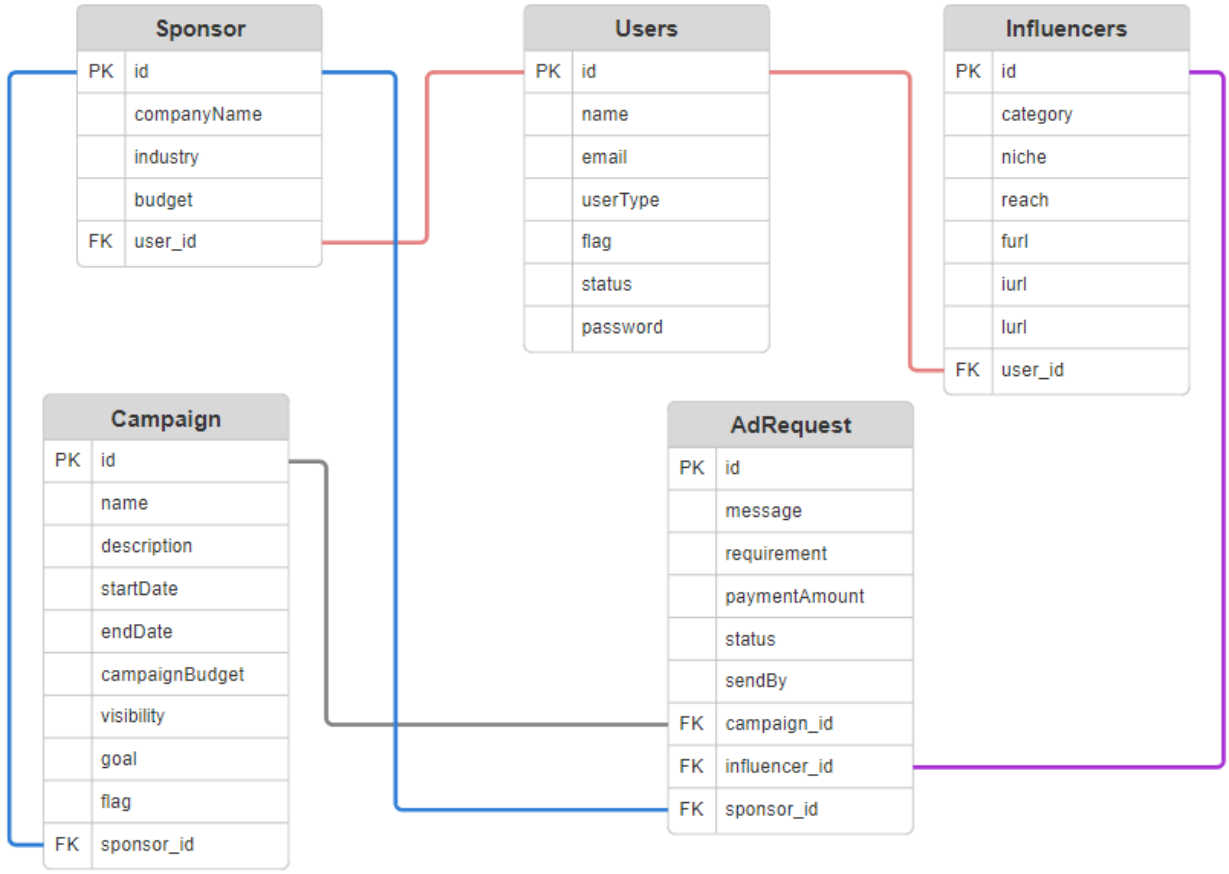
Name: Shagun Niranjan
Roll Number: 22dp2000121
Email: 22dp2000121@ds.study.iitm.ac.in
Introduction: Myself Shagun Niranjan current Pursuing Diploma in Programming from IIT Madras and also completed MCA from VIT BHOPAL. I possess technical skills in HTML, CSS, Java and Python.

Description: Influencer Engagement & Sponsorship Coordination (IE&SC) is a platform that facilitates collaboration between sponsors and influencers. Sponsors can create campaigns and send requests to influencers. Influencers can also initiate requests for campaigns. Both sponsors and influencers have the ability to reject, accept, or negotiate these requests. Sponsors can manage campaigns and requests by creating, updating, or deleting them. All users who sign up as sponsors must be approved by an admin. The admin has the authority to block or unblock users and campaigns and has visibility into all requests on the platform.

Technologies used
Frontend: CSS/HTML, Bootstrap, Vue.js, Pinia Stores, VeeValidation, Axios, Vue Routes.
Backend: Python, Flask, datetime, SQLite, SQLAlchemy

Database ER diagram



API Design

Authentication
The API uses token-based authentication for secure access to certain endpoints. The @token_required decorator is used to enforce this authentication.
Resources & Endpoints

1. **User Resource**
   - **Endpoint**: /user, /user/<int:user_id>
   - **Methods**:
     - GET: Retrieve all users or a specific user by ID. **Responses**: 200: Success, returns user(s), 404: User not found.
     - POST: Create a new user. **Body Parameters**: name (str), email (str, required), password (str, required), userType (str, required) ,flag (bool, required), status (str, required), **Responses**: 201: User created., 409: User already exists.
     - PUT: Update an existing user by ID. **Body Parameters**: Same as POST. **Responses**: 200: User updated, 404: User not found.
     - DELETE: Delete a user by ID. **Responses**: 200: User deleted, 404: User not found.
2. **Influencer Resource**

- **Endpoint**: /influencer, /influencer/<int:influencer_id>
- **Methods**:
  - GET: Retrieve all influencers or a specific influencer by ID. **Responses**: 200: Success, returns influencer(s), 404: Influencer not found.
  - POST: Create a new influencer profile. **Body Parameters**: category (str, required), niche (str, required), reach (int, required), furl, iurl, lurl (str, required), user_id (int, required). **Responses**: 201: Influencer created, 409: Influencer already exists.
  - PUT: Update an existing influencer by ID. **Body Parameters**: Same as POST. **Responses**: 200: Influencer updated, 404: Influencer not found.
  - DELETE: Delete an influencer by ID. **Responses**: 200: Influencer deleted, 404: Influencer not found.

3. **Sponsor Resource**
   - **Endpoint**: /sponsor, /sponsor/<int:sponsor_id>
   - **Methods**:
     - GET: Retrieve all sponsors or a specific sponsor by ID. **Responses**: 200: Success, returns sponsor(s), 404: Sponsor not found.
     - POST: Create a new sponsor profile. **Body Parameters**: companyName (str, required), industry (str, required), budget (int, required), user_id (int, required). **Responses**: 201: Sponsor created, 409: Sponsor already exists.
     - PUT: Update an existing sponsor by ID. **Body Parameters**: Same as POST. **Responses**: 200: Sponsor updated, 404: Sponsor not found.
     - DELETE: Delete a sponsor by ID. **Responses**: 200: Sponsor deleted, 404: Sponsor not found.
4. **Campaign Resource**
   - **Endpoint**: /campaign, /campaign/<int:campaign_id>
   - **Methods**:
     - GET: Retrieve all campaigns or a specific campaign by ID. **Responses**: 200: Success, returns campaign(s), 404: Campaign not found.
     - POST: Create a new campaign, **Body Parameters**: name (str, required), description (str, required), startDate, endDate (str, required), campaignBudget (int, required), visibility, goal (str, required), flag (bool, required), sponsor_id (int, required). **Responses**: 201: Campaign created, 409: Campaign already exists.
     - PUT: Update an existing campaign by ID. **Body Parameters**: Same as POST. **Responses**: 200: Campaign updated, 404: Campaign not found.
     - DELETE: Delete a campaign by ID. **Responses**: 200: Campaign deleted, 404: Campaign not found.
5. **Ad Request Resource**
   - **Endpoint**: /adRequest, /adRequest/<int:adRequest_id>
   - **Methods**:
     - GET: Retrieve all ad requests or a specific ad request by ID. **Responses**: 200: Success, returns ad request(s), 404: Ad request not found.
     - POST: Create a new ad request. **Body Parameters**: message (str, required), requirement (str, required), paymentAmount (int, required), status, sendBy (str, required), campaign_id, influencer_id, sponsor_id (int, required). **Responses**: 201: Ad request created, 409: Ad request already exists.
     - PUT: Update an existing ad request by ID. **Body Parameters**: Same as POST. **Responses**: 200: Ad request updated, 404: Ad request not found.
     - DELETE: Delete an ad request by ID. **Responses**: 200: Ad request deleted, 404: Ad request not found.

Common Response Codes

- **200 OK**: The request was successful.
- **201 Created**: The resource was successfully created.
- **404 Not Found**: The specified resource could not be found.
- **409 Conflict**: There was a conflict with the existing resource.

CORS

CORS is enabled for http://localhost:8081 to allow cross-origin requests during local development.

Celery Integration

Celery is used for background task processing, with configurations loaded from LocalConfig.

Video link: https://drive.google.com/file/d/1SMvWEKlzf2th2JzSV3QnuR5mllnTYXQ2/view?usp=sharing