**Overview of Grocery Project**

This project is a simple e-commerce web software designed to provide customers with a web grocery purchasing experience. Built with Python and the Flask net framework, this task leverages Jinja2 for HTML templating, MySQL for records storage, and HTML/CSS for styling, growing an easy yet dynamic platform wherein users can browse and select items from a grocery catalogue. The absence of price capabilities makes it a focused, amateur-pleasant task best for information net software fundamentals together with consumer authentication, CRUD operations, and consultation management.

**Project Structure**

**1)Backend:**

* **Flask Framework:** Manages the routing, user authentication, and enterprise logic. Flask enables interplay among the net server, database, and front end, serving pages based on user moves.
* **MySQL Database:** Stores consumer info and catalogue facts, together with products with details along with name, price, and outline. It allows statistics staying power throughout classes, essential for capabilities like order records or user cart gadgets.

**2)Frontend:**

* **HTML and CSS:** Ensures a responsive and user-pleasant interface. Users can navigate through various pages, view products in an established layout, and get admission to their cart comfortably.

**3)Key Features:**

* **User Authentication:** A simple signal-in/signal-up mechanism to manipulate consumer periods. After signing in, users can upload gadgets to their cart and look at selected products.
* **Product Catalogue**: A list of to be had groceries, every with information like name, price, and description. Users can browse via categories and select items to feature to their cart.
* **Shopping Cart:** Users can add products to their cart, view itemized totals, and update portions. Session-based storage helps keep cart information during a person’s session.
* **Checkout:** A basic receipt that allows users to enter their personal details to process future orders. The lack of combined pay justifies the teaching of foundational concepts.

**Business process overview**

This grocery project follows the MVC (Model-View-Controller) design pattern:

• **Model:** The MySQL database contains all the data; it acts as a model of grocery list and user details.

• **View:** HTML and CSS templates provide a dynamic view layer, presenting data from the database to users in real time.

• **Controller:** Flask acts as a controller, identifying user requests, updating the model (e.g., adding items to cart), and identifying appropriate items

This role allows for hands-on experience in routing, templating, and database operations while demonstrating the interaction between front-end connectivity and back-end databases in a systematic, real-world web application environment.