MAD-II Project Report

Author

- **Name:** Anuj Gupta
- **Roll Number:** 21f3001598
- **Student email:** 21f3001598@ds.study.iitm.ac.in

Project Description

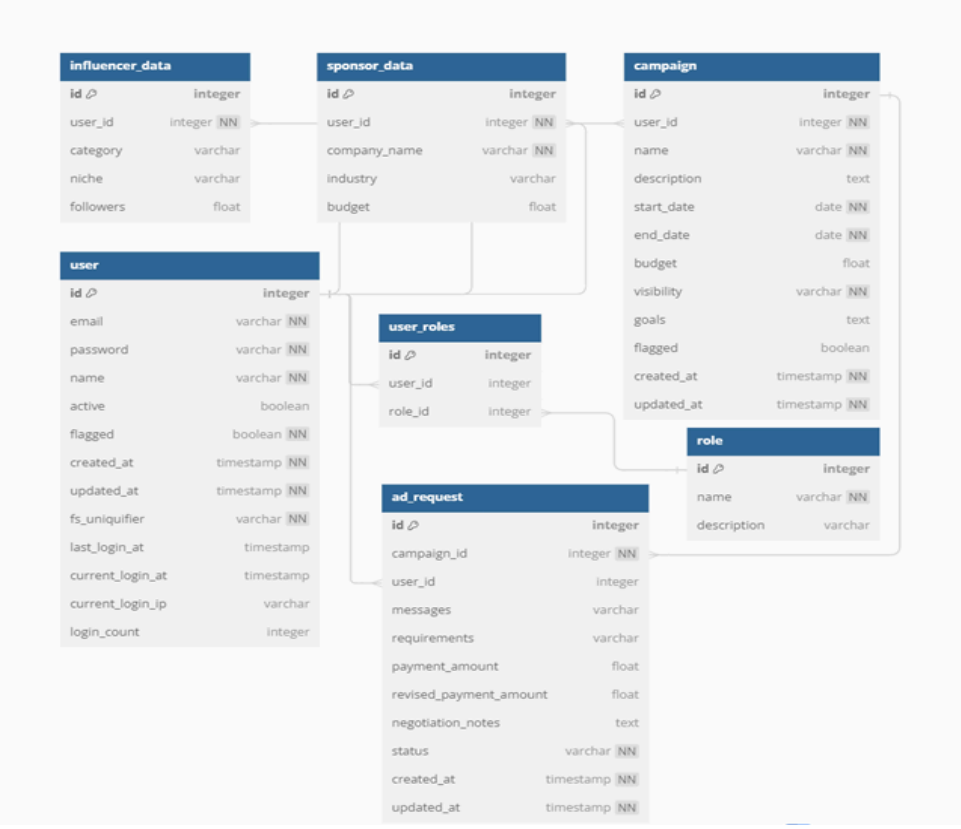
The **Influencer Engagement & Sponsorship Coordination Platform** is a web-based solution connecting sponsors with influencers for advertising collaborations. Built with SQLite, Flask, VueJS, Redis and Celery, the platform supports three user roles: Admin, Sponsor and Influencer. Admins manage users and campaigns, sponsors create and track campaigns and influencers browse and accept ad requests. The platform also includes daily reminders, monthly activity reports and RBAC ensuring efficient and secure interactions between all users. The platform emphasizes performance, responsive design and user-friendly interfaces.

Technologies used

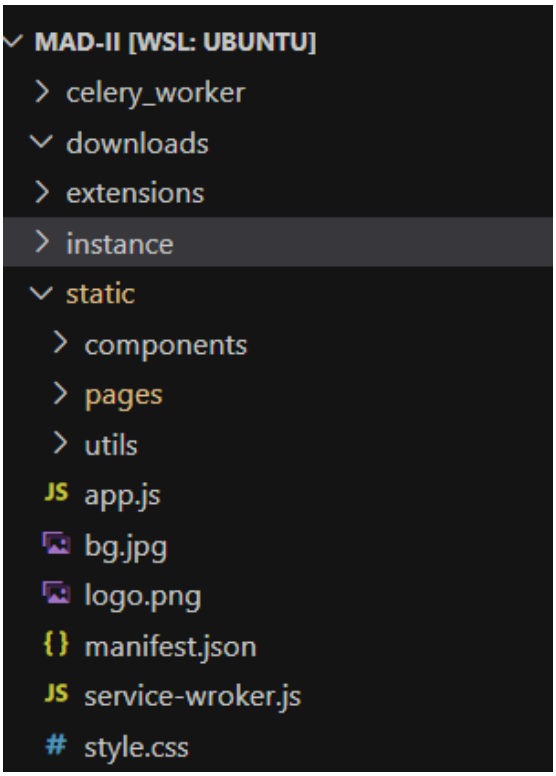
- **Python** is the core programming language used in the project.
- **Frameworks and Libraries**:
    - **Flask** is the main web framework used for developing the API.
    - **VueJS** is employed for building the user interface.
    - **Bootstrap** is used for styling and responsive design.
- **Database:**
    - **SQLite** is the database used for storing and managing application data.
    - **Flask-SQLAlchemy** is utilized as the ORM for interacting with the SQLite database.
- **Caching and Asynchronous Tasks:**
    - **Redis** is used for caching to enhance performance.
    - **Celery** is implemented for handling asynchronous tasks and scheduled jobs.

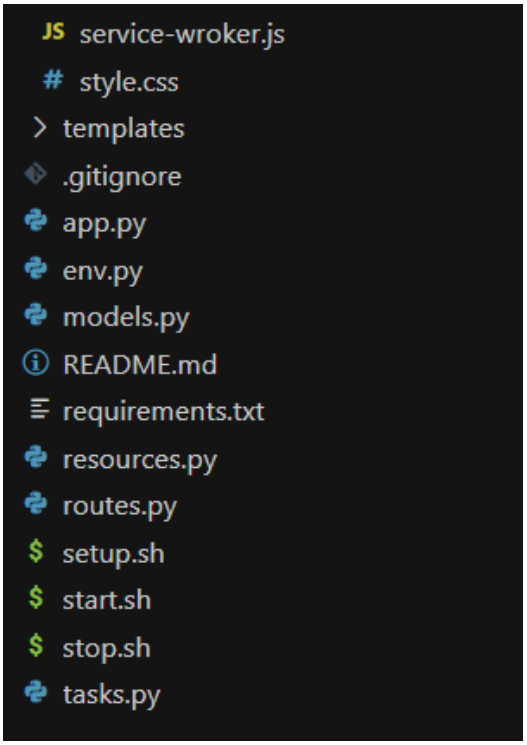
DB Schema Design

- Given below is the detailed description of the DB schema along with the ERD



Web App Architecture





Project Structure:

1. **Main Application Files:**
   - **app.py:** The core of the application, handling the initialization of the Flask app.
   - **env.py:** Contains environment variables.
   - **models.py:** Defines the SQLAlchemy models used for database interactions.
   - **resources.py:** Implements the RESTful API resources.
   - **routes.py:** Manages the routing of the application.
   - **tasks.py:** Defines Celery tasks for handling asynchronous jobs.

2. **Static Folder:**
   - **components/:** Houses VueJS components used throughout the application.
   - **pages/:** Contains VueJS pages for different views within the application.
   - **utils/:** Includes utility JavaScript files(router.js & store.js) used by VueJS.
   - **manifest.json:** Configuration file for the Progressive Web App (PWA).
   - **service-worker.js:** Service worker file to enable offline functionality.
   - **style.css:** Custom CSS for styling the application.
3. Templates folder contains HTML templates used by the Flask application.
4. Instance folder contains database file db.sqlite3
5. Celery Worker folder is a dedicated folder for managing & configuring workers.
6. **Shell Scripts:**
   - **setup.sh:** Script to set up the environment and dependencies.
   - **start.sh:** Script to start the application.
   - **stop.sh:** Script to stop the application.

Features

- Role-Based Access Control system
- Admin can manage users, campaigns and flagged content with approval of new sponsors.
- Sponsors can Create and manage campaigns, send ad requests and search for influencers.
- Influencers can Browse public campaigns, manage ad requests and negotiate terms.
- Daily reminders and monthly activity reports sent via email.
- Redis caching for faster data access with background task management via Celery.
- Sponsors can export campaign details as CSV files.

Video Link: [MAD-II Project Video]