**04170207**

**Software Engineering**

Muscles Monsters(fitness club)

**Final SRS**

**Group Participants**

1- Amr Samy RizkAllah (G4) ID: 2030386

2- Zeyad mohamed Hammad(G2) ID: 203052

Nov 2020

**Contents**

[Team](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.3rdcrjn) 1

[Document Purpose and Audience 3](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.26in1rg)

[Introduction 3](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.gjdgxs)

[Software Purpose 3](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.30j0zll)

[Software Scope](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.1fob9te) 3

[Requirements](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.2et92p0) 3

[Functional Requirements](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.tyjcwt) 3

[Non Functional Requirements](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.3dy6vkm) 4

[System Models](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.1t3h5sf) 5

[Use Case Model](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.4d34og8) 5

[Use Case Tables](https://docs.google.com/document/d/1G6mhnoQxQkNt9HhfKXaUzdQbfc8TuXyTXoziGpCraXY/edit#heading=h.2s8eyo1) 6 - 11

# Introduction

## Software Purpose

* **Automating our activities in fitness Clubs to encourage and attract more customers.**

## Software Scope

* **Automating admission and paying fees process through an online money transfer service.**
* **Giving the ability to members to select perfect Trainers as they wish.**
* **Attracting members with new Packages and discounts.**
* **Observing members activities and advising lazy members to do more Workouts.**
* **strict attendance system Managed by the admin to prevent absence.**

# Requirements

## Functional Requirements

## Members can:

1. Booking and paying Fees online using an online money transfer service.
2. View available skilled Trainers and pick anyone you want.
3. Selecting your own package depending on your own preferences and case.
4. Giving feedback to trainers to help others decide.
5. View Their profile

**Admin can**

1. Login to the System using his own credentials.
2. Managing packages to attract new members.
3. Managing members details (adding-updating-deleting)
4. view attendance and the package of a member.

**Trainer can**

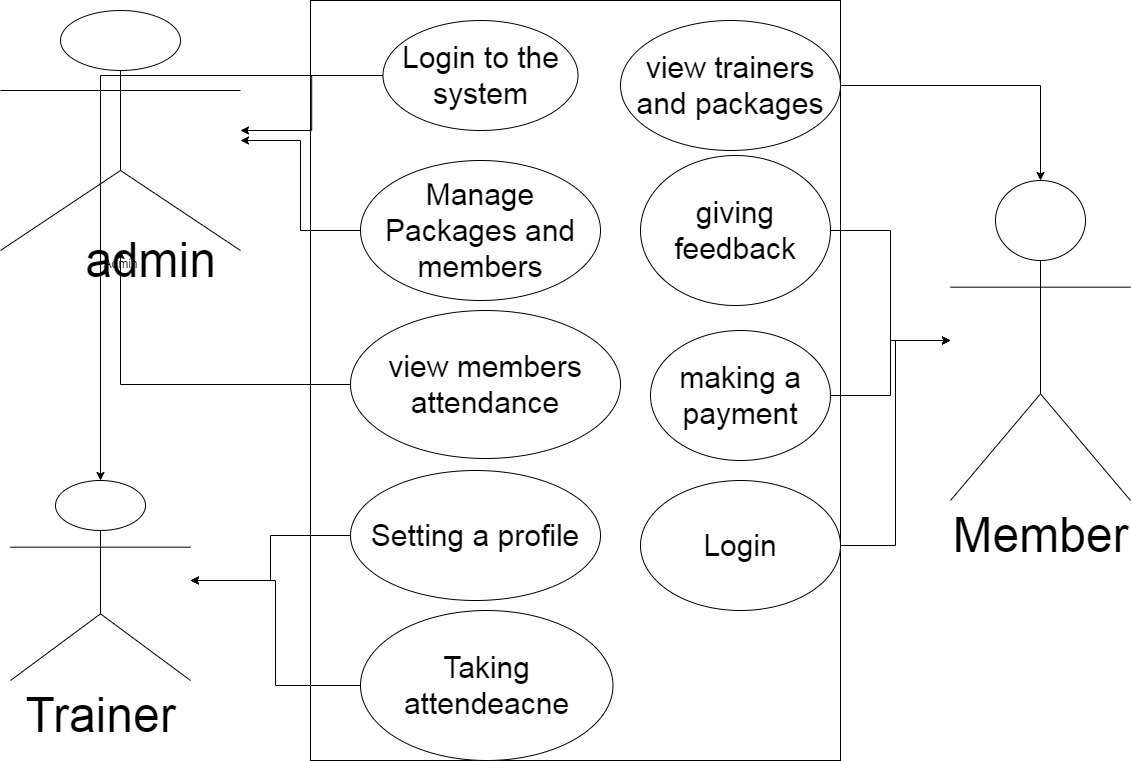
1. Setting his own profile and then login to it
2. Take the attendance.
3. Changing his own password and profile.

## Non Functional Requirements

1. Automating the admission process and making the registration process easy and fast.
2. An Powerful Encryption Algorithm to hash user’s credit card data from hackers.
3. Attracting members with a fully automated packages system and discounts offers.
4. Making a feedback System to help new members select their own suitable trainer.
5. Strict attendance system to make members participate more.

# System Models

## 1- Use Case Model



## 1- Use Case Tables

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 1 | |
| Use Case Name: | Login to the System | |
| Actors: | Admin-members-Trainer | |
| Pre-conditions: | display the login page for user | |
| Post-conditions: | Successful login or failed with a message | |
| Flow of events: | User Action | System Action |
| 1- user Enter his Email And password |  |
|  | 2-System search in the database |
|  | 3-if successed login, System will Verify user data and Specify what’s kind he is  (Admin-member-Trainer) Then displays The suitable page |
| Exceptions: | User Action | System Action |
| 1- the user Enter his Email And password |  |
|  | 2-if Email or password was not found in the database ,  the System will refuse it and Display a Message to the User for re-submitting the failed |
|  | 3-if the user forget his password | 4- sending an email to re-setting his password |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 2 | |
| Use Case Name: | Manage packages And Members | |
| Actors: | Admin | |
| Pre-conditions: | Displaying a page with ALL options to update Member’s data (adding -deleting-View Members registration details - view attendance ) | |
| Flow of events: | Admin Action | System Action |
| 1-Selecting adding Members |  |
|  | 2-displaying a new page asking the admin to send user-information to add it . |
| 3- Sending user-information |  |
|  | 4- Receiving user-information and then Inserting it into the database . |
| 1 - If the admin Select updating Member’s options |  |
|  |  | 2-Asking the admin for sending New member’s data |
|  | 3- Sending new user-information After Updating it. |  |
|  |  | 4-System receiving new information and then Inserting it into the database |
|  |  | 5-returns with a "successful edit" message. |
|  | 1 - If the admin Select Deleting a Member | 2-Asking for the member name to be removed. |
|  | 3-Selecting a member. |  |
|  |  | 4-returning with a confirmation message to remove. |
|  | 5-Accept the removing process. |  |
|  |  | 4-Deleting it permanently from the database. |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 3 | |
| Use Case Name: | View attendance of Members | |
| Actors: | Admin | |
| Flow of events: | Admin Action | System Action |
| 1- Navigate to Attendance Page and select a member |  |
|  | 2- Using this member-id , System collects data from the database. |
| 3- Checking if the saved data is Matching the real data |  |
| Exceptions: | Admin Action | System Action |
| 1- admin surprisingly discovers there is a Wrong data in the database , and Takes an action with the Trainer and making him pays fine. |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 4 | |
| Use Case Name: | View Packages and Trainers , Then making a payment | |
| Actors: | Members | |
| Pre-conditions: | display available packages and Perfect Trainers | |
| Flow of events: | User Action | System Action |
| 1- browsing then booking a suitable package price with a perfect trainer based on his rating |  |
|  | 2-asking for a payment methods |
| 3-adding his credit card information then submitting it |  |
|  |  | 4- checking his balance then booking his package. |
| Exceptions: | User Action | System Action |
|  | 1- Wrong credit card information or no balance will end with an error page |
|  | 2- Selecting new method to pay |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 5 | |
| Use Case Name: | Giving feedback to Trainers | |
| Actors: | Members | |
| Flow of events: | User Action | System Action |
| 1- Selecting his own trainer |  |
|  | 2- Asking for a feedback for 1 to 5 and a comment |
| 3-Giving a feedback and a commt |  |
|  |  | 4- Inserting his feedback into the database and then Re-Sorting the Trainers depending on their feedback |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 6 | |
| Use Case Name: | Setting a public profile | |
| Actors: | Trainers | |
| Flow of events: | User Action | System Action |
|  | 1- Asking for his name and college degree |
|  | 2- Asking for a public picture and description about himself |
| 3-Submitting all this information |  |
|  |  | 4- Adding the user to the database |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 7 | |
| Use Case Name: | Taking attendance | |
| Actors: | Trainers | |
| Flow of events: | User Action | System Action |
| 1- Selecting attendance page |  |
|  | 2- Searching into the database for the members menu |
| 3- Checking if members are already have a valid membership |  |
|  | 4- Taking the attendance |  |
|  |  | 5-Inserting it into the database |