Influencer Engagement and Sponsorship Platform
A platform to connect influencers to sponsors

A Project report for the MAD I Project

Submitted by:-

Name: Tathagat Suman

Roll number: 22F3002936

22f3002936@ds.study.iitm.ac.in

IITM Online BS Degree Program,

Indian Institute of Technology, Madras, Chennai

Tamil Nadu, India, 600036

Question Statement:

Create a web application for managing social media influencers and their campaigns. The application should allow users to register as influencers or sponsors, create and manage campaigns, send and manage requests, and perform administrative tasks such as flagging or deleting users and campaigns.

Approach:

Database Design: Designed the database schema to manage users, campaigns, requests, and influencer information. Ensured data integrity with relationships and constraints.

Backend Implementation: Used Flask for server-side logic. Implemented user authentication and authorization with Flask-Login, and handled database interactions using SQLAlchemy.

Frontend Implementation: Created templates using Flask's rendering capabilities. Developed forms for user registration, campaign management, and request handling.

APIs and Routes: Implemented RESTful routes for CRUD operations on users, campaigns, and requests, ensuring appropriate permissions and access control.

Frameworks and Libraries Used:

Flask: For creating the web application and handling routing, rendering templates, and request processing.

Flask-SQLAlchemy: For ORM-based database interactions.

Flask-Migrate: For handling database migrations.

Flask-Login: For user authentication and session management.

SQLite: As the database for storing user, campaign, and request data.

ER Diagram:

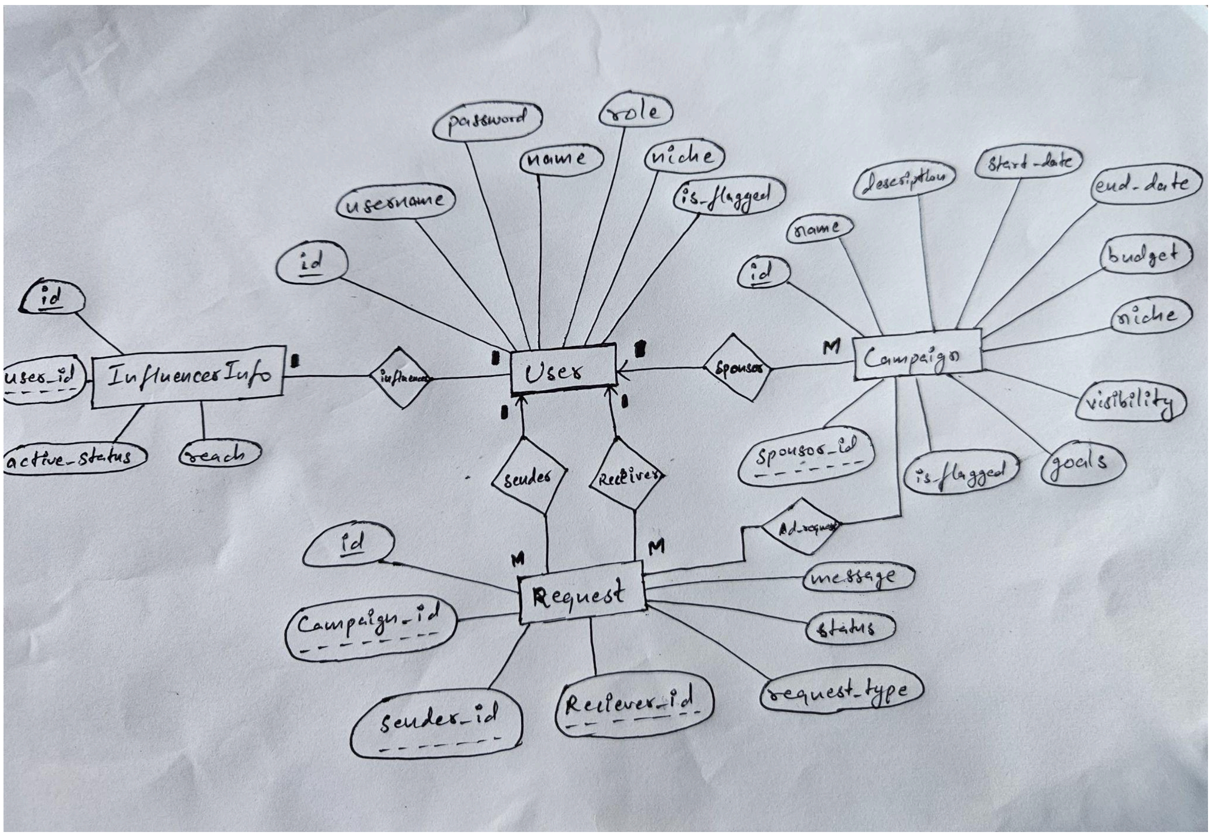

fig. ER-Diagram of the Database Schema that I will be using.

Entities and Relationships:

User: Represents users of the system with roles of admin, sponsor, or influencer. Manages influencer-specific information and campaigns.

InfluencerInfo: Contains additional details for users with the role of influencer.

Campaign: Represents campaigns created by sponsors, including attributes like budget, description, and visibility.

Request: Represents requests sent between influencers and sponsors related to campaigns.

Relationships:

One-to-One: User to InfluencerInfo

One-to-Many: User to Campaign, User to Request (as sender/receiver)

Many-to-One: Request to Campaign

API Resource Endpoints:

GET /: Home page.

GET /login: Login page.

POST /login: Handle login form submission.

GET /register: Registration page.

POST /register: Handle user registration.

GET /dashboard: Dashboard for users based on their role.

POST /admin/flag/user/int:user_id: Flag a user (admin only).

POST /admin/unflag/user/int:user_id: Unflag a user (admin only).

POST /delete_user/int:user_id: Delete a user (admin only).

POST /create_campaign: Create a new campaign.

POST /edit_campaign/int:campaign_id: Edit an existing campaign.

POST /admin/flag/campaign/int:campaign_id: Flag a campaign (admin only).

POST /admin/unflag/campaign/int:campaign_id: Unflag a campaign (admin only).

POST /delete_campaign/int:campaign_id: Delete a campaign.

POST /send_request/int:campaign_id/int:receiver_id: Send a request to an influencer.

POST /request_influencer: Send a request from a sponsor to an influencer.

POST /delete_request/int:request_id: Delete a request.

POST /manage_request/int:request_id/<action>: Accept or reject a request.

GET /profile: View and update profile information.

Drive Link of the Presentation Video:

Presentation Video Link: click here to view