Household Services Application V2

Project Report

Student Details

- Name: Vidhan Mertiya
- Roll No: 22f1001289
- Course: Modern Application Development II
- Project Title: Household Services Application V2

Project Overview

This project implements a multi-user platform that serves as a bridge between customers seeking household services and professionals providing these services. Following a role-based access control model with three distinct roles: Admin, Service Professional, and Customer.

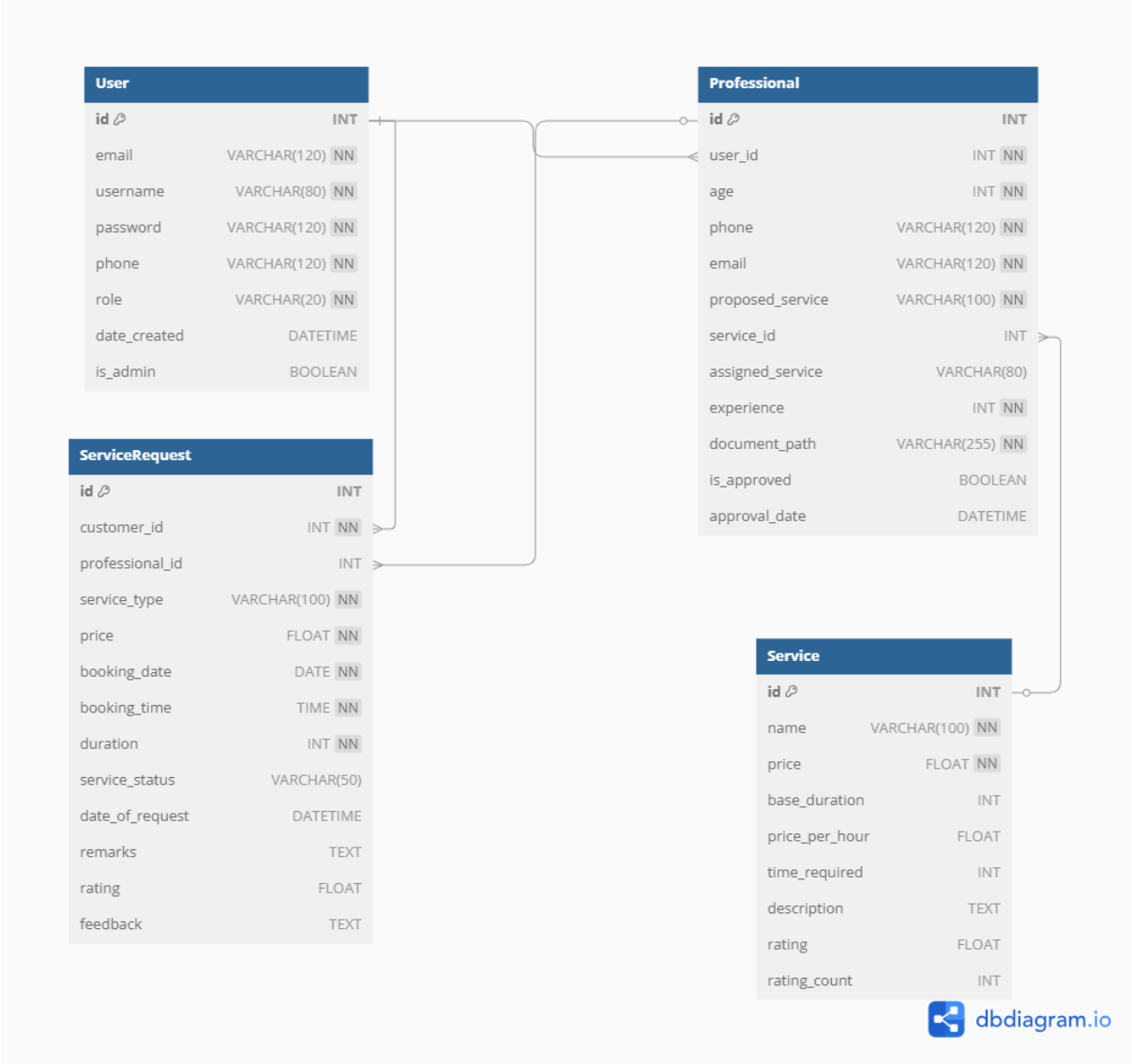
Problem Statement Approach

The application was built following the microservices architecture, separating concerns between user authentication, service management, request handling, and background processing. The focus was on creating a seamless experience for all stakeholders while ensuring security and performance.

Technical Stack

- Backend: Flask, SQLAlchemy ORM
- Frontend: Vue.js, Vuex for state management, Bootstrap for styling
- Database: SQLite
- Caching: Redis for performance optimization
- Background Jobs: Celery for scheduled tasks and asynchronous processing
- Authentication: JWT for secure role-based access

Database ER Diagram



The database consists of the following main entities with relationships:

- User: Core user data (used by all roles)
- Customer: Extends User for customer-specific data
- Professional: Extends User with service specialization
- Service: Available service types
- ServiceRequest: Links customers, professionals and services
- Review: Customer feedback for completed services

API Endpoints

Authentication

- POST /api/auth/register - User registration
- POST /api/auth/login - User authentication

Customer Endpoints

- GET /api/services - List available services
- POST /api/service-requests - Create service request
- GET /api/customer/ongoing-services - View ongoing services
- PUT /api/service-requests/{id}/close - Close service request

Professional Endpoints

- GET /api/professional/dashboard - View service requests dashboard
- PUT /api/service-requests/{id}/accept - Accept service request
- PUT /api/service-requests/{id}/reject - Reject service request

Admin Endpoints

- GET /api/admin/users - Manage users
- POST /api/admin/services - Create services
- PUT /api/admin/professionals/{id}/approve - Approve professionals
- POST /api/admin/export-csv - Trigger CSV export

Key Implementations

Background Jobs

1. Daily Reminders:
    - Scheduled job checking for pending service requests
    - Notification via email to professionals with pending requests
2. Monthly Activity Reports:
    - HTML report generation on the first of each month
    - Email delivery to customers summarizing their service history
3. CSV Export:
    - User-triggered async job for admin
    - Export of closed service requests with detailed information

Performance Optimization

- Redis caching for frequently accessed data
- Cache invalidation strategies for data consistency
- API performance tuning with proper indexing

Challenges and Solutions

- Implemented JWT token-based authentication to secure API endpoints
- Designed efficient background job processing with Celery
- Created responsive UI that works well across different devices

Project Demo

Video presentation link

Conclusion

This application successfully addresses the needs of a household services platform with a robust architecture that ensures scalability, security, and performance while offering a seamless experience for all users.