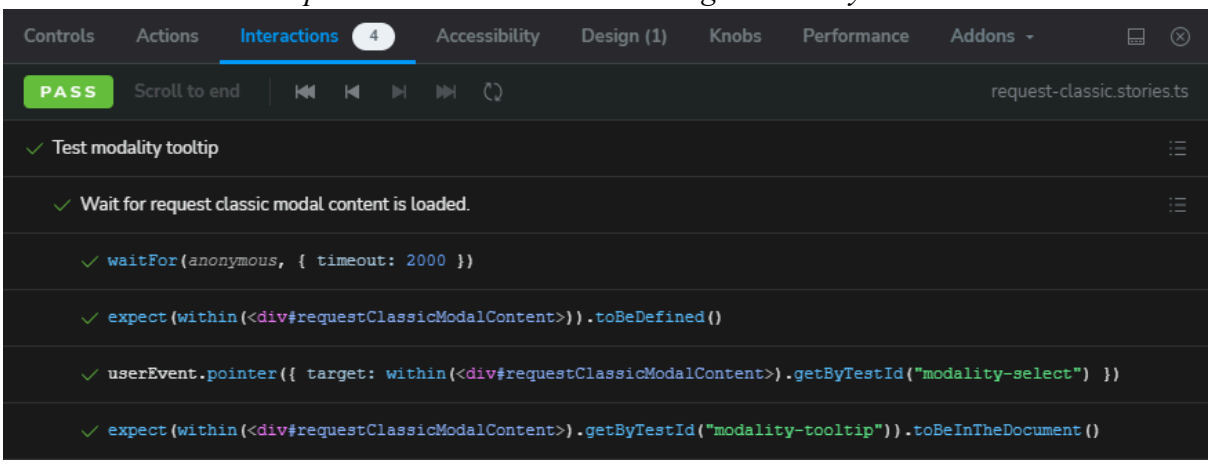
Debido a su diseño particular renderizado en un modal, el componente *RequestForm* es el único que no cuenta con documentación como todos los demás. Sin embargo se crearon pruebas unitarias y de interacción debido a que pertenece a uno de los módulos más

importantes y transversales de **Mis Aliados**, como lo es el módulo *Request Services*, mediante el que se hace posible la solicitud de servicios por parte de los usuarios.

Este componente tiene naturaleza bastante reactiva y existen diferentes maneras en las que muestra y/o solicita información al usuario, por lo que se tomó la prueba unitaria *One To One Single Modality* como ilustración para enseñar su funcionamiento en el caso particular de un servicio común, solicitado por un usuario a un único aliado, quien a su vez presta el servicio en una única modalidad; *Virtual*, en este caso.

Figura 20

Pruebas de interacción para instancia One To One Single Modality

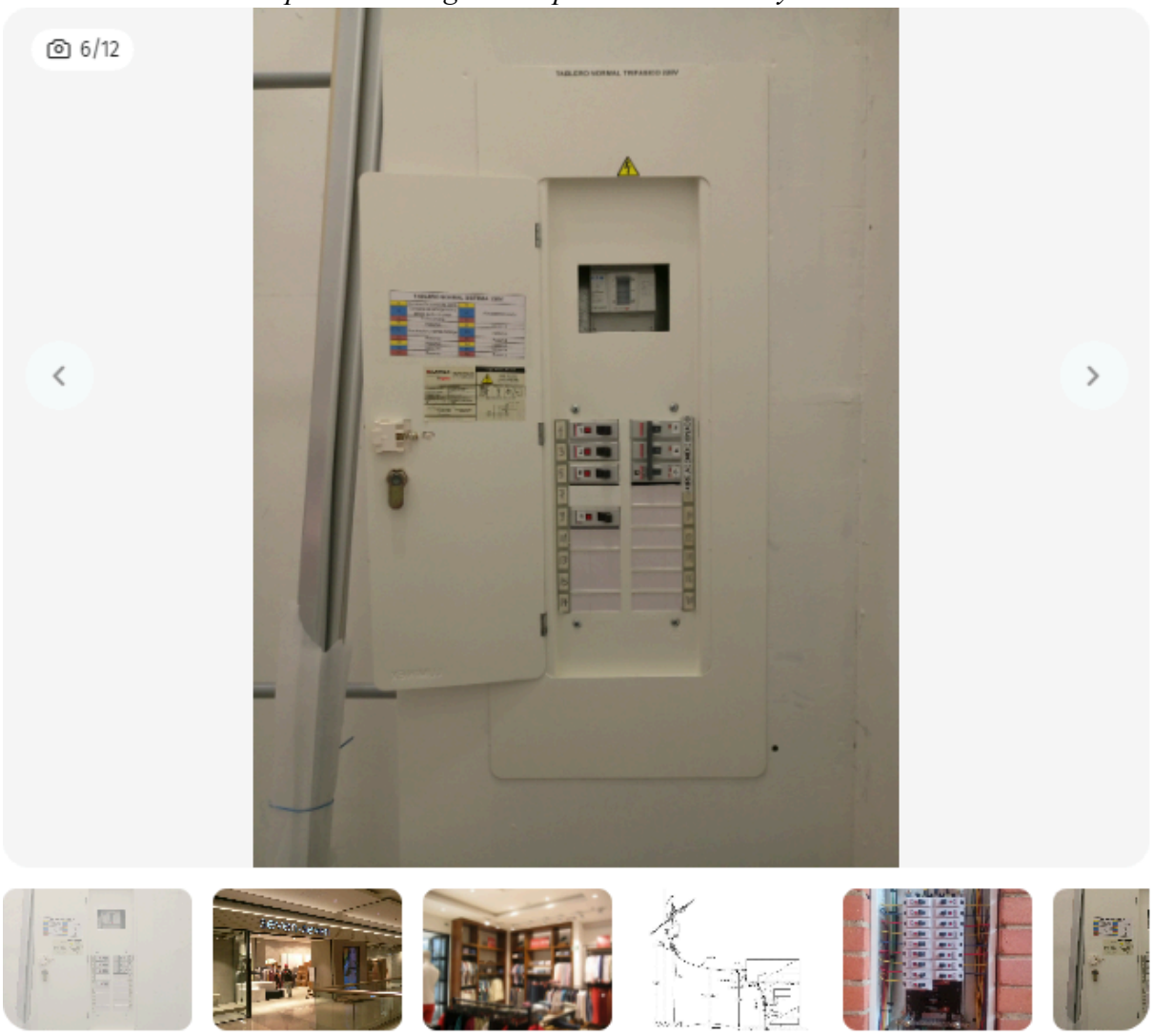


A pesar de la reactividad y complejidad del componente en cuestión, el conjunto de pruebas de interacción para esta prueba unitaria en particular es bastante simple y se eligió para esta ilustración adrede con la finalidad de mantener simple el entendimiento de estas pruebas.

Lo único que se quiere verificar en estas pruebas es que sea renderizado como es esperado el tooltip que informa que el servicio se presta en una única modalidad. Para esto, lo primero es esperar a que el modal se renderice, ya que este componente se crea de manera asíncrona y toma algo de tiempo para poder acceder a su instancia en el navegador. Una vez creada y hallada dicha instancia, se simula la interacción de un usuario haciendo click en la lista desplegable mediante la que se debería poder elegir entre las modalidades virtual o presencial, sin embargo, como se mencionó anteriormente, para nuestro contexto actual el servicio es prestado por el aliado únicamente en modalidad virtual. Esta situación hace que se renderice el tooltip y sea finalmente verificado en el último paso.

Figura 21

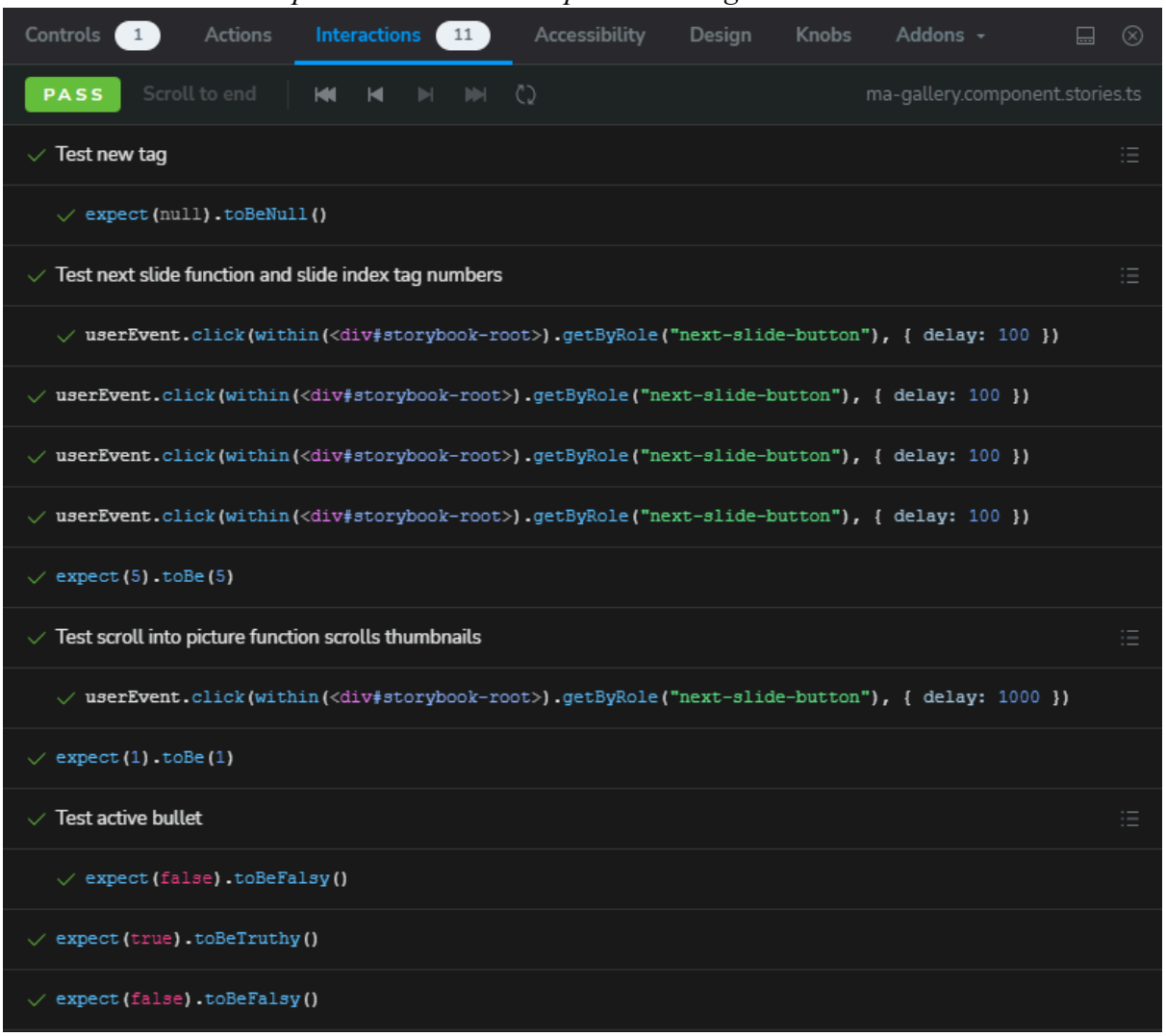
Prueba unitaria Desktop No New Tag en componente MaGallery



Esta es una prueba unitaria del componente *MaGallery* que se enseña en la **Figura 9**.

Figura 22

Pruebas de interacción para instancia Desktop No New Tag



Esta prueba unitaria, a diferencia de la enseñada en la **Figura 9**, no debe mostrar la etiqueta *Nuevo*. Teniendo esto en cuenta, en este conjunto de pruebas de interacción se prueba el comportamiento de los elementos principales del componente tales como la etiqueta mencionada, la funcionalidad de las flechas laterales para cambiar de imagen mediante la simulación de 4 clicks de un usuario, el valor de la etiqueta que muestra el número de imagen que se está viendo actualmente luego de los clicks (5), la funcionalidad de que se haga scroll en las miniaturas cuando la imagen actual no está a la vista en las mismas, y por último se verifica que el bullet activo sea el esperado.

En estas pruebas de interacción es importante notar que las acciones de usuario simuladas tienen configurado un retraso o *delay*, esto es necesario ya que el componente cuenta con animaciones y cada interacción debe esperar a que la anterior finalice antes de ser ejecutada.

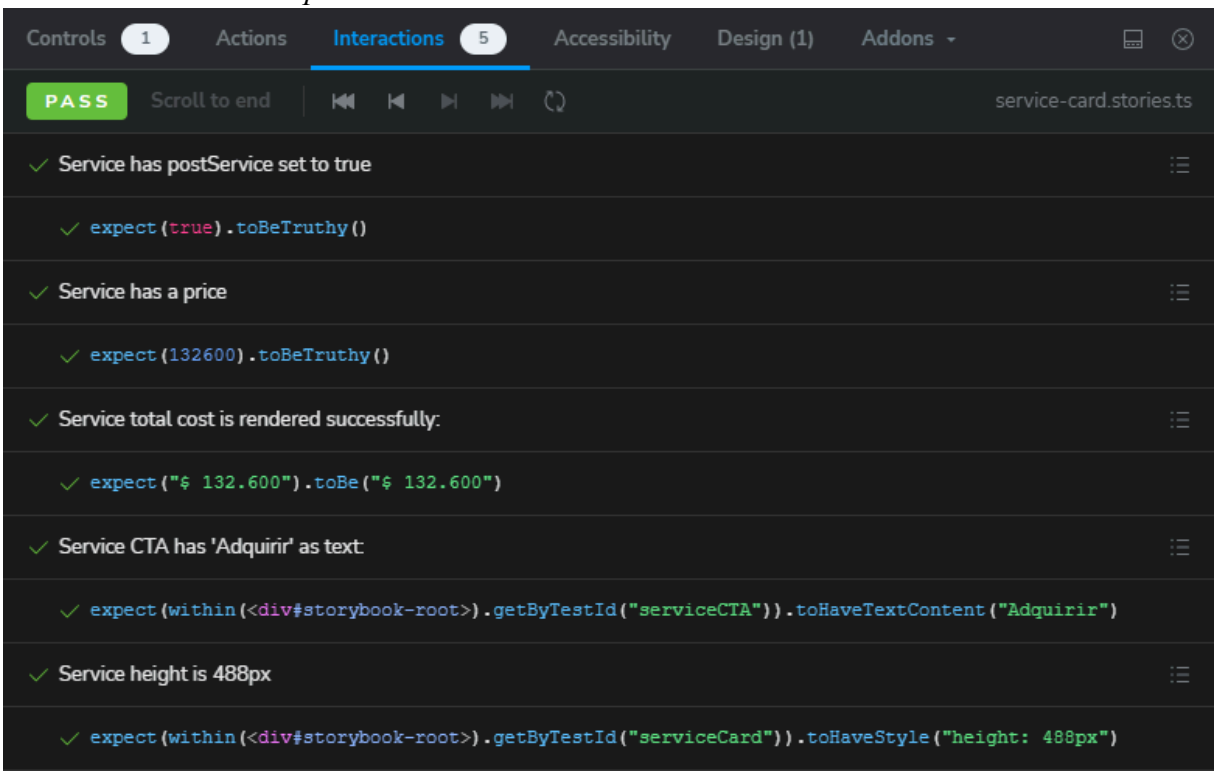
Figura 23

Prueba unitaria Post Service Card en componente ServiceCard

Como se mencionó anteriormente, el componente *ServiceCard* es uno de los más reactivos ya que la información que recibe de entrada puede variar demasiado. Por este motivo se eligió la misma prueba unitaria que se expuso anteriormente en la **Documentación** para este componente, esto teniendo en cuenta que esta es la que mayor información brinda sobre su funcionamiento y es más apropiada para llevar a cabo las pruebas de interacción.

Figura 24

Pruebas de interacción para instancia Post Service Card



Este conjunto de pruebas de interacción se enfoca en verificar que el componente muestre la información correcta en sus campos de acuerdo a los datos de entrada que recibe. En este contexto, se verifica que el componente reciba la entrada *postService* como *true* y la entrada *price* con algún valor entero, para indicar que el servicio corresponde a uno con precio fijo y no con cotización. Se verifica que el componente muestre el precio recibido en el formato esperado, que botón para obtener el servicio tenga el texto *Adquirir* y finalmente que el componente tenga una altura de *488px*. Todas estas verificaciones surgen de las decisiones de diseño tomadas en su momento por el equipo de UX/UI.

4.4 Automatización y cobertura de pruebas

Entre los objetivos planteados para este proyecto se consideró como general el de lograr una cobertura de pruebas del 50% por medio de los específicos, entre los que se definió el de automatizar los conjuntos de pruebas.

Lastimosamente, por decisiones de negocio en su momento, las prioridades del proyecto **Mis Aliados** se vieron afectadas y surgieron necesidades que se consideraron más importantes que el desarrollo de las pruebas, por lo que respecto al objetivo de automatizar las

pruebas se pudo avanzar únicamente en la elección y configuración de la herramienta mediante la que se pretendía lograr, llamada *Chromatic*.

Esta herramienta facilita la automatización de las pruebas llevadas a cabo por medio de Storybook mediante el almacenamiento de las mismas y la comparación de su contenido cuando se detectaba algún cambio en alguna de las pruebas o código del componente relacionado con estas. Entre los avances en *Chromatic*, se pudo conectar con el proyecto y se lograron subir la gran mayoría de las pruebas para ser analizadas, de manera que ya era posible ver los cambios efectuados entre cada actualización manual al código, si las pruebas seguían pasando sin problema, o si por el contrario comenzaban a fallar, arrojando logs mediante los que se podía tener detalles de lo que podía estar causando la falla.

La automatización del proceso de análisis de las pruebas se pretendía alcanzar de la manera descrita, con la diferencia de que las actualizaciones al código de *Chromatic* no debían subirse de manera manual, sino que debían ser ejecutadas automáticamente cada vez que se detectara un cambio en el repositorio.

Teniendo en cuenta lo mencionado anteriormente respecto a las decisiones de negocio en **Mis Aliados**, cabe resaltar que estas afectaron de manera directa también al objetivo general de alcanzar una cobertura en el código del 50%. Además, el posterior cierre oficial del proyecto se sumó a las circunstancias, por lo que en la actualidad no es posible ejecutar el comando mediante el que se pretendía obtener la cifra de cobertura alcanzada hasta el punto en el que se desarrollaron las pruebas. Sin embargo, la última cifra que se obtuvo en su momento en un ambiente local en la consola de VSCode rondaba en torno al 19%, esto teniendo en cuenta que se desarrollaron las pruebas para algunos de los componentes más usados.

5 Análisis

En el contexto de este proyecto, el factor mayormente influyente en los resultados obtenidos fue sin duda el de las decisiones de negocio y las prioridades que se tuvieron durante cada fase. A la falta de cumplimiento de los objetivos planteados en un principio se suma la causa mencionada, además de otras que es importante resaltar, tales como factores relacionados con la complejidad del proyecto, por ejemplo, en algunos componentes cuyos comportamientos asíncronos dificultan el acceso a los elementos fue bastante retador lograr que las pruebas funcionaran de la manera esperada, además, en muchos de estos casos las pruebas de interacción funcionaban correctamente cuando se ejecutaban en el host local de Storybook, caso contrario a cuando se subían a Chromatic, donde aparentemente por la configuración inicial y el diferente instanciamiento de las pruebas solían obtenerse errores durante la ejecución.

Por otro lado, teniendo en cuenta que para este tipo de pruebas se hace uso del patrón AAA, al momento de preparar el entorno aislado de la prueba con todas sus dependencias, servicios mockeados y demás información simulada, solían generarse bastantes dificultades debido a la gran cantidad de dependencias o comunicaciones de las que dependían algunos de los componentes más transversales del proyecto.

Sin embargo, gracias a la extensa documentación de Storybook y su manera de unificar las pruebas visuales con la documentación y pruebas unitarias, basta con conseguir una primera instancia del componente en cuestión para ser considerado un gran avance, teniendo en cuenta que en adelante solo resta replicarla con diferentes datos de entrada e identificar los elementos más relevantes para aplicar las pruebas de interacción.

6 Conclusiones

Gracias a la implementación de las pruebas en este proyecto se logra entender el esfuerzo que se necesita para llevar a cabo un buen plan de pruebas con una cobertura de código considerable e implementación de IC. Es sabido que estos planes se suelen considerar con la finalidad de garantizar calidad y estabilidad en los proyectos, sin embargo, para casos específicos de proyectos considerablemente robustos como el de **Mis Aliados** es muy relevante planear y estimar muy detenidamente el alcance.

En el caso de este proyecto, a pesar de las dificultades y cambios de prioridades durante el desarrollo, se puede concluir también que el objetivo general de alcanzar una cobertura de pruebas del 50% planteado en un inicio fue considerablemente ambicioso teniendo en cuenta los factores descritos anteriormente. Una vez finalizado el proyecto y con la experiencia adquirida, podría decirse que para el contexto en el que se llevó a cabo el plan de pruebas pudo alcanzarse a lo sumo una cobertura del 30%.

Por otro lado, es importante destacar lo útil de Storybook en el ámbito de las pruebas aisladas, partiendo del hecho de que los cambios que se efectúan en los datos de entrada de un componente se reflejan de manera reactiva, además de lo rápido que se pueden visualizar también los cambios en el código de las pruebas en el host local. A estos factores se suma el de la curva de aprendizaje, ya que teniendo en cuenta el contexto en el que se encontraba nuestro equipo de TI, donde ninguno de los integrantes había tenido contacto con Storybook previo a este plan de pruebas, no tomó más de unas cuántas horas tener lista una primera instancia de prueba para un componente.

Finalmente, queda como experiencia que a pesar de que se quiera seguir de manera estricta el plan de desarrollo de un proyecto en general, siempre hay cabida a imprevistos que se salen de nuestro alcance y alteran de manera directa el cumplimiento de los objetivos, en nuestro caso, ciertas decisiones tomadas por el negocio durante algunas etapas de **Mis Aliados** y, por último, su cierre definitivo.

Referencias

Chauhan, V. K. (2014). Smoke testing. Int. J. Sci. Res. Publ, 4(1), 2250-3153

Unit testing: Software testing (2023) GeeksforGeeks. Retrieved from: https://www.geeksforgeeks.org/unit-testing-software-testing/

Battat, M. (2022) What is visual testing?, Applitools. Retrieved from: https://applitools.com/blog/visual-testing/

Bourque, P., Dupuis, R., Abran, A., Moore, J. W., & Tripp, L. (1999). The guide to the software engineering body of knowledge. IEEE software, 16(6), 35-44

Rehkopf, M. (no date) Pruebas de software automatizadas para la Entrega Continua, Atlassian. Retrieved from: https://www.atlassian.com/es/continuous-delivery/software-testing/automated-testing

Palamarchuk, S. (2022) Cobertura de pruebas de software: ¿Cómo aumenta con la automatización?: Abstracta Chile, Blog de Testing y Calidad de Software | Abstracta Chile. Retrieved from: https://cl.abstracta.us/blog/cobertura-pruebas-software-automatizacion/