Report for Household Services Application

Student

Harsh Sanjay Roniyar

An enthusiastic individual eager to absorb new skills and techniques that will pave the way for my academic and professional growth

Description

The Household Services Application project is a multi-user app which acts as a platform for providing comprehensive home servicing and solutions by connecting service professionals with customers. The app allows seamless booking, tracking, and management of services, with admin oversight for smooth operations.

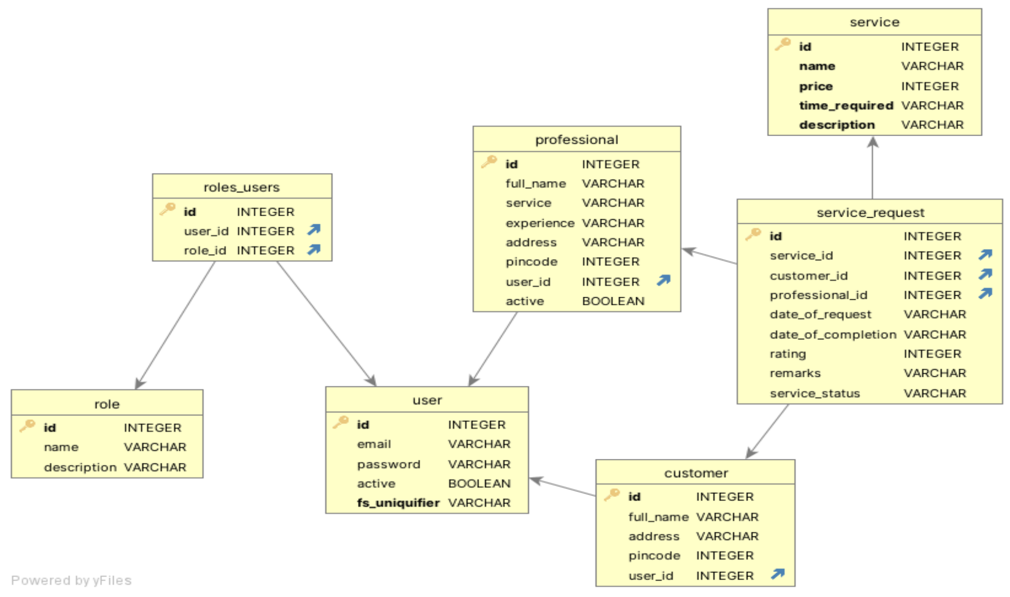
Project Approach

I started with creating the models for the database, and then set up the environment. For the backend, I used Flask to create API endpoints for managing services, customers, and professionals, with auth and Celery for scheduled tasks. The frontend, built with Vue.js and Bootstrap, which communicate with the backend via the Fetch API. MailHog was used to test the email functionality.

Frameworks and Libraries

- **Backend**:
  Flask | Flask-SQLAlchemy | Flask-RESTful | Flask-Security-Too | Flask-Caching | Redis | Celery | Werkzeug | Jinja2 | SMTP | MailHog | datetime
- **Frontend**:
  Vue.js | Bootstrap | Chart.js

DB Schema Design



API Resource Endpoints

1. **`/api/services`**:
   - **GET**: Retrieve all services.
   - **POST**: Create a new service.
2. **`/api/customers`**:
   - **GET**: Retrieve all customers.
   - **POST**: Create a new customer.
3. **`/api/professionals`**:
   - **GET**: Retrieve all professionals.
   - **POST**: Create a new professional.
4. **`/api/update/service/<int:id>`**:
   - **GET**: Retrieve a specific service by ID.
   - **POST**: Update a specific service by ID.
5. **`/api/request/service`**:
   - **GET**: Retrieve all service requests.
   - **POST**: Create a new service request.
6. **`/api/accept/service-request/<int:id>`**:
   - **GET**: Reject a service request by ID.
   - **POST**: Accept a service request by ID.
7. **`/api/service-request/customer`**:
   - **POST**: Retrieve service requests by customer.
8. **`/api/close/service-request/<int:id>`**:
   - **POST**: Close a service request by ID.
9. **`/api/user-details`**:
   - **GET**: Retrieve details of the current user..
10. **`/api/update-user-profile`**:
    - **POST**: Update the profile of the current user.

Features

Core Features	**Customer-Specific**	**Professional-Specific**	**Admin-Specific**
Login, Register, Password Auth	View/Search Services	Accept/Reject Service Requests	Approve/Reject Professionals
Role Segregation (RBAC)	Create/Edit/Close Service Requests	Service-Based Request Segregation	Block Customer/Professional
User Dashboard & Edit User Profile	Service Feedback/Rating	Scheduled Daily Reminders (E-Mail)	Create/Update/Delete Services
Caching and Expiry	Monthly Activity Reports (E-Mail)		ASync Service Detail Exports (E-Mail)