The Household Service Application is a digital platform that streamlines booking and managing household services using Flask, Vue.js, SQLite, and Redis. It features user authentication, service requests, real-time tracking, and background task processing with Celery for efficient operations.

PROJECT: HOUSEHOLD SERVICES

Modern Application Development - II

Yajat Kandregula (23f1002088)
23f1002088@ds.study.iitm.ac.in

Problem Statement

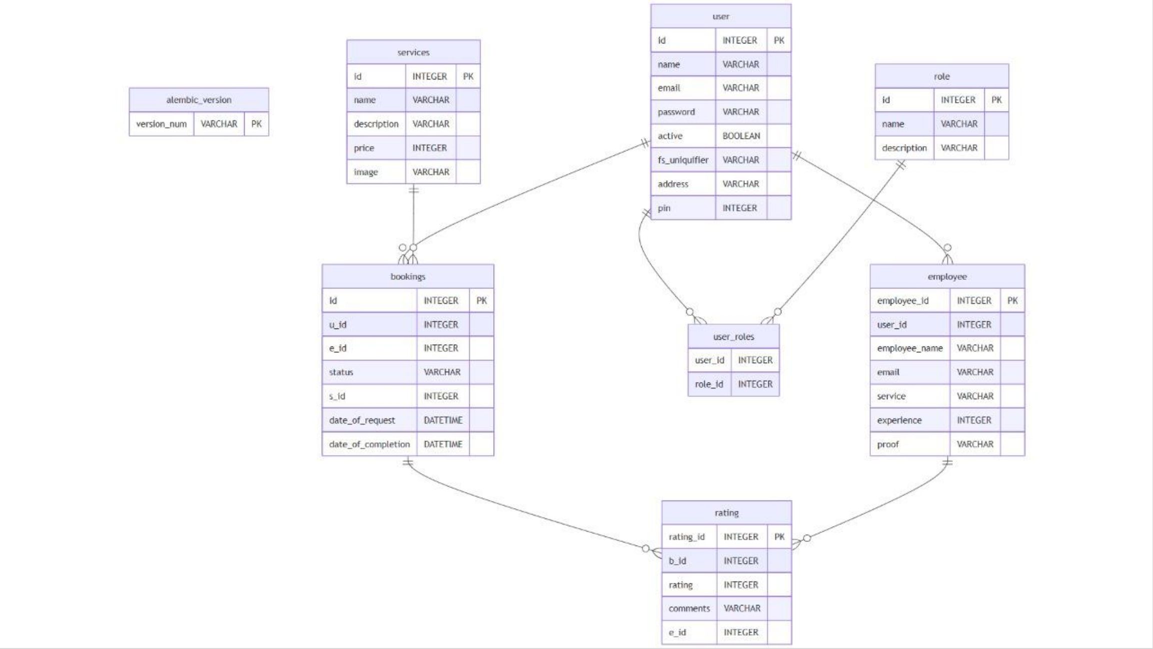
Managing household service requests manually can be time-consuming and inefficient. Customers face challenges in booking reliable service providers, and service providers struggle to manage appointments effectively. This application aims to bridge this gap by providing a seamless, digital solution for booking and managing household services.

Technologies used

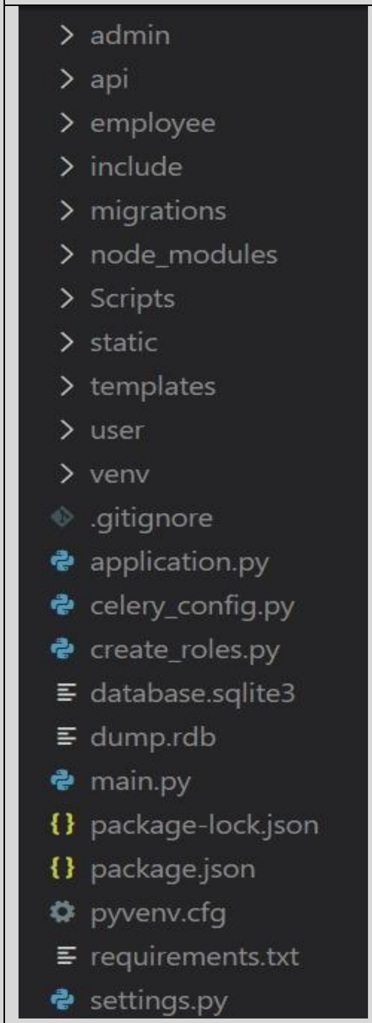
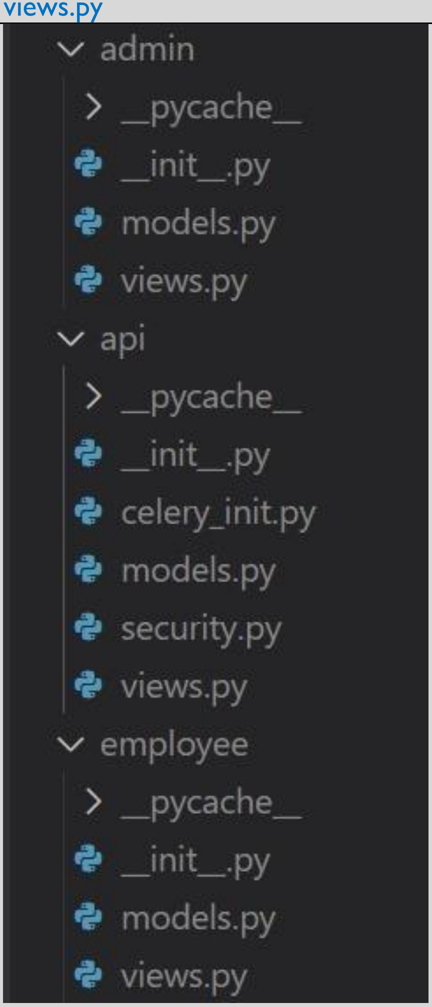
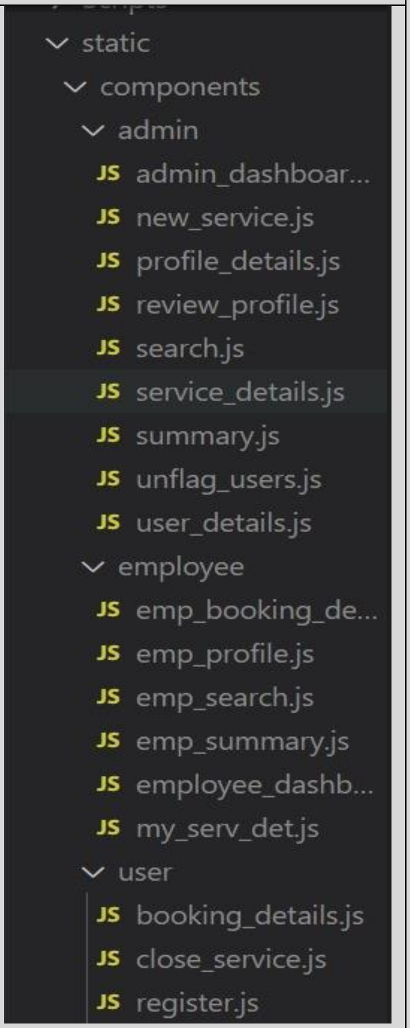
S No	Technologies used for	Technologies used
1	Backend	Flask (for API development)
2	Frontend	Vue.js (for dynamic and responsive UI)
3	Database	SQLite (for persistent data storage)
4	Templating	Jinja2 (only if required, not for UI)
5	Styling	Bootstrap (for responsive and mobile-friendly design)
6	Caching	Redis (for improving API responsive times)
7	Task Scheduling	Redis and Celery (for handeling batch jobs)

DB Schema Design

SQLLITE3 is the database.
Tables: bookings, employee, rating, role, services, user, user_roles
Attached ER diagram for database model.



Architecture and Features

Application root file is Main.py Create application including blueprints are in application.py. Standard Static folder is used.	Application is divided into 3 modules/applications. Book, Intro and librarian. Used blueprints to register all 3 applications to main application. Book: Book model is defined in model.py, forms are defined in forms.py and all controllers are in views.py librarian: librarian model is defined in model.py, forms are defined in forms.py and all controllers are in views.py intro: intro model is defined in model.py, forms are defined in forms.py and all controllers are in views.py	All HTML templates are centralized under a single base template, with individual module-specific UI components built as Vue.js components stored in respective static folders: admin, user, and employees. The base.html file is used for rendering the core structure, while Bootstrap is utilized to enhance the application's look and feel.
 > admin > api > employee > include > migrations > node_modules > Scripts > static > templates > user > venv .gitignore application.py celery_config.py create_roles.py database.sqlite3 dump.rdb main.py package-lock.json package.json pyvenv.cfg requirements.txt settings.py	 ∨ admin > __pycache__ __init__.py models.py views.py ∨ api > __pycache__ __init__.py celery_init.py models.py security.py views.py ∨ employee > __pycache__ __init__.py models.py views.py	 ∨ static ∨ components ∨ admin JS admin_dashboar... JS new_service.js JS profile_details.js JS review_profile.js JS search.js JS service_details.js JS summary.js JS unflag_users.js JS user_details.js ∨ employee JS emp_booking_de... JS emp_profile.js JS emp_search.js JS emp_summary.js JS employee_dashb... JS my_serv_det.js ∨ user JS booking_details.js JS close_service.js JS register.js

The E Library application has the following features

- Registration
- Admin Dashboard
- Flagging/unflagging users
- Requesting/accepting services
- Extracting work of users
- Sign in
- Employee Dashboard
- Summaries
- Updating/Closing Services
- User Dashboard
- Search Functionality
- Creating/Modifying Services
- Rating the service

Video

https://drive.google.com/file/d/1YdoSkqMwEF1A56nRg-SN8ZVkzMAhewXS/view?usp=sharing