



**Certified Tech
Developer**
The Ultimate Degree

DigitalHouse>
Coding School



Reporte Final de Testing

Digital Booking

Integrantes:

- Mauricio Boyé
- Sofia Rotondi
- Alexis Petrovich
- Richard Mora
- Juan Manuel Mansilla Quintana
- Salomón Morales

Contenido

Introducción	3
Resumen de las actividades de prueba	3
Alcance	4
Dentro del Alcance	4
Fuera de Alcance	5
Tipos de Pruebas Ejecutadas	5
Enfoque de la Prueba	6
Exit Criteria	7
Resumen de Resultados	7
Pruebas diseñadas y ejecutadas por funcionalidades generales.	7
Ejecución Automática	8
Reporte de Defectos	10
Todos los defectos	10
Defecto por Estado	10
Defectos Abiertos	11
Lecciones Aprendidas & Conclusión	12

Introducción

Este documento es el Informe Final de Pruebas de Digital Booking. El propósito de este documento es proveer evidencia de que el Exit Criteria para el proceso de Testing se cumplió y por lo tanto, se concluye la fase de pruebas y puede cerrarse. Se demuestra que los Issues de GitLab relacionados con testing fueron implementados desde los Sprint 1 a 4. Este documento va a ser utilizado como entrada para la revisión general de las actividades de prueba y para tomar la decisión si el sistema cumple con las expectativas.

Resumen de las actividades de prueba

Durante los 4 sprints se realizaron varios tipos de testeo a la aplicación que se encuentra alojada en el siguiente link :

<http://grupo4-digitalbooking-front.s3-website.us-east-2.amazonaws.com/>

Se realizaron mayoritariamente **tests manuales** para verificar el correcto funcionamiento de los atributos, tests automatizados en **Postman** para verificar el correcto funcionamiento de la API con peticiones de tipo GET/POST/DELETE, testeando que estas funcionaran de manera esperada y que los datos traídos sean los correctos. Se utilizó **Selenium** para hacer más pruebas automatizadas siguiendo el camino feliz de funciones como el registro de un usuario, el login, una reserva, entre otros. Se realizaron varios tests con **JEST y React test library** a la parte del front para verificar que se rendericen correctamente y por último se decidieron utilizar extensiones extras para testear la accesibilidad de la página (**Lighthouse, Pa11y y AxeDevTool**)

Alcance

Dentro del Alcance

Desde QA nos enfocamos en testear las principales funcionalidades de la página a medida que se iban incorporando sprint a sprint. A medida que se avanzaba y se iban agregando más, no solo se testeaban las nuevas funciones, sino también las ya existentes para confirmar que la integración de las actualizaciones no causarían defectos a las existentes ni interfirieran con ellas.

Del primer al cuarto sprint se testeó:

- Responsive (desktop, tablet, mobile y medidas intermedias)
- Peticiones a la API con Postman.
- Registro de usuario, verificando por Postman y manualmente por base de datos.
- Login de usuario.
- Validaciones en los formularios.
- Renderización de componentes del front con JEST.
- Creación de Reservas, verificando por Postman y manualmente por base de datos.
- Agregar, eliminar y visualizar favoritos.
- Verificación por email de registro de usuario.
- Verificación por email de confirmación de reserva.
- Creación de productos desde admin verificando por Postman y manualmente por base de datos.
- Contraste de colores.
- Accesibilidad de la página.
- Calendarios de inicio, producto y reservas.
- Filtros por categoría, fecha y ciudad.
- Filtros combinados.
- Limpiar filtros.
- Página de error 404.
- Verificación de que en las búsquedas solo se muestran los productos disponibles.
- Tests negativos en los forms para testear la seguridad de la página (registro, login, reserva y creación de producto)
- Compartir por redes sociales.



- Correcta recepción de emails y funcionamiento de verificaciones.
- Funcionamiento y redirección de los pop ups dentro de la página (confirmación de reserva, agregar a favoritos, eliminar, etc)
- Botón de visualización de contraseña en el login y registro.
- Funcionamiento y flow de la página en general.
- Carrusel automático de imágenes en móvil y tablet.
- Paginación
- Visualización de productos aleatorios en inicio.
- Menú de opciones de usuario.
- Pop up de la locación del producto en el mapa.

Fuera de Alcance

Algunas funcionalidades como la integración del mapa, la coherencia de lenguaje de la página, prueba de algunas partes de componentes particulares de los formularios como el login, el registro, etc, no pudieron ser testeados con Jest.

Tipos de Pruebas Ejecutadas

	<i>Sprint 1</i>	<i>Sprint 2</i>	<i>Sprint 3</i>	<i>Sprint 4</i>
<i>Prueba Estática</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>
<i>Prueba Exploratoria</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>
<i>Prueba de Sistema (Selenium IDE, Selenium Web Driver)</i>	<i>NO</i>	<i>NO</i>	<i>SI</i>	<i>SI</i>
<i>Prueba de Humo</i>	<i>NO</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>
<i>Prueba de Regresión</i>	<i>NO</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>
<i>Prueba de Componente / Unidad (JEST)</i>	<i>NO</i>	<i>NO</i>	<i>SI</i>	<i>SI</i>
<i>Prueba de Integración (Postman)</i>	<i>NO</i>	<i>SI</i>	<i>SI</i>	<i>SI</i>

Enfoque de la Prueba

Se trabajó en conjunto con el equipo de desarrollo, se agregó una columna de verificar a la tabla de tareas donde cada vez que se concluía una tarea, se pasaba a esa columna con el criterio de que ya se encuentre mergeada a la rama principal que estábamos utilizando, y el equipo de QA era el encargado de verificar que la tarea esté finalizada, funcione correctamente y cumpla con los requisitos pedidos.

Se generaba un test en una planilla, se documentaba el resultado y si cumplía con lo pedido, recién ahí se pasaba a cerrado. En caso negativo, se reportaba el defecto al miembro del equipo que tenía asignada esa tarea.

Por cada funcionalidad y tarea se generaba un test positivo, y si lo permitía también uno negativo.

Cuando comenzaba un nuevo sprint se volvían a testear las funcionalidades anteriores para verificar que sigan funcionando de manera correcta y a medida que se integraba una nueva tarea, se volvía a hacer una revisión general para verificar su correcta integración.

Todo esto se reflejaba en planillas de pruebas con descripciones de cómo repetir la acción y si estaba aprobado o defectuoso.

El enfoque siempre se puso en las funciones básicas como el registro de usuario, el login, el funcionamiento de la API, el acceso a los productos, las reservas y los filtros.

En todos los sprints se informaba al equipo de todos los defectos encontrados y se agregaban a la listas de tareas por hacer en gitlab para tenerlos más presentes.

Documentación:

[Sprint 1 Exploratorios](#) - [Planilla](#)

[Sprint 2](#)

[Sprint 3 y 4](#)

[link a postman](#)

La documentación de los tests de selenium se encuentran en gitlab, dentro de la carpeta tests.



Exit Criteria

Se definió los siguientes criterios de aceptación para finalizar las pruebas:

- No se debe tener defectos en estado abierto de severidad crítica y/o bloqueante.
- Los test de postman deben pasar en su 100% para verificar el correcto funcionamiento de la API.
- Cubrir al menos un 60% de los componentes con JEST.

Resumen de Resultados

Pruebas diseñadas y ejecutadas por funcionalidades generales.

	<i>Test Manuales</i>	<i>Test Automáticos</i>	<i>Test de Integración (Postman)</i>	<i>Test Total</i>
<i>Login de Usuario</i>	10	4	3	17
<i>Crear Cuenta</i>	8	4	3	15
<i>Seleccionar Producto</i>	5	4	0	9
<i>Realizar Reserva</i>	8	2	1	11
<i>Crear producto</i>	11	0	2	13
<i>Filtrado de productos</i>	8	2	0	10
<i>Responsive</i>	17	0	0	17
<i>Accesibilidad</i>	2	5	0	7
<i>Calendarios</i>	6	2	0	8
<i>Verificaciones por email</i>	4	1	0	5
<i>Botones</i>	7	2	0	9
<i>PopUps</i>	3	2	0	5
<i>Favoritos</i>	3	2	0	5



Ejecución Automática

JEST- un 60% de los componentes testeados, mínimamente la renderización.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	31.55	11.36	15.31	31.91	
src	50	50	50	50	
App.js	75	50	50	75	11
index.js	0	100	100	0	7-8
src/components/BloqueListado	54.76	19.23	36.36	54.76	
BloqueListado.jsx	54.76	19.23	36.36	54.76	20,24,28,32-33,49-54,58-59,63-64,91-
src/components/BloqueListado/card	2.94	0	0	2.94	
CardListado.jsx	2.94	0	0	2.94	6-111
src/components/Booking	32.09	9.89	5.55	32.7	
Booking.jsx	32.09	9.89	5.55	32.7	...,228-246,250,253,256,260-263,268,
src/components/Booking/Schedule	3.22	0	0	3.33	
Schedule.jsx	3.22	0	0	3.33	6-60
src/components/Buscador	37.5	14.63	25	37.93	
Buscador.jsx	37.5	14.63	25	37.93	...-84,89-92,98-101,107-112,117-125,
src/components/Buscador/impl	28.57	0	0	28.57	
SelectCities.jsx	25	100	0	25	4-11
SelectDate.jsx	33.33	0	0	33.33	6-13
src/components/Carousel	3.57	0	0	3.57	
Carousel.jsx	3.57	0	0	3.57	5-68
src/components/Categories	28.2	7.14	20	28.2	
Categories.jsx	66.66	50	33.33	66.66	19-25,42
Category.jsx	4.16	0	0	4.16	11-47
src/components/CreateProduct	33.92	8.33	7.4	35.84	
CreateProduct.jsx	33.92	8.33	7.4	35.84	21-29,35-43,49-58,63,66,69-70,73-74,
src/components/Error404	100	100	100	100	
ErrorProduct.jsx	100	100	100	100	
src/components/Footer	100	100	100	100	
Footer.jsx	100	100	100	100	
src/components/Header	35.48	37.5	28.57	35.48	
Header.jsx	35.48	37.5	28.57	35.48	20-22,36-67
src/components/ImagesDisplay	20	100	0	20	
ImagesDisplay.jsx	20	100	0	20	7-18
src/components/Login	34.28	18.75	21.42	34.28	
Logged.jsx	0	0	0	0	13-53

Postman- 100% de los tests aprobados:

testsApiAutomatizado-Sofi No Environment, Today, 12:40 pm Run Again + New Run

All Tests Passed (24) Failed (0) Skipped (0) [View Summary](#)

- Pass Response time is less than 1000ms
- Pass Name is Hostel







GET findAll-product-TEST {{apiURL}}/api/product/findAll / findAll-product-TEST 200 OK 635 ms 120.21 KB

- Pass Response time is less than 1800ms
- Pass Status code is 200

GET findAll-Categ-TEST {{apiURL}}/api/categories/findAll / findAll-Categ-TEST 200 OK 306 ms 594 B

- Pass Response time is less than 700ms
- Pass Status code is 200







GET find-one-Categ-TEST {{apiURL}}/api/categories/find/3 / find-one-Categ-TEST 200 OK 276 ms 20.283 KB

 Cookies  Bootcamp  Auto-select agent  Runner  Trash  ?

Selenium


Selenium IDE - DB sprint4*



Project: DB sprint4*

Tests +      

Search tests... Step over current command. Ctrl+ website.us-east-2.amazonaws.com/

	Command	Target	Value
1	open	http://grupo4-digitalbooking-front.s3-website-us-east-2.amazonaws.com/	
2	set window size	1360x728	
3	click	css=login > h2	
4	click	name=email	

Command: type // 

Target: name=password  

Value: 1234567

Description:

Log Reference

- 5. type on name=email with value rotondi.soha@gmail.com OK 13:14:14
- 6. click on name=password OK 13:14:16
- 7. type on name=password with value 1234567 OK 13:14:17
- 8. click on css=button:nth-child(1) OK 13:14:18
- 9. mouseOver on css=.render-container:nth-child(1) > .card-main:nth-child(1) .card-button OK 13:14:19
- 10. mouseOut on css=.render-container:nth-child(1) > .card-main:nth-child(1) .card-button OK 13:14:20

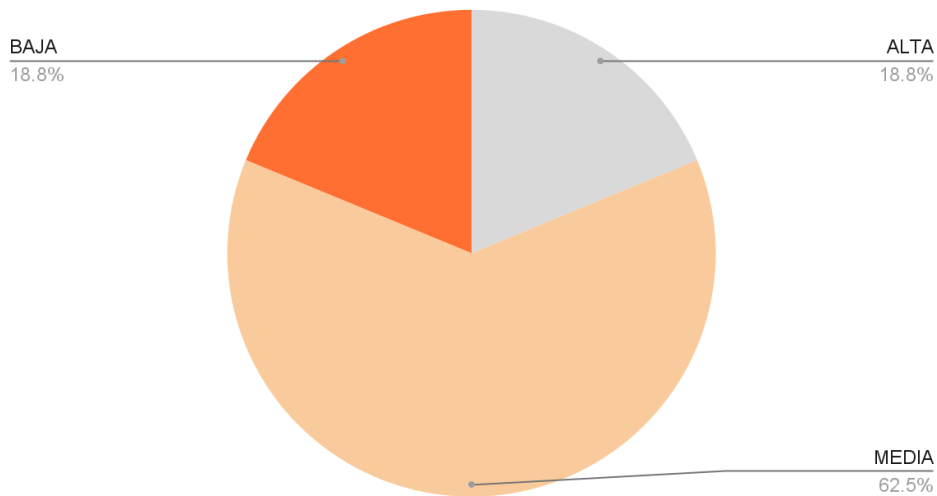
'login de usuario' completed successfully 13:14:21

Reporte de Defectos

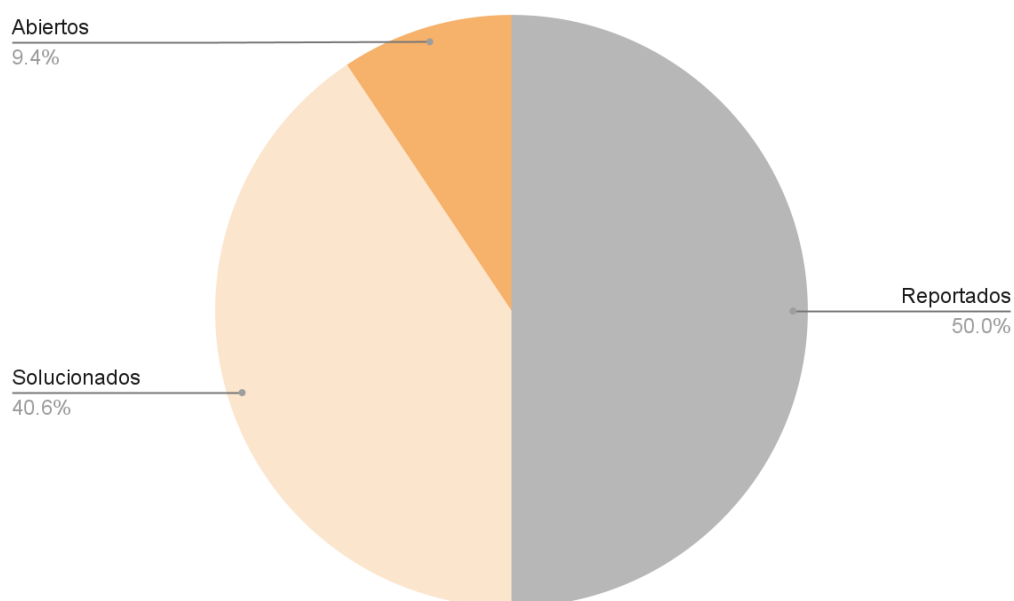
Todos los defectos

La siguiente sección muestra información con respecto al número total de defectos que se han presentado durante la duración de la fase de prueba.

Defectos reportados por severidad

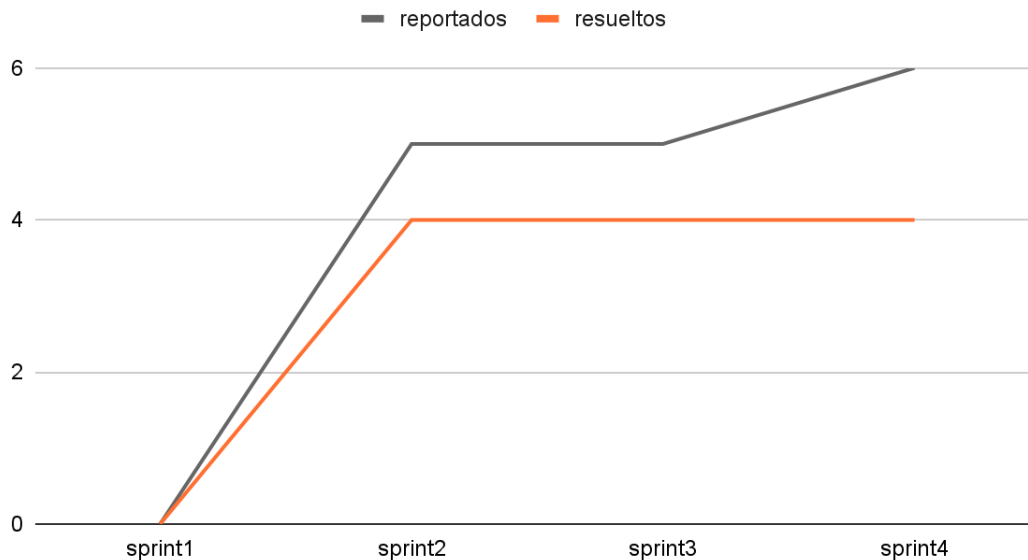


Defecto por Estado





Defectos Creados vs Resueltos



Defectos Abiertos

Al finalizar la fase de prueba del último sprint quedan a solucionar solo 3 defectos reportados de severidad media que no influyen en la funcionalidad primaria de la página:

- Faltan etiquetas “labels” en algunos elementos del formulario de Login
- Faltan etiquetas “labels” en algunos elementos del formulario de Registro.
- Falta una adaptación de la página a medidas mayores a una pantalla regular de notebook.

Estos defectos fueron considerados de severidad media al hacer el testeo de accesibilidad, pero en el funcionamiento regular de la página no ponen bloqueos.

Lecciones Aprendidas & Conclusión

A través de los sprints nos fuimos dando cuenta de la importancia del testeo durante el proceso de desarrollo a medida de que se iban integrando más funcionalidades.

El testear la aplicación continuamente nos ayudó a identificar rápidamente los defectos dándonos suficiente tiempo para ir modificándolos antes de las Reviews y antes de comenzar el siguiente sprint.

Tuvimos la posibilidad de aprender a usar muchas nuevas herramientas, o que ya conocíamos antes, pero con mayor profundidad. Nos permitió explorar el producto de diferentes maneras más allá de las convencionales para cubrir más formas de testear y pudimos desarrollar un camino feliz luego de explorar y conocer todas las demás alternativas. Esto nos ayudó a identificar las fortalezas y las debilidades de nuestro producto.

Llegamos a la conclusión que QA/Testing es una parte fundamental del proceso de desarrollo que nos permite presentar un producto final de buena calidad, con la menor cantidad de defectos posible y con un buen valor para el cliente.