**Plan de pruebas**

El objetivo de este documento es recoger los casos de pruebas que verifican que el sistema satisface los requisitos especificados, para esto iniciaremos creando un plan que nos servirá para enfocar las pruebas de forma organizada y realizable. Haremos todo por medio de explicaciones textuales y un algún diagrama para mostrar mejor como queremos realizar el proceso de pruebas y explicar con claridad que se va a llevar a cabo.

**Alcance:** Este documento va dirigido hacia las personas encargadas de revisar el funcionamiento del programa para la realización de un contrato, se entregará un documento completo con toda la información realizada en las pruebas.

**Roles:** Daniel Esteban Aguilera Figueroa - **Líder de pruebas de software**

José Vicente Rincón Celis - **Autor del documento**

**Criterios de aceptación o rechazo:**

* Completar 100% de pruebas unitarias

**Riesgos:**

Dado que el proyecto no sigue en desarrollo, no depende de otro proyecto y no se encuentran restricciones de tiempo graves podemos dar por hecho que no hay ningun tipo de riesgo presente durante el desarrollo de estas pruebas. Aunque, por el mismo hecho de que el proyecto no está siendo desarrollado, el hecho de que surja un error concreto en las pruebas significaría que se tendría que modificar y ajustar para así solucionar cualquier problema ocasionado con este.

**Requerimientos de Entornos** – Hardware:

-Equipos de computacion que tengan como sistema operativo windows.

-Conectividad a la red.

-Base de datos.

-Todas las herramientas analogas necesarias, es decir, mouse, teclado, etc..

**Requerimientos de Entornos** – Software:

-Equipos de computo con la aplicación instalada.

-Acceso a la base de datos.

**Software:**

Las pruebas seran realizadas en el entorno de eclipse gracias a la librería JUnit la cual nos permitirá hacer un display y realizamiento comodo de las pruebas. Adicionalmente, JUnit nos permite realizar pruebas unitarias, de esta forma es posible probar cada metodo perteneciente a una clase sin problema alguno. Las pruebas seran realizadas desde un entorno local (Computador de mesa). Esto nos permitirá un mejor monitoreo de forma local. Es necesario instalar eclipse para poder realizar las pruebas del programa. Mas información acerca de la instalación en:

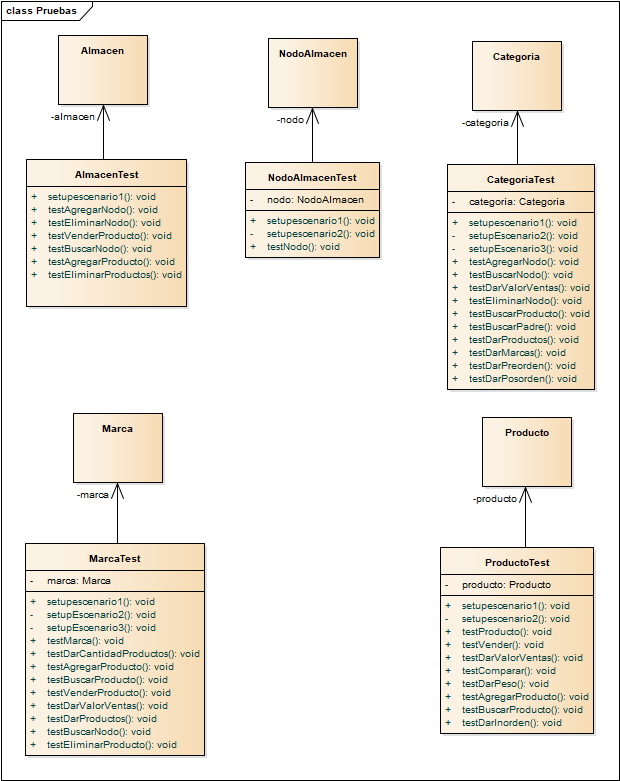
Instalacion eclipse:

[Eclipse Downloads | The Eclipse Foundation](https://www.eclipse.org/downloads/)

Instalación JUnit 5:

https://www.softwaretestinghelp.com/download-and-install-junit/

**Planeacion de pruebas:**

Para iniciar la planeacion es necesario analizar y comprender que queremos hacer y que aspectos del proyecto queremos tomar en cuenta para el realizamiento de las pruebas. En primer lugar, se nos fue encargada la tarea de realizar pruebas sobre dos clases del proyecto, las cuales son categoria y almacen. Despues de analizar el codigo y revisar aspectos relacionados obtuvimos un insight de su funcionamiento y demas.

MN

Modelo de pruebas (Carpeta Specs)

Gracias al modelo de pruebas otorgado en la carpeta specs del proyecto tenemos una mejor vista de la forma en la que todas serán realizadas. Para iniciar con la realización de todo el proceso es necesario tener en cuenta los aspectos a revisar de cada clase, en este caso con nuestra clase almacén tenemos unos requerimientos establecidos los cuales serán nuestros sujetos de prueba, aunque ya puede ser visualizado en el modelo mostrado anteriormente.

1. **Set up escenario**
2. **Agregar Nodo**
3. **Eliminar nodo**
4. **Buscar nodo**
5. **Agregar producto**
6. **Eliminar producto**

Por otro lado, para nuestra clase categoría tenemos más, los cuales son:

1. **SetUp escenario 1**
2. **SetUp escenario 2**
3. **SetUp escenario 3**
4. **Agregar nodo**
5. **Buscar nodo**
6. **Buscar Producto**
7. **Dar valor de ventas**
8. **Dar productos**
9. **Dar marcas**
10. **Dar Preorden**
11. **Dar PosOrden**

Ya teniendo los requerimientos que vamos a trabajar nada mas nos basta planear las pruebas de forma organizada. La forma que planeamos optar es organizar todo de tal forma que siempre que operemos o intentemos operar genere algún tipo de error ya que esto nos ayudará a identificar si en algún caso el método no funcionó y demás.

Un diagrama más ubicado a nuestra estrategia seria:

Diagram

Description automatically generated

En resumen, nuestra estrategia se enfoca en buscar el error de forma exhaustiva, si no se encuentra significa que el método funciona y no hay necesidad de realizar cambios en este.

**Referencias:**

* [**https://www.eclipse.org/downloads/**](https://www.eclipse.org/downloads/)
* **https://www.softwaretestinghelp.com/download-and-install-junit/**