Software Requirements Specification

for

PeakVisor

Version 1.0 approved

Prepared by:

Lim Yong Kang	U2221118J
Jeannie Wong Yi Lin	U2223262A
Brendan Yap Ming Thye	U2221347F
Chong Choy Jun	U2222812A
Yew Jia Le Benjamin	U2222923J
Lucas Ng Wei Jie	U2220046K

Nanyang Technological University, <PeakVisor>

April 14, 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Brendan	March 1	Drafting of the document	v0.1
Brendan	March 30	Adding new sections	v0.2
Brendan	April 9	Updated first draft	v0.3
Brendan	April 11	Updated second draft	v0.4
Brendan	April 14	Completed third draft	v0.5

1. Introduction

1.1 Purpose

This Software Requirement Specifications (SRS) document is intended for the **PeakVisor** web application [Version 1.0]. The purpose of this document is to explain the user requirements and specifications for the application, to better facilitate the development and production process for all stakeholders involved, and to describe how **PeakVisor** is expected to perform. The document will cover the functional and non-functional requirements, system features, potential limitations, use case and test case descriptions, and a breakdown of the software architecture.

The **PeakVisor** web application is a one-stop community platform that allows users to learn more about Singapore's nature trails and aims to help hiking enthusiasts and outdoor lovers keep track of their hikes and connect through community events aggregated from different platforms.

1.2 Document Conventions

This section describes all standards or typographical conventions used in this document.

The following fonts have been used in the writing of this document.

	Font	**Font Size**	**Font Weight**
Level 1 Heading	Times	18	Bold
Level 2 Heading	Times	14	Bold
Level 3 Heading	Times	11	Bold
Content	Arial	11	-

Appendix A: Data Dictionary delineates a glossary of specific terminology used in this document to resolve any ambiguity for all stakeholders.

Priority of Requirements: The priority for higher-level user requirements are assumed to be inherited by detailed requirements unless explicitly stated otherwise.

1.3 Intended Audience and Reading Suggestions

This section describes the different types of readers that the document is intended for, the contents of the document, and a suggested sequence for reading the document.

The document's intended audience and intended reading outcomes are described below.

Target Group	Reading Outcome(s)
Stakeholders	To provide a reference point for aligning objectives, resolving differences, and reviewing changes for the realization of a high-quality product with the user requirements and project objectives in mind.
Developers	To provide a single source of truth for developers to communicate and collaborate on software changes, technical analysis of user requirements, and implementation of system features.
Project Manager(s)	To provide a reference point for ensuring user requirements are met and translated appropriately into the software, and that the end product is delivered in a timely manner.
Testers	To provide a reference point for rationale behind the test cases, the specific system features involved, a step-by-step guide for producing each test case, and the expected outcomes for each test.
Marketing Staff	To provide a reference point for the intended objectives, target groups, use cases, and cost-benefit analysis that will help to better facilitate marketing of the application towards its intended end users.
Documentation Writers	To provide the basis for the entire project, including system features, interfacing considerations, design specifications, and use case outcomes that help with writing good software documentation.

The document begins with the introduction, which states the intended objectives of the document, the target audience and project scope. It is followed by the high-level overall description of the application that provides perspective on the project's background, a summary of software functionalities, design constraints, and assumptions made. It is prefaced by the external interfacing requirements for the application before providing a comprehensive description of the key system functionalities and other non-functional requirements.

Generally, the document is to be read in sequential order. All stakeholders are recommended to begin from the *1. Introduction*, notably *1.1 Purpose* to familiarize themselves with the SRS objectives, *1.3 Intended Audience and Suggestions* to understand the intended reading outcomes for their target group, and *1.4 Product Scope* for a high-level understanding of the project's goals. They should then proceed to *2. Overall Description* for the product perspective and a high-level summary of the application's features.

Sections *3. External Interfacing Requirements* and *4. System Features* provide a technical overview of the required external interfaces and system features that serve more as a point of

reference for the development team and testers when interacting with the application. Nevertheless, other stakeholders may read through these sections as they please.

1.4 Product Scope

This section provides a short description of the background, purpose, benefits, objectives, and goals of the **PeakVisor** application.

Singapore is a highly-urbanized city-state, but with an abundance of nature reserves, parks, and trails, it is not unusual that hiking is seen as a convenient, accessible, and often cost-free getaway for people in the city. In support of Singapore's vision to become a "City in Nature", the nation's hiking infrastructure has undergone groundbreaking transformations under the National Park Board's directions, which has provided the opportunity for people from all walks of life to engage with nature and for outdoor activities to flourish. Furthermore, this has also opened new doors for individuals with similar interests to connect through shared events on nature trails, and has helped to foster stronger community bonds.

Nevertheless, despite the "hiking boom" there is no centralized hub specifically made for people to gather to explore new locations, connect with others, and participate in shared experiences. Currently, there are online blogs and social media groups for hiking communities, but none are specifically engineered with the user experience of Singapore's hikers in mind.

As such, our team envisions a community platform where users can plan, track, relive, and share their hiking experiences, interact more intimately with Singapore's hiking terrain, and connect with informed individuals to form a larger community that can help enliven appreciation for our nature trails.

The **PeakVisor** web application aims to act as a one-stop hiker-friendly platform that consolidates nature trails and hiking events from across the nation. Users can manage their hiker profiles, seamlessly track and organize the trails they have visited, view shared reviews of nature trails, and sign up for community hiking events, all through our platform.

With our application, we hope to create a seamless and enjoyable hiking experience for all.

1.5 References

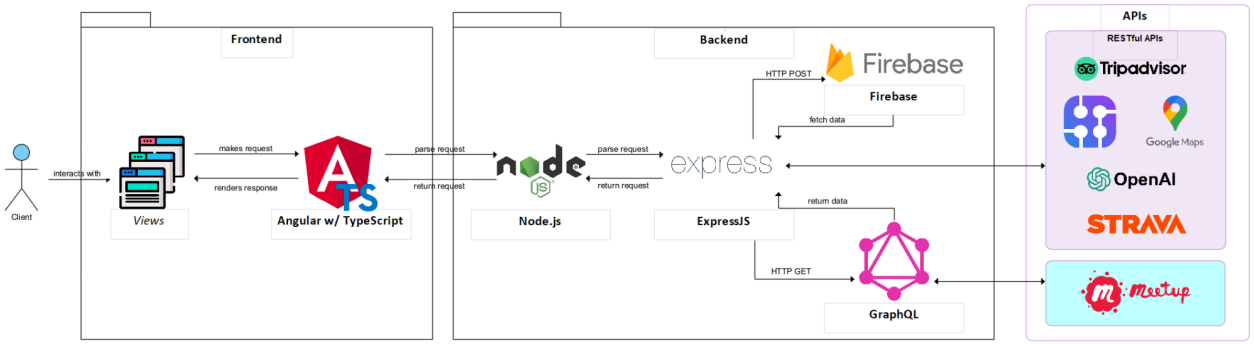
- TripAdvisor API: https://tripadvisor-content-api.readme.io/reference/overview
- Meetup API: https://www.meetup.com/api/general/#graphQl-introduction
- Google Maps API: https://developers.google.com/maps/documentation
- Strava API: https://developers.strava.com/docs/reference/
- SerpApi: https://serpapi.com/search-api
- Firebase API Reference: https://firebase.google.com/docs/reference/js
- Node.js Documentation: https://nodejs.org/docs/latest/api/
- Angular Documentation: https://angular.io/docs
- PeakVisor Source Code (Github): https://github.com/ykIsCoding/PeakVisor

2. Overall Description

2.1 Product Perspective

PeakVisor is a standalone and self-contained web application that is newly-developed to benefit hiking enthusiasts, outdoor lovers, and beginners alike to find, manage, and share their hikes and join community events at nature trails across Singapore. The application interfaces with multiple APIs, including TripAdvisor API to retrieve user reviews on Singapore's hiking spots, Meetup API for consolidating real-time updates on community events across nature trails, and Google Maps API to allow users to view the location of these trails.

A system diagram that shows an overview of the main components of the system, subsystem interconnections, and external interfaces is provided below.



2.2 Product Functions

A high-level summary of the major functions of the web application is provided in this section and can be divided into four sub-categories.

Category	*Description*
Account Management	Our users will be able to manage their saved hiking trips, their history of completed hiking trails, and hiking data retrieved from their linked Strava accounts, in addition to basic account management and security features.
Hiking Profile	Our users will be able to seamlessly manage their hiking profiles after connecting to their Strava account, which includes comprehensive hiking statistics and their saved hiking trips.
View Trails	Our users will be able to select from a variety of nature trails and view information on the trail, including a concise description, the location of the trail, and user ratings retrieved via TripAdvisor, so that they can make more informed decisions before embarking on a hike.
Join Events	Our users will be able to select from a variety of community events, view information on the event (e.g., location, dates), and sign up for the event via Meetup to connect with other outdoor lovers in the area.

.
The functions of the application that belong to each sub-category are listed below.

1. **Account Management**
    - Users can create a new account on the platform.
    - Users can log into their registered accounts.
    - Users can edit their profile details.
    - Users can initiate a password change if they forget their password.
    - Users can delete their account.
    - Users can view their own hiking trips and related statistics on their profiles.

2. **Hiking Profile**
    - Users can view a concise summary of their hiking statistics, such as distance traveled and number of hiking trips.
    - Users can view a list of their saved hiking trips.

3. **View Trails**
    - Users can view a list of available nature trails.
    - Users can select a nature trail to view its related information, including a description, location, and user ratings.

4. **Join Events**
    - Users can select a community event to view its related information, including a description, location, date, and user ratings.
    - Users can sign up for a community event on an external website.

2.3 User Classes and Characteristics

The **PeakVisor** all-in-one nature trails and hiking event web application is specifically designed to cater to the needs of experienced hikers / outdoor lovers, and beginners who are fresh to the hiking scene, and takes advantage of the flexibility, convenience, and accessibility of digital platforms in the hopes of fostering an appreciation for Singapore's nature and the hiking activity for all.

As such, the web application is equipped with a multitude of useful features to simplify the process of exploring new nature trails and hiking spots for both of these user groups alike, and also seeks to foster connectivity among those in the community through events. For users who might want to use our platform as a means to keep track of their hiking statistics, we also allow them to save their own hiking trips and connect their profiles to Strava, a social network for tracking exercise. Furthermore, we recognize that hiking is a versatile activity enjoyed by people from all age groups, and as such we have designed the application interface to be as straightforward to use as possible.

Below, we summarize the user classes for our application based on frequency of use, subset of product features projected to be most used, level of technical expertise, and hiking experience.

A. Experienced Hikers

Category	***Description***
Frequency of Use	Medium
Features Most Used	● Searching for and joining community events ● Checking on hiking statistics and updating saved trips
Technical Expertise	Low
Hiking Experience	These users have considerable experience and have been accustomed to many of our nation's nature trails. Our application will be of use to them in tracking their hiking statistics and connecting with other like-minded hikers via events they can find through our website.

B. Beginner Hikers

Category	***Description***
Frequency of Use	High
Features Most Used	● Searching for nature trails, filtered by their location ● Viewing information on nature trails.
Technical Expertise	Low
Hiking Experience	These users have little or average experience and may be on the lookout for more exciting nature trails. As such, our application will be useful to them in finding and assessing new trails to embark on.

2.4 Operating Environment

The web application incorporates responsive web design to support viewports in all modern web browsers (Google Chrome, Microsoft Edge, Firefox, etc.) and mobile devices across many operating systems (Windows, Linux distributions, Android OS, iOS, etc.), with no derivations in visual behavior or runtime performance. In other words, it is designed to adapt to different screen sizes and resolutions to provide a consistent and seamless viewing experience across platforms.

The web application employs the HTTPS protocol for secure encrypted communication between client and server, providing integrity and confidentiality. Furthermore, the web application requires a stable internet connection and is unable to operate in an offline environment.

The table below describes the development environment of our web application, namely the web frameworks, libraries, and backend components used.

Components		***Description***	***Version***
Frontend	Angular w/ Typescript	Angular is an open-source web development framework built on Typescript that emphasizes scalability and maintainability through the use of components and reusable logic modules, which allows for flexibility in designing and building interactive user interfaces.	17.2.0
	PrimeFlex	PrimeFlex is a lightweight CSS library for building responsive grids and UI components. It is used to style PrimeNG components and ensure responsiveness and flexibility in our web application.	3.3.1
	PrimeNG	PrimeNG is a UI component library for Angular, offering a rich set of buttons, forms, tables, etc.. It enhances our web application with rich and visually-appealing features and is used alongside PrimeFlex.	17.5.0
Backend	GraphQL	GraphQL is an open-source data querying language for APIs. It is used to submit real-time requests to query data selectively from API endpoints to support the main features of the web application.	16.8.1
	Firebase	Firebase is a backend cloud computing service hosting database and authentication services. It is used for its robust security and authentication features and to store our users' login credentials.	10.9.0
	Node.js	Node.js is a Javascript runtime used for server-side scripting and handling HTTP requests and responses.	20.11.1
	ExpressJS	ExpressJS is a backend web application framework that specializes in building REST APIs for Node.js applications, particularly setting up API endpoints for our web application.	4.18.2

2.5 Design and Implementation Constraints

The development of the web application presented some design and implementation constraints that might limit the options available to developers.

External Interfaces

The application is highly dependent on the data fetched from external applications such as Strava, TripAdvisor, and Meetup to support its features. Interfaces to other applications must be integrated with modularity and scalability in mind, and this is accomplished with ExpressJS routes that interact with external APIs in a clear and structured way.

However, the application relies on free services due to current budget considerations, and this might present some constraints:

(1) The free version of Strava API is used to link Strava user accounts and retrieve hiking data for each user, but as a free service it imposes a rate limit on the number of requests that can be sent for a given time period, and as such requires careful management of API calls to not only meet timing requirements, but also to support future scalability and maintainability of our application.

(2) SerpApi, a real-time Google Search API, is used to scrape image data of nature trails from search results on the web, but with a limit of 100 API calls per day. As the application is built with a REST API architecture, separate calls have to be performed to fetch data per nature trail. To accommodate for this constraint, we have temporarily disabled images on the Trails page, as fetching of image data requires multiple calls per page load and would exceed the API call limit after a few page refreshes.

Database Constraints

The application uses Google Firebase, a backend database hosting service, to fetch images of nature trails that are stored on the cloud. The Firebase cloud instance is initialized from scratch and experiences a “cold start” where it takes some time to warm up. This affects image loading times on the application, but as this only occurs during Firebase start-up, we see it as a temporary constraint on overall performance.

Additionally, Firebase is used to store user credentials securely, such as emails and passwords. However, Firebase policy states that any changed email needs to be verified through a multi-step process, which is an added layer of complexity to the application and prevents us from developing a function for users to request an email address change at this moment.

Regulatory Policies

As a development team based in Singapore, we are required to follow regulatory policies and web design guidelines mandated locally. These include data protection policies, such as the Personal Data Protection Act (PDPA) which restricts the collection and usage of user data, the Web Content Accessibility Guidelines (WCAG) which account for the inclusivity of persons with disabilities in web content.

To adhere to these regulatory policies, the development stack must be reassessed regularly, and the application needs to accommodate specific design principles, security protocols, and access control measures to safeguard user data.

Language Requirements

The application is currently only available in the English language, but we recognize that there is a need for the website content to accommodate for the linguistic diversity of our target user groups. Looking forward, the application needs adequate extensibility to cater for multilingual support, which will impose additional constraints on development.

Design Conventions

We have imposed the following programming and design standards for the application.

Category	***Standards***
Programming	● The frontend pages must be split into dedicated subfolders for better management. Each folder consists of a HTML file, a CSS file, a TypeScript file, and a testing file. ● Use semantic HTML to reflect the function of a component. For example, <app-button> is used for buttons, <input> for text input fields, etc.. Additionally, enclose component groups into blocks, each with a single responsibility. ● Use proper indentation and commenting to improve code readability and understanding of each component's functions by other developers. ● All functions and non-class variables must adopt the camelCase convention, while class names and class variables should adopt the PascalCase convention. ● Components that are frequently used across pages must not be duplicated. Instead, common functionality must be converted into reusable modules and placed into a shared folder that can be accessed by all frontend pages.
Design	● All pages must adopt the shared navigation header bar to allow users universal access to the main functionalities. ● The website's theme layout, including elements such as the color scheme, fonts, and logos, must remain consistent throughout all pages. ● Visual feedback must be provided to users whenever a correct/erroneous action is performed. This must be in the form of pop-up messages or warning text.

2.6 User Documentation

The demonstration video, which showcases the environment setup process, the flow of the application, and the system features will be delivered alongside the codebase.

Additionally, the *README.md* file provided in the codebase is a step-by-step manual that includes specific instructions for setting up the development environment and launching the website on local machines. A summary of the contents of the *README.md* file will be listed here for clarity.

Developer Use Manual

Step 1: Prerequisites

The developer is required to install the following libraries before running the project.

- Git
- Angular w/ Typescript
- Node.js & npm
- Express.js

Step 2: Cloning the repository

1. Open your terminal / command prompt.
2. Navigate to the directory where you want to clone the repository.
3. Clone the repository using ***git clone*** ***https://github.com/yklsCoding/PeakVisor.git***

Step 3: Installing dependencies

1. Open your terminal / command prompt.
2. Navigate to the *PeakVisor* directory i.e., using ***cd PeakVisor***.
3. Install the required dependencies for the application using ***npm install***.

Step 4: Setting up the backend

First, set up the *backend*.

1. Open a new terminal / command prompt.
2. Navigate to the *PeakVisor/backend* directory i.e., using ***cd PeakVisor/backend***
3. Run the command ***npm run dev***

Next, set up the *backend 2*.

4. Open a new terminal / command prompt.
5. Navigate to the *PeakVisor/backend 2* directory i.e., using ***cd PeakVisor/backend 2***
6. Run the command ***npm run serve***

Step 5: Setting up the frontend

Finally, set up the *frontend*.

1. Open a new terminal / command prompt.
2. Navigate to the *PeakVisor/frontend* directory i.e., using ***cd PeakVisor/frontend***
3. Run the command ***ng serve –open***

This will open the PeakVisor web application in a new browser, in a development environment.

2.7 Assumptions and Dependencies

This section lists the assumptions made during the development of the web application that could affect the user requirements.

- The User should have stable Internet access when accessing the application.
- The User is accessing the application through a browser that supports the latest web standards and the frameworks that the application is built on (Angular, HTML, CSS).
- The User has linked their Strava account to access their hiking statistics through our application.
- All data pulled from APIs or fetched from the database are up-to-date and reliable.
- The visual styling and design elements of the web application should be responsive to changes in screen size and/or resolution across platforms.

The team has also identified the following dependencies that the application relies on.

- The web application pulls data from numerous APIs for information on nature trails and community events, namely Meetup and TripAdvisor. Any changes in the API's terms of use may affect the application's future performance and reliability.

3. External Interface Requirements

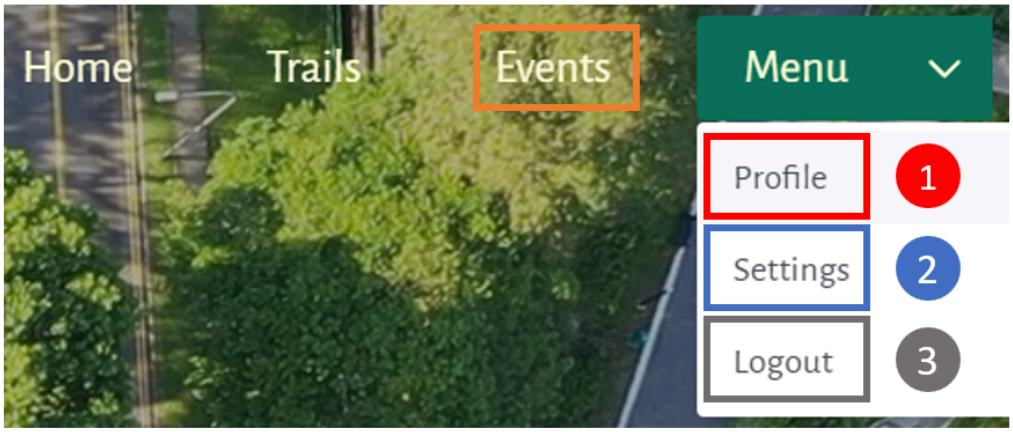
3.1 User Interfaces

This section will explore the user interfaces for the **PeakVisor** web application and the logical characteristics of each interface. The standard buttons that will appear on every screen will be introduced, and sample screens will be provided for each page to better illustrate their functionalities from a user perspective.

Standard Buttons

(1) The **Home** button redirects users to the Home page.
(2) The **Trails** button redirects users to the Trails page.
(3) The **Login** button redirects users to the Login page.

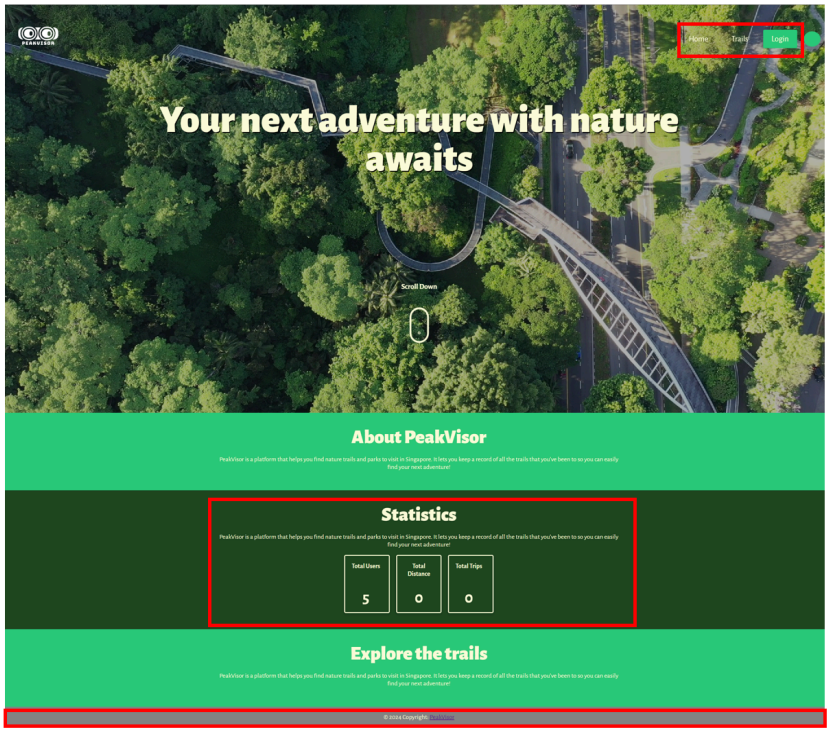
After a successful log-in, the Login button changes to the **Menu** button which produces a dropdown list of user account management functions. The **Events** button, which redirects the user to the Events page, will also be appended to the list of standard buttons.



(1) The **Profile** option redirects users to the Hiking Profile page.
(2) The **Settings** option redirects users to the User Settings page.
(3) The **Logout** option allows the user to log out of their account and returns them back to the Home page as a guest user.

These are the standard functions that will be present on all pages for the application. As a guest user, the user will not have access to the Events page and the account management functions.

Home Page

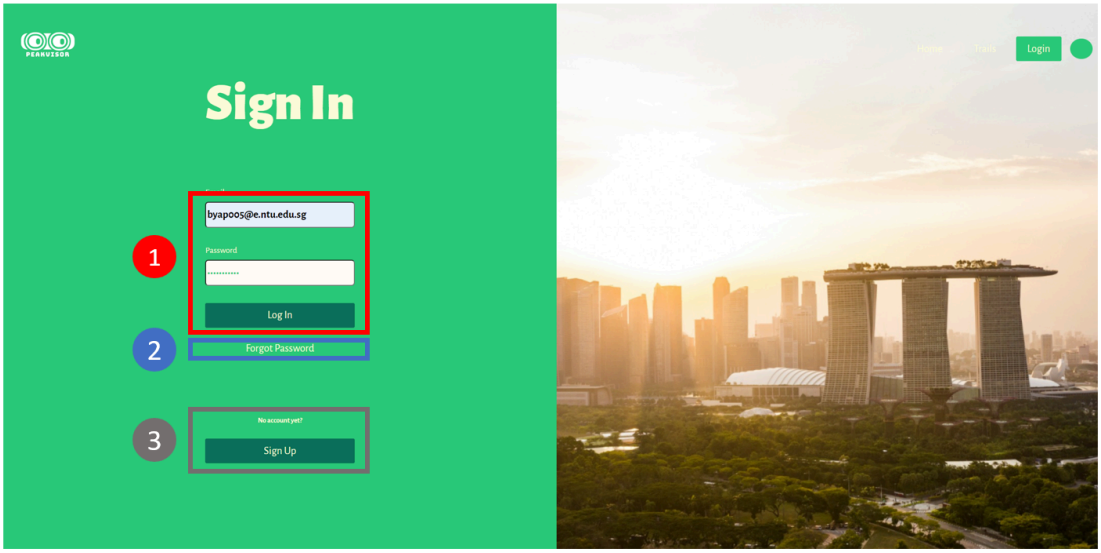


The Home page is the landing page and serves as a starting point for users to navigate the application. The Home page offers a few miscellaneous features that may be of interest to users.

- Under the **Statistics** section, if the user is logged in to their account, they will be able to view a brief insight into their hiking statistics.
- Under the **Explore the Trails** section, the user will be able to interact with an image slider of nature trails listed by the application.

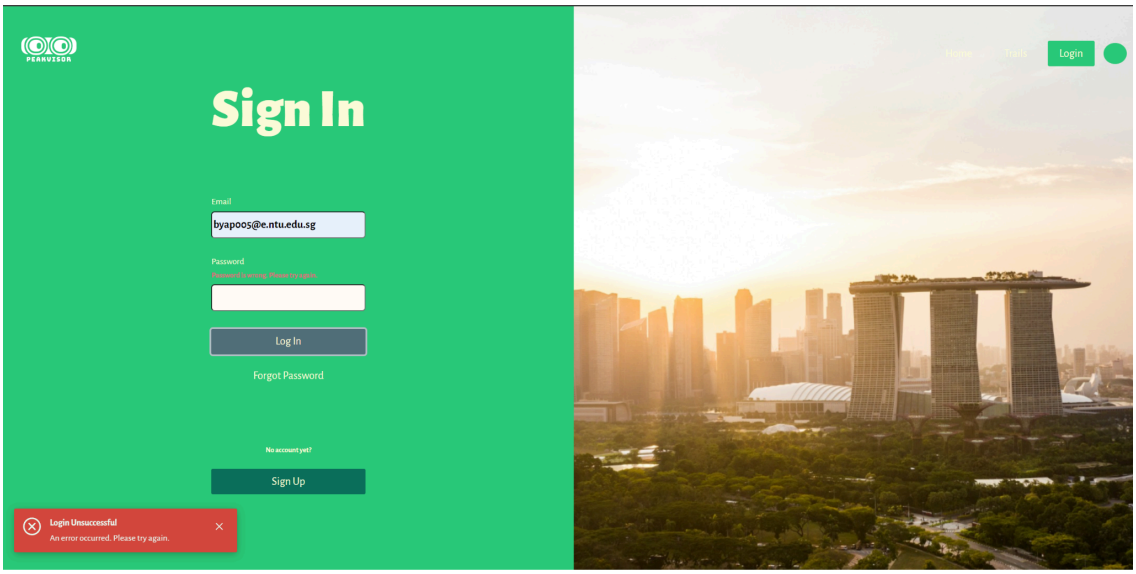
Login Page

If the user clicks on the Login button, they will be redirected to the Login page.

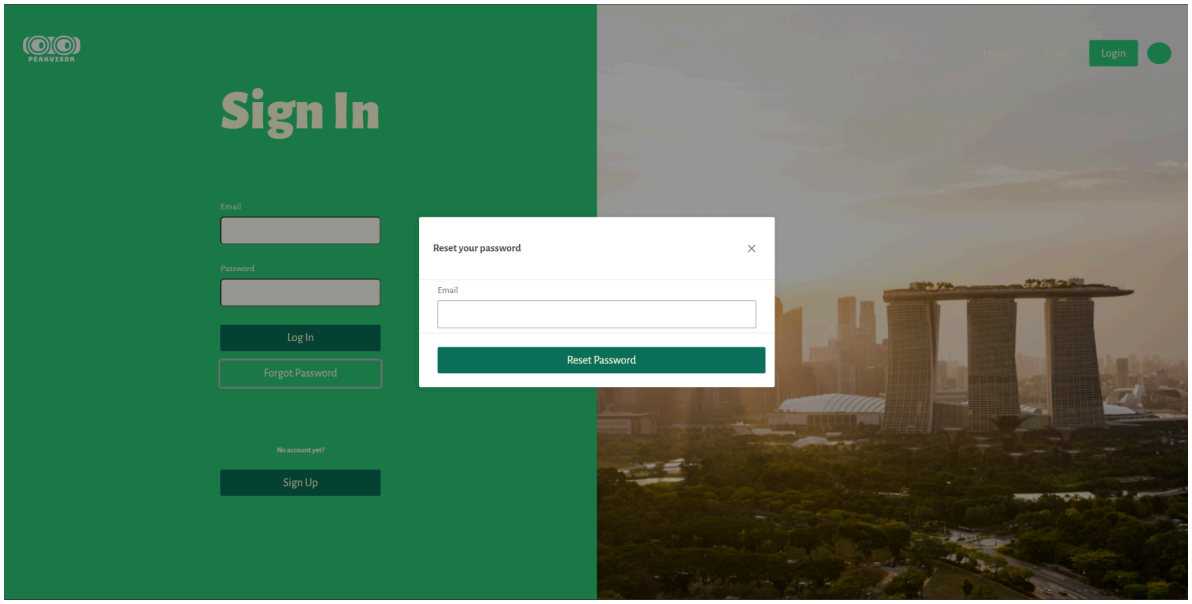


On the Login page, there are three options offered to users.

(1) The user may log into their account using the correct credentials in the text fields.
    (a) If successful, the user will be redirected to the Hiking Profile page.
    (b) If unsuccessful, an appropriate warning message will appear, and the user is prompted to re-try their login attempt.
(2) The user may issue a password reset request by clicking on the **Forget Password** button.
(3) The user may sign up for a new account by clicking on the **Sign Up** button.



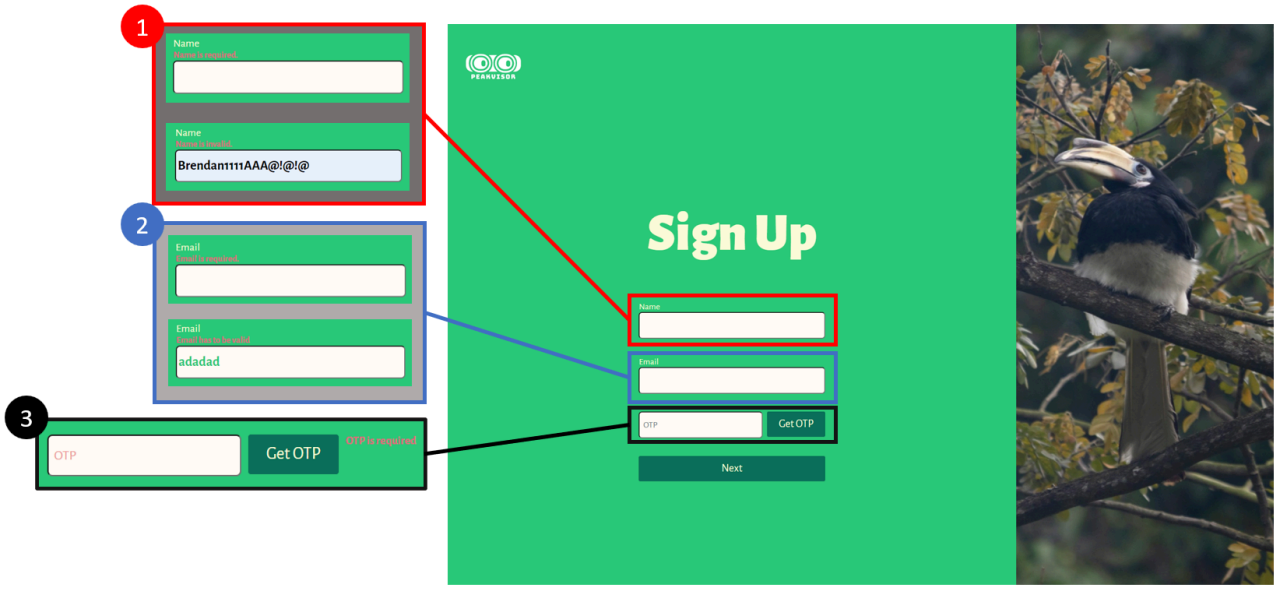
An example of an unsuccessful login attempt with an invalid email/password.
*Note the **Login Unsuccessful** warning message at the bottom left of the screen.*



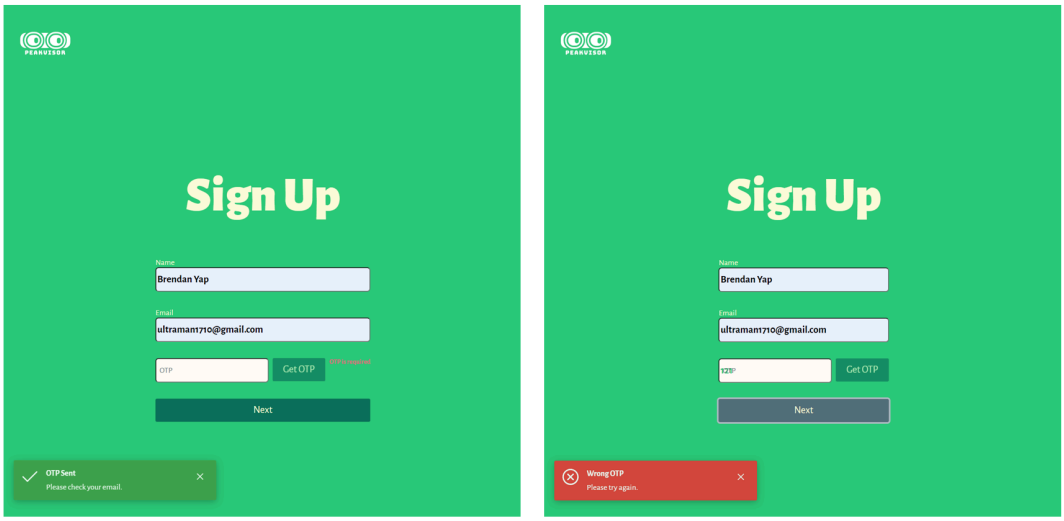
If the user has forgotten their password, they can click on the **Forget Password** button to request a password change. In this case, a pop-up will be displayed, where the user can enter the email address linked to their user account where a password change email will be sent.

Register Page

From the Login page, if the user clicks on the Sign Up button, they will be redirected to the Register page.

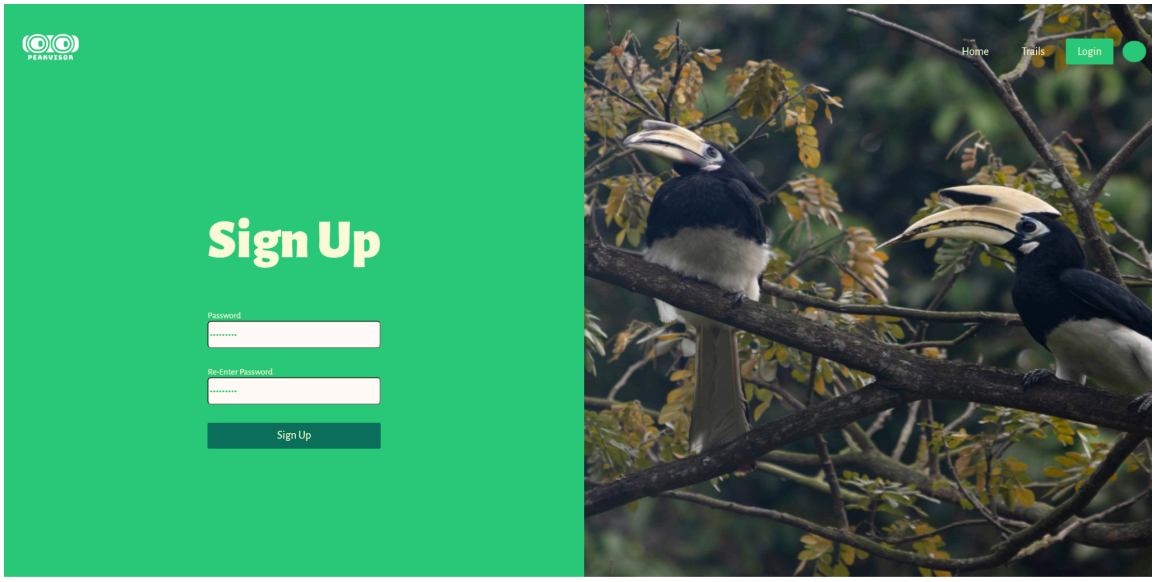


(1) The user may type their name into this field. Depending on the input, specific error messages may be displayed.
(2) The user may type their email address into this field. Depending on the input, specific error messages may be displayed.
(3) Once the user enters both their name and email address, they will click on the **Get OTP** button to receive an OTP in their email inbox. If the field is left empty, an error message will be displayed.

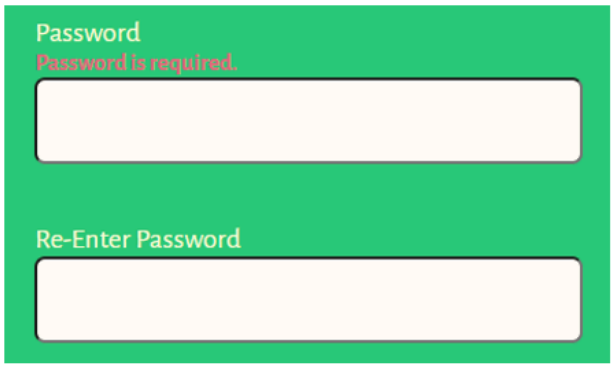


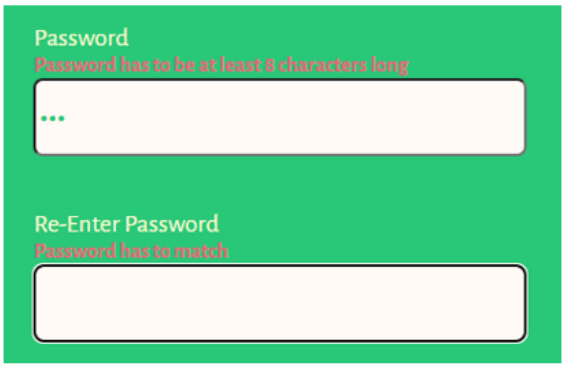
If the OTP is successfully sent, the “OTP Sent” pop-up is displayed (right). If the user enters the wrong OTP and clicks **Next**, then the “Wrong OTP” pop-up is displayed (left).

If the correct OTP is entered and the user clicks **Next**, they will be prompted to enter a password.



The user will enter their desired password into the following fields. Depending on the input, specific error messages will be displayed. Once the user passes all the checks, they can click on the **Sign Up** button to register their user account with our application.

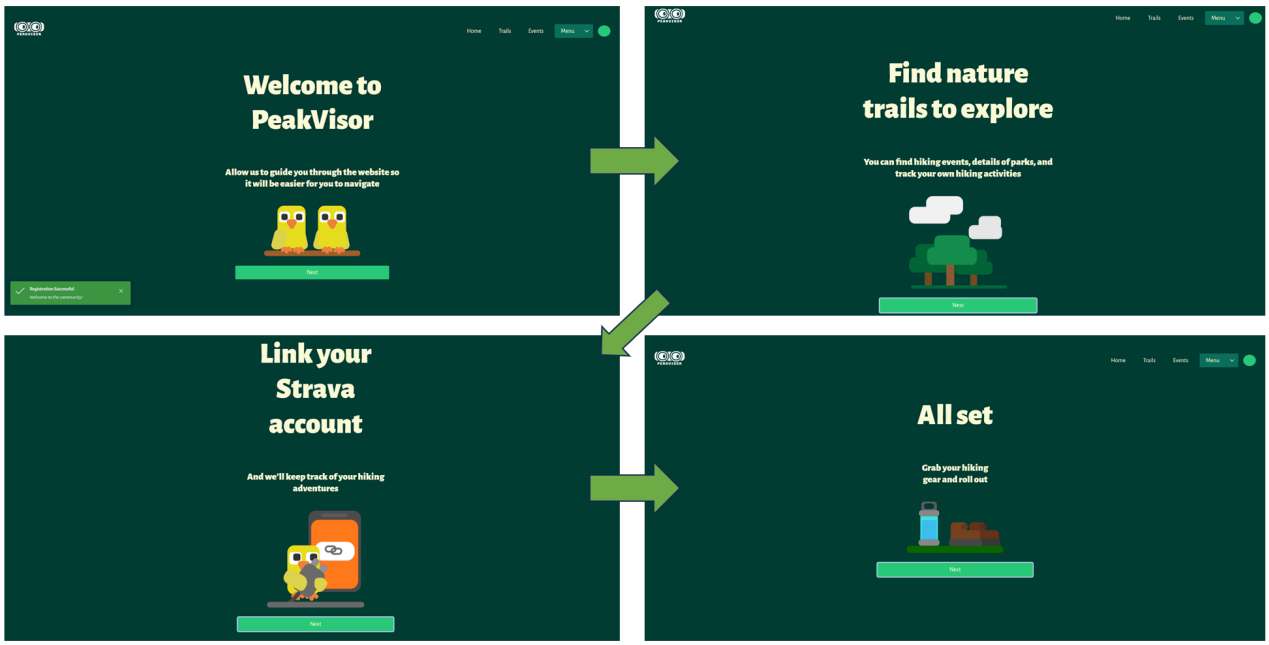




After a successful registration, the user will be redirected to the Onboarding page.

Onboarding Page

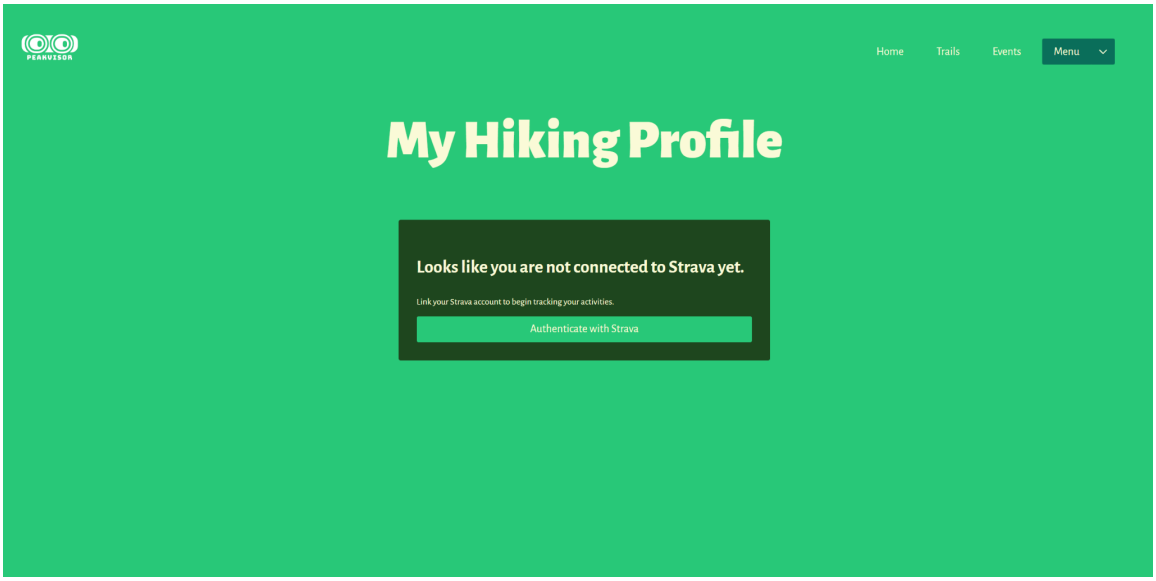
From the Register page, if the user successfully registers an account, they will be redirected to the Onboarding page where they will be introduced to the features of the application.



After the user completes the onboarding tutorial, they will be redirected to their Hiking Profile page.

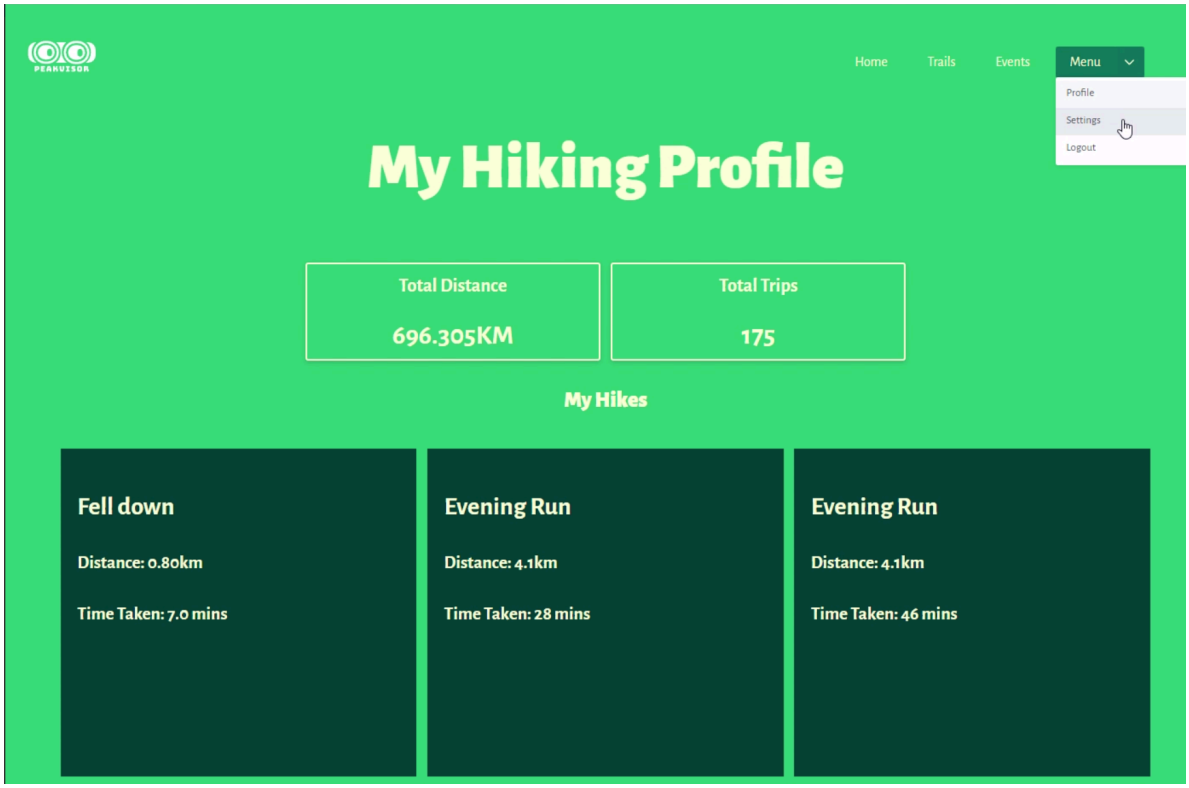
Hiking Profile Page

Once logged in, the user can click on the **Profile** button in the dropdown menu from the standard buttons to access their Hiking Profile page.



When the user first accesses the Hiking Profile page, they will be prompted to authenticate their account by linking it with their Strava account. When the user clicks on the **Authenticate with Strava** button, they will be redirected to Strava's external authentication interface.

After the user successfully authenticates their account with Strava, the Hiking Profile must be made available to them.



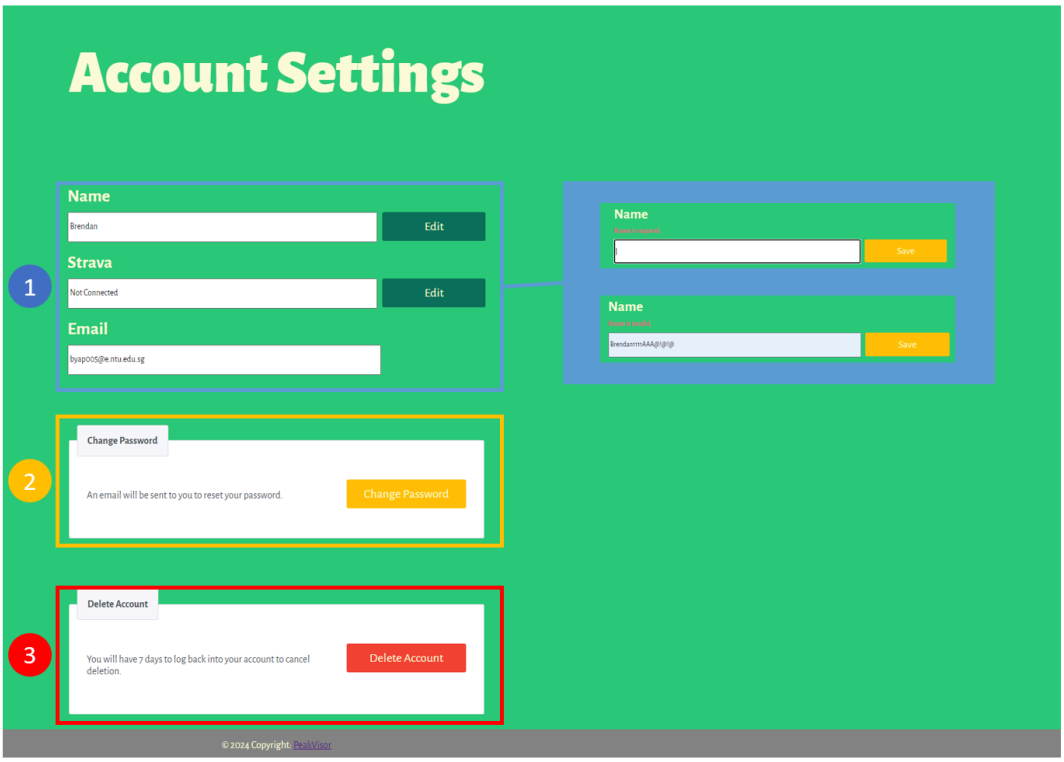
The Hiking Profile page is a simple dashboard which is divided into two sections:

- The user's hiking statistics, including the distance traveled and the total number of trips.
- The user's saved hiking trips ordered by the most recent date, under the "My Hikes" section.

This data is pulled from the user's Strava profile.

Account Settings Page

Once logged in, the user can click on the **Settings** button in the dropdown menu from the standard buttons to access their Account Settings page.

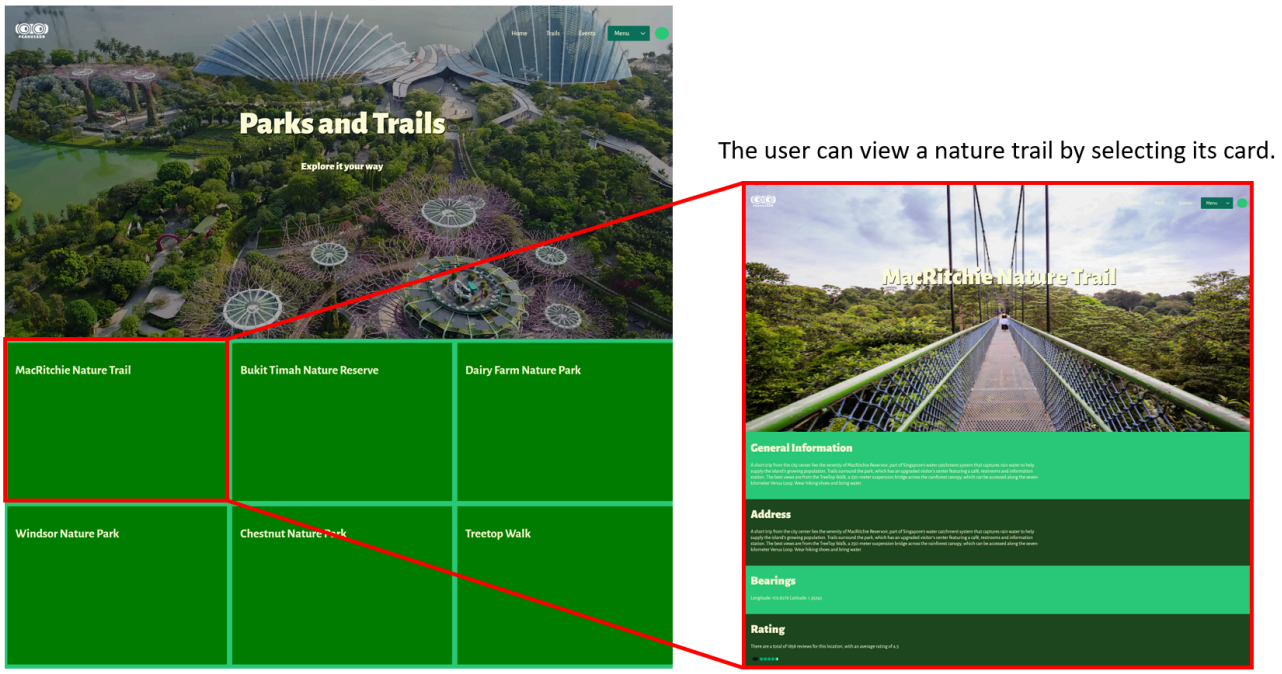


The user is given three options on this page.

(1) The user can edit and save their account details, including their Name and the Strava profile linked to their account. This can be done by clicking on the **Edit** button for that particular field.
    (a) If a change is successful, a pop-up message in green will be displayed.
    (b) If a change was unsuccessful, a pop-up message in red will be displayed.
(2) The user can change their password by clicking on the **Change Password** button. Once clicked, a pop-up window indicating that the password reset email has been sent will appear at the bottom left of the screen.
(3) The user can delete their account by clicking on the **Delete Account** button. Once clicked, they will be logged out and redirected to the Home page.

Trails Page

The user may click on the **Trails** button from the standard buttons to access the Trails page.

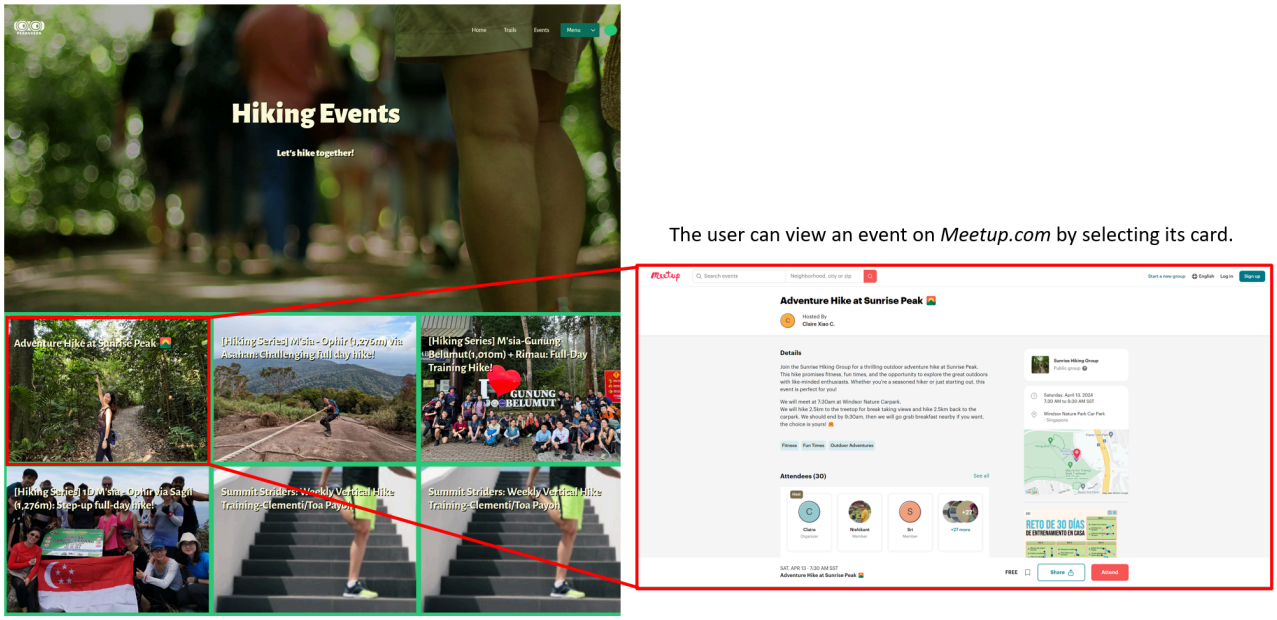


From this page, the user can view a list of nature trails and may select one of the trails to view its information in a dedicated page. This information includes:

(1) General information, a short description of the nature trail.
(2) Location information, the address of the nature trail.
(3) Bearings data (longitude and latitude) for experienced hikers to work with
(4) Reviews and ratings for the nature trail, retrieved from TripAdvisor.

Events Page

Once the user is logged in, they are allowed to access the Events page by clicking on the **Events** button from the standard buttons.



The user can view an event on *Meetup.com* by selecting its card.

From this page, the user can view a list of community outdoor events and may select one of the events to view its information. The user will be redirected to an external page on *Meetup.com*. From there, the user may sign up for an event of their choice.

GUI Standards & Style Guides

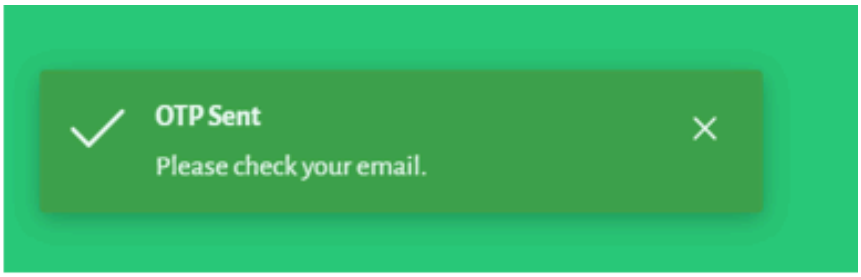
This section will set out the styling standards for the application.

(1) Fonts & Style Guides

Text Type	***Font***	***Example Use Case(s)***
Header 1 <h1>	Alegreya Sans	Page headers, section headers
Body Text <p>		Section descriptions, text field headers
Buttons <app-button>		Log in, sign up, edit, menu buttons

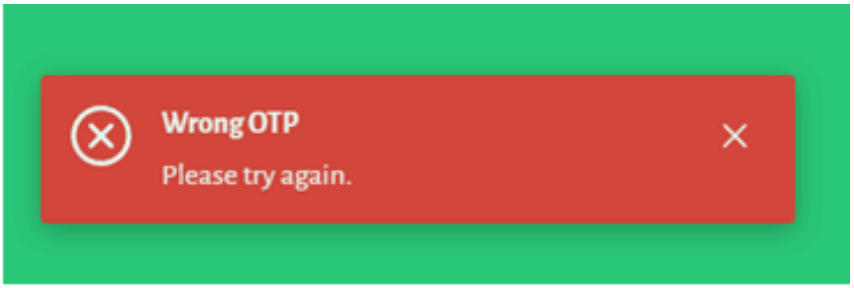
(2) Confirmation Messages

Whenever the user performs a valid action, the user must be informed of the action's success through a pop-up confirmation message. This message must appear from the bottom left of the screen, have an appropriate caption that informs the user of the action's success, and be colored green. An example is shown below.



(3) Warning Messages

Whenever the user performs an invalid action, the user must be informed of the action's failure through a pop-up warning message. This message must appear from the bottom left of the screen, have an appropriate caption that informs the user of the next step of action to take to correct the invalid action, and be colored red. An example is shown below.



3.2 Hardware Interfaces

The **PeakVisor** web application is compatible with a multitude of devices and indirectly interfaces with the hardware components of these devices through the web browser. As such, this section will describe the logical and physical characteristics of the interfaces between the application and hardware components in detail.

Logical Characteristics

1. Supported Device Types: The website is compatible with a multitude of client devices (e.g., desktop computers, laptops, mobile devices) with operating systems (e.g., Windows, Linux distributions) that supports web browsers (e.g., Google Chrome, Edge, Firefox, etc.). It is recommended to run the application on a localhost, for example, a local workstation that has a web development environment, and live deployment to a URL is being worked on.

2. Interactions:
    a. Data: Assuming a web development environment, the web interface and contents are rendered directly from the localhost server to the client device's display.
    b. Control: The user sends control signals (i.e., user input; the selection of menu options, selecting from the list of trails) from their input devices, which is thereafter received by the localhost server and processed to generate the appropriate outputs.

3. Communication Protocols
    a. Database: The client service communicates with the backend Firebase database via REST API endpoints. These endpoints are established from the localhost and there is utilization of local resources (e.g., CPU, GPU, etc.) to execute server-side logic, manage data, and network communications.
    b. Querying: The client service communicates with external APIs (e.g., Meetup.com, TripAdvisor, etc.) through querying with GraphQL, and localhost resources (e.g., CPU, GPU, etc.) are utilized to execute server-side logic, handling client requests, and to render graphics on the client device's display when queried data is received.

Physical Characteristics

The website is rendered directly from the server to the client's device display, and server-side logic and client requests are processed using local hardware resources (e.g., CPU, GPU, RAM, etc.).

There is no requirement for physical connectors of any kind to establish a direct link between the client device and server for data transfer as it is primarily performed via HTTP requests and responses.

3.3 Software Interfaces

This section describes the connections between the application and its underlying software components, including databases, tools, libraries, and integrated commercial components.

Database

The application interfaces with the backend Google Firebase database through REST API endpoints and handles data transfer via standard CRUD operations. The Firebase database handles the following interactions.

- Perform CRUD operations on user data, primarily:
    - Creation of a new user account record in the database during the registration process for a newly-registered user.
    - Authenticating against user records stored in the database during the login process.
    - Retrieval of user data to display on the Account Settings page.
    - Updating of user data based on saved edits made on the Account Settings page.
- Fetch image data for nature trails from the database to display on the Trails page.
- Fetch description, location, and bearings data for nature trails from the database to display on the Trails information page.

Web APIs

The application uses Express.js to efficiently interface with web APIs (such as those provided by Strava, Meetup, and TripAdvisor) and handle API requests, and GraphQL to programmatically access the respective platform's data using flexible querying. GraphQL handles the following interactions:

- Fetch image data for community events from Meetup.com to display on the Events page.
- Fetch reviews and ratings data for nature trails from TripAdvisor to display on the Trails information page.
- Fetch map data for nature trails from Google Maps to display on the Trails information page.

Libraries & Tools

Name	***Version***	***Description***
@angular	17.1.1	Angular web framework
@angular-devkit/ build-angular	17.1.1	Library used to build and test Angular applications
@ng/icons	27.2.0	Library for accessing the collection of Angular icon components.
@ngrx	17.1.0	Library that provides reactive state management for Angular applications.

@types	3.55.6 (Google Maps) 5.1.4 (Jasmine.js) 0.162.0 (Three.js)	Library that contains Typescript definitions for various modules. In this application, we imported Typescript definitions for the Google Maps API, Jasmine.js (framework for testing JS code), and Three.js (3D animated graphics in JS).
jasmine-core	5.1.1	Jasmine is a framework for testing Javascript code. Karma is a tool that allows us to create browsers to run Jasmine tests.
karma	6.4.2	
lottie-web	5.12.2	Lottie is a library that parses After Effects animations and renders them natively on the browser.
ngx-lottie	10.0.0	
ngx-scrollbar	13.0.3	Library for creating cross-platform, custom scroll bars in Angular.
rxjs	7.8.1	Library used to manage data streams and asynchronous operations using reactive programming principles.
tslib	2.6.2	Runtime library for Typescript
typescript	5.3.3	A superset of Javascript and the primary language for application development in Angular.
nodemailer	6.9.12	An easy-to-use node.js module for sending emails from web applications.
graphql	16.8.1	Library of services for GraphQL, a server-side runtime and querying language for executing queries against APIs in web applications.
express	4.18.3	Library of services for Express.js, including server-side routing and API endpoint development for web applications.
firebase	10.9.0	Library of backend services for Firebase, including real-time database solutions, cloud functions, and authentication in web applications.
axios	1.6.8	Javascript library that uses promises to simplify asynchronous HTTP requests in web applications.
webpack	5.90.3	A module bundler for Javascript that is used to bundle and optimize frontend assets for web applications.
cors	2.8.5	A node.js package that enables Cross Origin Resource Sharing (CORS) requests for APIs on different servers.

3.4 Communications Interfaces

This section describes the requirements associated with communications by the application.

Client-Server Communication

The Hypertext Transfer Protocol Secure (HTTPS) protocol adds an encryption and security layer on top of the standard HTTP communication protocol and is used to secure data transfers between the client and server. The protocol ensures secure communication and protects sensitive data from data tampering and man-in-the-middle attacks.

In the **PeakVisor** web application, it is used extensively in functions such as login authentication, account registration, retrieval and updating of user account information, viewing trails and events, and more.

Email Communication

There are two instances where an email is sent to the user's email address:

(1) When the user registers an account and an OTP is requested by the system, the user will click on the **Get OTP** button on the Register page. An email containing the OTP will be forwarded to the user's email address.

(2) When the user forgets their password, they may click on the **Forget Password** button in the Login / Account Settings page. An email containing a Firebase link to reset the user's password will be forwarded to the user's email address.

The Nodemailer module for Node.js applications is used to send emails to the user's email address for these instances. Before outgoing emails are relayed to recipients, simple authentication is performed against the Simple Mail Transfer Protocol (SMTP) server to prevent potential email interception or spoofing by malicious attackers.

4. System Features

The use case diagram (Figure X) illustrates the relationships between the different system features. Excluding the features *Register Page*, *Recover Account*, *Trails Page*, and *Trails Information*, the precondition to access the system features is that the User must first be logged into their account.

4.1 Register Page

4.1.1 Description and Priority

Description	Users visiting the website for the first time can register for a new account.
Priority	High
Frequency of Use	Once per user

4.1.2 Stimulus/Response Sequences

Flow of Events	1. On the landing page, the User clicks on the Login button. 2. The System will redirect the User to the Login page. 3. The User will click on the **Sign Up** button to register for a new user account. 4. The System will redirect the User to the Register page. 5. The User will enter a valid name and email address for their new user account in the respective text fields. 6. The User will click the **Get OTP** button. 7. The system queries the database to check if an identical email address already exists. 8. The system will automatically generate a One-Time Password (OTP), which is sent to the User's email address. 9. The User enters the OTP into the OTP field and clicks the **Next** button. 10. The System verifies the correctness of the OTP. 11. The User will be redirected to the next stage of the Register page.

	12. The User will enter a valid password into the password field. 13. The System will verify if the password satisfies all the requirements. 14. The User will re-enter the password in the respective field to confirm their password. 15. The User will click on the **Sign Up** button to complete the User Account Registration. 16. Upon verification, the System will store the new user account in the database securely. 17. Once the registration is successful, the System will automatically help the User to log into their account.
Alternative Flows	<u>AF-S5: The user enters an email address that has already been registered.</u> 1. The system will display the message “Email address has already been registered, please enter another email address”. 2. The system will return to Step 5. <u>AF-S5: The user enters an email address that is of an invalid format.</u> 1. The system will display an error message “Email is invalid” above the email address field. 2. The system will return to Step 5. <u>AF-S5: The user enters a name that contains invalid characters (numbers, special symbols)</u> 1. The system will display an error message “Name is invalid” above the name field. 2. The system will return to Step 5. <u>AF-S5: The user does not enter a name/email address.</u> 1. The system will display an error message “Name/Email is required” above the respective field. 2. The system will return to Step 5. <u>AF-S9: The user inputs an incorrect OTP</u> 1. The system will display the pop-up message “Wrong OTP! Please try again.”

	2. The system will return to Step 9 and wait for the user to enter the correct OTP. AF-S12: The user enters a password that does not satisfy all the requirements. 1. The system will display an error message “Password has to be at least 8 characters long” above the password field. 2. The system will return to Step 12 and wait for the user to enter another password. AF-S12: The user does not enter a password. 1. The system will display an error message “Password is required” above the password field. 2. The system will return to Step 12 and wait for the user to enter a password. AF-S14: The user enters a mismatched password. 1. The system will display the message “Passwords must match” above the re-entered password field. 2. The system will return to Step 12 and wait for the user to re-enter the password. AF-S15: The user did not complete all of the input fields. 1. When the user clicks on the “Sign Up” button, the system will display the pop-up message “Please check that all the fields are filled up.”. 2. The system will return to Step 15 and wait for the user to complete all remaining fields.
Exceptions	EX-1: The user did not receive the OTP in his email. 1. The user can click on the “Get OTP” button, made available after 1 minute, and a new OTP will be generated. 2. The system will return to Step 8 and wait for the user to enter OTP again. EX-2: The user requests for more than 3 OTP requests. 1. The fourth time the user requests to generate a new OTP, the system will generate the pop-up message “Please try again with a different email.”. 2. The system will return to Step 5 and wait for the user to enter a different email.

4.1.3 Functional Requirements

REQ-1: The User must be able to enter a name, email address, and password that will be used to register and login to their new user account.

1. The User must be able to enter their email address in the respective field.
    1.1. The system must be able to check that the email address points to a valid domain (e.g., *@gmail.com*).
    1.2. The system must be able to display an error message if the email address entered is invalid.
2. The User must be able to enter their name in the respective field.
    2.1. The system must be able to check that the name is of a valid format (no numbers, no special symbols).
    2.2. The system must be able to display an error message if the name entered is invalid.
3. The User must be able to click on the **Get OTP** button to generate a One-Time Password (OTP) forwarded to their email address.
    3.1. The system must be able to check that the OTP entered by the user in the text field matches the OTP sent to their email address.
    3.2. The system must be able to display an error message if the OTP entered does not match the OTP sent.
4. The User must be able to enter and re-enter their password in the respective fields.
    4.1. The system must be able to check that the password is of valid format (mixture of uppercase, lowercase, numbers, special symbols, longer than 8 characters).
    4.2. The system must be able to check that the password and re-entered password match.
    4.3. The system must be able to display an error message if the passwords mismatch.
5. The User must be able to click on the **Sign Up** button to complete the sign-up process.
    5.1. The system must be able to check that all the respective fields are filled and valid before allowing the user to sign up.
    5.2. The system should prompt the User to select another email address if the email they entered is unavailable.

4.2 Login Page

4.2.1 Description and Priority

Description	The User can login to his/her account with the correct credentials that he/she has entered when he/she registered for an account.
Priority	High
Frequency of Use	High

4.2.2 Stimulus/Response Sequences

Flow of Events	1. On the landing page, the user may click on the **Login** button, the System redirects the user to the Login page. 2. On the Login page, the User will enter their email address and password in the respective fields. 3. The User then clicks on the **Log In** button to login into their user account. 4. The System will verify the login credentials (email address and password) provided with the database. 5. If the email address and password entered are correct and verified, the User will be directed to their account on the My Hiking Profile page.
Alternative Flows	<u>AF-S3: The user did not fill up all the fields.</u> 1. The System will display a pop-up message “Login Unsuccessful” and an error message “Email/Password is required” above the respective field, prompting the User to fill in all required fields. 2. The System will return to Step 2 and wait for the user to fill all the fields appropriately. <u>AF-S4: The user has entered an incorrect email address or password.</u> 1. The System will display a pop-up message “Login Unsuccessful” and an error message “Email/Password is invalid. Please try again.” above the respective field. 2. The System will return to Step 2 and wait for the user to correct the fields appropriately.

Exceptions	<u>EX-1: The user has entered an incorrect email address and password more than 3 times.</u> 1. After 3 attempts, the System will display the message “Please try again after 10 minutes.” 2. The System will return to Step 2 after 10 minutes elapses. <u>EX-2 : The user has forgotten their email address or password.</u> 1. The User may click on the **Forgot Password** button that is situated below the **Sign In** button. 2. The User may then recover his/her account using the extended use case *Recover Account*. <u>EX-3: The user does not have an account.</u> 1. The User may click on the **Sign Up** button. 2. The User may then create a new account using the extended use case *Register Page*.

4.2.3 Functional Requirements

REQ-1: The User must be able to enter their login credentials (email address and password) to log into their user account.

1. The User must be able to enter their email address into the respective field.
    1.1. The system must be able to check that the email address points to a valid domain (e.g., *@gmail.com*).
    1.2. The system must be able to display an error message if the email address entered is invalid.
2. The User must be able to enter their password into the respective field.
3. The system must be able to check that the login credentials (email address and password) match that of a user record in the database.
4. The system must be able to login the User into their account after finding a matching user record and redirect the User to their Hiking Profile page.

REQ-2: If the User has forgotten their password, they must be allowed to request for a password recovery email.

1. The User must be able to click the Forgot Password button to open the pop-up modal.

REQ-3: If the User does not have an account, they must be allowed to sign up for a new account.

1. The User must be able to click on the Sign Up button to be redirected to the Register page for the sign-up process.

4.3 Recover Account

4.3.1 Description and Priority

Description	The Users who have forgotten and/or lost their password may request to recover their account.
Priority	High
Frequency of Use	Low

4.3.2 Stimulus/Response Sequences

Flow of Events	1. On the landing page, the User clicks on the “Login” button and the System directs the User to the login page. 2. The System prompts the User to enter their username and password to login. 3. If the User has forgotten their password, they may click on the **Forgot Password** button. 4. The System opens the *Recover Account* pop-up modal and prompts the User to type in a valid email address. 5. The User inputs a valid email address linked to their user account in the displayed field. 6. The User clicks on the **Reset Password** button. 7. The System sends a request to the Database and checks if the email address exists. 8. Upon verification, the System sends a recovery email to the corresponding email address. 9. The System informs the User that a recovery email has been sent successfully to their email address via a pop-up message “Success! Recovery email has been sent!”
Alternative Flows	<u>AF-S7: The email address does not exist in the database.</u> 1. The System displays a pop-up message that informs the User that the email address does not exist in the database and prompts them to try again.

	2. The System returns to Step 5 and waits for the User to enter a valid email address.
Exceptions	N/A

4.3.3 Functional Requirements

REQ-1: The User must be able to enter their email address into the respective field and click on the **Reset Password** button to receive a recovery email.

1. The User must be able to enter their email address into the respective field.
    1.1. The system must be able to check that the email address points to a valid domain (e.g., @gmail.com).
    1.2. The system must be able to display an error message if the email address entered is invalid.
2. The system must be able to query the database to check if the email address belongs to an existing user account.
    2.1. The system must be able to display the error message if the email address does not exist in the database.
3. The system must be able to forward a password reset email to the user's specified email address.

4.4 Hiking Profile Page

4.4.1 Description and Priority

Description	The User should be able to view the contents of their Hiking Profile, including their hiking statistics, distance traveled, and saved trips.
Priority	Medium
Frequency of Use	High

4.4.2 Stimulus/Response Sequences

Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Profile** option. 4. The system redirects the User to the Hiking Profile page.
Alternative Flows	AF-S4: The user has not linked his/her Strava account to his/her user account. 1. The System will display the message “Looks like you are not connected to Strava yet.” on the Hiking Profile page. 2. The System will prompt the user to link his/her user account to his/her Strava account. 3. The User clicks on the **Authenticate with Strava** button with the extended use case *Strava Authentication*.
Exceptions	N/A

4.4.3 Functional Requirements

REQ-1: The User must be able to view their hiking statistics from data retrieved via the Strava API, including their total distance traveled and the total number of trips.

REQ-2: The User must be able to view a list of their saved hiking trips from data retrieved via the Strava API.

1. The system must list the User’s saved trips in the order from most recent to oldest.

4.5 Edit Account Details

4.5.1 Description and Priority

Description	The User should be able to edit and update their Account Settings, including their name and linked Strava account.
Priority	Low
Frequency of Use	Low

4.5.2 Stimulus/Response Sequences

Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Settings** option. 4. The system redirects the User to the Account Settings page. 5. The User clicks on the “Edit” button to change their name and/or the “Unlink” button to unlink their Strava profile. 6. After performing a change, the User clicks on the “Save” button. 7. The System sends an update request to the database to update the User’s profile records accordingly. 8. The System displays a pop-up message “Changes saved!”, informing the User of the successful operation.
Alternative Flows	<u>AF-S6: The user entered a name that contains invalid characters (numbers, special symbols)</u> 1. The system will display an error message “Name is invalid” above the name field. 2. The system will return to Step 5. <u>AF-S6: The user does not enter a name.</u> 1. The system will display an error message “Name is required” above the respective field. 2. The system will return to Step 5.

Exceptions	EX-1: The database was not updated successfully after the User edits their account details. 1. The system displays a pop-up message informing the User of the unsuccessful operation. 2. The system returns to the Account Settings page.

4.5.3 Functional Requirements

REQ-1: The User must be able to edit and save their new account details.

1. The User must be able to edit and save a new name.
   1.1. The system must be able to check that the name is of a valid format (no numbers, no special symbols).
   1.2. The system must be able to display an error message if the name entered is invalid.
2. The User must be able to unlink their current Strava profile if they wish to change to a new Strava profile.
   2.1. The system must be able to redirect the User to an external Strava profile authentication page to change the Strava profile linked to their user account.
3. The system must be able to display an appropriate message depending on the outcome of the change of account details.
   3.1. The system must be able to display a success message if the change of account details was successful.
   3.2. The system must be able to display an error message if the change of account details was unsuccessful.

4.6 Change Password

4.6.1 Description and Priority

Description	The User must be able to change his/her user account password.
Priority	High
Frequency of Use	Low

4.6.2 Stimulus/Response Sequences

Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Settings** option. 4. The system redirects the User to the Account Settings page. 5. The User clicks on the **Change Password** button. 6. The system sends a password reset email to the User's email address. 7. The system displays the pop-up message "The password reset email has been sent to you!". 8. The User clicks on the external link in the password reset email. 9. The User enters their new password. 10. The User clicks "Save". 11. The system sends an update request to the database to update the User's password.
Alternative Flows	AF-S10: The user enters a password that does not satisfy all the requirements. 1. The system will display an error message "Password has to be at least 8 characters long" above the password field. 2. The system will return to Step 9.

	AF-S10: The user does not enter a password. 1. The system will display an error message “Password is required” above the password field. 2. The system will return to Step 9.
Exceptions	N/A

4.6.3 Functional Requirements

REQ-1: The User must be able to enter and save their new password via the password reset link sent to their email address.

1. The system must be able to check that the password is of valid format (mixture of uppercase, lowercase, numbers, special symbols, longer than 8 characters).
2. The system must be able to display an error message if the password is invalid.

4.7 Trails Page

4.7.1 Description and Priority

Description	The application displays a list of nature trails for the User to browse.
Priority	High
Frequency of Use	High

4.7.2 Stimulus/Response Sequences

Flow of Events	1. From any page, the User clicks on the **Trails** button from the standard buttons. 2. The User will be redirected to the Trails page. 3. The system fetches the title and image data for the nature trails from the database. 4. The system programmatically renders the Trails page and populates it with the list of nature trails.
Alternative Flows	N/A
Exceptions	N/A

4.7.3 Functional Requirements

REQ-1: The User must be able to view the list of nature trails offered by the application.

REQ-2: The User must be able to select their preferred nature trail to view its information.

4.8 Trail Information

4.8.1 Description and Priority

Description	The application displays the information for a trail that the User has selected from the Trails page. This information includes a concise description, the location (address and bearings) of the trail, and user ratings retrieved via TripAdvisor.
Priority	High
Frequency of Use	High

4.8.2 Stimulus/Response Sequences

Flow of Events	1. From the Trails page, the User clicks on their preferred trail from the list of nature trails displayed. 2. The User will be redirected to the Trail Information page for that selected nature trail. 3. The system fetches the required information associated with that nature trail from the database and via the TripAdvisor API. 4. The system programmatically renders the Trail Information page and populates it with the fetched data.
Alternative Flows	N/A
Exceptions	N/A

4.8.3 Functional Requirements

REQ-1: The User must be able to view the information associated with their selected trail.

4.9 Events Page

4.9.1 Description and Priority

Description	The application displays a list of community events for the User to browse and participate in.
Priority	High
Frequency of Use	High

4.9.2 Stimulus/Response Sequences

Flow of Events	1. From any page, the User clicks on the **Events** button from the standard buttons. 2. The User will be redirected to the Events page. 3. The system fetches the title and image data for the community events via the Meetup.com API. 4. The system programmatically renders the Events page and populates it with the list of community events. 5. The User clicks on their preferred community event. 6. The User will be redirected to the community event's external page hosted on Meetup.com.
Alternative Flows	N/A
Exceptions	N/A

4.9.3 Functional Requirements

REQ-1: The User must be able to view the list of community events offered by the application.

REQ-2: The User must be able to select their preferred community event to view its information.

1. The system must redirect the User to the external Meetup.com page where the details of the community page are hosted.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system response time for rendering pages, loading modals, and data handling over a stable Internet connection must be 10 seconds or less.
- The link for account recovery or password reset must be issued to the User's email address within 5 seconds of the request having taken place.
- The system should be able to scale reliably to handle high data throughput from at least 10,000 concurrent users without significant degradation in performance.
- The maximum amount of unplanned downtime that the system should experience is 4 hours per year, which equates to an acceptable annual uptime of 99.95%.
- Scheduled downtime for maintenance and/or updates is dedicated a maximum of 5 hours per month.
- Scheduled downtime for maintenance and/or updates should only take place in the time window between 2 a.m. to 4 a.m. on the day of maintenance.

5.2 Safety Requirements

- The web application development life cycle follows the ISO/IEC/IEEE 23026:2023 software engineering standards.
- Whenever the user enters invalid input into a text field (e.g., password with invalid characters), the system must display an appropriate error message above the text field to inform the user of the incorrect action taken and briefly, how to rectify it.
- Whenever the user clicks on a button to perform an action (e.g., logging into their user account), a pop-up message will be displayed by the system at the bottom left of the screen to provide the user (1) a confirmation message notifying the success of the action taken, or (2) a warning message notifying the failure of the action taken.
- The user must embark on an onboarding process to introduce the system features and ensure that the user is adequately informed about the proper use of the application before using it.

5.3 Security Requirements

- System security, personal data handling, and database operations must meet the standards delineated in the Personal Data Protection Act (PDPA).
- The system must employ secure client-server communication protocols (e.g., HTTPS for encryption) and appropriate database authentication rules to ensure personal data is not compromised during collection, transfer, and storage.

- The system must not store the user's personally identifiable information (e.g., login credentials, OTPs) in data structures or variables in the code. They must be stored securely in the database and only indirectly referenced when needed.

- Input validation and sanitization must be incorporated into any input text field to prevent code/query injection attacks.

5.4 Software Quality Attributes

1. **Extensibility**: The system must be designed to support extension to allow for new features to be added on top of the existing codebase without much modification.

2. **Portability**: The application must be responsive and compatible with both desktop environments and mobile devices.

3. **Reliability**:
    a. The system must support exception handling and provide fallbacks for unexpected errors, with a failure rate of less than 5%.
    b. The system must have a backup backend server to diversify infrastructural risk in case the main server goes down.

4. **Reusability**: The system components must be able to be replicated across different instances without much difficulty, supported by the use of modular and reusable code structures.

5. **Usability**: The application must be navigable with a simple user interface and its main features comprehensible by diverse age groups.

6. **Testability**: The application must be designed to support unit testing and ensure the correct functionality, reliability, and performance of individual components.

5.5 Business Rules

Access permissions to system features is controlled for different user groups.

- **Guest Users**: This group refers to users without a registered account. These users only have access to the Home and Trails pages, and are encouraged to Login to access further features of the PeakVisor application.
- **Registered Users**: This group refers to users who have been registered with the application. In addition to the system features made available to guests, registered users may also access Events and their Hiking Profile.

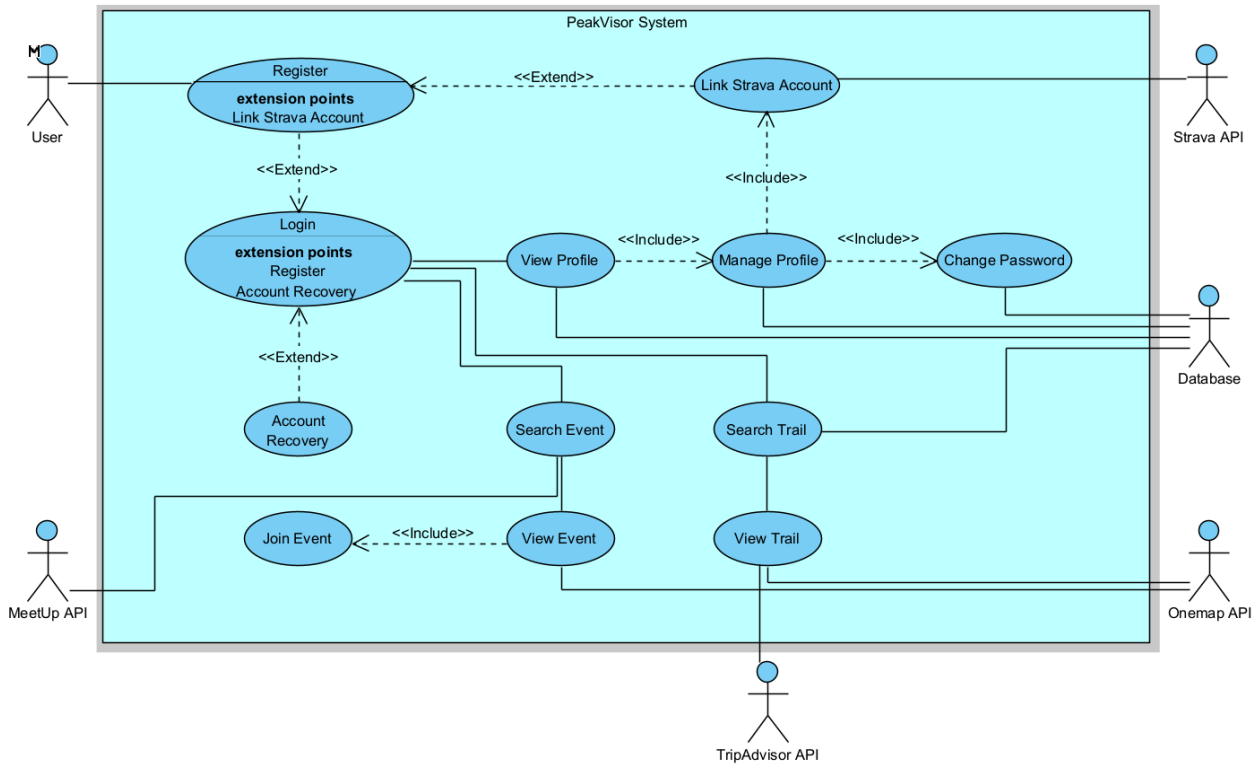
Appendix A: Glossary

Data Dictionary

User Account	Refers to the digital identity associated with the user of the application, created within the application, without which the user cannot access most of the application's features and services. It is associated with a username and password, and may contain additional personal information and settings.
Trail	Refers to one of the many outdoor locations where the user may wish to hike at.
Trail Information	Refers to the details associated with a trail, including its description, exact location, user ratings, and more.
Trail Rating	A scaling system that rates the user experience of a hiking trail. 5 stars means the best experience while 1 star is the worst. The rating of a specific trait will be the average of all the users' ratings. The rating is fetched via the TripAdvisor API.
Event	Refers to a scheduled outdoor activity that is usually organized by people, clubs, organizations, or businesses and involves walking or trekking along trails or roads in natural areas. These gatherings frequently seek to develop hiking enthusiasts' social relationships, encourage physical activity, and explore the surrounding area.
One-Time Password (OTP)	A 4-digit number which will be sent to a user's email address or phone number via email or SMS for the user to log in the platform. An OTP will expire in 3 minutes and the user can request another OTP after 1 minute.
Hiking Profile	Refers to the dashboard of personal hiking statistics accessible by each user through the application. This includes the list of previous hikes that they have completed previously. Each hike is described by data fetched via the Strava API, including a title, the distance traveled, and the total time taken.

Appendix B: Analysis Models

Use Case Diagram



Use Case Descriptions

Use Case ID:	001		
Use Case Name:	Register		
Created By:	Jeannie Wong	Last Updated By:	Ben
Date Created:	30 January 2024	Date Last Updated:	5 February 2024

Actor	User (Initiating Actor), System
Description	Users visiting the website for the first time can register for a new account.
Preconditions	The user must not have a user account tagged to their registering email address prior to the registration
Postconditions	The user has successfully registered for a user account in the website with a unique email address and password and their new user account is added into the system database. **OR** The user is notified of the reason(s) why the registration of the account is unsuccessful.
Priority	High
Frequency of Use	Once per user

Flow of Events	1. On the landing page, the User clicks on the Login button. 2. The System will redirect the User to the Login page. 3. The User will click on the **Sign Up** button to register for a new user account. 4. The System will redirect the User to the Register page. 5. The User will enter a valid name and email address for their new user account in the respective text fields. 6. The User will click the **Get OTP** button. 7. The system queries the database to check if an identical email address already exists. 8. The system will automatically generate a One-Time Password (OTP), which is sent to the User's email address. 9. The User enters the OTP into the OTP field and clicks the **Next** button. 10. The System verifies the correctness of the OTP. 11. The User will be redirected to the next stage of the Register page. 12. The User will enter a valid password into the password field. 13. The System will verify if the password satisfies all the requirements. 14. The User will re-enter the password in the respective field to confirm their password. 15. The User will click on the **Sign Up** button to complete the User Account Registration. 16. Upon verification, the System will store the new user account in the database securely. 17. Once the registration is successful, the System will automatically help the User to log into their account.

Alternative Flows	<u>AF-S5: The user enters an email address that has already been registered.</u> 1. The system will display the message “Email address has already been registered, please enter another email address”. 2. The system will return to Step 5. <u>AF-S5: The user enters an email address that is of an invalid format.</u> 1. The system will display an error message “Email is invalid” above the email address field. 2. The system will return to Step 5. <u>AF-S5: The user enters a name that contains invalid characters (numbers, special symbols)</u> 1. The system will display an error message “Name is invalid” above the name field. 2. The system will return to Step 5. <u>AF-S5: The user does not enter a name/email address.</u> 1. The system will display an error message “Name/Email is required” above the respective field. 2. The system will return to Step 5. <u>AF-S9: The user inputs an incorrect OTP</u> 1. The system will display the pop-up message “Wrong OTP! Please try again.” 2. The system will return to Step 9 and wait for the user to enter the correct OTP. <u>AF-S12: The user enters a password that does not satisfy all the requirements.</u> 1. The system will display an error message “Password has to be at least 8 characters long” above the password field. 2. The system will return to Step 12 and wait for the user to enter another password. <u>AF-S12: The user does not enter a password.</u> 1. The system will display an error message “Password is required” above the password field. 2. The system will return to Step 12 and wait for the user to enter a password.

	AF-S14: The user enters a mismatched password. 1. The system will display the message “Passwords must match” above the re-entered password field. 2. The system will return to Step 12 and wait for the user to re-enter the password. AF-S15: The user did not complete all of the input fields. 1. When the user clicks on the “Sign Up” button, the system will display the pop-up message “Please check that all the fields are filled up.”. 2. The system will return to Step 15 and wait for the user to complete all remaining fields.
Exceptions	EX-1: The user did not receive the OTP in his email. 1. The user can click on the “Get OTP” button, made available after 1 minute, and a new OTP will be generated. 2. The system will return to Step 8 and wait for the user to enter OTP again. EX-2: The user requests for more than 3 OTP requests. 1. The fourth time the user requests to generate a new OTP, the system will generate the pop-up message “Please try again with a different email.”. 2. The system will return to Step 5 and wait for the user to enter a different email.
Extends	Login
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	002		
Use Case Name:	Login		
Created By:	Jeannie Wong	Last Updated By:	Brendan
Date Created:	3 February 2024	Date Last Updated:	5 February 2024

Actor	User (Initiating Actor), System
Description	The User can login to his/her account with the correct credentials that he/she has entered when he/she registered for an account.
Preconditions	1. The user must be connected to the Internet. 2. The user must have a registered account.
Postconditions	The user has successfully logged into his/her own account. **OR** The user is notified of the reason(s) why he/she is unable to login into his/her account. For instance, the user has entered the wrong email address or password.
Priority	High
Frequency of Use	High

Flow of Events	1. On the landing page, the user may click on the **Login** button, the System redirects the user to the Login page. 2. On the Login page, the User will enter their email address and password in the respective fields. 3. The User then clicks on the **Log In** button to login into their user account. 4. The System will verify the login credentials (email address and password) provided with the database. 5. If the email address and password entered are correct and verified, the User will be directed to their account on the My Hiking Profile page.
Alternative Flows	<u>AF-S3: The user did not fill up all the fields.</u> 1. The System will display a pop-up message "Login Unsuccessful" and an error message "Email/Password is required" above the respective field, prompting the User to fill in all required fields. 2. The System will return to Step 2 and wait for the user to fill all the fields appropriately. <u>AF-S4: The user has entered an incorrect email address or password.</u> 1. The System will display a pop-up message "Login Unsuccessful" and an error message "Email/Password is invalid. Please try again." above the respective field. 2. The System will return to Step 2 and wait for the user to correct the fields appropriately
Exceptions	<u>EX-1: The user has entered an incorrect email address and password more than 3 times.</u> 1. After 3 attempts, the System will display the message "Please try again after 10 minutes." 2. The System will return to Step 2 after 10 minutes elapses. <u>EX-2 : The user has forgotten their email address or password.</u> 1. The User may click on the **Forgot Password** button that is situated below the **Sign In** button. 2. The User may then recover his/her account using the extended use case *Recover Account*.

	EX-3: The user does not have an account. 1. The User may click on the **Sign Up** button. 2. The User may then create a new account using the extended use case *Register Page*.
Extends	None
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	003		
Use Case Name:	RecoverAccount		
Created By:	Brendan	Last Updated By:	Brendan
Date Created:	1 February 2024	Date Last Updated:	1 February 2024

Actor	User (Initiating Actor), System
Description	The Users who have forgotten and/or lost their password may request to recover their account..
Preconditions	1. The User must have a user account prior to recovery. 2. The User must have a valid email address that is linked to their user account.
Postconditions	The System has successfully issued an account recovery email to the User's email address. **OR** The User is notified of the reason(s) why the recovery of the account is unsuccessful.
Priority	High
Frequency of Use	Low
Flow of Events	1. On the landing page, the User clicks on the "Login" button and the System directs the User to the login page. 2. The System prompts the User to enter their username and password to login. 3. If the User has forgotten their password, they may click on the

	Forgot Password button. 4. The System opens the *Recover Account* pop-up modal and prompts the User to type in a valid email address. 5. The User inputs a valid email address linked to their user account in the displayed field. 6. The User clicks on the **Reset Password** button. 7. The System sends a request to the Database and checks if the email address exists. 8. Upon verification, the System sends a recovery email to the corresponding email address. 9. The System informs the User that a recovery email has been sent successfully to their email address via a pop-up message "Success! Recovery email has been sent!"
Alternative Flows	AF-S7: The email address does not exist in the database. 1. The System displays a pop-up message that informs the User that the email address does not exist in the database and prompts them to try again. 2. The System returns to Step 5 and waits for the User to enter a valid email address.
Exceptions	N/A
Extends	Login
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	004		
Use Case Name:	ViewProfile		
Created By:	Jeannie Wong	Last Updated By:	Jeannie Wong
Date Created:	4 February 2024	Date Last Updated:	6 February 2024

Actor	User (Initiating Actor), System
Description	The User should be able to view the contents of their Hiking Profile, including their hiking statistics, distance traveled, and saved trips.
Preconditions	1. The user must be logged into his/her user account. 2. The user must have a Strava account that is linked to his/her user account.
Postconditions	The System displays the contents of the user's hiking profile on the Hiking Profile page. **OR** The System notifies the user of the reason(s) why it is unable to display the contents of the user's profile.
Priority	Medium
Frequency of Use	High
Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Profile** option.

	4. The system redirects the User to the Hiking Profile page.
Alternative Flows	AF-S4: The user has not linked his/her Strava account to his/her user account. 1. The System will display the message "Looks like you are not connected to Strava yet." on the Hiking Profile page. 2. The System will prompt the user to link his/her user account to his/her Strava account. 3. The User clicks on the **Authenticate with Strava** button with the extended use case *Strava Authentication*.
Exceptions	N/A
Extends	None
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	005		
Use Case Name:	ManageProfile		
Created By:	Brendan	Last Updated By:	Brendan
Date Created:	1 February 2024	Date Last Updated:	4 February 2024

Actor	User (Initiating Actor), System
Description	The User should be able to edit and update their Account Settings, including their name and linked Strava account.
Preconditions	The user must be logged into his/her user account.
Postconditions	The User must have successfully updated their account details. **OR** The User must be informed of the reason why their edits were unsuccessful.
Priority	Low
Frequency of Use	Low
Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Settings** option. 4. The system redirects the User to the Account Settings page. 5. The User clicks on the “Edit” button to change their name and/or

	the “Unlink” button to unlink their Strava profile. 6. After performing a change, the User clicks on the “Save” button. 7. The System sends an update request to the database to update the User’s profile records accordingly. 8. The System displays a pop-up message “Changes saved!”, informing the User of the successful operation.
Alternative Flows	AF-S6: The user entered a name that contains invalid characters (numbers, special symbols) 1. The system will display an error message “Name is invalid” above the name field. 2. The system will return to Step 5. AF-S6: The user does not enter a name. 1. The system will display an error message “Name is required” above the respective field. 2. The system will return to Step 5.
Exceptions	EX-1: The database was not updated successfully after the User edits their account details. 1. The system displays a pop-up message informing the User of the unsuccessful operation. 2. The system returns to the Account Settings page.
Includes	ChangePassword
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	006		
Use Case Name:	ChangePassword		
Created By:	Jeannie Wong	Last Updated By:	Jeannie Wong
Date Created:	3 February 2024	Date Last Updated:	3 February 2024

Actor	User (Initiating Actor), System
Description	The User must be able to change his/her user account password.
Preconditions	The user must be logged into his/her user account.
Postconditions	The user changes his/her password successfully.
Priority	Medium
Frequency of Use	Medium
Flow of Events	1. From any page, the User clicks on the dropdown menu at the top right hand corner of the page. 2. The system outputs a list of available options. 3. The User clicks on the **Settings** option. 4. The system redirects the User to the Account Settings page. 5. The User clicks on the **Change Password** button. 6. The system sends a password reset email to the User's email address. 7. The system displays the pop-up message "The password reset email has been sent to you!".

	8. The User clicks on the external link in the password reset email. 9. The User enters their new password. 10. The User clicks “Save”. 11. The system sends an update request to the database to update the User’s password.
Alternative Flows	AF-S10: The user enters a password that does not satisfy all the requirements. 1. The system will display an error message “Password has to be at least 8 characters long” above the password field. 2. The system will return to Step 9. AF-S10: The user does not enter a password. 1. The system will display an error message “Password is required” above the password field. 2. The system will return to Step 9.
Exceptions	N/A
Extends	N/A
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	007		
Use Case Name:	SearchTrail		
Created By:	Jeannie Wong	Last Updated By:	Jeannie Wong
Date Created:	3 February 2024	Date Last Updated:	6 February 2024

Actor	User (Initiating Actor), System
Description	The application displays a list of nature trails for the User to browse.
Preconditions	None
Postconditions	The system must display a list of available nature trails for users to consider hiking in Singapore.
Priority	Low
Frequency of Use	High
Flow of Events	1. From any page, the User clicks on the **Trails** button from the standard buttons. 2. The User will be redirected to the Trails page. 3. The system fetches the title and image data for the nature trails from the database. 4. The system programmatically renders the Trails page and populates it with the list of nature trails.
Alternative Flows	N/A

Exceptions	N/A
Extends	N/A
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	008		
Use Case Name:	ViewTrail		
Created By:	Jeannie Wong	Last Updated By:	Jeannie Wong
Date Created:	3 February 2024	Date Last Updated:	3 February 2024

Actor	User (Initiating Actor), System
Description	The User may click on a trail, and the application will display the information for the selected trail. This information includes a description, the location (address and bearings) of the trail, and user ratings retrieved via TripAdvisor.
Preconditions	The User must be on the Trails page to select the trail.
Postconditions	The system displays the information for the trail that the user has clicked on in a new page.
Priority	Low
Frequency of Use	High
Flow of Events	1. From the Trails page, the User clicks on their preferred trail from the list of nature trails displayed. 2. The User will be redirected to the Trail Information page for that selected nature trail. 3. The system fetches the required information associated with that nature trail from the database and via the TripAdvisor API. 4. The system programmatically renders the Trail Information page and populates it with the fetched data.

Alternative Flows	N/A
Exceptions	N/A
Extends	N/A
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	009		
Use Case Name:	SearchEvent		
Created By:	Brendan	Last Updated By:	Brendan
Date Created:	3 February 2024	Date Last Updated:	3 February 2024

Actor	User (Initiating Actor), System
Description	The application displays a list of community events for the User to browse and participate in.
Preconditions	The User must be logged into their account to access this feature.
Postconditions	The system must display a list of available hiking events for users to consider joining.
Priority	Low
Frequency of Use	High
Flow of Events	1. From any page, the User clicks on the **Events** button from the standard buttons. 2. The User will be redirected to the Events page. 3. The system fetches the title and image data for the community events via the Meetup.com API. 4. The system programmatically renders the Events page and populates it with the list of community events.

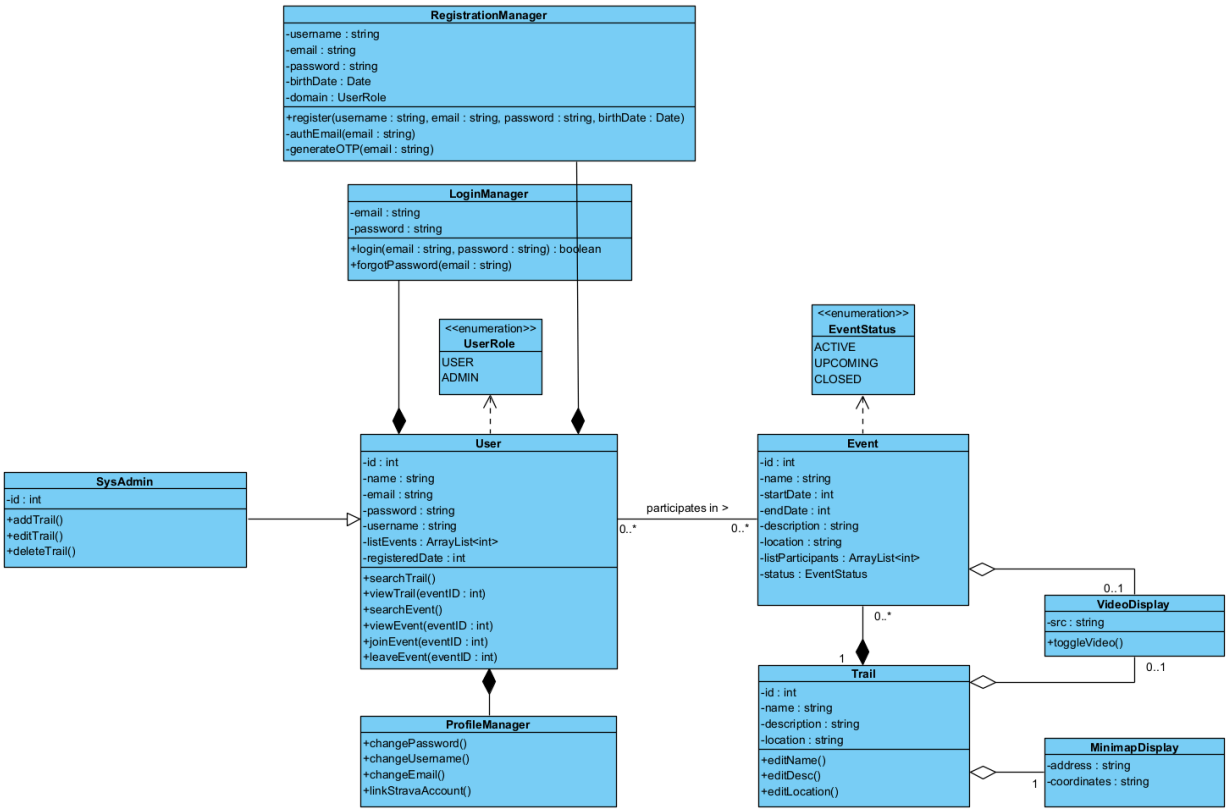
Alternative Flows	N/A
Exceptions	N/A
Extends	N/A
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Use Case ID:	010		
Use Case Name:	ViewEvent		
Created By:	Brendan	Last Updated By:	Brendan
Date Created:	3 February 2024	Date Last Updated:	4 February 2024

Actor	User (Initiating Actor), System
Description	The User may click on a community hiking event to view its contents and consider participating.
Preconditions	1. The User must be logged into their account to access this feature. 2. The User must be on the Events page to select an event.
Postconditions	The application redirects the user to an external application page for the selected hiking event.
Priority	Low
Frequency of Use	Medium
Flow of Events	1. From any page, the User clicks on their preferred event from the list of hiking events displayed. 2. The User will be redirected to the community event's external page hosted on Meetup.com.
Alternative Flows	N/A

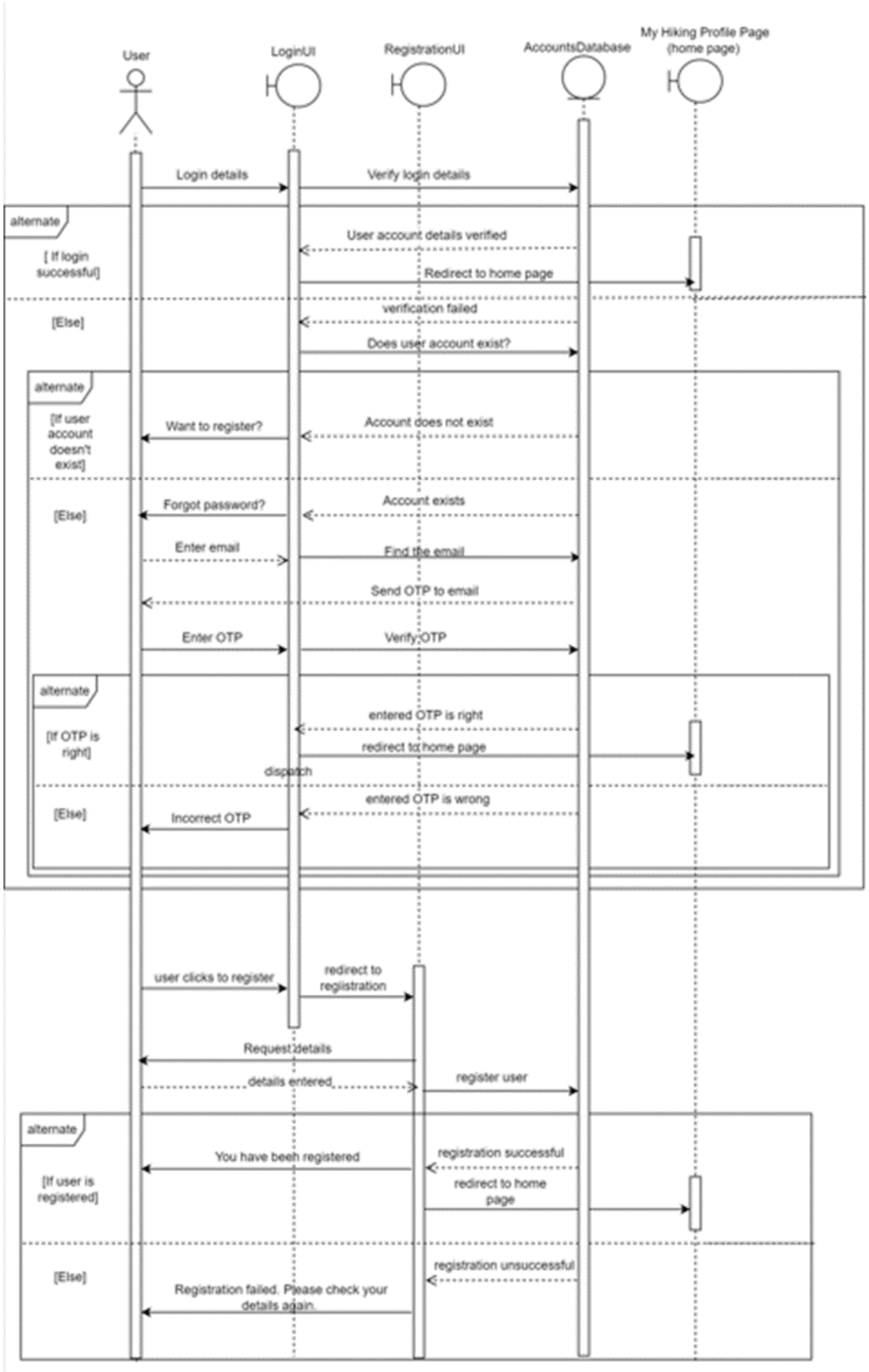
Exceptions	N/A
Extends	N/A
Special Requirements	N/A
Assumptions	The User has Internet access during this process.
Notes and Issues	N/A

Class Diagram

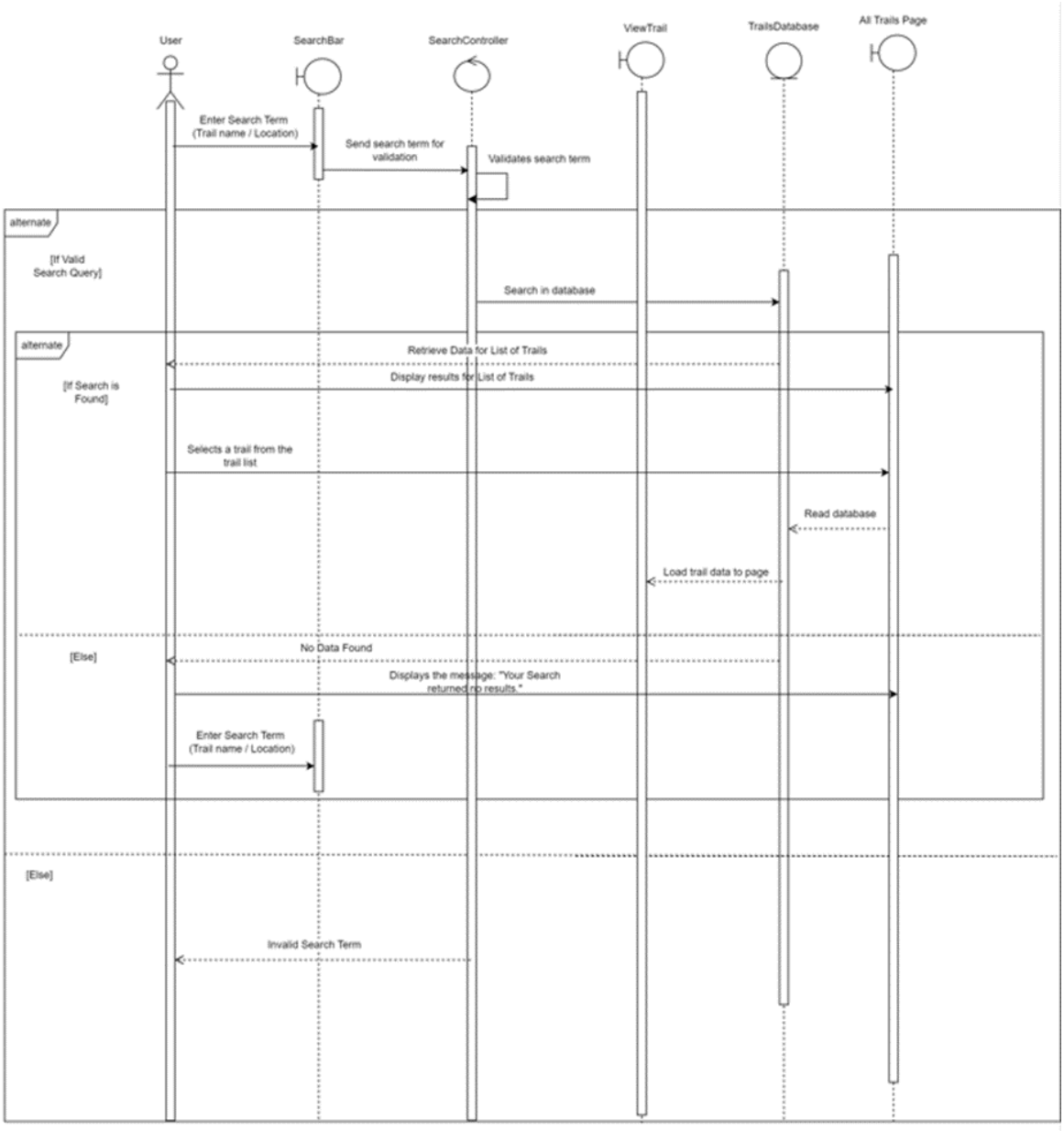


Sequence Diagrams

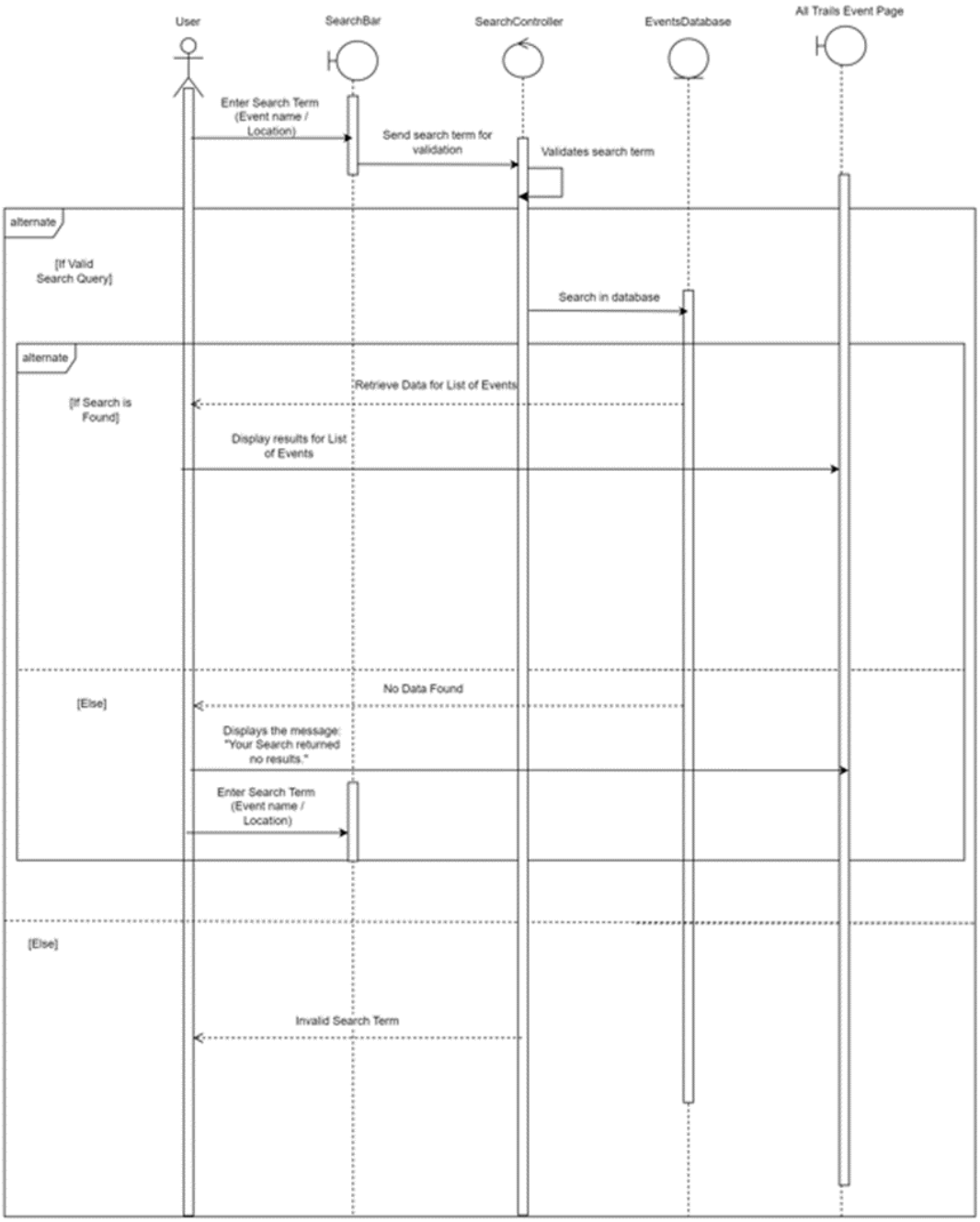
1. Login and Register



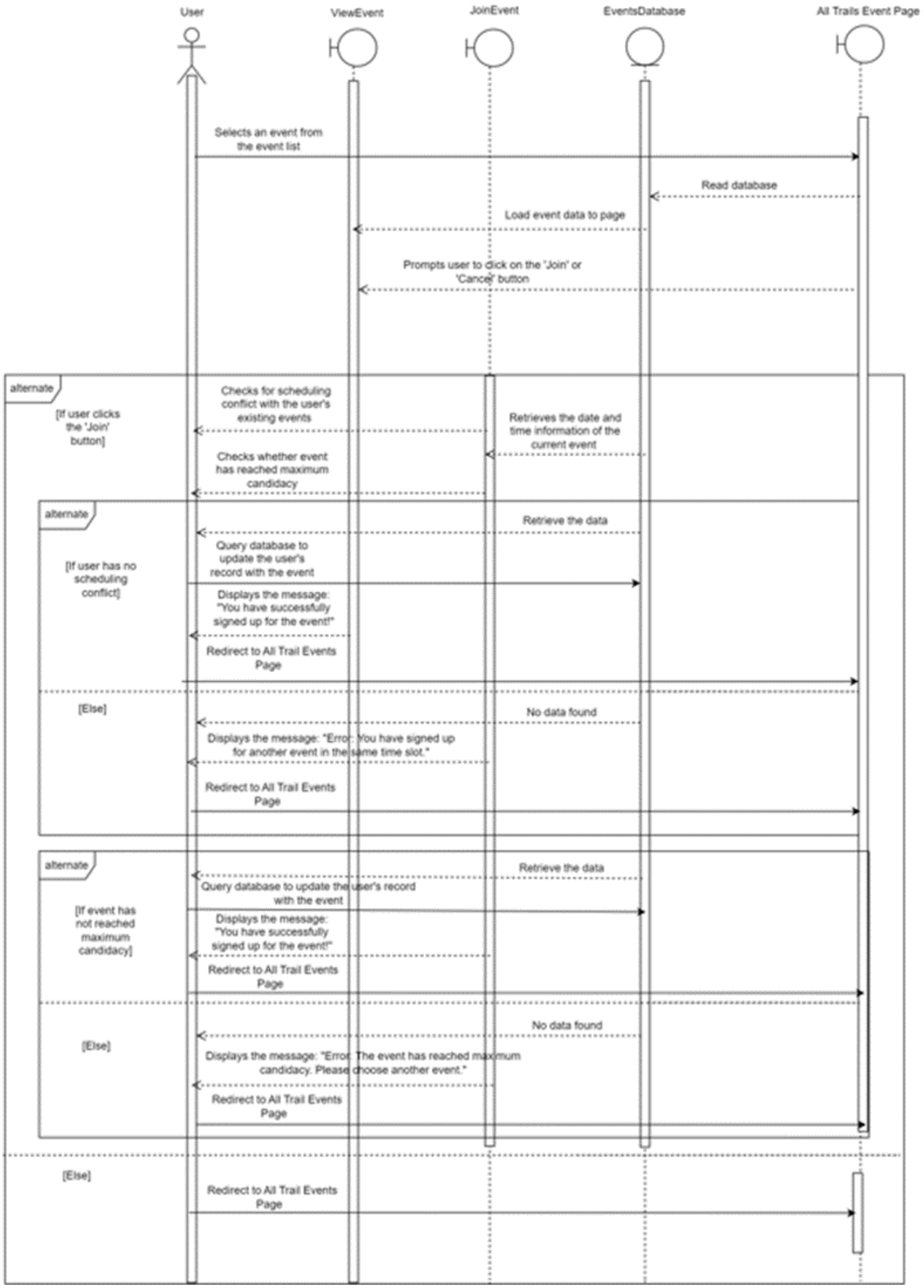
2. SearchTrail and ViewTrail



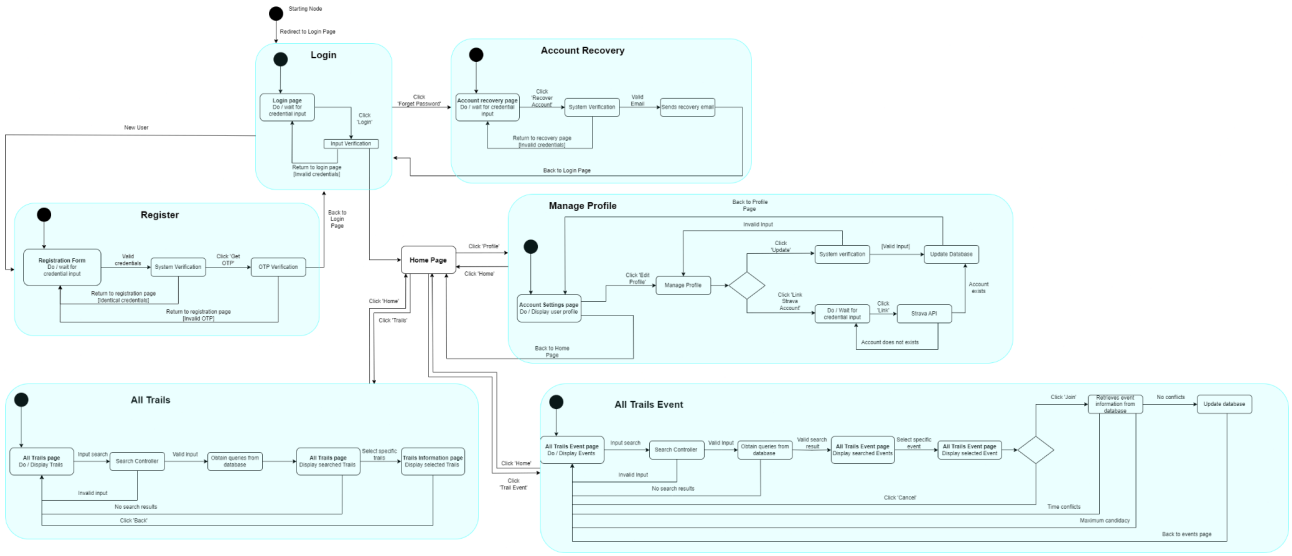
3. SearchEvent



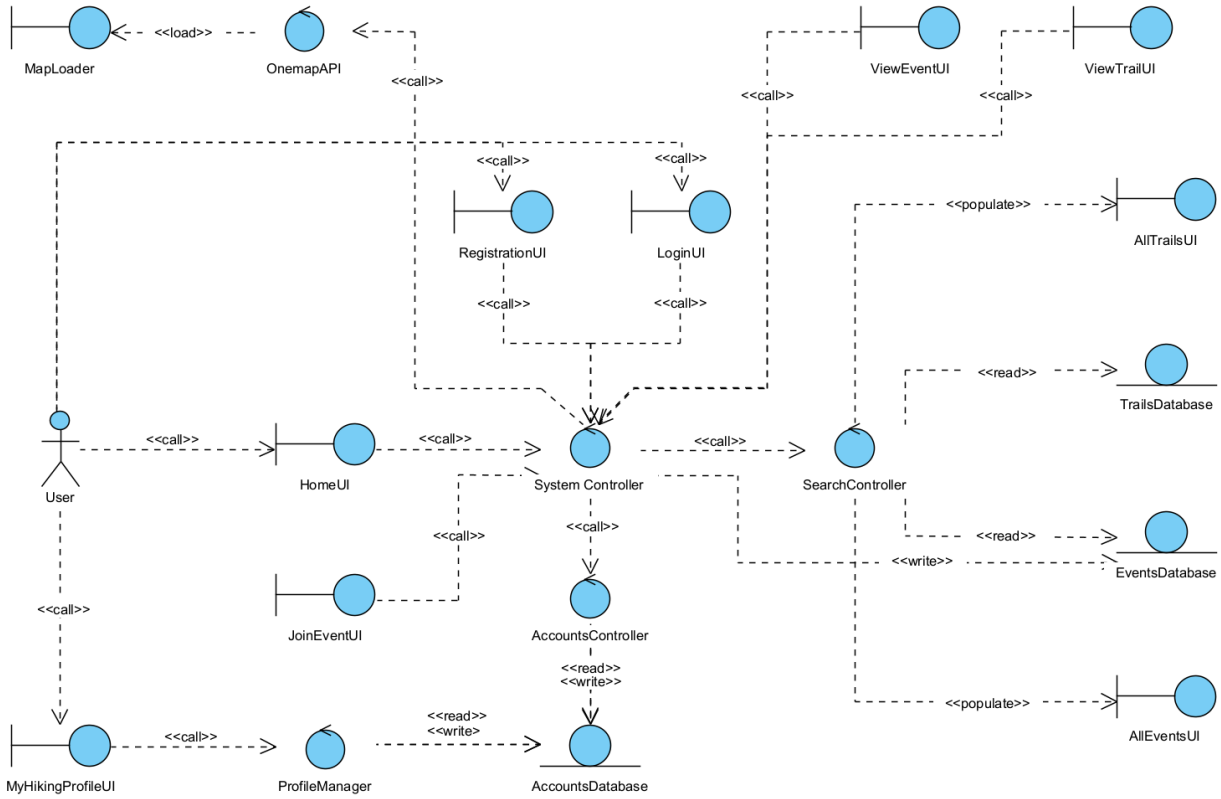
4. ViewEvent


User
ViewEvent
JoinEvent
EventsDatabase
All Trails Event Page
Selects an event from the event list
Read database
Load event data to page
Prompts user to click on the 'Join' or 'Cancel' button
alternate
[If user clicks the 'Join' button]
Checks for scheduling conflict with the user's existing events
Retrieves the date and time information of the current event
Checks whether event has reached maximum candidacy
alternate
[If user has no scheduling conflict]
Retrieve the data
Query database to update the user's record with the event
Displays the message: "You have successfully signed up for the event!"
Redirect to All Trail Events Page
[Else]
No data found
Displays the message: "Error: You have signed up for another event in the same time slot."
Redirect to All Trail Events Page
alternate
[If event has not reached maximum candidacy]
Retrieve the data
Query database to update the user's record with the event
Displays the message: "You have successfully signed up for the event!"
Redirect to All Trail Events Page
[Else]
No data found
Displays the message: "Error: The event has reached maximum candidacy. Please choose another event."
Redirect to All Trail Events Page
[Else]
Redirect to All Trail Events Page

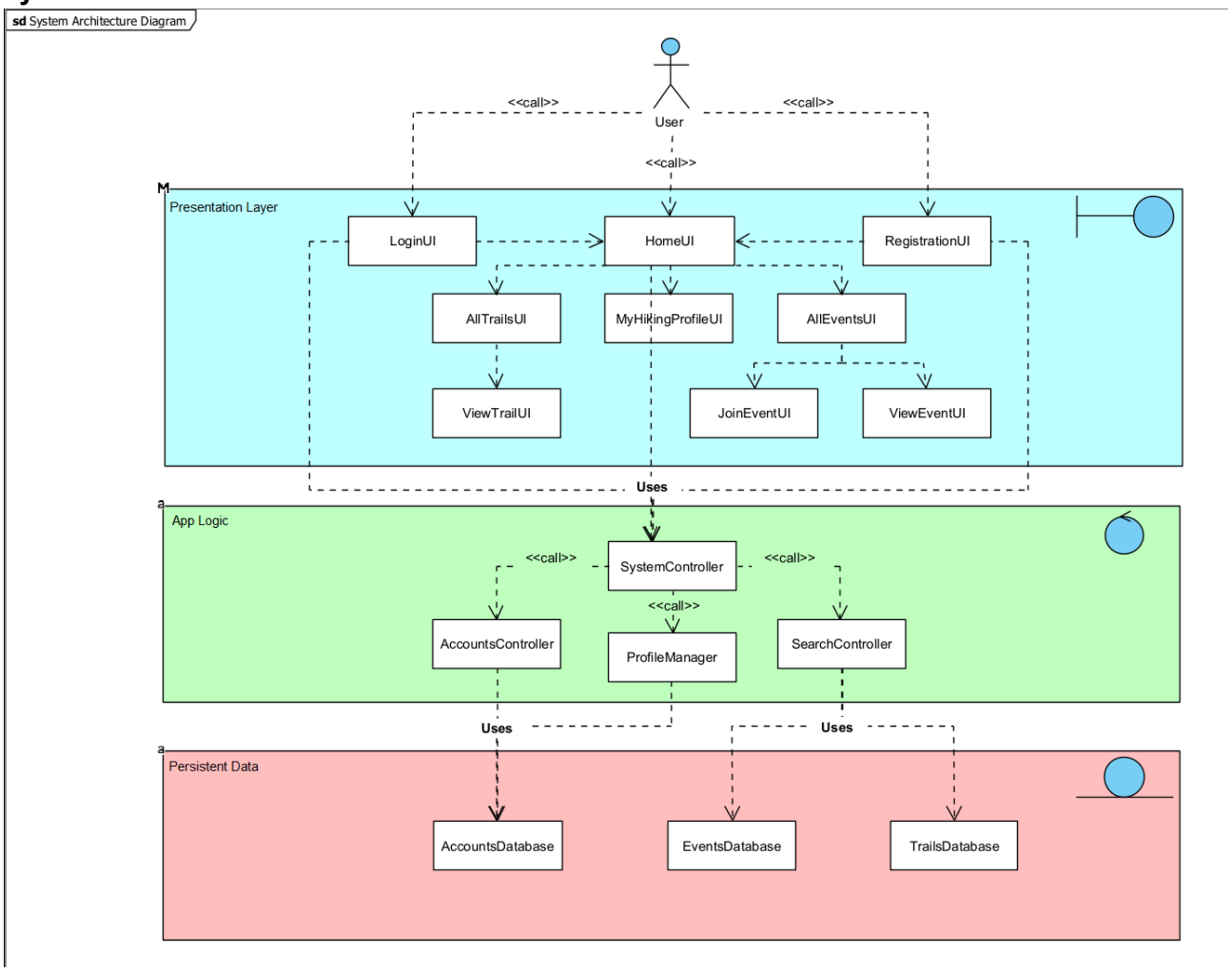
Dialog Map



Boundary, Control, and Entity Diagram



System Architecture



Appendix C: Testing & Control Flow

Black Box Testing

Login Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify successful login with valid email and password	Ensure the login functionality works correctly for valid credentials	User has registered for a user account on the website	Email: jwong165@e.ntu.edu.sg Password: SC2006pwd!02	User is able to log in successfully	1. Navigate to the website's login page 2. Enter the email and password 3. Click on the Log In button	User is logged in successfully	Pass	The user's email and password is added to the database successfully
EC2	Verify error message when email is invalid	Ensure the error message is displayed when the email is invalid	User has registered for a user account on the website	**Email: jwong165** Password: SC2006pwd!02	Error message "Email has to be valid" is displayed Error message (on the bottom left of the page): "Login Unsuccessful. An error occurred. Please try again."	1. Navigate to the website's login page 2. Enter the invalid email and valid password 3. Click on the Log In button	Under the email section, "Email has to be valid" is displayed in red Error message (on the bottom left of the page): "Login Unsuccessful. An error occurred. Please try again."	Pass	

B1	Verify error message when email is empty	Ensure the error message is displayed when the email field is left empty	User has registered for a user account on the website	**Email: (empty)** Password: SC2006pwd!02	Error message: “Email is required.” is displayed .Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”	1. Navigate to the website’s login page 2. Leave the email field empty 3. Enter a valid password 4. Click on the Log In button	Under the email section, “Email is required.” is displayed in red Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”	Pass	
B2	Verify error message when password is empty	Ensure the error message is displayed when the password field is left empty	User has registered for a user account on the website	Email: jwong165@e.ntu.edu.sg **Password: (empty)**	Error message: “Password is required.” is displayed Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”	1. Navigate to the website’s login page 2. Enter a valid email 3. Leave the password field empty 4. Click on the Log In button	Under the password section, “Password is required.” is displayed in red Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”	Pass	
B3	Verify error message when password is	Ensure the error message is displayed	User has registered for a user account on the website	Email: jwong165@e.ntu.edu.sg, **Password:**	Error message: “Password is wrong. Please try again.” is	1. Navigate to the website’s login page	Error message: “Password is wrong. Please try again.” is	Pass	

	wrong	when the password field is wrong		**SC2006**	displayed Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”	2. Enter a valid email 3. Enter the wrong password 4. Click on the Log In button	displayed Error message (on the bottom left of the page): “Login Unsuccessful. An error occurred. Please try again.”		

Register Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify successful sign up with valid information	Ensure the sign up functionality works correctly for valid inputs	User clicks on the sign up button and is on the sign up page	Name: Jeannie, Email: jwong165@e.ntu.edu.sg, OTP: 5233, Password: SC2006pwd!02 Re-Enter Password: SC2006pwd!02	User has signed up successfully for a user account and is logged in into the website	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields 3. Click Next to continue entering the inputs for the required fields (password and re-enter password) 4. Click on the Sign Up button	User has signed up successfully for a user account and is logged in into the website	Pass	
EC2	Verify error message when Password and Re-Enter	Ensure the error message is displayed when the password and	User clicks on the sign up button and is on the sign up page	Name: Jeannie, Email: jwong165@e.ntu.edu.sg , OTP: 5233, **Password:**	Error message: “Password has to match” is displayed Error message	1. Click on the Sign Up button and navigate to the Sign Up page	Error message: “Password has to match” is displayed in red Error message (on	Pass	

	Password do not match	re-enter password fields do not match		**SC2006pwd!02** **Re-Enter Password: SC2006pw02**	(on the bottom left of the page): "Registration Unsuccessful. Something went wrong. Please try again."	2. Enter valid input data for all the required fields 3. Click Next to continue entering the inputs for the required fields (password). However, for the re-enter password field, enter an invalid input 4. Click on the Sign Up button	the bottom left of the page): "Registration Unsuccessful. Something went wrong. Please try again."		
B1	Verify error message when Name is empty	Ensure the error message is displayed when the name field is left empty	User clicks on the sign up button and is on the sign up page	**Name: (empty)**	Error message: "Name is required" is displayed	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields, except for the name field which is to be left empty	Error message: "Name is required" is displayed in red	Pass	

B2	Verify error message when Email is empty	Ensure the error message is displayed when the email field is left empty	User clicks on the sign up button and is on the sign up page	Name: Jeannie, **Email:(empty)**	Error message: “Email is required” is displayed	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields, except for the email field which is to be left empty	Error message: “Email is required” is displayed in red	Pass	
B3	Verify error message when Email is invalid	Ensure the error message is displayed when the email field contains an invalid email address	User clicks on the sign up button and is on the sign up page	Name: Jeannie, **Email: jwong165 (invalid email address)**	Error message: “Email has to be valid” is displayed	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields, except for the email field which has an invalid email address format	Error message: “Email has to be valid” is displayed in red	Pass	

B4	Verify error message when OTP is empty	Ensure the error message is displayed when the OTP field is left empty	User clicks on the sign up button and is on the sign up page	Name: Jeannie, Email: jwong165@e.ntu.edu.sg , **OTP: (empty)**	Error message: “OTP is required” is displayed	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields, except for the OTP field which is to be left empty	Error message: “OTP is required” is displayed in red	Pass	
B5	Verify error message when password is empty	Ensure the error message is displayed when the password field is left empty	User clicks on the sign up button and is on the sign up page	Name: Jeannie, Email: jwong165@e.ntu.edu.sg , OTP: 5233, **Password: (empty)**	Error message: “Password is required” is displayed Error message (on the bottom left of the page): “Registration Unsuccessful. Something went wrong. Please try again.”	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields 3. Click Next and leave the password empty 4. Click on the Sign Up button	Error message: “Password is required” is displayed in red Error message (on the bottom left of the page): “Registration Unsuccessful. Something went wrong. Please try again.”	Pass	

B6	Verify error message when password does not meet requirements	Ensure the error message is displayed when the password field does not meet the requirements	User clicks on the sign up button and is on the sign up page	Name: Jeannie, Email: jwong165@e.ntu.edu.sg , OTP: 5233, **Password: 12345, Re-Enter Password: 12345**	Error message: "Password has to be at least 8 characters long" is displayed Error message (on the bottom left of the page): "Registration Unsuccessful. Something went wrong. Please try again."	1. Click on the Sign Up button and navigate to the Sign Up page 2. Enter valid input data for all the required fields 3. Click Next to continue entering the inputs for the required fields (password and re-enter password) - enter passwords which does not meet requirements 4. Click on the Sign Up button	Error message: "Password has to be at least 8 characters long" is displayed in red Error message (on the bottom left of the page): "Registration Unsuccessful. Something went wrong. Please try again."	Pass	

Forgot Password Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify successful reset password with valid email	Ensure the forgot password functionality works correctly for a valid email address	User has registered for a user account on the website	Email: jwong165@e.ntu.edu.sg (registered email)	User is sent an email with a link to reset password The message : "Password Changed. Don't forget next time!" will be displayed at the bottom left of the page	1. Navigate to the website's login page 2. Click on Forgot Password 3. Enter valid email address registered on the website 4. Click on the Reset Password button	User is sent an email with a link to reset password The message : "Password Changed. Don't forget next time!" will be displayed at the bottom left of the page	Pass	
B1	Verify error message when email field is empty	Ensure the error message is displayed when the email field is left empty	User is on the "Reset your password" page	**Email : (empty)**	Error message "Email is required." is displayed	1. Navigate to the website's login page 2. Click on Forgot Password 3. Leave the email field empty	Error message "Email is required." is displayed in red	Pass	

B2	Verify error message when email field is invalid	Ensure the error message is displayed when the email field contains an invalid email address	User is on the "Reset your password" page	**Email : jwong165 (invalid email)**	Error message "Email has to be valid." is displayed	1. Navigate to the website's login page 2. Click on Forgot Password 3. Enter a invalid email address	Error message "Email has to be valid." is displayed in red	Pass	
B3	Verify user is able to login into existing user account with new password	Ensure the forgot password functionality works correctly for a valid email address, the user is able to reset password which matches the requirements and is able to login into the existing user account with a new password	User has registered for a user account on the website	Email: jwong165@e.ntu.edu.sg (registered email) Current Password : SC2006pwd!02 **Reset Password: SC200602!**	User is able to reset password successfully and login into the existing user account with a new password The message : "Password Changed. Don't forget next time!" will be displayed at the bottom left of the page An alert box stating that "Password changed. You can now sign in with your new password."	1. Navigate to the website's login page 2. Click on Forgot Password 3. Enter a valid email address 4. Click on the Reset Password button 5. Click on the reset password link in the email 6. Type in a new password which matches the requirements (uppercase,	User is able to reset password successfully and login into the existing user account with a new password The message : "Password Changed. Don't forget next time!" will be displayed at the bottom left of the page An alert box stating that "Password changed. You can now sign in with your new password."	Pass	

						lowercase characters, with numbers and symbols which is at least 8 characters long) 7. Click on save 8. Navigate to the login page 9. Enter the existing email address and new password 10. Click login			

View Profile Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify that the user is able to access his profile and able to click on the link to authenticate his user account with Strava API	Ensure that the profile page is accessible and the link is working	User is on the profile page	Click on the link 'Authenticate with Strava'	User is able to access the profile page and click on the link which will redirect him to Strava API	1. Select the profile option in the menu 2. Click on the 'Authenticate with Strava' link 3. User will be redirected to the Strava API page	User is able to access the profile page and click on the link which will redirect him to Strava API	Pass	
EC2	Verify that the user is able to connect his user account to Strava API	Ensure that the user account on PeakVisor is able to connect to Strava API	User is on the profile page	1. Click on the link 'Authenticate with Strava' 2. Email: jwong165@e.ntu.edu.sg, Password: SC2006pwd!02 3. Click Log In 4. Click Authorize	The user can click the link to authenticate his account with Strava API, log in using his email address and password, and grant PeakVisor permission to connect to Strava API	1. Select the profile option in the menu 2. Click on the 'Authenticate with Strava' link 3. User will be redirected to the Strava API page 4. Enter the email and password 5. Click Log In	The user can click the link to authenticate his account with Strava API, log in using his email address and password, and grant PeakVisor permission to connect to Strava API	Pass	

						6. Click Authorize when asked to authorize PeakVisor to connect to Strava			
B1	Verify that the user is able to view his previous hiking details	Ensure that the user's previous hiking details and statistics are displayed on the page	User is on the profile page and User account is connected to Strava API	1. Click on the link 'Authenticate with Strava' 2. Click Authorize	User is able to view his previous hiking details and their statistics	1. Select profile option in the menu 2. Click on the 'Authenticate with Strava' link 3. User will be redirected to the Strava API page 4. Click Authorize when asked to authorize PeakVisor to connect to Strava 5. User will be redirected back to the PeakVisor website where he can view his previous hiking details	User is able to view his previous hiking details and their statistics	Pass	

Manage Profile Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify successful editing of user account details	Ensure that the user is able to edit his user account details and the updated user account information will be displayed and enabled after the user edits it	User is on the Account Settings page	**Name: Jean (edited)** Strava: Not Connected, Email: jwong165@e.ntu.edu.sg **Change Password: SC2006password!02 (edited)**	Updated user account details will be displayed and user can login with updated details A prompt saying "A password reset email has been sent to you. Check your email!" will be displayed in green at the side. An alert box stating that "Password changed. You can now sign in with your new password.	1. Select the settings option in the menu 2. Click on edit under the name field and edit the name 3. Click on save under the name field to save changes 4. Click on the change password button - an email will be sent to the user's inbox 5. Click on the reset password link in the email 6. Type in a new password which matches the requirements (uppercase, lowercase characters, with numbers and symbols which is at	Updated user account details will be displayed and user can login with updated details A prompt saying 'A password reset email has been sent to you Check your email!' will be displayed in green at the side. An alert box stating that "Password changed. You can now sign in with your new password.	Pass	Name and Password for the particular user account will be updated onto the database

						least 8 characters long) 7. Click on save 8. Navigate to the login page 9. Enter the existing email address and new password 10. Click login			
EC2	Verify that the user is able to delete his user account	Ensure that the user is able to delete his user account and user account will be removed after 7 days	User is on the Account Settings page	Click on Delete Account	User will be automatically logged out of their user account and it will be removed from the database after 7 days	1. Select the settings option in the menu 2. Click on delete account	User will be automatically logged out of their user account and it will be removed from the database after 7 days	Pass	User account will be removed from the database after 7 days
B1	Verify error message when name is empty	Ensure the error message is displayed when the name field is empty	User is on the Account Settings Page	**Name: (empty)**	Error message “Name is required.” is displayed	1. Select the settings option in the menu 2. Click on edit under the name field and leave it empty 3. Click on save under the name field to save changes	Error message “Name is required.” is displayed in red	Pass	

B3	Verify error message when name is invalid	Ensure the error message is displayed when the name field is invalid	User is on the Account Settings Page	**Name: 123jean (invalid name)**	Error message “Name is invalid.” is displayed	1. Select the settings option in the menu 2. Click on edit under the name field and enter an invalid input 3. Click on save under the name field to save changes	Error message “Name is invalid.” is displayed in red	Pass	

View Trails Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify that specific information of the trail will be displayed when the user selects it	Ensure that information of the trail selected will be displayed	User is on the Trails Page	Select a specific trail - e.g. 'Macritchie Nature Trail'	General information, address, bearings and rating of the specific trail will be displayed	1. Select Trails at the top of the website 2. Select 'Macritchie Nature Trail'	General information, address, bearings and rating of the specific trail is displayed	Pass	

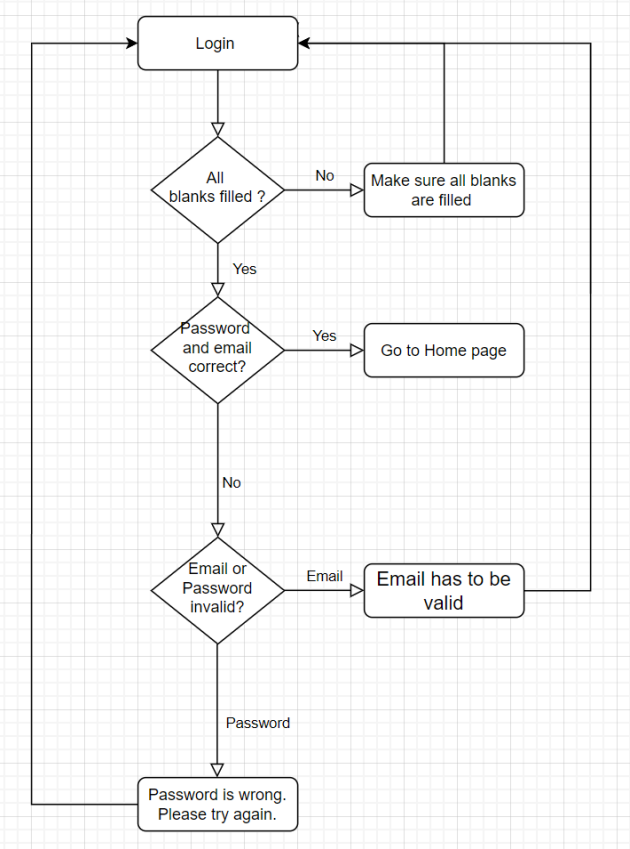
View Events Functionality

Test Case ID	Test Case Description	Test Objective	Preconditions	Inputs	Expected outcome	Test Steps	Actual outcome	Pass/ Fail	Notes
EC1	Verify that specific information of the event will be displayed when the user selects it	Ensure that information of the event will be displayed when it is selected	User is on the Events Page	Select a specific event - e.g. '1D Gunong Arong Hike'	Details of the event (date/time, location, attendees etc;) will be displayed and user can join the event	1. Select Events at the top of the website 2. Select '1D Gunong Arong Hike'	Details of the event (date/time, location, attendees etc;) will be displayed and user can join the event	Pass	

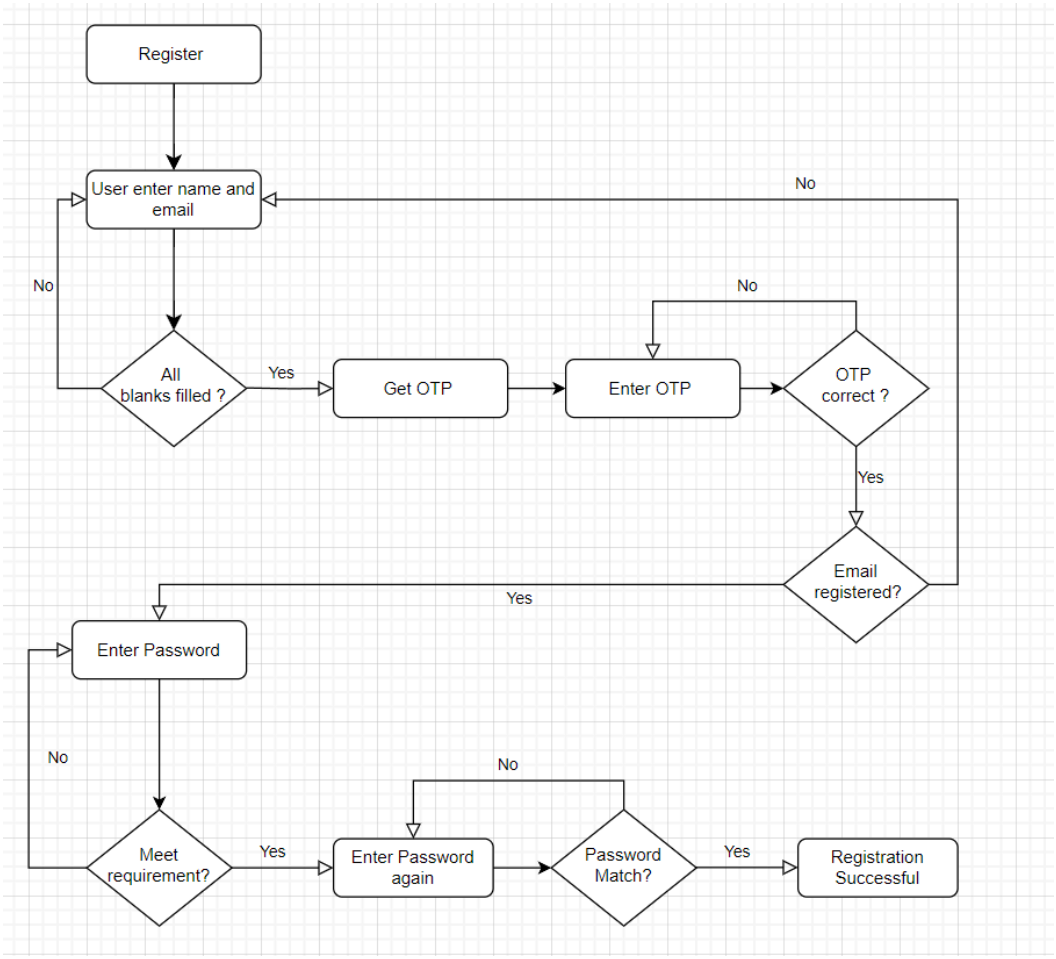
White Box Testing

Control Flow Diagrams

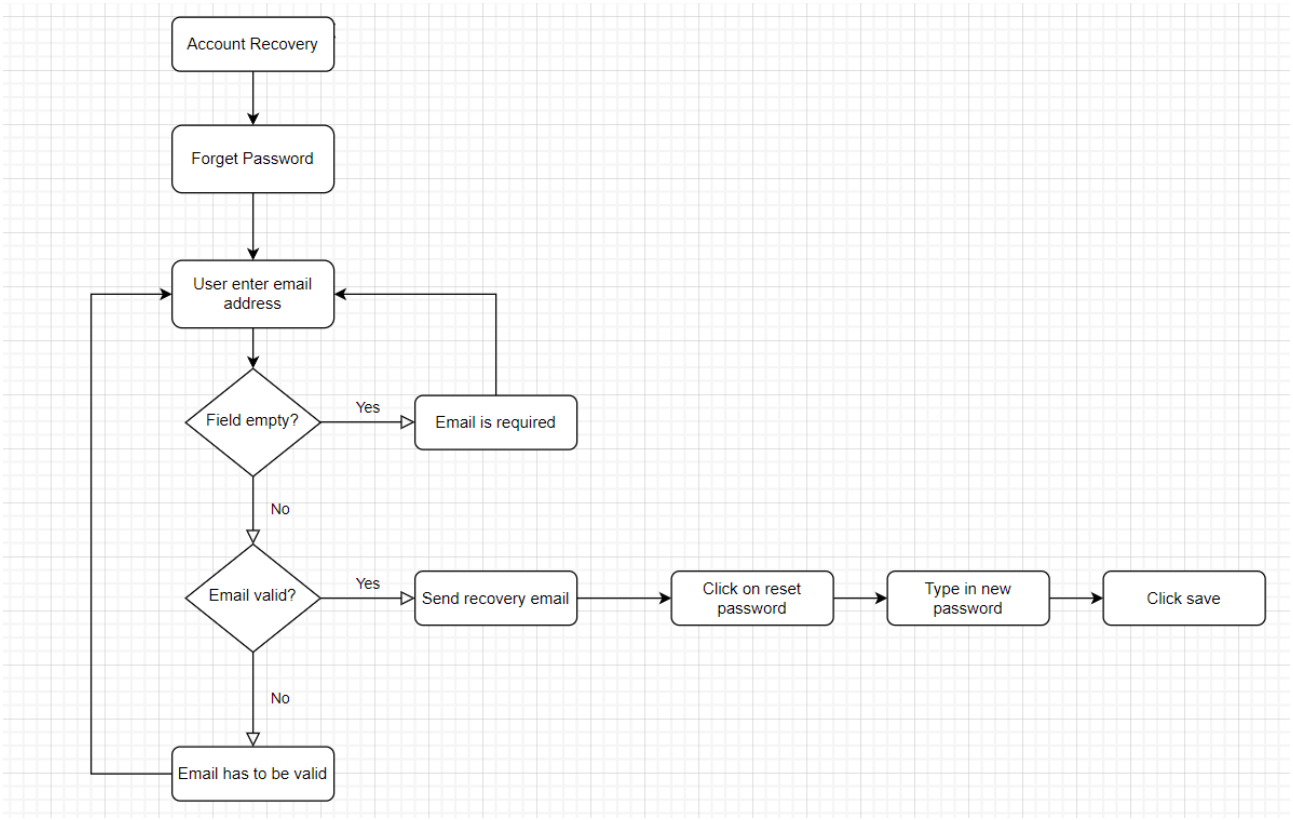
Functionality: Login



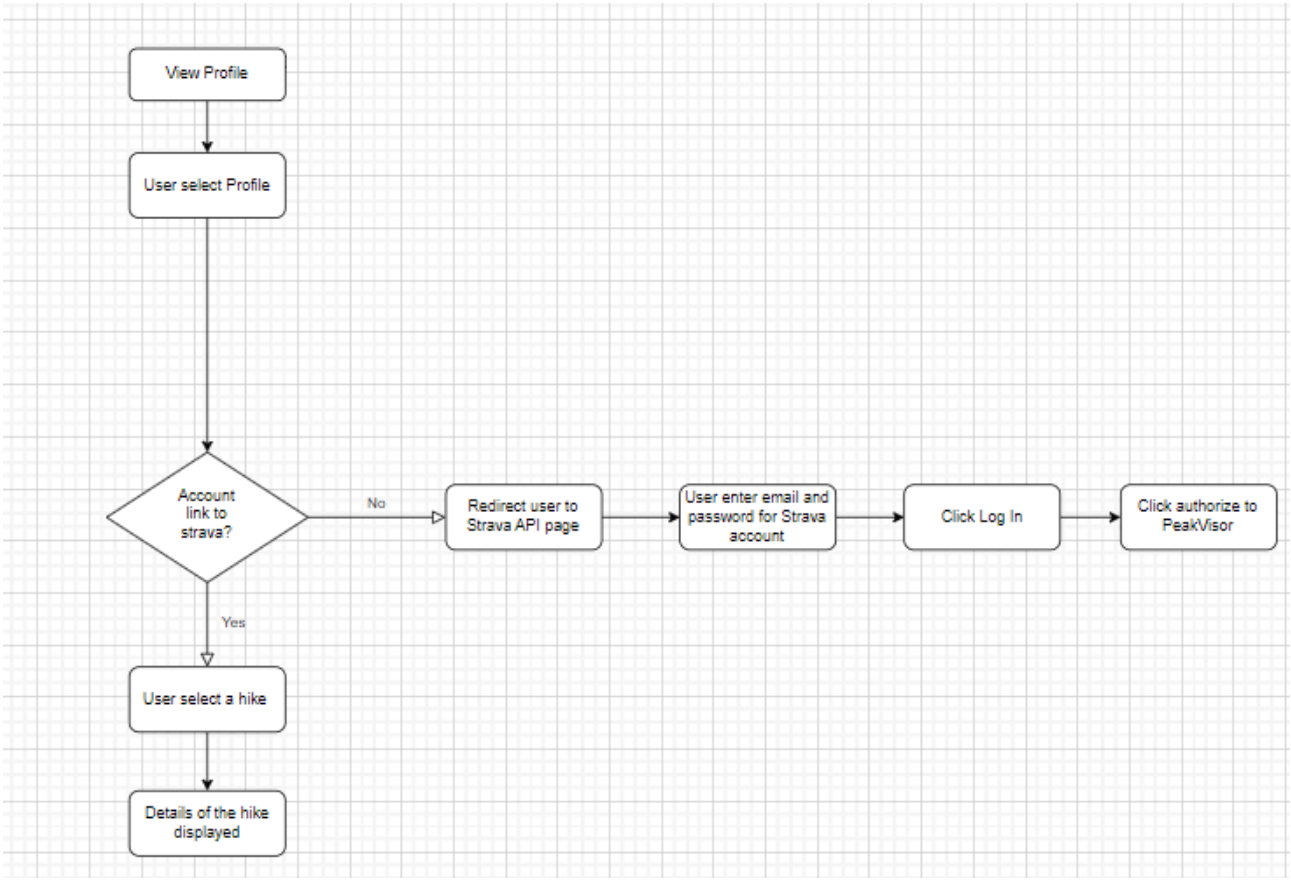
Functionality: Register



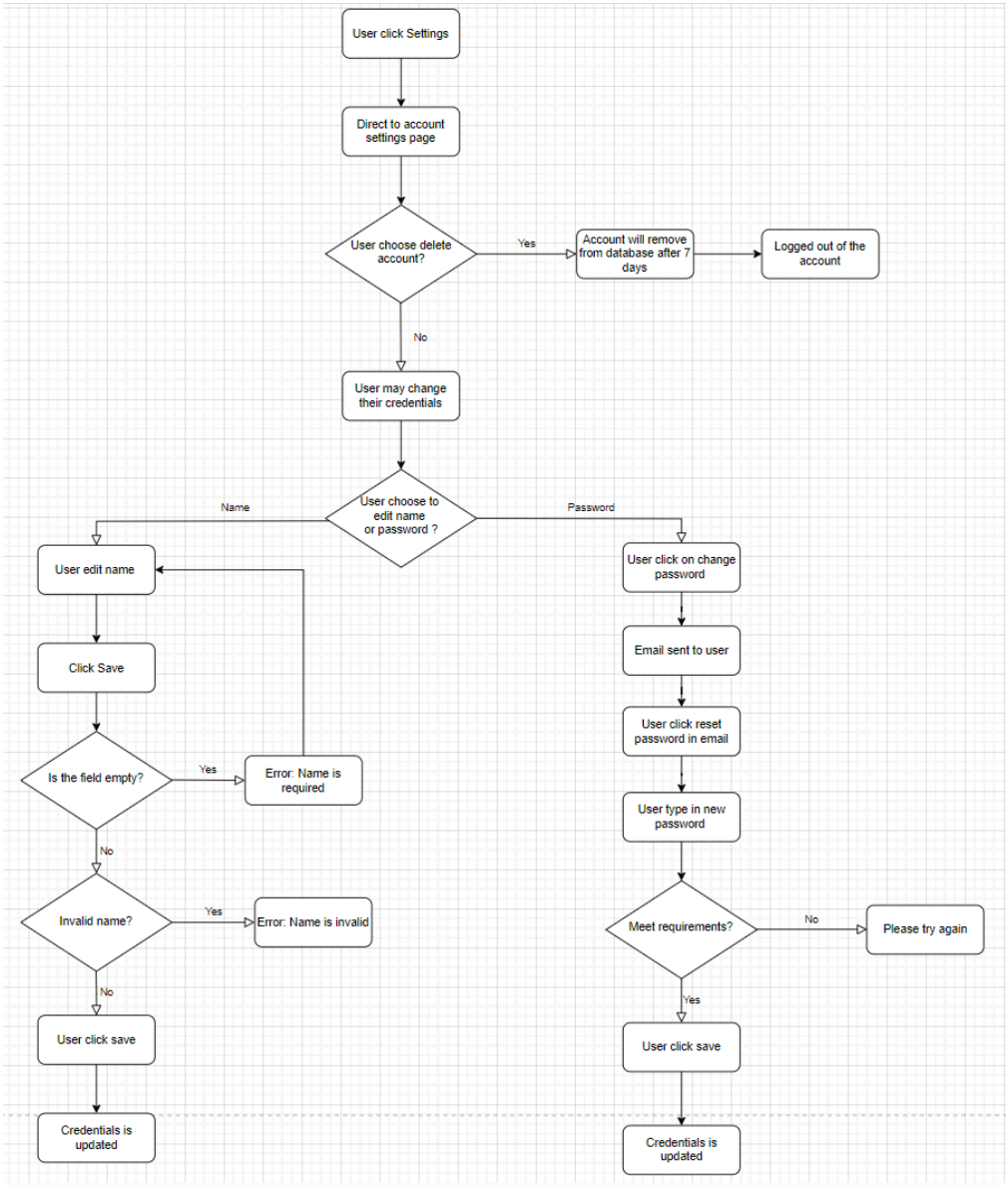
Functionality: Forgot Password



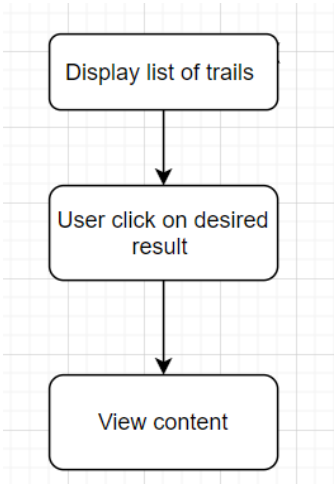
Functionality: View Profile



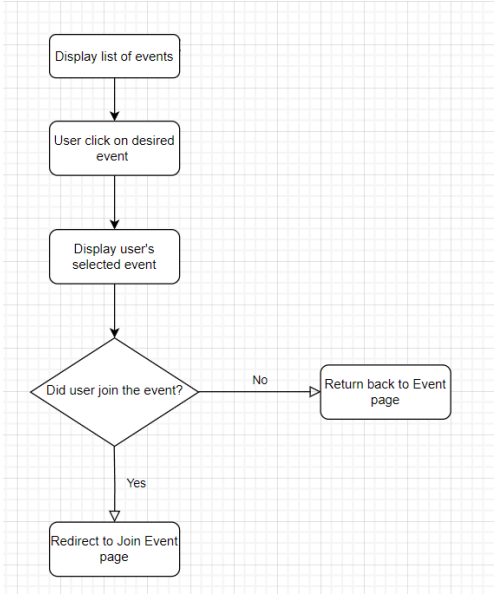
Functionality: Manage Profile



Functionality: Search Trails & View Trails



Functionality: Search Events & View Events



Source: http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc