**Flask**

Import all necessary libraries

* Flask is a lightweight web framework in Python.
* render\_template: Used to render HTML templates.
* request: Captures data from HTTP requests (like form submissions).
* redirect & url\_for: Redirect users to different routes (URLs) of the application.
* flash: Sends one-time messages (such as error messages) to the user.
* MySQL & MySQLdb.cursors: Modules for interacting with a MySQL database through Flask.

Initializes a new Flask web application

Sets up MySQL database connection with the database inventory\_system. It uses root as the user and the password is Ganesh@123

Flask uses routes to handle different URLs

**Login Page:**

* Retrieves the form data (username and password).
* Queries the MySQL database to check if the provided credentials exist.
* If the credentials are correct, the user is redirected to the home page.
* If not, an error message is flashed, and the user is redirected back to the login page.

**Signup Page:**

* Displays the signup page by rendering signup.html.
* Retrieves user details (username, password, email, phone number).
* Inserts the new user into the users table in MySQL.
* After successfully creating the account, it redirects the user to the login page.

**Home Page:**

* Main dashboard for the system

**Inventory Page:**

* Displays the current inventory from the inventory table in MySQL.
* Fetches all the inventory items and sends them to the view\_inventory.html template.

**Add Sales Page:**

* Handles the form submission when adding sales records.
* It retrieves sales data from the form.
* Inserts the sales record into the sales\_history table and updates the inventory table to reflect the sold products.
* Redirects the user to view the updated inventory.

**Sales History Page:**

* Displays the sales history from the sales\_history table.
* Fetches all sales records and sends them to the sales\_history.html template.

The Flask app will run in debug mode, meaning it will automatically reload when the code changes, and provide detailed error messages.

**Process:**

Login and Signup: Users can log in and sign up with credentials, which are verified or inserted into the MySQL database.

Inventory Management: Users can view the inventory and add new sales.

Sales History: Users can view the history of sales transactions.

Data Handling: The application uses MySQL to store and retrieve data, ensuring that form data is inserted or queried from the database.

**MySQL**

Create a new database called inventory\_system where all the tables will be stored.

**Users Table:**

* **id**: An integer that auto-increments with each new record. This column is the primary key, ensuring that each user has a unique identifier.
* username: A string with a maximum length of 50 characters. It cannot be NULL, meaning every user must have a username.
* password: A string with a maximum length of 100 characters. It cannot be NULL, meaning every user must have a password.
* email: A string with a maximum length of 100 characters. It cannot be NULL, meaning every user must provide an email address.
* phone\_number: A string with a maximum length of 15 characters. It cannot be NULL, ensuring every user provides a phone number.

**Inventory Table:**

* **id**: An integer that auto-increments for each new product added. It is the primary key for the table.
* product\_name: A string with a maximum length of 100 characters, which holds the name of the product.
* cost: A DECIMAL value representing the cost of the product. It has a precision of 10 (maximum 10 digits) and 2 decimal places.
* quantity: An integer that represents the number of units of the product in stock.
* date\_bought: A DATE field that records the date when the product was bought or added to inventory.

**Sales History:**

* **id:** An auto-incrementing integer used as the primary key.
* product\_name: A string with a maximum length of 100 characters that stores the name of the product sold.
* cost: A DECIMAL value representing the selling cost of the product, with 10 digits and 2 decimal places.
* quantity: An integer that indicates how many units of the product were sold.
* date\_sold: A DATE field that records when the product was sold.

**Scope of the Application**

**Functional Scope:**

User Management:

* Signup/Login system: Users can sign up and log in to access the system.
* Authentication: The system verifies users' credentials during login and ensures that only authorized users can manage inventory and sales data.

Inventory Management:

* Add Inventory: Users can add new products to the inventory by providing product details such as name, cost, quantity, and purchase date.
* View Inventory: Users can view all the products currently in the inventory, along with their associated information (e.g., cost, quantity, purchase date).

Sales Management:

* Add Sales: Users can record sales transactions, including product name, cost, quantity sold, and the date of sale. Each sale affects the inventory (e.g., if 10 units of a product are sold, they will be deducted from the inventory).
* Sales History: Users can view a history of all sales transactions made, enabling tracking of revenue, sales trends, and sold-out products.

Basic Reporting:

* Inventory valuation based on cost and quantity.
* Sales reports based on specific periods (daily, monthly, yearly).
* Stock alert system for low-quantity products.

**Database Scope:**

Database Entities and Relationships:

* users table
* inventory table
* sales\_history table

Data Integrity:

* Primary keys ensure that each record in the users, inventory, and sales\_history tables is unique.
* Not NULL constraints on critical fields like username, password, and quantity ensure that essential information is always provided.
* Auto-incremented IDs: Each table has an auto-incrementing id that serves as a unique identifier for the records.

Data Relationships:

* Currently, there are no foreign key relationships in the database schema, but this can be expanded.

**Limitations:**

* Passwords are stored in plain text without encryption, which is a major security risk.
* Products are not categorized (e.g., by type, supplier, or brand).
* Lack of Foreign Key Relationships: The database doesn’t explicitly link sales\_history to inventory, which may cause inconsistencies if inventory changes after a sale is made.
* The system does not have any mechanisms for alerting users when inventory levels are low or when restocking is required.
* There is no functionality to track suppliers or manage relationships with vendors.
* The system does not track customers or store customer data for sales transactions.
* There are no charts, graphs, or data visualization tools for analyzing sales performance or inventory trends over time.