

# TESTING INFORME FINAL

---

## Integrantes del equipo:

- Brenda Calzada
- Gonzalo Sibona
- Joan González
- Santino Lamberti
- Jorge Corredor
- Fernando Castillo

## Introducción

Este documento es el Informe Final de Pruebas del sistema **Digital Cars**. 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

### Resumen del sistema:

La aplicación Digital Cars es una App para alquilar vehículos en diferentes ciudades de Argentina que ofrece un gran abanico de posibilidades que van desde motos hasta buses, permitiéndole a los usuario elegir la ciudad de destino y/o las fechas que más les convenga con el fin de poder reservar la mejor opción.

### URL para acceder al sitio:

<http://www.digitalcars.com.s3-website.us-east-2.amazonaws.com/>

[Link a la tabla de actividades de prueba](#)

---

---

## **Alcance**

### **Dentro del alcance:**

**Testing Manual:** Se probaron las siguientes funcionalidades:

- Cumplimiento de pautas visuales
- Creación de usuario
- Logueo de usuario
- Filtrado por categoría de producto
- Filtrado por ciudad y/o fecha
- Selección de producto
- Detalle de producto
- Reserva del producto
- Calendarios
- Administración de producto
- Mis reservas

**Testing Automático:** Se probaron las funcionalidades Get, Post, Update y/o Delete en :

- Categorías
- Ciudad
- Fechas
- Producto
- Reservas
- Login
- Registro
- Usuario

### **Fuera del Alcance:**

- Pruebas de integración de mapas
- Calendarios sin test automatizado

---

## **Tipos de Pruebas Ejecutadas**

[Link a tabla de Tipos de Pruebas Ejecutadas](#)

## **Enfoque de la prueba**

Se crearon test funcionales tanto positivos como negativos para cada funcionalidad del sistema que así lo permitía, también fueron clasificadas en unitarias o de integración. A partir del segundo Sprint se empezaron a clasificar las pruebas en smoke y regresión según lo solicitado en las issues. Se clasificaron los defectos por prioridad y severidad.

Todos los defectos fueron volcados en su planilla correspondiente clasificados por Sprint.

Cada vez que se recibió un nuevo build se verificó que las funcionalidades ya existentes siguieran funcionando de forma correcta, una vez verificado esto se procedió a verificar las nuevas funcionalidades aplicadas. Se realizó la prueba exploratoria la cual está dividida por Sprint y funcionalidad. En el último Sprint se ejecutaron nuevamente todas las pruebas para verificar el funcionamiento y calidad del producto entregado.

**Link Planilla de Casos de Prueba:**

[Sprint 1](#)

[Sprint 2](#)

[Sprint 3](#)

[Sprint 4](#)

**Link Planilla de Defectos:**

[Sprint 1](#)    [Sprint 2](#)

[Sprint 3](#)    [Sprint 4](#)

---

## **Exit Criteria**

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

- ☐ No se debe tener defectos en estado abierto de severidad crítica y/o bloqueante.
- ☐ El coverage total de Jest debe superar el 40%.
- ☐ Las solicitudes de Postman deben alcanzar el 100%.
- ☐ Los defectos con severidad media/alta deben estar resueltos al 100%.

## **Resumen de Resultados**

**Diseño de pruebas**

[Link a Tabla de Diseños de Prueba](#)

## **Ejecución de Pruebas**

**Ejecución Manual**

Tabla con los resultados de las pruebas de Regresión ejecutado en el Sprint 4:

[Link a Tabla de Ejecución de pruebas manuales](#)

---

## Ejecución Automática

### Test con Jest por componente

Delta compression using up to 12 threads

```
PASS src/__test__/politicas.test.js
PASS src/__test__/pantallaExitoProducto.test.js
PASS src/__test__/gallery.test.js
PASS src/__test__/buscador.test.js (6.096 s)
PASS src/__test__/buscar.test.js (7.094 s)
```

```
PASS src/__test__/formDatos.test.js
```

```
PASS src/__test__/reservas.test.js
PASS src/__test__/horarioLlegada.test.js
PASS src/__test__/pantallaExitoReserva.test.js
PASS src/__test__/iniciarSesion.test.js (9.045 s)
PASS src/__test__/footer.test.js
PASS src/__test__/header.test.js
PASS src/__test__/producto.test.js
PASS src/__test__/categories.test.js
```

```
PASS src/__test__/crearCuenta.test.js (10.547 s)
```

```
PASS src/__test__/swipegallery.test.js
```

```
PASS src/__test__/reservasUsuario.test.js
```

# Test con Jest, coverage total (60.84%)

## All files container

100% Statements 3/3 100% Branches 8/8 100% Functions 3/3 100% Lines 3/3

Press n or j to go to the next uncovered block, b, p or k for the previous block.

File	Statements	Branches	Functions	Lines
App.jsx	100%	1/1	0/0	100%

## All files components

60.84% Statements 324/533 33.7% Branches 68/218 45.53% Functions 34/732 61.98% Lines 311/501

Press n or j to go to the next uncovered block, b, p or k for the previous block.

File	Statements	Branches	Functions	Lines
Buscador.jsx	69.23%	27/39	53.57%	27/38
Categorias.jsx	76.47%	13/17	75%	13/17
CrearCuenta.jsx	57.14%	8/14	50%	7/12
Footer.jsx	100%	1/1	0/0	1/1
FormDatos.jsx	69.23%	9/13	50%	9/13
Gallery.jsx	77.77%	7/9	50%	7/9
Header.jsx	60.46%	26/43	21.05%	26/42
Home.jsx	100%	1/1	0/0	1/1
HorarioLlegada.jsx	72.72%	8/11	100%	8/11
IniciarSesion.jsx	65.62%	21/32	38.88%	19/29
Loading.jsx	100%	1/1	0/0	1/1
PantallaExtoProducto.jsx	75%	3/4	100%	3/4
PantallaExtoReserva.jsx	75%	3/4	0/0	3/4
Politicas.jsx	100%	1/1	0/0	1/1
Producto.jsx	58.62%	34/58	18.18%	34/55
Reservas.jsx	40.9%	36/88	9.37%	36/86
ReservasUsuario.jsx	86.66%	13/15	75%	13/15
SwipeGallery.jsx	100%	4/4	0/0	3/3

## Test con Postman

**All Tests** Passed (45) Failed (0)

Iteration 1

**GET** GET-CiudadYFechas `http://localhost:8080/reservas/ciudadYFechas?ciudad=Bariloche&fechaInicial=2021-12-30&fechaFinal=2022-03-01`

Pass

Test JSON

Pass

Responde con 200

Pass

Respuesta en menos de 200ms

**GET** GET-Fechas `http://localhost:8080/reservas/fechas?fechaInicial=2021-09-18&fechaFinal=2021-11-19` / Sprint 4 / GET-Fechas

Pass

Responde con 200

Pass

Test JSON

Pass

Respuesta en menos de 200ms

**GET** GET-Producto-Ciudad `http://localhost:8080/productos/ciudad?nombre=Bariloche` / Sprint 4 / GET-Producto-Ciudad

Pass

Responde con 200

Pass

Test JSON

**GET** GET-Reserva `http://localhost:8080/reservas/buscar/38` / Sprint 4 / GET-Reserva

Pass

Responde con 200

Pass

Test JSON

Pass

Validar atributo: fechaInicial

Pass

Validar atributo: fechaFinal

Pass

Validar atributo: hora

Pass

Validar atributo: producto

Pass

Respuesta en menos de 200ms

---

**POST** Post-LoginAdmin <http://localhost:8080/login> / Sprint 4 / Post-LoginAdmin

Pass Responde con 200

Pass Validar atributo: nombre

Pass Validar atributo: apellido

Pass Validar atributo: email

Pass Validar atributo: token

**POST** Post-LoginCliente <http://localhost:8080/login> / Sprint 4 / Post-LoginCliente

Pass Responde con 200, Login Correcto

Pass Validar atributo: nombre

Pass Validar atributo: apellido

Pass Validar atributo: email

Pass Validar atributo: rol

Pass Validar atributo: token

**POST** POST-RegistroCliente <http://localhost:8080/registro> / Sprint 4 / POST-RegistroCliente

Pass Responde con 201

Pass Validar atributo: nombre

Pass Validar atributo: apellido

Pass Validar atributo: email

Pass Validar atributo: token



**GET** ALL-Reservas-Cliente <http://localhost:8080/productos/cantidad/usuario/79> / Sprint 4 / ALL-Reservas-Cliente

Pass Responde con 200

Pass Respuesta en menos de 200ms

**POST** POST-Reserva-Cliente <http://localhost:8080/reservas> / Sprint 4 / POST-Reserva-Cliente

Pass Responde con 200

Pass Test JSON

Pass Validando atributos: fechaInicial, fechaFinal, hora, producto

**POST** POST-Producto-Admin <http://localhost:8080/productos> / Sprint 4 / POST-Producto-Admin

Pass Responde con 200

Pass Validar atributo: nombre

Pass Validar atributo: descripcion

**GET** GET-Reservas-Cliente <http://localhost:8080/reservas/usuario/79> / Sprint 4 / GET-Reservas-Cliente

Pass Responde con 200

Pass Respuesta en menos de 200ms

Pass Test JSON

**GET** GET-AllProducto <http://localhost:8080/productos/todos> / Sprint 4 / GET-AllProducto

Pass Responde con 200

Pass Test JSON

Pass Validando atributos: nombre, descripcion, categoria, ciudad, imagenes, caracteristicas

**POST** POST-Categorías <http://localhost:8080/categorias> / Sprint 4 / POST-Categorías

- Pass Responde con 200
- Pass Validar atributo: titulo
- Pass Validar atributo: descripcion
- Pass Validar atributo: url

**PUT** PUT-Categorías <http://localhost:8080/categorias/modificar/1> / Sprint 4 / PUT-Categorías

- Pass Responde con 200
- Pass Validar atributo: url
- Pass Validar atributo: titulo
- Pass Validar atributo: descripcion

**GET** GET-Todas <http://localhost:8080/categorias/todos> / Sprint 4 / GET-Todas

- Pass Responde con 200
- Pass Validar atributos: nombre, descripción, url

**GET** GET- ID <http://localhost:8080/categorias/buscar/2> / Sprint 4 / GET- ID

- Pass Responde con 200
- Pass La categoría es: Autos
- Pass La descripción es: 35 Autos
- Pass La URL es: [https://buimagenes.s3.us-east-2.amazonaws.com/img/Categoria\\_sedan.jpg](https://buimagenes.s3.us-east-2.amazonaws.com/img/Categoria_sedan.jpg)



## **Reporte de Defectos**

Todos los defectos divididos por Sprint

[Sprint 1](#)

[Sprint 2](#)

[Sprint 3](#)

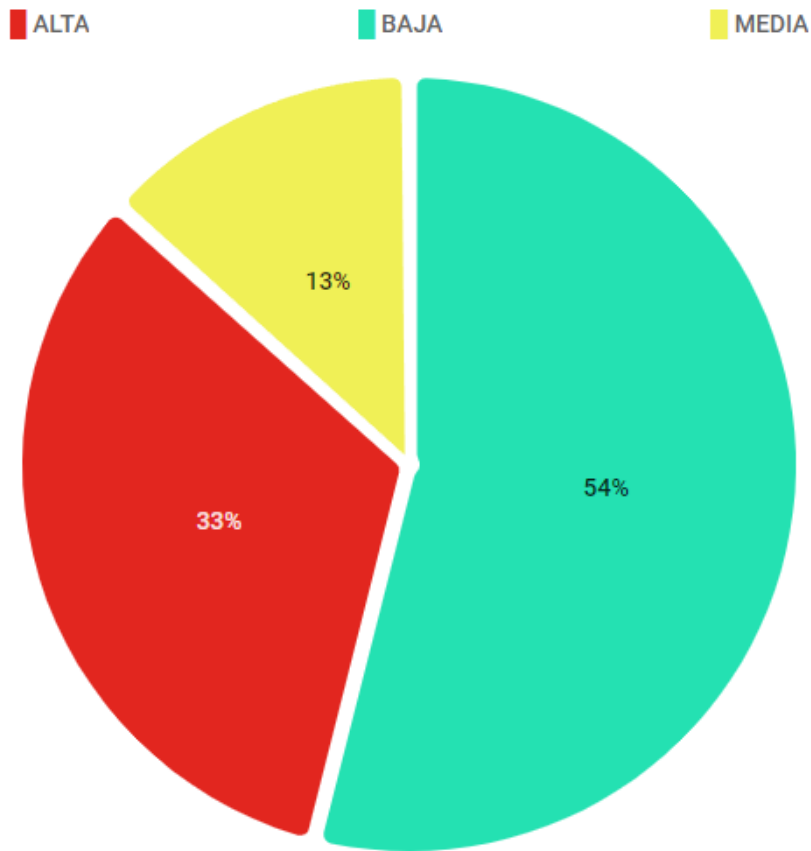
[Sprint 4](#)

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 por Prioridad

DEFECTOS POR PRIORIDAD



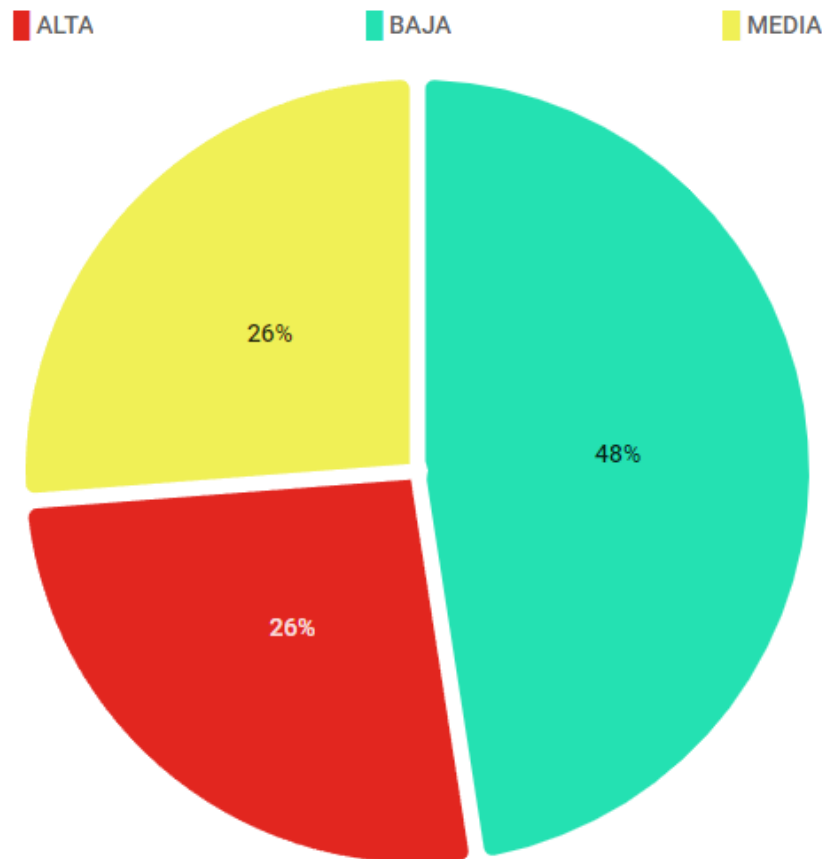
...

You can paste CSV data from other apps using Ctrl+V

	Values
ALTA	15
BAJA	25
MEDIA	6

## Defectos por Severidad

DEFECTOS POR SEVERIDAD



...

You can paste CSV data from other apps using Ctrl+V


		Values
	ALTA	12
	BAJA	22
	MEDIA	12

---

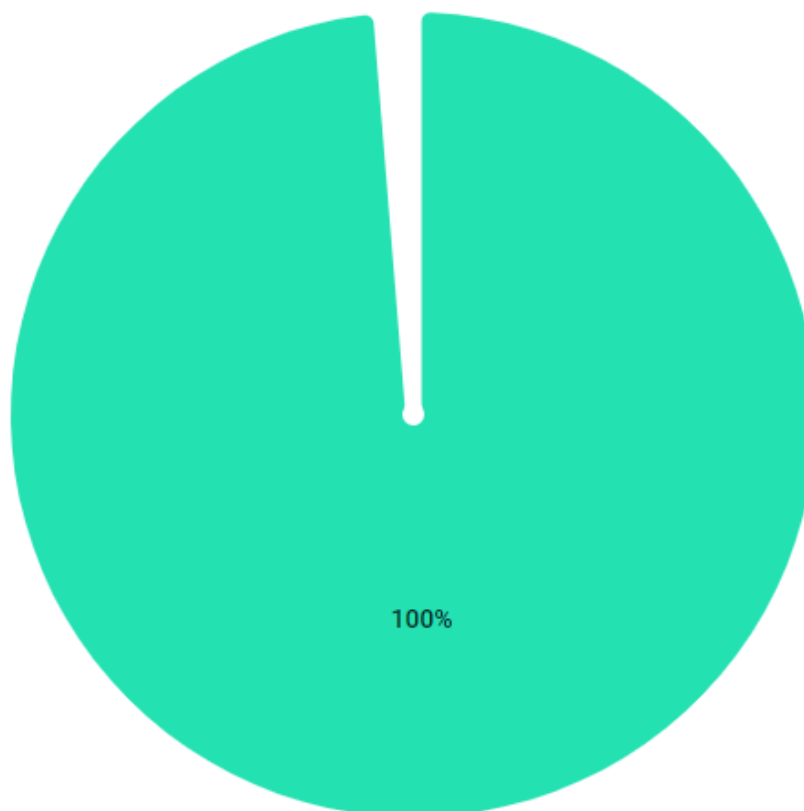
## Defectos por Estado

DEFECTOS POR ESTADO

 FAILED


 SOLUCIONADO

 PENDIENTE



...

You can paste CSV data from other apps using Ctrl+V

	Values
 FAILED	0
 SOLUCIONADO	46
 PENDIENTE	0

# Defectos Creados vs Resueltos



...

You can paste CSV data from other apps using Ctrl+V

	Inicio	Sprint 1	Sprint 2	Sprint 3	Sprint 4
CREADOS	0	15	11	14	6
RESUELTOS	0	14	11	13	8

---

## Defectos Abiertos

De los defectos encontrados no quedó ningún defecto abierto.

## Testing Exploratorio

### PI - Prueba Exploratoria | G3

## Lecciones aprendidas- Conclusión

En conclusión, a partir de mucha investigación además de prueba y error, hemos aprendido a manejar herramientas que desconocíamos, así como también la parte de testear componentes que no teníamos casi desarrollada. Los test nos hicieron ver la importancia de hacerlos a consciencia para poder verificar el correcto funcionamiento de una aplicación, por cada una de sus partes y por todo el conjunto. Testear desde el momento uno del desarrollo nos evitó arrastrar problemas que luego fueran muy difíciles de solucionar, por lo tanto nos ahorró mucho trabajo. Aprendimos que además de testear el software para su correcto funcionamiento, las pruebas también nos sirvieron para verificar que dicho software cumplía con los requisitos del cliente, así pudimos corregir errores a tiempo y entregar un producto de calidad.