MAD 2 REPORT
HOUSEHOLD SERVICES

Author

- Name: Gayathri Srinivasan
- Roll no: 21f3002753
- Email: 21f3002753@ds.study.iitm.ac.in
- I am a final-year engineering student. I have also been pursuing BSc degree at IIT Madras for the past three years and am expected to complete the diploma level in this term. I focus on activities that interest me and always strive to give my best effort.

Description

This project provides service bookings for customers and professionals, offering features like user registration, service browsing, and request management. Additionally, the admin monitors the entire system and manages users, services, and service requests to ensure smooth operations.

Technologies used

- Frontend: Vue JS
- Framework: Flask - Python Framework, Vue JS
- Database: SQLite
- Packages: Bootstrap, JQuery, Datatables, ChartJS, Redis, Celery

For this project, I have used Flask, for building the backend, integrating it with SQLite for database management. For the frontend, I utilized Vue JS along with Bootstrap for responsive design. Additionally, JQuery, Datatables, and ChartJS were used for interactivity and data visualization. Redis was implemented for caching, while Redis and Celery were utilized for batch jobs.

DB Schema Design

I have attached the details of tables used for this project
https://docs.google.com/spreadsheets/d/1fYVaj4OYMwQvdzudWgEtv00-PPpFDsG5PCKIvHY2tg4/edit?gid=0#gid=0

This schema is designed to store the essential data for customers, professionals, services, and service requests, ensuring that each entity is properly related using foreign keys. The constraints like Primary Key, Not Null, and Default values are used to maintain data integrity and ensure that the system works efficiently and accurately.

API Design

In this project, I have created APIs for various elements such as user registration, login, service browsing, service request creation, and admin functionalities (like viewing and managing service requests). These APIs facilitate communication between the front-end and back-end of the application, allowing users and professionals to interact with the system efficiently.

- User registration and login API to register new users (customers and professionals) and authenticate login credentials.
- Admin API which allows for the acceptance/rejection of professionals, as well as searching and retrieving available services.
- Customer API enables customers to book services, retrieve and search for available services, and submit ratings.
- Professional API allows professionals to accept service requests made by customers and retrieve/search for available services.

I have also attached the YAML file for further reference
https://drive.google.com/file/d/1dUmjh0Z99OEbMZYi5G22F97oK_f8uUo3/view?usp=sharing

Architecture and Features

This project follows a Model-View-Controller (MVC) architecture. The *models* handle the database structure and operations (e.g., Customer, Professional, Service, and ServiceRequest tables), while the *views* are rendered using Jinja2 templates, allowing dynamic content for users and professionals based on their roles. The *controllers* (Flask routes) manage the logic for handling HTTP requests, user interactions, and data processing. The views are developed using Vue.js for the frontend, providing dynamic and interactive content tailored to users and professionals based on their roles. The controllers manage the logic for handling HTTP requests, user interactions, and data processing. Templates for base structures and layouts are integrated with Vue.js, ensuring consistent structure and design across the application.

Features Implemented

- Both customers and professionals can register and log in to their accounts.
- Customers and professionals can browse available services. Admins can manage the services and view requests.
- Customers can request services, and professionals can accept or reject these requests. Admins can monitor and manage the status of requests.
- The application has different views and access levels for customers, professionals, and admins, ensuring appropriate actions and information are available based on user roles.
- After service completion, customers can rate professionals and leave remarks on the services provided.
- Additionally, reports can also be generated for daily and monthly reminders.

Video

https://drive.google.com/file/d/1Xi7iHXtet4eFful3XrBKyklxLOFH-Ls4/view?usp=drive_link