# Functional Requirements

1. **Login Process**

1.1. Upon first bootup, App displays a **login page** with the following:

- Two text input fields, Email and Password
- A button labeled ‘Log In’, which when clicked, sends the signal to start the authentication
- A button labeled ‘Create Account’ for new users
- A button labeled ‘Forgot Password’

1.2. If ‘Create Account’ is chosen, app displays an **account creation page** which has the following:

- Language choice input field, a scrollable multiple-choice menu with at least four pre-defined language options. When a different option is changed, the form text changes to that language and the app text will load in that language.
- Text input field for Username with an onchange function that displays whether the username is available or not
- Two text input fields for Password and Confirm Password, with onchange functions displaying if the password meets the requirements and if the passwords match
- A text input for the user’s email
- A button labeled ‘verify email’, which on click, sends a signal to the function that sends a verification email to the entered email
- If an email already associated with the account exists, an email is sent instead directing the user to the ‘forgot password’ page
- A display that changes when the email has been verified
- A checkbox for users to accept the terms and conditions of the app
- If any input field is left empty, a warning should appear above it prompting the user to fill it up
- A ‘Create Account’ button at the bottom of the form that can only be clicked when every field has been filled, which sends the signal to load the **profile page**

1.3. If ‘Log In’ is chosen,

- If authentication is unsuccessful, the Password field is cleared and the user is prompted with a warning, ‘Username/Password is incorrect.’
- If authentication is unsuccessful five consecutive times, the user account is locked out and user must use the **forgot password functionality** to log in
- If authentication is successful, app displays an **explore page**

1.4. If ‘Forgot Password’ is chosen, app prompts user for their email, and sends the user an email to log in and change their password if an account associated with it exists.

2. **Profile Setup Process**

2.1. If this is an initial login, the user is taken on a ‘tour’ of the app where popups show them how to enter their location and preferences, and how to use the explore page. The popups disappear upon clicking on the specified input field.
2.2. The profile page provides the user an overview of their specifications (area, preferred language, activity preferences) and the groups they have joined, as well as their upcoming events.
2.3. The profile can be edited by clicking a button labeled ‘Edit Profile’. This leads the user to the **settings page**.

- In the settings page, the Area input field is a single-choice dropdown menu with a list of HDB areas in Singapore. (Note: the app will not use the user’s location to determine their area, instead letting them choose for themselves)
- The ‘Delete Account’ button will prompt users to re-enter their username to verify the deletion and warn them that the action is irreversible. When the account is deleted, the user will get an email about it.
- The ‘Deactivate Account’ button will prompt users for confirmation before executing. Upon deactivation, the user account must not be visible or active, but data should remain stored for reactivation.
- The ‘Change Password’ button will prompt users to enter their old password, and then to enter their new password twice. Once the password is changed successfully, the user will get an email about it.
- The Activity Preferences input field is a scrollable multiple-choice menu with pre-defined options as well as a ‘Create Activity’ option, which prompts the user with a **create activity form**
- The language input field is a scrollable multiple-choice menu with at least four pre-defined language options. When a different option is chosen, the app text changes to that language.
- The create activity form
  - prompts the user for the activity name and cross-checks it with existing activities, informing the user if it already exists.
  - If it does not exist, a request is sent to an administrator to accept it being added to the database. This check is to ensure no inappropriate activities are added.
  - Once the activity is created, all users can view and add it to their profile.
- The Change Username text field can be edited, with an onchange function that displays whether the username is available or not

- At the end of the page is a button labeled ‘Save Changes’. When clicked, all changes are saved and immediately reflected across the app. If no changes were made, no changes will be reflected.

2.4. In the profile page, users can view all the interest groups they have joined.

- They can click on the mini group banner to load the **interest group page**.
- Each mini group banner has a button labeled ‘Leave Group’, which when clicked, removes the user from the interest group, but does not remove the meet-ups that the user has RSVP-ed ‘Yes’ to.

2.5. In the profile page, users can also view all the upcoming meet-ups from their interest groups that they have RSVP-ed ‘Yes’ to, with the meet-up location, date and time.

- There is a button labeled ‘See More’ which takes the user to the **event page**

2.6. Every page has a sticky navbar that contains links to the Explore Page, Community Page and Profile Page.

## 3. Explore Page

3.1. Upon loading, the page displays a list of facilities

- sorted by distance if location access is available, and facilities suiting the user’s preferences are higher up

3.2. Upon loading, there is a sticky dropdown menu that allows users to search by type of facility (e.g. basketball court, playground, fitness corners, gyms)

3.3. At the top of the page, there will also be a filter option, with a toggle so users can filter facilities by distance from them

- This toggle will prompt for location access if the access has not been granted

3.4. The page loads a scrollable column of facility listings, each displaying:

- A banner picture of the facility
- The type of facility
- Distance from the user (if location access is given)
- Operating hours (if applicable)

3.5. There should be a button labeled ‘Submit new facility’

- This button, if clicked, prompts the user to fill up a form with the following input fields:
    - Type of facility: Dropdown menu
    - Location of facility: Map Pinpoint
    - The form does not accept photos as that creates room for a vulnerability in the app.
    - ‘Submit’ button
- This will send a request to an app administrator to add a new facility to the database, that all users can view.

## 4. Community Page

4.1. Upon loading, the community page has a dropdown menu that allows users to change the residential area they are viewing.
4.2. The community page loads to the user's residential area by default. If the user has not initialized their profile, the app will prompt the user to do so.
4.3. The community page loads a scrollable list of links to **group pages** for interest groups in the area, sorted by an algorithm where the groups that match the user's preferences are higher up.
4.4. The group pages contain information about the interest group such as

- What activity the group was created for (from the dropdown menu)
- Description by group administrator/creator
- Usernames of members in the group
- Upcoming **meet-up events** scheduled for the group which the user can RSVP to if they choose to join the group
- Option to join the group if the user has not joined

4.5. The user can browse interest groups from different areas and join any
4.6. The user can create their own interest group

- There will be a button labeled 'Create New Group' in the community page. Onclick, it takes the user to a form page. The form page has the following input fields:
- Activity Type: Single-choice Dropdown menu
- Residential Area: Single-choice Dropdown menu
- Group Name and Description: Text input
- Groups are approved by administrators before being created to prevent anything inappropriate.

4.7. The user can edit the details of their own created interest groups, including deleting the groups.

- There will be a button on the group page labeled 'Edit Group'. Onlick, it takes the user to a form page with the following input fields:
- Group Name and Description: Text input
- 'Submit Changes' Button
- 'Delete Group' Button
- The 'delete group' button, if clicked, will warn the user that the action is irreversible and prompt the user to re-enter the name of the group to verify the deletion.
- When a group is deleted, every member will be notified via email and notification.

## 5. Managing Meet-Up Events

5.1. Any member of an interest group can create a meet-up event.
5.2. The group page will contain a button labeled 'schedule meet-up'. Onclick, it prompts the user with a form with the following input fields:

- Date and time of meet-up (they will be warned if they have a clashing event in that group or if they have RSVPed to another clashing event from another group)
- Location of meet-up- facility selection from app's database or map pinpoint
- A short description (optional)

5.3. Any creator of a meet-up event can edit the event details by clicking a button labeled 'Edit Meet-Up' on the meet-up event page.

- Clicking the button directs the user to a form with the following input fields:
- Date and time of meet-up (they will be warned if they have a clashing event in that group or if they have RSVPed to another clashing event from another group)
- Location of meet-up- facility selection from app's database or map pinpoint
- A short description (optional)
- A 'Delete Event' button which if clicked, will warn the user that the action is irreversible and prompt the user to confirm the deletion of the event.

5.3. Every group page contains a list of upcoming meet-up events sorted by earliest to latest, and displays an option to RSV if the user has not yet

- RSVP options: Single choice- Yes or No.
- If users RSVP 'No', they will not get notified of the upcoming event.
- If users RSVP 'Yes', the event will appear on their profile page for their own reference. The user will be warned if they have a clashing event in that group or if they have RSVPed to another clashing event from another group.
- If users do not RSVP, they will get notified of the event closer to the date and time, but it will not appear on their profile.

5.4. Users can change their RSVP status by clicking on the meet-up link from their profile or the group page.

5.5. Users can view the creators of meet-ups, usernames of members who RSVP-ed and meet-up details before RSVP-ing.

6. **Notifications and Alerts**

6.1. Users are notified of upcoming events in groups they joined if they RSVP-ed 'Yes' or did not RSVP.

6.2. Users are notified when a facility they submitted, an activity preference they added or a group they created is approved.

6.3. Users are notified when a significant number of users RSVP 'Yes' to an event they created, if the event is at risk of low attendance, when users join a group they created, and if a group they joined was deleted.

6.4. In the settings page, users should be able to use a toggle to choose which types of notifications they want to receive.

## Non-Functional Requirements:

1. **Usability**

    1.1. The app must be intuitive and user-friendly, allowing users to seamlessly access multiple features and navigate easily within the app.
    1.2. When accessing features where users have to make choices, all options must be clearly shown to the user
    1.3. Need to be able to provide support with at least the 4 official languages of Singapore
    1.4. The app is required to be elderly-friendly with large fonts and high contrast

2. **Reliability**

    2.1. System should be up 99% of the time

    2.1.1. This means 7h 14m 41s of permissible downtime per month, or about 3 days 15 hours of downtime per year. This is permissible while the app is still not widely used, but as the user base grows, the uptime should increase.

    2.2. Scalability

    2.2.1. The number of HDB households was 1,108,100 in 2023. Ignoring varying household sizes, if we have one user of the app per household, that is around 1,100,000 users. Our app should support 25% of those users using the app concurrently, so 275,000 concurrent users.

    2.3. Data security: All of the users' personal information such as email, residential area, location, password, or preferences should remain Confidential and accessible to none except the user and privileged accounts to specific information for help desk purposes.
    2.4. Data integrity:

    2.4.1. None of the users' information such as email, residential area, location, password, groups or preferences should be modified.
    2.4.2. None of the information on the application databases that should be visible to app users should be modified without the permission of the administrators, such as the facility information or activity preference options.

3. **Performance**

    3.1. Upon bootup, app must load the login page (if not logged in) or explore page (if logged in) within 30 seconds.
    3.2. If a new page is called, it must load within 30 seconds.

    3.2.1. If the page has not loaded within 5 seconds, a 'loading' animation must appear.

4. **Supportability**

    4.1. Databases must be replaceable by any others that support standard SQL queries
    4.2. The app must be compatible with major web browsers
    4.3. Maps must be in GEOJSON

# Atomic Requirements

## Login Page

**User Authentication:**

1.1 The system must prompt the user to log in with their email and password upon accessing the app.

1.1.1 The email input must be text of at least 5 characters and a maximum of 50 characters.

1.1.2 The password must be at least 12 characters, including at least one uppercase letter, one lowercase letter, one numeric digit, and one special character.

1.2 The system must allow users to create a new account if they do not have one.

1.2.1 The system must verify the uniqueness of the email and username provided by the user.

1.2.2 The system must send a verification email to the user's email address to confirm account creation.

1.2.3 The password must be validated for strength based on the criteria specified in 1.1.2.

1.3 The system must provide a password recovery option.

1.3.1 Users must enter their registered email address to receive a password reset link.

**User Profile Initialization:**

2.1 Upon creating a new account, the system must prompt the user to enter their area and activity preferences.

2.1.1 The area input must be selected from a predefined list of HDB areas.

2.1.2 The activity preferences input must be selected from a predefined list of sports activities (e.g., badminton, football).

2.1.3 Users must select at least one activity preference.

2.2 Users must be able to propose a new activity if their desired activity is not present in the predefined list.

2.2.1 The system must provide an input field for the proposed activity name, which must accept text between 3 and 100 characters.

2.2.2 The system must notify the administrator of the proposed activity for approval.

2.2.3 The proposed activity must not appear in the list until it has been approved by the administrator.

2.2.4 Upon approval, the new activity must be added to the predefined list and become available for all users to select.

2.3 Users must be able to update their profile information at any time.

2.3.1 Profile information must include the area, preferred language, and list of activity preferences.

2.3.2 Changes to any profile information must be saved and reflected immediately.

## Explore page

**Explore Facilities:**

3.1 The system must display nearby facilities based on the user's current selected Area of Residence.

3.1.1 The facilities data must include the facility name, type, distance from the user's location, and operating hours.

3.1.2 The system must allow users to filter facilities by name, distance (e.g., within 1 km, 5 km) and by type (e.g., playgrounds, fitness corners).

3.2 The system must load the explore page with facility data within 5 seconds of the user's request.

3.2.1 If the loading time exceeds 5 seconds, the system must display a loading animation or message to inform the user.

## Community Page

**Interest Group:**

4.1 Users must be able to create a new interest group.

4.1.1 The system must provide an input field for the group name, which must accept text between 3 and 100 characters.

4.1.2 The system must display a dropdown menu for users to select an activity type (e.g., badminton, yoga, reading).

4.1.3 The system must display a dropdown menu for users to select the area (e.g., Boon Lay, Tampines) where the group is based.

4.1.4 A "Create Group" button must be available to confirm the creation of the group.

4.1.5 Upon successful creation, the group must be added to the user’s profile and visible in the list of available groups.

4.2 Users who created the group must be able to edit the details of an existing group.

4.2.1 The system must provide input fields for editing the group name, which must accept text between 3 and 100 characters.

4.2.2 The system must display dropdown menus for editing the activity type and the area where the group is based.

4.2.3 A "Save Changes" button must be available to confirm the updates to the group.

4.2.4 Upon successful editing, the updated group details must be reflected in the user’s profile and the list of available groups.

4.2.5 The system must validate the edited group name, activity type, and area before saving changes.

4.3 Users who created the group must be able to delete an existing group.

4.3.1 The system must prompt for confirmation and require the user to re-enter the name of the group before permanently deleting a group.

4.3.2 Upon confirmation, the system must remove the group from the user’s profile and the list of available groups.

4.3.3 The system must notify all group members of the deletion via email and notification.

**Joining and Leaving Interest Groups:**

5.1 Users must be able to view a list of existing interest groups on the page.

5.1.1 Each listed group must display the group name, activity type, and location.

5.2 Users must be able to join a group by clicking the "Join Group" button in the group page.

5.2.1 Once joined, the group must appear in the user’s profile under “My Groups.”

5.2.2 A "Leave Group" button must be available in the user's group listing to allow users to leave a group.

5.2.3 Upon leaving a group, the system must remove the group from the user's profile and stop any associated notifications.

**Meet-Up Scheduling and Management:**

6.1 Users within a group must be able to create a new meet-up event.

6.1.1 The system must provide input fields for the meet-up date (in DD/MM/YYYY format), time (in 24-hour HH format), and location (text input or selection from a list). The user will be warned if they have a clashing event in that group or if they have RSVPed to another clashing event from another group.

6.1.2 Users must enter a brief description of the meet-up (at least 10 characters, max 256 characters).

6.1.3 A "Schedule Meet-Up" button must be available to finalize and save the meet-up.

6.1.4 Once scheduled, the meet-up must be visible under the group’s meet-up list with details.

6.2 Users must be able to vote on their attendance for each meet-up.

6.2.1 A voting section must be displayed below each meet-up listing, allowing users to select one of the two options ("Yes," "No"). Any user voting “Yes” will be warned if they have a clashing event in that group or if they have RSVPed to another clashing event from another group.

6.2.2 Users must be able to change their vote at any time before the meet-up starts. The most recent selection will be recorded.

6.2.3 The system must display the total number of votes for each option (e.g., 5 Yes, 2 No).

6.2.4 Users should be able to see a list of members who have voted "Yes" to attend the meet-up.

6.3 Users must be able to modify or cancel meet-up events created by them

6.3.1 A "Modify Meet-Up" option must allow users to change the date, time, and location of events created by them.

6.3.2 Any modifications must be saved and updated in real-time for all group members.

6.3.3 A "Cancel Meet-Up" button must allow users to cancel the event, which will remove it from the schedule.

6.3.4 The system must send notifications to all group members when a meet-up is scheduled, modified, or canceled.

**Notifications and Alerts:**

7.1 The system must send a notification to users when a new group is created in their area of residence if the user has agreed to receive notifications of this type.

7.2 Users must receive notifications for upcoming meet-ups from their joined groups, changes to meet-up details if they have RSVPed ‘Yes’, and relevant cancellations.

7.3 Users must be notified when a significant number of attendees vote "Yes" or if the meet-up is at risk of low attendance.

7.4 Notifications must be displayed within the app and sent via email if the user has opted into email notifications.

**Settings page:**

8.1 Users must be able to access the Settings Page from the main navigation menu.

8.1.1 A "Settings" icon or link must be clearly visible and accessible from all primary pages (e.g., profile, explore, group management)

8.2 Account Management: Users must be able to view and update their personal information, including: Name (text input, 3-50 characters), Email address (valid email format, e.g., user@example.com).

8.2.1 If the user edits the email, the system must verify the uniqueness of the email provided by the user. If the user edits the username, the system must verify the uniqueness of the username provided by the user.

8.2.2 The system must send a verification email to the user’s email address to confirm account creation.

8.3 Users must be able to change their password from the Settings Page. A "Change Password" button must be available, prompting the user to enter the current password and new password.

8.3.1 The system must verify the current password and validate the new password against the strong password policy before saving changes.

8.3.2 Users must receive a confirmation notification or email upon successfully changing their password.

8.3.3 Users must be able to update their area of residence. The system must provide a dropdown menu or search functionality to select or change their area (e.g., Boon Lay, Tampines).

8.3.4 Users must be able to change their activity preferences. - The system must provide checkboxes or dropdown lists for users to select or deselect their preferred sports activities (e.g., gym, football).

8.4 Notification Settings: Users must be able to manage their notification preferences for different types of events (e.g., new group creation, upcoming meet-ups, meet-up changes).

8.4.1 The system must provide toggle switches or checkboxes to enable/disable notifications for each event type.

8.4.2 Users must be able to opt in or out of email notifications. - A checkbox must be available to enable or disable email notifications. The default setting should be enabled.

8.5 Language Preferences: Users must be able to select their preferred language from a list of supported languages (e.g., English, Mandarin, Malay, Tamil).

8.5.1 A dropdown menu must be provided to select the language. The system must update the language throughout the app interface according to the user's selection.

8.6 Deactivation and Deletion of Account: Users must be able to deactivate their account temporarily or permanently delete their account.

8.6.1  A "Deactivate Account" button must be available, providing a confirmation prompt before proceeding. Upon deactivation, the user account must not be visible or active, but data should remain stored for reactivation. The system should then send an email to the user informing them of the deactivation.

8.6.2. A "Delete Account" button must be available, providing a confirmation prompt and asking the user to re-enter their username before proceeding with permanent deletion. Upon deletion, all user data must be permanently removed from the system. The system should then send an email to the user informing them of the deletion.