

Requirement Analysis

🕒 Created	@October 22, 2022 6:27 PM
🕒 Last Edited Time	@February 4, 2023 10:59 AM
▼ Type	Technical Spec
▼ Status	Done
👤 Created By	
👤 Last Edited By	
👥 Stakeholders	
☰ Property	

1. Introduction

1.1 Goal

The goal of this document is to delineate the relationship between the entities of the application, demonstrate user interface and show the flows of use cases presented in the previous document.

1.2 Content

1. Introduction

1.1 Goal

1.2 Content

1.3 Organization

2. System Requirements

Functional Requirements

Non-functional Requirements

3. Use Cases

3.1 User types

3.2 User scenarios

3.3 Use case diagram

Use Case Flow Diagram

3.4 Use cases

4. User Interface Model

5. Flow Diagrams

5.1 Flow Diagrams

5.2 General data model

5.3 Important data considerations

5.4 Data flow

Level - 0

Level - 1

Level - 2

1.3 Organization

In the section “System Requirements”, functional and non-functional requirements are presented.

In the section “Use Cases”, all kinds of user types with their associated scenarios and uses cases and related diagrams are included. User types defines the authorization of different users. Thinking possible events to be handled, user scenarios are prepared. Considering the requirements, use cases and their diagrams are provided.

In the section “User Interface Model”, all the designed mobile application screens are presented.

In the section “Flow Diagrams”, we first showed the flowcharts of the use cases. Then, we delineated the relationship of entities as well as providing their attributes. Finally, we presented data flow diagram for level-0, level-1 and level-2.

2. System Requirements

Functional Requirements

- Create, Read, Update, Delete operations on users
 - by Admin
 - We shall have these operations to give authority to admins for manage users
- Create, Read, Update, Delete operations on events

- by Admin
- We shall have these operations to give authority to admins for manage events
- Create, Read, Update, Delete operations on event groups
 - by Admin
 - We shall have these operations to give authority to admins for manage groups
- The admins can add user to any group and delete user from any group
 - by Admin
 - We shall have these operations to give authority to admins for manage people in user groups
- Read operation on logs
 - by Admin
 - We shall have these operations to give ability to admins for keeping track of all app activities
- Create special events to promote company events among all users
 - by Admin
 - We may have these operations to create events that will be seen in every group via request of companies
- Create operation on logs
 - by System
 - We shall have these operations to produce trackable app activity information for admins and users
- Delete operation on logs (2 weeks after the event's finished date)
 - by System
 - We shall have these operations to use database more frugal and save money by cleaning logs after a certain period
- Notify users when they are invited to a group
 - by System

- not mandatory (desirable)
- We may implement this operation to make users informed about received group invitations
- Notify users as the time of the event approaches
 - by System
 - not mandatory (desirable)
 - We may implement this operation to make users informed about upcoming events
- Update member status of event invitation after a user action
 - by System
 - We shall have this operation to announce user participation to all other users in same group
- Automatically set all member statuses as pending when an event is created
 - by System
 - We shall have this operation to mark user participation status as unknown
- Create an account using email, username and password
 - by User
 - We shall have these operations to give users the ability of creating their own accounts
- Update their account details
 - by User
 - We shall have this operation for users to be able to change their account information
- Delete their accounts
 - by User
 - We shall have these operations to give users the ability of deleting their own accounts

- Sign in to a created account
 - by User
 - We shall have these operations to give users the ability of reaching out to their own accounts
- Sign out from the account
 - by User
 - We shall have these operations to give users the ability of finishing their current activity on app
- Create and read operations on event groups
 - by User
 - We shall have these operations in order to give users the ability of creating event groups at will and browse the already existing one
- The group owner can invite a person to the group
 - by User
 - We shall have these operations in order to give group owners the ability of add other users to the group
- Update and delete operations on event groups if the user is the owner of the group
 - by User
 - We shall have these operations in order to give group owners the ability of editing information of group and also deleting the existing group
- Create, Read, Update, Delete operations on the belonging account
 - by User
 - We shall have these operations to give users the ability of manage their own accounts
- Read operations on logs of joined groups
 - by User
 - We shall have these operations to give ability to users for keeping track of app activities in belonged groups

- Accept, reject an event invitation or update the status
 - by User
 - We shall have these operations to give ability to users to be able to indicate their response about received event invitations
- Add or remove a user from group
 - by User
 - We should have these operations in order to give ability to group creator user to add or remove others
- Quit group
 - by User
 - We shall have this operation in order to make users able to exit from an already existing joined group
- Read operation on today's invited events
 - by User
 - We shall have this operation in order to make users able to keep track of current events and their information
- Create, Read, Update, Delete operations by event owners on events in joined groups
 - by User
 - We shall have these operations to give authority to event owner users for manage events
- Read operation on events in joined groups
 - by User
 - We shall have this operation to give ability to users for learning details of the events which were organized in belonged groups

Non-functional Requirements

Aa No.	≡ Type	≡ Description
<u>1</u>	Performance	The application should not require unnecessary computing power and should respond to the actions of users fastly (60-90ms)
<u>2</u>	Performance	The load time for fetching event groups should not exceed 2 seconds
<u>3</u>	Security	The personal information of users and admins' must be secured by the software
<u>4</u>	Security	The actions that a person can take must be determined by a role assigned to their account.
<u>5</u>	Usability & Appearance	The application's user interface should be user-friendly as much as possible and the colors used in the application should be as simple as possible.
<u>6</u>	Data Integrity	Consistency and continuous integration are the musts of the system.
<u>7</u>	Reusability	The code repository must obey the principles of software engineering and the written modules should be reusable.
<u>8</u>	Reliability	The application should be running properly and must be reliable.

3. Use Cases

3.1 User types

- Admin: Role for system managers who have *god mode* abilities on the system. They can read and manipulate any data kept in databases.
- User: Role for ordinary end-users who use the application to track their events in their friend groups.

3.2 User scenarios

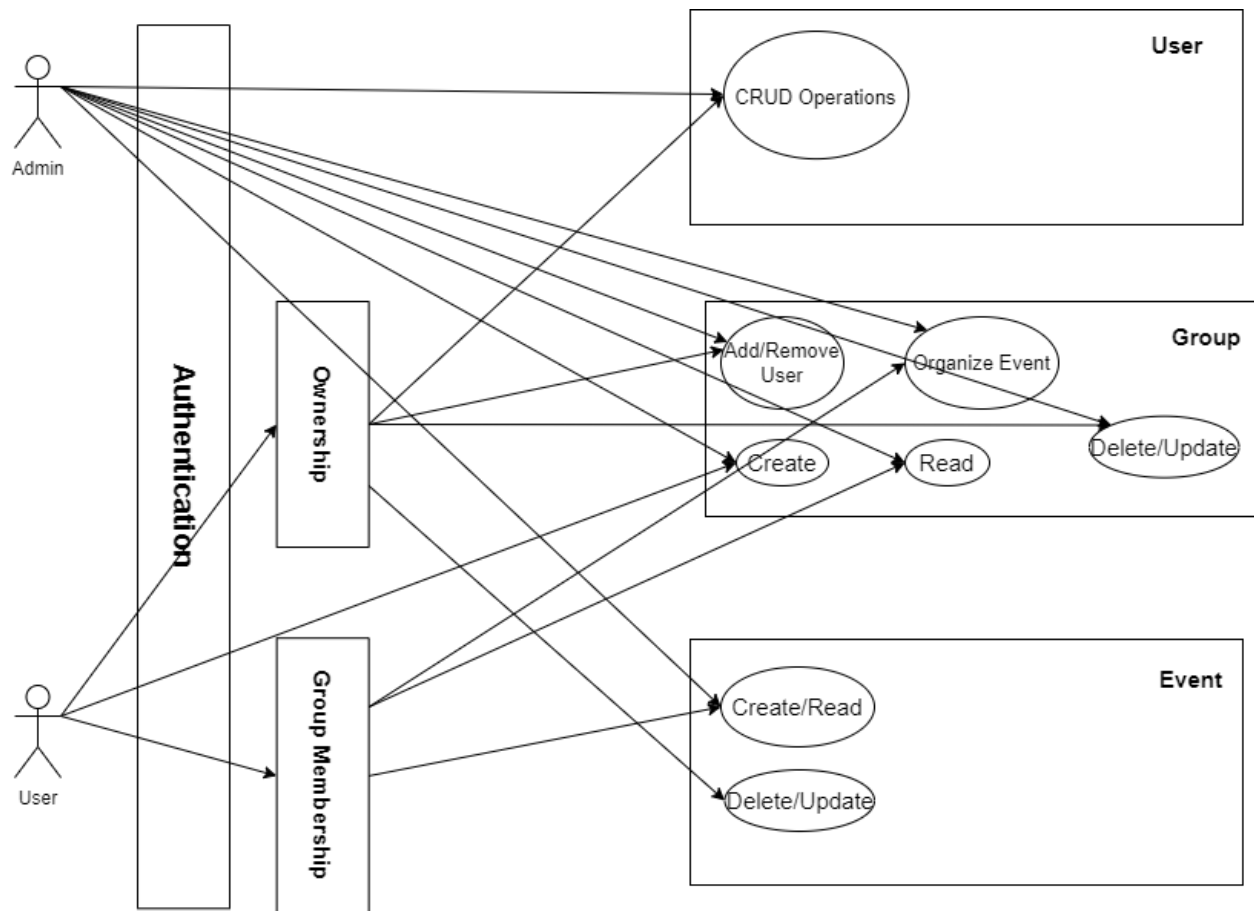
Admin: Turgut

- Turgut wants to access his account
- Turgut wants to see all registered users and manipulate them
- Turgut wants to see all events and manipulate them
- Turgut wants to see all groups and manipulate them

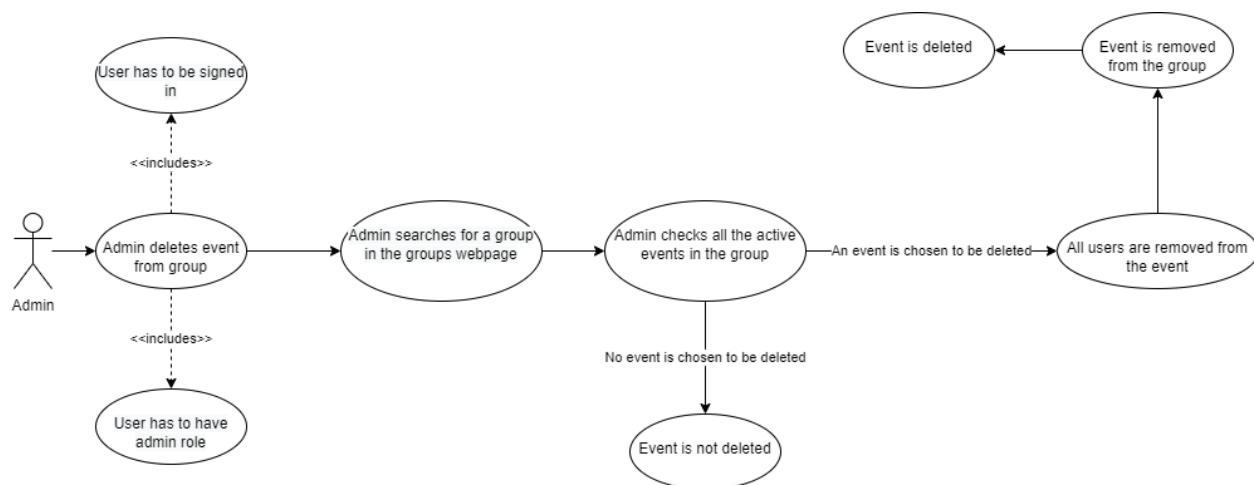
User: Uyar

- Uyar wants to create an account
- Uyar wants to access his account
- Uyar wants to see all accepted and invited events
- Uyar wants to respond to an event invitation
- Uyar wants to see details of an event
- Uyar wants to see his groups
- Uyar wants to details of one of his groups
- Uyar wants to create an event in one of his groups
- Uyar wants to invite all friends in one of his groups to the event
- Uyar wants to add people to the group
- Uyar wants to update one of his groups
- Uyar wants leave from one of his groups
- Uyar wants to update his profile
- Uyar wants to logout from his account
- Uyar wants to delete his account

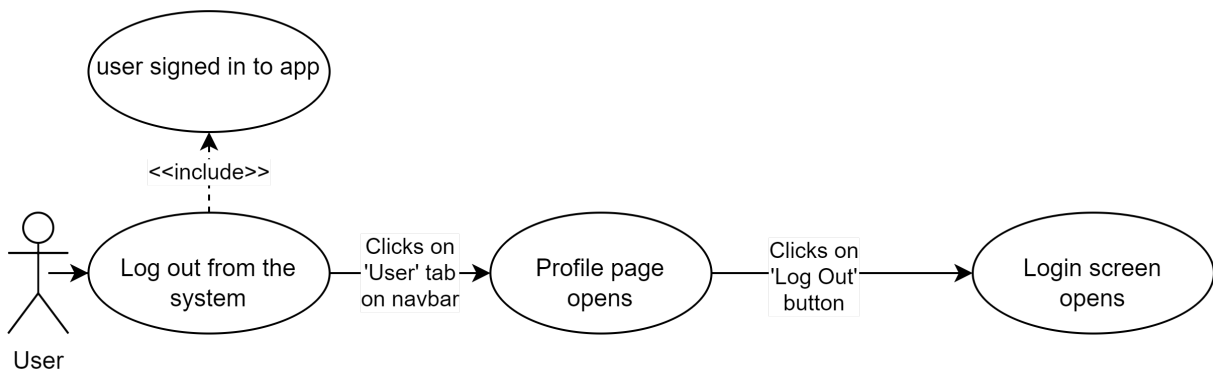
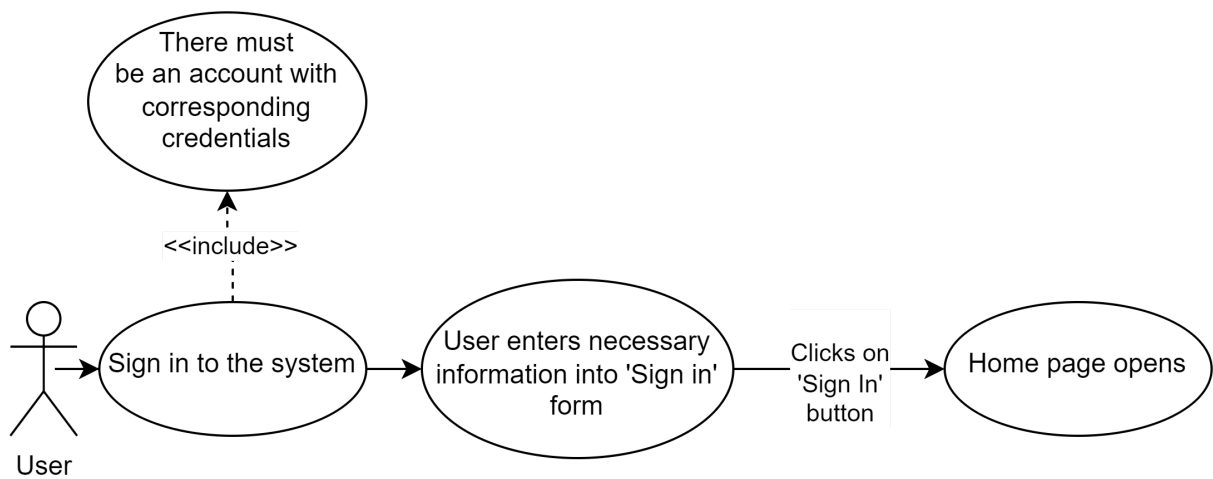
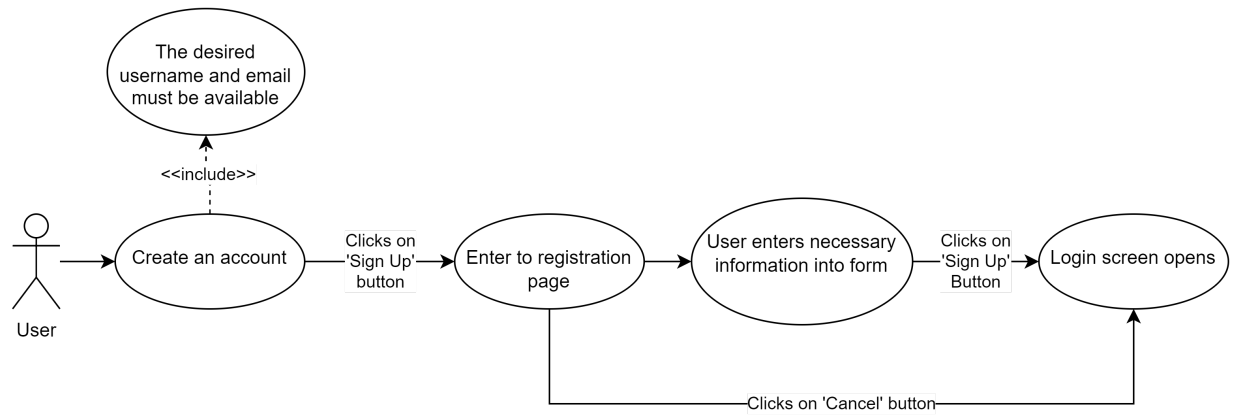
3.3 Use case diagram

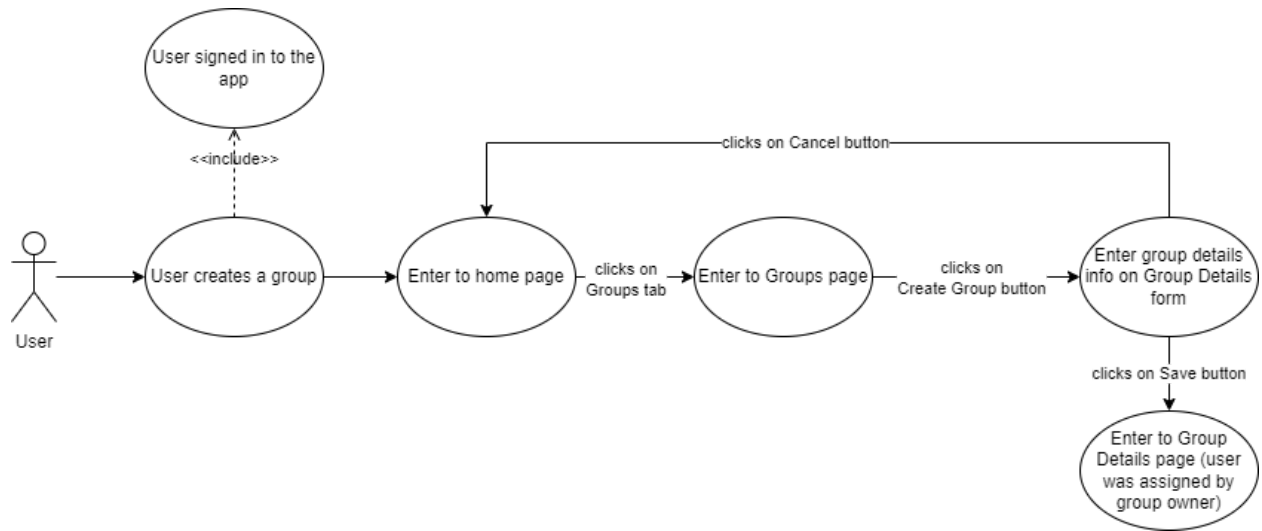
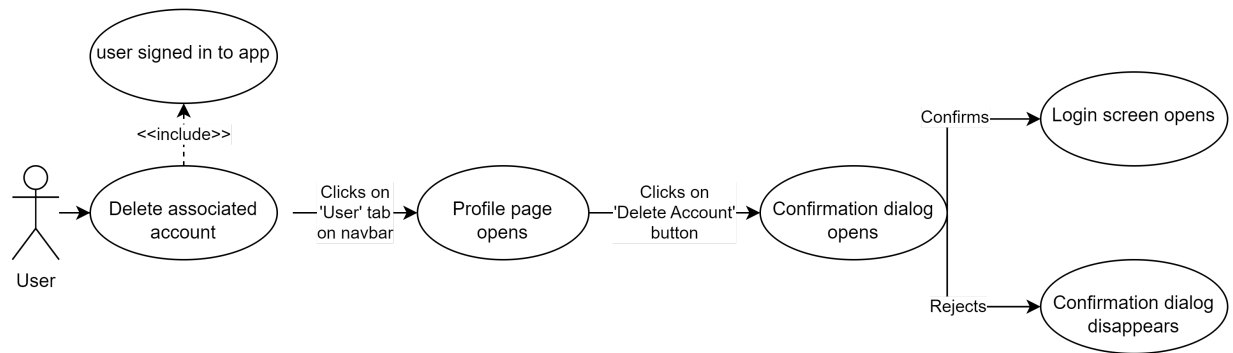
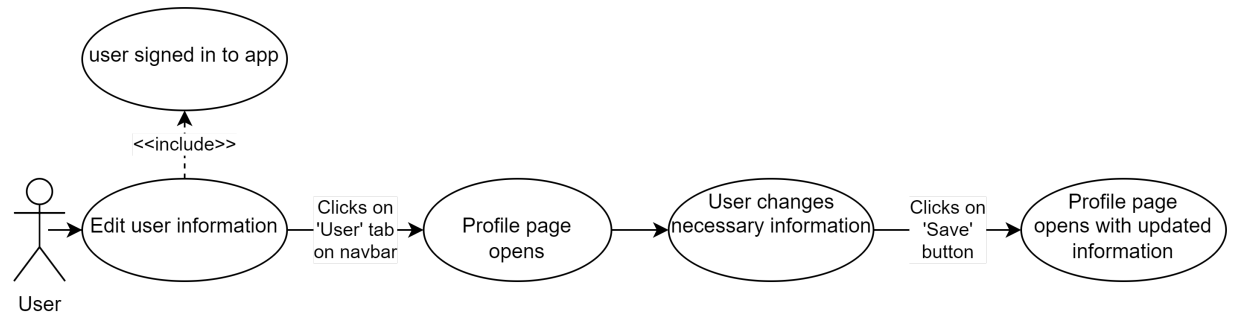


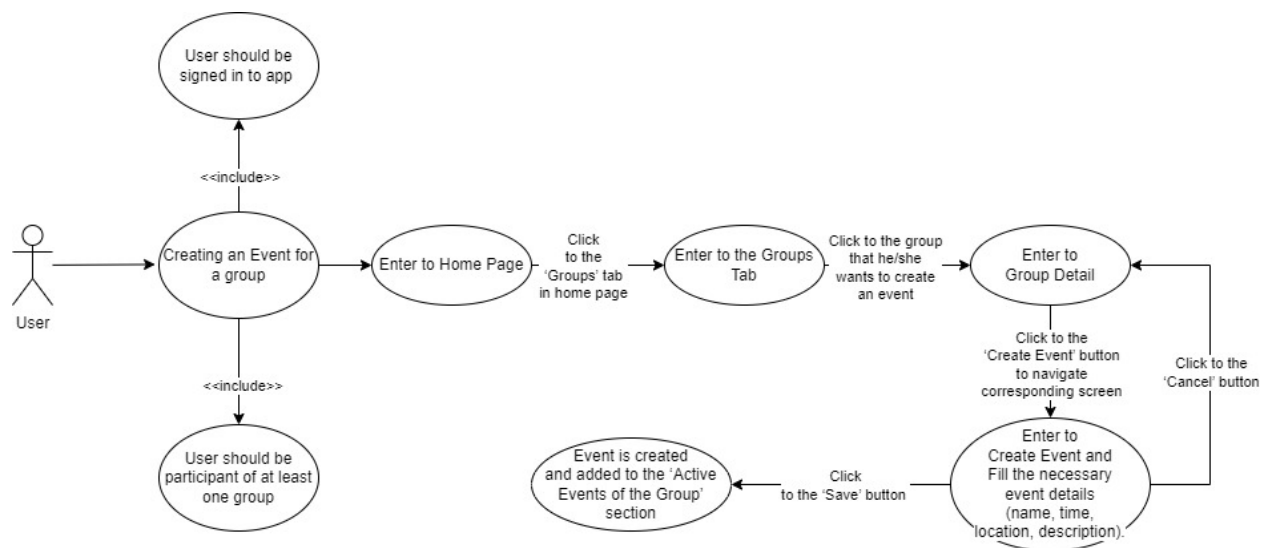
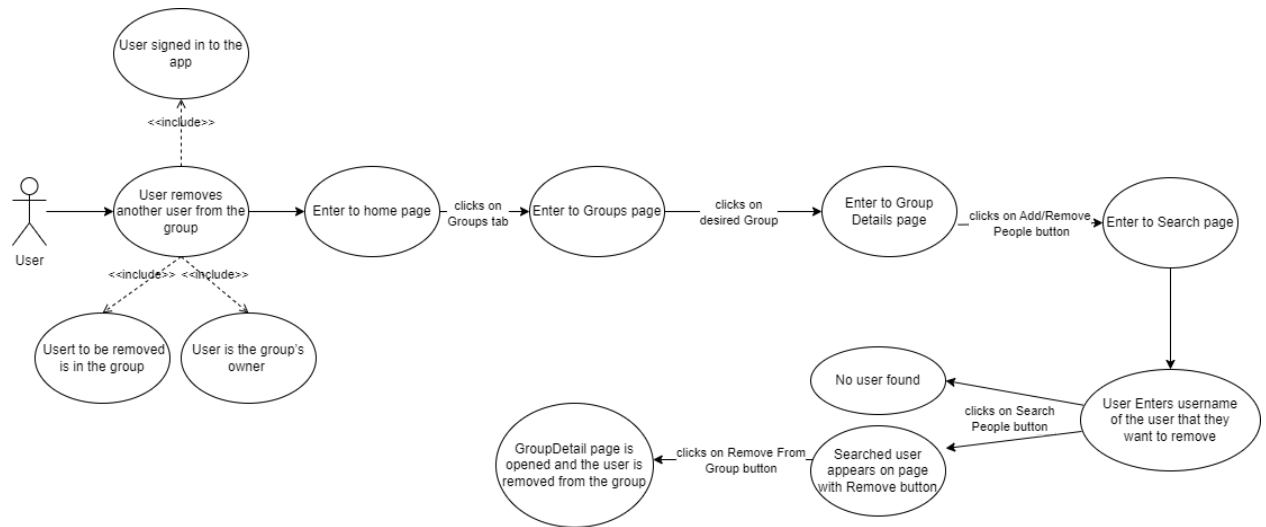
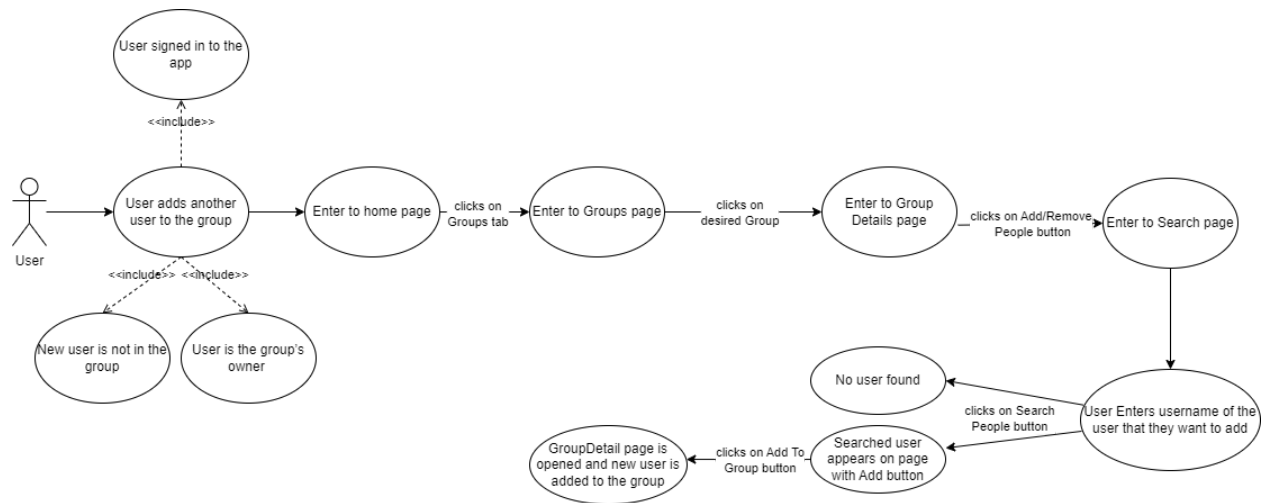
Use Case Flow Diagram

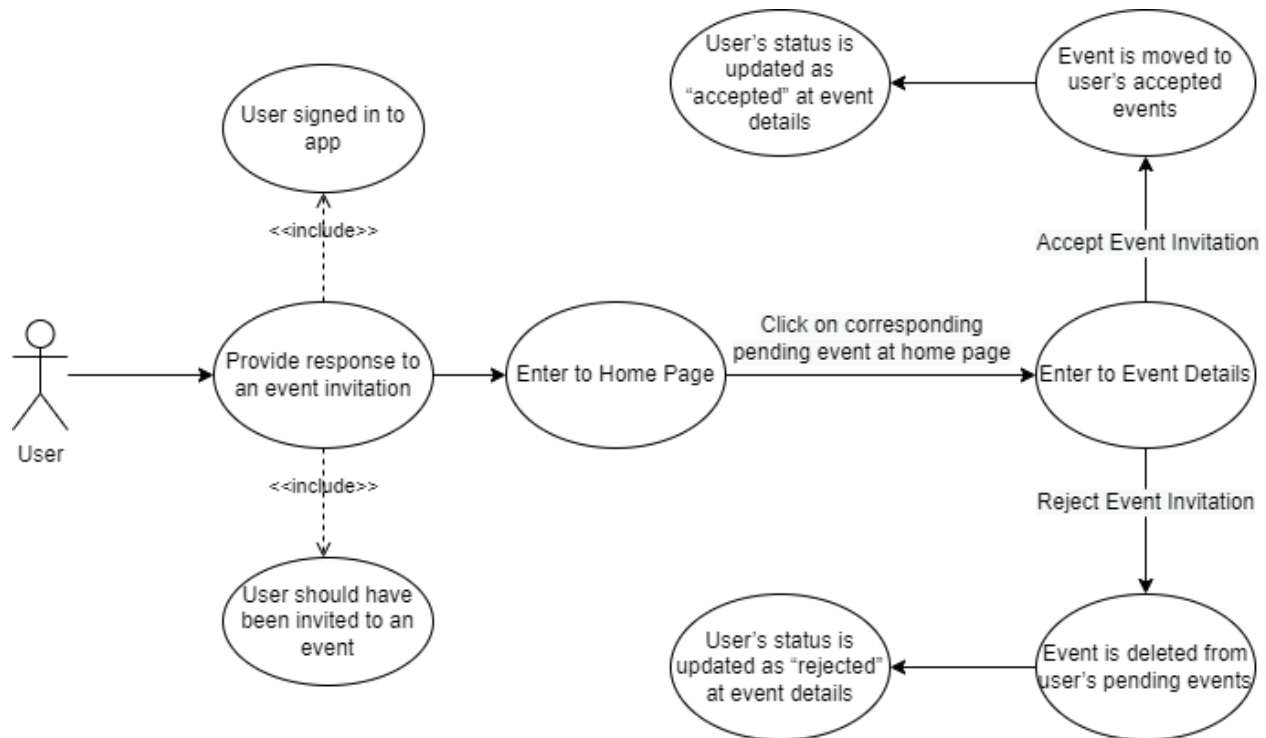












3.4 Use cases

Use Case Name	Create an account
Participating Actors	User
Entry Condition	-
Main Flow	1) User clicks on the 'Sign Up' button in the login page 2) User enters necessary information into registration form 3) User clicks on the 'Sign Up' button in the registration page 4) Login screen opens
Alternative Flow	2) User clicks on the 'Cancel' button in the registration page 3) Login screen opens
Special Requirements	The desired username and email should be available

Use Case Name	Sign in to the system
Participating Actors	User
Entry Condition	There must be an account with corresponding user credentials

Use Case Name	Sign in to the system
Main Flow	1) User enters necessary information into sign in form in entrance page 2) User clicks on the 'Sign In' button in the login page 3) Home screen opens
Alternative Flow	-
Special Requirements	-

Use Case Name	Log out from the system
Participating Actors	User
Entry Condition	User has to be signed in
Main Flow	1) User clicks on the 'User' tab on the navigation bar 2) User clicks on the 'Log Out' button in the 'User' tab 3) Login screen opens
Alternative Flow	1) Session time-out 2) Login screen opens
Special Requirements	-

Use Case Name	Edit user information
Participating Actors	User
Entry Condition	User has to be signed in
Main Flow	1) User clicks on the 'User' tab on the navigation bar 2) User changes the necessary information 3) User clicks on the 'Save' button 4) Profile page opens with updated information
Alternative Flow	-
Special Requirements	-

Use Case Name	User deletes associated account
Participating Actors	User
Entry Condition	User has to be signed in

Use Case Name	User deletes associated account
Main Flow	1) User clicks on the 'User' tab on the navigation bar 2) User clicks on the 'Delete Account' button 3) User clicks on the 'Yes' button on the confirmation dialog 4) Login screen opens
Alternative Flow	3) User clicks on the 'No' button on the confirmation dialog 4) Confirmation dialog is closed
Special Requirements	The account should not have the admin role

Use Case Name	User creates a group
Participating Actors	User
Entry Condition	User has to be signed in
Main Flow	1) User clicks on the 'Groups' tab in home page 2) User clicks on 'Create Group' button 3) User enters group details at Create Group page 4) User clicks on 'Save' button 5) GroupDetail page is opened and user is assigned as group owner
Alternative Flow	3/4) User clicks on Cancel 4/5) Home Page is opened
Exit Condition	Save Group Details / Cancel
Special Requirements	-

Use Case Name	User adds another user to the group
Participating Actors	User
Entry Condition	User has to be signed in User is the group's owner New user is not in the group
Main Flow	1) User clicks on the 'Groups' tab in home page 2) User clicks on the group that they desire to add an user into it 3) User clicks on 'Add/Remove People' button at GroupDetail page 4) User enters username of the user that they want to add 5) User clicks on search people button 6) Searched user appears on page 7) Click on Add To Group button 8) GroupDetail page is opened and new user is added to the group
Alternative Flow	6) No user found appears on page
Exit Condition	Returning back/ Add user to group

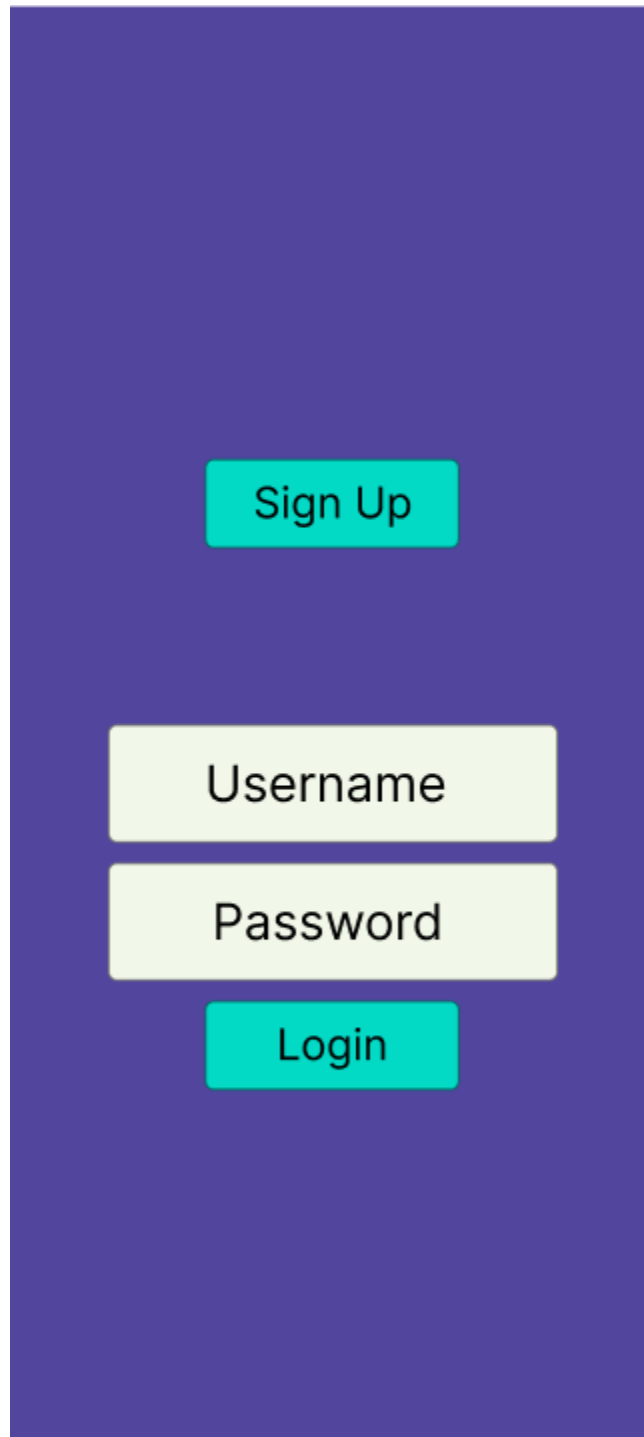
Use Case Name	User adds another user to the group
Special Requirements	User knows username of the user that they want to add Add To Group button appears for new user when their name is searched
Use Case Name	User remove another user from the group
Participating Actors	User
Entry Condition	User has to be signed in User is the group's owner User to be removed is in the group
Main Flow	1) User clicks on the 'Groups' tab in home page 2) User clicks on the group that they desire to remove an user from it 3) User clicks on 'Add/Remove People' button at GroupDetail page 4) User enters username of the user that they want to remove 5) User clicks on search people button 6) Searched user appears on page 7) Click on Remove From Group button 8) GroupDetail page is opened and the user to be removed is removed from the group
Alternative Flow	6) No user found appears on page
Exit Condition	Returning back/ Remove user from group
Special Requirements	User knows username of the user that they want to remove Remove From Group button appears for user to be removed when their name is searched
Use Case Name	Create an Event for a group
Participating Actors	User
Entry Condition	User should be signed in to app User should be participant of at least one group
Main Flow	1) User clicks to the 'Groups' tab in home page 2) User clicks to the group that he/she wants to create an event 3) User clicks to the 'Create Event' button to navigate corresponding screen 4) User fills the necessary event details (name, time, location, description) and clicks to the 'Save' button 5) Event is created and added to the 'Active Events of the Group' section
Alternative Flow	4) User clicks to the 'Cancel' button
Exit Condition	Creating Event / Cancelling process
Special Requirements	-

Use Case Name	Provide response to an event invitation
Participating Actors	User
Entry Condition	User should be signed in to app User should be a member of group that the event belongs to User should have been invited to an event
Main Flow	1) User click on corresponding pending event at home page 2) User accept invitation from the event detail page 3) Event is moved to user's accepted events 4) User's status is updated as "accepted" at event details
Alternative Flow	2) User reject invitation from the event detail page 3) Event is deleted from user's pending events 4) User's status is updated as "rejected" at event details
Exit Condition	Accept invitation/Reject invitation
Special Requirements	User got invitation for corresponding event which was sent by event creator
Use Case Name	Admin deletes user account
Participating Actors	Admin
Entry Condition	User has to be signed in User has to have admin role
Main Flow	1) Admin searches for an account in the users webpage 2) Admin clicks on delete User 3) User is removed from all joined groups and events 4) User's account is deleted
Alternative Flow	2) Admin does not click on delete user 3) User's account is not deleted
Exit Condition	Delete account/Do not delete account
Special Requirements	-
Use Case Name	Admin deletes event group
Participating Actors	Admin
Entry Condition	User has to be signed in User has to have admin role
Main Flow	1) Admin searches for a group in the groups webpage 2) Admin clicks on delete event group 3) All users are removed from the group 4) All events that belong to the group are deleted and removed from users 5) Event group is deleted

Use Case Name	Admin deletes event group
Alternative Flow	2) Admin does not click on delete event group 3) Event group is not deleted
Exit Condition	Delete event group/Do not delete event group
Special Requirements	-

Use Case Name	Admin deletes event from group
Participating Actors	Admin
Entry Condition	User has to be signed in User has to have admin role
Main Flow	1) Admin searches for a group in the groups webpage 2) Admin checks all the active events in the group 3) An event is chosen to be deleted 4) All users are removed from the event 5) Event is removed from the group 6) Event is deleted
Alternative Flow	3) No event is chosen to be deleted 4) Event is not deleted
Exit Condition	Delete event from group/Do not delete event from group
Special Requirements	-

4. User Interface Model

A vertical rectangular screen with a solid purple background. Centered at the top is a red rounded rectangle with the text "Sign Up" in black. Below this, there are two white rounded rectangles stacked vertically. The top white rectangle contains the text "Username" in black, and the bottom white rectangle contains the text "Password" in black. At the bottom of the screen is another red rounded rectangle with the text "Login" in black.

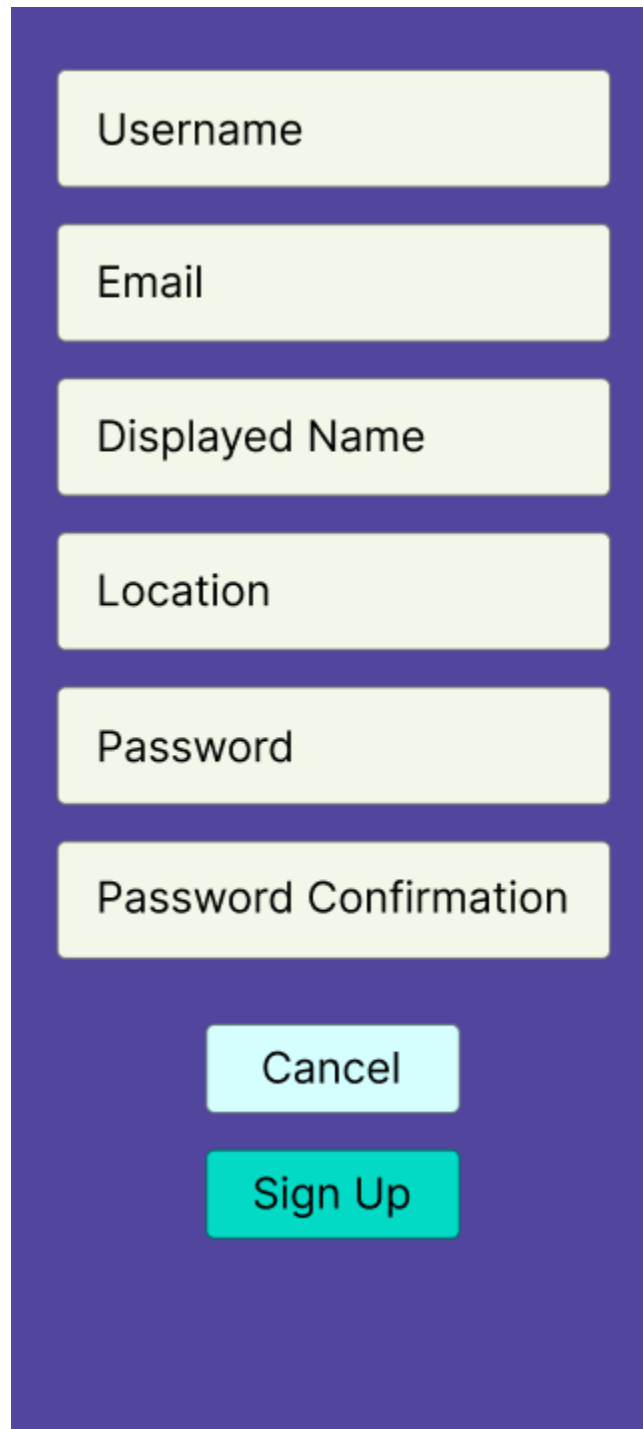
Sign Up

Username

Password

Login

Login Screen



A vertical sign-up form on a purple background. It consists of six light green input fields stacked vertically, followed by two buttons: a light blue 'Cancel' button and a teal 'Sign Up' button, both centered below the input fields.

Username

Email

Displayed Name

Location

Password

Password Confirmation

Cancel

Sign Up

Sign Up Screen

Events	Groups	User
--------	--------	------

Approved Events	
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time
Event Topic	Event Time

Home Screen/ Events Tab

Events Groups User

Group Name

Group Location

Group Name

Group Location

Group Name

Group Location

Group Name

Group Location

Group Name

Group Location

Group Name

Group Location

Create Group

Home Screen/ Groups Tab

Events	Groups	User
--------	--------	------

Username

DisplayName (Edit)

Location (Edit)

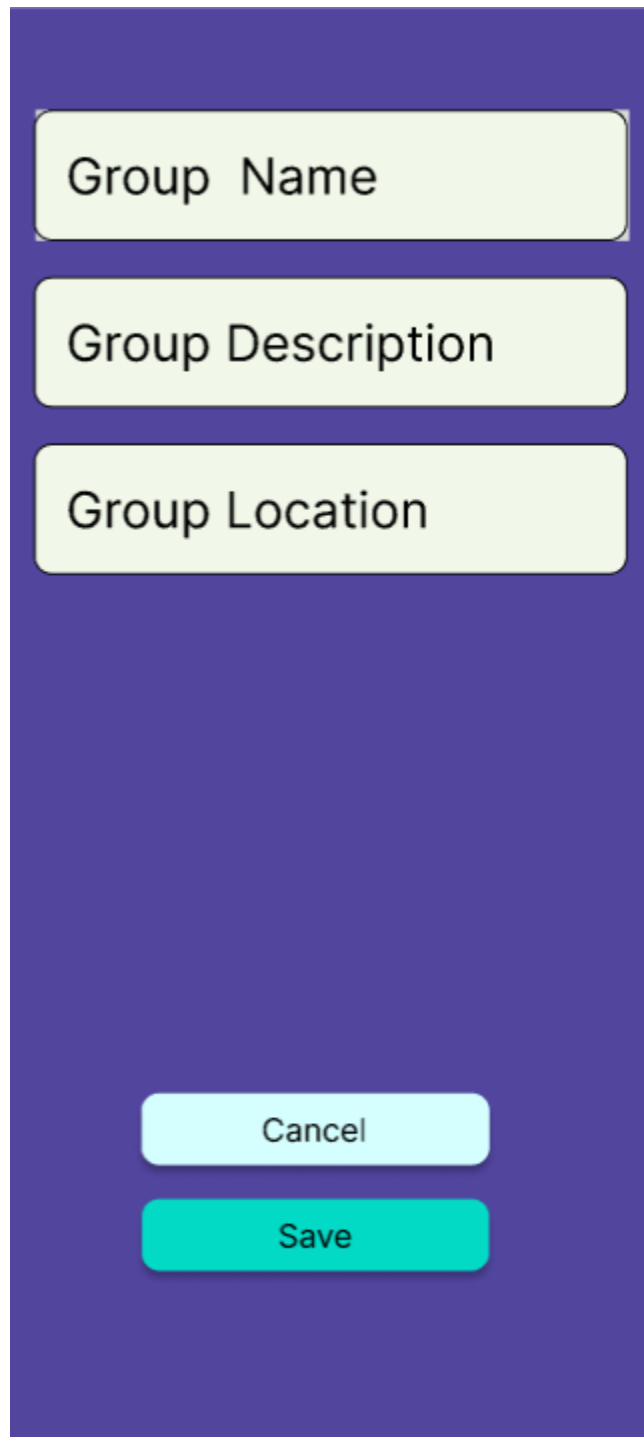
Save

Change Password

Log Out

Delete Account

Home Screen/ User Tab

A vertical mobile app screen with a dark purple background. It features three stacked, light green rounded rectangular input fields. The first field is labeled 'Group Name', the second 'Group Description', and the third 'Group Location'. At the bottom of the screen, there are two buttons: a light blue 'Cancel' button and a teal 'Save' button, both with rounded corners and a slight shadow.

Group Name

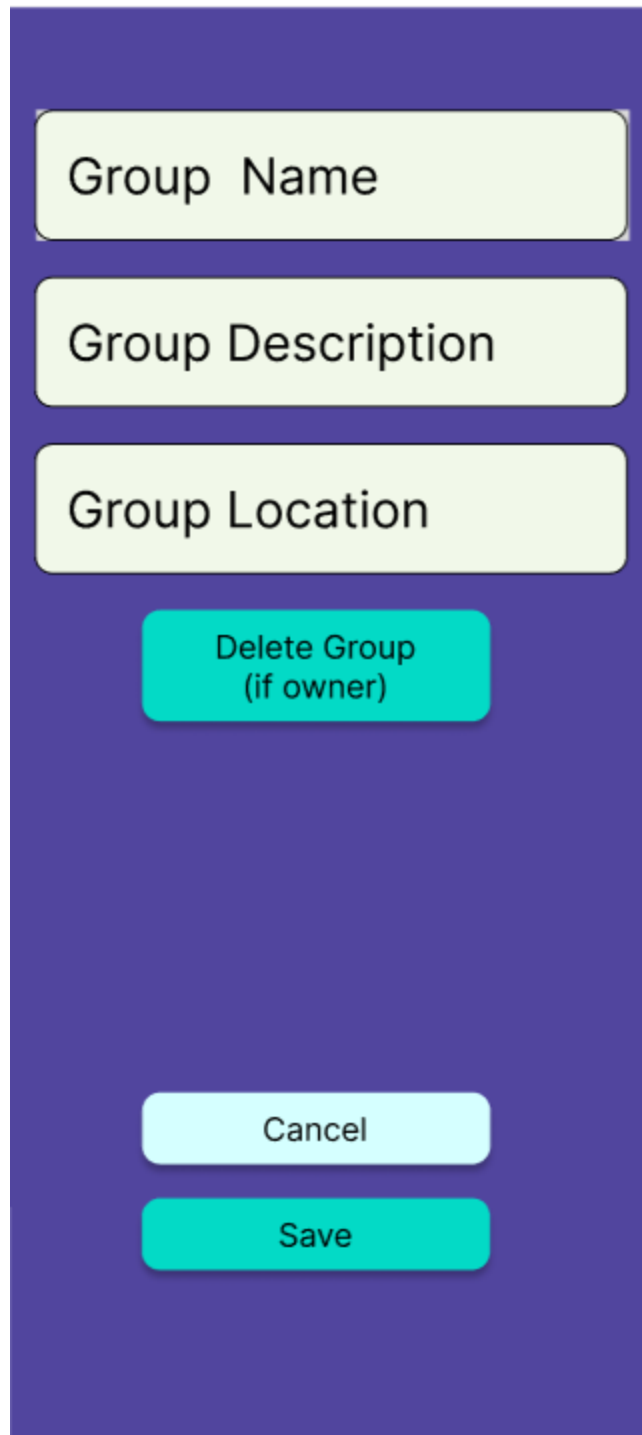
Group Description

Group Location

Cancel

Save

Create Group Screen

The image shows a mobile application screen for updating a group. It has a dark purple background. At the top, there are three light yellow input fields with rounded corners, each containing a label: 'Group Name', 'Group Description', and 'Group Location'. Below these fields is a red button with rounded corners that says 'Delete Group (if owner)'. At the bottom of the screen, there are two more buttons with rounded corners: a light blue 'Cancel' button and a red 'Save' button.

Group Name

Group Description

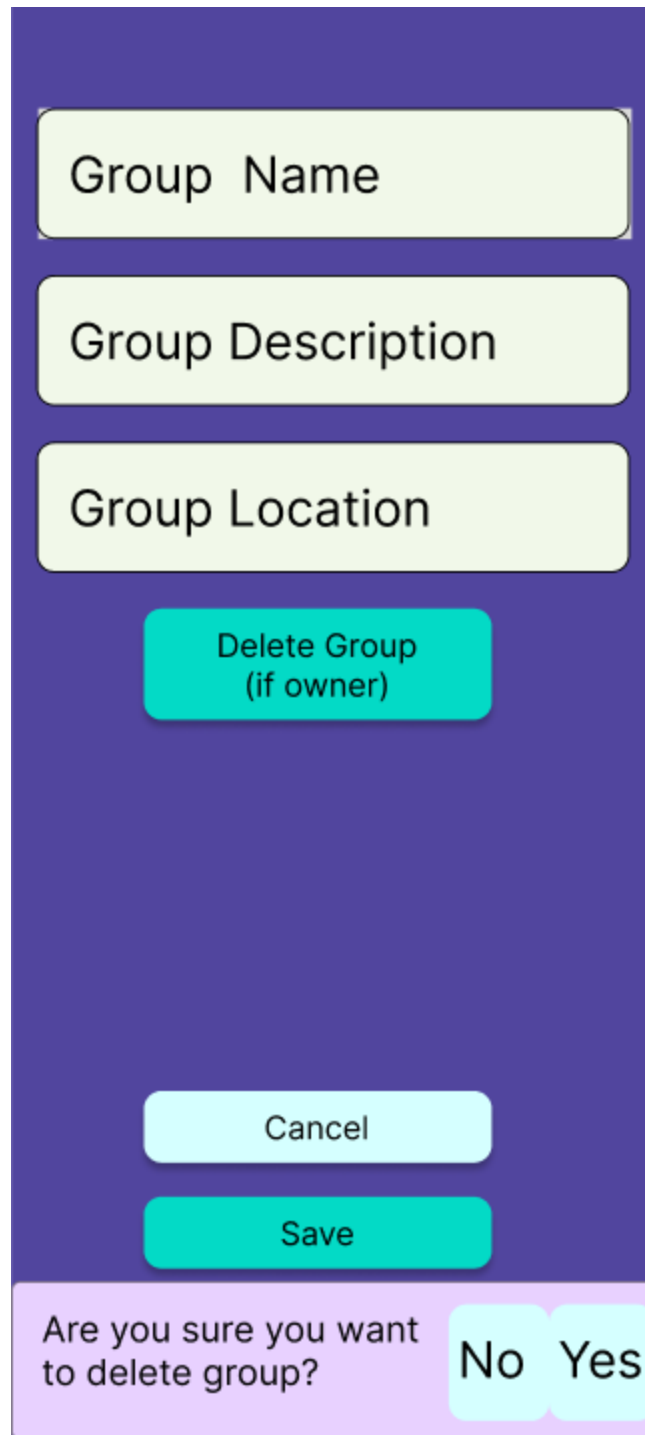
Group Location

Delete Group
(if owner)

Cancel

Save

Update Group Screen



The image shows a mobile application screen for deleting a group. It features a purple background with three light green input fields for 'Group Name', 'Group Description', and 'Group Location'. Below these is a red button labeled 'Delete Group (if owner)'. At the bottom, there are two buttons: a light blue 'Cancel' button and a red 'Save' button. A confirmation dialog is at the very bottom, with a light blue background, asking 'Are you sure you want to delete group?' and providing 'No' and 'Yes' options in red buttons.

Group Name

Group Description

Group Location

Delete Group
(if owner)

Cancel

Save

Are you sure you want
to delete group?

No Yes

Delete Group Screen

Group Name U X ←

Group Description

Member Names...

Active Events of the Group

Event1

Event2

Event3

Loglar

Ahmet created "Event1".

Yigit rejected to attend "Event1".

Mihri accepted to attend "Event1".

Tumay created new event named
"Event2".

Add/Remove
People

Create Event

Group Details Screen

Group Name

Group Description

Member Names...

Active Events of the Group

Event1

Event2

Event3

Loglar

Ahmet created "Event1".

Yigit rejected to attend "Event1".

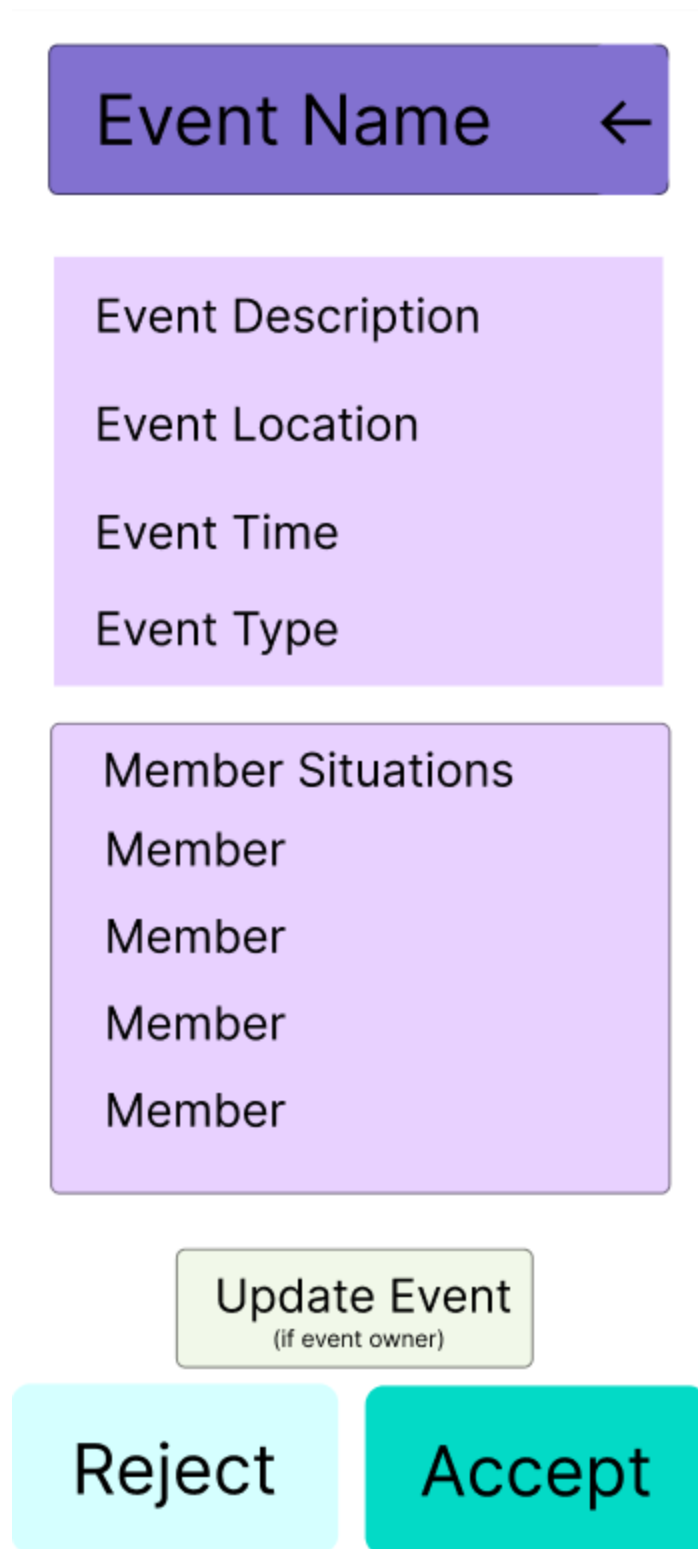
Mihri accepted to attend "Event1".

Tumay created new event named
"Event2".

Are you sure you want
to exit from group?

No Yes

Exit From Group Screen



The image shows a mobile application screen titled "Event Details Screen". At the top is a purple header bar with the text "Event Name" and a back arrow icon. Below the header is a light purple rectangular area containing four text labels: "Event Description", "Event Location", "Event Time", and "Event Type". Underneath this is another light purple rectangular area containing the text "Member Situations" followed by four instances of the word "Member" stacked vertically. Below these two purple areas is a light green button labeled "Update Event" with the text "(if event owner)" in smaller font below it. At the bottom of the screen are two large buttons: a light blue button labeled "Reject" and a teal button labeled "Accept".

Event Name ←

Event Description

Event Location

Event Time

Event Type

Member Situations

Member

Member

Member

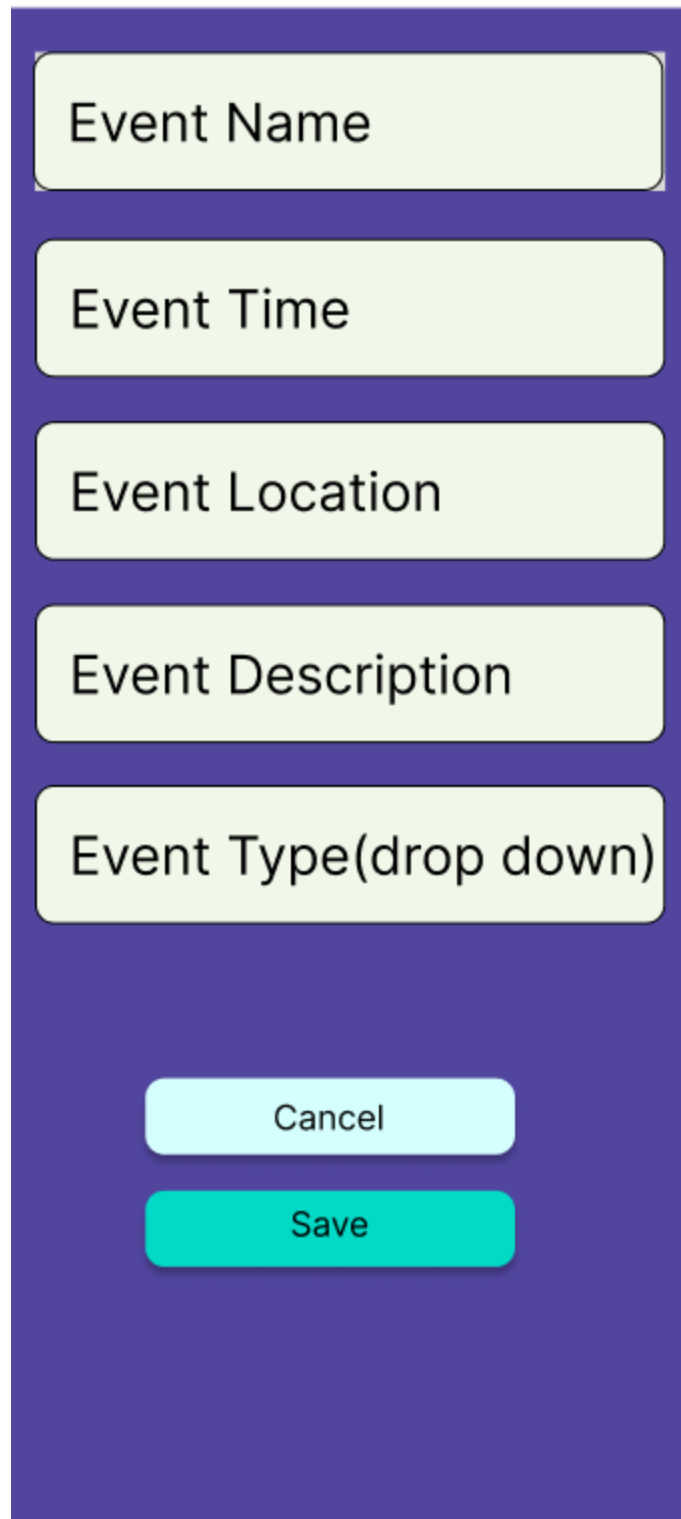
Member

Update Event
(if event owner)

Reject

Accept

Event Details Screen



The image shows a mobile application screen for creating an event. It features a dark purple background with five light green input fields stacked vertically. The fields are labeled 'Event Name', 'Event Time', 'Event Location', 'Event Description', and 'Event Type(drop down)'. Below these fields are two buttons: a light blue 'Cancel' button and a teal 'Save' button.

Event Name

Event Time

Event Location

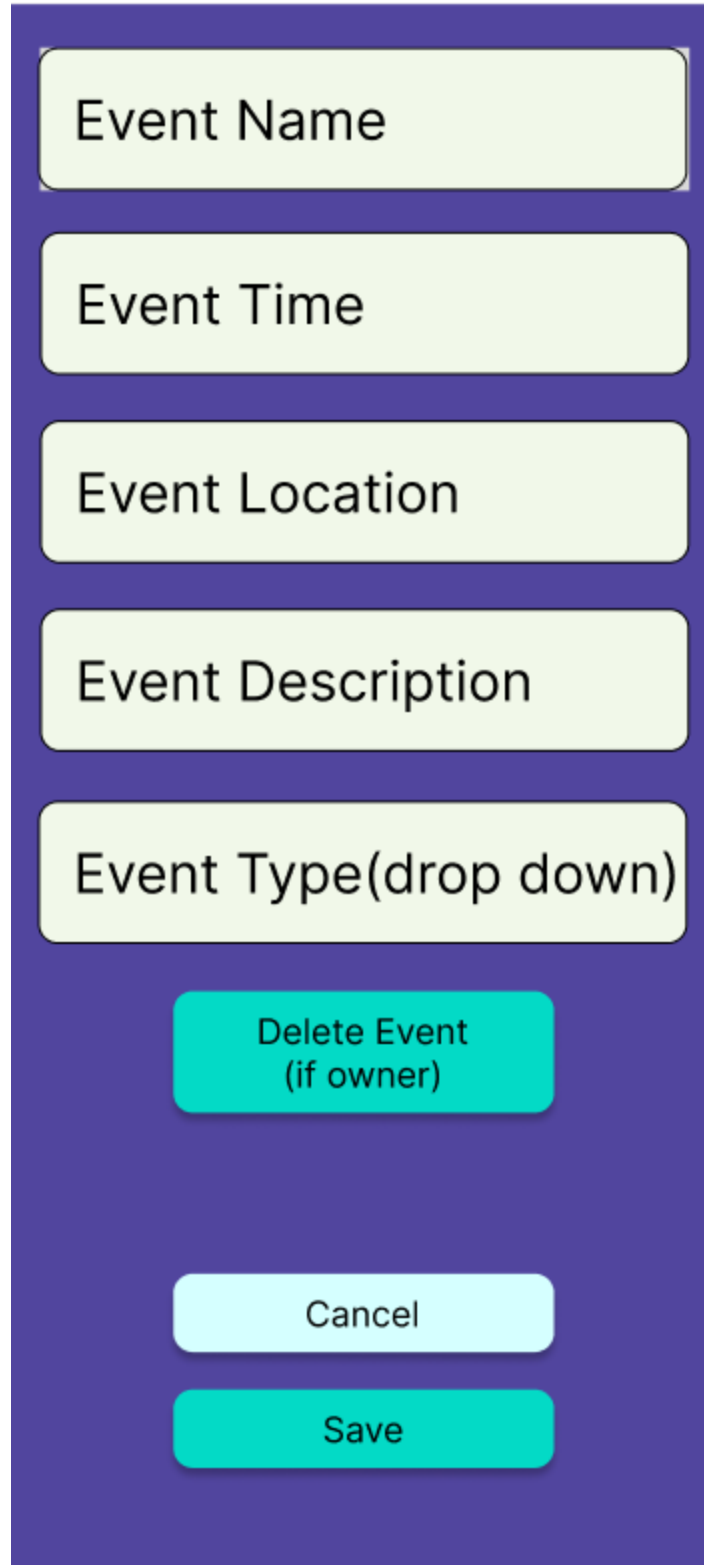
Event Description

Event Type(drop down)

Cancel

Save

Create Event Screen

A mobile application screen for updating an event. The screen has a dark purple background. It contains five light green input fields stacked vertically, each with a dark border and rounded corners. The fields are labeled 'Event Name', 'Event Time', 'Event Location', 'Event Description', and 'Event Type(drop down)'. Below these fields is a red button with white text that says 'Delete Event (if owner)'. At the bottom are two more buttons: a light blue 'Cancel' button and a red 'Save' button, both with white text and rounded corners.

Event Name

Event Time

Event Location

Event Description

Event Type(drop down)

Delete Event
(if owner)

Cancel

Save

Update Event Screen

Username

Search People

Founded People

Username Add to/Remove from Group

Add(will appear if user not already in the group) or Remove(otherwise) People Screen

Events	Groups	User
--------	--------	------

Username

DisplayName (Edit)

Location (Edit)

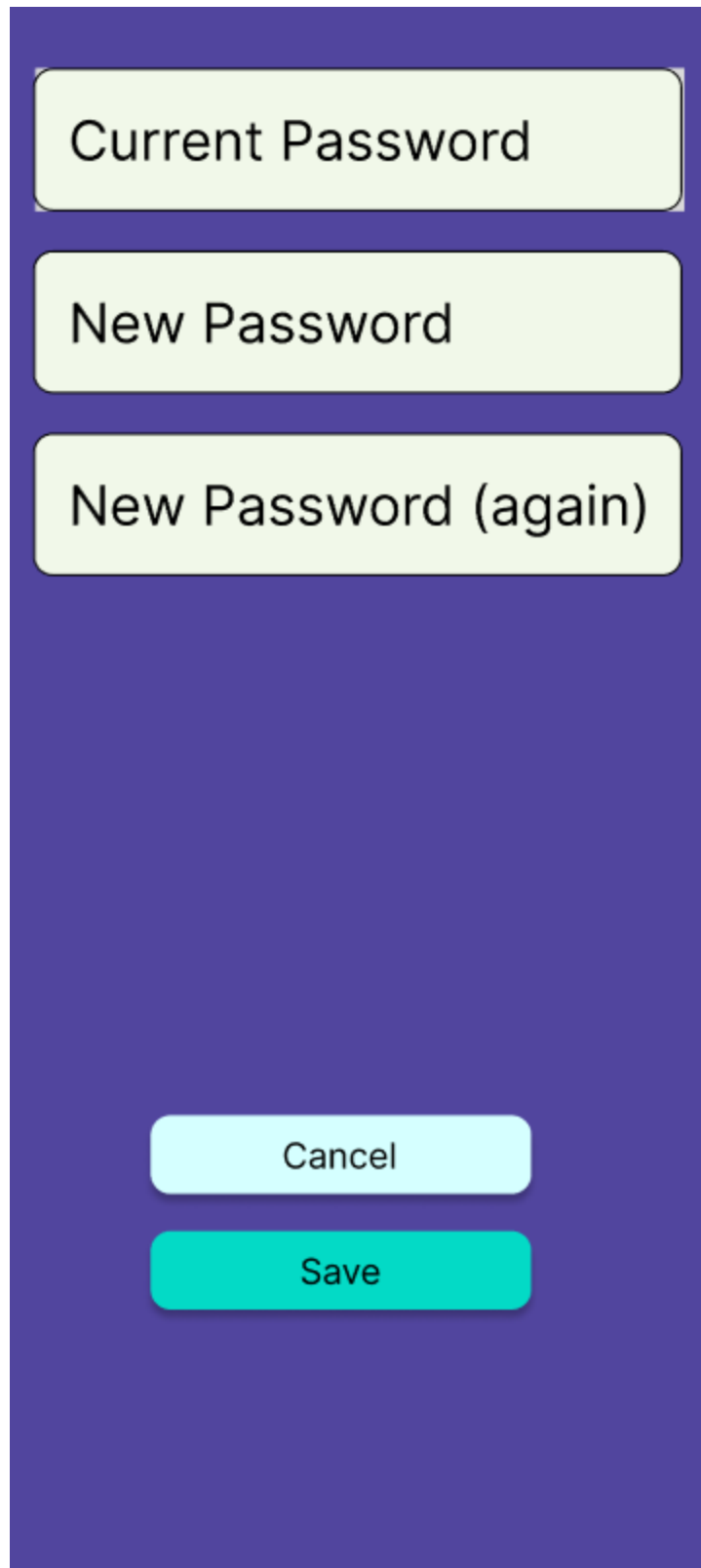
Save

Change Password

Log Out

Are you sure you want
to delete your account? No Yes

Delete Account Screen



A vertical UI mockup for a 'Change Password Screen'. It features a dark purple background. At the top, there are three stacked, light yellow rounded rectangular input fields. The first field is labeled 'Current Password', the second 'New Password', and the third 'New Password (again)'. At the bottom of the screen, there are two stacked rounded rectangular buttons: a light blue 'Cancel' button on top and a teal 'Save' button on the bottom. Both buttons have a slight drop shadow.

Current Password

New Password

New Password (again)

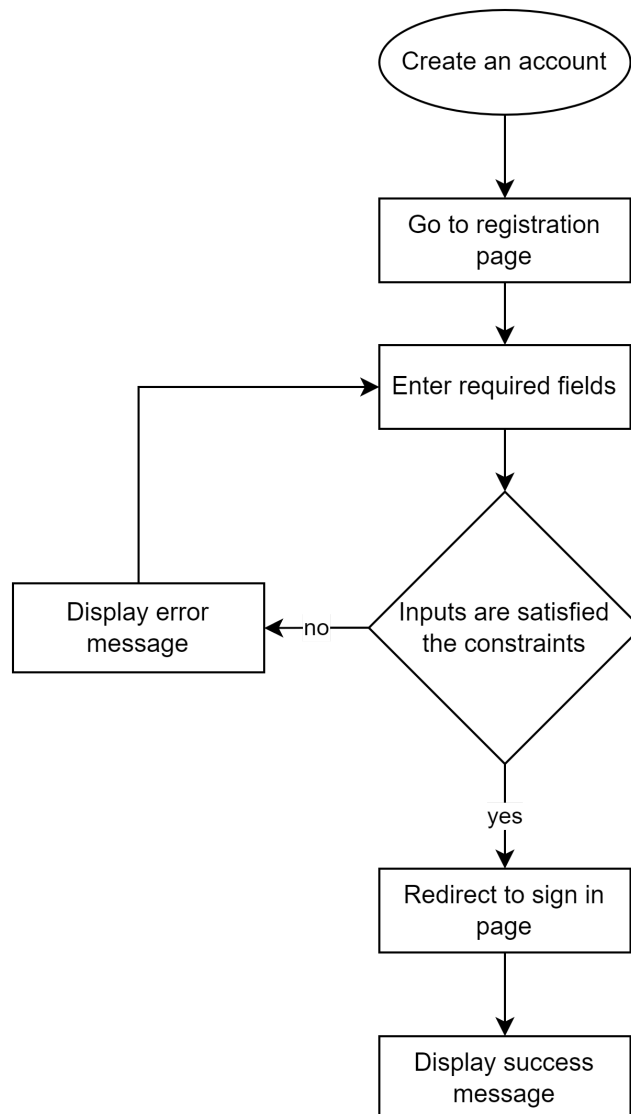
Cancel

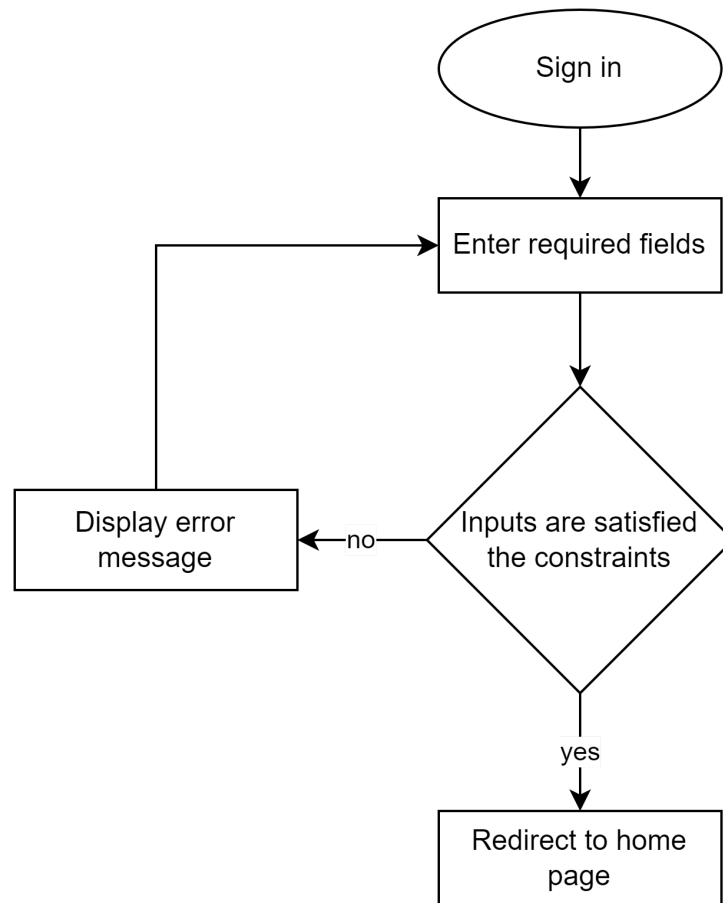
Save

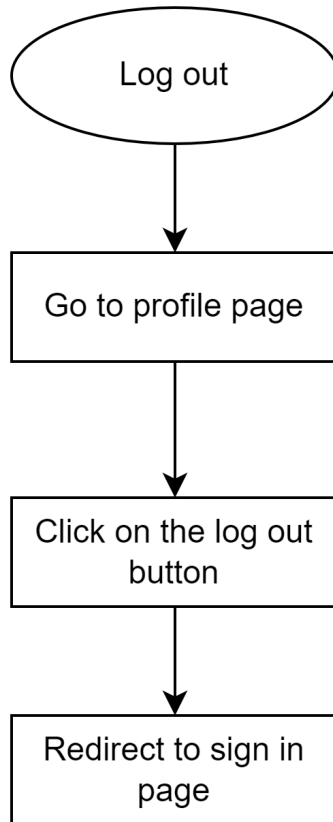
Change Password Screen

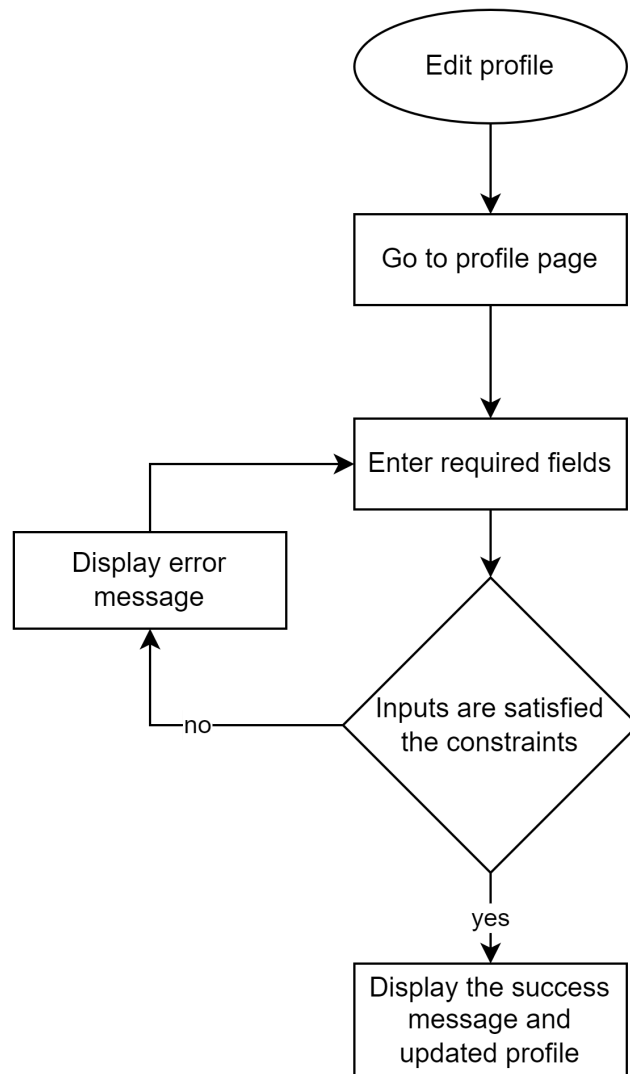
5. Flow Diagrams

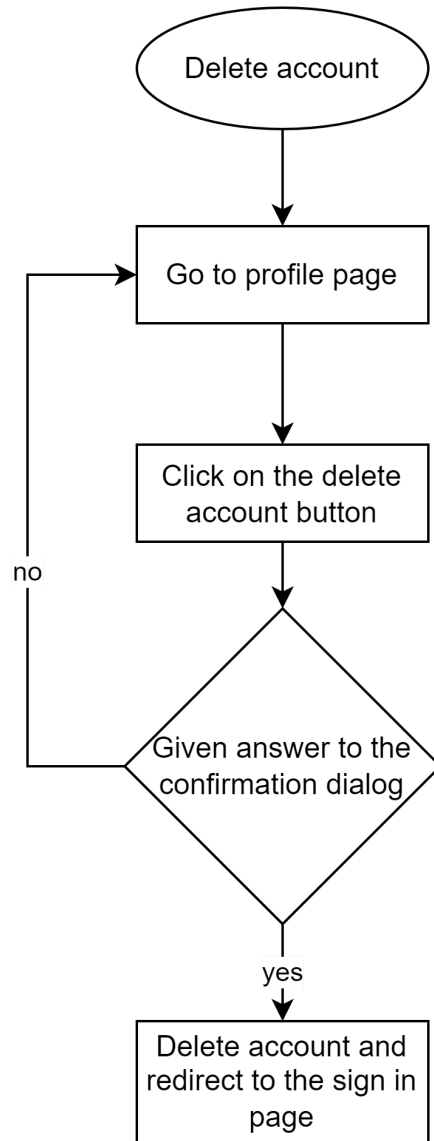
5.1 Flow Diagrams

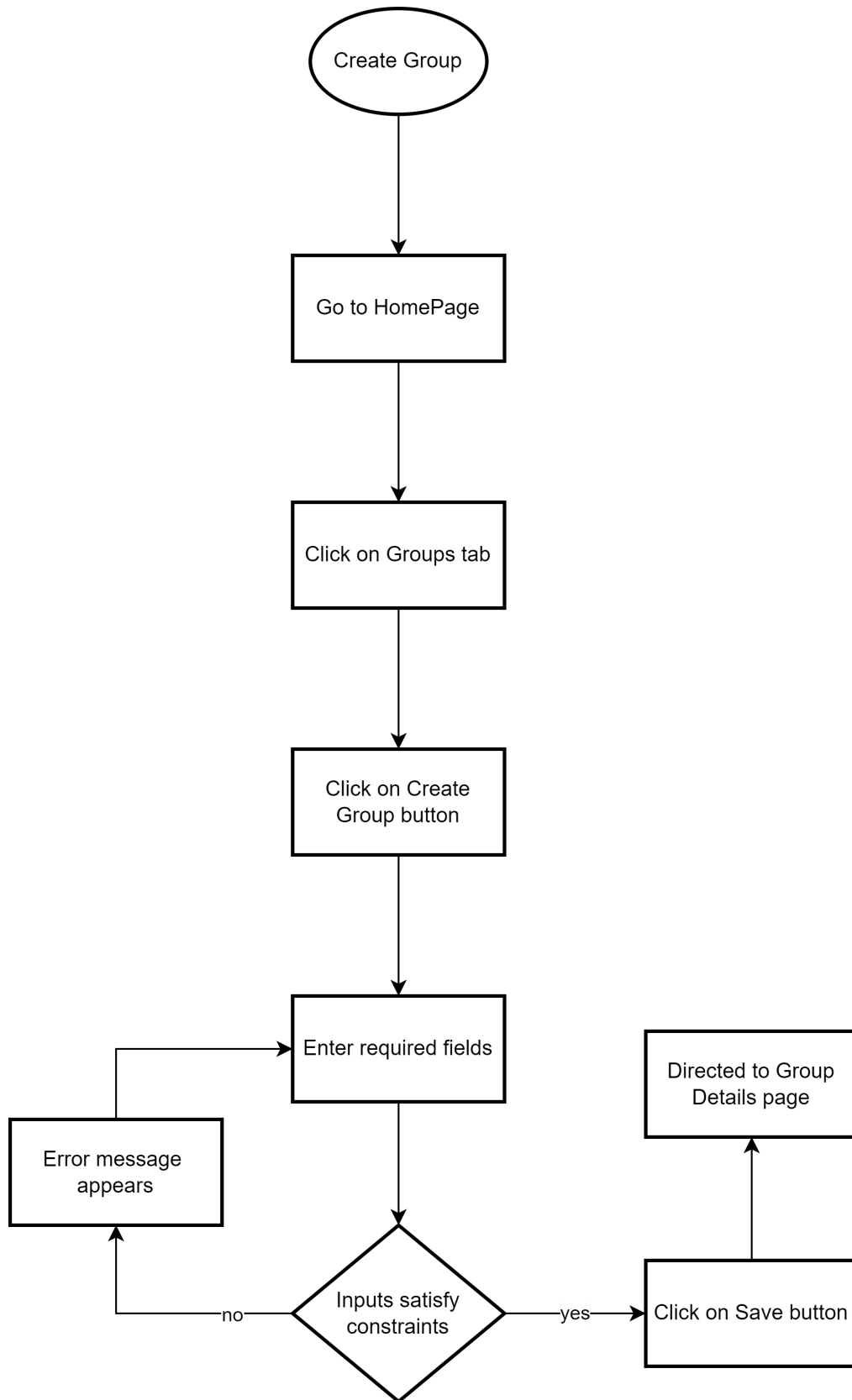


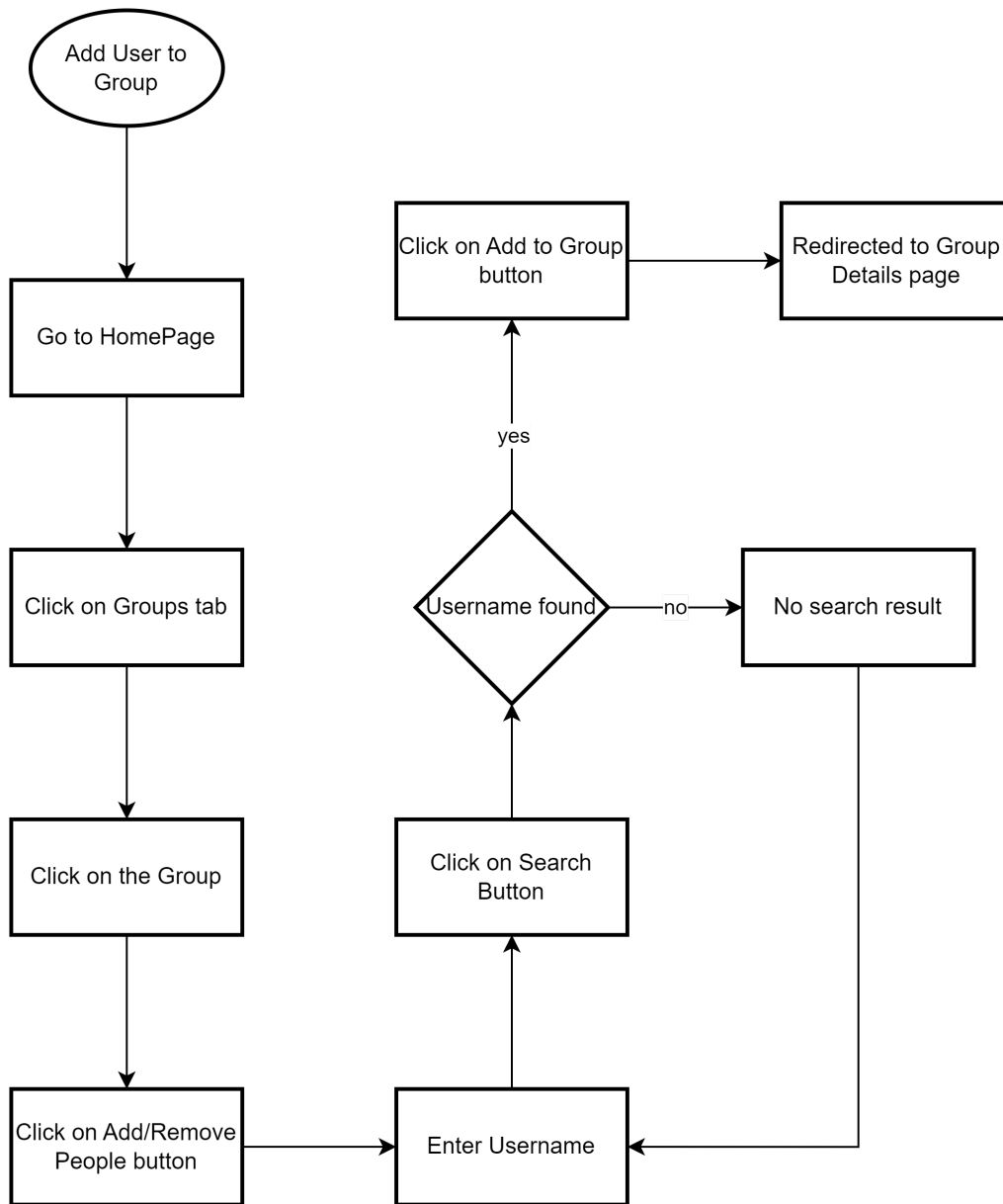


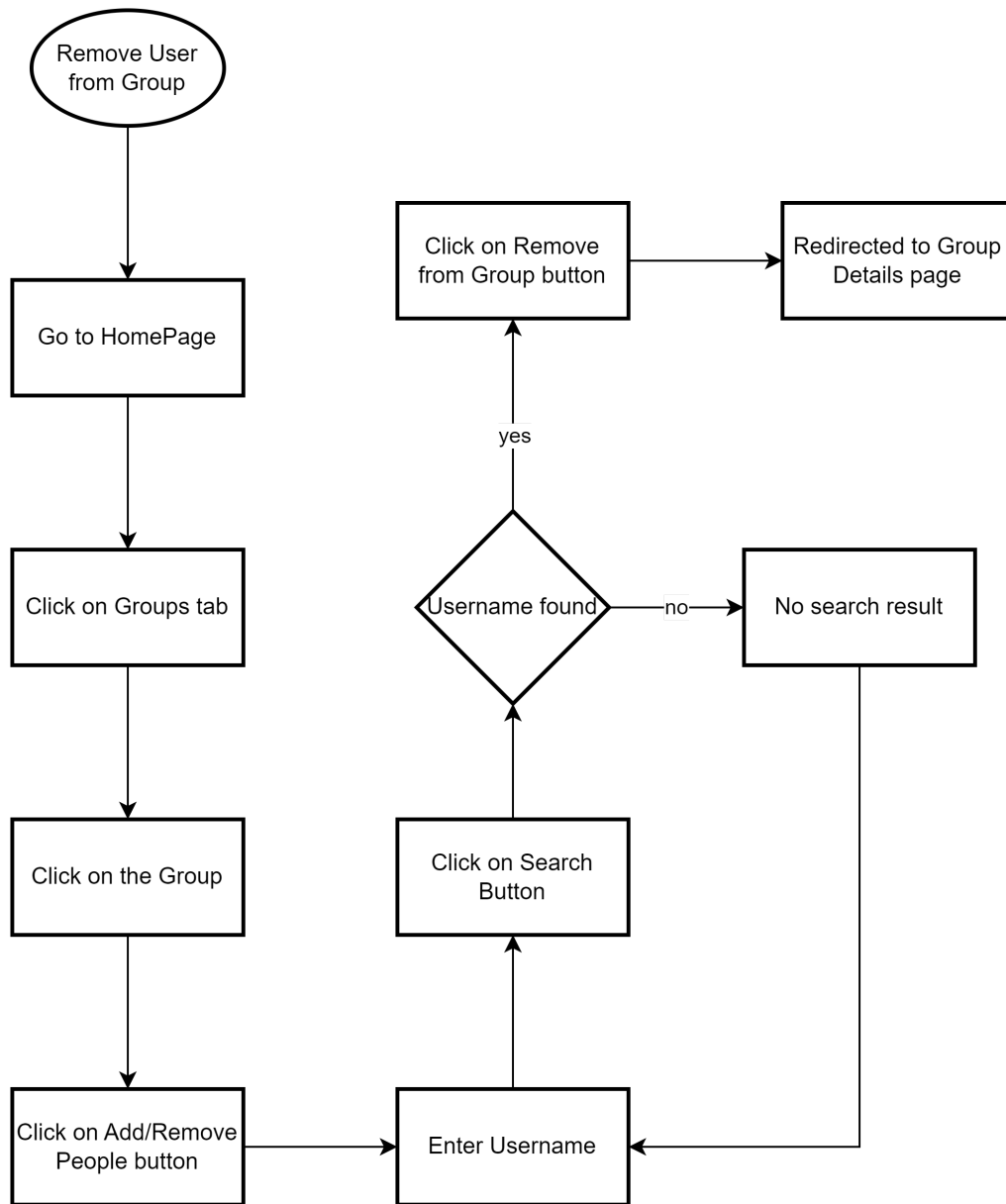


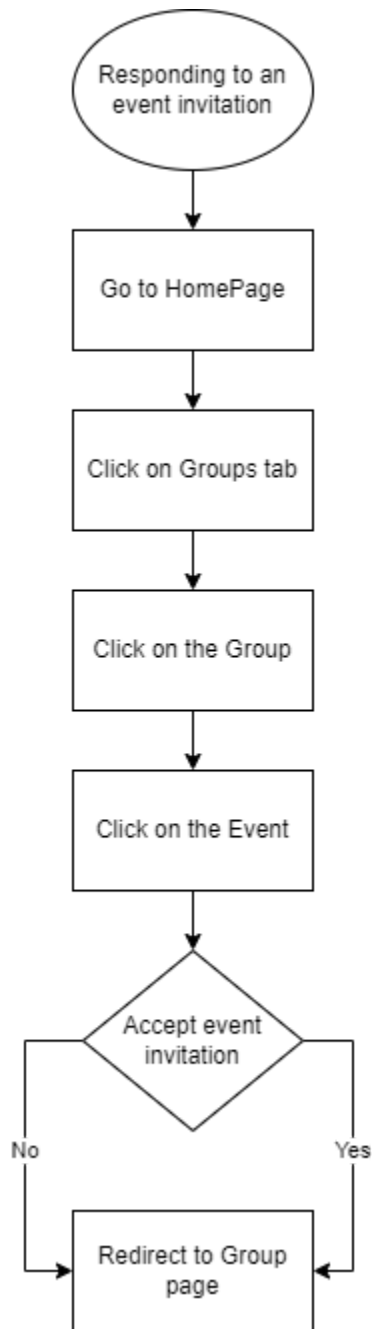


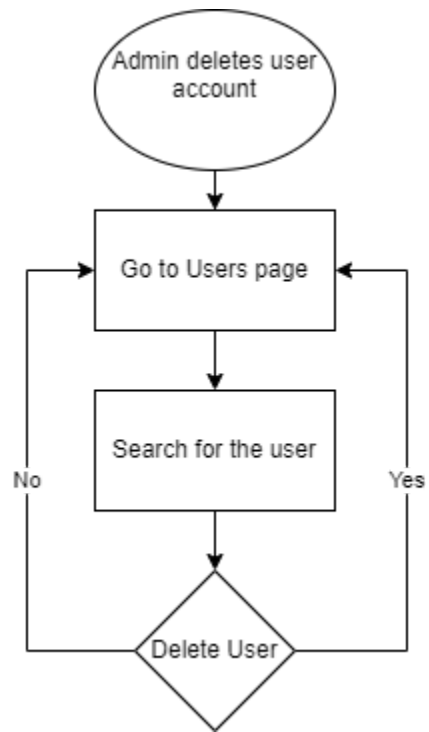


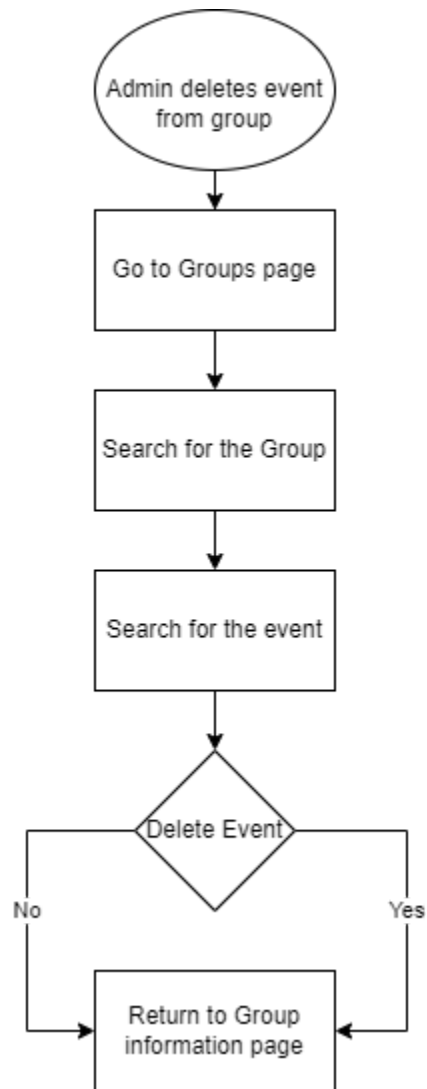


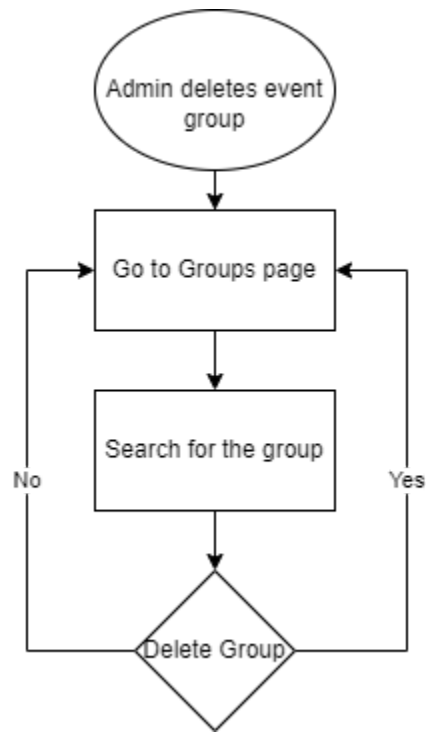


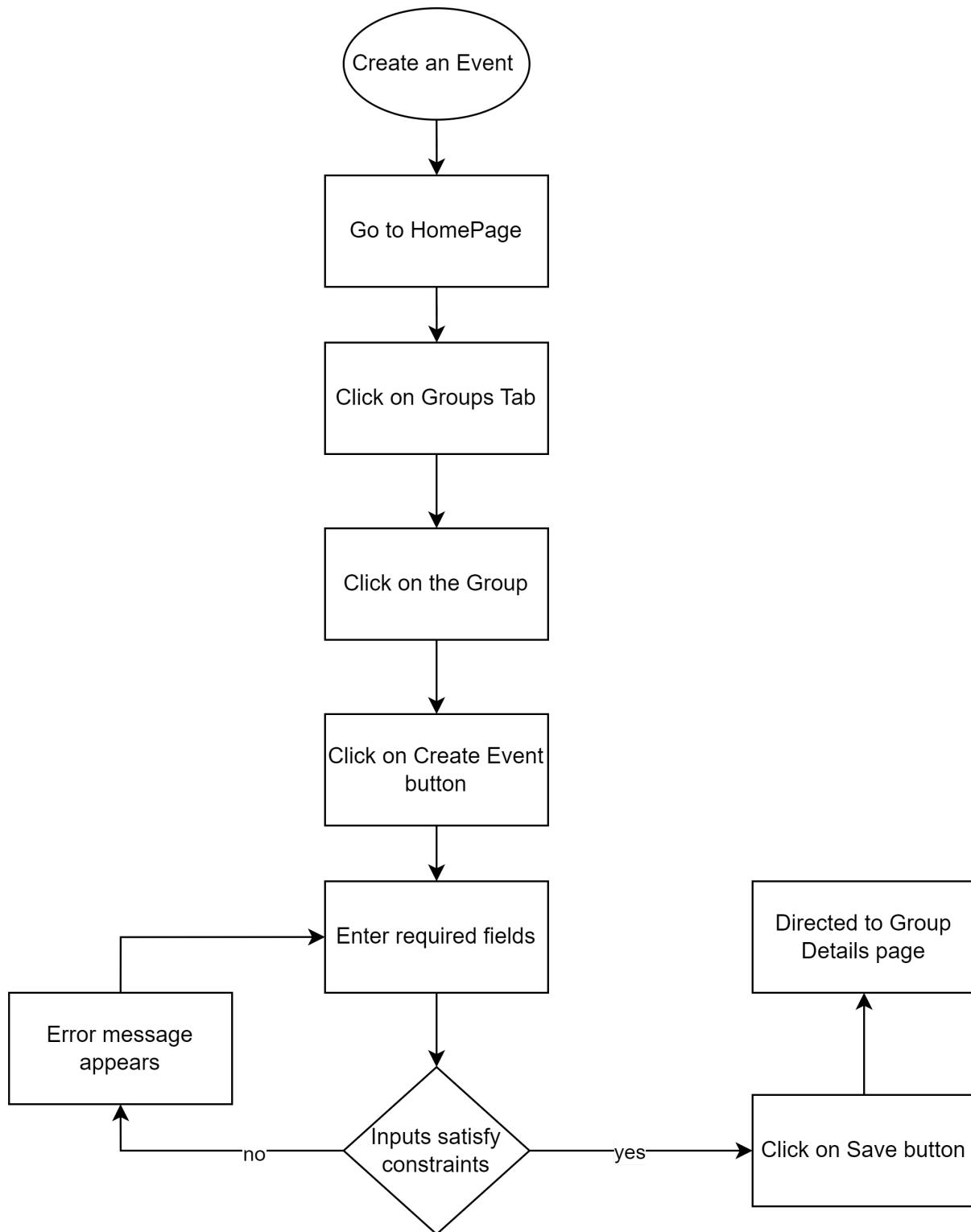




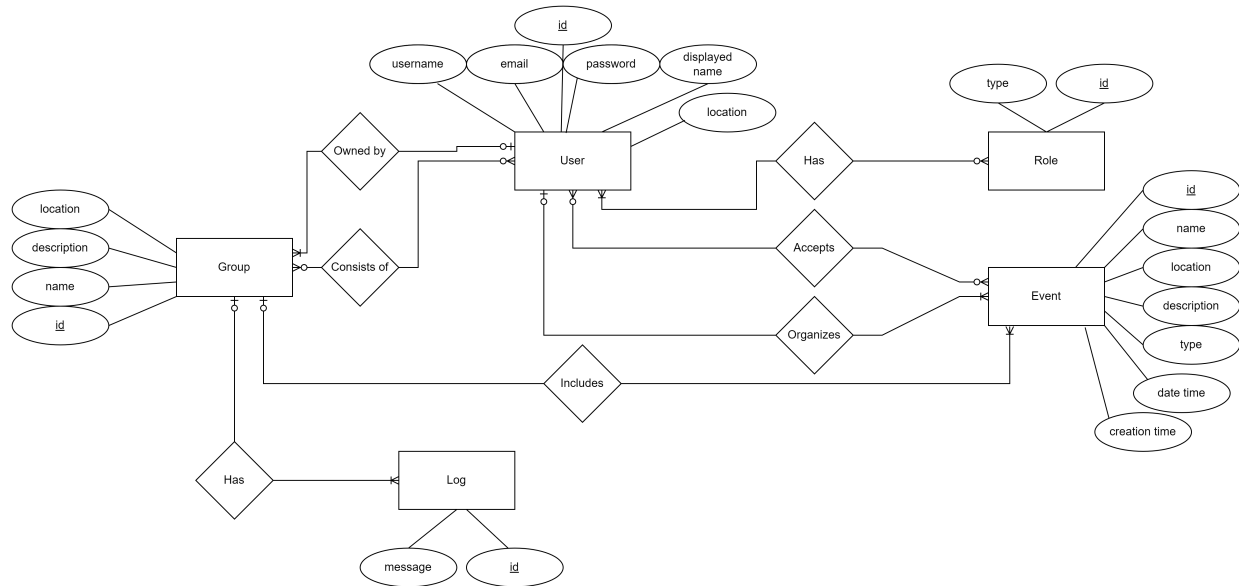








5.2 General data model



5.3 Important data considerations

For this project, PostgreSQL is well suited to store data in the database. Since we are using Spring Boot for the backend development, each record on the database must be represented as a Java class. All the transactions between the backend functions done using these classes. However, it is not possible to pass a Java object to the frontend of the application. Thus, each object record must be converted to the JavaScript Object Notation (JSON) for the transaction between backend and frontend. The JSON representations of each data model except role will be used in the frontend. The example JSON representations of the main entities are shown below.

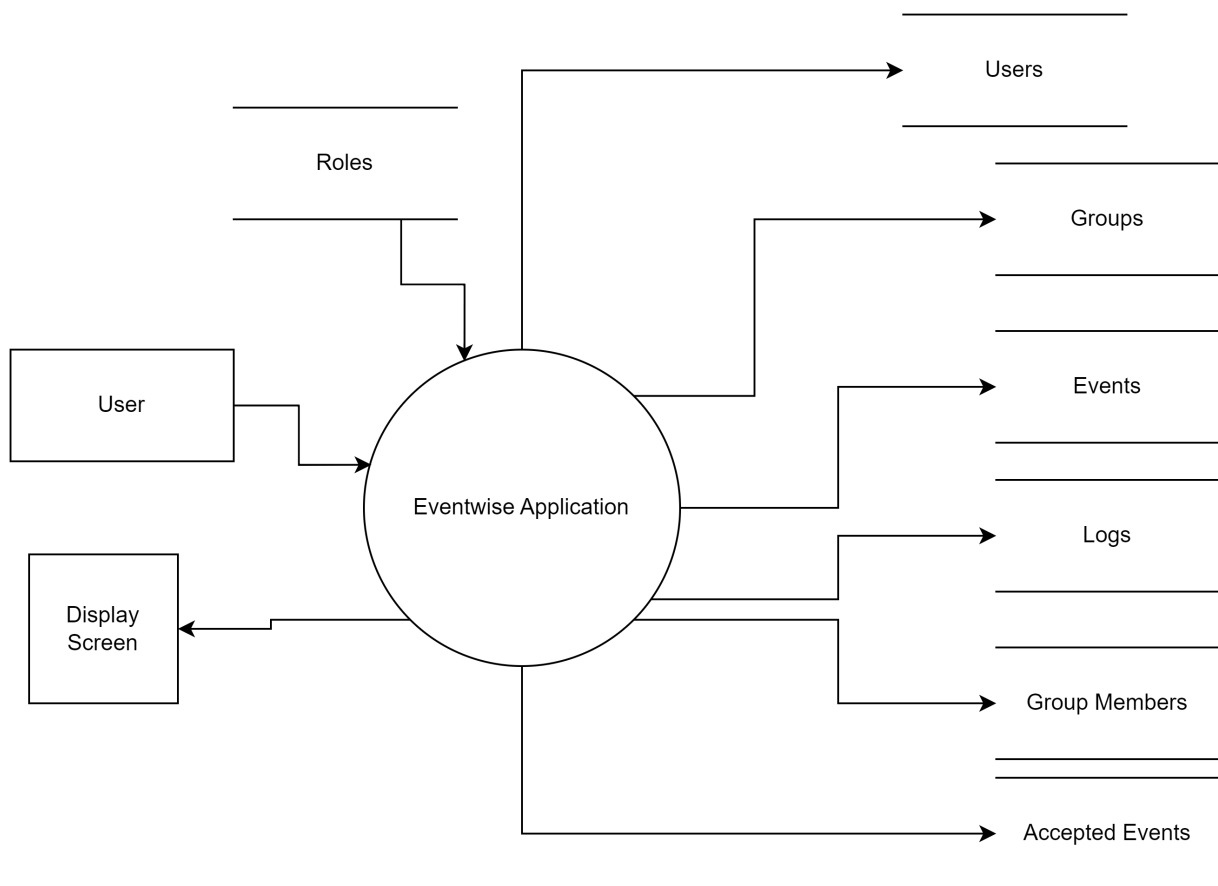
Important note: the password of the user will not be included in any transaction.

example_group	example_user	example_event	example_log
<pre>{ "id": "1", "name": "Turgut's Group", "location": "Istanbul", "description": "Open source python developer's hangout group", "owner": example_user,</pre>	<pre>{ "id": "1", "username": "Turgut99", "displayed_name": "snake_tamer", "email": "turgut@itu.edu.tr", "location": "Istanbul", "accepted_events": [example_event,...], "groups": [example_group,...],</pre>	<pre>{ "id": "1", "group":: example_group, "organizer":: example_user, "accepted_members":: [example_user,...], "name": "Monty Python Comedy Night", "location": "Istanbul", "description": "We are going to watch an episode from famous comedy show Monthy Python", "type": "watch", "date_time":</pre>	<pre>{ "id": "1", "group":: example_group, "message": "Turgut created an event named 'Monty Python Comedy Night'" }</pre>

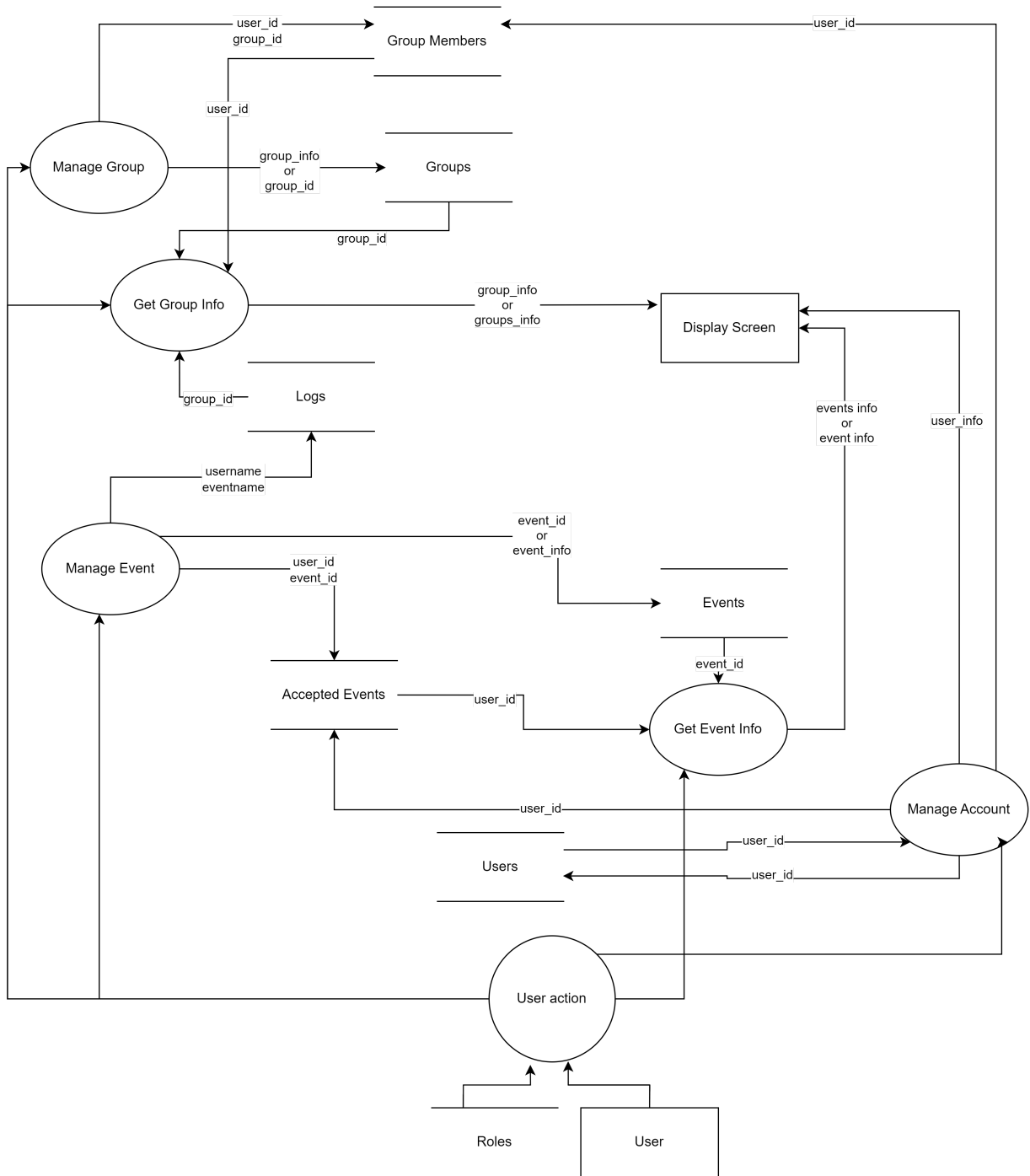
"members": [example_user, ...]}	"role": ["admin", "user"] }	"18.11.2022", "creation_time": "03.12.2008" }	
---------------------------------------	--------------------------------	--	--

5.4 Data flow

Level - 0



Level - 1



Level - 2

