Author:

Name Rakshith Subramanian
Rollno: 22f2001190
student email: 22f2001190@ds.study.iitm.ac.in
about: I am a pre final year student in Computer Science Department with a huge passion for Web Development and Algorithms

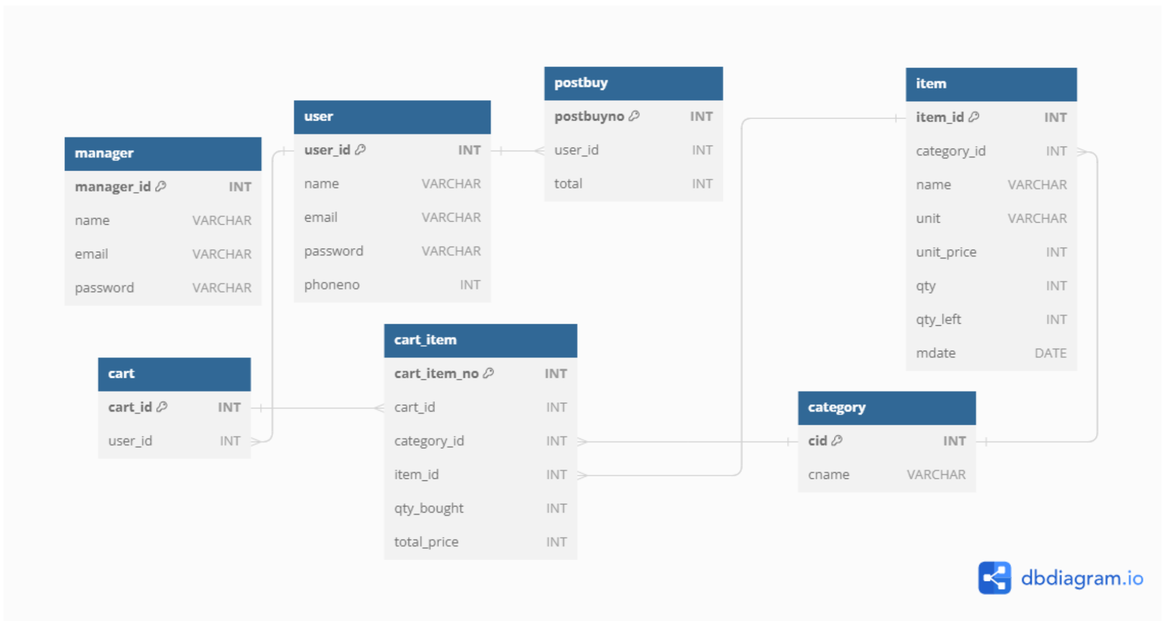
Description:

The application is a grocery store buying platform, which allows searching for available items or categories. The users can search, buy items from one or more categories and the manager can create, update, delete and view the various categories and items which are available to the user.

Technologies used:

- **Python** (Used for creating the backend)
    - **Flask** (Used to create the web application)
    - **Flask-SQLAlchemy** (Used for interacting with the SQLite database)
    - **Flask-RESTful** (Used for creating REST APIs in Flask)
    - **Jinja2** (Used for templating in Flask)
    - **Fernet** (Used for encrypting / decrypting data within the application)
    - ...and related dependencies for the above packages to function
- **SQLite** (Used for storing data)
- **HTML5** (Used to create a structure for the front end)
- **CSS3** (Used for custom styling)
- **Bootstrap** (Used for basic styling of the webpages)

DB Schema Design:



- The database was designed to scale, taking a vague inspiration from Big Basket.
- There are 4 main entities: **category** , **item** , **cart** , **cart_item**.
- These are related to each other via one-to-many relationships, as shown in the diagram above. The `user` and `manager` table share similarities but have different levels of access within the backend

API Design: The API has been created for the following features:

- **Categories:** To perform CRUD operations for Categories

- **Items**: To perform CRUD operations for items.

The API is not secure, it currently allows unrestricted access to perform these operations. The API documentation **(api.yaml)** file can be found in the **`docs`** folder in the **project folder.**

Architecture and Features:

The project follows an iteration of the **MVC model** and follows the fundamental idea of separation of concerns. The models for each of the tables are defined in the **`models`** folder, the controllers which contain the functional methods with respect to the models are stored in the **`controller`** folder, the files which control the UI and the usage of controllers with authentication validations are stored in the **`views`** folder.Additionally, there are some utility methods / classes for authentication, database connection, configuration and exceptions, which are stored in the **`utils`** folder. All these folders are collectively part of the **`app`** folder. The SQLite database is stored in the folder called **`db`.** The folder named **`docs`** contains the API documentation file and database schema design. The **`static`** and **`templates`** folder contain the static assets, like CSS files needed for styling the appearance, and the design templates respectively. Further details have been explained in the **README.md** file in the **root folder.**

Features implemented in the application are:

- Proper login and signup page for users and manager with symmetric encryption using Fernet

- Validation and authenticated access using a custom token-based authentication

- CRUD on Categories, Items , cart.

- Buying items and viewing cart for users

- Searching for categories/items using various parameters like name, price and manufacture date

- Displaying all the categories and their items with their details to the user

Video:

https://drive.google.com/file/d/152XZp98LdgCDIHId6yb--H_M8PM11aqi/view?usp=drive_link