Executive Summary

In today's digital age, individuals maintain parallel lives—a virtual existence in social media populated with online connections such as friends, mentors, and followers, alongside similar relations in the tangible physical world. However, these two realms often lack proper integration. Our project aims to bridge this gap by using user information, including interests, friends, GPS location, and booking plans, to establish connections between users in both the virtual and physical spheres.

Our proposed solution to connect users virtually and physically involves utilizing user interests and booking plans. The system extracts users' booking plans from their emails and searches a database to identify connections living nearby or planning to attend similar events at the same time and location. If a match is found, the system, in accordance with the user's privacy settings, sends recommendations to facilitate a physical connection between these virtually linked users.

Furthermore, our project proposes an additional solution to foster virtual-to-physical connections based on user interests. The system allows businesses to share event details and relevant interest tags. By matching users' interests with these events, the system sends event recommendations. Additionally, if any of the user's connections plan to attend events aligned with the user's interests, the system sends event recommendations along with information about their connections attending those events.

To establish connections from the physical to the virtual world, our solution utilizes the user's GPS location to send potential connection recommendations based on their current location and time.

These proposed solutions aim to break down the barriers between users' virtual and physical lives, empowering them to maintain both their online and offline connections.

Problem Statement:

In the current times, there is a lack of proper physical connection between virtual users and similarly users who are aware of each other physically but not connected virtually.

Project Objectives:

- **Virtual to Physical:**
  Develop a system to connect users physically who are virtually connected using booking details in the user email and interests of the user.
- **Physical to Virtual:**
  Develop a system to connect users virtually who knows each other in the physical world but not connected virtually using GPS location.

Data Collection:

- The system reads user emails every 15 minutes and extracts information such as time, duration, dates, number of people, place which are related to the booking details such as flight itinerary, restaurant appointments, event registrations and other relevant booking information.
- The system tracks user GPS location and user movements.
- Event manager provides information and other attributes of the events.

Proposed Solution:

- **Virtual to Physical:**

  Process 1:
  From the booking information, which is extracted from user emails, the system will check for other users who are currently or plan to be in the same location at the same time. If the system finds another user, then the system will notify both the users about the other user's presence and suggest to them to message them and meet physically.

  Process 2:
  - ➢ Using the event information details provided by the event manager, the system checks for users who have similar interests and suggests them regarding the event details.
  - ➢ If the user books any event, then process 1 will be repeated.

  Process 3:
  Users can directly message to the other virtual user to connect physically.
- **Physical to Virtual:**

  Process 1:
  The system uses user location to connect people who know each other but are not connected virtually. System tracks the user movements, GPS location and time. If a user is in the same location for more than 10 minutes the system checks for other

users who are in the same location for more than 10 minutes at the same time. If system finds other user, then system checks whether the other user is already virtually connected, if they are not virtually connected then system considers the possibility that they may know each other, and system suggest the users to connect them virtually.

Assumptions:

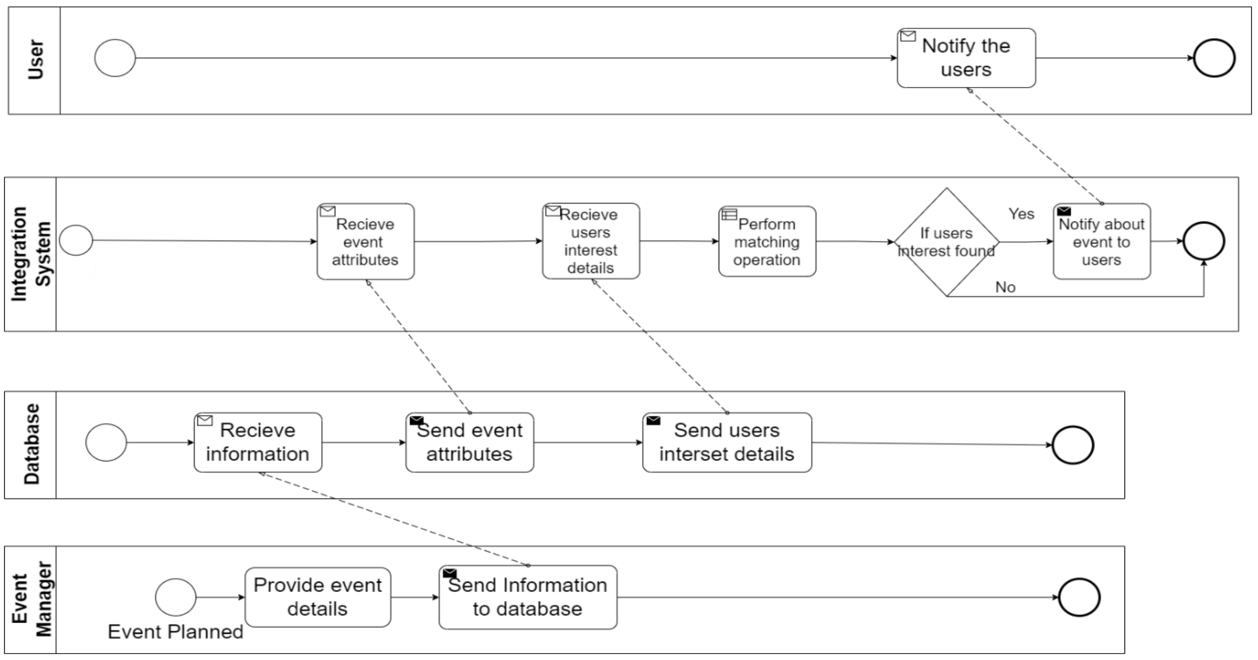
- System has permission from the user to access and read their emails.
- The system has permission from the user to track user movements and GPS location.
- Users adhere to community guidelines and privacy settings.
- The system reads user email every 15 minutes.
- System facilitates connecting the users who are currently in the system.
- The user has provided his interests to the system.
- System has access to user contacts and has permission to access information of other social media account of users.

Out of Scope:

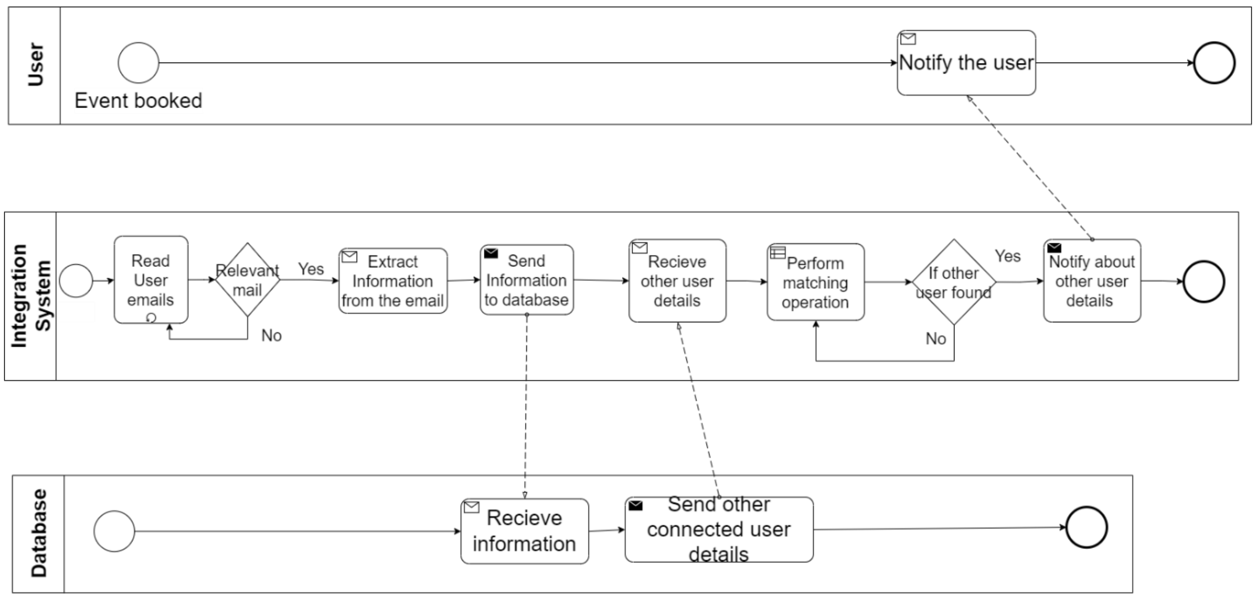
- Develop business algorithm rules for the system.
- Obtain user permission for GPS tracking and read user emails.
- Develop user login and signup system.
- Obtain user interest details.

Business Process Modelling Notation (BPMN) model:

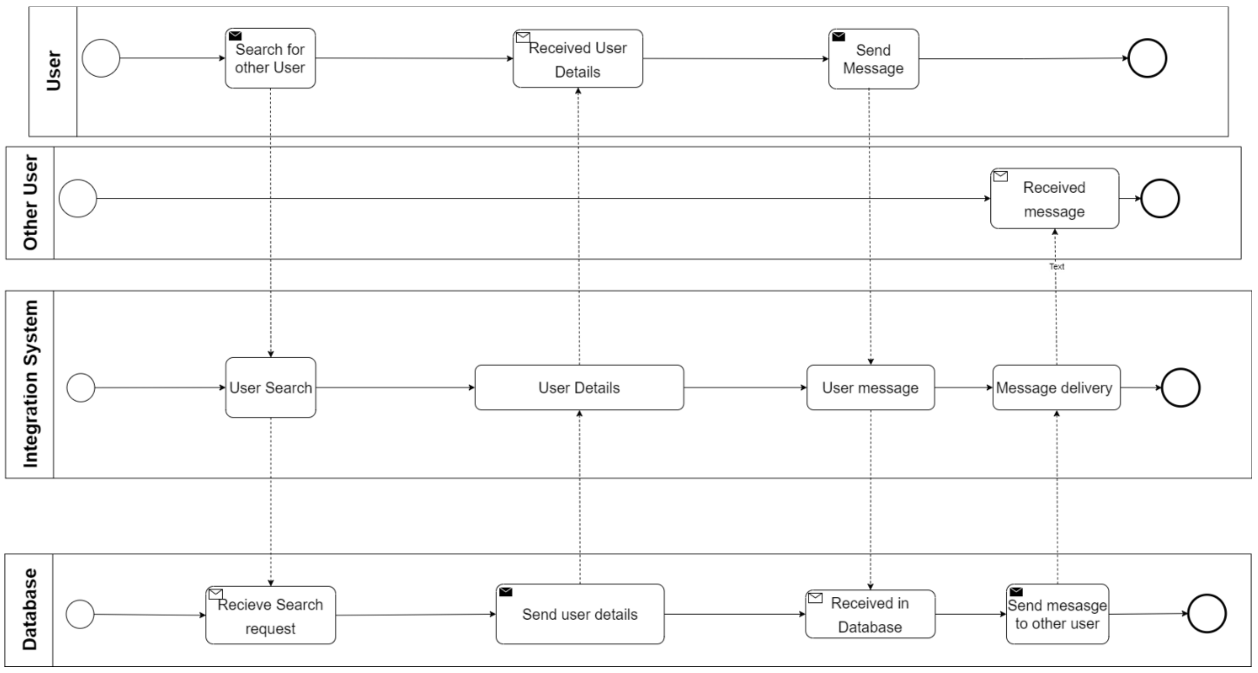
1. **Integration of Virtual to Physical life based on planned events and user intersets.**



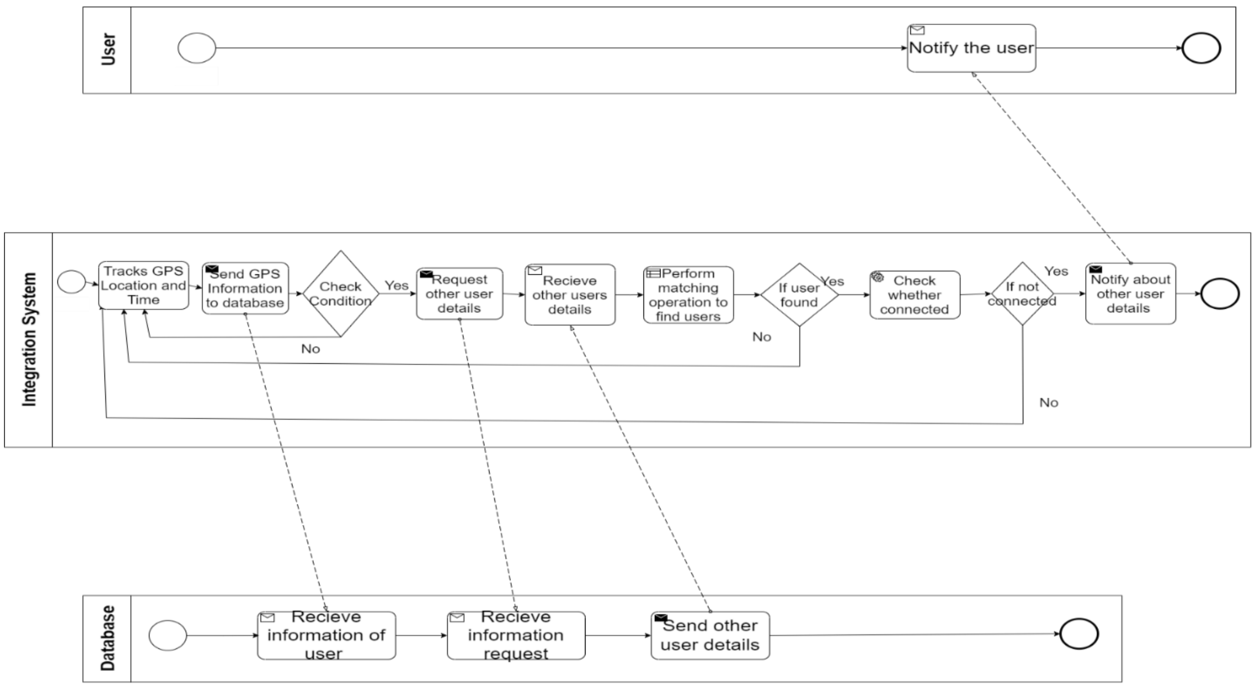
2. **Integration of Virtual to Physical life based on user booking details.**


User
Event booked
Notify the user
Integration System
Read User emails
Relevant mail
Yes
No
Extract Information from the email
Send Information to database
Recieve other user details
Perform matching operation
If other user found
Yes
No
Notify about other user details
Database
Recieve information
Send other connected user details

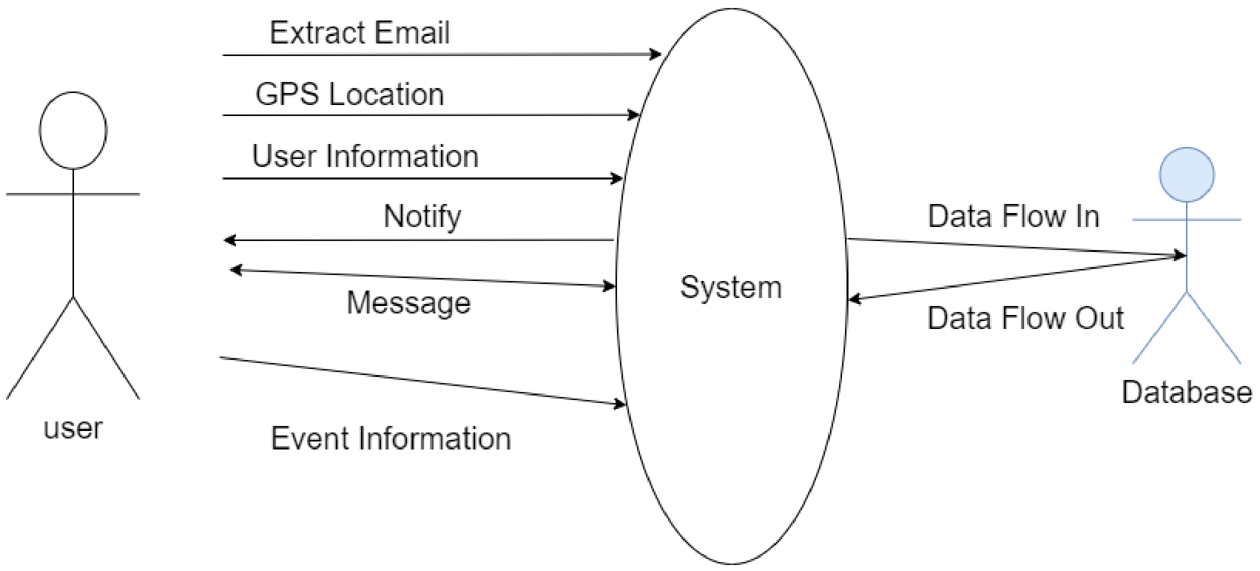
3. **Integration of Virtual to Physical life based on sharing messages.**



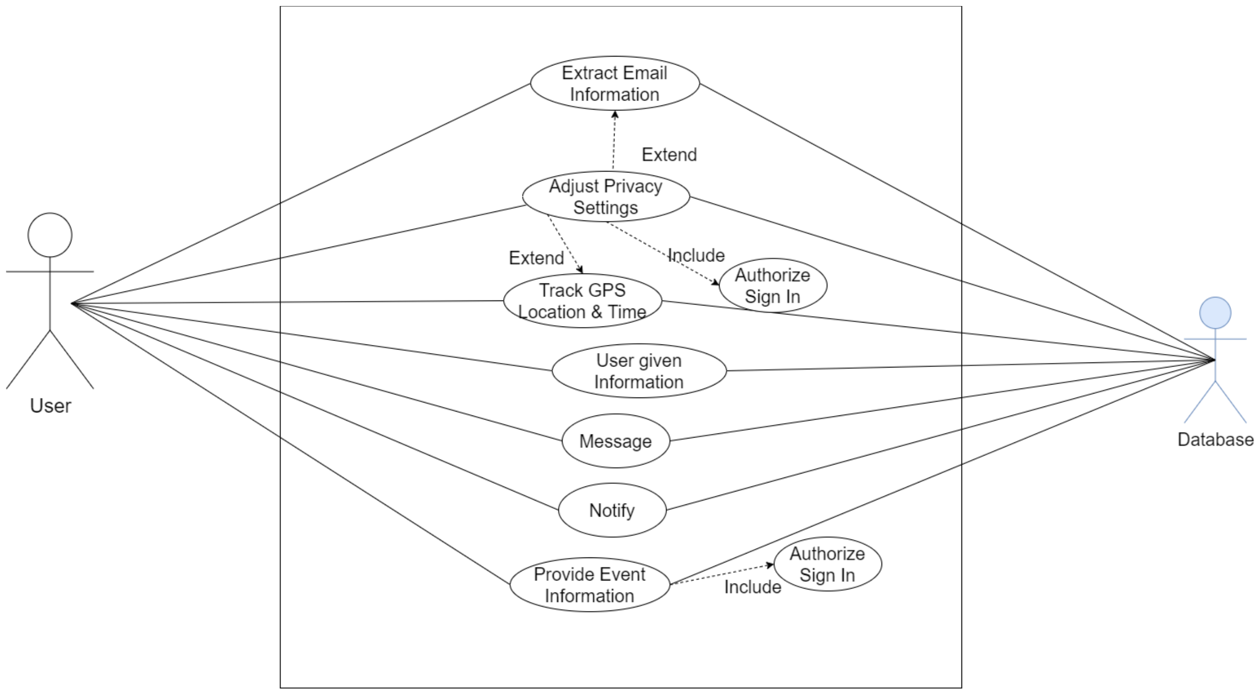
4. Integration of Physical to Virtual life based on user location.


User
Notify the user
Integration System
Tracks GPS Location and Time
Send GPS Information to database
Check Condition
Yes
No
Request other user details
Recieve other users details
Perform matching operation to find users
If user found
Yes
No
Check whether connected
If not connected
Yes
No
Notify about other user details
Database
Recieve information of user
Recieve information request
Send other user details

Context Diagram:



Use Case Diagram:



Use Case Description

Use Case Description 1:

Use Case Name:	Information retrieval from email.
Primary Actor:	User
Stakeholders:	Database
Brief Description:	Retrieving relevant booking information from email.
Trigger:	Read user email for booking details for every 15 mins.
Normal Flow of Events:	1) The system reads user email every 15 minutes. 2) The system checks whether the email pertains to bookings. 3) If email pertains to booking, then retrieve the booking information such as venue, dates, time, duration, number of people, destination address from the email. 4) Write retrieved information to the Database. 5) Execute << Notify user about other user details>> use case.
Exception Flow of Events:	3.1) If the email doesn't pertain to the booking, then repeat from step 1.

Use Case Description 2:

Use Case Name:	Information provided by user.
Primary Actor:	User
Stakeholders:	Database
Brief Description:	User providing the booking information.
Trigger:	User
Normal Flow of Events:	1) User provides the information of the event which was planned. 2) Write this information in Database. 3) Execute << Notify user about other user details>> use case.

Use Case Description 3:

Use Case Name:	Information provided by Event Manager.
Primary Actor:	Event Manager.
Stakeholders:	Database
Brief Description:	Event Organizer provides the information about event.
Trigger:	Event planned
Normal Flow of Events:	1) Event Organizer signs in. 2) Event Organizer provides information about the event and provides <u>event attributes</u> which are used to match with user interests. 3) Write the <u>event details</u> in the database. 4) Execute << Notifications to user based on user interest >> use case.

Use Case Description 4:

Use Case Name:	Notify user based on user interests
Primary Actor:	Database
Stakeholders:	User
Brief Description:	Notifications to user based on user interest
Trigger:	Event Organizer provided the event details
Normal Flow of Events:	1) Retrieve <u>event attributes</u> from the details provided by the business user. 2) Check for user who has similar <u>interests</u> as that of event attributes. 3) If event attributes == user interest, then notify user about event.

Use Case Description 5:

Use Case Name:	Notify user about other user details
Primary Actor:	Database
Stakeholders:	User
Brief Description:	Notifications to user about other user details (Virtual to Physical)
Trigger:	Event Information
Normal Flow of Events:	1) From the retrieved information from user's email, System searches for other users in the database with similar <u>bookings</u> or in the same nearby <u>locations</u>. 2) If system founds another user, then notify user about other user details.
Exception Flow of Events:	2.1) If system doesn't find another user, then repeat step 1 for every 15 minutes till the expiry of the event date.

Use Case Description 6:

Use Case Name:	Notify user based on GPS
Primary Actor:	Database
Stakeholders:	User, Other Users
Brief Description:	Notifications to user about other user details (Physical to Virtual)
Trigger:	GPS tracked
Normal Flow of Events:	1) The system tracks the user's physical moments by collection of the user's <u>GPS co-ordinates</u> and <u>timestamp</u> details. 2) System sends <u>GPS co-ordinates</u> details to the database. System checks the following condition 'Check condition': If <u>the GPS co-ordinates</u> of user doesn't change for more than 10 minutes then system requests database for other user details who are in the same location and satisfy the same 'Check condition' (i.e doesn't change for more than 10 minutes). 3) Perform matching operation to find other users by equating <u>GPS co-ordinates</u>. 4) If the system finds the other user, then check whether they are connected. 5) If the users are not connected, then send notification to the user about other users.

Exception Flow of Events:	3.1) If the system doesn't satisfy 'Check condition' then start step 1. 4.1) If the system doesn't find another user, then start step 1. 5.1) If the users are connected, then start step 1.

Use Case Description 7:

Use Case Name:	Message between users
Primary Actor:	User
Stakeholders:	Database
Brief Description:	User messaging directly to another connected user
Trigger:	Message from a user
Normal Flow of Events:	1) User searches for another connected user. 2) System extracts other user details from database. 3) System provides other user details to the user. 4) User sends message. 5) System sends the message to database. 6) System stores the message in the database. 7) The system sends the message from the database to another user. 8) Another user receives the message.

Data Dictionary:

user = user_id + first_name + last_name + email_id + user_phone_no + {friend} + address + {event_details} + {interests}

event_details = event_name + address + details + event_date

address = street + city + state + zip_code + country

general_user = user + age + *recent_location*

business_user = user + business_type

business_type = [restaurant | event | hotel | cinema theatre | travel | recreational_activities]

address = street + city + state + zip_code + country

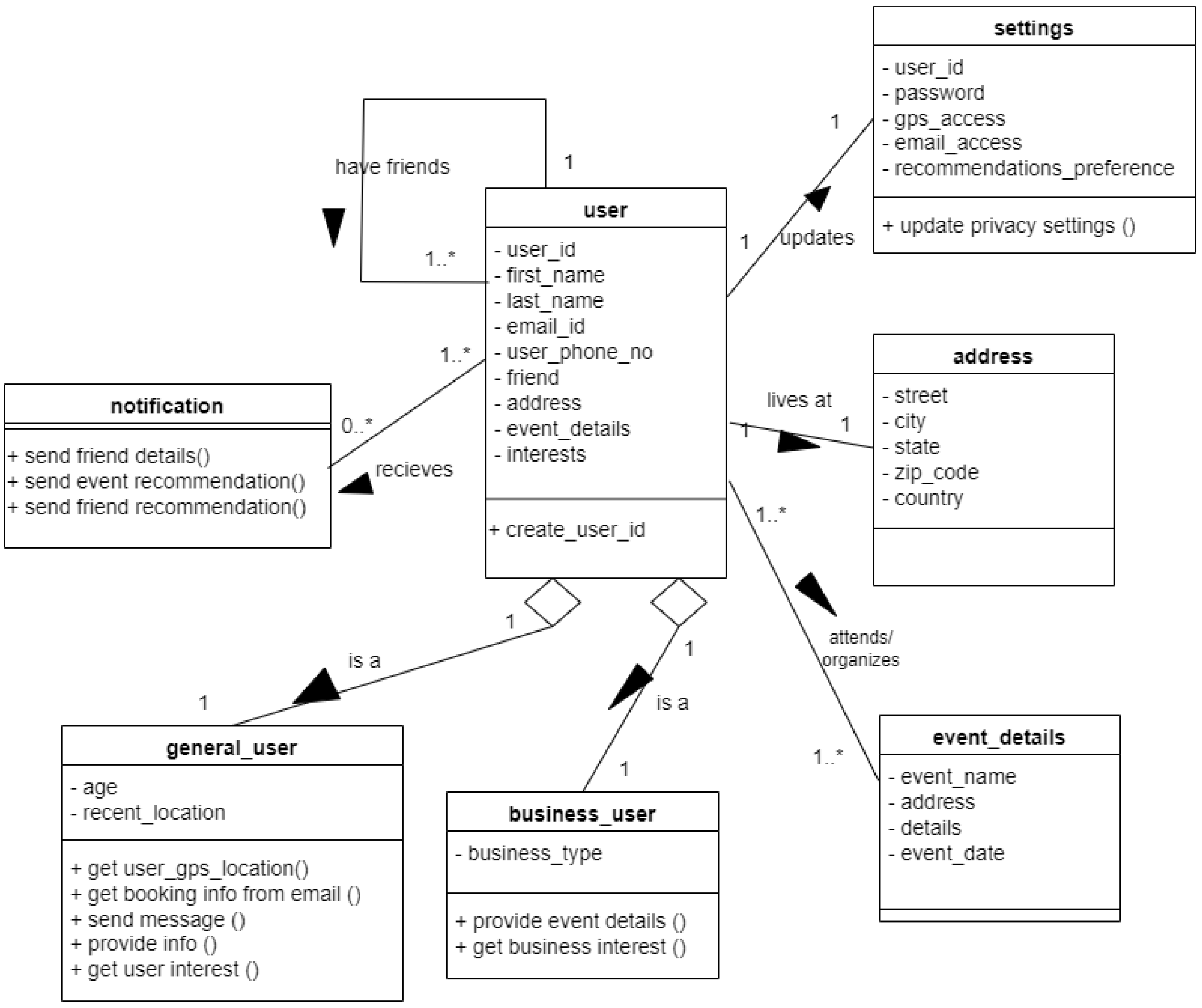
settings = user_id + password + email_access + gps_access + recommendations_preference

email_access = [yes | no]

gps_access = [yes | no]

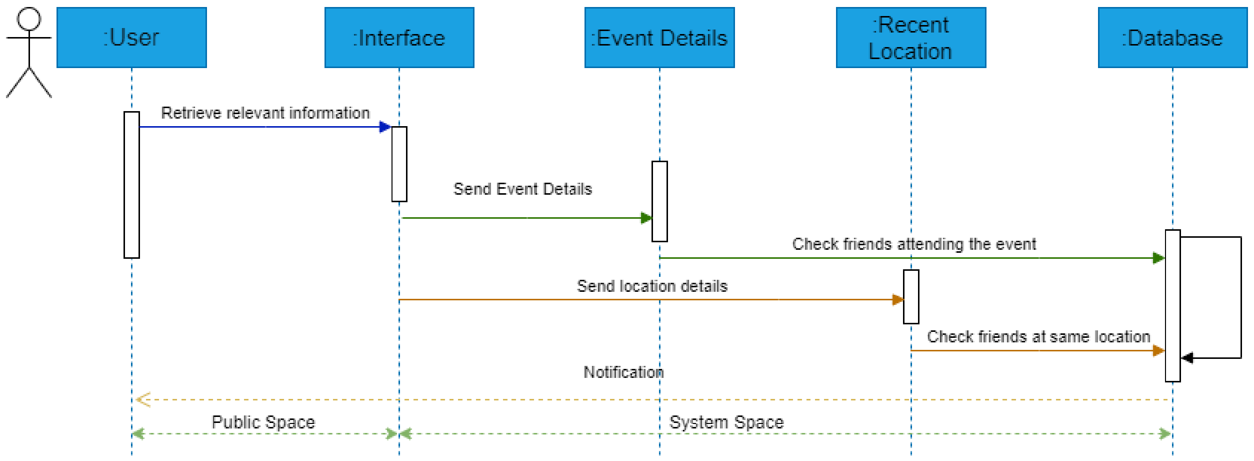
recommendations_preference = [yes | no]

Data Model - Class Diagram:

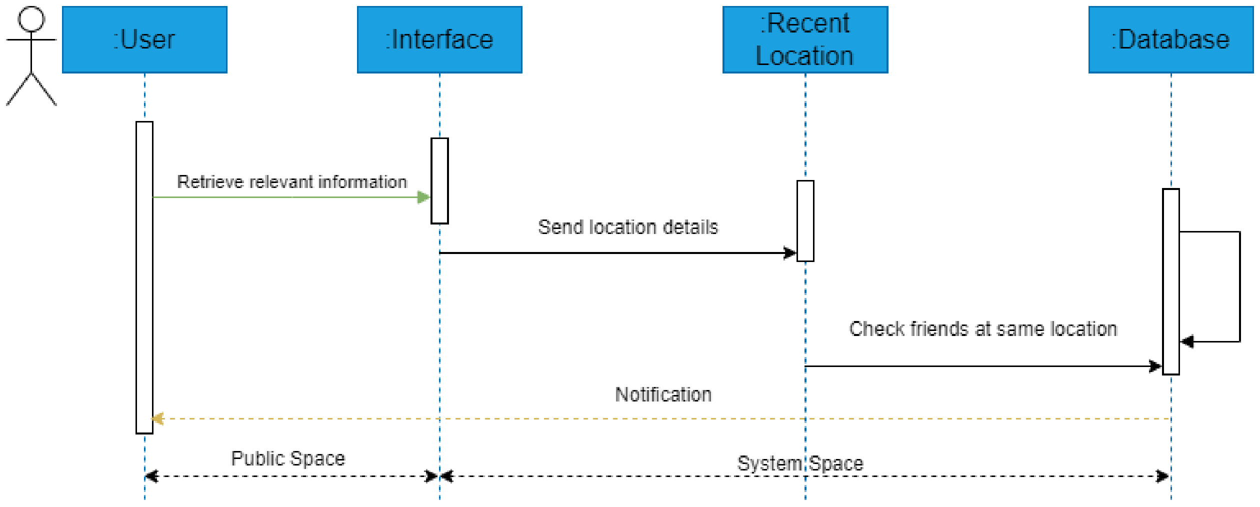


Sequence Diagram:

Sequence Diagram for the major Use Case of virtual to physical integration:



Sequence Diagram for the major Use Case of physical to virtual integration:



Functional Specification:

Authorize Sign In:

The system will allow users to register and create their profiles.

The system will allow users to adjust their privacy settings such as email access, location access, recommendations.

Extract Email Information:

The system extracts booking information from the user's email and processes the information to provide the user with user’s connection details for physical connections.

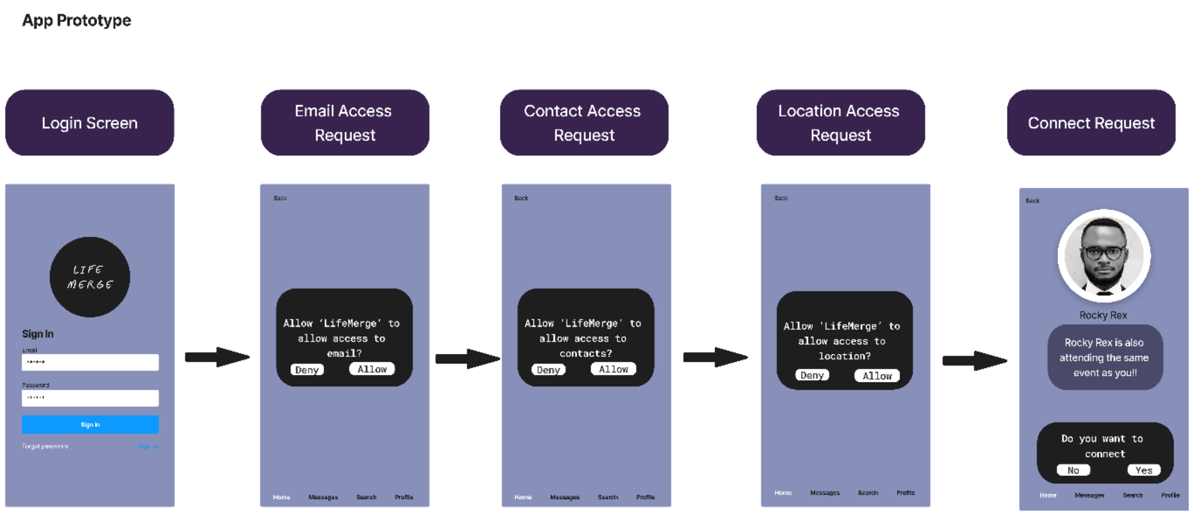
Track GPS Location & Time:

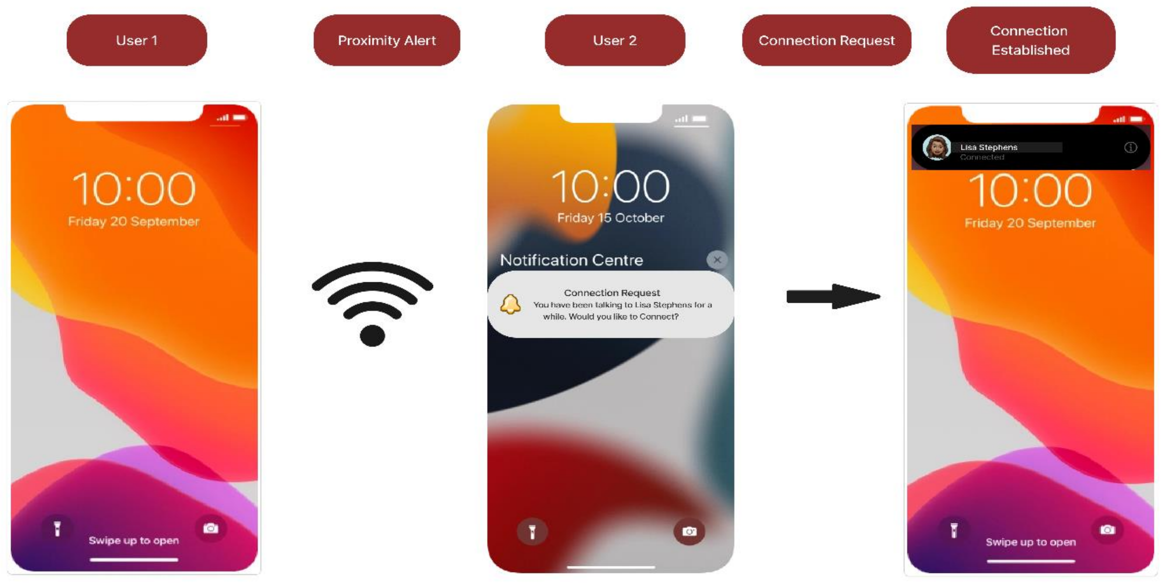
The system accesses the user’s GPS co-ordinates information and real-time location to provide the potential connection recommendation to the user.

Provide Event Information

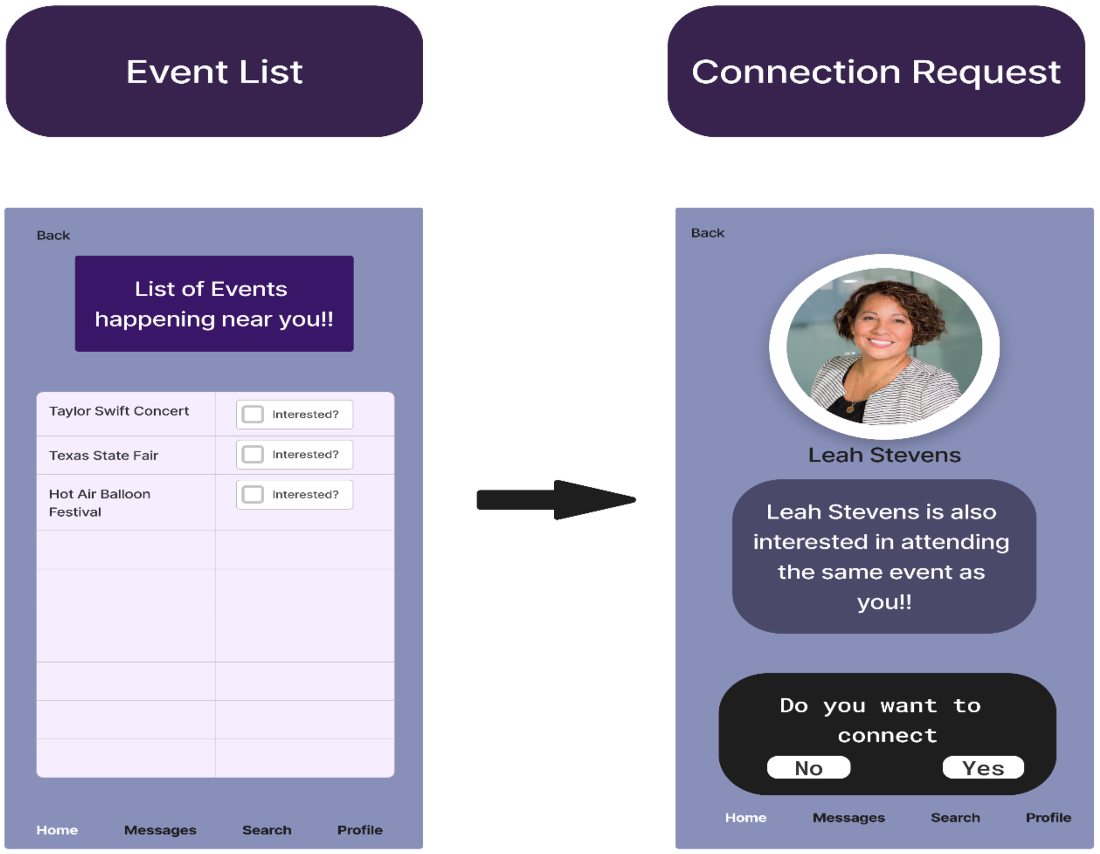
The system takes event information from business users and based on general user interests it provides event information to the users.

Interface Design:





Connection Request
You have been talking to Lisa Stephens for a while. Would you like to Connect?



List of Events happening near you!!

Taylor Swift Concert	☐ Interested?
Texas State Fair	☐ Interested?
Hot Air Balloon Festival	☐ Interested?

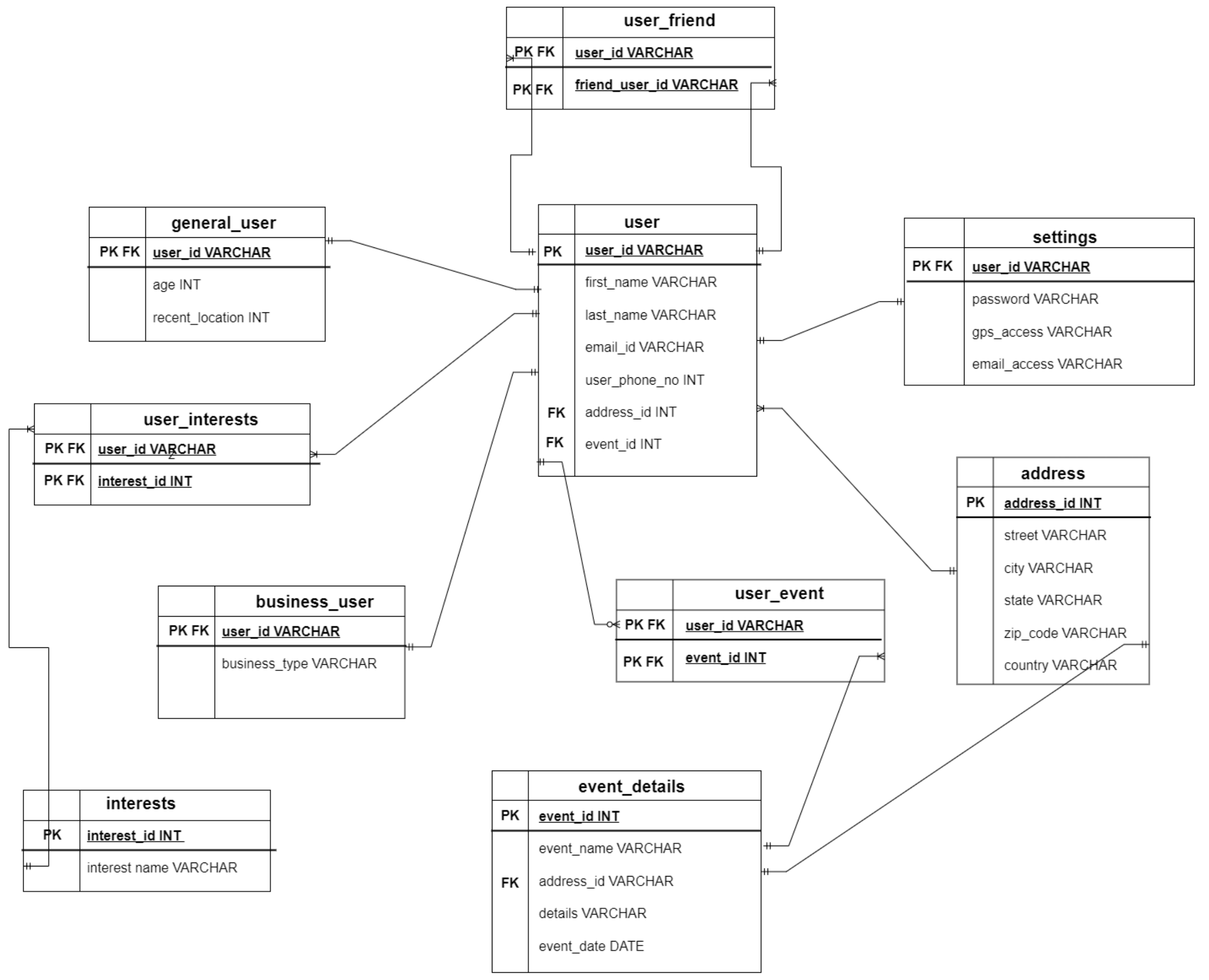
Leah Stevens

Leah Stevens is also interested in attending the same event as you!!

Do you want to connect

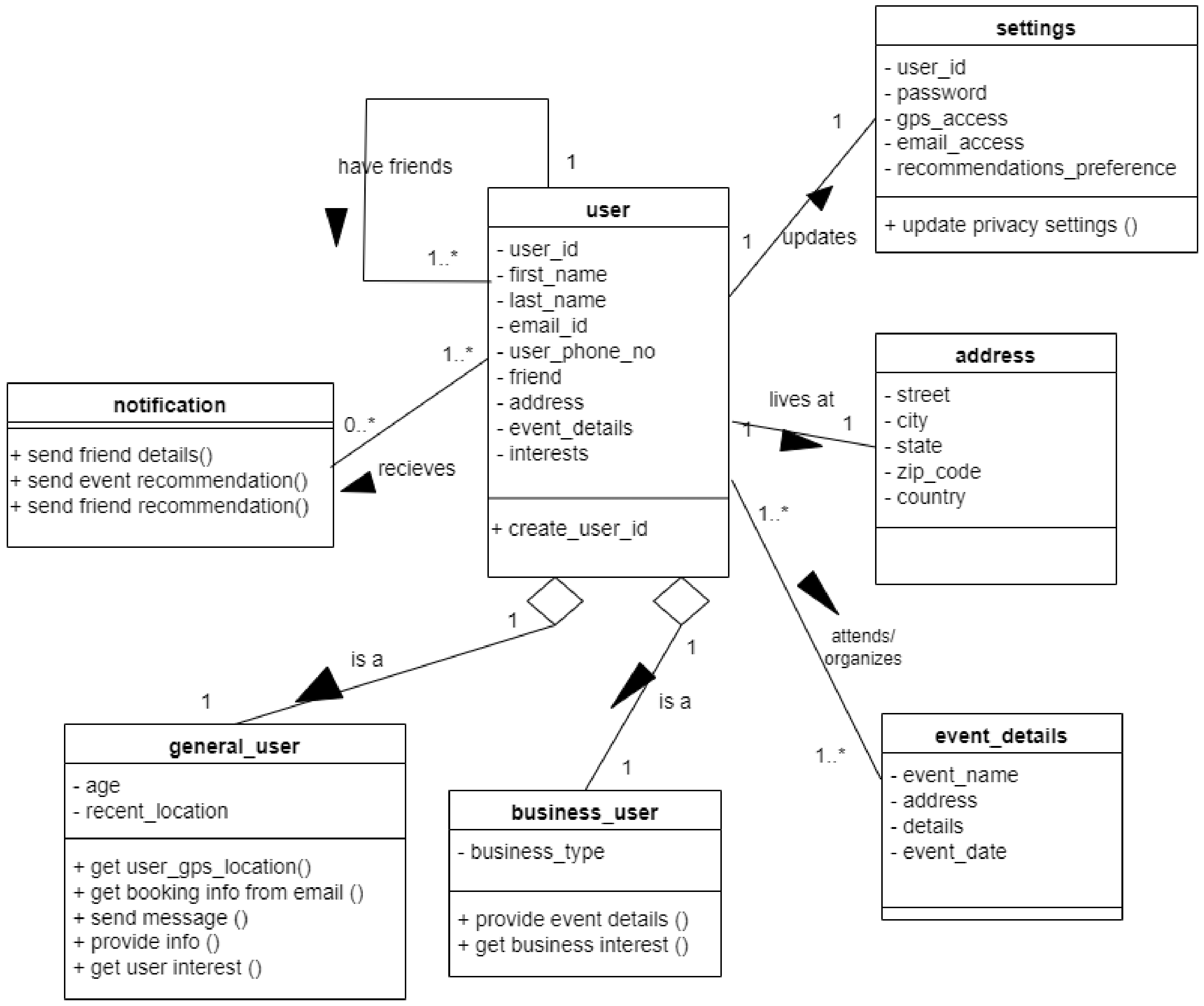
No Yes

Database Design:



All Primary key variables are non-null and has integrity constraint, details of cardinality are provided in the above diagram.

Complete Class Diagram:



Software Design:

Contract 1:

Method Name: Send friend recommendation.

Class Name: Notification

ID: 1

Clients (Consumers): Main method

Associated Use Cases: Notify user *based on* GPS.

Description of Responsibilities: Send friend recommendation based on GPS.

Arguments Received: GPS location and timestamp.

Type of Value Returned: User ID.

Pre-Conditions: General User needs to be signed in and have access to GPS and GPS is on.

Post-Conditions: User received potential friend recommendation.

LOGIC:

```
The system continually collects the GPS coordinates and timestamp of the user's location.

IF the user's GPS coordinates haven't changed for more than 10 minutes.

        Then the system sends the GPS coordinates to the database.

        The system searches the database for details of other users who are in the same location
        and haven't moved for more than 10 minutes at the same timestamp.

        IF the system finds another user AND users are not connected:

                Then the system sends a friend recommendation to the users.

        ENDIF

ENDIF
```

Contract 2:

Method Name: Send event recommendation.

Class Name: Notification

ID: 2

Clients (Consumers): Main method

Associated Use Cases: Notify user *based on* GPS.

Description of Responsibilities: Notify user based on user interests, Notify user about other user details.

Arguments Received: User Interests, events.

Type of Value Returned: User ID.

Pre-Conditions: General User needs to be signed in.

Post-Conditions: User received potential event recommendation.

LOGIC:

```
The system searches for the general user's interests.

IF the system finds general user interest ==  business user interest  # (event interests)

    Then system searches the database for other users who are attending these events

    IF system finds user connection attending the event

        THEN notify the general user about the details of the event and details of
        connection attending the event

    ELSE notify the general user about the details of the event.

    END IF

END IF
```

Contract 3:

Method Name: Send message.

Class Name: General user

ID: 3

Clients (Consumers): Main method

Associated Use Cases: Message between users.

Description of Responsibilities: Messaging friend.

Arguments Received: Messages.

Type of Value Returned: Messages.

Pre-Conditions: Users need to be connected.

Post-Conditions:

LOGIC:

The user searches for another user.

The system searches the database to get details of the other user.

The user composes a message and sends it.

The system receives this message and sends it to the database.

The database stores the message.

The system then retrieves the message from the database and sends it to the other user.

The other user receives the message sent by the first user.

Contract 4:

Method Name: Create User Id.

Class Name: User

ID: 4

Clients (Consumers): Main method

Associated Use Cases:

Description of Responsibilities: Creating account.

Arguments Received: Name, user_id, email, phone number, address, interests.

Type of Value Returned: Name, user_id, email, phone number, address, interests.

Pre-Conditions: User had an Email Id

Post-Conditions:

LOGIC:

```
The user enters the details - name, user_id, email, phone number, address, and interests.
i = 1
While ( i =1):
    The system searches for the user_id whether it is already taken.
    IF the user_id is used
        Print ("Try different user_id")
        i = 1
    else i = 0
    END IF
```

Contract 5:

Method Name: Update privacy settings

Class Name: Settings

ID: 5

Clients (Consumers): Main method

Associated Use Cases:

Description of Responsibilities: Updating Privacy Settings

Arguments Received: Preferences

Type of Value Returned: Preferences.

Pre-Conditions: Users need to have an account.

Post-Conditions:

LOGIC:

```
n = 1
count = 0
While (n = 1 AND count < 7)
    count = count +1
    IF count == 6
        PRINT(" maximum tries exhausted, try again later")
        BREAK
    END IF
    pass = STR (INPUT (PRINT ( " Enter old password" ) ) )
    IF pass == password   # 'password' refers to password in the database
```

```
        new = STR( INPUT( PRINT ( “ Enter new password” ) ) )
        IF new == pass
            PRINT( “ Password same as old password, try new password “ )
            n = 1
        ELSE
            password = new
            PRINT( “ Password updated successfully“ )
            n = 0
        END IF
    ELSE
        PRINT(“ Wrong password, Try again”)
        n = 1
    END IF
```

Revenue Generation Strategies:

Our system, has following potential revenue streams:

Data Monetization: Analyze user behavior, preferences, and event attendance patterns to generate valuable insights. Aggregate and anonymize this data to sell to third parties, such as marketers or event organizers, looking to understand trends and user behavior for targeted marketing purposes.

Advertisement and Premium subscription: Business users who are hosting events or offering services which can align with user interests. Offering premium subscription for these business users for targeted advertising within the app.

Commission from Ticket Sales: Facilitate event hosting within the app and charge fees for ticket sales or event promotions. Business users organizing events could use our platform to reach a targeted audience, and a percentage of ticket sales could be collected as revenue.