Household Services App - MAD 2 Jan2025 Project Report

Name: Hella Musammil

Roll number: 22f3002179

1. **Introduction**:

   This project involves the development of a web-based multi-user platform designed to streamline home servicing solutions. The platform includes three types of users: Admin, Professionals, and Customers. Each user has distinct roles and permissions, ensuring smooth interactions and efficient service management.

   The **Admin** has root access and oversees the platform by managing services, professionals, and customers. **Professionals** can accept service requests, while **Customers** can create, edit, or delete booking requests, mark them as completed, and provide ratings.

2. **Project Approach and Architecture and Features:** The development process was structured in multiple phases:

   - **Phase 1: Understanding Requirements**

     Studied the wireframe and analyzed relationships between user roles. Defined key functionalities, including service management, authentication, and request handling.

   - **Phase 2: Backend Development**

     - Created the development environment in Visual Studio, organizing backend and frontend in separate folders. Built data models using SQLite to manage relationships among users, services, and bookings. Developed secure password handling using werkzeug.security to hash and verify user passwords. Created RESTful APIs in Flask for various functionalities, including authentication, service creation, and booking management. Implemented caching using Redis to optimize API performance, setting cache expiry for professional and customer APIs. Integrated Celery for scheduled jobs: **Daily Reminders (**Notifications for service professionals regarding pending service requests and uncompleted requests) and **Monthly Activity Reports (**Summary of customer activity emailed as an HTML report) and for triggered asynchronous jobs: **CSV Export (**Batch job that can triggered by admin to send professionals their completed requests).Tested API endpoints using Thunder Client. Used MailHog to test email notifications.

   - **Phase 3: Frontend Development**

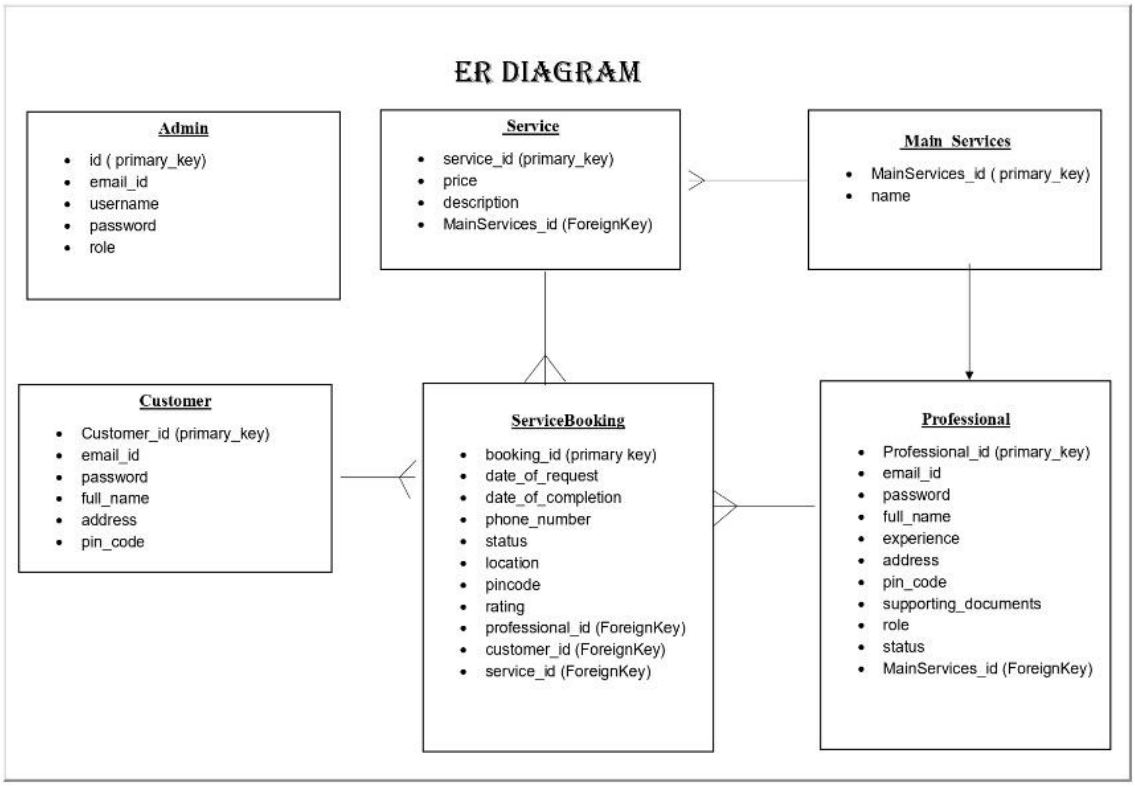
     - Used Vue.js for UI development and integrated Bootstrap for responsive design. Developed user dashboards tailored to each role (Admin, Professional, Customer). Ensured intuitive navigation and role-based access.

3. **Technologies Used**

   - **Backend:** Flask, SQLite, Redis, Celery, SQLAlchemy, Flask-Caching
   - **Task Scheduling:** Celery (with crontab for scheduling tasks)
   - **Emailing:** Python's email.mime.text, email.mime.base, email.mime.multipart, and encoders for creating and sending complex emails with attachments
   - **Frontend:** Vue.js, Bootstrap
   - **Security:** werkzeug.security for password hashing

- **Testing:** Thunder Client, MailHog

4. **DB Schema Design**



Each user type is assigned its own separate table to accommodate their unique requirements. Additionally, I have created a main services table, which is then followed by related sub-services tables.

5. **API Design:** Below are the APIs used, along with their respective functions:

   - **adminserviceAPI** – Includes GET, POST, PUT, and DELETE methods to fetch, create, edit, and delete services.
   - **ProfessionalSignupAPI** – Includes GET to create a new professional.
   - **CustomerSignupAPI** – Includes GET to create a new customer.
   - **LoginAPI** – Includes POST to enable user login and PATCH to block or activate professionals.
   - **ExportDataAPI** – Includes GET to trigger the Celery function for exporting data.
   - **bookingAPI** – Includes GET, POST, PUT, and DELETE methods to fetch, create, edit, and delete bookings.
   - **MainServiceAPI** – Includes GET, POST, and DELETE methods to fetch, create, and delete main services.
   - **ProfessionalAPI** – Includes GET, PUT, and DELETE methods to fetch, edit, and delete professional profiles.
   - **CustomerAPI** – Includes GET, PUT, and DELETE methods to fetch, edit, and delete customer profiles.

6. **Video:**
   https://drive.google.com/file/d/1IEuJCIRoDiwwoICt7fY6J15aChJ0DNv0/view?usp=sharing