FreshPick

Author

Arjun Ashish Pukale
22dp1000099
22dp1000099@ds.study.iitm.ac.in

I am a computer science engineer with 2.5 years of work experience, currently working as System Engineer in TCS in stock market domain creating web applications for trading.

Description

FreshPick is an online grocery shopping web app using which users can search and buy grocery products across multiple categories and Admin/Store managers can easily manage the products, categories and make changes that can reflect in live market.

Technologies used

Database:

- **SQLite**

Backend:

- **Python3**
- **Flask, Flask API**
- **Flask_sqlalchemy** (simplifies database interactions, reducing the complexity of SQL queries)
- **Flask_login** (to manage user authentication and jwt token)
- **Flask_bycrypt** (for security and password hashing)
- **Celery,Redis** (to enable background asynchronous tasks and scheduled jobs)

Frontend:

- **Vue js 3** (Empowering dynamic and reactive user interfaces)
- **Bootstrap** (expedites frontend development by providing a grid system and pre-designed UI components,)
- **Vuex**(Implemented for centralized state management, Vuex simplifies DOM manipulation)
- **Axios** (for seamless HTTP requests, ensures efficient communication with the backend)

DB Schema Design

1. **USER** (Table):
   Columns: user_name (text), password (text), first_name (text), last_name (text) ,phone_number (text), address (text) , user_type (text , default='N').
   Constraints: PRIMARY KEY("user_name")
   Additional info: user_type {'N': normal user , 'SM': store manager}
2. **CATEGORY** (Table):
   Columns: category_id(Integer), category_name(text).
   Constraints: PRIMARY KEY("category_id")

3. **PRODUCT** (Table):
   Columns: product_id(Integer), product_name(text), product_desc(text), product_img(text) ,product_price_per_unit (NUMERIC), product_stock(Integer) , product_exp_date(Date) , product_man_date(Date),category_id(Integer),product_discount(NUMERIC) ,veg_nveg(text),product_unit(text, default='unit').
   Constraints: FOREIGN KEY("category_id") REFERENCES "category"("category_id"),
   PRIMARY KEY("product_id" AUTOINCREMENT)
   Additional info: veg_nveg{'N': veg , 'NV': non-veg}
4. **BASKET_MASTER** (Table):
   Columns: user_name (text), product_id(Integer), qty(Integer),item_id (Integer)
   Constraints:PRIMARY KEY("item_id" AUTOINCREMENT),
   FOREIGN KEY("user_name") REFERENCES "user"("user_name"),
   FOREIGN KEY("product_id") REFERENCES "product"("product_id")
5. **ORDERS**(Table):
   Columns: item_id(Integer), user_name(text), product_id(Integer),qty (Integer),order_price(NUMERIC),order_id(Integer), address(Text), contact_no(Text),created_at(Date), status(text, default=P),total_price(NUMERIC)
   Constraints:FOREIGN KEY("product_id") REFERENCES "product"("product_id"),
   FOREIGN KEY("user_name") REFERENCES "user"("user_name")
   Additional info: status{'P': placed , 'D': delivered}
6. **ADMIN_REPORT_MASTER**(Table):
   Columns: report_id(text), filename(text), report_type(text),status(text)
   ConstraintsPRIMARY KEY("report_id")
   Additional info: status{'N': under process , 'Y': generated, 'E':error}

Architecture and Features

The project structure is meticulously organized, featuring a core 'application' folder housing 'models.py' for database models and 'database.py' for initialization. 'Api.py' for all the crud apis.'tasks.py' contains all the celery tasks/jobs such as order report, alerting user to login, etc. 'Masterdata.py' contains system level cache for all the products and categories.

FreshPick features separate login pages for regular and admin users, employing secure password hashing via the Bycrypt library during user registration. Backend requests undergo jwt token validation, while cached product data (at server startup) for user homepage , reduces database queries at login. Dynamic details like price and stock are fetched as needed from the database.User can search for products using the search bar with category selection and other advanced filters like price range, discount, veg, non-veg, etc. Unavailable products are marked as "Not available" to prevent adding to the basket.User can modify qty or remove items from their basket. Product availability is checked during actions like adding to the basket, saving the basket or placing orders. The admin dashboard includes Summary, Category and Product management, instantly updating the cache with admin changes for real-time web updates. It also has feature to generate product report asynchronously.

Video: https://drive.google.com/file/d/1_hpvOerOaeWbpuqhjq7A_gS1fz9W0i0t/view?usp=drive_link