



---

# Software Requirements Specification

for

## A5-Swimming Pool

Version 1.0

Prepared by

Team Number: 06

HRIDIKA KV  
APARNA V  
JAGRITI SHARMA  
POOJA S NAIR  
SRUSHTI WAGHODE

B190464CS  
B190364CS  
B190864CS  
B190428CS  
B190580CS

**Project Owner:** Ms.SREEKALA

**Course:** CS4096D Software Engineering Laboratory

**Date:** 5th February 2022

This template is based on the one available from the GMU site by Dr. Rob Pettit.

Modifications specific to NITC are made and will be used for academic purposes only.

<b>1 Introduction</b>	<b>4</b>
1.1 Document Purpose	4
1.2 Product Scope	4
1.3 Intended Audience and Document Overview	5
1.4 Definitions, Acronyms and Abbreviations	5
1.5 Document Conventions	6
1.6 References and Acknowledgments	6
<b>2 Overall Description</b>	<b>6</b>
2.1 Product Overview	6
2.2 Product Functionality	8
2.3 Design and Implementation Constraints	8
2.4 Assumptions and Dependencies	8
<b>3 Specific Requirements</b>	<b>8</b>
3.1 External Interface Requirements	8
3.1.1 User Interfaces	8
3.1.2 Hardware Interfaces	13
3.1.3 Software Interfaces	13
3.2 Functional Requirements	14
3.3 Use Case Model	15
3.3.1 Use Case #1 (Login-U1)	16
3.3.2 Use Case #2 (Invalid login– U2)	16
3.3.3 Use Case #3 (Add user – U3)	17
3.3.4 Use Case #4 (Get user details – U4)	18
3.3.5 Use Case #5 (Search membership id – U5)	20
3.3.6 Use Case #6 (Add visit-U6)	21
3.3.7 Use Case #7 (Add payment– U7)	23
3.3.8 Use Case #8 (Add amount– U8)	24
3.3.9 Use Case #9 (Add date of payment – U9)	25
3.3.10 Use Case #10 (Delete user-U10)	26
3.3.11 Use Case #11 (Get visit details – U11)	27
3.3.12 Use Case #12 (Get payment details - U12)	29
3.3.13 Use Case #13(Get information about today’s visit – U13)	29
3.3.14 Use Case #14 (Get users with dues – U14)	30
3.3.15 Use Case #15(Get quarterly report – U15)	32
3.3.16 Use Case #16 (Sort by category – U16)	32

3.3.16 Use Case #17 (Logout - U17)	33
<b>4 Other Non-functional Requirements</b>	<b>34</b>
4.1 Performance Requirements	34
4.2 Safety and Security Requirements	34
4.3 Software Quality Attributes	35
<b>5 Other Requirements</b>	<b>36</b>
<b>Appendix A - Activity Log</b>	<b>36</b>

## Revisions

Version	Primary Author(s)	Description of Version	Date Completed
SRS 1.0	APARNA V HRIDIKA KV JAGRITI SHARMA POOJA S NAIR SRUSHTI WAGHODE	Initial release	05/02/22

---

# 1 Introduction

The NITC swimming pool management application project mainly aims to help the office staff to keep track of the number of visits and quarterly dues of the visitors. Currently, the swimming pool management is done manually. Person appointed at the swimming pool have to manually mark the visits of and payment done by users. Since thousands of users use the swimming pool, it will be difficult to handle the registers. To get the users with dues, the office staff will have to manually check the entire list which is time consuming and exhausting. With developments in technology, an automated swimming pool management system will help to reduce the penned paperwork and to mark the visits and get payment details within seconds. To make the swimming pool management easier, automation with the help of android application is needed.

On automation of swimming pool management by our android application, marking of visits made by the users and addition of their payment details can be done easily. All the details are stored in a DB and thus there is better organisation and less documentation. Since we make use of the search option to get details of users, the user details can be easily accessed by the office staff. This saves time and effort. In this system, the students will get notified by the app after their free visits are over. This ensures that all the students are informed about their dues. With the android application, the management becomes easier and more efficient.

This section explains the scope of the product, its objectives, benefits, intended audience and also gives an overview of what this SRS document contains. It further gives an idea of document conventions, reference and acknowledgement and definitions, acronyms and abbreviations used.

## 1.1 Document Purpose

The purpose of this SRS document is to give a detailed overview of the android application for NITC swimming pool management version 1.0. This document contains both functional and non-functional requirements of the system. It will explain the purpose and features of the system, product scope, use case diagrams, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

## 1.2 Product Scope

The swimming pool application is intended to have a better organisation and access to the details of the users of the swimming pool. This will help the office staff to view the details of the user before permitting them to use the pool.

### OBJECTIVES :

- Allows office staff to add a new swimming pool user to the system
- Allows office staff to delete existing inactive pool users.
- Allows office staff to get details of the swimming pool users searching by their unique membership id.
- Allows office staff to mark the visit of pool users.
- Allows the staff to get reports of a quarter.

- 
- Allows office staff to add payment details of a pool user.
  - Obtain the list of pool users whose fees are overdue.
  - Allows office staff to get information about today's visitor details.
  - Notifies the visitors if there are any pending dues by email.

#### BENEFITS :

- This application will reduce the penned paperwork that is required for storing all information related to visit and payment.
- Provide easy and faster access to information to the user.
- Provide a user-friendly environment.
- Improve efficiency.

### **1.3 Intended Audience and Document Overview**

The intended audience or readers of this document are the developers of the application, testers and users.

The purpose of this document is to describe the functionality and specifications of the design of an application for managing swimming pool visitors.

The first section gives an overview of product scope and objectives of the project.

Next section, the overall description section, will give a complete overview of product functionality and implementation.

The third section, specific requirement section, will provide a more in-depth and comprehensive description of functionality of the product in technical terms. It contains details about the exact specifications required for each of the application's components and their interface is explained separately along with behavioral and functional requirements.

Fourth section describes the other non-functional requirements of the system.

### **1.4 Definitions, Acronyms and Abbreviations**

Sl.no	Abbreviation/Acronym	Expansion
1	App	Application
2.	Admin	Administrator
3..	Db	Database
4.	SDLC	Software Development Life Cycle
5.	SRS	Software Requirements Specification

---

## DEFINITION

Words	Meaning
Visitor/User	One who visits/uses the swimming pool. Includes students, faculty, family members of faculty
Admin	The person in charge of managing the swimming pool system

## 1.5 Document Conventions

This document follows the IEEE formatting requirements. Arial font size 11, or 12 throughout the document for text has been used. For headings font size of 14 or higher is used. Italics have been used for comments. The document text is single-spaced and maintains the 1" margins.

## 1.6 References and Acknowledgments

1. IEEE format
2. <https://staruml.io/>
3. <https://app.diagrams.net/>
4. Object oriented software engineering using UML, patterns, and Java

# 2 Overall Description

## 2.1 Product Overview

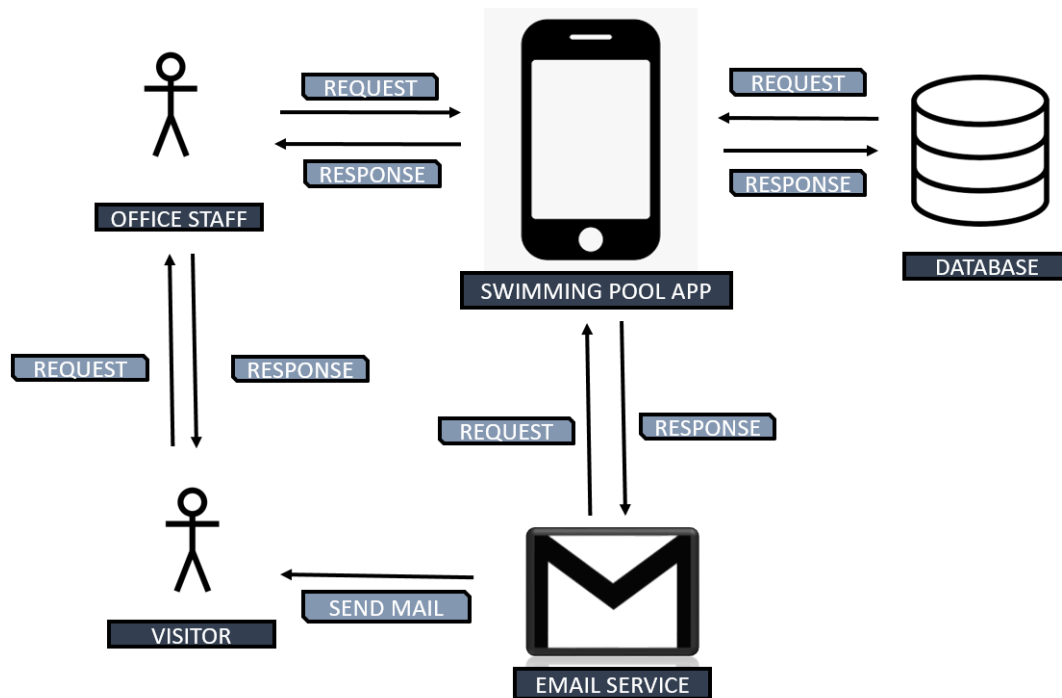
In the existing system, people appointed at the swimming pool have to manually mark down the number of visits made by the user and the track of dues and other details are also done in a manual way. Processes like getting the list of users with dues and informing users about their dues is also time consuming and cumbersome as the office staff need to go through the whole list of users for getting this information. As there are a lot of users of the swimming pool, maintaining the record of every user and keeping track of their visits is troublesome. Making entries of users manually can lead to errors too. Automation of the swimming pool management system through an android app, the nitc swimming pool app, will reduce these problems.

This application is a new, self-contained product. It will automate the swimming pool management system to reduce paper work and to make the management of users easy. All the information of users of the swimming pool will be stored in the db for better organization of data. This app will eliminate the need of manually marking visit details of visitors on paper. Person appointed at the swimming pool can mark the details of the visitor and he can also add and update payment details of the visitor using this app. If the visitor is a student, then there is a provision that he/she will get

five free trials per quarter. This system will notify students by sending a mail when their five free trials get over and if they haven't already paid fees for a particular quarter. Other categories of users will also be notified by sending mails about their dues. Once the payment status of a particular user gets updated in the app, the user will receive a confirmation mail stating that their due for this quarter is cleared.

#### Features:

- Allows office staff to mark the visit and payment details of visitors.
- Allows office staff to retrieve visit details as well as payment details of visitors.
- Allows addition of new visitors to the system as well as deletion of existing visitors.
- Inform the students, when their free trials are over, through email.
- Inform the visitors about the dues through email
- This application can generate the following types of reports-
  - ❖ Report about everyday visits - This report will contain statistics about the visits on a particular day.
  - ❖ Report containing users with dues - This report will contain the details of all the users which have pending dues for the quarter.
  - ❖ Quarterly Report - This report will contain all the information of all the users for a particular quarter. Information such as user details, number of visits made by the user in that quarter, payment status, dues etc will be there in this report. This report can also be sorted by category of the users. For example, the admin can specifically get the quarterly report for all the students who are the users of the swimming pool.



---

## 2.2 Product Functionality

### Admin Activity-

- Admins can log in to the system.
- Admin can add a new user to the system.
- Admin can delete an existing user from the system.
- Admin can get details of existing users by their unique membership id.
- Admin can increment the count of entries made by a visitor.
- Admin can mark if the visitor has paid dues for a quarter and he can add the date on which the dues were paid.
- Admin can view the report containing information about everyday visits.
- Admin can view the report which will include the details of users who haven't paid their dues for the quarter.
- Admin can get the quarter reports which will contain the list of all the users and with the number of visits for that quarter.
- Admin can logout from the system.

## 2.3 Design and Implementation Constraints

- The project will follow SDLC.
- The application will be using Android Studio for development .
- Third-party applications may be used to send emails to the visitors.

## 2.4 Assumptions and Dependencies

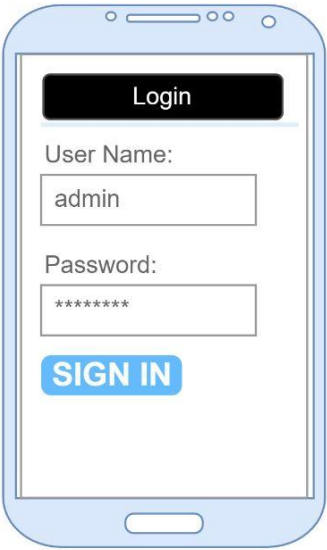
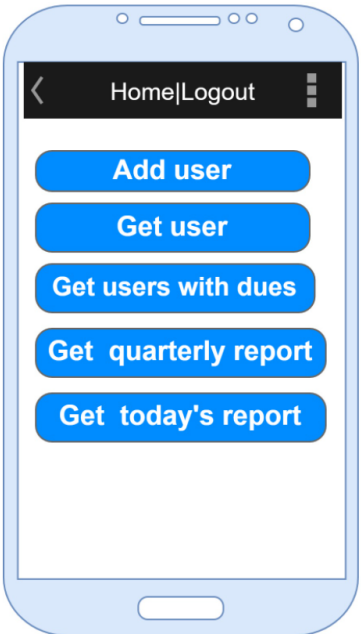
1. The system is designed specifically for the use of the NITC swimming pool.
2. The users of swimming pool will be given a unique membership id which is:
  - The roll no for the student.
  - The employment id for the faculty.
  - A unique id generated at the time of application for the family members of faculty.
3. Login credentials will be provided to the admins.
4. Third party email services will be used to send notification emails to the user of swimming pools.

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces



USER INTERFACE	DESCRIPTION
 <p>The mockup shows a mobile app interface for login. At the top is a black bar with the word 'Login' in white. Below it are two input fields: 'User Name:' with the text 'admin' and 'Password:' with masked characters '*****'. At the bottom is a blue button with the text 'SIGN IN' in white.</p>	<p>The user can login to the system by using his/her membership id as username and password.</p> <p>If the entered credentials are valid, the user can login to his/ her account.</p> <p>In case of invalid login, an error message will be displayed on the screen.</p>
 <p>The mockup shows a mobile app interface for the home screen. At the top is a black bar with a back arrow, the text 'Home Logout', and a menu icon. Below it are five blue buttons with white text: 'Add user', 'Get user', 'Get users with dues', 'Get quarterly report', and 'Get today's report'.</p>	<p>The user is redirected to the home page after a successful login.</p> <p>The home screen will contain the options 'Add user', 'Get user', 'Get users with dues' 'Get quarterly report' and 'Get today's report'.</p>

Name	Student1
Membership id	B210001CS
Contact no	99999999
Email	S1220@gmail.com
Category	Student

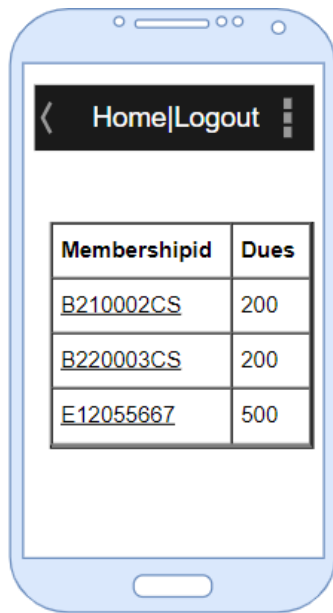
**Add user**

On clicking the 'Add user' button, the user will get fields to fill in the user details.

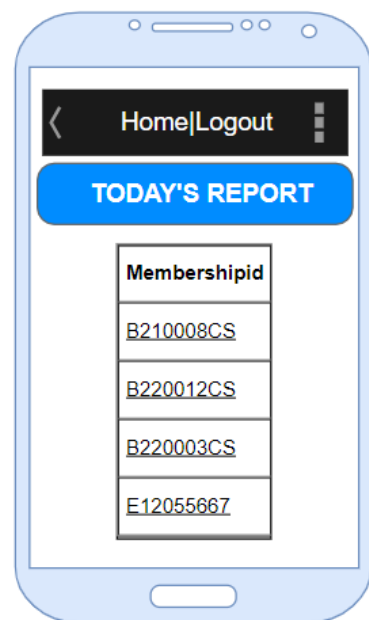
**QUARTERLY REPORT**

Membershipid	Visits
<a href="#">B210001CS</a>	6
<a href="#">B220002CS</a>	5
<a href="#">B220003CS</a>	5
<a href="#">E1205566Z</a>	3

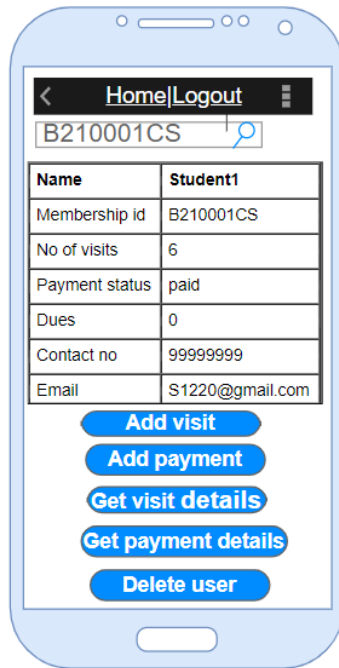
On clicking the 'Get quarterly report' button, a list will be displayed for all the members along with their number of visits to the swimming pool facility.



On clicking the 'Get users with dues' button, a list of all the members with pending dues will get displayed on the screen.

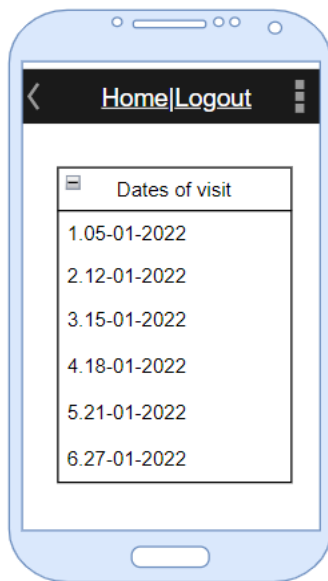


On clicking the 'Get today's report' button, the user can get a list of visitors who visited the facility on a particular day.

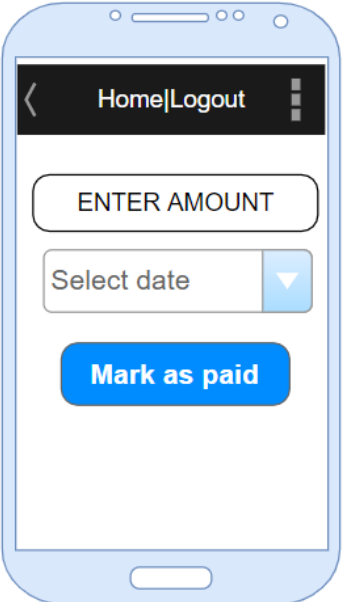
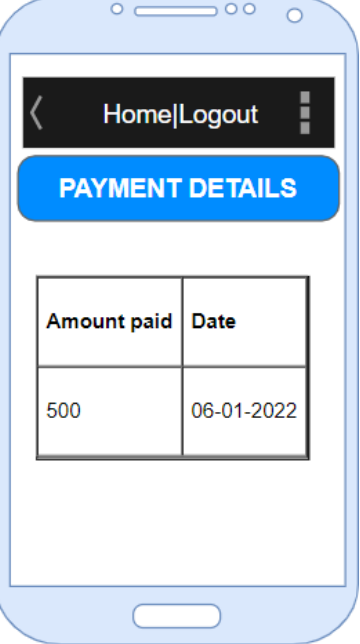


On clicking the 'Get user details' button, a search box is displayed on the screen where the admin can type the membership id to search for a member.

On clicking the 'add visit' option, the visit count for the member gets incremented and the date of visit is automatically added.



On clicking the 'Get visit details' button, the admin can get visit details for all visits made by the member in a quarter.

	<p>On clicking the 'Add payment' button, the user can add payment details such as date of visit and amount paid by the user.</p>				
 <table border="1" data-bbox="240 1140 511 1323"> <thead> <tr> <th>Amount paid</th><th>Date</th></tr> </thead> <tbody> <tr> <td>500</td><td>06-01-2022</td></tr> </tbody> </table>	Amount paid	Date	500	06-01-2022	<p>On clicking the 'Get payment details' button, the payment details are displayed on the screen.</p>
Amount paid	Date				
500	06-01-2022				

*The user interfaces may differ in terms of design during implementation, this is to show the basic features of the user interface*

### 3.1.2 Hardware Interfaces

The application is compatible with any Android/iOS device. No additional hardware needed.

### 3.1.3 Software Interfaces

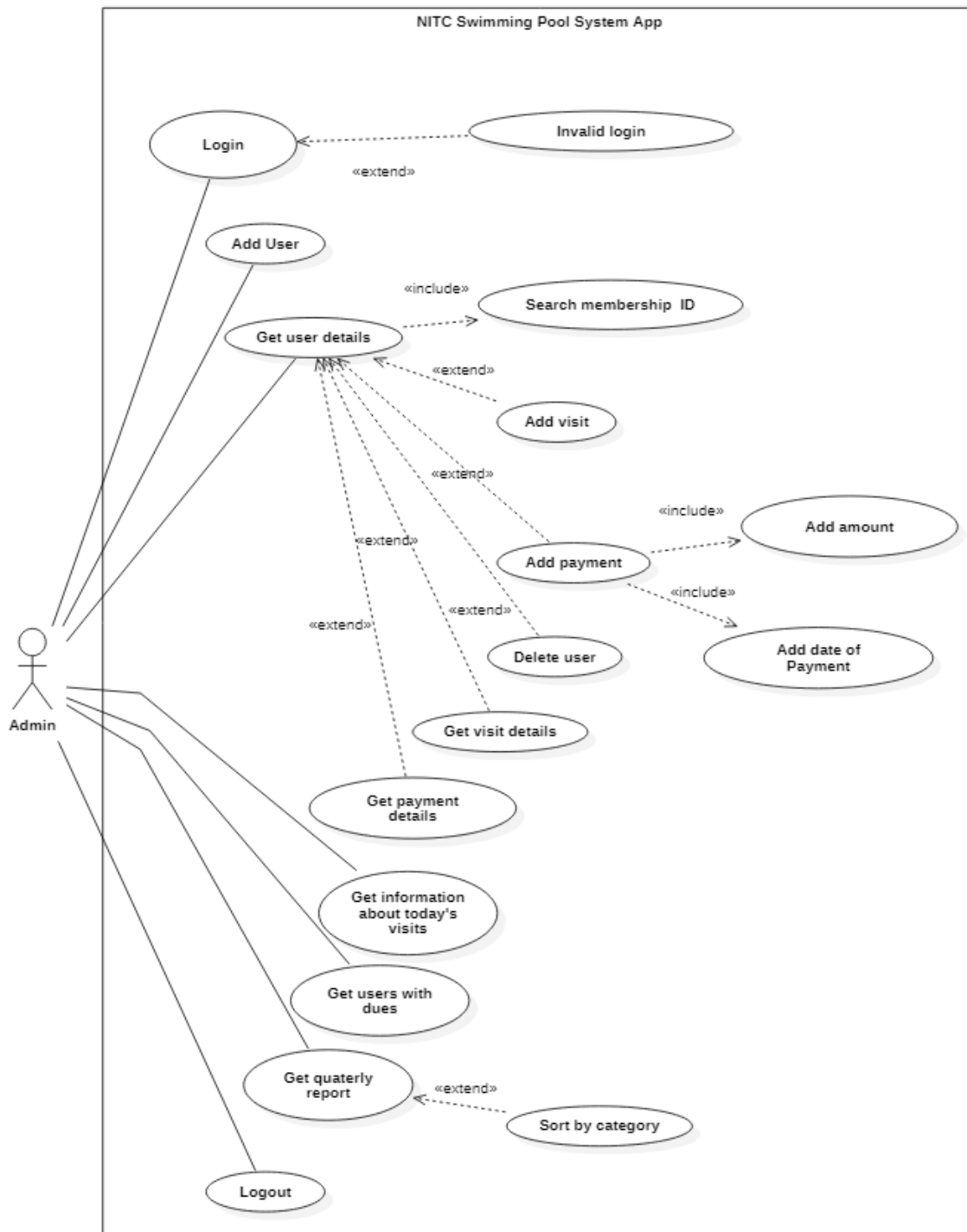
---

The application is a standalone app and no additional software interfaces are required.

## **3.2 Functional Requirements**

- F1: The system shall allow admin to login.
- F2: The system shall validate the login details.
- F3: The system shall allow admin to add users.
- F4: The system shall allow admin to get user details.
- F5: The system shall allow admin to add visit details of the visitor.
- F6: The system shall allow admin to add details of payment made by visitors.
- F7: The system shall allow admin to search by membership id.
- F8: The system shall allow admin to get payment details of a visitor.
- F9: The system shall allow admin to get visit details of a visitor.
- F10: The system shall allow admin to delete a user.
- F11: The system shall allow the admin to get a quarterly report.
- F12: The system shall allow the admin to add the amount paid by the visitor.
- F13: The system shall allow the admin to display the list of users with dues.
- F14: The system shall allow the admin to display information about the visits on that particular day.
- F15: The system shall allow admin to log out from the system.
- F16: The system shall allow the admin to add the date of payment.

### 3.3 Use Case Model



---

### 3.3.1 Use Case #1 (Login-U1)

**Author** – Aparna V

**Purpose** - User will be able to login to the system with username and password.

**Requirements Traceability** – F1

**Priority** - High

**Preconditions** - Admin should be given login credentials.

**Post conditions** - The admin has logged into the system.

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow

Actor's action	System responses
Enter username and password	
Click 'login' button	System will validate the login credentials and direct the user to the homepage.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** NIL

**Notes/Issues** - NIL

### 3.3.2 Use Case #2 (Invalid login– U2)



---

**Author** – Jagriti sharma

**Purpose** - To display login error if admin enters invalid login credentials.

**Requirements Traceability** – F2

**Priority** - High

**Preconditions** - Admin must enter login credentials.

**Post conditions** - Login error message will be displayed on the screen and the admin will be redirected to the login page.

**Actors** – Admin

**Extends** – U1

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Enter username and password	
Click the 'login' button.	System will validate the username and password. Admin will be redirected to the login page and the error message stating that incorrect login information is entered will appear on the screen

2. Alternative Flow - NA

3. Exceptions - NA

**Includes** – NIL

**Notes/Issues** - NIL

**3.3.3 Use Case #3 (Add user – U3)**

---

**Author** – Pooja S Nair

**Purpose** - Add a new swimming pool user using unique membership id

**Requirements Traceability** – F3

**Priority** - High

**Preconditions** - Admin must have logged in into the application.

**Post conditions** - Users are successfully added into the application.

**Actors** – Admin

**Extends** – NA

**Flow of Events**

1. Basic Flow

Actor's actions	System responses
Click 'add user' button	"Add user" page will be displayed.
Unique membership id and other user details are filled and then 'submit' button is clicked	Message "Successfully added" is displayed and admin will be redirected to the home page

2. Alternative Flow -NA

3. Exceptions -NA

**Includes** - NIL

**Notes/Issues** -NIL

### 3.3.4 Use Case #4 (Get user details – U4)

---

**Author** – Srushti Waghode

**Purpose** - Admin will be able to get all the basic details of any member using their unique membership id to search.

**Requirements Traceability** – F4, F7

**Priority** - High

**Preconditions** - Admin should have logged into the system.

**Post conditions** - The member details should get displayed on the screen.

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
The admin clicks on the 'get user details' button	A new page with a search box appears on the screen.
The admin types the membership id of the member in the search box and clicks on the search button.	The member details get displayed on the screen.

2. Alternative Flow - NIL

3. Exceptions -

Actor's actions	System responses
The admin types an invalid membership id in the search box and clicks on the search button.	An error message is displayed on the screen.

---

**Includes** U5 - Search membership id

**Notes/Issues** - NIL

### **3.3.5 Use Case #5 (Search membership id – U5)**

**Author** – Aparna V

**Purpose** - This use case allows admin to search the visitor by membership id.

**Requirements Traceability** – F7

**Priority** - High

**Preconditions** - The admin should logged into the system and enter the membership id in the search bar and then the search button should be clicked.

**Post conditions** - On entering membership id in the search bar, if the membership id exists, the user can continue using the app. Otherwise a message ‘membership id not found’ is displayed.

**Actors** – Admin

**Extends** – NIL

#### **Flow of Events**

##### **1. Basic Flow -**

<b>Actor’s actions</b>	<b>System responses</b>
Admin enters membership id in the search bar	
Click ‘search’ button	Details related to that visitor and various other options are displayed on the screen

##### **2. Alternative Flow - NIL**

---

### 3. Exceptions -

Actor's actions	System responses
Admin enters membership id in the search bar	
Click search button	If the membership id doesn't exist a message 'membership id not found' is displayed on the screen.

**Includes** NIL

**Notes/Issues** - NIL

#### 3.3.6 Use Case #6 (Add visit-U6)

**Author** – Hridika K V

**Purpose** - This use case allows admin to enter visit details of a particular user .This can be used for future references.

**Requirements Traceability** – F5

**Priority** - High

**Preconditions** - The admin should be logged into the application. The visitors have used the swimming pool and the user is found by searching their membership id.

**Post conditions** - The details of the visit would be stored in the database and can be retrieved when required.

**Actors** – Admin

**Extends** – U4

**Flow of Events**

#### 1. Basic Flow -

Actor's actions	System responses
-----------------	------------------

Admin clicks on get user details button in home page and get the details of user by searching using membership id	The details of the user will be displayed.
The admin clicks on add visit	The count of the visitor's visit would be incremented and the date of the particular day would be saved as date of visit in the database.

## 2. Alternative Flow -

Actor's actions	System responses
1.Admin clicks on get user details button in home page and get the details of user by searching using membership id	2.The details of the user will be displayed.
3.The admin clicks on add visit	4.The count of the visitor's visit would be incremented and the date of the particular day would be saved as date of visit in the database.
	5.If the count of free visits made by the student exceeds 5, an email would be sent to the student to notify them about the dues they have to pay.

## 3. Exceptions - NA

**Includes** - NIL

**Notes/Issues** - NIL

### 3.3.7 Use Case #7 (Add payment– U7)

---

**Author** – Srushti Waghode

**Purpose** - User will be able to add payment details for a member.

**Requirements Traceability** – F6, F12, F16

**Priority** - High

**Preconditions** - The admin should have logged into the system. The admin should have searched for the member using the membership id. The member should have paid his/her dues and provided the payment receipt at the counter.

**Post conditions** - The payment details get added to the system database.

**Actors** – Admin

**Extends** – U4 - Get user details

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
The admin clicks on the 'add payment' button.	Options to add the payment amount and date of payment get displayed on the screen.
The admin enters the payment details.	The payment details are added to the database.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** U8 - Add amount, U9 - Add date of payment

**Notes/Issues** - NIL

### 3.3.8 Use Case #8 (Add amount– U8)

**Author** – Aparna V

---

**Purpose** - This use case allows the admin to add the amount , once the visitor pays the dues.

**Requirements Traceability** – F6, F12

**Priority** - High

**Preconditions** - The admin must have logged in and added the visitor into the system. The visitor must have paid the dues.

**Post conditions** - The admin add the amount and new payment status will be updated successfully

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Admin clicks on Add payment option	Two options, add amount and add date of payment is displayed on the screen.
Admin clicks add amount option and enter the amount paid by the visitor	
Click submit button	The amount paid is added and updated in the database.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** NIL

**Notes/Issues** - NIL

### 3.3.9 Use Case #9 (Add date of payment – U9)

**Author** – Hridika K V



---

**Purpose** - Allows admin to add date of payment ,once the visitor pays the dues.

**Requirements Traceability** – F6, F16

**Priority** - High

**Preconditions** - The admin must be logged in to the app.The visitor must have paid the dues, and brought the receipt.

**Post conditions** - The date of payment for the corresponding quarter will be added to the database.

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Admin clicks on the add payment button for the corresponding visitor.	A page would be displayed which has options like add amount, select date of payment.
The admin selects the date as well as enters the amount paid by the visitor and clicks on the 'submit' button.	The date of payment as well as payment amount of the particular user would be added and updated in the database.

2. Alternative Flow -NIL

3. Exceptions - NIL

**Includes** -NIL

**Notes/Issues** - NIL

### 3.3.10 Use Case #10 (Delete user-U10)

**Author** – Jagriti Sharma

---

**Purpose** - To delete a particular user of the swimming pool.

**Requirements Traceability** – F10

**Priority** - Medium

**Preconditions** - Admin must be logged in to the system. Admin should first search the user to be deleted.

**Post conditions** - User will be deleted from the database.

**Actors** – Admin

**Extends** – U4 - Get user details

**Flow of Events**

1. Basic Flow-

Actor's actions	System responses
Type the membership id in the search box and click on the search button.	The user will be searched in the database, if found, details of user and various other options like 'delete user' will appear on the screen.
Choose the 'delete user' option.	A modal will appear on the screen, asking the admin if they are sure about deleting the user.
Click on 'confirm' button	User will be deleted from the database

2. Alternative Flow -

Actor's actions	System responses
Search for a particular user.	The user will be searched in the database, if found, details of user and various other options like 'delete user' will appear on the screen.

Choose the 'delete user' option.	A modal will appear on the screen, asking the admin if they are sure about deleting the user.
Click on 'cancel' button	Previous page containing the user's information will be displayed

### 3. Exceptions

Actor's actions	System responses
Search for a particular user.	User not found in the database. Admin will be redirected to the page with a search button and a message will be displayed on the screen stating that 'user not found'.

**Includes** - NIL

**Notes/Issues** - NIL

#### 3.3.11 Use Case #11 (Get visit details – U11)

**Author** – Pooja S Nair

**Purpose** - Admin would be able to see the visit details of pool users.

**Requirements Traceability** – F9

**Priority** - High

**Preconditions** - User must have been added into the application .

**Post conditions** - Visit details of the users will be displayed.

**Actors** – Admin

**Extends** – U4 - Get user details

#### Flow of Events

##### 1. Basic Flow -

---

Actor's actions	System responses
Clicks 'get visit details' button	A dialogue box to enter the membership id is displayed
Enter the membership id and submit	Details of the user's visit are displayed.

2. Alternative Flow - NIL

3. Exceptions -

Actor's actions	System responses
Clicks get visit details button	A dialogue box to enter the membership id is displayed
Enter the membership id and submit	User not found in the database. Admin will be redirected to the page with a search button and a message will be displayed on the screen stating that 'user not found'.

**Includes** - NIL

**Notes/Issues** - NIL

### 3.3.12 Use Case #12 (Get payment details - U12)

**Author** – Srushti Waghode

**Purpose** - Admin should be able to get payment details of the member.

**Requirements Traceability** – F8

**Priority** - Medium

---

**Preconditions** - The admin should be logged into the system. The admin should have searched for the member using the membership id.

**Post conditions** - Payment details of the member should get displayed on the screen.

**Actors** – Admin

**Extends** – U4 - Get user details

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
The admin clicks on the 'get payment details' button.	Payment details of the member get displayed on the screen.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** NIL

**Notes/Issues** - NIL

**3.3.13 Use Case #13(Get information about today's visit – U13)**

**Author** – Aparna V

**Purpose** - The admin will be able to get the information about the visits on that particular day.

**Requirements Traceability** – F14

**Priority** - Medium

**Preconditions** - The admin must be logged into the system and there should be visitors on that day.

**Post conditions** - Information about today's visit is displayed on the screen

**Actors** – Admin

---

**Extends – NIL**

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Admin clicks on 'get information about today's visit' option.	Information about visitors who visit on that day is displayed on the screen.

2. Alternative Flow - NIL

3. Exceptions -

Actor's actions	System responses
Admin clicks on 'get information about today's visit' option.	If there are no visitors on that day a message will be displayed on the screen stating 'no visitors today'.

**Includes NIL**

**Notes/Issues - NIL**

**3.3.14 Use Case #14 (Get users with dues – U14)**

**Author –** Hridika K V

**Purpose -** Allows admin to get the list of visitors who have not paid the dues.

**Requirements Traceability –** F13

**Priority -** High

**Preconditions -** The admin must be logged into the application.

**Post conditions -** The admin can view the list of users with dues.

---

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow - The admin can view the list of users with dues.

Actor's actions	System responses
Admin clicks on get users with dues from login page	A page would be displayed which has the list of users with dues

2. Alternative Flow -Search for each user and check if they have dues.

Actor's actions	System responses
Admin clicks on get user details from login page and search by membership id	The details of the particular user would be displayed along with their payment status and dues.
Continue the same for every user	

3. Exceptions - The list would be empty if everyone has paid dues for a quarter.

**Includes** -NIL

**Notes/Issues** - NIL

**3.3.15 Use Case #15(Get quarterly report – U15)**

**Author** – Jagriti Sharma

**Purpose** - To get the quarterly report containing all the information about users of the swimming pool.

**Requirements Traceability** – F11

---

**Priority** - High

**Preconditions** - Admin must be logged in to the system.

**Post conditions** - Admin will be able to view quarterly reports.

**Actors** – Admin

**Extends** – NIL

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Select 'get quarterly reports' option.	Quarterly reports will be displayed.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** - NIL

**Notes/Issues** - NIL

### **3.3.16 Use Case #16 (Sort by category – U16)**

**Author** – Pooja S Nair

**Purpose** - Admin would be able to see separate quarterly reports of different categories of pool users.

**Requirements Traceability** – F11

**Priority** - Low

**Preconditions** - Visit details of users must have been stored in the database.

**Post conditions** -Quarterly reports of different categories of pool users will be visible.



---

**Actors** – Admin

**Extends** – U15 (Get quarterly report)

**Flow of Events**

1. Basic Flow -

Actor's actions	System responses
Clicks sort by category button	List of categories of pool users are displayed
Select the category for which quarterly report has to be generated	Quarterly reports of that category of pool users are displayed.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes** - NIL

**Notes/Issues** -NIL

### **3.3.16 Use Case #17 (Logout - U17)**

**Author** – Srushti Waghode

**Purpose** - The admin should be able to log out of the system.

**Requirements Traceability** – F15

**Priority** - High

**Preconditions** - The user should have logged into the system.

**Post conditions** - The user should be able to log out of the system.

**Actors** – Admin

---

**Extends – NIL**

**Flow of Events**

1. Basic Flow - The admin should click on the logout button to log out of the system.

Actor's actions	System responses
The admin clicks on the log out button.	The admin is logged out of the system.

2. Alternative Flow - NIL

3. Exceptions - NIL

**Includes NIL**

**Notes/Issues - NIL**

## **4 Other Non-functional Requirements**

### **4.1 Performance Requirements**

- Concurrent users:The application is able to support 50 concurrent users.
- Response time:The average response time between the click and reaction is 2s and the maximum response time must not be greater than 5s.
- Normalisation:The database is normalised to prevent redundancy.

### **4.2 Safety and Security Requirements**

- The password and username for admin is given
- Passwords must be saved encrypted in the database in order to ensure the user's privacy
- Validation checks must be performed.
- The system must be protected against vulnerabilities such as SQL injection attacks.

### **4.3 Software Quality Attributes**

#### **4.3.1 Correctness**

The application must satisfy all the functional and nonfunctional requirements mentioned above.

---

### **4.3.2 Reliability**

The chances of having a system failure must be low and if the system crashes the mean time to recover must be at most 15 minutes. So that application will be reliable to record the data of the user's visit to the pool.

### **4.3.3 Maintainability**

The system code should be written to allow future modifications. Code will be fully commented.

### **4.3.4 Reusability**

Code is designed in a modular manner so that it can be reused for the future versions.

### **4.3.5 Availability**

Application must be available to all users under normal levels of concurrency.

### **4.3.6 Usability**

The interface of the application is designed in a user friendly manner such that the user doesn't need any training to use the application.

### **4.3.7 Recoverability**

After a system failure, almost 95% of data can be recovered.

## **5 Other Requirements**

NIL

## **Appendix A - Activity Log**

A meeting was conducted on 2nd February 2022. The meeting lasted for half an hour. In the meeting, the document was divided and each team member has been assigned a part as follows:

1. Introduction - Aparna V
2. Product Overview, Design and implementation constraints - Hridika K V

- 
3. Definitions, Acronyms and Abbreviations, Product Functionality, Assumptions and dependencies - Jagriti Sharma
  4. External interfaces Requirement - Srushti Waghode
  5. Other Non-functional Requirements - Pooja S Nair
  6. Functional requirements - By all the team members.

The use case diagram was made collectively by the team members. Each member has contributed to the use cases as mentioned in the document.

Further discussions were through the whatsapp group.