Arquitecturas Web

Integrador nº5

Grupo 9
Integrantes:
- Lopez, Camila,
- Cañada, Nicolas,
- Amici, Cristian Matias,
- Cesario, Guillermo Tomas.

¿Qué queremos lograr con la app?

La principal idea es facilitar la administración de una despensa online. Principalmente permite llevar un registro de los productos que se almacenan y se venden. También un registro de los clientes que consumen los productos y poder a través de ellos estudiar su comportamiento en la tienda, permitiendo mejorar y adaptar las ofertas y el stock a las demandas de los clientes.

Lenguajes y tecnologías utilizadas:

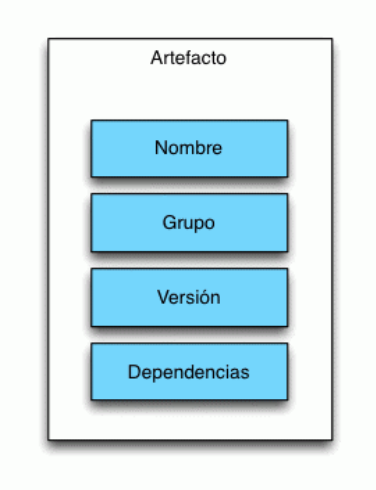
- Java, Html, Css, JavaScript, Sql.
- Maven, Junit, Spring Boot, Hibernate, Tomcat.
- Documentación: JavaDOC, OpenApi.

¿Cómo funcionan estas tecnologías?

Java, Html, Css , JavaScript, se usan para el desarrollo de aplicaciones que convierten a la Web en un elemento más interesante y útil. Java no es lo mismo que javascript, se trata de una tecnología sencilla que se usa para crear páginas web y solamente se ejecuta en el explorador.

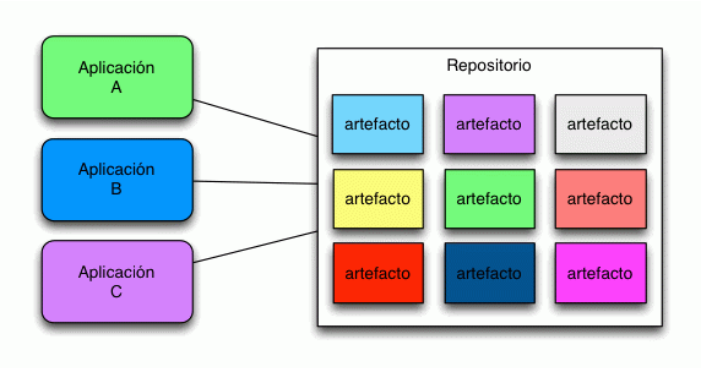
En cambio, SQL puede administrar datos, en particular la información organizada en tablas que se encuentra en los sistemas de administración de bases de datos relacionales. Mediante el uso de SQL, se puede consultar, actualizar y reorganizar datos, también crear y modificar la estructura de un sistema de base de datos y algo muy importante controlar el acceso a esos datos.

Maven funciona a través del concepto de Artefacto. Un Artefacto puede verse como una librería que agrupa más conceptos. Contiene las clases propias de la librería pero además incluye toda la información necesaria para su correcta gestión (grupo, versión, dependencias etc).



Para definir un Artefacto necesitamos crear un fichero POM.xml (Project Object Model) que es el encargado de almacenar toda la información que hemos comentado anteriormente

Una vez que definimos todos los artefactos que necesitamos, Maven nos provee de un Repositorio donde alojar, mantener y distribuir estos. Permitiéndonos una gestión correcta de nuestra librerías, proyectos y dependencias.



Hibernate es una herramienta de mapeo objeto-relacional (ORM) que facilita el mapeo de atributos en una base de datos tradicional, y el modelo de objetos de un aplicación mediante archivos declarativos o anotaciones en los beans de las entidades que permiten establecer estas relaciones.

Java Spring Boot es una de las herramientas principales del ecosistema de desarrollo web backend con Java, principalmente en arquitecturas basadas en servicios web (REST y SOAP) y microservicios, provee una serie de contenedores web servlet para que se despliegue nuestra aplicación automáticamente solo con un “Run”, de los contenedores web usamos el Tomcat.

OpenAPI, fue elegida principalmente porque define una interfaz estándar, independiente del idioma, para las API HTTP que permite a cualquier entidad descubrir y comprender las capacidades del servicio sin acceso al código fuente, la documentación o mediante la inspección del tráfico de la red. Cuando se define correctamente, un consumidor puede comprender e interactuar con el servicio remoto con una cantidad mínima de lógica de implementación. El documento de OpenAPI, tiene que contener al menos un campo de rutas , un campo de componentes o un campo de webhooks.

JUnit es utilizada para realizar testing unitario, sabiendo que es un conjunto de clases que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente.

La documentación Javadoc es una colección de páginas HTML de todas las clases, métodos, parámetros y retornos junto con la información y especificaciones que quiera incluir el desarrollador de la API que en el caso de las clases de JDK incluye abundantes e interesantes detalles de implementación a tener en cuenta al usar las clases. El Javadoc es también es una herramienta de línea de comandos que permite generar la colección de páginas HTML a partir del código fuente Java.

¿Cómo lo implementamos?

El Integrador 5 es la documentación, testing y mejora en la implementación visual del Integrador 4 que, como ya mencionamos, es de una despensa online.
En cuanto a desarrollo está implementado con spring boot, en un contenedor web servlet Tomcat y está basado en un servicio web Rest.
Para el mapeo de atributos de la base de datos, utilizamos Hibernate, que nos permitió y facilitó la implementación de entidades relacionales de la base de datos con las clases necesarias en Java y vincular nuestra aplicación web con la base de datos a través de un mapeo de dichas entidades (Diagrama de Objetos, figura 1). Las consultas a la base de datos fueron realizadas en el lenguaje nativo de Hibernate JPQL, pese a que no brinda toda la funcionalidad que ofrecería realizar consultas nativas, tenemos la ventaja que nos permite trabajar con la base de datos por medio de entidades en vez de Querys y nos ofrece un paradigma 100% orientado a objetos.
Utilizamos la Arquitectura MVC (Diagrama de Objetos, figura 1)que permite aislar sus tareas en capas y consiste en un patrón de diseño de software que se utiliza para separar en tres componentes los datos, la metodología y la interfaz gráfica de una aplicación. La gran ventaja que posee esta técnica de programación es que permite modificar cada uno de ellos sin necesidad de modificar los demás, lo que permite desarrollar aplicaciones modulares y escalables que se puedan actualizar fácilmente y añadir o eliminar nuevos módulos o funcionalidades de forma aislada del resto, ya que cada “paquete” utiliza el mismo sistema con sus vistas, modelos y controladores.
Para el proceso de testing usamos JUnit que está escrito en java puro y te permite crear tu propia suite de casos de prueba unitaria y es una de las herramientas más usadas a la hora de testear en java. Se crearon clases para probar los métodos y su comportamientos. También se instanciaron clases y se utilizaron métodos para comprobar que los retornos sean los esperados y las clases cumplieran sus funciones.
Para la documentación de la API usamos OpenApi, que nos pareció una herramienta sencilla y fácil de entender ya que es bastante descriptiva y permite comprender las capacidades del servicio sin acceso al código fuente.

En cuanto a la documentación de la aplicación usamos Javadoc, mediante comentarios se explicaron las distintas clases y los distintos métodos en lenguaje natural de cómo funcionan. Además se generó el archivo HTML que contiene todas las clases, métodos, parámetros y retornos junto con su información y especificaciones.

Diagrama objetos

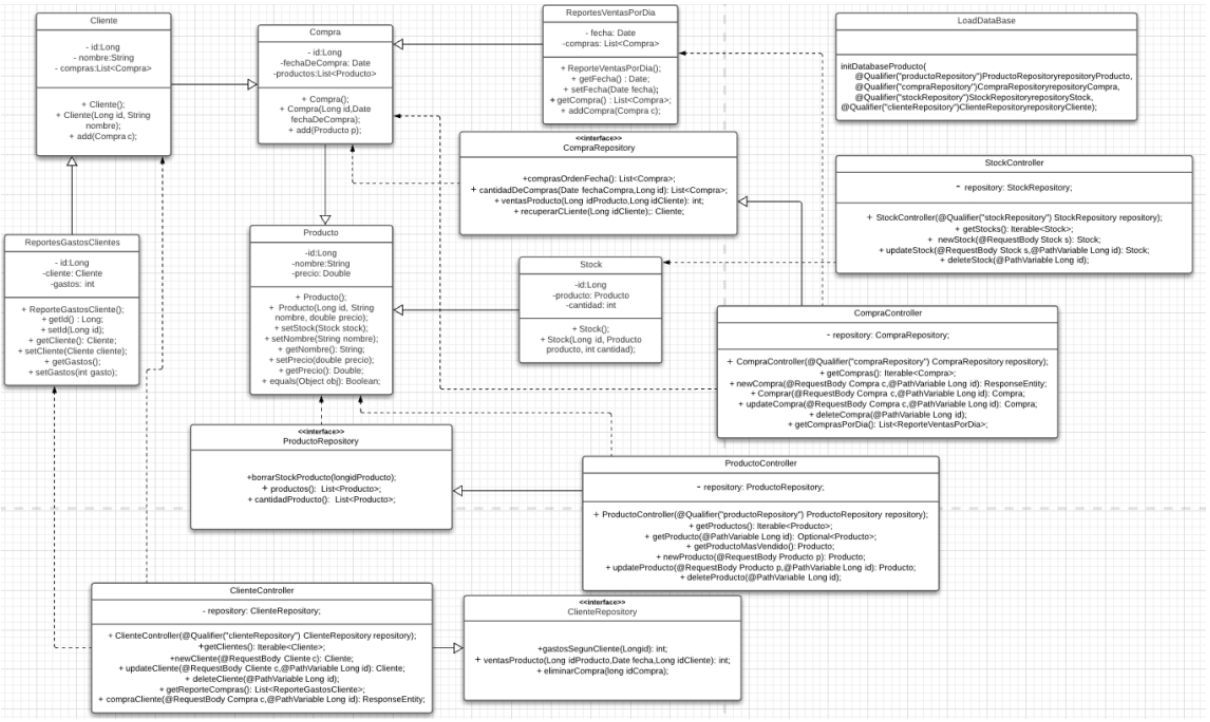


figura 1

Diagrama DERE

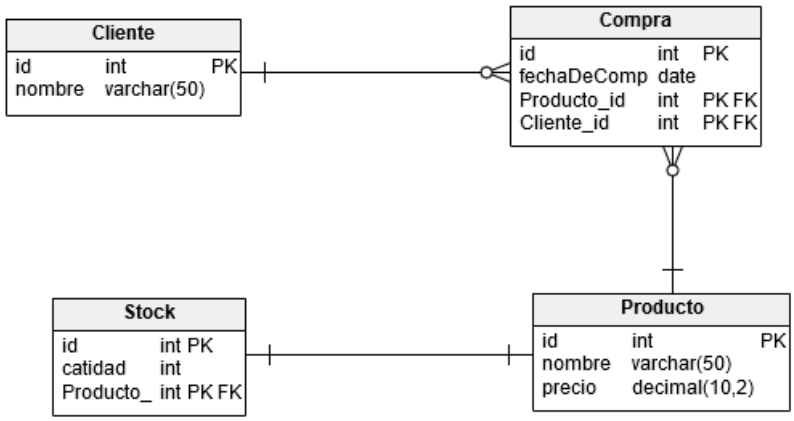


figura 2

Vista

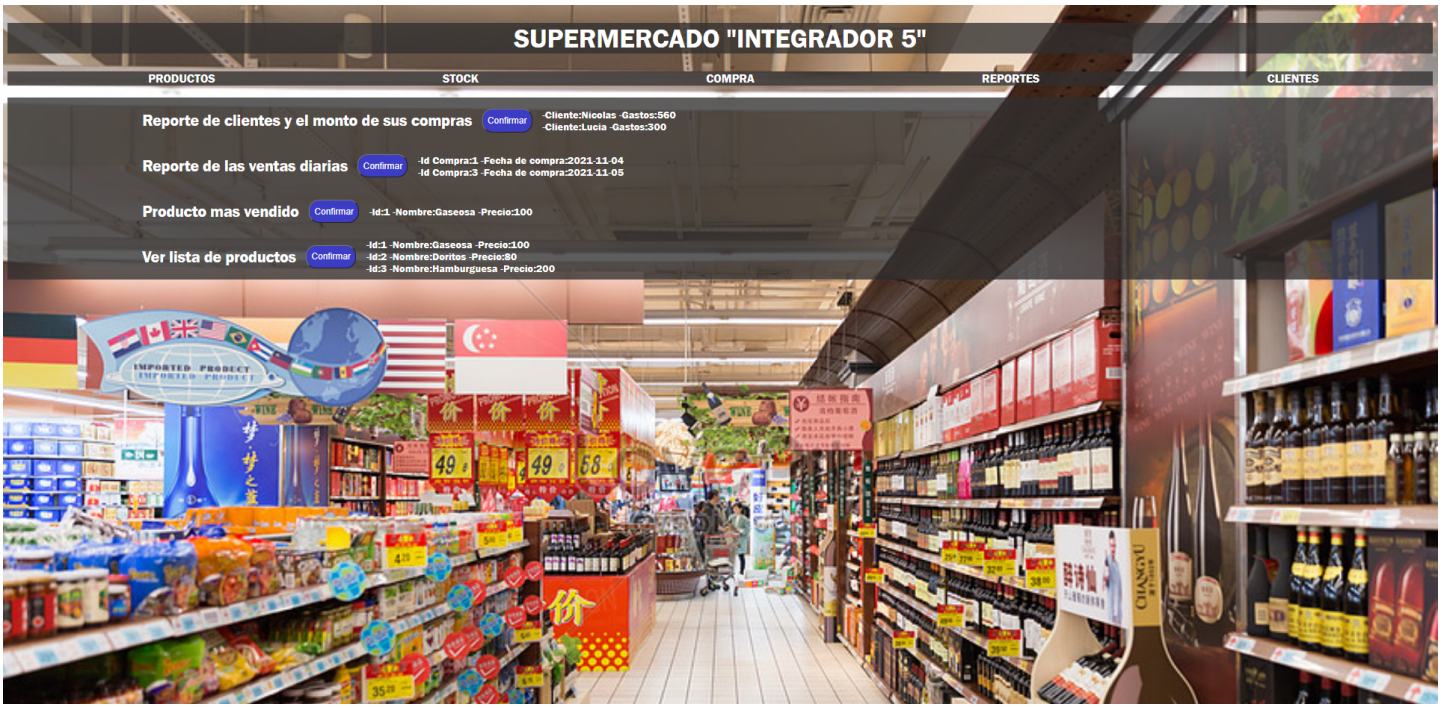


figura 3