Author:

Shubham Sheshnarayan Atkal

Roll Number: 23f1002838

Email: 23f1002838@ds.study.iitm.ac.in

I am a tech enthusiast currently pursuing a BS degree standalone.

Description:

Household Services - V2

This project focuses on building an A2Z Household Services web application which can help professionals and customers to reach each other in a better way.

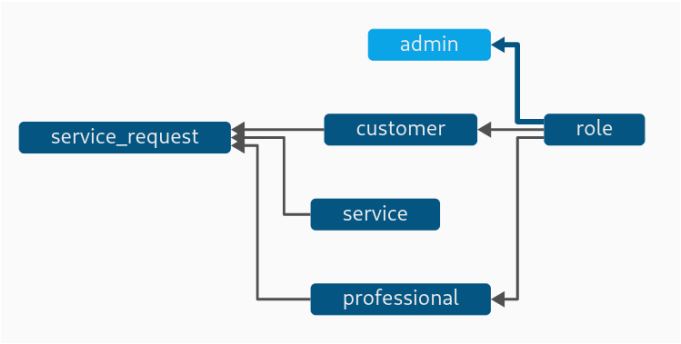
Technologies Used

Flask,SQLAlchemy,Redis (for caching),Celery (beat),MailHog local server (for testing mails),Vue js (frontend),JWT (for authentication),Flask restful (apis)

Purpose of These Technologies:

- **Flask**: A lightweight and flexible web framework used to build the backend of the application.
- **SQLAlchemy**: ORM for handling database operations efficiently and ensuring data integrity.
- **Redis**: Used for caching to optimize API response times and reduce database load.
- **Celery (beat)**: Manages background task scheduling, such as periodic email notifications and data processing.
- **MailHog**: A local email testing server used to simulate email sending during development.
- **Vue.js**: A frontend framework for creating a dynamic and responsive user interface.
- **JWT**: Used for authentication to ensure secure API access and session management.
- **Flask-RESTful**: A structured way to build RESTful APIs, making API development and maintenance easier

DB Schema Design:



API Design:

The API manages **service requests, professionals, and customers** in a service booking platform. It provides endpoints for booking services, updating statuses, retrieving professional details, and rating experiences.

Key API Elements:

- **Service Requests:** Create, update, close, fetch details, and rate requests.
- **Professionals:** List professionals, filter by service, block/unblock.
- **Customers:** Manage accounts, toggle customer status.

Implementation Details:

Built with **Flask-RESTful**, using **path parameters** (`request_id`, `professional_id`) for entity operations. Supports **CRUD** operations with structured **JSON responses** and proper HTTP status codes.

This API serves as the **backend for a service management system**, enabling efficient service bookings and professional/customer interactions.

Yaml file: [api.yaml]

Architecture and Features

The project follows an **MVC-like architecture** with:

- **Controllers:** Manage API endpoints for service requests, professionals, and customers.
- **Models:** Handle database interactions for storing service details, user data, and bookings.
- **Middleware:** Implements JWT authentication, request validation, and rate limiting.

Features Implemented:

User Authentication: Secure login/logout with JWT for protected API access.
Service Provider Management: View, manage, and block/unblock professionals.
Service Requests: Users can create, update, close, and rate service requests.
Customer Management: Toggle customer status and retrieve customer details.
Caching with Redis: Optimized API response times for frequently accessed data.

This API efficiently manages service bookings, professional listings, and customer interactions while ensuring security and performance optimization.

Video: 🎬 Demo-household-services-app-23f1002838 link