MAD-II Project : ServiceHub

Student Details

Name:- Aditya Siddharth Jyoti
Email :- 22f3001792@ds.study.iitm.ac.in
Roll No :- 22f3001792
Submitted in :- September2024 term

Project Objective

The project is centered around creating a household services platform designed to seamlessly connect customers with skilled professionals for various daily tasks, including cleaning, plumbing, pest control, and more. Finding reliable help for short-term household duties is often a challenging and time-consuming process. People often struggle with searching for trustworthy individuals and negotiating terms, which can lead to frustration and delays in addressing their needs.
Our platform aims to resolve this issue by providing a one-stop solution where customers can explore a wide range of services and hire professionals who are thoroughly verified and registered. The platform ensures not only convenience but also peace of mind by offering access to dependable service providers.

Technology Used

Following are the technology used for building this web application :-

1. Flask framework is used to build the backend of the application , which includes APIs , Security , App Integrations and the Views.
2. Vue.js is the primary frontend framework for the application along with supported css like bootstrap and animate.css .
3. SQLite server is used for database management and ETL.
4. Caching and background job management is done by Redis.
5. Task scheduling and batch job management is done by celery.

ER Diagram (Fig 1.2)

The database schema consists of the following tables :-

- **user**: Contains user details and login credentials.
- **role**: Defines user roles such as admin, customer, and professional.
- **roles_users**: Links users to their respective roles.
- **service**: Lists available services (e.g., plumbing, AC servicing).
- **customer**: Holds customer profile information.
- **professional**: Stores professional profile information.
- **service_request**: Tracks service requests, linking customers with professionals.

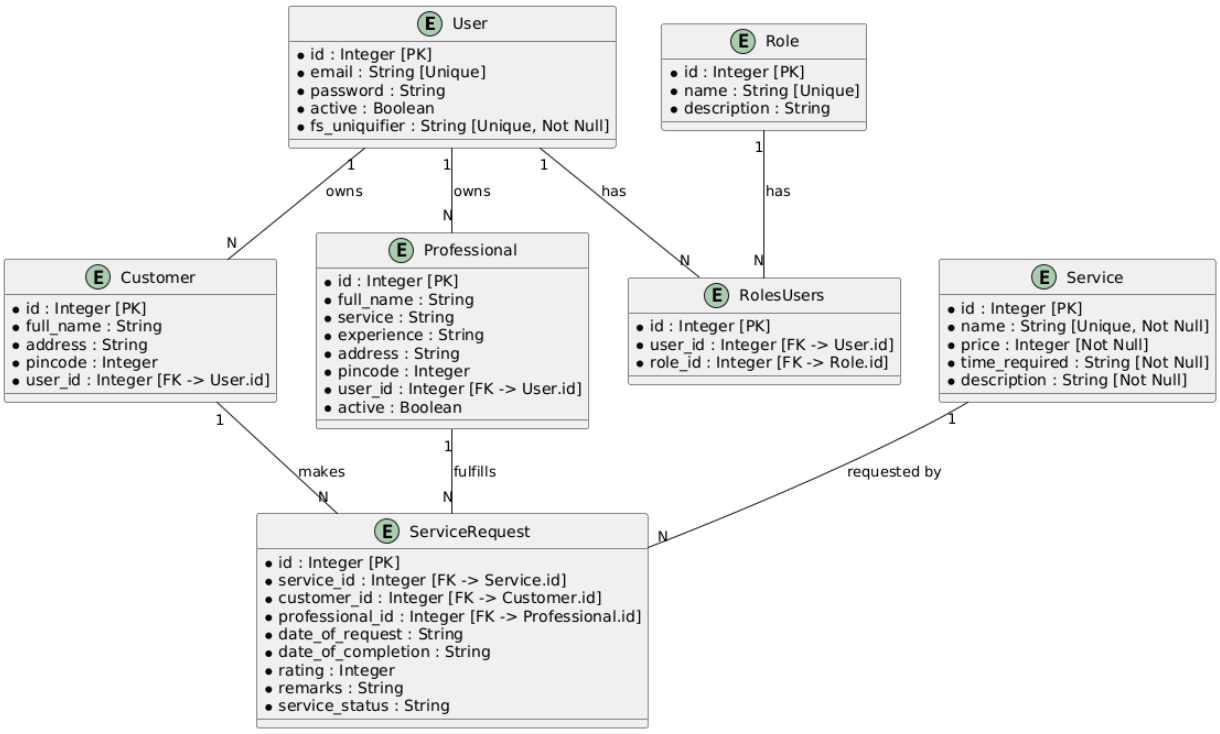

Fig 1.2

Resources (APIs)

All the APIs are built using `flask_restful` framework and it uses the prefix ‘api’.
Following are the functionalities of each resources(APIs) :-

/services
GET: Retrieve all services.
POST: Add a new service (admin only).

/update/service/<int:id>
GET: Retrieve service details by ID.
POST: Update service details by ID (admin only).

/customers
GET: Get all customers (admin only).
POST: Add a new customer.

/professionals
GET: Get all professionals (admin only).
POST: Add a new professional.

/request/service
GET: Get all service requests with service details.
POST: Create a new service request.

/accept/service-request/<int:id>
GET: Reject a service request.
POST: Assign a professional to a service request.

/service-request/customer
POST: Get all service requests for a specific customer.

/close/service-request/<int:id>
POST: Close a service request with rating and remarks.

Technical Features Included

- Role Based Access Control (RBAC) with the help of JWT token for Customers , Professionals and Admin.
- Secured routing to the endpoints with the help of RBAC .
- Password hashing for data privacy concerns.
- Caching of the frequently used data on the client side using redis.
- Background task and batch jobs scheduling using celery.
- InApp database control using SQLAlchemy.
- Dynamic frontend navigation and optimization using JavaScript frameworks like VueJS

Presentation Video Link

Click here to navigate to the presentation

Or use the following link :-
https://drive.google.com/file/d/127DHk-ZfPov5OtdJRTIx8PYKCG_Y7QQZ/view?usp=sharing