Project Report: Influencer Engagement and Sponsorship Coordination Platform

1. Student Details

- **Name:** ASWIN BALA T
- **Roll Number:** 22F1001832
- **Course:** Modern Application Development I

2. Project Details

Project Title: Influencer Engagement and Sponsorship Coordination Platform (IESCP)

Problem Statement:

The Influencer Engagement and Sponsorship Coordination Platform (IESCP) is designed to connect sponsors with influencers to facilitate collaboration for product or service advertisements. Sponsors can create campaigns, send ad requests, and track campaign progress, while influencers can accept or negotiate ad requests and promote the sponsor's products or services in exchange for compensation. The platform also includes an admin role responsible for overseeing activities, managing flagged content, and ensuring the platform's integrity.

3. Approach

3.1 Development Process: The development of the IESCP was divided into several phases:

- **Requirement Analysis:** Understanding the project requirements, including user roles, functionalities, and key features.
- **Design:** Creating an ER diagram to map the database schema and designing the application wireframe to visualize the user flow.
- **Implementation:**
    - **Backend Development:** Flask was used to build the application’s backend, managing routes, handling requests, and interacting with the database.
    - **Frontend Development:** The frontend was developed using Jinja2 templates for rendering HTML combined with Bootstrap for responsive design.

    - **Database Management:** SQLite was chosen for data storage, ensuring a lightweight yet efficient database system.
- **Testing:** Testing was conducted to ensure the platform's functionality, including form validations, data storage, and user interactions.
- **Deployment:** The project was set up to run locally, allowing demonstrations and testing to be conducted on any machine.

3.2 Database Schema: The database schema was designed to handle various entities such as users, campaigns, and ad requests. The key tables include:

- **User:** Stores information about all users (admin, sponsors, influencers), including their roles and relevant details.
- **Campaign:** Manages campaign-related data, including campaign names, descriptions, budgets, and visibility (public or private).
- **AdRequest:** Links campaigns with influencers, managing ad request details like requirements, payment amounts, and statuses.
- **Message:** (Optional) Stores negotiation messages between sponsors and influencers.

3.3 Frontend Design:

- The frontend was developed using Bootstrap for a clean and responsive design.
- Jinja2 templates were used to dynamically render content based on user interactions and database queries.
- Form validation was implemented both on the frontend (HTML5 and JavaScript) and backend (Flask) to ensure data integrity.

4. Frameworks and Libraries Used

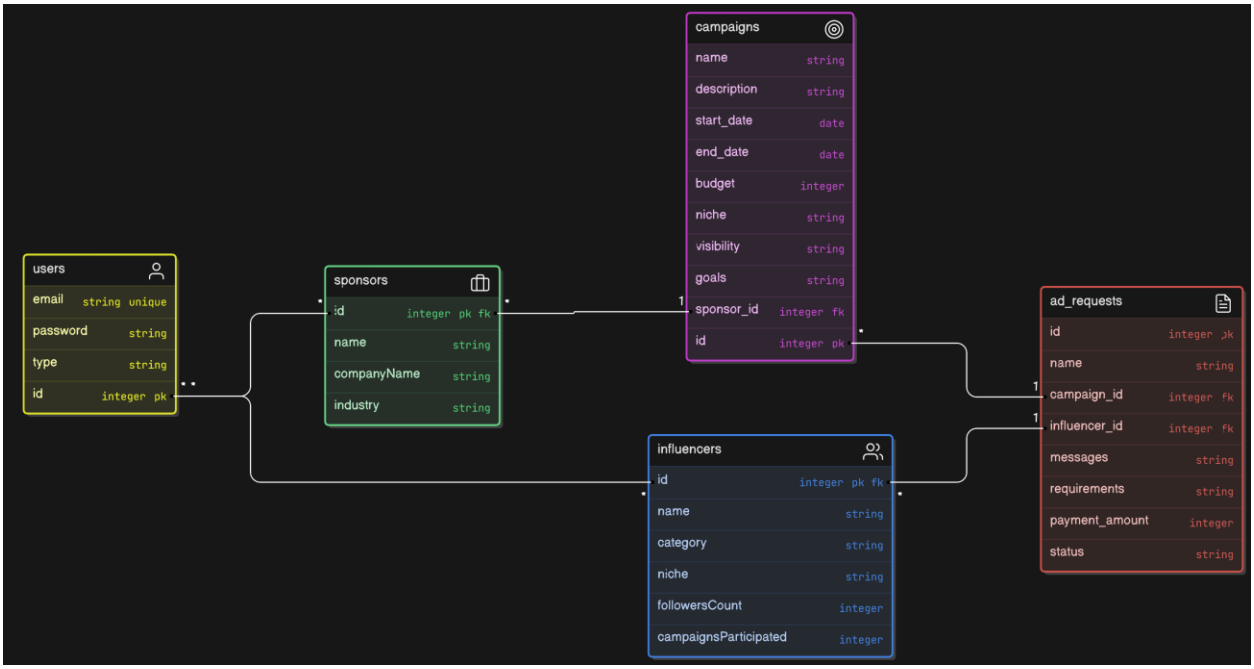
- **Flask:** A micro web framework used for backend development, managing routes, requests, and interactions with the database.
- **Jinja2:** A templating engine for rendering HTML templates dynamically, based on data from the Flask backend.
- **Bootstrap:** A popular CSS framework used for designing a responsive and aesthetically pleasing frontend.
- **SQLite:** A lightweight relational database management system used to store user data, campaigns, and ad requests.

- **flask_restful (Optional):** A Flask extension used to create RESTful API endpoints for interacting with the application data.
- **ChartJS (Optional):** A JavaScript library used for creating interactive charts to display statistics on the admin dashboard.

5. ER Diagram

The ER diagram provides a visual representation of the database schema, showing the tables and their relationships. It includes the following tables:

- **User:** Contains attributes such as user_id (primary key), username, password, role (admin, sponsor, influencer), and other relevant details.
- **Campaign:** Contains campaign_id (primary key), name, description, start_date, end_date, budget, visibility (public/private), and other campaign details.
- **AdRequest:** Contains ad_request_id (primary key), campaign_id (foreign key), influencer_id (foreign key), requirements, payment_amount, status, and other ad request details.
- **Message (Optional):** Contains message_id (primary key), ad_request_id (foreign key), content, timestamp, and other message details.



6. Core Functionalities

6.1 User Authentication and Authorization:

- Users can register and log in as either an admin, sponsor, or influencer.
- Separate forms were created for each user role with basic HTML form validation.
- The user role determines the features accessible within the platform.

6.2 Admin Dashboard:

- The admin dashboard provides an overview of platform statistics, including active users, campaigns, ad requests, and flagged content.
- Admins can monitor user activities, flag inappropriate content, and manage the overall platform.

6.3 Campaign Management (For Sponsors):

- Sponsors can create new campaigns, categorizing them into various niches.
- Existing campaigns can be updated or deleted based on the sponsor's requirements.
- Campaign visibility can be set as public or private.

6.4 Ad Request Management (For Sponsors):

- Sponsors can create ad requests linked to their campaigns, specifying requirements, payment amounts, and statuses.
- Existing ad requests can be edited or deleted.

6.5 Influencer Interaction with Ad Requests:

- Influencers can view all ad requests from different campaigns.
- They have the ability to accept, reject, or negotiate ad requests.
- Influencers can search for public campaigns based on niche, relevance, and budget.

7. Recommended Functionalities (Optional)

7.1 API Resources:

- RESTful API endpoints were created for managing users, campaigns, and ad requests.
- The APIs return JSON responses and are useful for integrating with other platforms or automating tasks.

7.2 Frontend and Backend Validation:

- Form fields were validated using HTML5 and JavaScript on the frontend.
- Additional backend validation was implemented in Flask to ensure data integrity and security.

7.3 External Libraries:

- ChartJS was integrated to display interactive charts on the admin dashboard, providing insights into user activities and campaign performance.

8. Presentation Video

- http://drive.google.com/drive/folders/1NREWUGM_PH_IH6eSqyuMNDDQEEIplCiy?usp=drive_link
- The video provides an overview of the project, demonstrating the key functionalities and explaining the approach taken during development.