
# Software Requirements Specification

## for

# A5-swimiNIT

**Version 1.0**

**Prepared by**

**Team Number: 05**

| | |
|---|---|
| **Lenoah Chacko** | **B190657CS** |
| **Pavithra Rajan** | **B190632CS** |
| **Abin Gigo Joseph** | **B190880CS** |
| **Joseph Mani Jacob Mani** | **B190529CS** |
| **Akshay Kuttikkattuparambil Biju** | **B190803CS** |
| **Varun Chittezhath Anilkumar** | **B190621CS** |

**Project Owner:** **Ms. Lekshmy P Chandran**

**Course:** **CS4096D Software Engineering Laboratory**

**Date:** **05/02/2022**

**This template is based on the one available from GMU site by Dr. Rob Pettit. Modifications specific to NITC are made and will be used for academic purpose only.**

# Contents

# Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Draft Type and Number | Names of all team members | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 00/00/00 |

# 1 Introduction

## 1.1 Document Purpose

The purpose of this document is to provide a detailed description of the requirements for the swimiNIT mobile application. The application aims to help the user keep track of the number of visits of each swimmer that enters the pool. It also informs the user whether the swimmer's five free trials are over and gives a live update on how long the swimmer has been in the pool. The document contains a general description, interface, the performance and functional requirements of the application, a list of other relevant application attributes, application usage scenarios, and use case diagrams.

## 1.2 Product Scope

The aim of the system is to streamline the process of registration and regulate the entry of swimmers into the NITC swimming pool. Additionally, this system aims to facilitate the administrator with the retrieval of the swimmer's visit history and payment information. A mailing feature will notify the swimmers after each visit, the start and end of each quarter and pool maintenance. We expect this application to be actively integrated into the day-to-day working of the NITC swimming pool to automate record-keeping and ensure a smooth and efficient operation.

## 1.3 Intended Audience and Document Overview

This document is intended for the admins who will manage the swimming pool registration and entry system. This includes the staff member who regulates the entry into and out of the pool.

The next chapter, the overall description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, specific requirements section, of this document is written primarily for the developers and describes the details of the functionality of the product in technical terms.
Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different languages.

## 1.4 Definitions, Acronyms and Abbreviations

| S.L No. | Abbreviation/Acronym | Definition |
|---|---|---|
| 1. | SPO | Swimming Pool Officer |
| 2. | QR | Quick Response |

## 1.5 Document Conventions

This document follows the IEEE formatting requirements.

## 1.6 References and Acknowledgments

IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998

# 2 Overall Description

## 2.1 Product Overview

The primary objective of swimiNIT is to seamlessly regulate the entry of swimmers into the NITC swimming pool. As of now, the visitors to the swimming pool are tracked using a log book which is tedious. This system is a self-contained product. With the provision of five free trials for the student swimmers in each quarter, it has become crucial to track the visits. The retrieval of visit history and payment information is necessary.

Additionally, the application aims to implement a mailing feature to inform the swimmers of each visit, begin and end of each quarter, remaining trials in the case of student swimmers and relay information regarding pool maintenance. Subsequently, tracking the duration of each swimmer in the pool and informing them once the permissible time has been reached eases out the duties of the SPOs.

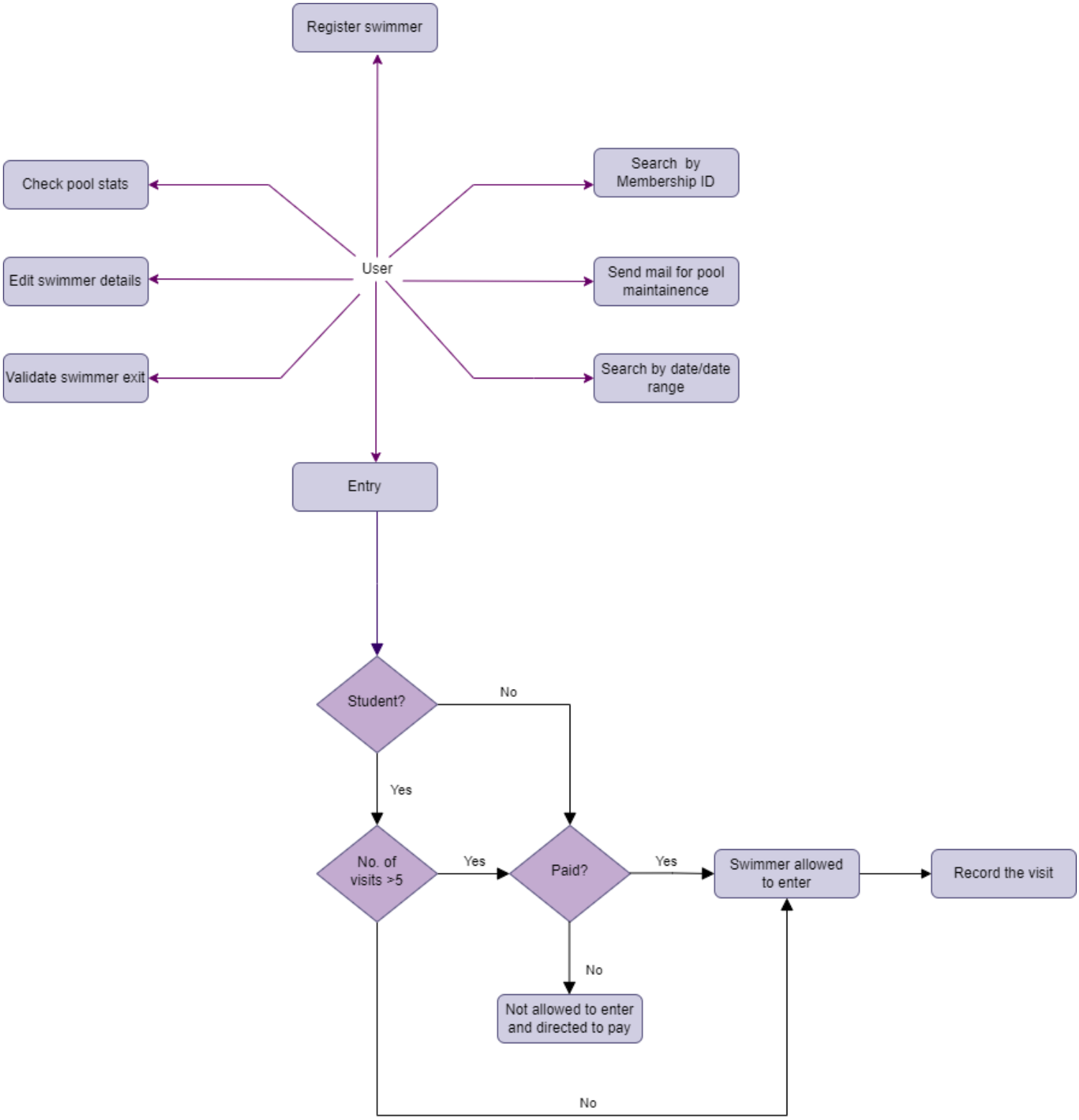



*Fig. 1. High-level description of the product*

## 2.2 Product Functionality

The swimiNIT App allows the user to:

- Mark the entry/exit of a swimmer into the pool.
- View the number of swimmers currently in the pool and the amount of time they've spent there.
- View the history of a day (number of swimmers and time) in the swimming pool.
- View the history (number of visits and date) of a swimmer in the pool.
- Mark a student swimmer as "paid" once they have used up their five free sessions and payment has been made. Thereby permitting them to continue using the pool.
- Send a mail to all swimmers regarding the status of the pool.
- Send a mail to a swimmer if their free visits have been exhausted.

## 2.3 Design and Implementation Constraints

**Database:** Cloud Firebase will be utilised to store the data. Assuming that each quarter only about 512 members would register with each record (copy of the receipt and relevant attributes) being 2MB, the free-tier of Cloud Firebase can be availed. Additionally, only 20,000 reads and writes are possible per day, which would fall within requirements. If it goes past these specifications, the next tier has to be used with an additional cost of $0.108 (approximately Rs 8.1).

**Hardware constraints**: A smartphone with Android OS version 6 or above. Minimum 1 GB RAM to ensure seamless run of the application. The smartphone should have a camera with 3 MP or above to capture and upload the copy of the receipt during swimmer registration.

**Internet:** The smartphone must be connected to the internet with adequate bandwidth to connect to the Cloud database and upload the receipt as well.

**Design conventions and standards:** This application development process will follow the principles specified by the Software Development Life Cycle (SDLC).

**Development Environment:** The development environment used for the application will be Android Studio/VS Code.

**Programming Standards:** Effective Dart programming styles as per official documentation will be adhered to while developing the application. Consistent naming, ordering and formatting will ensure that the application is maintainable.

## 2.4 Assumptions and Dependencies

- We assume that there will be three admin accounts that can be used by the swimming pool office staff. In the interest of security, new admin users must be manually added to the database.
- The application will be provided directly to the employees of the SPO and will not be available for download on any public platform.

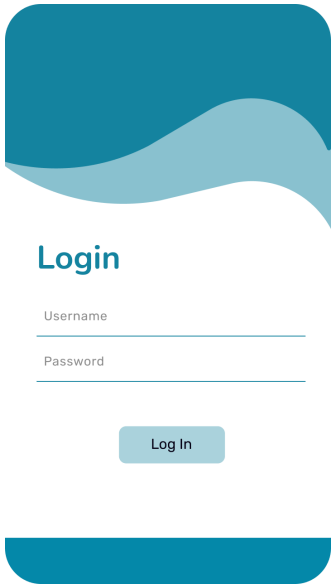
- The admin will keep an eye on the dashboard of the application and inform the swimmers who checked into the pool one hour ago to leave in a timely manner.

# 3 Specific Requirements

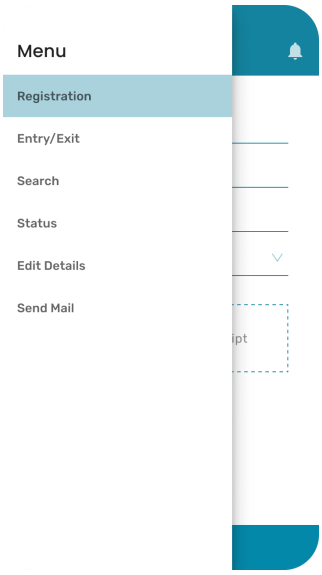
## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

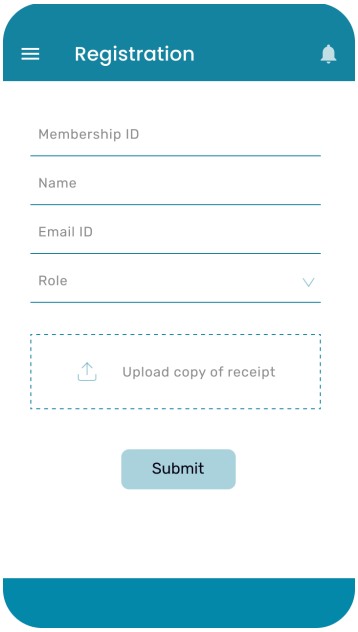
- **Login Page**
  - In this page, the user will enter the username and password and presses the ‘Log In’ button to log in to the system.




- **Navigation Menu**
  - Navigation Menu shows four options: Registration, Entry/Exit, Search and Status.



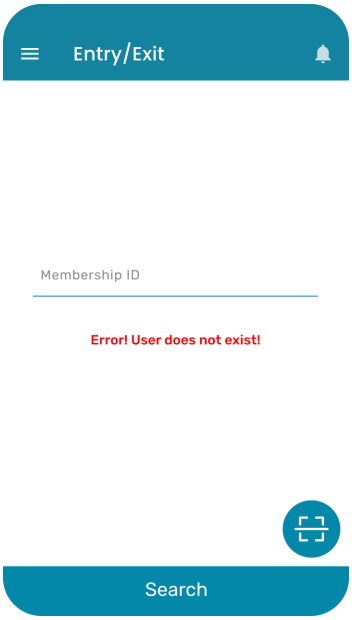

- **Registration Page**
  - The swimmers are registered in the app by the user. In this page, the user will enter the membership id, name, email id, role and upload a copy of the payment receipt which is applicable only to the faculty/staff and their dependents as a student is not required to pay for their first five sessions. The swimmer will be registered once the ‘Submit’ button is pressed.



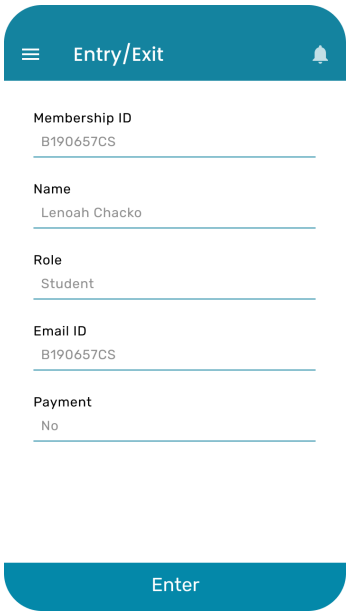

- **Entry/Exit Page**
  - This page is used to check whether the swimmer exists in the database or not by the user. The user enters the membership id of the swimmer on entry or exit and presses the ‘Search’ button.


- **Membership ID QR Code Scanner**
  - Alternatively, the user can scan the QR code on the swimmer's membership card instead of typing in the membership id. Pressing the 'Scan' button will scan the QR code and search for the swimmer.

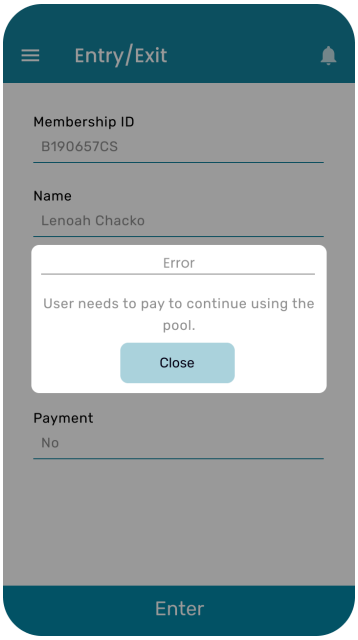

- **Entry/Exit Page for Invalid User**
  - This error shows up if a particular swimmer does not exist in the database.



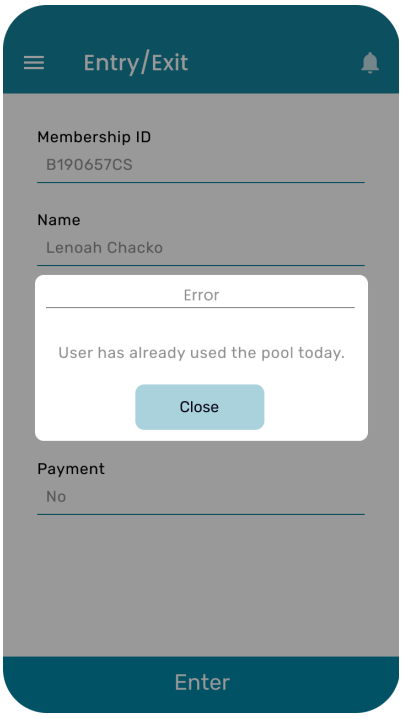

- **Entry Confirmation Page**
  - This page confirms the entry of a swimmer into the swimming pool. Pressing the ‘Enter’ button by the user will record the swimmer’s entry into the pool.



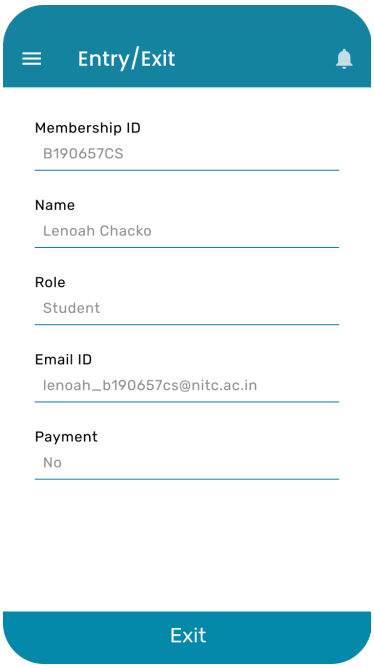

- **Error : If the swimmer has exhausted their five free sessions**
  - This error shows up if the swimmer has used up his/her five free sessions and has not paid yet.



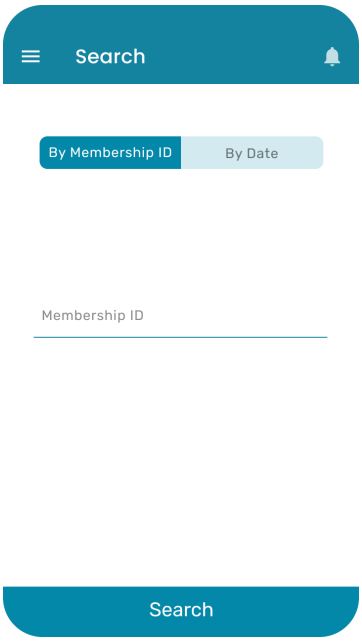

- **Error : If the swimmer had already used the pool that day**
  - This error shows up if the swimmer has already used the pool on that same day.




- **Exit Confirmation Page**
  - This page confirms the exit of a swimmer from the swimming pool. Pressing the ‘Exit’ button by the user will record the swimmer’s exit from the pool.

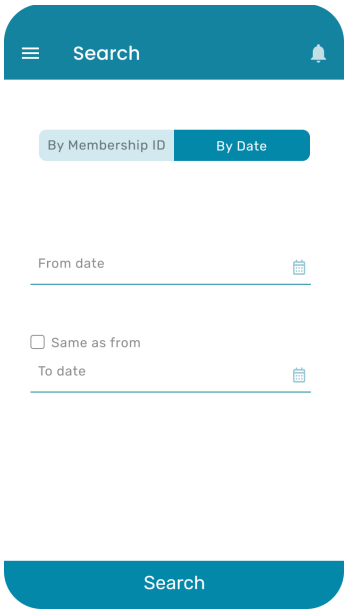



- **Search Page : “By Membership” active**
  - This page is used to search the history of a specific swimmer’s visits. The user will enter the membership id of the swimmer and pressing the ‘Search’ button will show the detailed history of the swimmer’s visits.

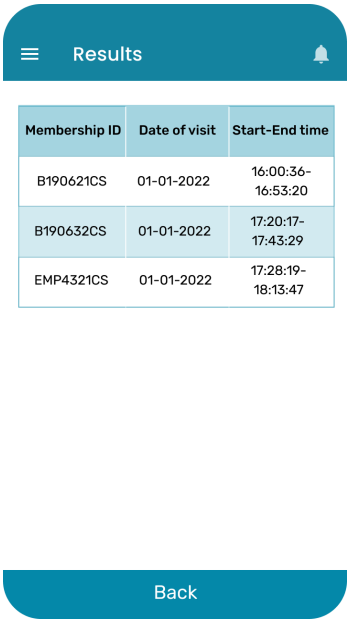



  - **History of a specific swimmer**
    - This page shows the detailed history of the searched swimmer’s visits. The details of the swimmer’s visits shown are Date of visit, Time of entry and Time of exit. Pressing the ‘Back’ button will take the user back to the Search Page.


- **Search Page : “By Date” active**
  - This page is used to search the history of the visits on a particular day or a date range. The user will enter the ‘From date’ and select ‘Same as from’ to get the history of a particular day. To get the history of a particular date range, the user will have to enter the ‘To date’ as well. Pressing the ‘Search’ button will display the history.



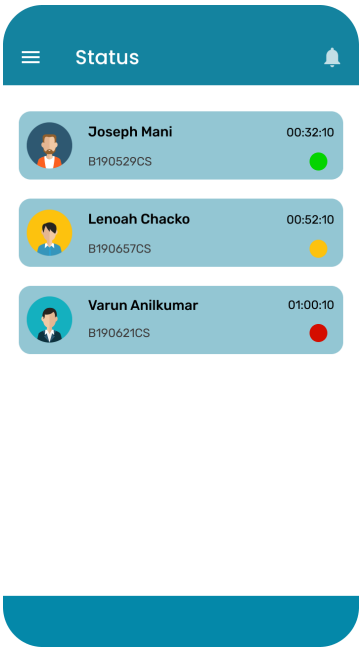

  - **History of a specific day**
    - This page displays the detailed history of the visits on a specific day. The details of the visits shown are Membership ID, Date of visit and Start-End time. Pressing the ‘Back’ button will take the user back to the Search Page.



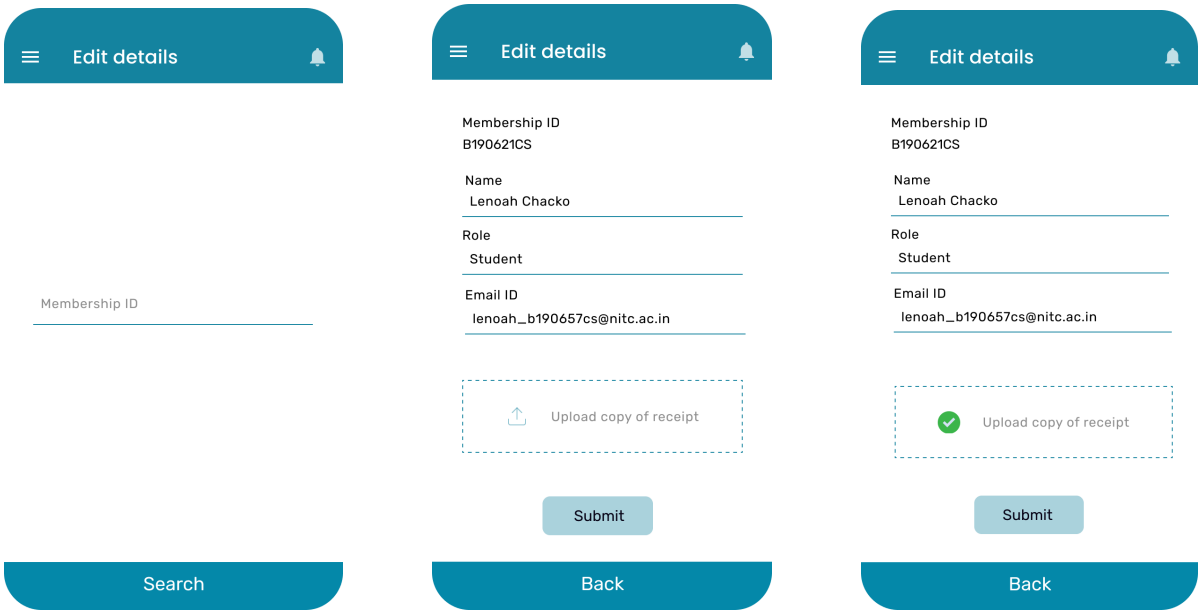

- **History of a specific date range**
  - This page displays the detailed history of the visits on a specific date range. The details of the visits shown are Membership ID, Date of visit and Start-End time. Pressing the 'Back' button will take the user back to the Search Page.


- **Swimmer Status Page**
  - This page displays the status of all the swimmers currently in the pool. The details shown are Name, Membership ID, the time remaining for a swimmer (duration: 1 hour) and a visual indicator showing when it's time for a swimmer to leave.

- Initially, the indicator will be *green*. It will turn *yellow* when there are 10 minutes left for the swimmer's session to end and *red* when the session is over.




- **Swimmer Details Edit Page**
  - This page allows the user to edit the details of a particular swimmer. First, the admin has to search for the swimmer using their Membership ID. If the swimmer doesn't exist, it will be displayed on the screen. Else, the details of the swimmer will be made available for editing.



### 3.1.2 Hardware Interfaces

There must be a smartphone with a minimum of 1 GB RAM to ensure a seamless run of the application. The smartphone should have a camera with 3 MP or above to capture and upload the copy of the receipt during swimmer registration.

### 3.1.3 Software Interfaces

The smartphone's Operating system must be Android OS version 6 or above.

## 3.2 Functional Requirements

F1: The system shall allow the user to log in to the system after verifying the login credentials.

F2: The system shall allow the user to add the swimmer to the database after registration.

F3: The system shall allow the user to renew the membership of a swimmer.

F4: The system shall provide an interface to validate and record the swimmer's entry to the pool by verifying their registration, payment or number of free trials and if they had previously visited the pool on the same day.

F5: The system shall provide an interface to validate and record the swimmer's exit from the pool.

F6: The system shall allow the user to search for the list of swimmer visits to the pool on a specific day or a given date range.

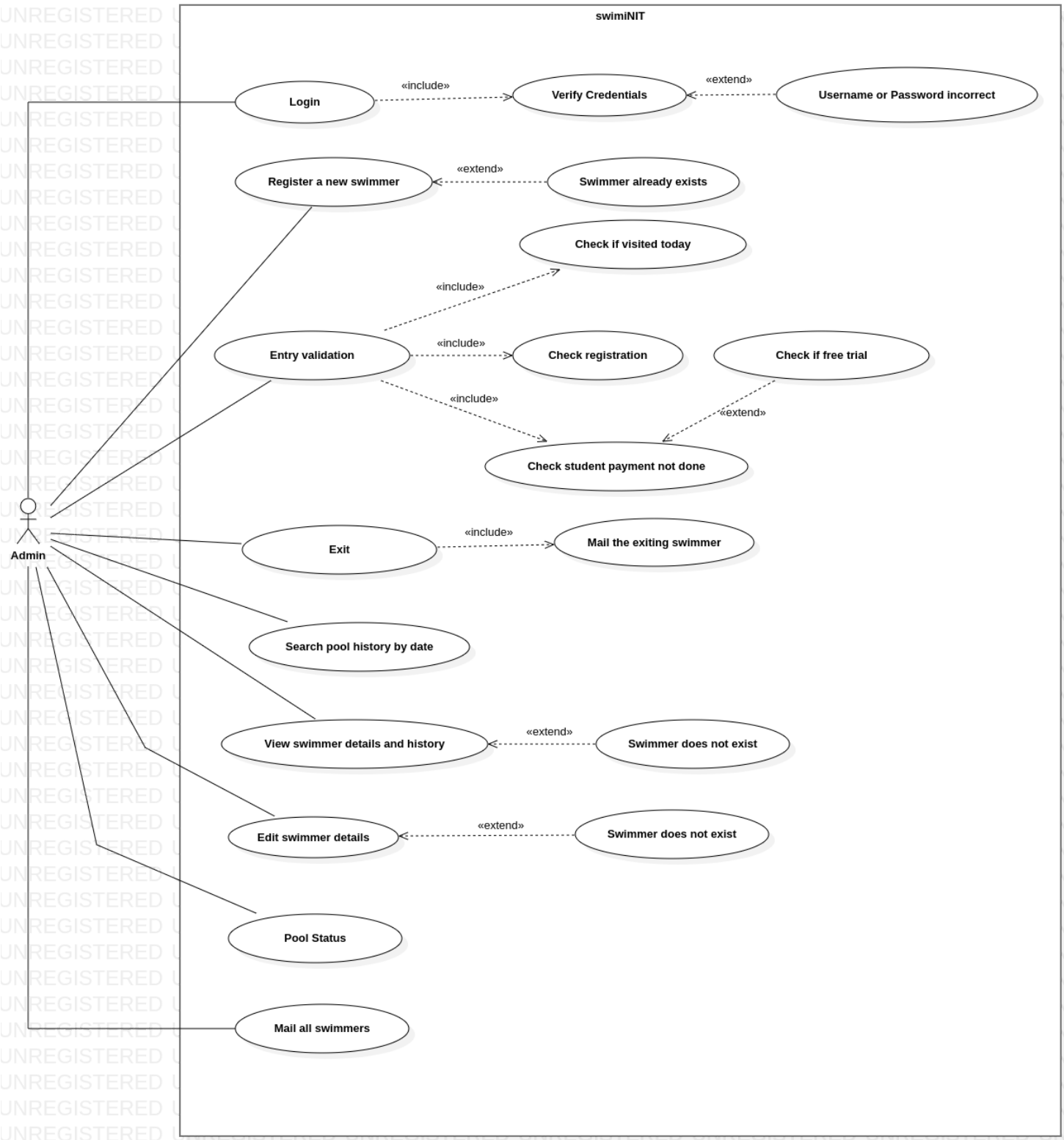
F7: The system shall allow the user to search for the details of a swimmer by their membership ID.

F8: The system shall provide an interface that displays the status of all the swimmers currently in the pool with visual indicators and a timer indicating their time remaining in the pool.

F9: The system shall mail the swimmer updating their daily visits, quarter and payment reminders and other pool announcements.

F10: The system shall allow the user to edit the details of the swimmer after searching by their membership ID.

## 3.3 Use Case Model



### 3.3.1 Use Case #1 (Login – U1)

**Author –** Joseph Mani Jacob Mani

**Purpose –** To allow the existing user to be logged into the system using valid login credentials.

**Requirements Traceability –** F1

**Priority –** High

**Preconditions –** None

**Post conditions –** The user will be successfully logged into the system.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters the login credentials (username and password which are provided by the application developer) and clicks the 'Submit' button. | The system verifies the entered login credentials with the database and logs the user into the system. |

- **Alternative Flow:** None
- **Exceptions:**

| Actor's Action | System's Response |
|---|---|
| The user enters the login credentials and clicks the 'Submit' button. | The system displays an error message stating that the entered username or the password is wrong. |

**Includes –** None

**Notes/Issues –** None

### 3.3.2 Use Case #2 (Register a new swimmer – U2)

**Author –** Akshay Kuttikkattuparambil Biju

**Purpose –** To register a new swimmer into the system.

**Requirements Traceability –** F2

**Priority –** High

**Preconditions –** The admin should be logged into the system.

**Post conditions –** The swimmer will be successfully registered into the system.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters the details of the swimmer and clicks the submit button. | The system displays a message stating the swimmer has successfully been registered into the system. |

- **Alternative Flow (*renew membership*):**

| Actor's Action | System's Response |
|---|---|
| The user scans the QR code shown to them by the returning swimmer. | The system prefills the swimmer details |

- **Exceptions:**

| Actor's Action | System's Response |
|---|---|
| The user enters the details of the swimmer and clicks the submit button. | The system displays an error message stating that the swimmer is already registered. |

**Includes –** None

**Notes/Issues –** None

### 3.3.3 Use Case #3 (Swimmer Entry – U3)

**Author –** Varun Chittezhath Anilkumar

**Purpose –** To provide an interface to validate and record the swimmer's entry into the pool

**Requirements Traceability –** F4

**Priority –** High

**Preconditions –** The admin should be logged into the system.

**Post conditions –** The swimmer's entry into the pool will be successfully recorded.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters the membership id of the swimmer and clicks the 'Search' button. | The system displays the details of the swimmer if the swimmer is registered in the system. |
| The user presses the 'Enter' button to record the entry of the swimmer into the pool. | The system records the swimmer's entry into the pool successfully if the swimmer has already paid or has free sessions left in their name. |

- **Alternative Flow:**

| Actor's Action | System's Response |
|---|---|
| The user presses the 'Scan' button to scan the QR code on the swimmer's membership card instead of typing in the membership id. | The system displays the details of the swimmer. |

- **Exceptions:**

| Actor's Action | System's Response |
|---|---|
| The user enters the membership id of the swimmer and clicks the 'Search' button. | The system displays an error message stating that the swimmer does not exist if the swimmer is not registered in the system. |
| The user presses the 'Enter' button to record the entry of the swimmer. | The system displays an error message if:<br><br>1. The swimmer has used up their five free sessions.<br><br>2. The swimmer has not paid yet.<br><br>3. The swimmer has already used the pool on that same day. |

**Includes –** None

**Notes/Issues –** None

### 3.3.4 Use Case #4 (Swimmer Exit – U4)

**Author –** Varun Chittezhath Anilkumar

**Purpose –** To provide an interface to validate and record the swimmer's exit from the pool

**Requirements Traceability –** F5

**Priority –** High

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered in the system.

**Post conditions –** The swimmer's exit from the pool will be successfully recorded.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user presses the 'Exit' button to record the exit of the swimmer from the pool. | The system records the swimmer's exit from the pool successfully. |

- **Alternative Flow:** None
- **Exceptions:** None

**Includes –** U7

**Notes/Issues –** None

### 3.3.5 Use Case #5 (Search history by date – U5)

**Author –** Pavithra Rajan

**Purpose –** To search for the list of swimmers who visited the pool on a specific date or date range

**Requirements Traceability –** F6

**Priority –** High

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered into the system and each visit must be recorded.

**Post conditions –** The swimmer details related to the particular search will be displayed.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters the 'from' and 'to' date. | The system displays the membership ID, date of visit and start-end time of all the |

| | |
|---|---|
| | swimmers that visited the pool in the given date range. |

- **Alternative Flow:**

| Actor's Action | System's Response |
|---|---|
| The user selects the 'same as from' option. | The system displays the membership ID, date of visit and start-end time of all the swimmers that visited the pool on that specific date. |

- **Exceptions:** None

**Includes** – None

**Notes/Issues** – None

### 3.3.6 Use Case #6 (View swimmer details and history– U6)

**Author –** Joseph Mani Jacob Mani

**Purpose –** To search for a swimmer's visit history using their membership ID

**Requirements Traceability –** F7

**Priority –** Medium

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered into the system and each visit must be recorded.

**Post conditions –** The detailed history of the swimmer's visits to the pool will be displayed.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters the 'Membership ID' of the swimmer to be searched. | The system displays the swimmer's details such as the name, role (student or faculty), date of visit, time of entry and exit. |

- **Alternative Flow:** None
- **Exceptions:**

| Actor's Action | System's Response |
|---|---|
| The user enters the 'Membership ID' of the swimmer to be searched. | The system displays an error message stating that the entered membership ID is invalid. |

**Includes –** None

**Notes/Issues –** None

### 3.3.7 Use Case #7 (Mailing system 1 – U7)

**Author –** Abin Gigo Joseph

**Purpose –** To send a mail to the swimmers when they exit the pool.

**Requirements Traceability –** F9

**Priority –** Medium

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should have entered the pool.

**Post conditions –** The swimmer will receive an email stating the total number of visits made in the quarter and how many visits are left in the free trial.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user removes a swimmer from the pool. | The system sends an email to the student's email ID stating that they have left the pool and the number of visits left in the free trial. |

- **Alternative Flow:** None
- **Exceptions:** None

**Includes –** None

**Notes/Issues –** None

### 3.3.8 Use Case #8 (Mailing system 2 – U8)

**Author –** Abin Gigo Joseph

**Purpose –** To send emails regarding pool updates.

**Requirements Traceability –** F9

**Priority –** Medium

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered into the system.

**Post conditions –** The swimmer will receive an email regarding the pool update.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|

| | |
|---|---|
| The user types out the email content and clicks send. | The system sends an email to all the student's email IDs with the given content. |

- **Alternative Flow:** None
- **Exceptions:** None

**Includes –** None

**Notes/Issues –** None

### 3.3.9 Use Case #9 (Edit swimmer details– U9)

**Author –** Akshay Kuttikkattuparambil Biju

**Purpose –** To edit swimmer details.

**Requirements Traceability –** F10

**Priority –** Medium

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered into the system.

**Post conditions –** The swimmers details will have changed.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user enters new details and clicks submit. | The system updates the swimmer details. |

- **Alternative Flow:** None

- **Exceptions:** None

**Includes –** None

**Notes/Issues –** None

### 3.3.10 Use Case #10 (See active swimmers – U10)

**Author –** Lenoah Chacko

**Purpose –** To view the swimmers who are currently in the pool and how much time has elapsed since they have checked in.

**Requirements Traceability –** F8

**Priority –** High

**Preconditions –**

- The admin should be logged into the system.
- The swimmer should be registered into the system.

**Post conditions –** The admin has seen the swimmers in the pool and how much time they have left. They can take actions accordingly.

**Actors –** Admin

**Extends –** None

**Flow of Events**

- **Basic Flow:**

| Actor's Action | System's Response |
|---|---|
| The user opens the dashboard interface. | The system displays all the swimmers currently in the pool and how much time they have left.<br><br>● All students that have over 10 minutes left will have a green indicator.<br>● All students that have less than 10 minutes and still haven't exhausted their 60 minutes session will have a yellow indicator. |

| | |
|---|---|
| | • All students that exceed the 1-hour limit will have a yellow indicator. |

- **Alternative Flow:**

| Actor's Action | System's Response |
|---|---|
| The user opens the dashboard interface. | The system displays that there are no swimmers in the pool if the pool is empty or closed. |

- **Exceptions:** None

**Includes –** None

**Notes/Issues –** None

# 4 Other Non-functional Requirements

## 4.1 Performance Requirements

- **System:** The application will run on all devices running Android 6 (Marshmallow) or later. The application will be responsive.
- **Response Time:** The application should load and be usable within 3 seconds. It should update the interface on interaction within 2 seconds.

## 4.2 Safety and Security Requirements

The application will be provided directly to the employees of the SPO and will not be available for download in any public platform. Hence, there will be no vulnerabilities.

## 4.3 Software Quality Attributes

### 4.3.1 Reliability

The application requires Internet connectivity. For fetching any data, connectivity is indispensable to the functioning of this application. With the assumption that there is an uninterrupted connection, this application is reliable.

### 4.3.2 Correctness

The application will be correct in terms of adhering to the functional requirements and will exhibit all the necessary functionalities.

### 4.3.3 Availability

The application will run seamlessly. Since the data is stored on the cloud, any failure in the device used by the user will not impact the data.

### 4.3.4 Scalability

Though the free-tier does limit the data records to about 512 members, a nominal payment can scale the application to handle more records.

### 4.3.5 Maintainability

The development team will follow the best practices and adhere to the officially documented effective Dart techniques. In addition, software modularity will be followed throughout, making the application maintainable. An extensive maintenance document will be provided for future reference.

### 4.3.6 Portability

The application will be made using Flutter and will run on devices running Android 6.0 or higher.

### 4.3.7 Extensibility

The code for the application will be available, and extensions can easily be created as and when required.

### 4.3.8 Reusability

The components of this application are very simple and will likely not be reused.

### 4.3.9 Usability

The application will have a UI that will be simple to use and explain. New users will be able to use it with ease.

### 4.3.10 Robustness

The application will have robust features correctly implementing the above use cases and will handle exceptions when required.

# 5 Other Requirements

There is no other requirement needed for this project.

# Appendix A - Activity Log

| S. No. | Date | Time |
|---|---|---|
| 1. | 01-02-2022 | 04:00 to 05:20 PM |
| 2. | 02-02-2022 | 09:30 to 11:30 PM |
| 3. | 02-02-2022 | 12:15 to 02:05 AM |
| 4. | 03-02-2022 | 12:00 to 2:30 AM |
| 5. | 04-02-2022 | 12:00 to 01:10 AM |

| S. No. | Team Member | Contribution |
|---|---|---|
| 1. | Lenoah Chacko | Intended Audience and Document Overview, Assumptions, Performance Requirements, Use Case #10 |
| 2. | Pavithra Rajan | Product Scope, Design and Implementation constraints, Software Quality Attributes 4.3.1-4.3.5, Flow Chart, Use Case #5 |
| 3. | Abin Gigo Joseph | Product Functionality, Software Quality Attributes 4.3.6-4.3.10, Use Case #7, #8 |
| 4. | Joseph Mani Jacob Mani | Product Overview, Functional Requirements, Safety and Security Requirements, Use Case #1, #6 |
| 5. | Akshay Kuttikkattuparambil Biju | Document Scope, Functional Requirements, Use Case #2, #9 |
| 6. | Varun Chittezhath Anilkumar | User Interfaces, Hardware and Software Interfaces, Use Case #3, #4 |