Application Overview Report

Project : Influencer Engagement and Sponsorship Coordination Program

Backend	Python , Flask , Jinja 2 , uuid , werkzeug , Flask_login, FLask *mail, Flask*sqlalchemy, utils
Frontend	HTML, CSS , JAVASCRIPT , BOOTSTRAP

1. User Routes

Home Page

- **Path:** /
- **Method:** GET
- **Function:** Displays the main landing page of the app.

Login

- **Path:** /login
- **Methods:** GET, POST
- **Purpose:** Handles user sign-in, including username, email, and password verification, as well as role-based access (admin vs. regular user). Two-factor authentication is also available.

Logout

- **Path:** /logout
- **Method:** GET
- **Function:** Logs users out, updates their status to offline, and redirects them to the login page.

Profile Management

- **Path:** /profile/<string:user>
- **Methods:** GET, POST
- **Features:** Lets users view and edit their profiles, including uploading or removing profile pictures. Displays user details, posts, events, and announcements.

Change Password

- **Path:** /change-password/<string:user>
- **Methods:** GET, POST
- **Use:** Allows users to update their passwords after verifying their old password and confirming the new one.

Delete Account

- **Path:** /delete_account/<string:user>
- **Methods:** GET, POST
- **Purpose:** Enables users to delete their accounts, but only after verifying their email.

2. Help & Support

Help Requests

- **Path:** /help
- **Methods:** GET, POST
- **Function:** Allows users to submit help requests detailing their issues, which are tracked by a unique ID.

Forgot Password

- **Path:** /forgot
- **Methods:** GET, POST
- **Purpose:** Assists users in recovering their passwords via email verification and OTP. Users can reset their passwords once they verify the OTP.

3. Admin Routes

Admin Dashboard

- **Path:** /admin/dashboard
- **Method:** GET
- **Function:** Displays the admin dashboard, including a list of active campaigns.

Admin List

- **Path:** /admin/list
- **Methods:** GET, POST
- **Purpose:** Provides admins with lists of users, flagged items, campaigns, and ads for management.

Admin Actions

- **Path:** /admin/<userId>/<action>

- **Methods:** GET, POST
- **Use:** Allows admins to perform actions like deleting, deactivating, or activating user accounts.

Admin Search

- **Path:** /admin/search
- **Methods:** GET, POST
- **Purpose:** Enables admins to search for flagged items within the admin panel.

4. Influencer Routes

Influencer Dashboard

- **Path:** /influencer_dashboard/<string:user_1>
- **Methods:** GET, POST
- **Features:** Displays the influencer's dashboard with posts, campaigns, events, and announcements. Influencers can also manage their profile visibility and two-factor authentication settings.

Campaign Analysis

- **Path:** /chart-data/influencer-campaigns/<sponsor_id>
- **Method:** GET
- **Purpose:** Provides analysis of campaigns associated with influencers.

Influencer Search

- **Path:** /search/inf
- **Method:** GET
- **Use:** Enables searching for influencers, with filters to exclude flagged users.

5. Sponsor Routes

Sponsor Dashboard

- **Path:** /sponsor_dashboard/<string:user_1>
- **Methods:** GET, POST
- **Features:** Displays the sponsor's dashboard with campaigns. Sponsors can manage their profile visibility and two-factor authentication settings.

Add Campaign

- **Path:** /sponsor_dashboard/add_campaigns/<string:user>
- **Methods:** GET, POST

- **Purpose:** Allows sponsors to create new campaigns, including the option to upload campaign images.

Ads Analysis

- **Path:** `/chart-data/ads/<sponsor_id>`
- **Method:** GET
- **Function:** Provides analysis of ad statuses (pending, negotiation, accepted, rejected) for a specific sponsor.

6. Reporting & Analysis

General Reports

- **Path:** `/info/<string:data>`
- **Method:** GET
- **Purpose:** Offers statistics and information based on the selected category (admin or user).

Chart Data

- **Path:** `/chart-data/<category>/admin`
- **Method:** GET
- **Function:** Provides visual analytics for various categories (e.g., sponsor, user, ads, campaigns).

7. File Management

File Uploads

- **Path:** `/uploads/<filename>`
- **Method:** GET
- **Use:** Serves files uploaded to the server.

Save File Function

- **Function:** `save_file(file)`
- **Purpose:** Saves files to the server directory.

Allowed File Types

- **Extensions:** `png`, `jpg`, `jpeg`, `gif`

8. Security & Utility

OTP Verification

- **Function:** `send_otp(email, otp)`
- **Purpose:** Sends OTP to email for password reset verification.
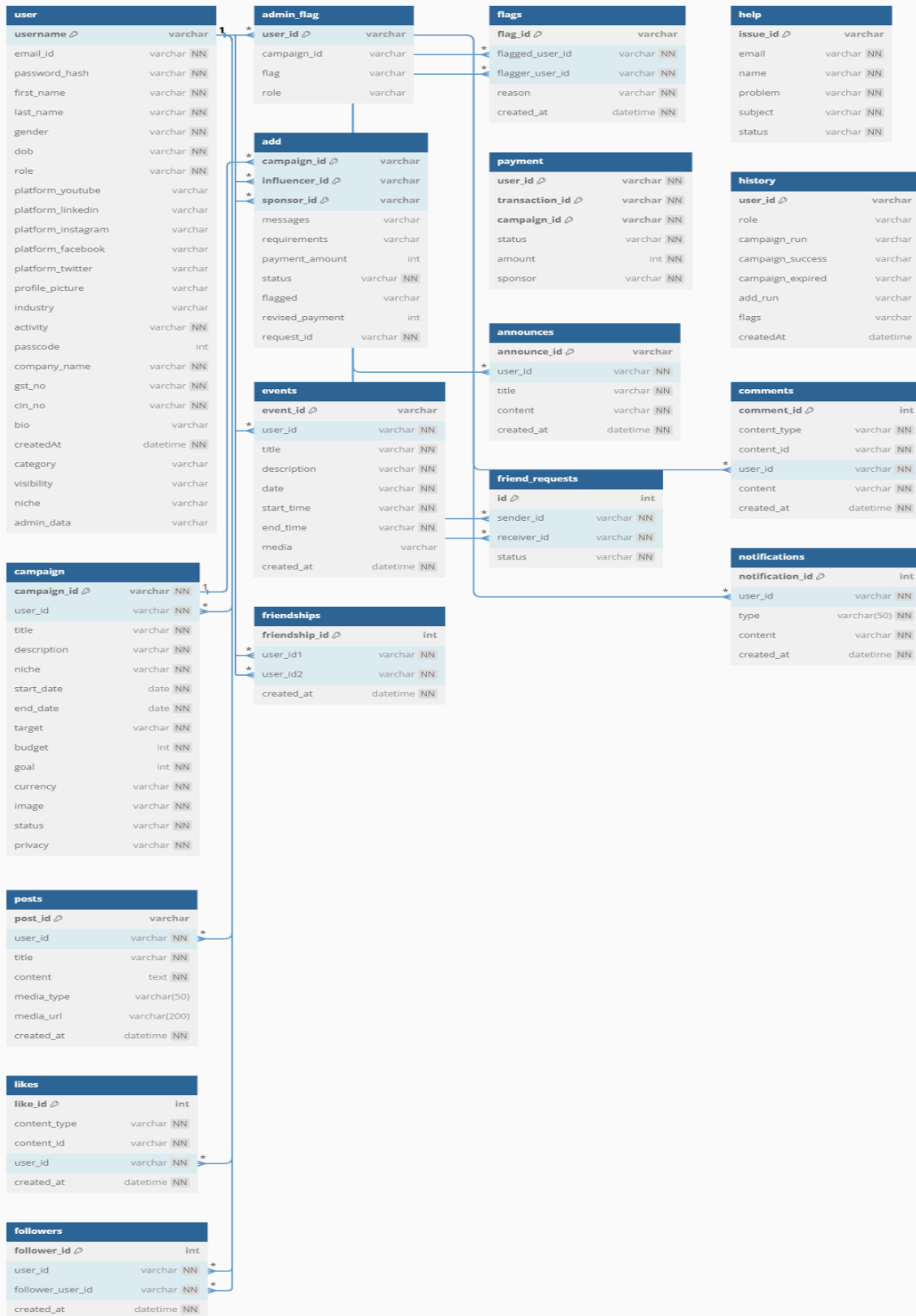
User Activity Tracking

- **Function:** Updates user status to online or offline during login and logout.

Password Hashing

- **Use:** Ensures secure storage and verification of passwords.

Entity-Relationship Diagram

user

username	varchar	
email_id	varchar	NN
password_hash	varchar	NN
first_name	varchar	NN
last_name	varchar	NN
gender	varchar	NN
dob	varchar	NN
role	varchar	NN
platform_youtube	varchar	
platform_linkedin	varchar	
platform_instagram	varchar	
platform_facebook	varchar	
platform_twitter	varchar	
profile_picture	varchar	
industry	varchar	
activity	varchar	NN
passcode	int	
company_name	varchar	NN
gst_no	varchar	NN
cin_no	varchar	NN
bio	varchar	
createdAt	datetime	NN
category	varchar	
visibility	varchar	
niche	varchar	
admin_data	varchar	

campaign

campaign_id	varchar	NN
user_id	varchar	NN
title	varchar	NN
description	varchar	NN
niche	varchar	NN
start_date	date	NN
end_date	date	NN
target	varchar	NN
budget	int	NN
goal	int	NN
currency	varchar	NN
image	varchar	NN
status	varchar	NN
privacy	varchar	NN

posts

post_id	varchar	
user_id	varchar	NN
title	varchar	NN
content	text	NN
media_type	varchar(50)	
media_url	varchar(200)	
created_at	datetime	NN

likes

like_id	int	
content_type	varchar	NN
content_id	varchar	NN
user_id	varchar	NN
created_at	datetime	NN

followers

follower_id	int	
user_id	varchar	NN
follower_user_id	varchar	NN
created_at	datetime	NN

admin_flag

user_id	varchar	
campaign_id	varchar	
flag	varchar	
role	varchar	

add

campaign_id	varchar	
influencer_id	varchar	
sponsor_id	varchar	
messages	varchar	
requirements	varchar	
payment_amount	int	
status	varchar	NN
flagged	varchar	
revised_payment	int	
request_id	varchar	NN

events

event_id	varchar	
user_id	varchar	NN
title	varchar	NN
description	varchar	NN
date	varchar	NN
start_time	varchar	NN
end_time	varchar	NN
media	varchar	
created_at	datetime	NN

friendships

friendship_id	int	
user_id1	varchar	NN
user_id2	varchar	NN
created_at	datetime	NN

flags

flag_id	varchar	
flagged_user_id	varchar	NN
flagger_user_id	varchar	NN
reason	varchar	NN
created_at	datetime	NN

payment

user_id	varchar	NN
transaction_id	varchar	NN
campaign_id	varchar	NN
status	varchar	NN
amount	int	NN
sponsor	varchar	NN

announces

announce_id	varchar	
user_id	varchar	NN
title	varchar	NN
content	varchar	NN
created_at	datetime	NN

friend_requests

id	int	
sender_id	varchar	NN
receiver_id	varchar	NN
status	varchar	NN

help

issue_id	varchar	
email	varchar	NN
name	varchar	NN
problem	varchar	NN
subject	varchar	NN
status	varchar	NN

history

user_id	varchar	
role	varchar	
campaign_run	varchar	
campaign_success	varchar	
campaign_expired	varchar	
add_run	varchar	
flags	varchar	
createdAt	datetime	

comments

comment_id	int	
content_type	varchar	NN
content_id	varchar	NN
user_id	varchar	NN
content	varchar	NN
created_at	datetime	NN

notifications

notification_id	int	
user_id	varchar	NN
type	varchar(50)	NN
content	varchar	NN
created_at	datetime	NN

dbdiagram.io

Best regards,
Yashvardhan