**Plan de Pruebas de Software**

***[GreenMarket]***

***Fecha: [20/11/2024]***

**Tabla de contenido**

[**Historial de Versiones 4**](#_heading=h.9sqk3ynu3ml8)

[**Información del Proyecto 4**](#_heading=h.30j0zll)

[**Aprobaciones 4**](#_heading=h.q1gqgbsfah2m)

[**Resumen Ejecutivo 5**](#_heading=h.paj7jbiiwpuj)

[**Alcance de las Pruebas 5**](#_heading=h.j7ra6is06tiz)

[**Elementos de Pruebas 5**](#_heading=h.403b8s4ibfew)

[**Nuevas Funcionalidades a Probar 7**](#_heading=h.6tl5yjl5y6hb)

[**Pruebas de Regresión 8**](#_heading=h.9b4hrh21d44k)

[**Funcionalidades a No Probar 9**](#_heading=h.1pi643u70swt)

[**Enfoque de Pruebas (Estrategia) 10**](#_heading=h.soppj0vkqdhf)

[**Criterios de Aceptación o Rechazo 11**](#_heading=h.shm74sudja1u)

[Criterios de Aceptación o Rechazo 11](#_heading=h.3rdcrjn)

[Criterios de Suspensión 12](#_heading=h.ebuv3uttgvrz)

[**Criterios de Reanudación 14**](#_heading=h.vxwiakbzghls)

[**Entregables 15**](#_heading=h.z916als40pf2)

[**Recursos 15**](#_heading=h.g7zqkqsj126d)

[Requerimientos de Entornos – Hardware 15](#_heading=h.44sinio)

[Requerimientos de Entornos – Software 16](#_heading=h.elm2geomfys4)

[Herramientas de Pruebas Requeridas 17](#_heading=h.yrxgkbk68yiv)

[**Personal 1**](#_heading=h.2hqwm2zh11i7)

[**Entrenamiento 1**](#_heading=h.vg1ilubu5nzz)

[**Planificación y Organización 1**](#_heading=h.4i7ojhp)

[Procedimientos para las Pruebas 1](#_heading=h.2xcytpi)

[**Matriz de Responsabilidades 1**](#_heading=h.ehz2gptjr3q0)

[**Cronograma 1**](#_heading=h.pleyfsazjbgf)

[**Premisas 1**](#_heading=h.7ms7j3siss2k)

[**Dependencias y Riesgos 1**](#_heading=h.qsh70q)

[**Referencias 1**](#_heading=h.nfgxmqx9019z)

[**Glosario 1**](#_heading=h.799ytybmt2o1)

# Historial de Versiones

| **Fecha** | **Versión** | **Autor** | **Organización** | **Descripción** |
| --- | --- | --- | --- | --- |
| 20/11/2024 | 1.0 | Juan Herrea | Bugui Fost |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Información del Proyecto

| Empresa / Organización | Bugifost |
| --- | --- |
| Proyecto | GreeMarket |
| Fecha de preparación | 20/11/2024 |
| Cliente | Asociacion de jardineros |
| Patrocinador principal | Líder de la asociación |
| Gerente / Líder de Proyecto | Juan Herrera |
| Gerente / Líder de Pruebas de Software | Dante Ruiz |

# Aprobaciones

| **Nombre y Apellido** | **Cargo** | **Departamento u Organización** | **Fecha** | **Firma** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Resumen Ejecutivo

Resumen de todo el contenido del plan de Pruebas de Software, describe cuál es su propósito, establece si es un plan maestro o un plan detallado, identifica el alcance del plan de pruebas en relación con el plan de Proyecto de Software, restricciones (por ejemplo de recursos o presupuesto), alcance del esfuerzo de pruebas entre otros aspectos.

Lo que se busca con este documento es dejar establecido las pruebas que se harán, como se llevarán acabo y los responsables de que estas pruebas cumplan con el objetivo de lograr los estándares de calidad, del producto que se está ofreciendo, cuyo plan principal es albergar a una gran cantidad de personas en la plataforma y para ello se debe de revisar y testear su la infraestructura que se está teniendo, cumple cumple con lo necesario para poder lanzar el producto al mercado con la calidad que se estableció.

# Alcance de las Pruebas

# Elementos de Pruebas

Listado de todos los módulos, componentes o elementos que se van a probar. Si es de alto nivel, se listan las áreas funcionales (módulos o procesos que cubre el Testing), por otro lado, si es de un nivel detallado se listan los programas, unidades o módulos.

| **Área Funcional** | **Módulo** | **Elemento de Prueba** | **Nivel de Prueba** |
| --- | --- | --- | --- |
| **Registro de Usuario** | Sistema  Proveedor | Formulario de registro, validaciones | Funcional, Unidad |
| **Inicio de Sesión** | Sistema Administrador, Proveedor | Verificación de credenciales y condiciones especiales | Funcional, Integración |
| **Autenticación Doble Factor** | Sistema Administrador, Proveedor | Envío y validación del código por correo electrónico | Funcional, Integración |
| **Catálogo de Productos** | Sistema  Cliente | Búsqueda y visualización de productos | Funcional |
| **Carro de Compras** | Sistema  Cliente | Agregar, eliminar, calcular total | Funcional, Unidad |
| **Historial de Compras** | Sistema Cliente | Visualización de compras previas | Funcional, Integración |
| **Historial de Ventas** | Sistema  Proveedor, Administrador | Registro de ventas por proveedor | Funcional |
| **Integración de Pago** | Sistema  Cliente | Conexión con sistema de pago externo (Transbank) | Integración |
| **Lista de Proveedores** | Sistema Administrador | Visualización y filtrado de proveedores | Funcional |
| **Gestión de Productos** | Sistema  Proveedor | Agregar, eliminar, modificar productos | Funcional |
| **Geolocalización** | Sistema Cliente | Localización de proveedores cercanos | Funcional, Integración |
| **Notificaciones** | Sistema  Cliente | Recepción de notificaciones sobre pedidos y ofertas | Funcional |
| **Chatbot Respuestas** | Sistema  Chatbot | Respuestas automatizadas, capacidad de diálogo | Funcional, Unidad |
| **Reporte Inteligente** | Sistema Administrador | Generación de reportes mensuales | Funcional |

# Nuevas Funcionalidades a Probar

Es un listado de lo que se va a probar “Desde el Punto de vista del Usuario”. No es una descripción técnica del software sino sus características y funcionalidades. Se incluyen tanto las que son nuevas como las que se están modificando.

1. **Autenticación de Doble Factor**Descripción: Validar que administradores y vendedores puedan iniciar sesión con un código enviado a su correo electrónico para mayor seguridad.
2. **Geolocalización**Descripción: El sistema pedirá la ubicación del usuario para localizar proveedores de venta cercanos, este funcionará de manera de filtro para encontrar a los proveedores cercanos a la ubicación del cliente.
3. **Historial de Compras**Descripción: Asegurar que el cliente pueda visualizar todas las compras realizadas con detalles como fecha, productos, total pagado y estado del pedido.
4. **Carrito de Compras**Descripción: Validar que el cliente pueda agregar y eliminar productos del carrito, y visualizar el total de la compra antes de proceder al pago.
5. **Catálogo de Productos**Descripción: Probar que los clientes puedan visualizar los productos ofrecidos por los proveedores, incluyendo imágenes, precios y descripciones.
6. **Autocompletado del Cliente**Descripción: Comprobar que al momento de pagar, los campos de información del cliente (RUT, nombre, dirección) se completen automáticamente para facilitar la experiencia de compra.
7. **Integración de Sistema de Pago**Descripción: Validar que el cliente pueda realizar pagos a través del sistema externo integrado, asegurando la correcta comunicación con el servicio de pago.
8. **Detalle del Producto**Descripción: Asegurar que los clientes puedan ver información detallada del producto, como descripción, precio, calificación y reseñas.
9. **Registro de Usuario**Descripción: Comprobar que los vendedores puedan registrarse correctamente en el sistema con los datos solicitados.
10. **Local de Venta**Descripción: Validar que los proveedores puedan agregar la ubicación de sus locales en el sistema, para que los clientes puedan encontrarlos fácilmente.

# Pruebas de Regresión

Listado de las funcionalidades no directamente involucradas en el desarrollo, pero cuyos componentes están siendo afectados y por ende deben probarse para asegurar que continúan funcionando adecuadamente. Al igual que en el punto anterior, se describen desde el punto de vista del usuario.

**Inicio de Sesión**

* **Motivo**: Es fundamental verificar que los usuarios puedan seguir iniciando sesión correctamente, especialmente después de implementar la autenticación de doble factor.

**Registro de Usuario**

* **Motivo**: Validar que el proceso de registro para nuevos vendedores o administradores no se vea afectado por los cambios en la seguridad o la integración de otras funcionalidades.

**Historial de Ventas**

* **Motivo**: Probar que los vendedores y administradores puedan seguir visualizando las ventas realizadas, incluso después de los cambios relacionados con el historial de compras o la integración de datos entre diferentes usuarios.

**Lista de Productos (Proveedor)**

* **Motivo**: Asegurar que los proveedores puedan listar, agregar, modificar y eliminar sus productos correctamente tras las modificaciones relacionadas con el catálogo de productos y el carrito de compras.

**Agregar Nuevos Proveedores**

* **Motivo**: Verificar que el administrador pueda seguir añadiendo nuevos proveedores, asegurándose de que las funcionalidades de registro no interfieran con esta acción.

**Eliminar Proveedores**

* **Motivo**: Comprobar que el administrador pueda eliminar proveedores sin que esto afecte a otros procesos del sistema, como el catálogo de productos.

**Creación de Reportes Inteligentes**

* **Motivo**: Validar que el sistema continúe generando reportes de ventas mensuales de manera correcta, a pesar de los cambios realizados en las funcionalidades de ventas o compras.

**Recuperación de Contraseña**

* **Motivo**: Probar que la funcionalidad de recuperación de contraseña sigue funcionando después de la implementación de nuevos métodos de autenticación.

**Sistema de Notificaciones**

* **Motivo**: Asegurar que las notificaciones relacionadas con pedidos, ofertas y actualizaciones de estado sigan enviándose correctamente después de los cambios en el carrito o en la integración del sistema de pago.

**Buscar Producto (Proveedor)**

* **Motivo**: Validar que los proveedores puedan buscar productos en su lista después de implementar las nuevas funcionalidades en el catálogo de productos

# Funcionalidades a No Probar

Listado de las funcionalidades que NO se van a probar. Debe incluir información de las razones por las cuales no se van a probar y los riesgos que se están asumiendo.

1. **Sistema de Notificaciones (App Móvil):**
   * **Razón:** Las notificaciones se han desactivado temporalmente debido a la falta de integración con el backend.
   * **Riesgo Asumido:** Los usuarios móviles podrían no recibir alertas de ofertas o actualizaciones de pedidos.
2. **Chatbot Multilingüe:**
   * **Razón:** La funcionalidad de cambio de idioma del chatbot no es crítica para la versión inicial y será abordada más adelante.
   * **Riesgo Asumido:** Usuarios que no hablen el idioma predeterminado podrían tener dificultades en la interacción inicial.
3. **Generación de Reportes Inteligentes (Administrador):**
   * **Razón:** Esta funcionalidad está planificada para la última etapa del proyecto.
   * **Riesgo Asumido:** Los administradores no tendrán análisis mensuales automáticos en las primeras etapas.
4. **Autocompletado de Datos del Cliente al Pagar:**
   * **Razón:** Este módulo está en desarrollo y no se encuentra totalmente integrado con la base de datos.
   * **Riesgo Asumido:** Los clientes deberán ingresar manualmente sus datos durante el proceso de pago.
5. **Detalles Avanzados de Productos:**
   * **Razón:** El sistema de información extra proporcionado por el bot sobre las plantas aún no está completamente implementado.
   * **Riesgo Asumido:** Los usuarios podrían no acceder a datos adicionales sobre los productos.

# Enfoque de Pruebas (Estrategia)

La Estrategia de Pruebas puede definirse como un documento aparte, o puede ser incluido dentro del Plan de Pruebas según su extensión. Aquí pueden definirse los tipos de pruebas a realizar (funcionales, de desempeño, de interfaces, no funcionales, etc.), requerimientos especiales de las pruebas, configuraciones a probar, subconjuntos de datos a considerar, nivel de pruebas de regresión, entre otros aspectos.

La Estrategia de Pruebas para el proyecto GreenMarket se basa en un enfoque integral, que busca asegurar que todas las funcionalidades sean probadas de acuerdo con su criticidad, impacto en el sistema y complejidad. Esta estrategia abarca los siguientes tipos de pruebas:  
**Tipos de Pruebas a Realizar:**

1. **Pruebas Funcionales**
   * **Objetivo**: Asegurar que todas las funcionalidades del sistema funcionen correctamente, según los requerimientos definidos en la plantilla de requerimientos.
   * **Descripción**: Se probarán todos los módulos del sistema, incluyendo el registro de usuario, inicio de sesión, carrito de compras, y funcionalidades específicas como la búsqueda de productos, historial de compras, y sistema de pagos.
   * **Configuración**: Entorno de pruebas en dispositivos móviles y web, con la base de datos predefinida.
2. **Pruebas de Integración**
   * **Objetivo**: Validar que los diferentes módulos y componentes del sistema trabajen correctamente cuando interactúan entre sí.
   * **Descripción**: Se probarán las integraciones entre el carrito de compras y el sistema de pagos, la conexión entre el catálogo de productos y el chatbot, y la integración entre el sistema de geolocalización y el listado de proveedores cercanos.
   * **Configuración**: Base de datos de productos y usuarios simulados, junto con la integración de los módulos backend.
3. **Pruebas Unitarias por Componente**
   * **Objetivo**: Asegurar que cada componente individual del sistema funcione correctamente de manera aislada.
   * **Descripción**: Se probarán las funciones del frontend (ej. validación de formularios) y los servicios del backend (ej. autenticación, gestión de carrito).
   * **Configuración**: Entorno de desarrollo con pruebas automatizadas para cada módulo o componente.
4. **Pruebas con Usuarios**
   * **Objetivo**: Validar la experiencia del usuario y la interacción con las funcionalidades del sistema.
   * **Descripción**: Se realizarán pruebas de usabilidad y aceptación, donde los usuarios reales interactúan con el sistema y proporcionarán retroalimentación sobre su facilidad de uso y la intuición de la interfaz.
   * **Configuración**: Prototipos y versiones alfa del sistema para pruebas de usuario en dispositivos reales.

# Criterios de Aceptación o Rechazo

## Criterios de Aceptación o Rechazo

Son los criterios que serán considerados para dar por completado el Plan de Pruebas de Software, por ejemplo: Completar 100% de pruebas unitarias, cierto porcentaje de casos exitosos, cobertura de todos los componentes y líneas de código, porcentaje de defectos corregidos, entre otros.

Se sacará el porcentaje mediante la cantidad de pruebas que se hagan en el área respectiva, un ejemplo práctico o aplicado sería si son 10 pruebas funcionales que se realizarán, de estas se verán cuántas fueron exitosas y a raíz de eso se sacará el porcentaje que se aplicará a esta tabla de abajo.

**Resumen de Criterios para Aceptación y Rechazo**

* **Éxito:**
  + Si el porcentaje de pruebas exitosas es **del 90% al 100%**.
  + Si el 100% de los defectos críticos están corregidos y el porcentaje de defectos menores corregidos es **más del 80%**.
  + Si la cobertura de código es **del 80% o más**.
  + Si todas las funcionalidades principales se probaron y cumplieron los requisitos funcionales.
* **Rechazo:**
  + Si el porcentaje de pruebas fallidas es superior al **50%**.
  + Si la cobertura de código es inferior al **80%**.
  + Si más del **20%** de los defectos no se han corregido.
  + Si alguna funcionalidad clave no ha sido probada.
* **Segunda vuelta (50% - 80%):**
  + Si la tasa de pruebas exitosas está entre el **50% y el 80%**, se requiere una segunda vuelta de pruebas después de realizar las correcciones necesarias.

## Criterios de Suspensión

Establece claramente bajo qué condiciones se detienen un conjunto de casos de pruebas, por ejemplo en caso de existir defectos que impidan la ejecución de más casos de pruebas, cierto porcentaje de casos fallidos, o cualquier otro que se especifique.

1. **Porcentaje de Pruebas Fallidas**:
   * **Suspensión Automática**: Si el **porcentaje de pruebas fallidas** supera el **50%** en cualquiera de los tipos de pruebas (funcionales, unitarias, de integración, etc.), las pruebas se suspenderán y no se continuará hasta que se corrijan los defectos identificados.
   * **Causa**: Las pruebas no tienen suficiente cobertura ni confiabilidad para continuar.
2. **Defectos Críticos No Corregidos**:
   * **Suspensión Automática**: Si hay defectos críticos (por ejemplo, aquellos que afectan las funcionalidades principales como el inicio de sesión, el proceso de pago, etc.) que no se han corregido, las pruebas se suspenderán hasta que se resuelvan los problemas.
   * **Causa**: Estos defectos impiden la correcta funcionalidad del sistema y afectan la ejecución de las pruebas de otras funcionalidades.
3. **Cobertura de Código Insuficiente**:
   * **Suspensión Automática**: Si la cobertura de código es inferior al **80%**, se suspenderán las pruebas para evitar que funcionalidades importantes no sean probadas adecuadamente.
   * **Causa**: La baja cobertura implica que muchas áreas del sistema no están siendo evaluadas, lo que puede resultar en una evaluación incorrecta del software.
4. **Módulos No Probados**:
   * **Suspensión Automática**: Si algún módulo o funcionalidad crítica (como carrito de compras, pagos, geolocalización) no ha sido probado debido a problemas de integración o falta de pruebas, las pruebas se suspenderán hasta que dichos módulos sean evaluados.
   * **Causa**: La falta de pruebas de funcionalidades clave podría poner en riesgo la estabilidad general del sistema.
5. **Defectos No Corregidos por Sobrepasar Umbrales Definidos**:
   * **Suspensión Condicional**: Si más del **20%** de los defectos identificados no han sido corregidos, las pruebas se suspenderán hasta que los defectos sean corregidos y se pueda realizar una nueva ronda de pruebas.
   * **Causa**: La alta tasa de defectos no corregidos sugiere que el sistema no está listo para continuar con el siguiente ciclo de pruebas.

**Criterios de Suspensión por Tipo de Prueba:**

**1. Pruebas Funcionales:**

* **Suspensión Automática**: Si más del **50%** de las pruebas funcionales fallan o si existen defectos críticos que afectan la funcionalidad principal del sistema (como el proceso de pago o la autenticación), se suspenderán hasta que los defectos sean corregidos.

**2. Pruebas de Integración:**

* **Suspensión Automática**: Si el **porcentaje de pruebas de integración fallidas** supera el **50%** o si existen defectos de integración entre componentes clave que afecten las funcionalidades críticas, las pruebas se suspenderán.

**3. Pruebas Unitarias por Componente:**

* **Suspensión Automática**: Si las pruebas unitarias de un componente fallan en más de un **50% de los casos** o si los componentes clave no están siendo probados correctamente debido a la falta de cobertura o errores en el código, se suspenden las pruebas hasta que los componentes sean corregidos.

**4. Pruebas con Usuarios (Pruebas de Aceptación de Usuario):**

* **Suspensión Automática**: Si los usuarios reportan fallos importantes en más del **50%** de las funcionalidades clave que afectan la experiencia de usuario (por ejemplo, funcionalidades como búsqueda de productos, pagos, historial de compras), las pruebas se suspenden.

**5. Pruebas de Regresión:**

* **Suspensión Condicional**: Si más del **20%** de las pruebas de regresión fallan debido a cambios recientes en el sistema, las pruebas se suspenden hasta que los cambios sean revisados y se resuelvan los defectos detectados.

# Criterios de Reanudación

Luego de haber suspendido las pruebas, aquí se establece bajo qué criterios se reanudaran.

1. **Resolución de Defectos Críticos Identificados:**
   * Todos los defectos críticos que impidan la ejecución de las pruebas deben ser resueltos y verificados.
   * Se debe realizar una prueba de validación para garantizar que el defecto corregido no introduzca nuevos errores.
2. **Ejecución de las Pruebas de Regresión:**
   * Completar un ciclo de pruebas de regresión para confirmar que los cambios realizados no afectan funcionalidades ya existentes.
   * Las pruebas de regresión deben obtener al menos un 80% de resultados exitosos para proceder.
3. **Disponibilidad del Entorno de Pruebas:**
   * El entorno de pruebas debe estar operativo y estable, con todos los recursos necesarios configurados (datos, servicios, conexiones).
4. **Aprobación del Equipo de Calidad:**
   * El equipo de QA debe evaluar y aprobar los cambios realizados, asegurando que se cumplan los estándares de calidad establecidos.
   * Debe emitirse un informe de estado que certifique que las pruebas pueden reanudarse.
5. **Actualización del Plan de Pruebas:**
   * Si los criterios de suspensión generaron cambios significativos en el alcance, deben reflejarse en una actualización del plan de pruebas antes de continuar.

# Entregables

Establece que se entregará como parte de la ejecución del plan, por ejemplo: Documento de Plan de Pruebas, Casos de Pruebas, Especificación de Diseño de Casos, Logs de errores, Reportes de incidencias, evidencias de pruebas, reportes emitidos por herramientas de pruebas y cualquier otro que se establezca.

# Recursos

## Requerimientos de Entornos – Hardware

Lista de los requerimientos de equipos, hardware y red necesarios para completar las actividades del Plan de Pruebas de Software. Incluye Servidores de Aplicación, Bases de Datos, Equipos de PC que necesitan los Testers, Conectividad a la red (incluyendo accesos), entre otros.

1. **Hardware:**
   * **Interfaz Web**
     + Procesador: Intel Core i3 3100 o superior, o equivalente AMD.
     + Memoria RAM: 4 GB como mínimo.
     + Conexión a Internet: 10 Mbps como velocidad mínima recomendada.
   * **Interfaz Móvil**
     + Dispositivos Android con conexión a Internet.
     + Memoria RAM: 2 GB como mínimo.
     + Espacio de almacenamiento disponible: 1 GB.
   * **Servidores**
     + **Servidor de Aplicación**:
       - Procesador: Intel Xeon E5 o superior, o equivalente AMD.
       - Memoria RAM: 8 GB como mínimo.
       - Espacio de almacenamiento: 500 GB HDD o 250 GB SSD.
     + **Servidor de Base de Datos (MariaDB)**:
       - Procesador: Intel Xeon E5 o equivalente.
       - Memoria RAM: 16 GB.
       - Espacio de almacenamiento: 1 TB HDD o 512 GB SSD.
   * **Red**
     + Red local o VPN para el acceso a los servidores desde diferentes ubicaciones.

## Requerimientos de Entornos – Software

Lista de los requerimientos de software necesarios para completar las actividades de prueba, puede incluir accesos a Sistemas (en entorno de pruebas) y Bases de Datos, así como instalación de software en los Computadores asignados a los Testers.

1. **Software**
   * **Interfaz Web**
     + Navegador Web Compatible:
       - Cualquier navegador que use Chromium (ej.: Google Chrome, Edge, Opera).
     + Compatibilidad:
       - Windows 10 u 11.
   * **Interfaz Móvil**
     + Compatibilidad:
       - Android 8.0 o versiones superiores.
   * **Servidores**
     + **Servidor de Aplicación**:
       - Sistema operativo: Ubuntu Server 20.04 o superior (ajustado para Python y MariaDB).
     + **Servidor de Base de Datos**:
       - Conexión a Internet para acceso remoto.
   * **Red**
     + Configuración del servidor con HTTPS para la comunicación segura entre cliente y servidor.
     + Firewall configurado para permitir el tráfico en los puertos necesarios (ej.: puertos 80 y 443).
   * **Herramientas de Desarrollo y Frameworks**
     + Framework de Desarrollo Frontend: Ionic Angular.
     + Backend y Servicios: Python (con Rasa como sistema de chatbot).
     + Base de Datos: MariaDB.
     + Plataforma de pruebas: Navegadores web y dispositivos móviles.
     + Soporte de API REST (Django): Para integración con servicios externos (ej.: sistema de pago).

## Herramientas de Pruebas Requeridas

Especifica las herramientas de software, metodologías o técnicas especiales empleadas en las pruebas, por ejemplo Herramientas de Automatización de Pruebas, Software de Gestión de Pruebas, entre otros.

En este caso dejaremos un listado de las sugerencias que se podrían usar para los variados casos de pruebas que se realizarán para la plataforma, estás sugerencias están sujetas a cambios o modificación, en caso de que se quiera agregar más herramientas o eliminar las sugerencias en caso de ser necesario:

1. **Herramientas de Automatización de Pruebas**
   * **Selenium**: Para pruebas funcionales automatizadas en navegadores, especialmente para la interfaz web.
   * **Appium**: Para automatizar pruebas en dispositivos móviles Android.
2. **Software de Gestión de Pruebas**
   * **TestRail** o **Zephyr**: Para planificar, rastrear y organizar los casos de pruebas, así como documentar resultados.
3. **Herramientas de Pruebas Unitarias**
   * **Pytest**: Framework para pruebas unitarias en el backend desarrollado en Python.
   * **Jest** o **Karma**: Para pruebas unitarias y de integración en el frontend usando Ionic Angular.
4. **Pruebas de Desempeño**
   * **Apache JMeter**: Para evaluar el desempeño del sistema bajo carga.
   * **Locust**: Para simular múltiples usuarios interactuando con la aplicación de manera concurrente.
5. **Pruebas de API y Servicios**
   * **Postman**: Para pruebas de integración y validación de APIs REST implementadas con Django.
   * **Swagger**: Para validar y documentar las APIs del sistema.
6. **Pruebas de Seguridad**
   * **OWASP ZAP**: Para identificar vulnerabilidades en la aplicación web, especialmente en autenticación y conexiones HTTPS.
7. **Pruebas de Accesibilidad**
   * **Lighthouse** (integrado en navegadores basados en Chromium): Para verificar la accesibilidad, velocidad y usabilidad de la interfaz web.
8. **Gestión de Defectos**
   * **Jira**: Para gestionar y rastrear defectos encontrados durante las pruebas.
9. **Herramientas de Colaboración y Gestión del Proyecto**
   * **Slack/Trello**: Para facilitar la comunicación y coordinación entre los testers y el equipo de desarrollo.
10. **Simulación de Red y Dispositivos**
    * **BrowserStack** o **Sauce Labs**: Para pruebas en diferentes navegadores y dispositivos simulados.
11. **Entorno de Pruebas Local**
    * **Docker**: Para la configuración y despliegue de entornos consistentes para pruebas.

# Personal

Lista del personal necesario para completar las actividades de pruebas, especificando sus roles, por ejemplo: Un (1) Líder de Pruebas, Cinco (5) Analista de Pruebas (Testers), Dos (2) especialistas en automatización de pruebas, entre otros.

En este caso el encargado de la realización de las pruebas sería Dante ruiz el se encargaría de todo, por otro lado el encargado de supervisión y de ayudar tester sería el ingeniero de software Matias Arteaga.

# Entrenamiento

Necesidades de entrenamiento en el Sistema o Aplicación, así como en las herramientas de prueba a utilizar.

**Entrenamiento en herramientas de prueba**Formación sobre cómo usar las herramientas que se definieron en el apartado de **Herramientas de Pruebas Requeridas**, por ejemplo:

* **Selenium/Appium**: Cómo escribir y ejecutar pruebas automatizadas para interfaces web y móviles.
* **Postman**: Cómo crear y ejecutar pruebas para APIs REST.
* **JMeter/Locust**: Cómo realizar pruebas de carga y desempeño.
* **Jira/TestRail**: Para gestionar casos de prueba y defectos.

**Técnicas de prueba y metodologías**Capacitación en buenas prácticas y estrategias para realizar pruebas de software, como:

* Diseñar casos de prueba efectivos (cubriendo funcionalidad, integración y regresión).
* Identificar, reportar y rastrear defectos.

# Planificación y Organización

## Procedimientos para las Pruebas

Especifica los procedimientos o metodología de pruebas a emplear durante la ejecución del plan de pruebas de software.

Los procedimientos de pruebas seguirán la metodología espiral, abordando iteraciones incrementales que incluyen: **revisión de requerimientos**, **definición de casos de prueba**, y **configuración del entorno**. Las pruebas se realizan en niveles: unitarias (validación de componentes individuales), integración (interacción entre módulos), funcionales (flujos completos), regresión (verificar que nuevas funciones no afecten las existentes) y pruebas con usuarios (retroalimentación de usabilidad). Se gestionan los resultados en función de criterios de aceptación predefinidos, y se suspenderán pruebas si surgen defectos bloqueadores. Tras corregir estos defectos, las pruebas se reanudarán garantizando estabilidad. Al finalizar cada iteración, se analizarán los resultados para mejorar los procedimientos en ciclos posteriores.

# Matriz de Responsabilidades

Lista cada una de las personas integrantes del equipo de QA y sus responsabilidades. Se puede hacer uso de una Matriz RACI (Responsable, Aprobador, Consultado, Informado).

*Matriz de Responsabilidad RACI.xlsx*. (s. f.). Google Docs. [Matriz de Responsabilidad RACI.xlsx](https://docs.google.com/spreadsheets/d/1hTHfM_vD7-wbiCeuf0lW4Bur3_U2R-ji/edit?gid=1342520566#gid=1342520566)

# Cronograma

Debe estar basado en estimaciones de actividades realizadas por el equipo de prueba. En él se Identifican los hitos relevantes en las pruebas de software, se establecen las dependencias (actividades predecesoras) y demás aspectos componentes de un cronograma.

| **Ciclo 4: Fase de Pruebas** |
| --- |
| **Pruebas funcionales** |
| **Pruebas de Integración** |
| **Pruebas Unitarias por componente** |
| **Pruebas con Usuarios** |
| **Pruebas de integración Final** |
| **Evaluación de Riesgos de Pruebas** |

# Premisas

Las premisas relacionadas con las tareas de pruebas de software, incluyendo limitaciones de tiempo, disponibilidad de recursos que se asumen, uso de una metodología de pruebas, uso de una herramienta, entre otros.

**1. Metodología de Pruebas**

* Se utilizará la **metodología espiral** para el desarrollo y las pruebas del sistema.
* Las pruebas se ejecutarán en ciclos iterativos al finalizar cada fase de desarrollo para garantizar la calidad antes de avanzar al siguiente nivel.

**2. Limitaciones de Tiempo**

* Las pruebas deberán completarse dentro de los plazos definidos para cada iteración en la metodología espiral.
* Cada fase de pruebas contará con un margen de tiempo específico para la corrección de defectos detectados.

**3. Disponibilidad de Recursos Humanos**

* El equipo de pruebas estará compuesto por:
  + Un líder de pruebas encargado de supervisar las actividades.
  + Testers responsables de ejecutar casos de prueba manuales y automatizados.
  + Ingenieros de QA para pruebas de integración, regresión y desempeño.

**4. Herramientas de Prueba**

* Se utilizarán herramientas específicas para facilitar y optimizar las pruebas, incluyendo:
  + **Selenium/Appium** para pruebas automatizadas.
  + **Postman** para pruebas de API.
  + **JMeter o Locust** para pruebas de carga y desempeño.
  + **Jira/TestRail** para la gestión de casos de prueba y defectos.

**5. Entornos de Prueba**

* El sistema será probado en entornos que reflejen las configuraciones reales del usuario:
  + Interfaz Web: Navegadores compatibles con Chromium, en sistemas operativos Windows 10 y 11.
  + Interfaz Móvil: Dispositivos Android con versiones 8.0 o superiores.
  + Servidores: Configuración de Ubuntu Server 20.04 o superior con MariaDB para la base de datos.

**6. Limitaciones Técnicas**

* Se requiere conectividad constante a internet para realizar pruebas funcionales y de integración.
* Las pruebas de desempeño se realizan en horarios de baja actividad para evitar interrupciones en el desarrollo.

**7. Datos de Prueba**

* Los datos de prueba incluirán escenarios reales simulados y se validará la capacidad de manejo de errores en entradas incorrectas.
* Se generarán subconjuntos de datos específicos para pruebas de integración y regresión.

**8. Criterios de Aprobación**

* Las pruebas serán aprobadas si cumplen con los criterios establecidos previamente (90%-100% de éxito).

# Dependencias y Riesgos

Aquí se listan los riesgos asociados con el proceso de pruebas de software, por ejemplo, algunas fuentes de riesgos suelen ser:

* Dependencias con Desarrollos.
* Dependencias con otros proyectos.
* Disponibilidad de recursos.
* Restricciones de tiempo.
* Premisas que resulten no ser ciertas.

Los riesgos se pueden clasificar en función de su probabilidad e impacto, cada uno debe contemplar un plan de mitigación para evitar que ocurra o plan de contingencia cuando el riesgo no puede mitigarse y tiene que aceptarse.

**Premisas no cumplidas**:

* Si el equipo de desarrollo no entrega los módulos en los plazos establecidos, las pruebas podrían retrasarse.
* La conectividad a internet es esencial para ejecutar pruebas en servicios externos.

**Disponibilidad de recursos**:

* Disponibilidad de testers con experiencia en herramientas de prueba específicas (Selenium, Postman, etc.).
* Acceso a entornos de pruebas configurados correctamente (servidores, base de datos, dispositivos móviles, etc.).

En el caso de los riesgos se adjuntará la plantilla de gestión de riesgos, debidos a que hay se expresa de mejor manera los planes de mitigación y contingencia que se asignaron a los riesgos que se evaluaron en el ciclo 4, que corresponde a la fase de pruebas

*Riesgos de GreenMarket.xlsx*. (s. f.). Google Docs. [Riesgos de GreenMarket.xlsx](https://docs.google.com/spreadsheets/d/1tSP-nFxv0WDXg7lL6VdMFawFc92UPY1O/edit?gid=744787509#gid=744787509)

# Referencias

Lista de todos los documentos que pueden citarse como apoyo o para ampliar el contenido del plan de pruebas. Algunos ejemplos de lo que se puede hacer referencia aquí son:

* Plan de Proyecto.
* Plantilla de Requerimientos.
* ERS
* Gestión de riesgos

# Glosario

Definiciones de términos usados en la documentación, y general sobre el área de pruebas.