Software Requirements Specification

for

<Urban>

Version 1.1 approved

Prepared by <Arun Ezekiel, Dinesh, Kai, Ryan, Wei Xian>

<SC2001 Software Engineering Z59_Group1>

<22/03/2023>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Lee Wei Xian	22/03/2023	Updated Community Functionality Minor edit in flow of use cases	1.1
Kan Huai feng, Kai	08/04/2023	Updated all use cases	1.2

1. Introduction

1.1 Purpose

This document details the software requirements of the android application “Urban”. It defines the feature requirements, use cases and control flow of the application.

1.2 Product Scope

Urban is a mobile application designed to cater to the fitness and well-being of its users. With its powerful GPS tracking feature, Urban allows users to monitor and analyze their running or cycling sessions with precision and ease, providing detailed reports on users' progress, including historical data and goal tracking to help them stay motivated and on track with their fitness objectives.

The benefits of Urban are vast, offering users a comprehensive solution to their fitness tracking needs. The app's sophisticated design and user-friendly interface ensure that users can seamlessly navigate through its features and functions, making the tracking experience as smooth and effortless as possible.

For those who are already using the "Healthy365" Singapore app, the Urban mobile application can complement and enhance their experience. While "Healthy365" focuses primarily on tracking health metrics and providing personalized health advice, Urban offers a specialized fitness tracking solution with GPS tracking capabilities, enabling users to monitor and analyze their running or cycling sessions in greater detail.

2. Overall Description

2.1 Product Perspective

Urban is an application that enhances several current product families. By utilizing GoogleMaps APIs, Urban provides precise tracking and visualization of the map, while also personalizing the user experience with MongoDB by storing and displaying past cycling sessions and goals. Urban is developed using React Native and Expo.

2.2 Product Functions

Major Functions:

- Login
- Register for an Urban account
- Display Map
- Search for destination and find route
- Track Cycling / Running session
- View session history

Minor Functions:

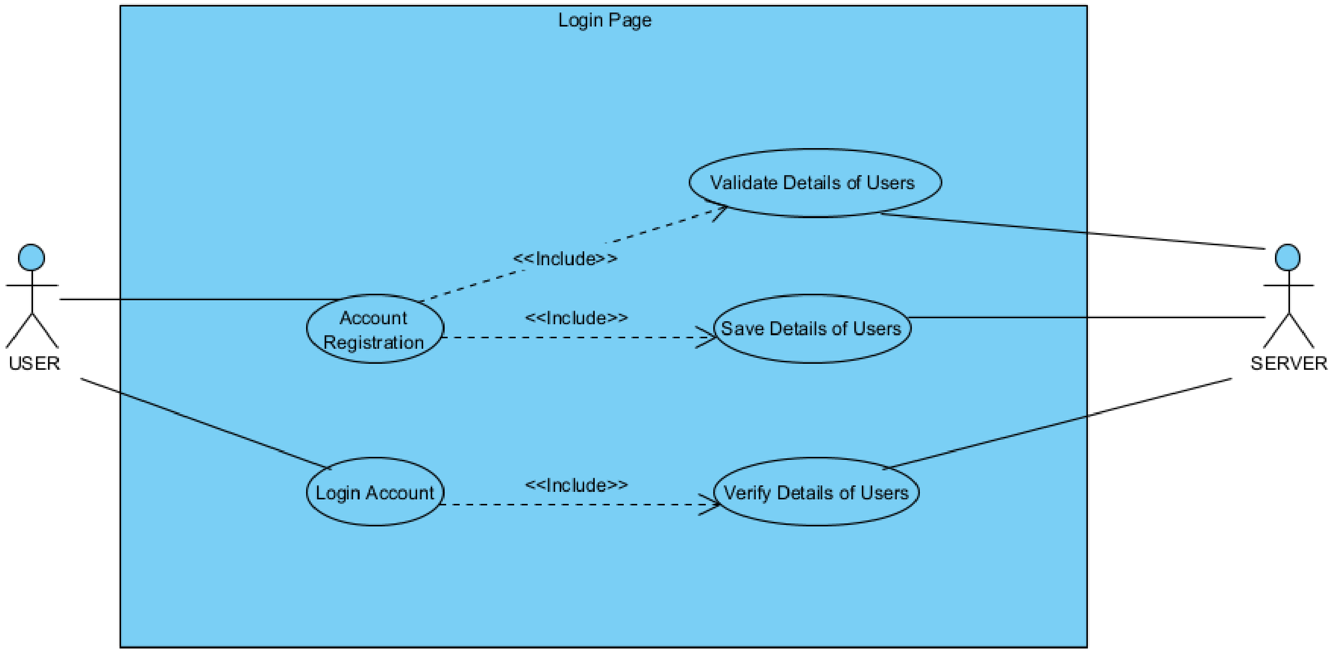
- Set and Edit goal
- Manage profile
- Pause/Resume live session

2.3 Assumptions and Dependencies

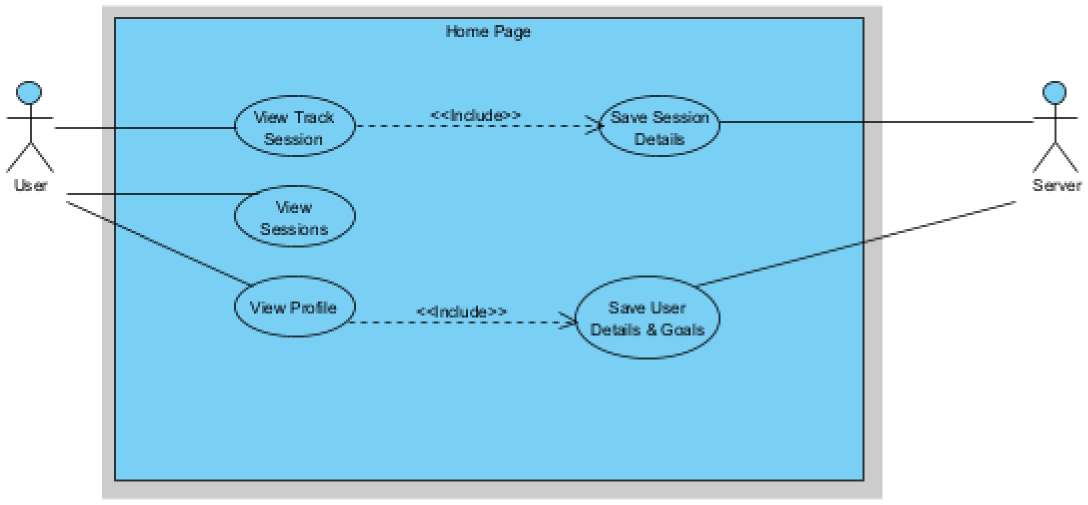
The team assumes that users are connected to the internet and that they have their location services enabled.

2.4 Use case diagrams for Functionalities

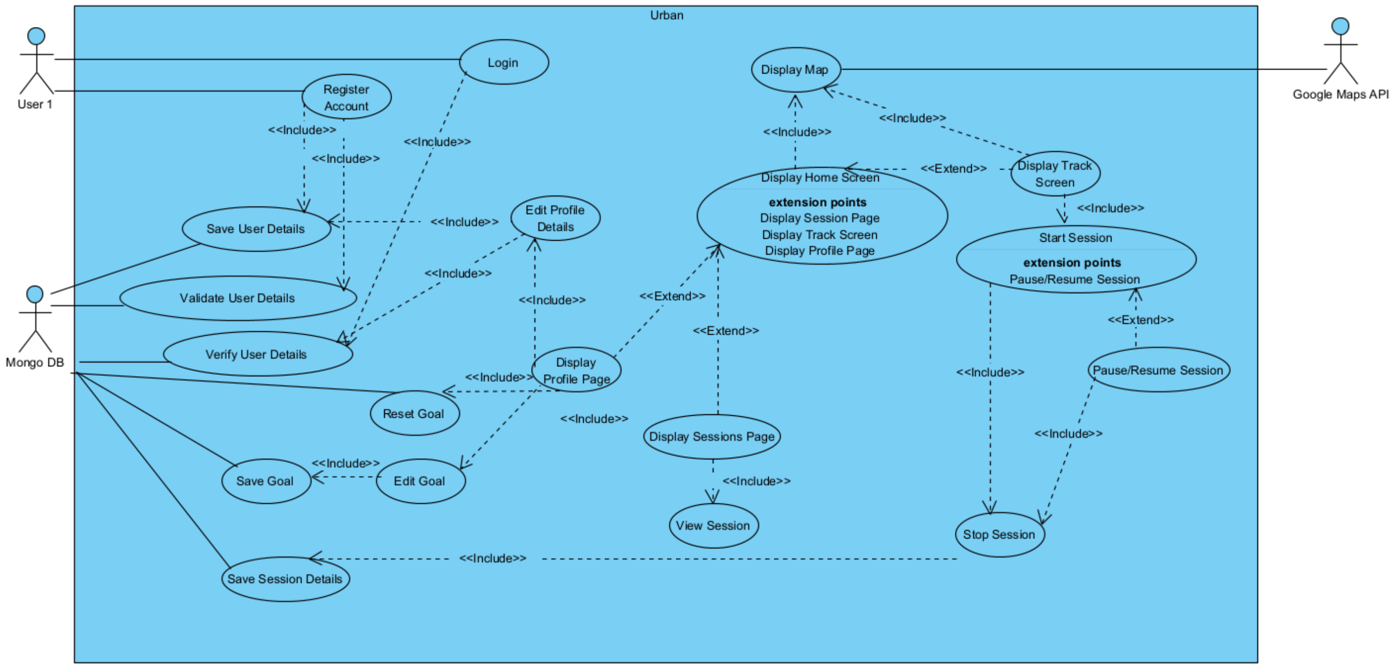
2.4.1 Use Case Diagram for Login Page "Urban"



2.4.2 Use Case Diagram for Home Page "Urban"



2.4.3 Full Use Case Diagram for "Urban"



3. System Feature

3.1 Account Registration

Use Case ID:	UC001		
Use Case Name:	Account Registration		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB
Description:	Users have to register for an account if they do not possess one to use the functionalities of the application.
Preconditions:	User does not have a valid account and wants to register for one. The user must be on the application's start page.
Postconditions:	Account is created in MongoDB with input details. The user will be redirected to the application's home page.
Priority:	High
Frequency of Use:	Low
Flow of Events:	1. The user presses on the <**Register**> button on the application's start page. 2. The system displays the application's registration page. 3. The user enters a username, password, and a unique email. 4. The user submits valid username, password, and email by pressing the <**Register**> button. 5. The server validates the input information and sends a request to MongoDB. 6. MongoDB ensures the email input is unique. 7. MongoDB saves the username, password, and email. 8. The system displays a successful registration message. 9. The system displays the application's home screen. 10. The use case ends.
Alternative Flows:	UC001.AC.1 Invalid username, password, or email 1. The system prompts the user for the invalid inputs. 2. Use case resumes at main flow step 3. UC001.AC.2 Email taken 1. The system displays an "Email registered" message. 2. The system prompts the user for an unregistered email. 3. Use case resumes at main flow step 3.
Exceptions:	UC001.EX.1 Cancellation of User Registration 1. The user aborts the use case by pressing on the <**Back>** button. 2. The system does not save any details. 3. The user will be redirected to the application's start page.

	4. The use case ends.
Includes:	SVR.UC001, SVR.UC002
Special Requirements:	
Assumptions:	When the user enters the application's registration page, the user must successfully create an account.
Notes and Issues:	

3.2 Login Account

Use Case ID:	UC002		
Use Case Name:	Login Account		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB
Description:	The user indicates intent to log in to his previously registered account with his registered email and password to access the functionalities of the application.
Preconditions:	User has previously created a valid account (stored in MongoDB). User is on the application's start page.
Postconditions:	User successfully logs in. The user is redirected to the application's home screen.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The user presses on the <**Login**> button on the application's start page. 2. The system displays the application's login page. 3. The user enters a registered email and the associated password. 4. The user submits the email and password by pressing the <**Login**> button and sends a request to MongoDB. 5. MongoDB verifies the email and password. 6. The system displays the application's home screen. 7. The use case ends.
Alternative Flows:	UC002.AC.01 Missing email and/or password 1. The system prompts for missing email and/or password. 2. The use case resumes at main flow step 3. UC002.AC.02 Incorrect email and/or password 1. The system displays an "Incorrect email and/or password" message. 2. The system prompts for an email and password. 3. The use case resumes at main flow step 3.
Exceptions:	UC002.EX.1 Cancellation of Account Login 1. The user aborts the use case by pressing on the <**Back**> button. 2. The system displays the application's start page. 3. The use case ends.
Includes:	SVR.UC003
Special Requirements:	
Assumptions:	When the user enters the application's account login page, the user must successfully login to their registered account.
Notes and Issues:	

3.3 Manage Profile

Use Case ID:	UC003		
Use Case Name:	Manage Profile		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB
Description:	The user indicates interest to edit his profile. The profile will display his username, email, profile picture, current goal progress, goal and goal completion status. The user can edit his profile details.
Preconditions:	User is logged in. User is on the application's home page.
Postconditions:	The user's information is updated on the MongoDB.
Priority:	High
Frequency of Use:	Mid
Flow of Events:	1. The user presses the <**Profile**> button on the application's home page. 2. The system displays the application's profile page. 3. Use case ends.
Alternative Flows:	UC003.AC.01 Edit profile 1. The user can make changes to his profile by pressing the <**Edit**> icon on the application's profile page. 2. Users can reset their goal process by pressing the <**Reset**> icon on the application's profile page. 3. The system displays the application's edit profile page. 4. Users can update his profile by inputting a new username. 5. Users can update his profile picture by uploading a new picture. 6. Users can save his changes by pressing the <**Save**> button. 7. The server sends a request to MongoDB. 8. MongoDB will update the user profile information. 9. The use case resumes at main flow step 2. UC003.AC.02 Cancel edit 1. The user can revert any changes to his profile during current edit by pressing the <**Cancel**> button on the application's edit profile page. 2. The use case resumes at main flow step 2.
Exceptions:	UC003.EX.01 Invalid username, first name and/or last name 1. The system will display a "Invalid username, first name and/or last name" message. 2. The system will prompt the user to enter a new username, first name and/or last name. 3. The use case resumes at alt flow UC004.AC.01 step 3.
Includes:	SVR.UC002
Special Requirements:	
Assumptions:	

Notes and Issues:	

3.4 Manage Tracking

Use Case ID:	UC004		
Use Case Name:	Manage Tracking		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB, Google Maps API
Description:	The user indicates intent to record a session. The user can start, stop, pause and resume a session. The session will show the user's current session details, including his current location on the map, the route from the initial start point of the session to the user's current location on the map, total distance travelled, time taken, and average speed.
Preconditions:	User is logged in. User is on the application's home page.
Postconditions:	The session is stopped and the user's session details are stored on MongoDB.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The user can select the type of session by pressing either the <**Cycle**> or <**Run**> icon. 2. The user can change the session name by pressing <**Urban Adventure**>. 3. The user starts a session by either pressing the <**Start**> icon. 4. The system will start recording the session. 5. The system will constantly update the current session's details. 6. The user stops the current session by pressing the <**Stop**> button. 7. The system will stop recording the session and send a request to MongoDB. 8. MongoDB will save the session with the user's id and session's details. 9. The system displays the session details. 10. The system displays the application's home page.
Alternative Flows:	UC004.AC.01 Pause and Resume 1. The user can pause a current session by pressing the <**Pause**> button either on main flow step 2 or 3. 2. The system will temporarily stop updating the session. 3. The system will continue updating the session after the user presses the <**Resume>** button. 4. The use case resumes at main flow step 3.
Exceptions:	
Includes:	SVR.UC004
Special Requirements:	
Assumptions:	The user will not <**Pause**> the session indefinitely.

	The user will eventually press the <**Stop**> button.
Notes and Issues:	

3.5 View Session

Use Case ID:	UC005		
Use Case Name:	View Session		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB
Description:	The user indicates the intent to view details about his past sessions. The details include the name of the session, the type of the session, date, start time, distance travelled, timing, average speed and cycling route of the session.
Preconditions:	User is logged in. User has past sessions. User is on the application's home page.
Postconditions:	The system displays the relevant information about the user's past sessions.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The user can view his past sessions by pressing the <**Sessions>** button on the application's home page. 2. The system displays the application's past sessions which were saved in MongoDB. 1. 3. The user can press on the sessions to view past session details. 4. The system will display the date, end time, distance travelled, timing and route of the session. 2. 5. The use case ends.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

3.6 Edit Goal

Use Case ID:	UC006		
Use Case Name:	Edit Goal		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	User, MongoDB
Description:	The goal will be displayed on the top of the application’s session page with a default goal of 50 kilometres. The goal counter will start with a value of 0 and is the sum of distance travelled during the sessions recorded. The user indicates the intent to set a new goal.
Preconditions:	User is logged in. User is on the application’s session page.
Postconditions:	The goal is successfully updated on MongoDB. The goal is shown on the session page.
Priority:	High
Frequency of Use:	Mid
Flow of Events:	1. The user can edit the goal by pressing the <**Edit**> icon on the application’s sessions page. 3. 2. The user inputs the goal (in kilometres). 3. The user submits the goal by pressing the <**Confirm**> button. 4. The system validates the goal and sends a request to MongoDB. 4. 5. The user goal is updated on the MongoDB. 5. 6. The system successfully displays the new goal. 6. 7. The use case ends.
Alternative Flows:	
Exceptions:	UC006.EX.1 Invalid goal exceeding 999KM 1. The system displays “Goal must not exceed 999KM” message. 2. The system prompts the user to enter a valid goal. 3. The use case resumes at main flow step 2. UC006.EX.2 Cancellation of Goal Setting 1. User aborts the use case by pressing the <**X**> icon. 2. The use case ends.
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

3.7 SVR.UC001 Validate User Details

Use Case ID:	SVR.UC001		
Use Case Name:	Validate User Details		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	Server, MongoDB
Description:	The use case begins when the server receives user details to validate. The use case ends when the input email is unique, while username and password is filled and valid.
Preconditions:	Email, username and password entered.
Postconditions:	Accurately verified the user details and sends an appropriate message to the system.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. 1. The server receives user details. 2. The server validates that all user details, including email. username and password, are not missing and are valid. 2. 3. The server sends a request to MongoDB to validate input email. 4. The MongoDB validates the email is not used and in valid format. 5. The server successfully validates all the user details. 3. 6. The use case ends.
Alternative Flows:	1. SVR.UC001.AC1 Email Taken 1. The server sends an “Email Taken” message to the system. 2. The use case ends. SVR.UC001.AC2 Empty inputs 1. The server sends a “Missing inputs” message to the system. 2. The user case ends.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

3.8 SVR.UC002 Save User details

Use Case ID:	SVR.UC002		
Use Case Name:	Save User details		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	Server, MongoDB
Description:	The use case begins when the actor receives validated user details - email, username and password.
Preconditions:	User details, including email, password and username validated.
Postconditions:	User details are saved successfully on MongoDB.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The server sends a request to MongoDB with validated user details. 1. 2. The server saves the email on MongoDB as a key. 2. 3. The server saves the username and password on MongoDB. 3. 4. The server successfully saved all user details on MongoDB. 4. 5. The use case ends.
Alternative Flows:	
Exceptions:	SVR.UC002.EX1 Storage space full 1. The server aborts the case due to not enough space left in storage. 2. The server sends a “Insufficient Storage, please contact an admin” message to the system. 3. The use case ends.
Includes:	
Special Requirements:	
Assumptions:	The user details will always be successfully saved after validation of details.
Notes and Issues:	

3.9 SVR.UC003 Verify Login details

Use Case ID:	SVR.UC003		
Use Case Name:	Verify Login details		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	Server, MongoDB
Description:	The use case begins when the actor receives user details to verify and ends when it has been verified with MongoDB.
Preconditions:	Valid email and associated password entered in the login page.
Postconditions:	Accurately verifies the authenticity of email and password pair with MongoDB and server sends an appropriate message to the system.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The server receives an email and password pair. 2. The server verifies the email is in MongoDB and the password. received matches with the password stored in MongoDB. 3. The server sends a “Verified” message to the system. 4. The use case ends.
Alternative Flows:	SVR.UC003.AC1 Email not found 1. The server sends a “Email not found” message to the system. 2. The use case ends. SVR.UC003.AC2 Password mismatch 1. The server sends a “Password mismatch” message to the system. 2. The use case ends.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

3.10 SVR.UC004 Save Session details

Use Case ID:	SVR.UC004		
Use Case Name:	Save Session details		
Created By:	Arun Ezekiel	Last Updated By:	Kan Huai Feng, Kai
Date Created:	01/02/2023	Date Last Updated:	08/04/2023

Actor:	Server, MongoDB
Description:	The use case begins when the server receives session details - Route, end time, total distance travelled and time taken.
Preconditions:	User is logged in. User has stopped a session on the session page.
Postconditions:	User session details are saved successfully on MongoDB.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. 1. The server receives user session details. 2. 2. The server saves the session details, including user id, route, total distance travelled, time taken, and end time on MongoDB. 3. 3. The use case ends.
Alternative Flows:	
Exceptions:	SVR.UC004.EX1 Storage space full 1. The server aborts the case due to not enough space left in storage. 2. The server sends a “Insufficient Storage, please contact an admin” message to the system. 3. The use case ends.
Includes:	
Special Requirements:	
Assumptions:	The user details will always be successfully saved.
Notes and Issues:	