- # Project Report: Scarlett - Household Services Web Application

Student Details

- Name: Dabgar Milav Jayeshkuamr
- Roll Number: 21f1005510
- Course: Modern Application Development 2 (MAD2)

Project Overview

Scarlett is a comprehensive web application designed to connect household service professionals with customers. The platform facilitates seamless service booking, professional onboarding, and service management through a modern, responsive interface.

Technical Implementation

Backend Structure (/backend)

1. **Core Application (/app)**
   - `__init__.py`: Application factory pattern implementation
   - `models.py`: SQLAlchemy database models
   - `schemas.py`: Marshmallow serialization schemas
   - `extensions.py`: Flask extensions configuration
   - `config.py`: Environment-specific configurations
2. **API Modules (/app/api)**
   - `/auth`: Authentication and authorization endpoints
   - `/customers`: Customer-specific operations
   - `/professionals`: Professional service management
   - `/services`: Service listing and management
   - `/admin`: Administrative controls and monitoring
   - `/search`: Search functionality implementation
   - `/statistics`: Analytics and reporting
3. **Utility Modules (/app/utils)**
   - `auth.py`: Authentication helpers
   - `cache.py`: Redis caching implementation
   - `search.py`: Search functionality
   - `validation.py`: Input validation
   - `errors.py`: Error handling
   - `response.py`: Response formatting

4. **Background Processing**
    - `celery_app.py`: Celery configuration
    - `jobs.py`: Background task definitions
    - `email.py`: Email service implementation

Frontend Structure (/frontend)

1. **Core Components (/src)**
    - `App.vue`: Root component
    - `main.js`: Application entry point
    - `router/`: Route definitions
    - `store/`: Vuex state management
2. **View Components (/src/views)**
    - `/auth`: Login and registration
    - `/customer`: Customer dashboard and services
    - `/professional`: Professional workspace
    - `/admin`: Administrative interface
3. **Reusable Components (/src/components)**
    - `/layout`: Common layout components
    - `/auth`: Authentication forms
    - `/profile`: User profile management
    - `/admin`: Administrative components

Technologies Used

Backend Stack

- Flask Framework
- SQLAlchemy + PostgreSQL
- Redis + Celery
- JWT Authentication
- Flask Extensions (Mail, CORS, etc.)

Frontend Stack

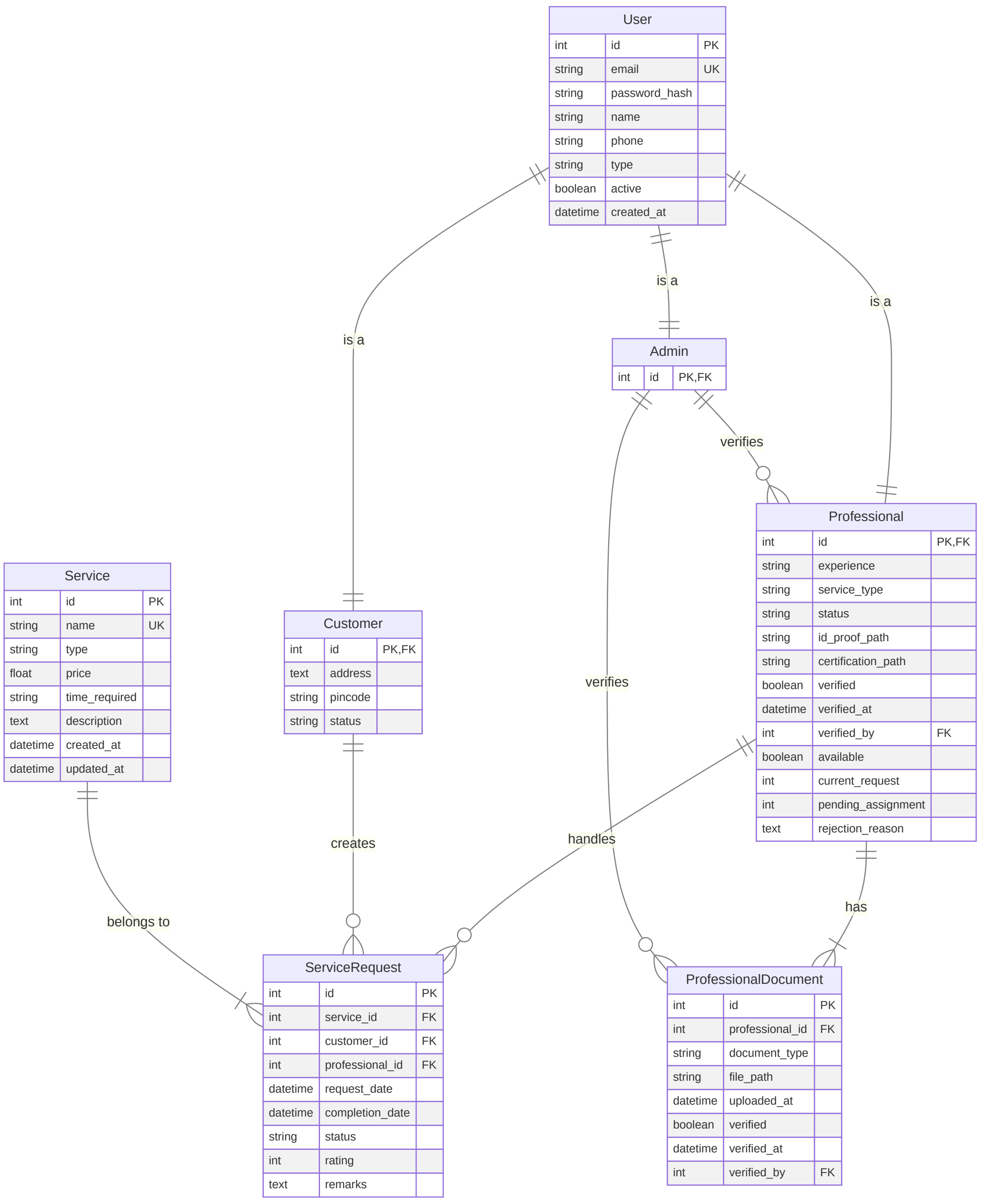
- Vue.js 3
- Vuex + Vue Router
- Bootstrap 5
- Axios
- Chart.js

API Resource Endpoints

The API follows a RESTful architecture with these main endpoint groups:

- `/api/auth/*`: Authentication and authorization
- `/api/customers/*`: Customer operations
- `/api/professionals/*`: Professional service management
- `/api/services/*`: Service-related operations
- `/api/admin/*`: Administrative functions
- `/api/search`: Search functionality
- `/api/statistics`: Analytics and reporting

ER Diagram



User		
int	id	PK
string	email	UK
string	password_hash	
string	name	
string	phone	
string	type	
boolean	active	
datetime	created_at	

Admin		
int	id	PK,FK

Professional		
int	id	PK,FK
string	experience	
string	service_type	
string	status	
string	id_proof_path	
string	certification_path	
boolean	verified	
datetime	verified_at	
int	verified_by	FK
boolean	available	
int	current_request	
int	pending_assignment	
text	rejection_reason	

Service		
int	id	PK
string	name	UK
string	type	
float	price	
string	time_required	
text	description	
datetime	created_at	
datetime	updated_at	

Customer		
int	id	PK,FK
text	address	
string	pincode	
string	status	

ServiceRequest		
int	id	PK
int	service_id	FK
int	customer_id	FK
int	professional_id	FK
datetime	request_date	
datetime	completion_date	
string	status	
int	rating	
text	remarks	

ProfessionalDocument		
int	id	PK
int	professional_id	FK
string	document_type	
string	file_path	
datetime	uploaded_at	
boolean	verified	
datetime	verified_at	
int	verified_by	FK

Presentation Video Link:- https://drive.google.com/drive/folders/1xA1VxGCQsIf0_7UJWy7vE-zj7yGiYqbY?usp=sharing