Household Services Application : ABODE-AIDE

Developed By : Sampriti Raha

Roll no. : DS22F1000064

Student Email Id : @ ***SAMPRITI RAHA***

Drive Link for Presentation Video: ***DS22F1000064 MAD1***

Problem Statement:

The objective of the project is to build a platform to connect Customers who want to avail household services which are being provided by Professionals who have registered in the website.

Technologies used:

This application integrates Flask, Flask-SQLAlchemy, SQLite, Python, and Jinja2 to deliver a robust and portable solution. Flask serves as the backbone, enabling dynamic web development, while Flask-SQLAlchemy ensures seamless interaction with the SQLite database, chosen for its simplicity and reliability. Python powers the application with its scalability and readability, while Jinja2 provides dynamic templating for an elegant user interface. Together, these technologies create a lightweight, efficient, and easily deployable architecture tailored for both functionality and user experience.

Architecture and Features:

Interactions with the database are handled using models.py and the functions are given in the backend folder via controllers.py. The HTML and CSS files reside in templates folder and static folder respectively, whereas the database is in the instance folder & all the python codes inside the backend folder with app.py which is the main file lies outside of every folder.

DB Schema Design:

The application uses 3 tables in an SQLite database named site.sqlite3 found in the instance folder for keeping track of and managing the database. The ER diagram is given below:

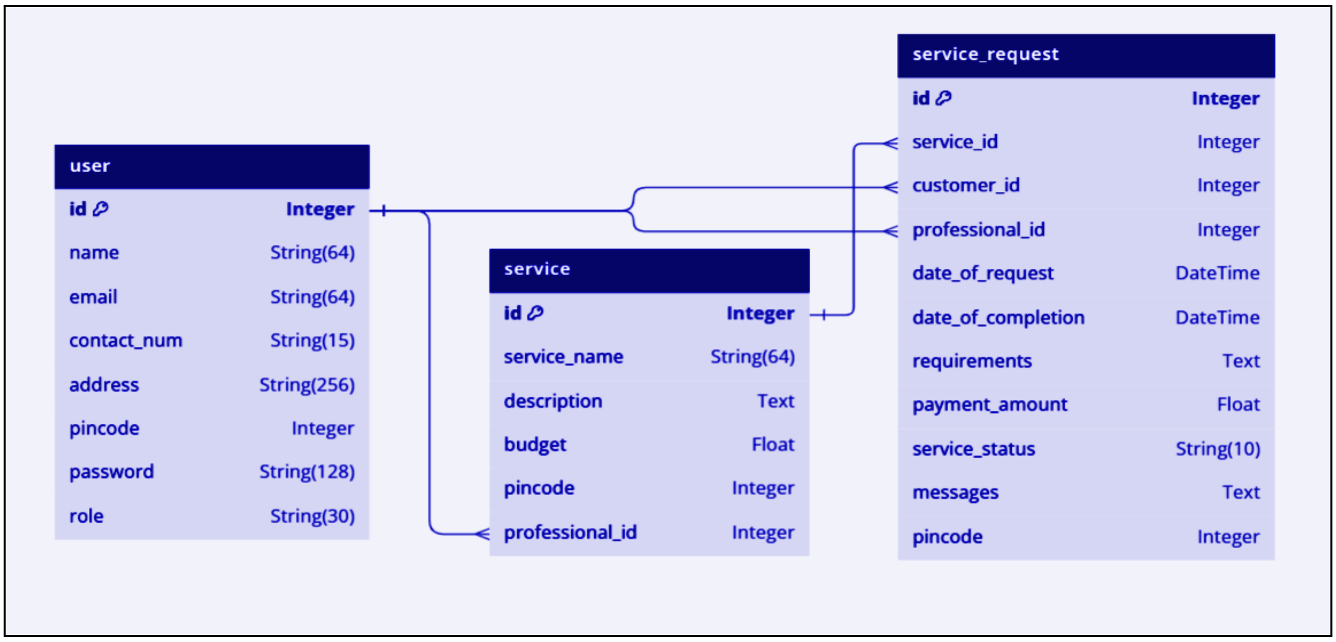

Fig:1 – ER Diagram of the model

The following Recommended and Optional functionalities (as per the [Project Grading Doc]) have also been implemented in the project along with the Core Functionalities.

Recommended Functionalities

- Implementing frontend validation on all the form fields using HTML5 form validation.
- Implementing backend validation within the controllers of your app.

Optional Functionalities

- Provide styling and aesthetics to your application by creating a beautiful and responsive frontend using simple CSS or Bootstrap
- Incorporate a proper login system to prevent unauthorized access to the app using flask extensions like flask_login, flask_security etc.
- Implement a dummy payment portal (just a view taking payment details from customers for an ad request)