# **Plan de pruebas**

## **Introducción plan de pruebas**

Este plan de pruebas se diseñó para definir cuáles pruebas serán realizadas en el proyecto de EcoMarket 2.0 y cómo se harán, además de los riesgos y suposiciones. Para llevarlo a cabo, se hará bajo el estándar de calidad ISO 25000, el cual proporciona un marco de referencia para la evaluación de la calidad del software y sus requisitos y el estándar de calidad ISO/IEC/IEEE 42010.

## **Recursos**

| **Tester** | **Porcentaje de participación** |
| --- | --- |
| Diego Muñoz | 100% |

## **Alcance**

Las pruebas incluyen las pruebas unitarias de los componentes Service del proyecto de EcoMarket 2.0, las pruebas de integración de los componentes RestController del proyecto y las pruebas de interfaz de usuario simuladas.

## **Fuera del alcance**

Por el momento, solo se tomarán en cuenta las pruebas unitarias, de integración y de interfaz de usuario simuladas, por lo que otras pruebas no serán parte de este plan de pruebas.

## **Pruebas unitarias**

Las pruebas unitarias serán realizadas usando los frameworks JUnit y Mockito. Se harán simulaciones del comportamiento de los métodos.

## **Pruebas de integración**

Las pruebas de integración para los componentes RestController serán realizadas usando los frameworks JUnit y Mockito, además de MockMVC. Se harán simulaciones a las peticiones HTTP.

## **Pruebas de interfaz de usuario simuladas**

Las pruebas de interfaz de usuario simuladas se realizarán en Figma, la cual es la herramienta que usamos para crear el mockup, y en la cual se evaluará que se muestren pantallas en distintos casos.

## **Infraestructura**

El dispositivo que será usado para realizar las pruebas debe contar con todos los programas y frameworks necesarios para realizar las pruebas.

## **Riesgos**

| N° | Riesgos | Probabilidad (1 - 5) | Impacto (1 - 5) | Severidad (Probabilidad \* Impacto) | Plan de mitigación |
| --- | --- | --- | --- | --- | --- |
| 1 | Los microservicios no están completamente implementados. | 3 | 5 | 15 | Reunirse con el equipo de desarrollo y evaluar lo que falta. |
| 2 | Las pruebas tienen errores. | 2 | 5 | 10 | Hacer una revisión del código para encontrar el error y corregirlo. |
| 3 | Errores en las anotaciones de Mockito o JUnit. | 4 | 5 | 20 | Revisar meticulosamente que las anotaciones incluídas en las clases sean las correctas. |