

Author

MONISH SANTOSH MISHRA

23f2005276

23f2005276@ds.study.iitm.ac.in

IITM Diploma level student passionate about Machine Learning and Programming, have built some react projects till now which helped me design this project in a better way, it was really new for me to do the state management of the project manually since I was used to use certain hooks and features in react which do the state management manually, really good experience with building this project and looking forward to code the 2nd version of this project if possible

Description

So A2Z_household_services deal with connecting professionals offering household services with customer seeking those services, where admin of the web app makes sure the transactions and exchange of resources from both sides are fair and equal

Technologies used

- The core flask package
- flask-sqlalchemy (ORM of our project to interact with database)
- flask-wtf (a fork of wt_forms package designed specifically for core flask package to handle forms for authentication and handling various other input data from the user's side)
- email-validator (to validate email fields when the user submits the forms having fields where email format data is expected)
- flask-login (to apply the backend logic of authenticating a user and managing sessions using the UserMixin class of flask-login package)
- flask-bcrypt (to hash password before storing them in the database to make sure that attackers don't get the real passwords during a data breach)
- python-dotenv (to handle the env variables like database keys and secret keys used for form validation to prevent csrf attacks)

DB Schema Design

- There are 3 main tables User, ServiceTypes, Service_requests_central and there are other small tables supporting some other features around these tables
- Although I could've made changes in the main tables of the database itself but didn't know how to do changes in the attributes like packages like flask-migrate were mandatory to do the changes so like I've planned to integrate these changes in the future MAD2 project

- These 3 tables have all the attributes required to describe all sorts of data related to their domain
- For example User Table has attributes like id, name, category_of_user (Admin, Customer, Professional), service_offered (different values in case of Professional but in case of Admin and Customer the values are Admin and Professional only), pincode etc.
- ServiceTypes table has attributes like base_price, service_name, service_Description
- Service_requests_central has many attributes like customer_id, professional_id, service_id, customer_pincode, professional_pincode, negotiated_price and purpose of these attributes is self-understanding
- Although these relational designs are not the in context of the normalization theory of DBMS, but these practices are required when the user base of the project increases
- But still it is my target to implement these changes in the MAD-2 project


API Design

- I haven't built any apis for the project since I want to keep it for the MAD-2 project, did everything in the project on the basis manual coding itself

Architecture and Features

- The project is organized in a python package format where the __init__.py file is the main file in the code folder
- And there is a templates folder for all the frontEnd html files serving the web application
- All the images used for the web app and the files given by the users are also stored in the assets folder itself
- For visual features you can refer to the video that I made to explain the project below

Video

 23f2005276.mp4