Software Requirements Specification

For

Body Zone (GYM) Desktop

Version 1.0 approved

Prepared by 17L-4135, 17L-4175, 17L-4252

FAST Dev.

11/11/2020(dd/mm/yy)

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
FAST Dev.	11/11/2020	Initial Document	0.0.0

1. Introduction

In the below sections introduction to Desktop Application of Body Zone GYM and to SRS document is to be provided.
With addition to that, conventions and short terms are explained in section no. 1.2 and 6.

1.1 Purpose

Desktop application is being developed and its release number is 0.0.0. The desktop application of Body Zone GYM includes mainly the management of customer fee w.r.t to their due deadline.

1.2 Document Conventions

*The document includes sectioning. Where we have headings in **bold** style, Times New Roman and large font. Sub-headings leading to headings are in **bold** style with relatively smaller font.*

*And factors/points of a topic are considered with bullet points and addition of **bold** style. And text in **bold** tells about the highlights of the paragraph or its most important part of writing.*

1.2.1 Document Word Abbreviations & Meanings

- ***Client = Customer = member=** The person who comes to the gym and does exercise.*
- ***Operator = Front Desk Person (FDP) = gym administration=**the person on the front desk of the gym and records data of clients in the system, takes fee, marks attendance.*
- ***Membership** = The person who is paying a fee to the gym on a monthly basis.*
- ***30 Day Formulae** = Discontinuation policy, that says to remove a member from the system if a member fails to pay the fee within the next 30 days of due date.*
- ***Front-end framework** = React.js.*
- ***Components** = Html, Css, Javascript.*
- ***Styling-libraries** = Bootstrap, React Windows, React-UI.*

1.3 Intended Audience and Reading Suggestions

In the sections below we will describe the document's intended readers and the way this SRS is organized as well as its reading pattern to understand the development pattern.

1.3.1 Intended Audience

There are different types of readers to the Body Zone gym's desktop application's document. Our main readers of the document are Developers, testers, project managers and software house owners.

- **Project Managers**

  The document is intended mainly for this desktop application's project managers. The project managers can understand the decided requirements, technologies to be used and complete software's requirement

- **Developers**

  SRS document's essential part consists of functional requirements, non-functional requirements, data flow understanding, data formation and et cetera. Therefore, all these parts are required by the developer to build the code without any ambiguity.

- **Testers**

  Similarly, testers require these documents to verify requirements and to deduce test cases from the document. To test the system thoroughly, this document will be required by them.

- **Software House Owners**

  The whole document gives the complete insights to the project being implemented, hence, this is important for software house owners to know about the project and its extensiveness.

1.3.2 SRS Organization & Reading Sequence

The proceeding SRS document describes the category of users of this system w/said knowledge, the description of the system being developed and its behavior including the functionalities that are to be provided.

It further includes flow of data, cases that the system will handle, interaction of users and who will eventually interact with this software and how the system will respond to that user.

The document must be read from an ***introduction*** *that classifies the users and their needs inside this document. Moreover, after the introduction part the detailed* ***system description*** *of* ***behaviors and responses*** *is documented and that is followed by* ***system requirements****, system responses and* ***conventions*** *followed in the document being considered.*

*The **requirements** part and **overall description** are to be considered **mainly** to understand the behavior of this system.*

1.4 Product Scope

BodyZone desktop application is a fee management tool that allows gym administration to monitor fees that eventually manages the members of BodyZone.

The BodyZone software targets to manage the members in a soft hand w/monthly closing of finance, the business gives clients a deadline of 30 days to pay for the coming month, failure to do so does not initiate any action instead more of the 30 days are added, so that client pays easily.

After additional 30 days, the member is deleted from the fee management tool and membership is discarded.

2. Overall Description

The description of overall software being developed is explained in the sections below

2.1 Product Perspective

BodyZone gym traditionally manages the attendance, membership and fees of members through manual documentation on papers. BodyZone Desktop application developers believe in digital solutions that provide transparency, finance benefits and cheat free income.

Therefore, BodyZone desktop's dynamic application is a solution to the gym fees management problem, along with many sub problems being covered in it.

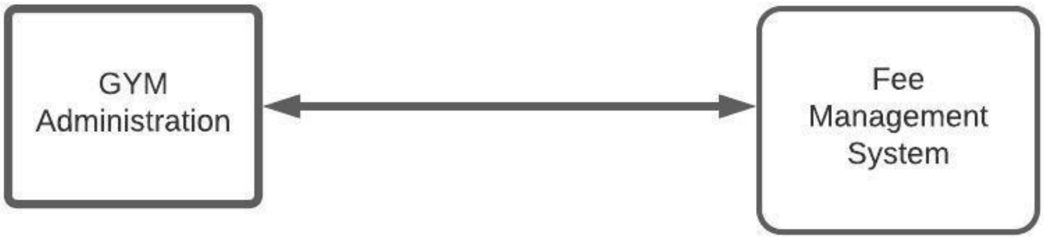
The customers of the gym who want to do exercise at the gym will come to the front desk person (FDP). FDP will use the software to enter the customer details in the system. So customers will not directly interact with the system. FDP will directly interact with the system.

2.2 Product Functions

BodyZone desktop application's main functions are listed below and are all managed by gym administration (users)

- *Users will enroll members.*
- *System will generate Id for them.*
- *Mark Fee collected.*
- *System will delete the user after 30 days from his due fee date, if not collected.*

- *Deleted users can be re-enrolled, but with a new id.*
- *Users can be deleted and their information can be edited*
- *Users can search for members. Search can be based on Id, Mobile No.*
- *Users can apply filters for members with late fees and members whose fees are expected today.*



2.3 User Classes and Characteristics

- **Gym Administration**
  - ***Who****: Users of this software include gym administration only that includes owner and gym manager.*
  - ***Why****: The privileges to manage the members fee are reserved for them*
  - ***How****(qualification to system)*
    - *Windows: GYM Admin must be sound to operating windows*
    - *System UI/UX: GYM Admin must be able to use UI buttons, drop downs and all micro UIs and must be UX literate.*
    - *Search Engine: GYM Admin must know the usable search engine embedded.*
    - *Filter: GYM Admin must be sound to information filtering, through the relative filter option provided.*

2.4 Operating Environment

The system will operate on Windows 10, and will use the Standard Operating Environment of Windows 10. On Windows 10 the JavaScript library of Node is being used to manage the project along with technologies that make up Electron.js for Desktop development.

The application developed will be initially compatible for the desktop application. But, as APIs being built are in REST, we can extend the project compatibility with mobile and web applications.

Moreover, the software can be modified to work on different operating systems such as linux. Keeping the overhead of modification cost in mind, is important to consider.

2.5 Design and Implementation Constraints

There are no special constraints for the developers. But, using the NoSql database is a constraint introduced by the developers. And maintaining the software is solely associated with the developers of the software.

Further, security considerations and rights to access are for future evolution in Body Zone's Desktop application.

2.6 User Documentation

As the scope of the project is limited and requires a very short training to get used to it. Therefore no user documentation is required here.

2.7 Assumptions and Dependencies

The software is purely independent project, but some dependencies and assumptions are to be mentioned below

- **Operating System**:

  The operating system for which the software is being developed is intended to be Windows 10. The installations, development happens to be on this operating system. Failing to do so may result in installation errors.

- **Technology Deprecation**:

  The technology being used for the development of BodyZone gym desktop application is known to be Electron.js based on Node.js. Therefore, the depreciation of node modules/methods can be another factor that requires maintenance at backend.

  The node.js/Electron.js/Express.js if not undergone drastic changes of version can be catered till that type of change occurs.

3. External Interface Requirements

In this section we will document the external interface requirements

3.1 User Interfaces

- Screen that will show **all the members** of the gym that are registered.
    - List View of members where each individual cell contains member ID and Name.
    - Each individual list item will have a remove and edit button next to it.
    - Search bar on the top of the page.
- Screen that will allow the user to **enter member** information.
    - Label and Text Input Box for each entry for a member.
    - Ok button at the bottom to finalize the entry.
- Screen that will show the names of the **members** who have their **payment due**.
    - List View of members where each individual cell contains member ID and Name.
- Screen that will allow the user to **edit member** information.
    - Label and Text Input Box for each entry for a member.
    - Ok button at the bottom to finalize the entry.

Buttons and views will be designed to adhere to the material design guidelines set by Google.

3.2 Software Interfaces

Software used	***Description***
Operating system	*We have chosen Windows operating system for its best support and user-friendliness.*
Database	*To save the member records, payment records we have chosen a NoSQL database namely Mongo db.*
Electron	*To implement the project we have chosen Electron framework for its more interactive support.*

ORM	*To communicate with the database we have chosen an ORM mongoose for easy querying of data.*

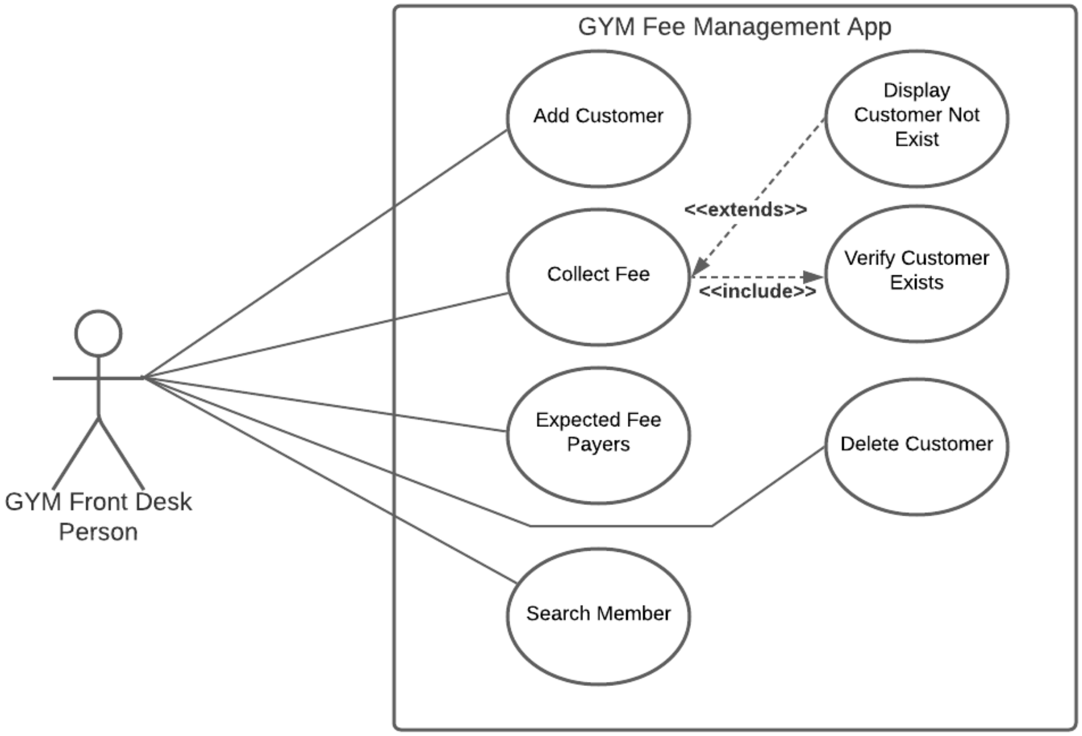
The data passing between the Electron framework and the React front-end will be done through inter processing communication and event handling. The nature of the data being passed will be of primitive types to avoid crashes. More complex data will be first converted to primitive type and then converted back into original after it has been passed to the process.

3.3 Communications Interfaces

This software will run on all versions of Windows 10 and all systems with minimal specs. The software will use HTTP for communicating with the remote database through a library called mongoose.

4. System Features

A general overview of system is shown in a use-case diagram:



4.1 Register the Client (Admissions Open)

4.1.1 Description and Priority

The system should be able to register new clients. System will get the client details, get the registration date, generate a roll number and add this data to the system. Client details are name, phone number. This requirement is on High priority. Priority = 10.

4.1.2 Stimulus/Response Sequences

The client comes to the front desk of the gym. Client tells that he wants to register as a new client in the gym. The front desk person will record client details, get the fee amount and add this user to the system. The system responds as the customer added successfully that stimulate the behavior defined for this feature.

4.1.3 Functional Requirements

Get the data fields and add these to system:

- Name (Required Field) (Client provides this information)
- Phone (Optional Field Due to Female Clients)(Client provides this information)
- Date Joined (Required Filled) (Operating system provides this information)
- Roll Number (Auto filled Field) (Software system provides this information)
- Other information (Optional Field) (Client provides this information)

If the values provided are invalid then:

Show a message to the user if any field(s) are invalid. So he can correct the values and the system can accept data.

Valid Data format for the fields is provided below. Any format other than this will be considered invalid.

- Name = String, can not empty, max length 255 characters.
- Phone = String, can be empty, max length 14 characters
- Date Joined = String, can not empty, max length 255 characters.
- Roll Number= Integer, can not empty
- Other information = String, can not empty, max length 2048 characters.

For Example: Muneeb, +923014440289, Joined on 13 Nov 2020, Roll: 204, Other:(empty)

4.2 Collect Client Fee

4.2.1 Description and Priority

The system should be able to record the fee given by the client. This feature has priority = 9.

4.2.2 Stimulus/Response Sequences

The client will come to the front desk of the gym. Provide his name, roll number or phone number and give the fee amount to the front desk person (FDP). The FDP will add the details in the system. And the system will show a message that the fee is submitted successfully.

4.2.3 Functional Requirements

Functional requirements associated with this feature:

- User provides Name OR Phone Number OR Roll Number (Required)
- User provides fee amount (Required)
- System auto fills current data and due date in form (Required)
- Other information (Optional Field) (Client provides this information)

If the values provided are invalid then:

Show a message to the user if any field(s) are invalid. So he can correct the values and the system can accept data.

Valid Data format for the fields is provided below. Any format other than this will be considered invalid.

- Name = String, can not empty, max length 255 characters, Must exist in Client register DB
- Phone = String, can be empty, max length 14 characters, Must exist in Client register DB
- Roll Number = Integer, can not empty, Must exist in Client register DB.
- Current Date = String, can not empty, max length 255 characters
- Due Date = String, can not empty, max length 255 characters
- Other information = String, can not empty, max length 2048 characters

For example: Muneeb submitted 3000 Fee, Due Date is: 11 December 2020

4.3 Show Clients with Expected Fee

4.3.1 Description and Priority

The system should be able to tell those clients of the gym who have to pay a fee at the current date or their due date has passed. So clients can be tracked by the gym management and asked to pay the fee. This would be a huge benefit in terms of cost and profit for the gym. The priority is 8.

4.3.2 Stimulus/Response Sequences

The user will be shown those clients who did not pay fees in time in a list with their name, phone number and roll number.

4.3.3 Functional Requirements

Show those clients details who did not pay the fee in time. The client details are registration date, roll number, name, and phone number.

4.4 Search Member by Name by Phone by Roll No.

4.4.1 Description and Priority

The system should be able to show details of a client when the client provides his name or phone number or roll number. This feature will save a lot of time for gym management and clients to see their fee paid or fee pending and details such as name, phone, roll number. The priority is 7.

4.4.2 Stimulus/Response Sequences

The client of the gym can see his activity in the past as well as fee submitted, details such as name, phone, roll number.. The FDP will ask the client for name, phone number or roll number and will press the button to show details and the system will show the past member activity in the gym as well as fee submitted and details such as name, phone, roll number.

4.4.3 Functional Requirements

Functional requirements associated with this feature:

- User provides Name OR Phone Number OR Roll Number (Required)
- System shows:
  - Member attendance
  - Fee submitted or not
  - Name
  - Phone
  - Roll number

If the values provided are invalid then:

Show a message to the user if any field(s) are invalid. So he can correct the values and the system can accept data.

Valid Data format for the fields is provided below. Any format other than this will be considered invalid.

- Name = String, can not empty, max length 255 characters, Must exist in Client register DB
- Phone = String, can be empty, max length 14 characters, Must exist in Client register DB
- Roll Number = Integer, can not empty, Must exist in Client register DB.

For example: Search Muneeb gave: Name: Muneeb, Roll: 201, Phone: +923014440289, Registered on 13 Nov 2020, Fee gave 13 Nov 2020 = 3000Rs, Came in november = 12 days

4.5 Delete Member by Name by Phone by Roll No.

4.5.1 Description and Priority

The system should be able to delete a client. This requirement is needed when a client wants to leave the gym or some time a client account must be deleted due to some policy violation of the gym. FDP adds the client name or phone number or roll number and the system will delete the customer. The priority is 6.

4.5.2 Stimulus/Response Sequences

The client will come to the front desk of the gym. Provide his name or roll number or phone number to the front desk person (FDP). The FDP will delete the client in the system. And the system will show a message that the user is deleted successfully.

4.5.3 Functional Requirements

Functional requirements associated with this feature:

- Client provides Name OR Phone Number OR Roll Number (Required)

If the values provided are invalid then:

Show a message to the user if any field(s) are invalid. So he can correct the values and the system can accept data.

Valid Data format for the fields is provided below. Any format other than this will be considered invalid.

- Name = String, can not empty, max length 255 characters, Must exist in Client register DB
- Phone = String, can be empty, max length 14 characters, Must exist in Client register DB
- Roll Number = Integer, can not empty, Must exist in Client register DB.

For example: Client: Muneeb, deleted from system successfully.

4.6 Delete Defaulter Clients

4.6.1 Description and Priority

The system should delete the user who has not given the fees after 60 days of due date. The priority is 5.

4.6.2 Stimulus/Response Sequences

The system will automatically delete the user who has not given the fees after 60 days of due date. And show notification to the user that this user is deleted.

4.6.3 Functional Requirements

A popup to show that a customer is deleted.

For example: Umar is deleted. Umar did not pay a fee after 60 days of due date.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>