Author

Sannidhi Hemant Naik

23f1002758

23f1002758@ds.study.iitm.ac.in

I am currently a Diploma student in Bachelor of Science Degree in Data Science and Applications by IIT Madras. I am also a third-year student pursuing a B.Sc. in Computer Science, with a strong passion for technology. In addition to my academic pursuits, I have been dedicated to studying Hindustani Classical music for the past nine years. I thrive on learning through new experiences.

Description of the project:

The project involves creating a Flask application with SQLAlchemy to manage influencers, sponsors, and campaigns. Key tasks include implementing user authentication, API endpoints for CRUD operations, and dynamic views with Bootstrap for handling profiles, campaigns, ad requests, transactions, and statistics.

Technologies used

Flask, SQLite, HTML & Bootstrap, Jinja2, Chart.js.

Their purpose:

Flask – It is a web framework. It has been used for handling routes, templates, and for request/response management.

SQLite- For managing data management with models.

HTML & Bootstrap- For layout, responsive design and styling.

Jinja2- to render HTML with dynamic data.

Chart.js – for creating dynamic charts and graphs.

Flask-extensions and their purpose:

Flask-SQLAlchemy: Provides an Object-Relational Mapping (ORM) layer for interacting with the database, making it easier to work with database records as Python objects, **Flask-Migrate**: Manages database migrations. It allows for easy changes to the database schema and keeps track of these changes over time, **Flask-Login**: Manages user sessions, including login, logout, and user authentication. It simplifies user management and session handling, **Werkzeug**: Provides utilities for handling file uploads securely with functions like secure_filename to prevent directory traversal attacks.

DB Schema Design

1. **User Table**: **id**: Integer, Primary Key. Unique identifier for each user, **name**: String(150), Not Null. The full name of the user, **username**: String(150), Not Null. The unique username for login, **password**: String(150), Not Null. The hashed password for authentication, **role**: String(50), Not Null. The role assigned to the user (e.g., 'admin', 'sponsor'). ***Constraints***: Primary key on id, Unique constraint on username to ensure each username is unique.

2. **Influencer Table** : **id**: Integer, Primary Key. Unique identifier for each influencer, **user_id**: Integer, Foreign Key. References user.id, Not Null. Links the influencer to a user account, **name**: String(64), Not Null. The name of the influencer, **platform**: String(64), Not Null. The social media platform used by the influencer, **niche**: String(64), Not Null. The niche or category of the influencer's content, **reach**: Integer, Not Null. The number of followers or audience reach, **profile_picture**: String(128), Nullable. The file path of the influencer's profile picture. ***Constraints*:** Foreign key on user_id to link with the User table.

3. **Section Table:** **id**: Integer, Primary Key. Unique identifier for each section, **name**: String(64), Not Null. The name of the section, **date_created**: DateTime, Not Null, Default: Current UTC time. The creation date of the section. ***Constraints*:** Primary key on id.

4. **Campaign Table**: **id**: Integer, Primary Key. Unique identifier for each campaign.name: String(64), Not Null. The name of the campaign**, date_created**: DateTime, Not Null, Default: Current UTC time. The creation date of the campaign, **description**: String(84), Not Null. A brief description of the campaign, **start_date**: DateTime, Not Null. The start date of the campaign, **end_date**: DateTime, Not Null. The end date of the campaign, **budget**:

Integer, Not Null. The budget allocated for the campaign, **visibility**: String(84), Not Null. The visibility setting of the campaign (e.g., 'public', 'private'), **goals**: String(84), Not Null. The goals of the campaign, **niche**: String(64), Not Null. The target niche of the campaign, **user_id**: Integer, Foreign Key. References user.id, Not Null. Links the campaign to a user account. ***Constraints*:** Foreign key on user_id to link with the User table.

5. **Ad Table**: **id**: Integer, Primary Key. Unique identifier for each ad**, campaign_id**: Integer, Foreign Key. References campaign.id, Not Null. Links the ad to a campaign**, influencer_id**: Integer, Foreign Key. References influencer.id, Not Null. Links the ad to an influencer, **messages**: Text, Not Null. The content of the ad, **requirements**: Text, Nullable. Any additional requirements for the ad, **payment_amount**: Float, Not Null. The payment amount for the ad, **status**: String(50), Not Null, Default: 'Pending'. The status of the ad (e.g., 'Pending', 'Accepted', 'Rejected'), **created_at**: DateTime, Default: Current UTC time. The creation date of the ad. ***Constraints*:** Foreign key on campaign_id to link with the Campaign table, Foreign key on influencer_id to link with the Influencer table.

6. **Transaction Table**: **id**: Integer, Primary Key. Unique identifier for each transaction, **influencer_id**: Integer, Foreign Key. References influencer.id, Not Null. Links the transaction to an influencer, **user_id**: Integer, Foreign Key. References user.id, Not Null. Links the transaction to a user, **amount**: Float, Not Null. The amount of the transaction, **date**: DateTime, Default: Current UTC time. The date of the transaction, **status**: String(50), Not Null. The status of the transaction**, ad_request_id**: Integer, Foreign Key, Nullable. References ad.id. Links the transaction to an ad request. **request_type**: String(50), Not Null. The type of the request (e.g., 'payment', 'refund'). ***Constraints***: Foreign key on influencer_id to link with the Influencer table, Foreign key on user_id to link with the User table, Foreign key on ad_request_id to link with the Ad table (if applicable).

8. **Flag Table: id**: Integer, Primary Key. Unique identifier for each flag, **item_type**: String(50), Not Null. Type of item being flagged (e.g., 'Influencer', 'Sponsor', 'Campaign'), **item_id**: Integer, Not Null. Identifier of the flagged item, **reason**: String(255), Nullable. The reason for flagging the item, **timestamp:** DateTime, Default: Current UTC time**.** The date and time when the flag was created.

API Design

User Management: Implemented user registration and authentication using Flask-Login, managing user sessions and roles

Influencer Management: APIs to add, update, and view influencer profiles, including profile picture uploads,

Campaign Management: Endpoints to create, update, and manage campaigns, including associating them with influencers,

Ad Management: APIs to create and manage ads linked to campaigns and influencers, including status updates

Transaction Handling: Supported recording and querying transactions related to ad payments and refunds

Flagging System: Provided functionality to flag items for review, with endpoints to create and retrieve flags.

Basic Architecture:

Controllers, which handle business logic and routing, are located in the **routes** module. Templates are housed in the **templates** directory, with HTML files for rendering views such as user dashboards, campaign, influencers, campaign and ad details. Static assets like CSS and JavaScript files are stored in the **static** folder.

Video of the project:

https://drive.google.com/file/d/1bQE6aO9o4NqH1-buYZMhtIBwZQ89XdcZ/view?usp=sharing