Meet Up

Virtual-Physical Life Integration

MIS 6308.0W1 - System Analysis and Project Management - F24

Akshay Kadakol (AXK230315)

TABLE OF CONTENTS

Executive Summary

In today's increasingly digital world, social media platforms have become integral to how people interact, build relationships, and share experiences. However, most times, these interactions remain virtual and separate from one's natural life. The disconnection that exists between virtual and physical relationships denies people the possibility of deeper associations, meaningful collaborations, or personal growth. But despite all these digital networking tools and event-planning software, many systems fail at bridging this gap correctly, thus giving users fragmentary experiences and missed opportunities.

This proposed system approaches this challenge with the idea of incorporating virtual preferences with real-world interactions on a single platform. The system endeavors to give power to its users through smooth transitions of their online connections to real-life relationships. The system will analyze user preferences and interests to recommend to them events ofline that best match their social and professional goals. Users can join these events and share the invitations on their social media to encourage their online friends, followers, and mentors to participate in person. This feature promotes meaningful in-person connections, strengthening social bonds and fostering a sense of community.

Aside from connecting online relationships to ofline ones, the system will provide suggestions for online events fitting the users' interests; it will also allow users to invite their friends or acquaintances in real life into joining online activities. Be it virtual game interactions, live events, or common activities, these shared experiences will improve the quality of interactions between virtual and physical networks. Invitations can be created through multiple channels, such as via text message, email, or direct links to social media, making it accessible on several platforms and devices.

The platform will focus on robust data integration, ensuring data privacy for a personalized and enriching user experience. It will securely manage the data of user preferences and activity, offering accurate recommendations with the assurance of safeguarding user information. Feedback collected after events will further refine the system's algorithms, ensuring continuous improvement and greater user satisfaction over time.

By bridging the divide between virtual and physical connections, this system redefines how individuals navigate their social lives. It empowers users to foster deeper relationships, unlock networking opportunities, and enjoy a more cohesive and fulfilling lifestyle. This innovative approach will create a dynamic and interconnected social experience, bringing the best of both worlds together to enrich personal and professional interactions.

Problem Statement:

In today's world, social media users maintain separate virtual and physical lives, with connections like friends, followers, and mentors in each sphere often remaining detached. This disconnect limits opportunities for deeper relationships, collaboration, and personal growth.

With the rise of digital platforms, users often struggle to find meaningful ways to connect their virtual preferences with physical interactions. While many tools exist to help users network, plan events, and interact with others, these systems often fail to deliver personalized and seamless experiences that cater to connecting an individual's virtual and real-world.

There is a need for a system that bridges this gap by integrating virtual and physical connections. The system should enable users to transition online relationships into real-life interactions, enhance physical connections with virtual tools, and provide personalized recommendations for meaningful engagement. This concept will enable a more connected and fulfilling social experience.

Objective

In order to create a unified and seamless world in terms of virtual and physical lifestyle, this system aims at all actual possible interaction purposes through the use of the platform by users. It will allow the users to integrate their preferences, interests, and connections from the virtual world into real-world opportunities, creating a tighter cohesive experience socially. Additionally, the platform will recommend virtual events to allow real-world connections to interact virtually and participate in engaging virtual events. This will help in bridging the gap between both worlds for a dynamic and interconnected experience.

Key goals include:

Personalized Event Recommendations: Analyze user preferences and stored data, and create recommendations for events, activities, or meetings that best suit the interests and social circles of a user.

Connecting Virtual with Ofline Users: Find ways for users to meet their online friends, followers, or mentors in a physical setting and invite their ofline acquaintances to virtual events.

Data Integration: Smooth retrieval of data from the databases with a keen eye on user privacy and data security.

Improve through Feedback: User feedback should be collected and stored soon after the events to improve the recommendation algorithm for better and more accurate recommendations in the future.

By achieving these goals, the system aims to bridge the gap between user's virtual and physical worlds, fostering deeper relationships, enhancing networking opportunities through events, and creating a more engaging and personalized social experience. This system will empower users to seamlessly transition between virtual and real-world interactions, unlocking the potential of their combined social circles.

Scope

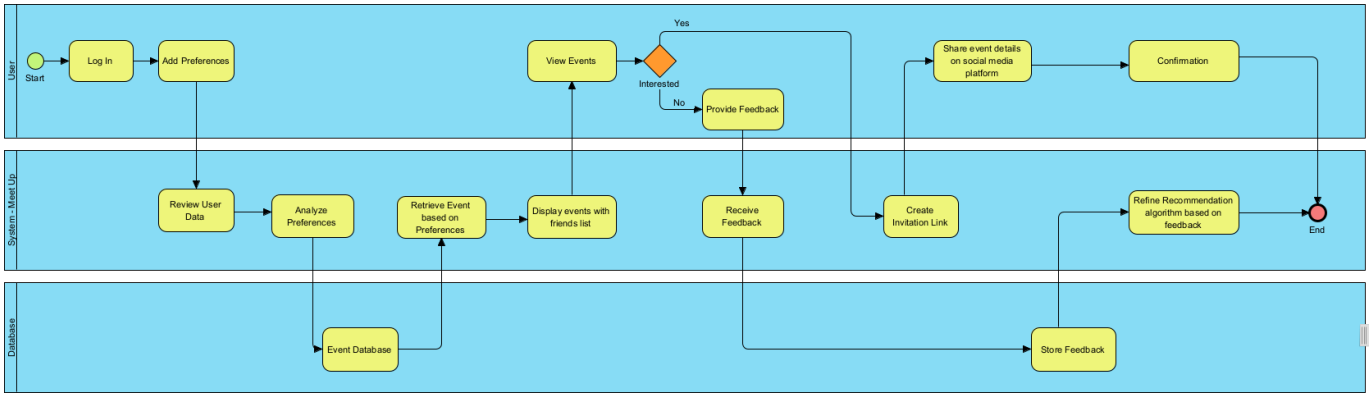
The system would recommend ofline events to online users according to their preference, enabling them to join and further share these event invitations on their social media. This would let all their online friends, mentors, and connections meet in person with them.

Also, the system will recommend online events to users based on their interests, which the online user can use to invite their ofline friends to join in. This may involve watching an event together or playing games online to better connect the virtual world with the real world. The system will have a feature to invite ofline friends to an online event via Text Messages or emails.

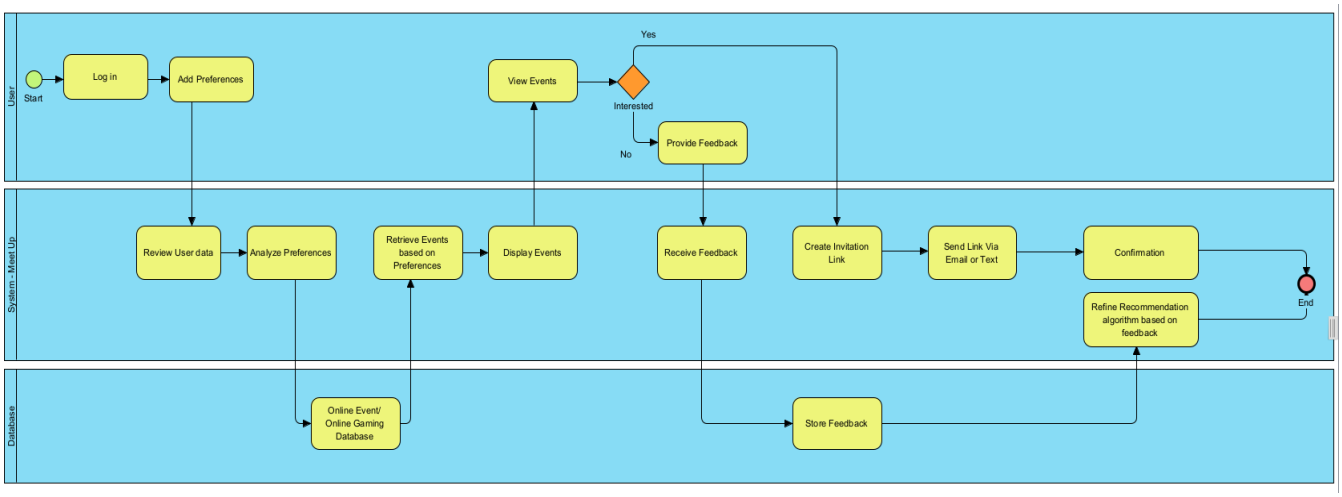
By doing so, the platform will enable users to transition between their online and ofline lives, fostering deeper relationships, improving networking opportunities, and delivering an enriched social experience. This system will focus on event-based interactions and connections while maintaining a high standard of privacy and security for all users. The system will prioritize data integration, ensure efficient handling of user information, and continuously refine its recommendation algorithm using user feedback.

Business Process Model Notation (BPMN)

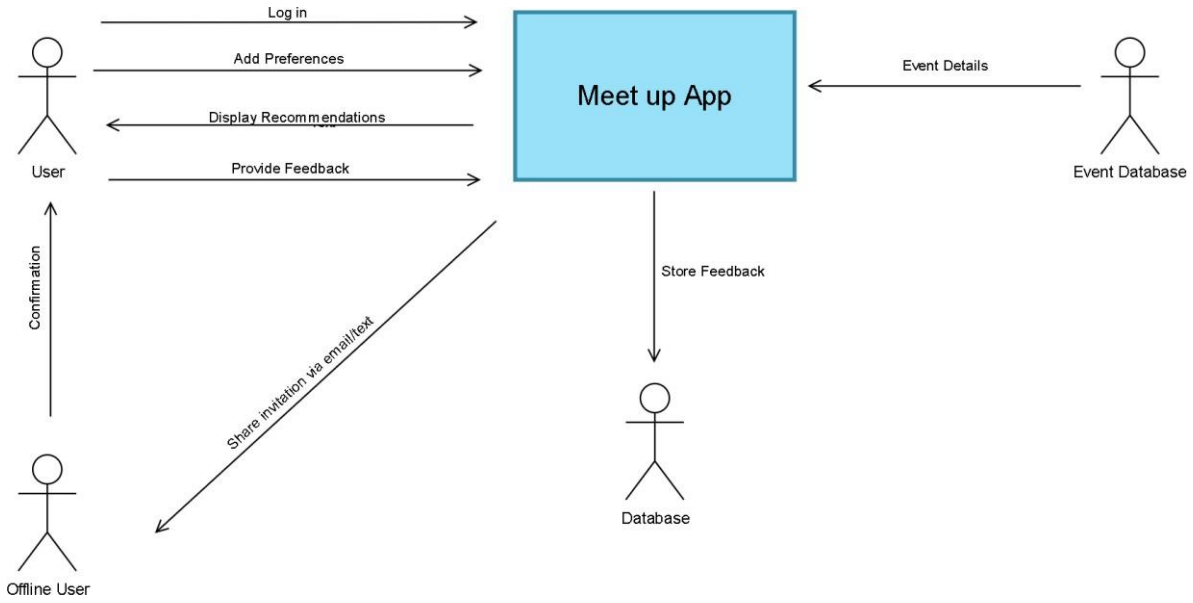
Offline Meet up



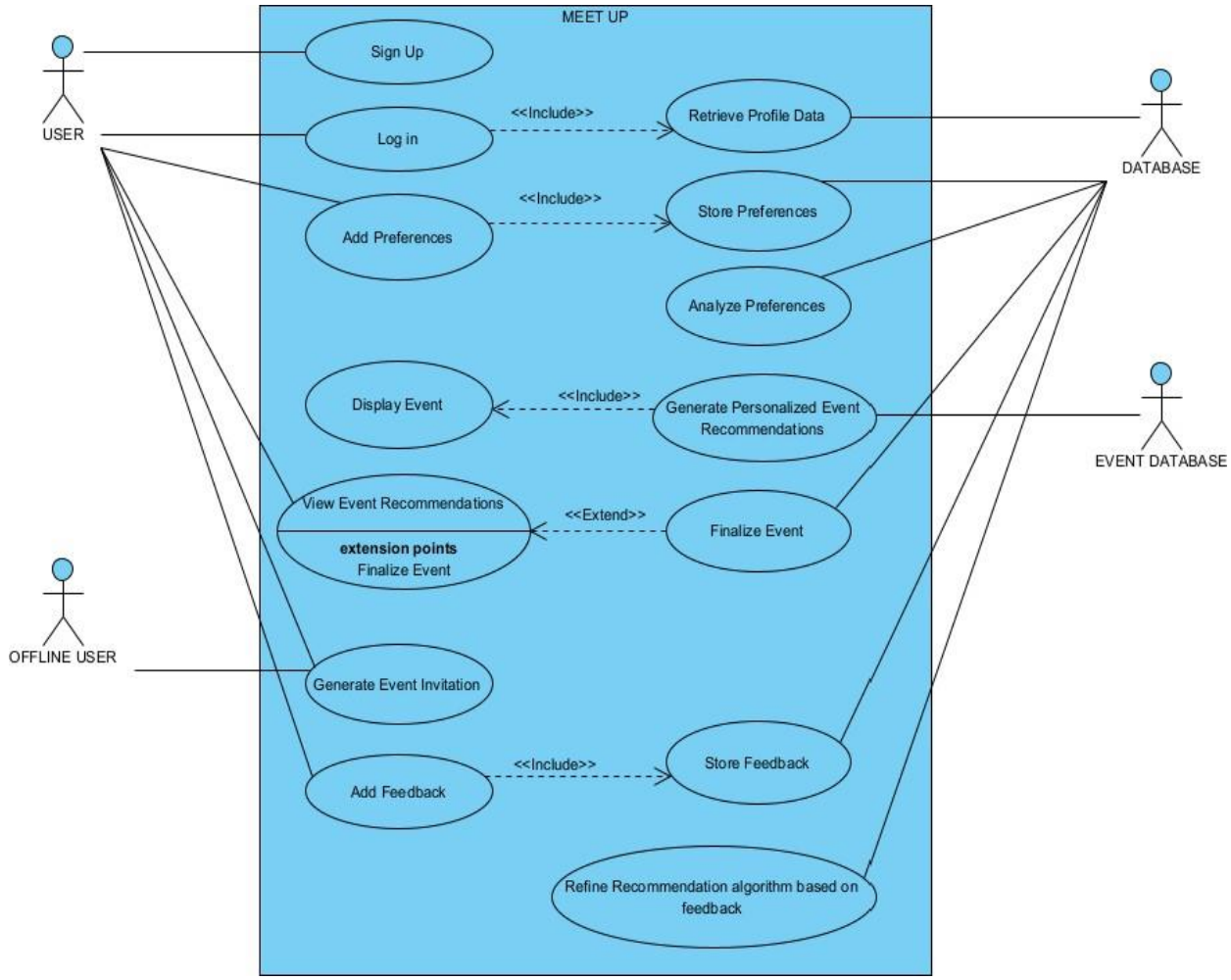
Online Meet up



Context Diagram



Use Case Diagram



Use Case Description:

Use Case Description #1

Use Case Name: Sign Up
Primary Actor: User
Stakeholders: Meet Up Application
Brief Description: The user creates an account using the Meet Up application.
Trigger: The user opens the application and selects the "Sign Up" option.
Normal Flow of Events: 1. The user selects "Sign Up" on the Meet Up application. 2. The system prompts users to enter user details (e.g., name, email, and password). 3. The user provides the required information. 4. The system validates the provided details. 5. The user is successfully registered and redirected to the home screen.
Exception Flow: Invalid Data: The system prompts the user to correct invalid or missing information.

Use Case Description #2

Use Case Name: Log In
Primary Actor: User
Stakeholders: Meet Up Application
Brief Description: The user logs into their account to access features.
Trigger: The user opens the application and selects the "Log In" option.
Normal Flow of Events: 1. The user selects "Log In" on the Meet Up application. 2. The system prompts the user to enter their credentials (username and password). 3. The user provides the credentials. 4. The system authenticates the credentials. 5. The user is logged into the application and redirected to the dashboard.
Exception Flow: Invalid Credentials: The system displays an error and prompts the user to re-enter valid login details.

Use Case Description #3

Use Case Name: Add Preferences
Primary Actor: User
Stakeholders: Meet Up Application
Brief Description: The user updates or sets preferences for events.
Trigger: The user accesses the preferences section from their account.
Normal Flow of Events: 1. The user navigates to the "Preferences" section. 2. The system displays options for event categories, locations, and interests. 3. User selects and updates their preferences. 4. The system stores the preferences.
Exception Flow: Invalid Input: The system prompts the user to correct or complete the preferences section.

Use Case Description #4

Use Case Name: Analyse preferences and generate Personalized Event Recommendations
Primary Actor: User
Stakeholders: Meet Up Application, Event Database
Brief Description: The system generates event recommendations based on the user's preferences.
Trigger: After the user updates or adds preferences to their profile, the user asks for an event recommendation.
Normal Flow of Events: 1. User adds or updates preferences in their account. 2. The system prompts if the user wants to see his recommended events 3. The user confirms and asks for the event recommendations 4. The system analyses the user preferences. 5. The system retrieves data from the Event Database. 6. The system generates a list of personalized event recommendations based on user preferences. 7. The recommendations are displayed to the user.
Exception Flow: No Events Found: The system notifies the user that no relevant events are currently available and asks to update the preferences

Use Case Description #5

Use Case Name: Display Event
Primary Actor: User
Stakeholders: Meet Up Application, Event Database
Brief Description: The system displays event details to the user.
Trigger: The user selects an event from the recommendations.
Normal Flow of Events: 1. The system generates the event recommendations based on user preferences 2. The system displays the list of events to the user. 3. The user selects an event from the list to view the details 4. The system displays the event name, time, location, and other information.
Exception Flow: Event Data Missing: The system displays an error if it fails to display the list of events and asks the user to input preferences once again

Use Case Description #6

Use Case Name: View Event Recommendations
Primary Actor: User
Stakeholders: Meet Up Application
Brief Description: The user views a list of personalized events based on their preferences.
Trigger: The user selects the "View Recommendations" option.
Normal Flow of Events: 1. The user navigates to the recommendations page. 2. The system retrieves the user's preferences and analyses event data. 3. The system displays a list of recommended events. 4. The user can explore individual event details.
Exception Flow: No Recommendations Found: The system informs the user that no matching events are currently available.

Use Case Description #7

Use Case Name: Finalize Event
Primary Actor: User
Stakeholders: Meet Up Application
Brief Description: The user confirms attendance at an event.
Trigger: The user decides to attend an event and selects the "Finalize Event" option.
Normal Flow of Events: 1. The user selects an event to finalize. 2. The system confirms the user's selection and updates the database. 3. The user is marked as attending the event.
Exception Flow: Event Full: The system notifies the user that the event has reached its capacity.

Use Case Description #8

Use Case Name: Generate Event Invitation
Primary Actor: User
Stakeholders: Offline User, Meet Up Application
Brief Description: The user generates an event invitation to share with online friends/followers as well as with ofline friends.
Trigger: The user selects an event and opts to generate an invitation.
Normal Flow of Events: 1. The user selects an event from the recommendations. 2. The system generates a shareable event invitation. 3. The user shares this invitation on social media platforms for inviting online friends/followers 4. System prompts if the user would like to send an event invitation to ofline friends via email or text message 5. The user provides the email address and/or phone number for sending an event invitation 6. The system sends an email and/or text message to the ofline friend based on the information provided by the user
Exception Flow: Failure to Generate Invitation: The system notifies the user and if it fails to send email or text messages to ofline friends.

Use Case Description #9

Use Case Name: Provide Feedback
Primary Actor: User
Stakeholders: Meet Up Application, Event Database
Brief Description: The user provides feedback on an event or a recommendation.
Trigger: The user does not like any event recommended by the system.
Normal Flow of Events: 1. System displays all the events based on the recommendation algorithm to the user 2. None of the events interests the user and he/she decides to provide feedback to the system 3. The user selects the feedback option. 4. The system prompts the user to provide a rating or comments. 5. The user submits the feedback. 6. The system stores the feedback in the Database.
Exception Flow: Incomplete Feedback: The system prompts the user to complete missing fields before submission.

Use Case Description #10

Use Case Name: Refine Recommendation Algorithm Based on Feedback
Primary Actor: System
Stakeholders: Meet Up Application, Event Database
Brief Description: The system improves the recommendation algorithm based on the feedback provided by the user.
Trigger: Feedback is added to the system.
Normal Flow of Events: 1. Feedback is submitted and stored in the Database. 2. The system analyses the feedback to identify trends or areas of improvement. 3. The recommendation algorithm is updated based on the analysis. **4.** The updated algorithm is applied for future recommendations.
Exception Flow: Insufficient Feedback: The system logs the issue and continues using the existing algorithm.

Data Dictionary

Use Case: Sign Up

user = email + password

Use Case: Log In

credentials = username + password

username = email

Use Case: Add Preferences

preferences = User profile + interests

User Profile = first name + last name + (phone number) + (birth date) + (gender) + (bio) + location

location = city + state + country + zip code

interests = food + arts + music + business + film + sports + health + science + travel + fashion

Use Case: Analyze Preferences and Generate Personalized Event Recommendations

preferences = profile + interests

User Profile = first name + last name + email + (phone number) + (birth date) + (gender) + (bio) + location

location = city + state + country

interests = food + arts + music + business + film + sports + health + science + travel + fashion

recommendations = event name + event time + event location + event details

event location = street + city + state + zip code

event details = event description + (organizer info) + (additional notes)

Use Case: Display Event

Recommendations = event name + event time + event location + event details

event location = street + city + state + zip code

event details = event description + (organizer info) + (additional notes)

Use Case: View Event Recommendations

recommendations = event name + event time + event location + event details

event location = street + city + state + zip code

event details = event description + (organizer info) + (additional note)

Use Case: Finalize Event

finalized event = event name + user confirmation

Use Case: Generate Event Invitation

invitation = event name + event time + event location + sharing details

sharing details = email addresses + phone numbers

Use Case: Provide Feedback

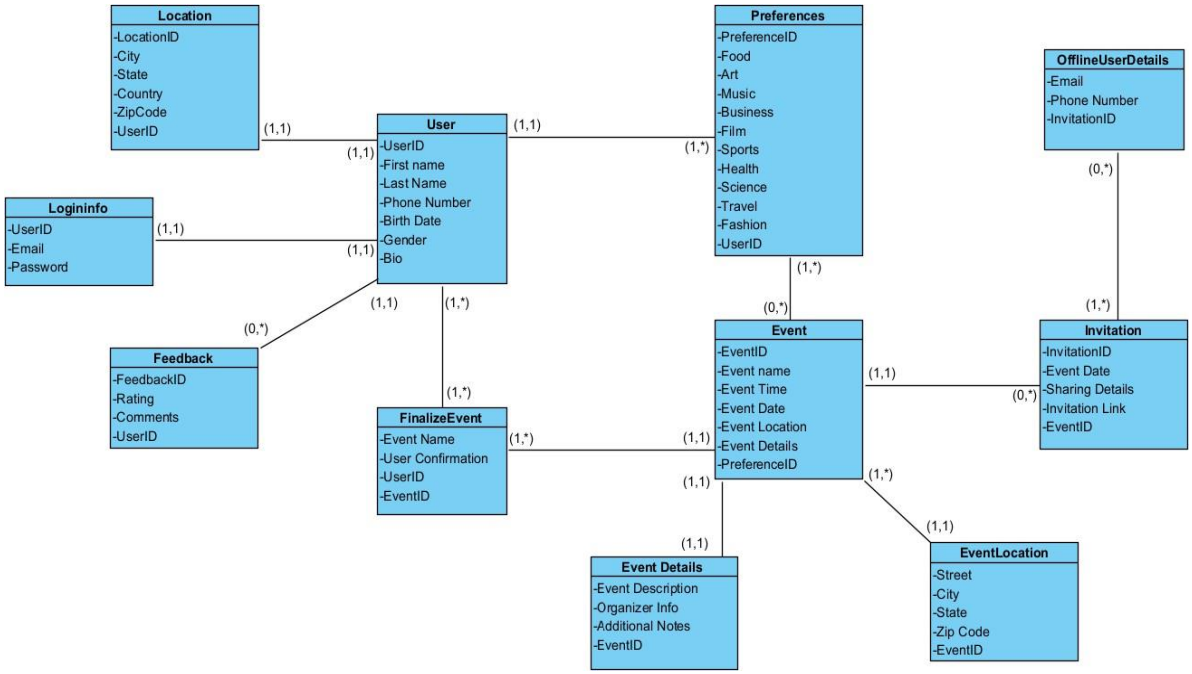
feedback = rating + comments + user-id

rating = [1|2|3|4|5]

Use Case: Refine Recommendation Algorithm Based on Feedback

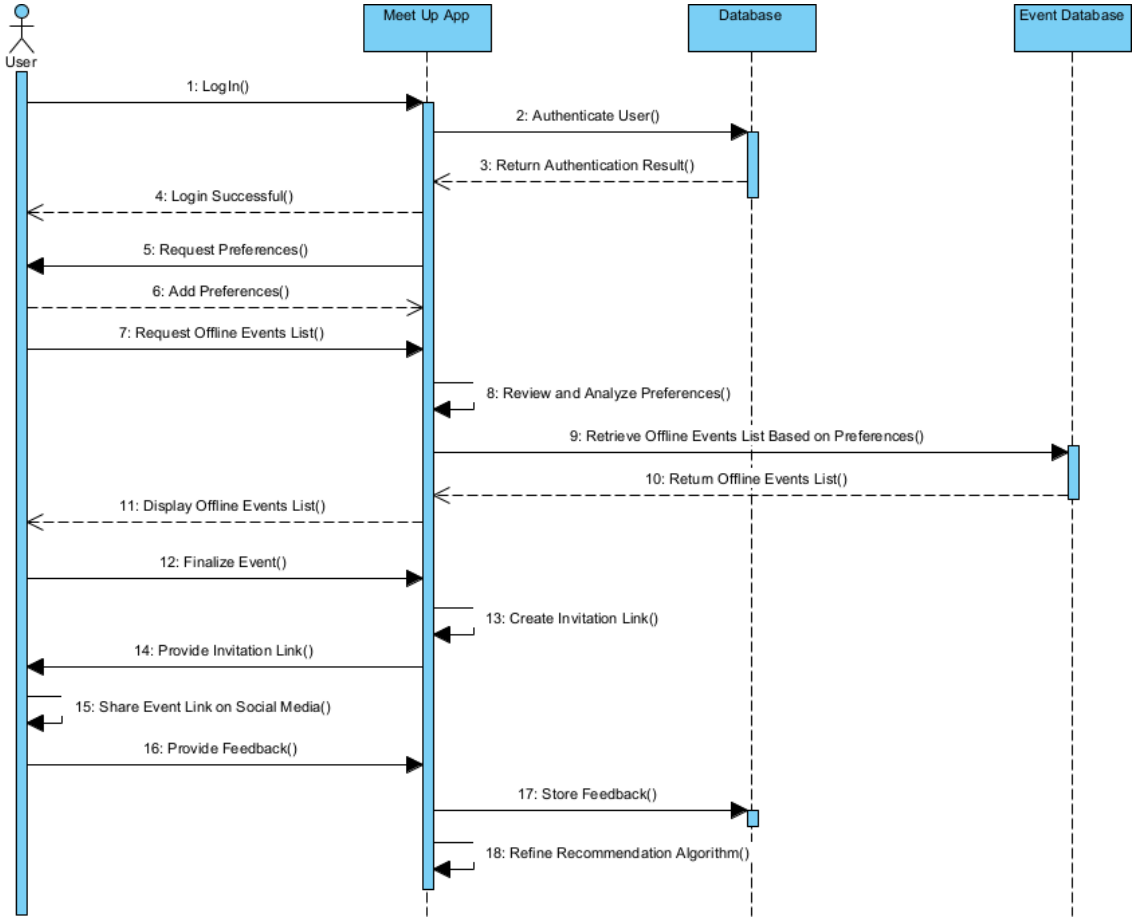
feedback = rating + comments + user-id

Class Diagram without Methods

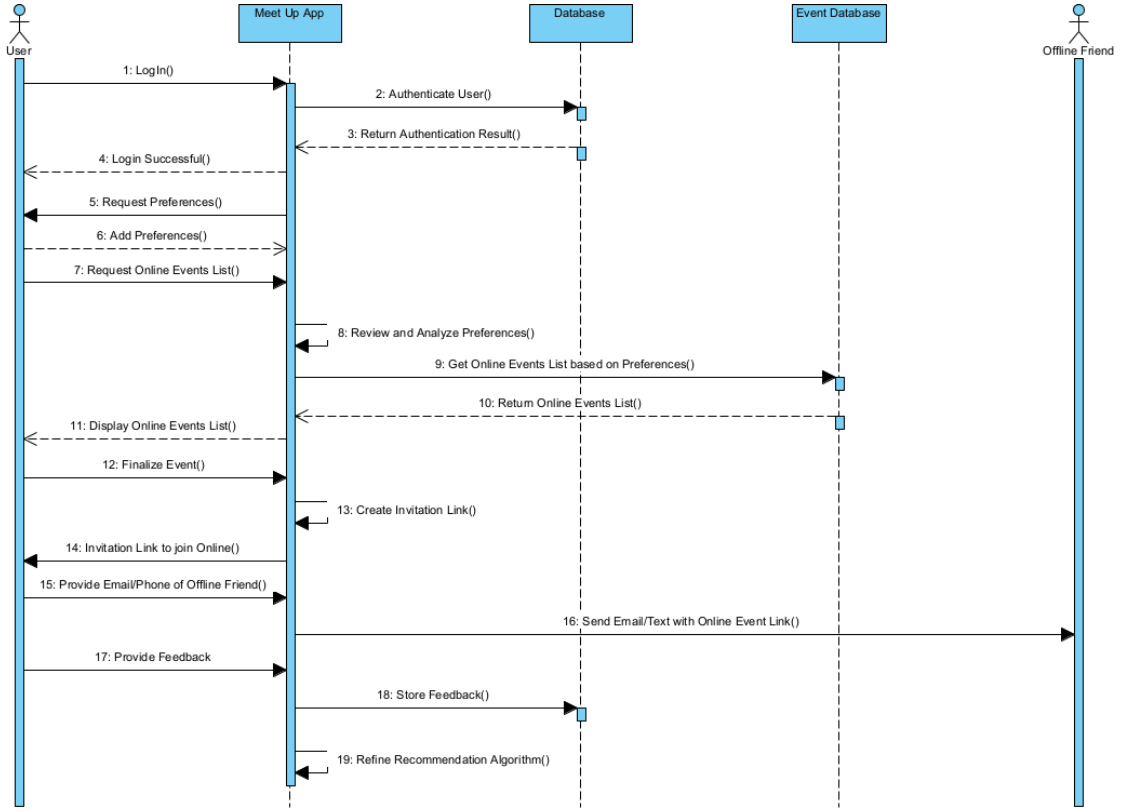


Sequence Diagram

Meet Online Friends/Mentors in an In-Person/Offline Event



Meet Offline Friend in an Online Event



Functional Specifications for the Proposed System

User Story 1: Sign Up

- **As a user, I want to sign up for the application.**
    - This step is required before accessing online/ofline events and other features of the system.

User Story 2: Sign In

- **As a user, I want to sign in to the application so I can access it with my account details.**
    - This corresponds to the Login and authentication steps in the sequence diagram.

User Story 3: Set Preferences

- **As a user, I want to set my preferences for online events.**
    - This step appears when the user sets preferences for event recommendations.

User Story 4: View Online Events

- **As a user, I want to view a list of online events based on my preferences.**
    - The system fetches and displays events according to the user's selected preferences.

User Story 5: Invite Friends to Online Events

- **As a user, I want to send invitations to my friends to join online events.**
    - Users can create and send event invitation links via email or text message to invite friends.

User Story 6: Provide Feedback for Online Events

- **As a user, I want to provide feedback about online events after participating.**
    - After an event, users are prompted to give feedback, which is stored for refining future recommendations.

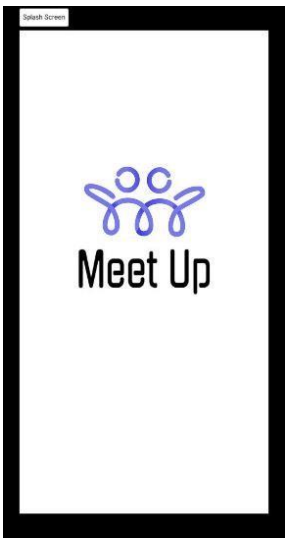
User Story 7: Invite Friends to Offline Events

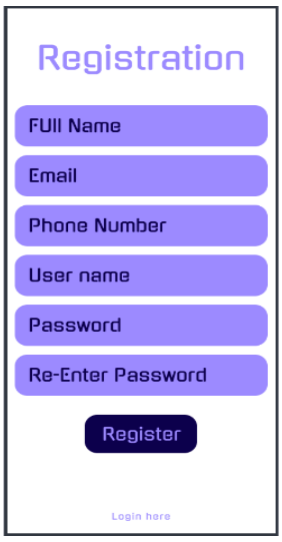
- **As a user, I want to send invitations to my friends to join offline events.**
    - Users can share event invitations with friends/mentors by sharing it on their social media platforms.

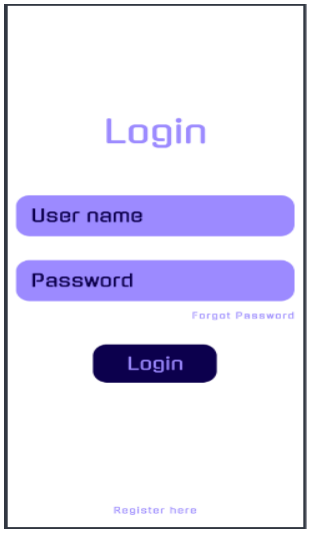
User Story 8: Provide Feedback for Offline Events

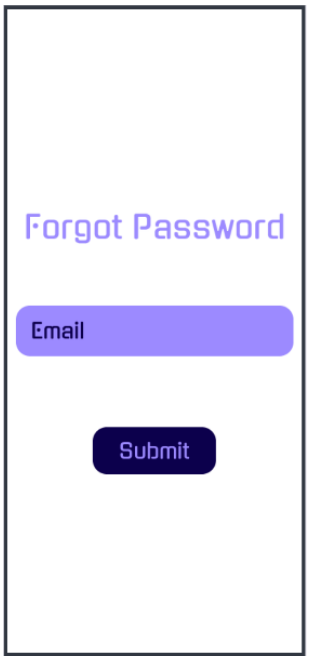
- **As a user, I want to provide feedback about offline events after participating.**
    - Similar to the online event feedback, users can provide ratings and comments about ofline events.

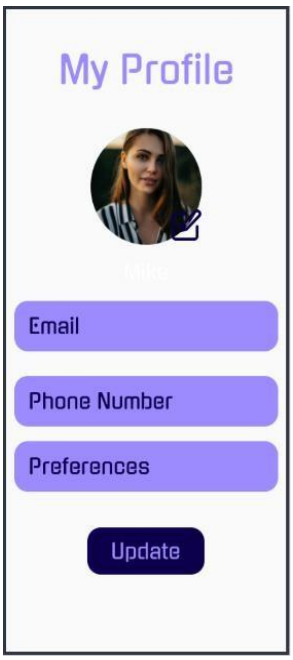
Interface Design

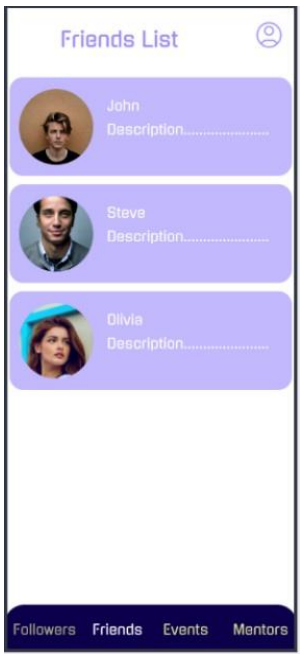




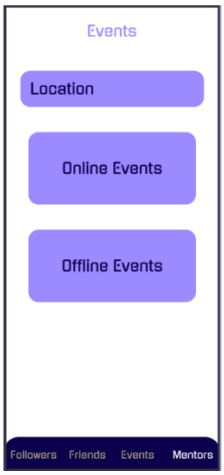


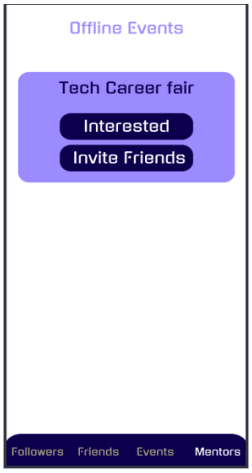







Tell your preference
Music
Movies
Food
Games
Submit
Followers Friends Events Mentors

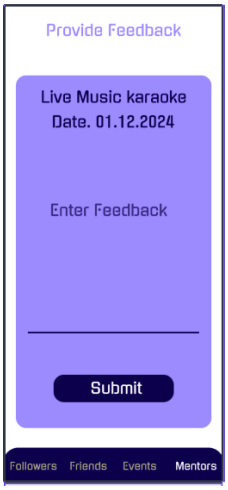

Events
Location
Online Events
Offline Events
Followers Friends Events Mentors


Offline Events
Tech Career fair
Interested
Invite Friends
Followers Friends Events Mentors

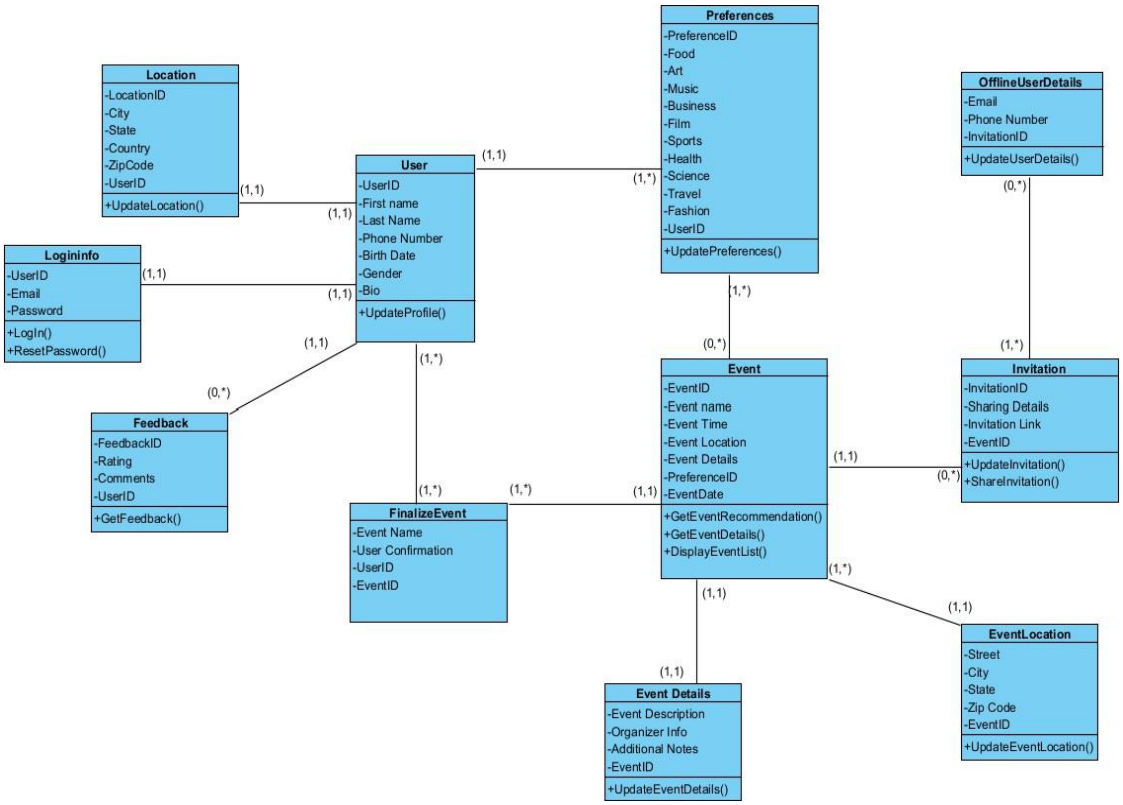

Online Events
Live Music karaoke
Interested
Invite Friends
Webinar
Interested
Invite Friends
Followers Friends Events Mentors


Interested Events
Live Music karaoke
Date . 01.12.2024
Enter Feedback
Submit
Followers Friends Events Mentors


Invite Offline friends
Email
Phone
Invite
Followers Friends Events Mentors

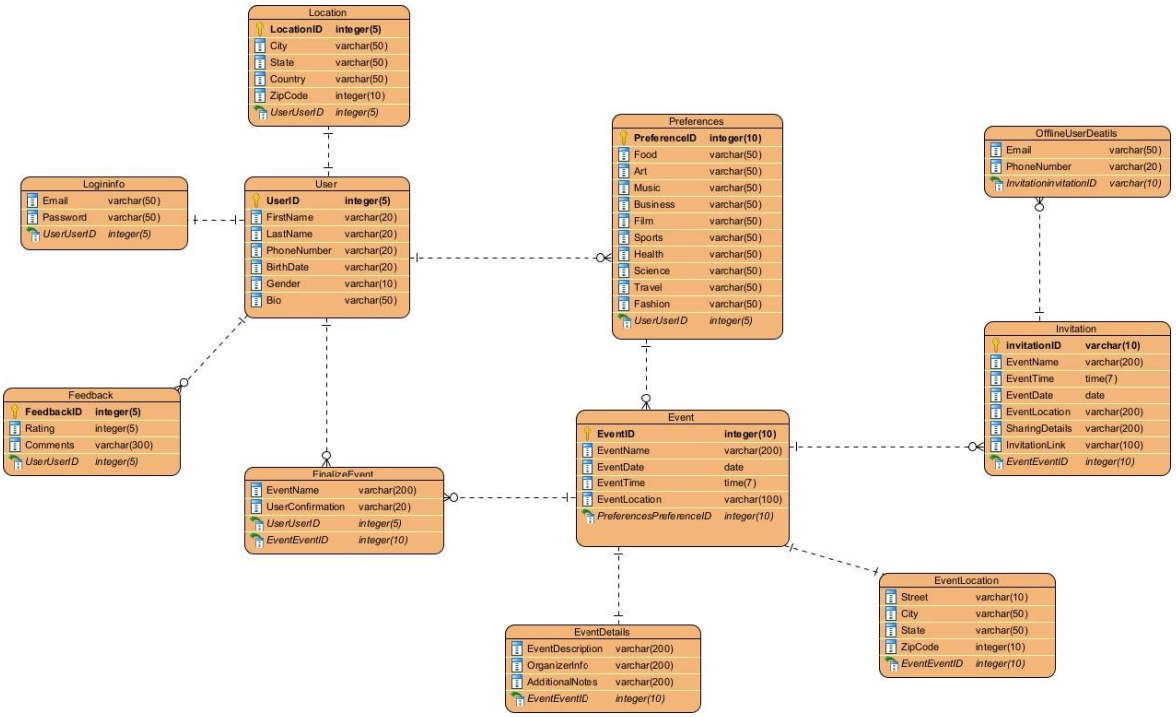

Provide Feedback
Live Music karaoke
Date. 01.12.2024
Enter Feedback
Submit
Followers Friends Events Mentors

Class Diagram with Method



Database Design

Entity – Relationship Diagram



Database Constraints:

User Table

- Primary Key: UserID
- Not Null Constraints: FirstName, LastName

Logininfo Table

- Not Null Constraints: Email, Password
- Foreign Key: UserID

Location Table

- Primary Key: LocationID
- Not Null Constraints: City, State, Country, ZipCode
- Foreign Key: UserID

Preferences Table

- Primary Key: PreferenceID
- Foreign Key: UserID

Event Table

- Primary Key: EventID
- Not Null Constraints: EventName, EventDate, EventTime, EventLocation
- Foreign Key: PreferenceID

EventDetails Table

- Not Null Constraints: EventDescription
- Foreign Key: EventID

Invitation Table

- Primary Key: InvitationID
- Not Null Constraints: EventName, EventTime, EventDate, EventLocation, SharingDEtails, InvitationLink
- Foreign Key: EventID

EventLocation Table

- Not Null Constraints: Street, City, State, ZipCode
- Foreign Key: EventID

OfflineUserDetails Table

- Not Null Constraints: Email, PhoneNumber
- Foreign Key: InvitationID

FinalizeEvent Table

- Not Null Constraints: EventName, UserConfirmation
- Foreign Key: UserID, EventID

Feedback Table

- Primary Key: FeedbackID
- Foreign Key: UserID

Software Design

Method Name: userSignUp()
Class Name: Sign Up
ID: #1
Clients: Users
Associated Use Case(s): Sign Up
Description of Responsibilities: The user can register into the application by entering all details.
Arguments Received: Email, Password, and re-enter password
Type of Value Returned: True or False
Pre-Condition(s): The user should enter all the fields.
Post-Condition(s): The user successfully registered message will be shown

Method SignUp(Email,Password)

Get Email, Password
Then
Return True
Message: user Registered successfully
Return False
Message: user already exists, please register with new email.
End Method

Method Name: LogIn()
Class Name: Login
ID: #2
Clients: users
Associated Use Case(s): Log In
Description of Responsibilities: user will enter into the application using login credentials that are given while signing up.
Arguments Received: email and password
Type of Value Returned: True or False
Pre-Condition(s): user should give valid credentials.
Post-Condition(s): user enters into application.

Method userLogin(email, password)

Get Password for provided Email from the Database

If Password equals provided passwordMethod

Then

Message: user Login successful

Return True

Else

Message: Invalid Details please try again

Return False

End Method

Method Name: AddPreferences()
Class Name: Preferences
ID: #3
Clients: Users
Associated Use Case(s): Storing user preferences
Arguments: userID: Unique identifier for the user preferencesList: List of user preferences
Type of Value Returned: Boolean
Responsibilities: Display options for user preferences.

Save preferences for personalized experiences.
Pre-Condition: User is logged in.

Method getPreferences(UserID, PreferenceList)
If PreferencesList is empty
Return False
Message: "Please select at least one preference."
Else
Save the PreferencesList to the database for the given UserID
If Save operation is successful
Then
Return True
Message: "Preferences saved successfully."
Else
Return False
Message: "Error: Unable to save preferences. Try again."
End If

End Method

Method Name: GetEvents()
Class Name: Events
ID: #4
Clients: Users
Associated Use Case(s): Analyze Preferences and Generate Personalized Event Recommendations
Arguments: Interests(e.g., Food, Arts, Music, etc) Profile: Name, Gender, Location
Type of Value Returned: List
Responsibilities: Display events that match the user preferences
Pre-Condition: User is logged in. User has provided his/her preferences.
Post-Condition: Event list is displayed to the user and asked to choose for interested event.

Method: getEvents(Profile, Preferences)
Analyze events and query database for events matching with preferences

If relevant events found

Return events List

Else

Message: “No Relevant events found. Update your Preferences”

End Method

Method Name: GenerateInvitation()
Class Name: Invitation
ID: #5
Clients: Users
Associated Use Case(s): Send Invite to Ofline Friends
Arguments: eventId: Unique identifier of the event Email: Email ID of ofline user Phone: Phone number of the ofline user
Type of Value Returned: Boolean (True if invitation sent successfully, otherwise False)
Responsibilities: Display an input field for the Email ID and phone number of the ofline user.
Pre-Condition: User is logged in and have finalized the event to attend
Post-Condition: Invitation sent successfully.

Method: GenerateInvitation(UserID, EventID, Email, Phone)

If Email, Phone is empty

Message: "Email, Phone cannot be empty."

Return False

Else

Get User details from Usertable based on UserID

Get Event details from Event table based on EventID

Generate an email and text message to be sent

Initiate an email, text to the ofline user

If email and text sent successfully

Message: “Invitation sent successfully”

Return True

Else

Return False

End Method

Method Name: ProvideFeedback()
Class Name: Feedback
ID: #6
Clients: Users
Associated Use Case(s): Providing Feedback
Arguments: eventId: Unique identifier of the event feedbackText: User-provided feedback
Type of Value Returned: Boolean (True if feedback is successfully submitted, otherwise False)
Responsibilities: Display an input field for the feedback. Submit feedback to the backend.
Pre-Condition: User is logged in
Post-Condition: Feedback saved in the database.

```
Method: submitFeedback(UserID, FeedbackText)
 Get UserID, FeedbackText
        If FeedbackText is empty
                Return False
                Message: "Feedback cannot be empty."
        Else
                Insert the FeedbackText into the Feedback table UserID
                Return True
                Message: "Feedback submitted successfully."
End Method
```