Arquitectura back end para un comercio electrónico

Arquitectura con MVC para un comercio electrónico, se usará tecnologías como Nodejs, Express, para el desarrollo back end para la base de datos se usará MondoDB, ya que es base de datos no relacional, que nos ayudara a tener una mayor velocidad en nuestras consultas y tendremos una mayor escalabilidad verticalmente

Estructura de las carpetas

Project-comerce

App.js

Config.js

Server.js

- Controllers/
- ProductosController.js
- CarritoController.js
- OrdenController.js
- Models/
- Producto.js
- Carrito.js
- Orden.js
- Views/
- producto/
- carrito/
- orden/
- Routers/
- productoRoutes.js
- carritoRoutes.js
- orderRoutes.js

Controllers: contienen los controladores que manejan la lógica de la aplicación, cada controlador puede tener métodos para manejar operaciones especificas

Models: Almacena los modelos que representan los datos de la aplicación

Views: Esta carpeta podría contener plantillas o vistas para renderizar información, si se decide usar algún motor de platilla para el backend

Routes: Define las rutas de la aplicación y conecta las rutas a los controladores correspondientes

App: Inicializa y configura la aplicación Express, define el middleware y conecta las rutas

Config: Puede contener configuraciones globales como las credenciales de la base de datos, entre otras cosas importantes

Server: Inicia el servidor y escucha en un puerto especifico

Para el control de los archivos y cambios en el proyecto se usará Git, para tener registros de los cambios en algunos de los archivos.

Usando la arquitectura MVC buscamos organizar el código en tres componentes principales como el modelo, vista y controlador cada uno de los componentes tiene una responsabilidad especifica en la aplicación.

Modelo: El modelo representa la capa de datos y la lógica de negocio de la aplicación, se encarga de gestionar los datos y realizar operaciones sobre ellos y notificar a las otras partes de sistema sobre cualquier cambio.

Vista: La vista se encargará de la presentación de la información al usuario y de mostrar los datos proporcionados por el modelo.

Controlador: El controlador actúa como un intermediario entre el modelo y la vista, recibe las interacciones del usuario en la vista, procesa las interacciones y actualiza el modelo.

Algunas de las ventajas de la arquitectura MVC nos puede proporcionar una clara separación entre la lógica de negocio, la presentación y la gestión de las interacciones del usuario, debido a la separación de responsabilidades los componentes pueden ser reutilizados en diferentes partes también nos facilita el mantenimiento cuando se necesite hacer cambios en la lógica de negocio o en la interfaz de usuario pueden hacerse de manera más aislada sin afectar el sistema.