**Software Requirements and Design Document**

**For**

**Group 17**

Version 1.0

**Authors**:

Alex F

Benji C

Brisan B

Justin M

Sri Harshini D

# Overview

We Created Party Finder App, which will stand a sort of social media exclusively for the sake of creating and finding events. Users can add their own listings with custom guidelines for each of their events along with a functional mapping system to the event. Additionally, a rating system will be applied for users to view and also allow our users to chat in our app.

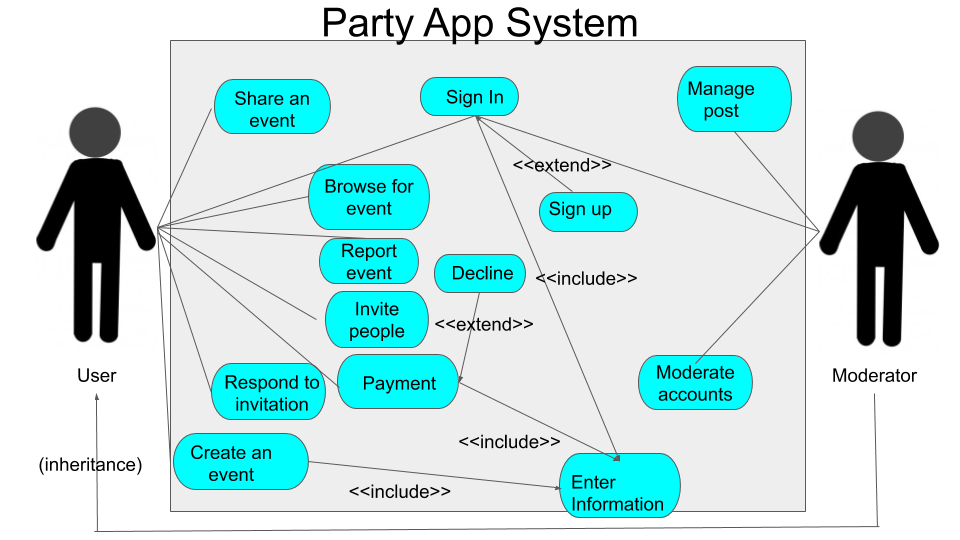
# Functional Requirements

* 1. Login: Application allows for users to create and manage their accounts
  2. Attendance: The app is capable of showing the number of people present at each party (low priority).
  3. Geo-Locating: Able to find the exact location of the event through the given address.
  4. Sharing: We can share the events on different locations using our app
  5. Listings: Application is capable of allowing users to post events (high priority).
  6. Rating: You can rate the events over a star scale
  7. Chat: Users are able to chat with each other and share party details.
  8. Error Checking: We have required error checking for every function.

# Non-functional Requirements

* ***GPS System:*** *The device being used requires a GPS system to be able to utilize the Geo-tracking features of the application.*
* ***Device Lock System:*** *If application remains logged in until signing out, the users security would remain protected by the systems internal lock system.*
* ***Updated Operating System:*** *The platform requires a minimum OS requirement that may be subject to change upon further development, so the device being used requires a up to date OS to continue using this program in an efficient manner.*
* ***Strong Connection:*** *A strong internet connection is required to log on and fully utilize Party Finder’s features.*

# Use Case Diagram

**

UseCase: Sign In

Actors: User, Moderator

Initial conditions: The user decides to sign into his account

Exit conditions: Signed into the account and homepage displayed

Flow of events:

1. The user enters the email and password

2. user hits sign in

3. Signed in and then the user is redirected to the homepage.

UseCase: Sign up

Actors: User, moderator

Initial conditions: when the user decides to sign in but does not have an account and decides to make one.

Exit conditions: account created and homepage displayed

Flow of events:

1. The user enters the username, email, password.

2. user hits sign up.

3. then the user is redirected to the homepage.

UseCase: Create an event

Actors: User

Initial conditions: User is signed in and decide to create an event.

Exit conditions: Event created

Flow of events:

1. User goes to his profile

2. Clicks on add an event

3. User enters the required information like name, description, address etc.

4. Hit create button and this creates an event

UseCase: Enter Information

Actors: User

Initial conditions: User either wants to create an account or event or make a payment

Exit conditions: redirects to the exit condition of the dependent use case, since this is an include usecase.

Flow of events:

1. User selects the action that he wants to perform from the above ones

2. The user is asked to give the required information

3. Enters the required information like username etc. for account or event information for event or payment details for payment

4. Hit submit information

UseCase: Browse for event

Actors: User

Initial conditions: the user is in homepage

Exit conditions:

Flow of events:

1. The user enters some keywords/filters to find an event

2. the page populates with the list of events that matches the requirements

UseCase: Invite people

Actors: User

Initial conditions: User already created an event

Exit conditions: Person invited

Flow of events:

1. The user goes to profile

2. choose the event from his list of events

3. Then select the desired people to invite

4. Hit submit, and the person is invited

UseCase: Share an event

Actors: User

Initial conditions: The user is interested in an event and wants to share it with others

Exit conditions: Event shared

Flow of events:

1. The user clicks on the event interested

2. Choose the share button provided in the event

3. Then given an option to share it to different people on his profile

4. shares the event with other people

UseCase: Report Event

Actors: User

Initial conditions: The user decides to report an event

Exit conditions: The moderator receives the report

Flow of events:

1. The user opens the event to be reported

2. User hits report button on the event.

3. Hit submit, this sends a message to the moderator.

UseCase: Payment

Actors: User

Initial conditions: User selects an event to pay for

Exit conditions: payment successful or decline

Flow of events:

1. User selects an event to pay

2. User enters information required for the payment

3. Hit submit, if the information is correct then the user is directed back to homepage and if declined then redirected to decline usecase.

UseCase: Decline

Actors: User

Initial conditions: User got declined from the payment

Exit conditions: redirected to payment

Flow of events:

1. A decline message is displayed

2. The user is redirected to payment page to enter the information again

UseCase: Respond to invitation

Actors: User

Initial conditions: User has been invited by another user for an event

Exit conditions: accept or decline invitation

Flow of events:

1. User receives an invitation from another user

2. User decides to either accept the invitation or decline the invitation

UseCase: Manage Post

Actors: Moderator

Initial conditions: Moderator reviews report.

Exit conditions: Remove post if necessary.

Flow of events:

1. Review report and post.

2. Delete post if necessary.

UseCase: Moderate accounts

Actors: Moderator

Initial conditions: Moderator reviews report about account.

Exit conditions: Suspend or remove account if necessary.

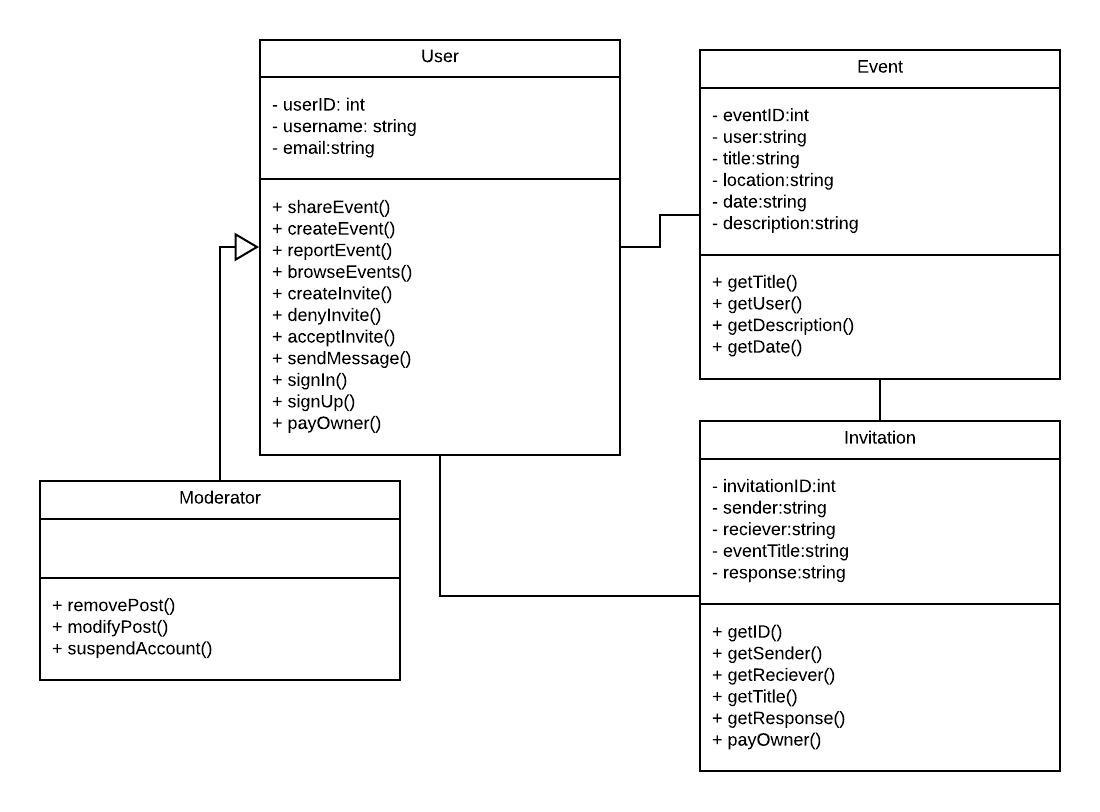
Flow of events:

1. Review account activity and posts.

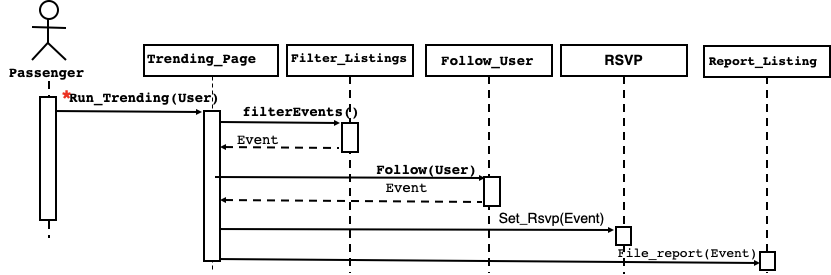
2. Suspend or delete account if necessary.

# Class Diagram and/or Sequence Diagrams

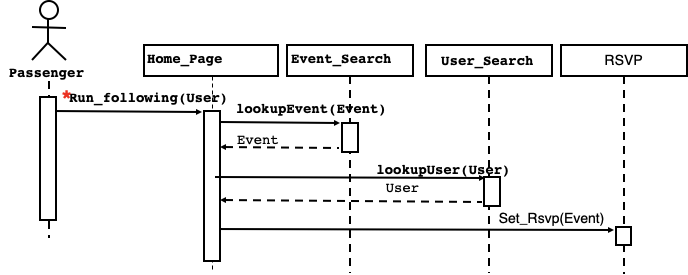
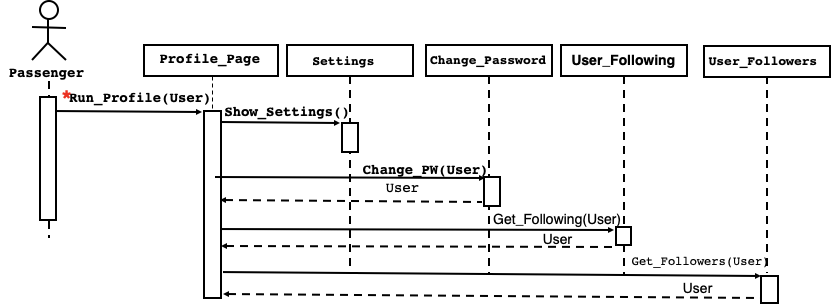
*Class Diagram:*

**

*Sequence Diagrams:*

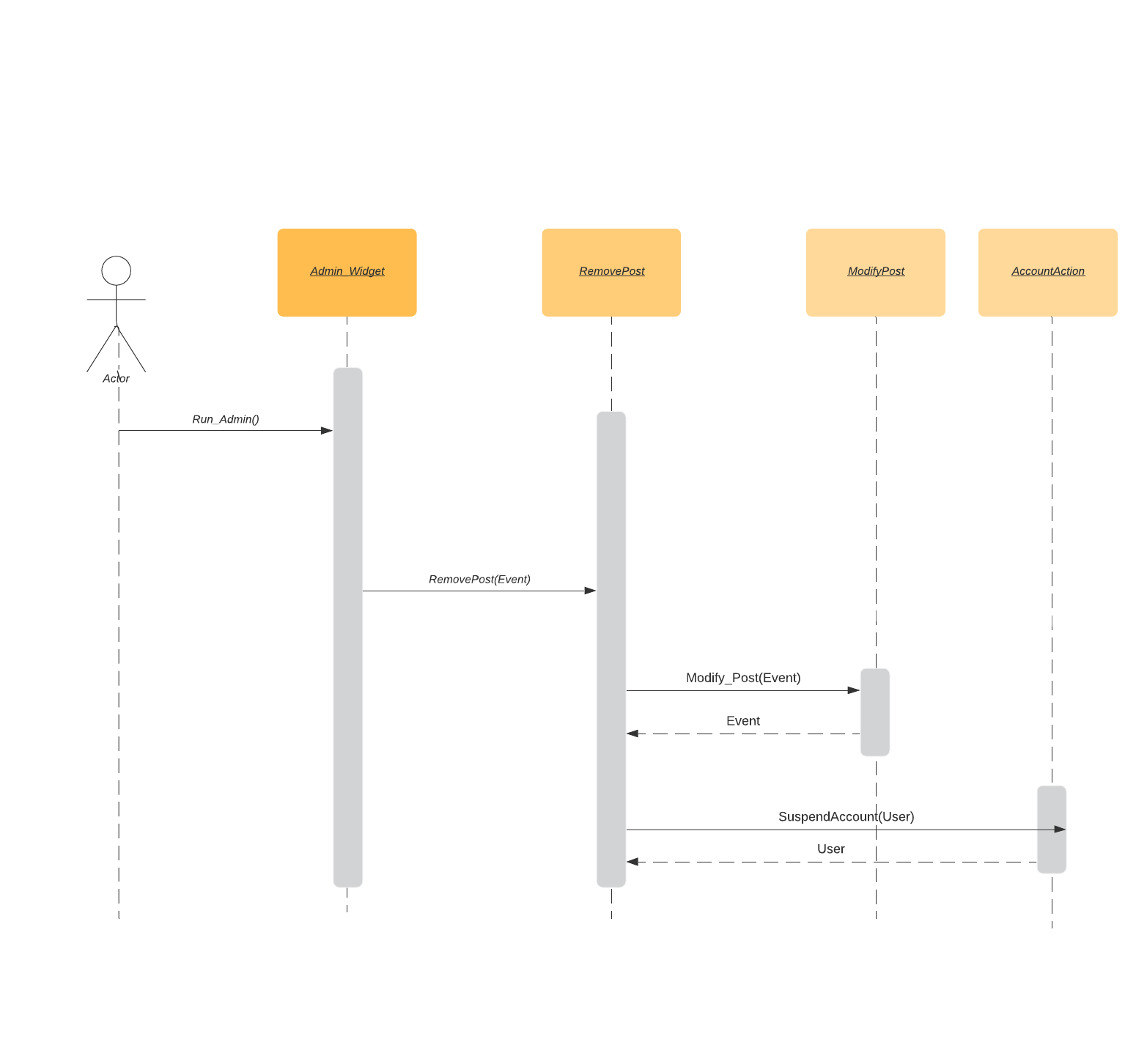


User

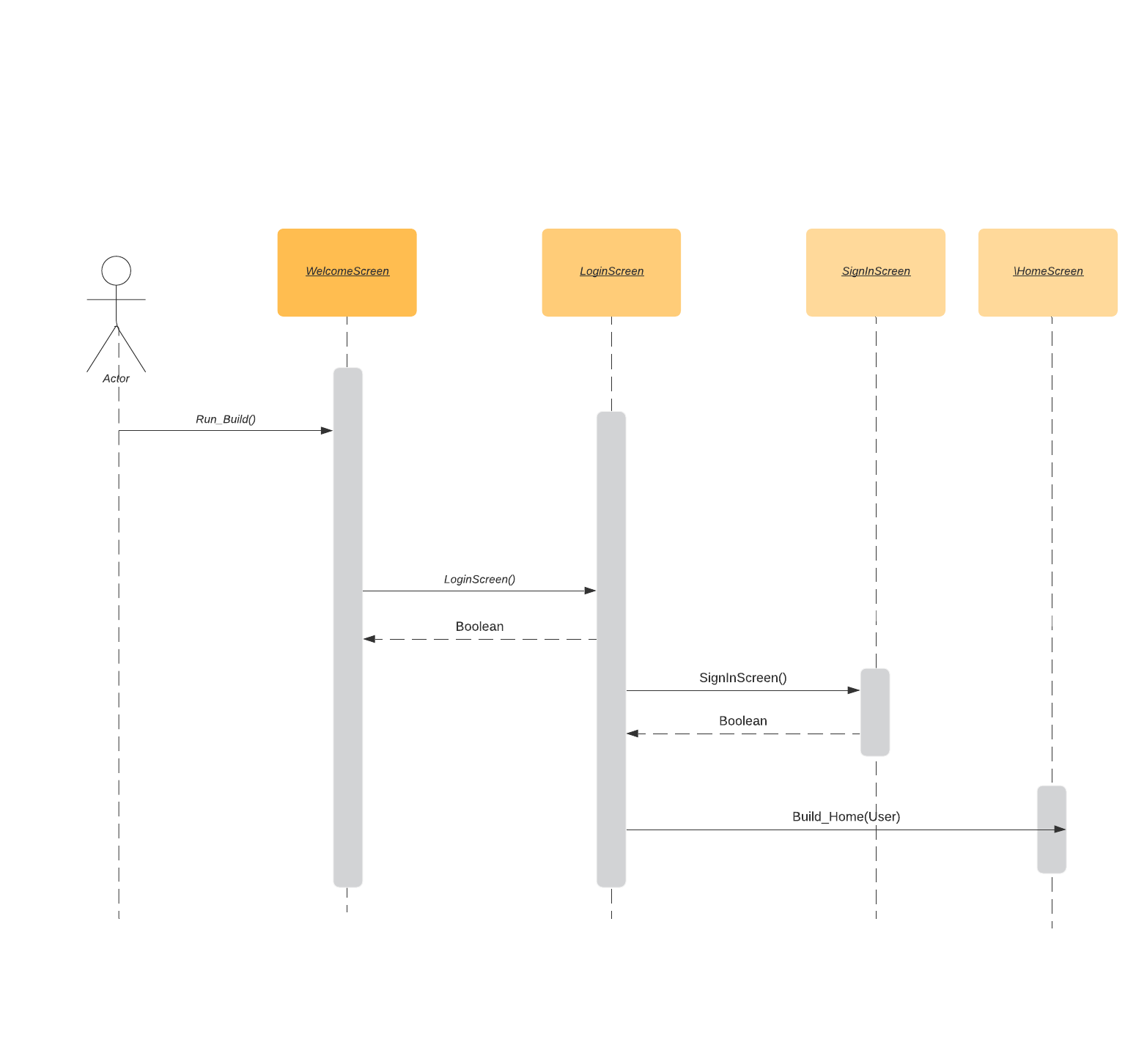


User

User



User



User

# Operating Environment

Party Finder is mainly a mobile app so it will be running on both iOS and android. Specifically we are shooting for a minimum of Android Oreo(8.0) and iOS 8. This way Party Finder will work on most older devices as well as the new ones.

# Assumptions and Dependencies

1. Google Maps API
2. Firebase messaging
3. We assume that users are honest about RSVP and rating. So, we assume they won’t rate the event if they didn’t attend.
4. Users are allowed to chat in a group, not to other users.
5. API’s:

firebase\_auth

firebase\_storage

cloud\_firestore

flutter\_share\_me

google\_sign\_in

smooth\_star\_rating

geocoder

place\_picker

flutter\_google\_places

datetime\_picker\_formfield