Final Project Plan

Group members: Logan Miles, Abhi Bedi, Faris, Nicholas Fason, Taylor Shirah

**1. Choose Technology Stack:**

* Frontend: React
* Backend: Flask or Django
* Database: PostgreSQL (relational) or MongoDB (non-relational)
* Hosting: AWS, Heroku, or similar

**2. Initialize Project:**

* Set up version control with Git
* Initialize a repository on GitHub
* Set up the frontend and backend frameworks

## **Features and Implementation**

**1. Sign up/Sign in System**

* User Model:
  + Fields: `username`, `password` (hashed), `email`
* Endpoints:
  + `/signup`: Create a new user
  + `/signin`: Authenticate user
* Frontend:
  + Sign up form
  + Sign in form
* Security:
  + Use JWT for session management

**2. Profile Management**

* Profile Model:
  + Fields: `bio`, `profile\_picture`, `user` (foreign key)
* Endpoints:
  + `/profile`: GET (view profile), PUT (update profile)
* Frontend:
  + Profile page to display user information
  + Edit profile form to update information and upload a profile picture

**3. Search Functionality**

* Search Endpoint:
  + `/search`: GET (query parameter for search term)
* Search Logic:
  + Query the user model based on username or other profile fields
* Frontend:
  + Search bar
  + Display search results in a list with links to user profiles

**4. Follow System**

* Follow Model:
  + Fields: `follower` (foreign key to user), `followed` (foreign key to user), `status` (pending, accepted)
* Endpoints:
  + `/follow`: POST (send follow request)
  + `/follow/accept`: POST (accept follow request)
  + `/follow/decline`: POST (decline follow request)
* Frontend:
  + Follow button on user profiles
  + Notification or dashboard for managing follow requests

**5. Posting**

* Post Model:
  + Fields: `user` (foreign key), `content` (text), `image`, `video`, `timestamp`
* Endpoints:
  + `/post`: POST (create post), GET (get posts)
* Frontend:
  + Form to create a new post (text, image, video)
  + Display posts in a feed format

**6. Interactions**

* Like Model:
  + Fields: `user` (foreign key), `post` (foreign key)
* Comment Model:
  + Fields: `user` (foreign key), `post` (foreign key), `comment`, `timestamp`
* Endpoints:
  + `/like`: POST (like a post)
  + `/comment`: POST (comment on a post)
* Frontend:
  + Like button and comment form on each post
  + Display number of likes and comments

## **Bonus Features**

**1. Groups**

* Group Model:
  + Fields: `name`, `description`, `admin` (foreign key to user)
* Group Membership Model:
  + Fields: `group` (foreign key), `user` (foreign key)
* Endpoints:
  + `/group`: POST (create group), GET (list groups)
  + `/group/join`: POST (join group)
  + `/group/post`: POST (create post in group), GET (get group posts)
* Frontend:
  + Create group form
  + Group page to display group details and posts
  + Join group button
  + Group feed for posts shared within the group

**2. Notifications:**

* Notify users of new follow requests, accepted requests, new likes, and comments
* Messaging:
  + Private messaging between users
* Real-time Updates:
  + Use WebSockets to update feeds and notifications in real time

## **Deployment**

**1. Configure Deployment:**

* Set up CI/CD pipeline
* Configure environment variables

**2. Deploy:**

* Deploy backend and database
* Deploy frontend

**3. Testing:**

* Test all features thoroughly
* Perform user acceptance testing