# Sarthak Jain

Sarthak Jain

23f200839

23f3000839@ds.study.iitm.ac.in

I am an aspiring AI Engineer from Jabalpur, MP. I am currently pursuing a Diploma level of IITM BS Degree. I have Skills in Web Development and AI.

## Description

A comprehensive web platform that connects customers with verified household service professionals. The system features role-based access (Admin/Professional/Customer), service management, a booking system, and a professional verification workflow. Built with Flask, it provides a secure and efficient way to manage household service requests.

## Technologies Used

**Core Framework & Extensions**

- Flask - Lightweight web framework

- Flask-SQLAlchemy - ORM

- Flask-Login - Authentication

- Flask-WTF - Forms

- Flask-Bootstrap - Frontend

**Additional Technologies**

- SQLite - Database

- Purpose: Lightweight, serverless, perfect for development

- Matplotlib - Visualization

- Bootstrap 4 - Frontend Framework

## DB Schema Design

https://drive.google.com/file/d/1qyzkOVYNYWP7w7-sP4moVkZkQ3es01tx/view?usp=sharing

## API Design

https://drive.google.com/file/d/1Ckp8phBwN_6O89RYnVxJ_wNdw6mcWwso/view?usp=sharing

# Architecture & Features

app/
├── views/ # Route handlers by role
├── models/ # Database models
├── templates/ # Jinja2 templates
├── forms/ # WTForms classes
└── static/ # CSS, JS, images

### Key Features

**1. Authentication & Authorization**
- Role-based access control
- Professional verification system

**2. Service Management**
- CRUD operations for services
- Service request lifecycle

**3. Professional Dashboard**
- Request management - Service tracking

**4. Customer Features**
- Service booking
- Review system

**5. Analytics & Reporting**
- Service statistics
- Performance metrics

**1.User Authentication**, **Endpoints**:
POST /login: Authenticates users and returns a session token.
POST /register: Allows new users to register with role-based access.
**Implementation**: Utilizes Flask-Login for session management.

**2.Service Management**
**Endpoints**:
GET /services: Retrieves a list of all available services.
POST /services: Allows admins to create new services.
PUT /services/{id}: Enables admins to update existing services.
DELETE /services/{id}: Allows admins to delete services.
**Implementation**: Uses SQLAlchemy for ORM to interact with the database and manage service records.

**3.Service Requests**, **Endpoints**:POST /service_requests: Allows customers to create service requests.
GET /service_requests: Retrieves all service requests (admin only).
PUT /service_requests/{id}: Updates the status of a service request (professional only).
**Implementation**: Service requests are managed through SQLAlchemy, ensuring proper relationships between users and services.

**4.Reviews**, **Endpoints**: POST /reviews: Allows customers to submit reviews for completed services.
**Implementation**: Reviews are linked to service requests and users, ensuring that feedback is associated with the correct service.

# Video

https://drive.google.com/file/d/1WJMHK7IZ6EZxDaYpYYWU6Yc-2PoiUxHm/view?usp=sharing