Author

Name: Ashish Mehta
Roll number: 21f1006271
Email: 21f1006271@ds.study.iitm.ac.in
About: I am a bachelor interested in machine learning , research and development, and music.

Description

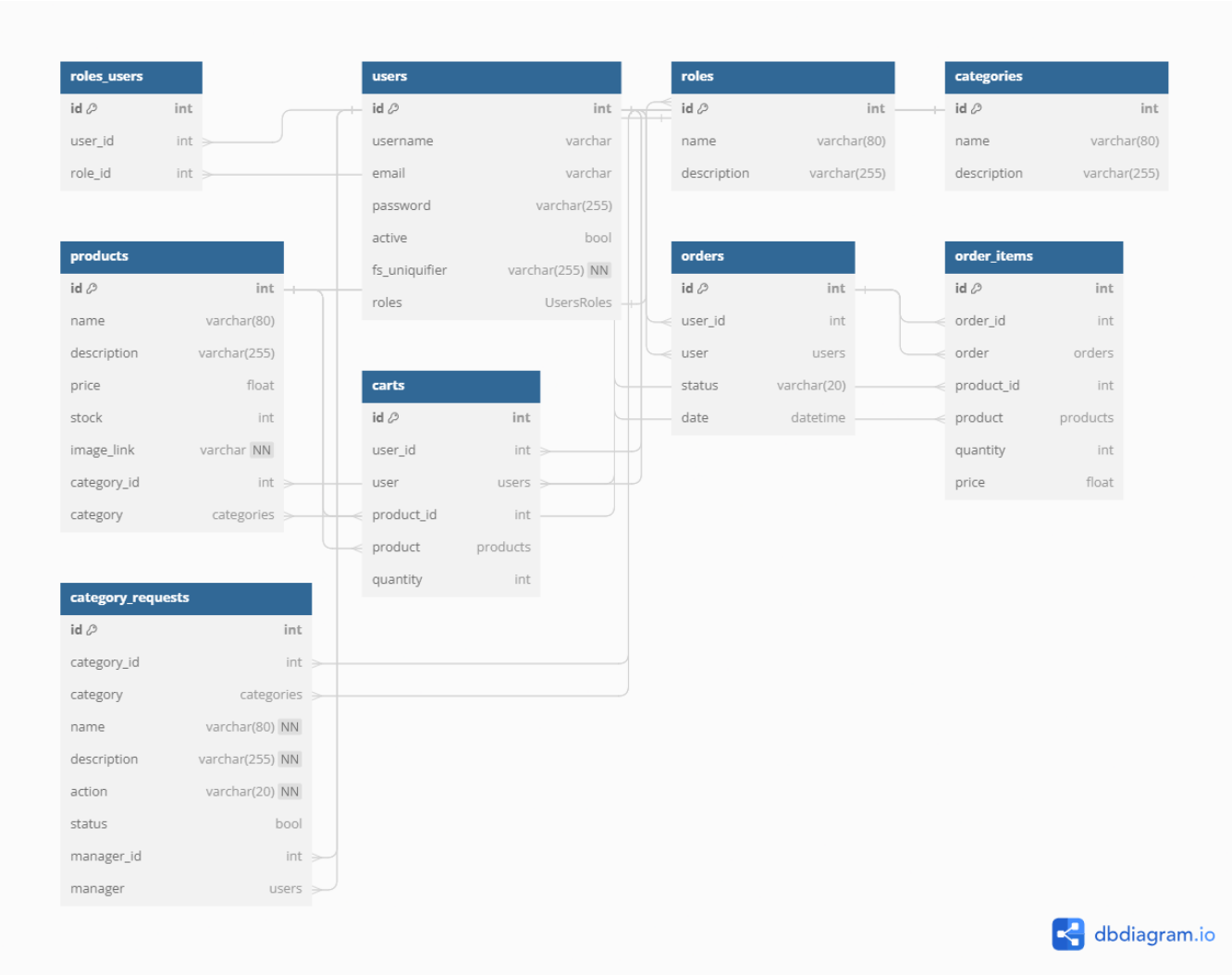
The application is a grocery app, which allows searching for products based on their name or category. The users can buy products,, the admin can create, update, delete, and view products and categories to be displayed to the users and the manager can request to create,update and delete categories.

Technologies used

- ***Python*** (Used for creating the backend)
    - ***Flask*** (for backend)
    - ***VueJS*** for UI
    - ***Flask-Security*** (For RBAC)
    - ***Flask-SQLAlchemy*** (Used for interacting with the SQLite database)
    - ***Flask-RESTful*** (Used for creating REST APIs in Flask)
    - ***Jinja2*** (Used for templating in Flask)
    - …and related dependencies for the above packages to function
- ***SQLite*** (Used for storing data)
- ***HTML5*** (Used to create a structure for the front end)
- ***Celery*** for backend asynchronous jobs
- ***Redis*** As an in-memory data store used for caching and improving data retrieval performance.

Database Schema Design

This database architecture employs SQLAlchemy in a Flask app, featuring user authentication, roles, categories, products, carts, orders, and category requests. It utilizes relational models like User, Role, Product, and their associations, facilitating e-commerce functionalities such as shopping cart management and order processing while enabling category requests managed by users with assigned roles.



API Design

- **Architecture & Frameworks:**
    - Utilizes Flask framework with Flask-RESTful for endpoint management.
    - Follows a RESTful design pattern for resource access and manipulation.
- **Resources & Endpoints:**
    - 'Product_Category' resource handles category operations.
        - GET: Retrieve categories
        - POST: Add a new category
        - DELETE: Remove a category
        - PUT: Update a category

- 'Product_Item' resource manages product-related actions.
    - GET: Fetch products
    - POST: Add a new product
    - DELETE: Delete a product
    - PUT: Update a product

- **Authentication & Authorization:**
    - Requires 'admin' role for specific actions; otherwise, access is denied.
    - Authenticates users using Flask-Security and role-based access control.
- **Caching & Performance:**
    - Implements caching with Flask-Caching to enhance performance.
    - Utilizes caching mechanisms for both 'Product_Category' and 'Product_Item' resources.
- **Error Handling:**
    - Utilizes standard HTTP status codes for error responses.
    - Provides structured error payloads for better interpretation.
- **Security Measures:**
    - Implements basic security measures to mitigate common threats (SQL injection, XSS, etc.).
    - Enforces data encryption and sanitization practices.
- **Request-Response Format:**
    - Requests and responses follow JSON payloads for structured data interchange.

Architecture and Features

The project is organized with distinct components.The backend folder contains the codes related to database models, celery, API, and templates for reports. API endpoints and routes are found in api.py, handling user interactions and logic. Configuration for asynchronous tasks are in celery_config.py. Database models are defined in models.py using SQLAlchemy. HTML templates reside in the templates folder. At the root level of " application" folder resides all the backend folders.. And the frontend files are in static folder

Along with the core features listed in the project description, like user authentication and authorization, RBAC, backend jobs, performance and caching, and all the other CRUD functionalities,.

Features implemented in the application are:

- Proper login and signup page for users, manager and admin.
- Approve New Manager and managers action request for Admin
- CRUD on category and product
- RBAC using Flask Security

- Searching for products using various parameters like name, description, and category
- Displaying all the products and their details to the user.
- Marks the unavailable products
- Ability for users to cart multiple products and purchase them

Video

https://drive.google.com/file/d/16YUzhC7Gwpl5nB4Tb11CQpYvaZBdm9YR/view?usp=sharing