**Cairo University**
**Faculty of Computers and Information**

# CS251

# Software Engineering-I

## GoFo

Software Design Specifications

And Implementation

**(PM-132)**

Version 2.0

| ID | Name | Email |
|---|---|---|
| 20180158 | عبد الرحمن محمد رمضان | **yousef777906@gmail.com** |
| 20180392 | عبد العزيز جمال موسى | **Wqer20122@gmail.com** |
| 20180075 | توفيق ياسر توفيق ابوسيف | **tawfekyassertawfek@gmail.com** |

Jun 9, 2020

# CS251: Phase 2 – < PM-132 >
# Project: <GoFo>

# Software Design Specification

## Contents

# Software Design Specification

CS251: Phase 2 – < PM-132 >
Project: <GoFo>

# Software Design Specification

## Instructions

### Software Purpose

- Allow players to book playing hours in football playgrounds, (Software's audience Player, Playground Owner, and Administrator).

### Software Scope

- Software identification: Playground Booking System.
- System will connect player with playground owner to simplify booking playground process.
- System will not allow playground owners to send messages to player.
- Use Classes: User – Booking – Payment – Playground.
- Software Components / Major Features:
    - Book a playground, Create Profile, Payment, Mention, Search, and update time slots.

### Definitions, acronyms, and abbreviations

- Booking: the process of book a specific time slot for a specific playground.
- Administrator: Oversees the system operations and control a whole system.
- Playground Owner: a person how adds his playground.
- Player: a person how book a playground.
- eWallet: the way that users can send and receive money a cross it.

CS251: Phase 2 – < PM-132 >
Project: <GoFo>

# Software Design Specification

## Team

| Mobile | Email | Name | ID |
|---|---|---|---|
| 01129348206 | tawfekyassertawfek@gmail.com | Tawfik Yasser Tawfik | 20180075 |
| 01102053810 | yousef777906@gmail.com | Abd el-rhman Mohamed Ramadan | 20180158 |
| 01020687495 | Wqer20122@gmail.com | Abdulaziz Jmal mousa | 20180392 |

## Document Purpose and Audience

### Document Purpose:

- This document is written for SRS as a documentation for Go-Fo Application.
- To be an agreement between us and the client.
- This document contains information and instructions for developers and customers.

### Document Audience:

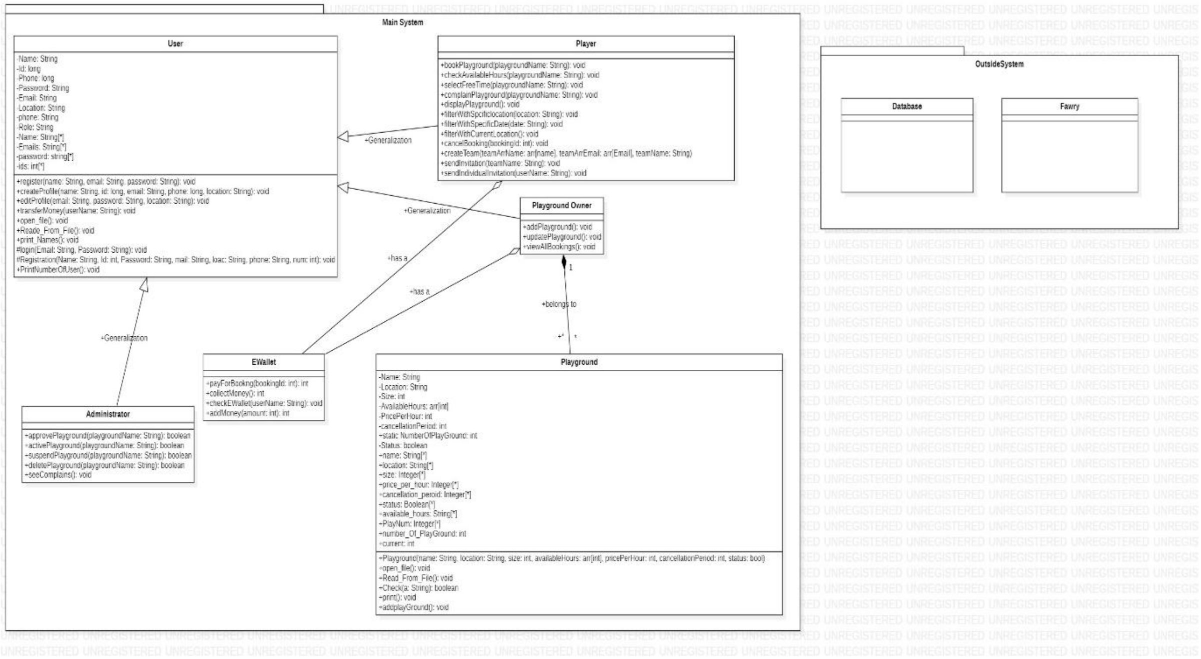
- Client – Developers – Judge (if happens disagreement) – Project Manager.

CS251: Phase 2 – < PM-132 >
Project: <GoFo>

# Software Design Specification

## System Models

### Class Diagram(s)



Main System
User
-Name: String
-Id: long
-Phone: long
-Password: String
-Email: String
-Location: String
-phone: String
-Role: String
-Name: String[*]
-Emails: String[*]
-password: string[*]
-ids: int[*]
+register(name: String, email: String, password: String): void
+createProfile(name: String, id: long, email: String, phone: long, location: String): void
+editProfile(email: String, password: String, location: String): void
+transferMoney(userName: String): void
+open_file(): void
+Reade_From_File(): void
+print_Names(): void
#login(Email: String, Password: String): void
#Registration(Name: String, Id: int, Password: String, mail: String, loac: String, phone: String, num: int): void
+PrintNumberOfUser(): void
Player
+bookPlayground(playgroundName: String): void
+checkAvailableHours(playgroundName: String): void
+selectFreeTime(playgroundName: String): void
+complainPlayground(playgroundName: String): void
+displayPlayground(): void
+filterWithSpcificlocation(location: String): void
+filterWithSpecificDate(date: String): void
+filterWithCurrentLocation(): void
+cancelBooking(bookingId: int): void
+createTeam(teamArrName: arr[name], teamArrEmail: arr[Email], teamName: String)
+sendInvitation(teamName: String): void
+sendIndividualInvitation(userName: String): void
+Generalization
+Generalization
+Generalization
Playground Owner
+addPlayground(): void
+updatePlayground(): void
+viewAllBookings(): void
+has a
+has a
1
+belongs to
+*
*
EWallet
+payForBookng(bookingId: int): int
+collectMoney(): int
+checkEWallet(userName: String): void
+addMoney(amount: int): int
Administrator
+approvePlayground(playgroundName: String): boolean
+activePlayground(playgroundName: String): boolean
+suspendPlayground(playgroundName: String): boolean
+deletePlayground(playgroundName: String): boolean
+seeComplains(): void
Playground
-Name: String
-Location: String
-Size: int
-AvailableHours: arr[int]
-PricePerHour: int
-cancellationPeriod: int
+static NumberOfPlayGround: int
-Status: boolean
+name: String[*]
+location: String[*]
+size: Integer[*]
+price_per_hour: Integer[*]
+cancellation_peroid: Integer[*]
+status: Boolean[*]
+available_hours: String[*]
+PlayNum: Integer[*]
+number_Of_PlayGround: int
+current: int
+Playground(name: String, location: String, size: int, availableHours: arr[int], pricePerHour: int, cancellationPeriod: int, status: bool)
+open_file(): void
+Read_From_File(): void
+Check(a: String): boolean
+print(): void
+addplayGround(): void
OutsideSystem
Database
Fawry

### Class Descriptions

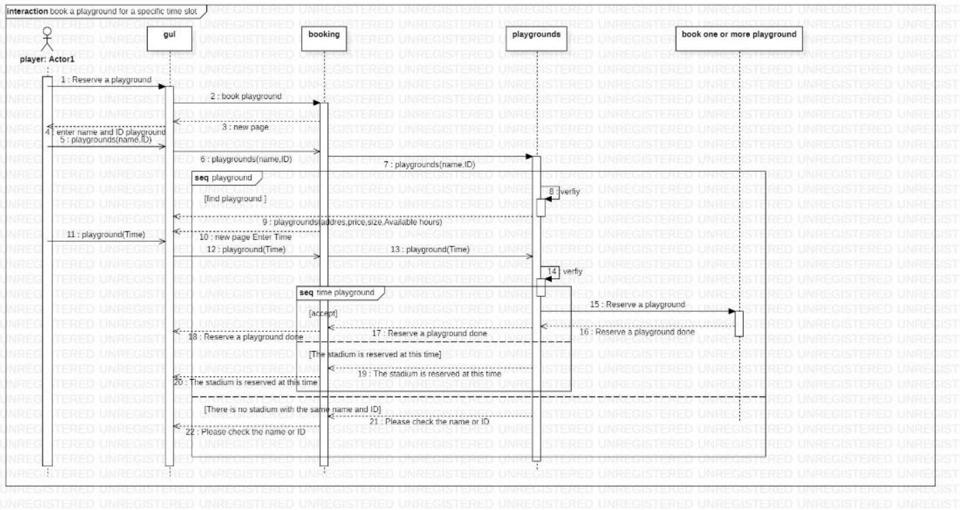
| Description & Responsibility | Class Name | Class ID |
|---|---|---|
| Responsible for recording and saving the data of all users | User | 1 |
| Through the chapter, the user can search for stadiums, book a stadium, send an invitation to his friends, and delete a reservation | Player | 2 |
| Class is the pitch measurements of the stadium location, stadium price and hours available by the stadium owner | Playground | 3 |
| He has the right to add a stadium, adjust his hours, and set a specific price per hour | Playground Owner | 4 |

# Software Design Specification

| Description & Responsibility | Class Name | Class ID |
|---|---|---|
| It can suspend stadiums and add stadiums and permanently delete them so players cannot book them if they find complaints or violate the laws | Administrator | 5 |
| Through it, the user can reserve stadiums with money as the reservation needs money and communicate with his bank account | E-Wallet | 6 |

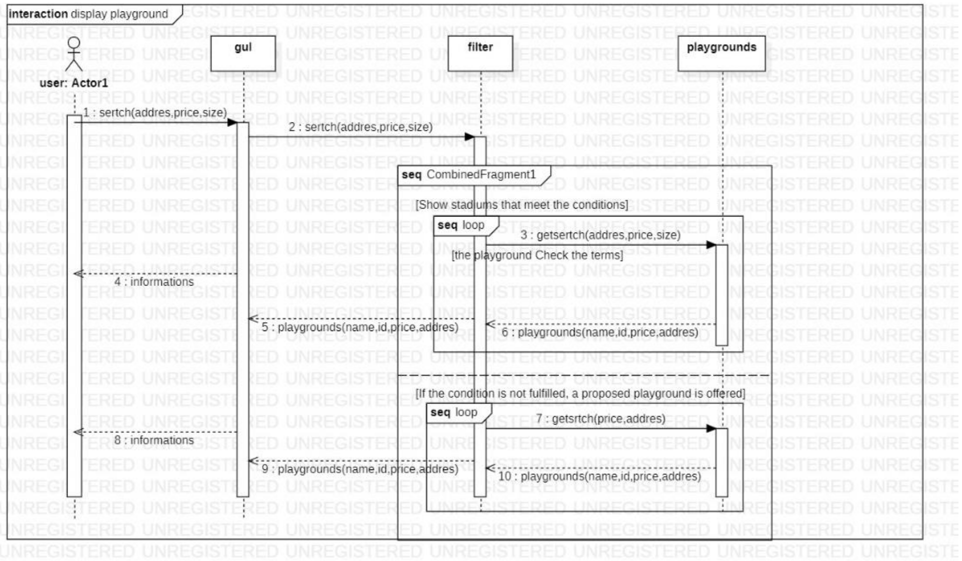
## Sequence diagrams

1.

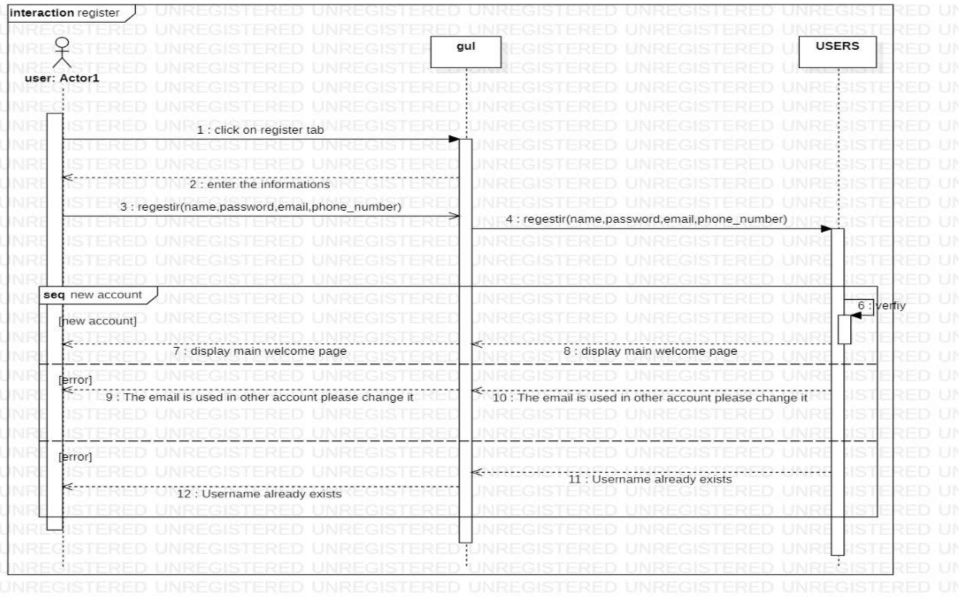

interaction book a playground for a specific time slot
player: Actor1
gui
booking
playgrounds
book one or more playground
1 : Reserve a playground
2 : book playground
3 : new page
4 : enter name and ID playground
5 : playgrounds(name,ID)
6 : playgrounds(name,ID)
7 : playgrounds(name,ID)
seq playground
[find playground ]
8 : verfiy
9 : playgrounds(addres,price,size,Available hours)
10 : new page Enter Time
11 : playground(Time)
12 : playground(Time)
13 : playground(Time)
14 : verfiy
seq time playground
[accept]
15 : Reserve a playground
16 : Reserve a playground done
17 : Reserve a playground done
18 : Reserve a playground done
[The stadium is reserved at this time]
19 : The stadium is reserved at this time
20 : The stadium is reserved at this time
[There is no stadium with the same name and ID]
21 : Please check the name or ID
22 : Please check the name or ID
UNREGISTERED

# Software Design Specification

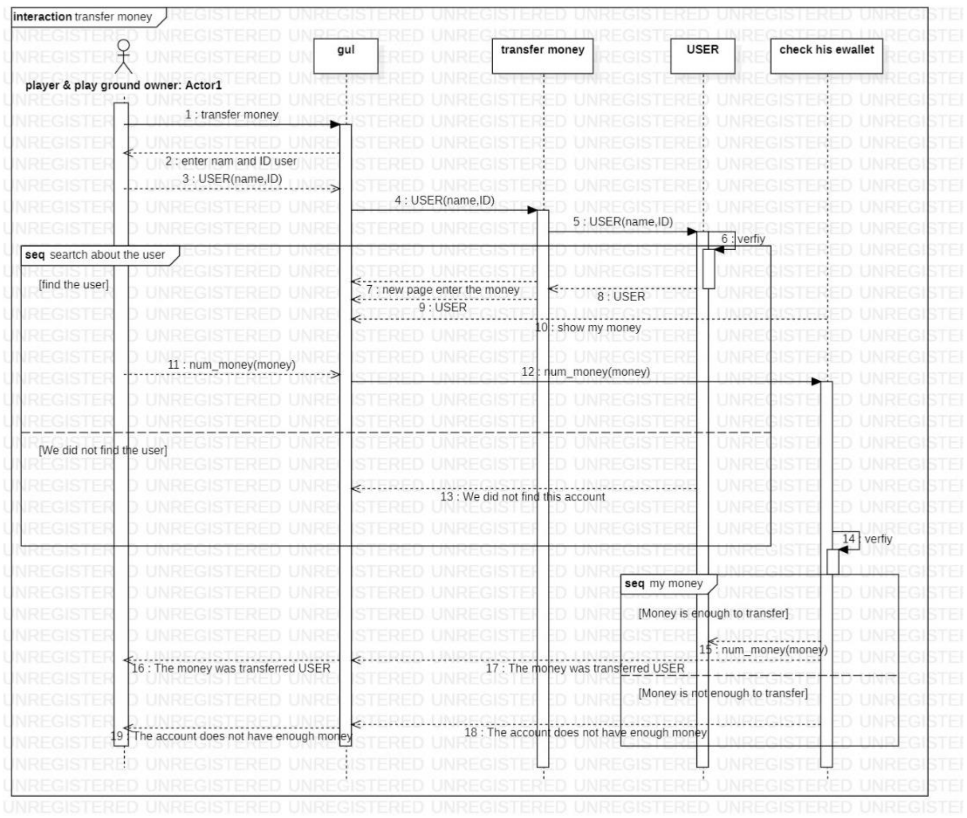
2.


interaction display playground
user: Actor1
gul
filter
playgrounds
1 : sertch(addres,price,size)
2 : sertch(addres,price,size)
seq CombinedFragment1
[Show stadiums that meet the conditions]
seq loop
3 : getsertch(addres,price,size)
[the playground Check the terms]
4 : informations
5 : playgrounds(name,id,price,addres)
6 : playgrounds(name,id,price,addres)
[If the condition is not fulfilled, a proposed playground is offered]
seq loop
7 : getsrtch(price,addres)
8 : informations
9 : playgrounds(name,id,price,addres)
10 : playgrounds(name,id,price,addres)

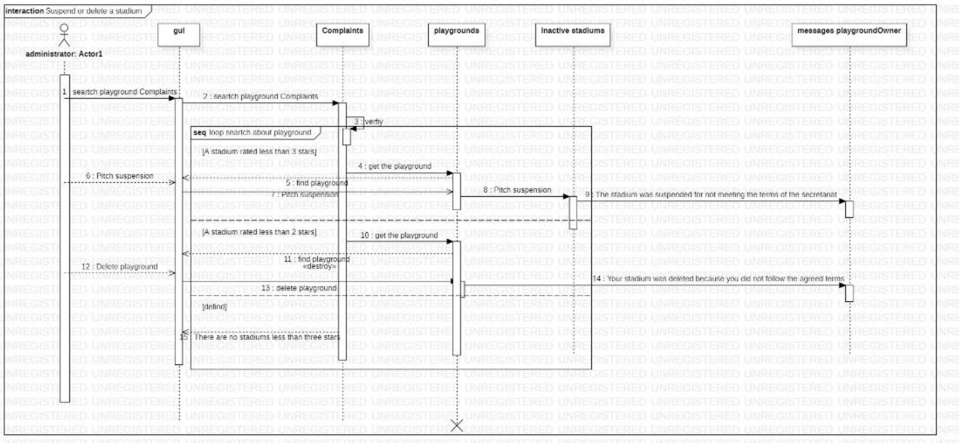
3.


interaction register
user: Actor1
gul
USERS
1 : click on register tab
2 : enter the informations
3 : regestir(name,password,email,phone_number)
4 : regestir(name,password,email,phone_number)
seq new account
[new account]
6 : verfiy
7 : display main welcome page
8 : display main welcome page
[error]
9 : The email is used in other account please change it
10 : The email is used in other account please change it
[error]
11 : Username already exists
12 : Username already exists

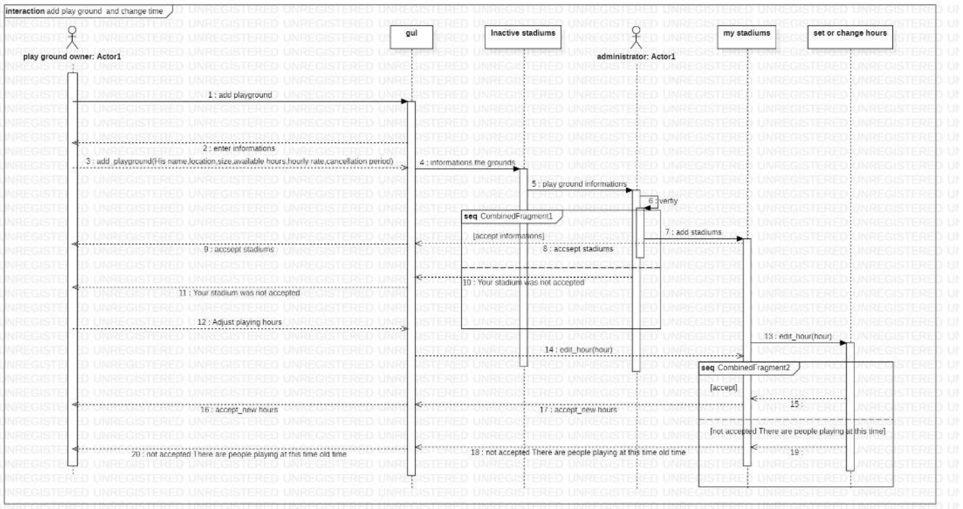
4.


interaction transfer money
player & play ground owner: Actor1
gui
transfer money
USER
check his ewallet
1 : transfer money
2 : enter nam and ID user
3 : USER(name,ID)
4 : USER(name,ID)
5 : USER(name,ID)
6 : verfiy
seq searth about the user
[find the user]
7 : new page enter the money
8 : USER
9 : USER
10 : show my money
11 : num_money(money)
12 : num_money(money)
[We did not find the user]
13 : We did not find this account
14 : verfiy
seq my money
[Money is enough to transfer]
15 : num_money(money)
16 : The money was transferred USER
17 : The money was transferred USER
[Money is not enough to transfer]
18 : The account does not have enough money
19 : The account does not have enough money

5.


interaction Suspend or delete a stadium
administrator: Actor1
gui
Complaints
playgrounds
Inactive stadiums
messages playgroundOwner
1 : seartch playground Complaints
2 : seartch playground Complaints
3 : verfiy
seq loop seartch about playground
[A stadium rated less than 3 stars]
4 : get the playground
5 : find playground
6 : Pitch suspension
7 : Pitch suspension
8 : Pitch suspension
9 : The stadium was suspended for not meeting the terms of the secretariat
[A stadium rated less than 2 stars]
10 : get the playground
11 : find playground
«destroy»
12 : Delete playground
13 : delete playground
14 : Your stadium was deleted because you did not follow the agreed terms
[defind]
15 : There are no stadiums less than three stars

6.


interaction add play ground and change time
play ground owner: Actor1
gui
Inactive stadiums
administrator: Actor1
my stadiums
set or change hours
1 : add playground
2 : enter informations
3 : add_playground(His name,location,size,available hours,hourly rate,cancellation period)
4 : informations the grounds
5 : play ground informations
6 : verfiy
seq CombinedFragment1
[accept informations]
7 : add stadiums
8 : accsept stadiums
9 : accsept stadiums
10 : Your stadium was not accepted
11 : Your stadium was not accepted
12 : Adjust playing hours
13 : edit_hour(hour)
14 : edit_hour(hour)
seq CombinedFragment2
[accept]
15 :
16 : accept_new hours
17 : accept_new hours
[not accepted There are people playing at this time]
18 : not accepted There are people playing at this time old time
19 :
20 : not accepted There are people playing at this time old time

# Software Design Specification

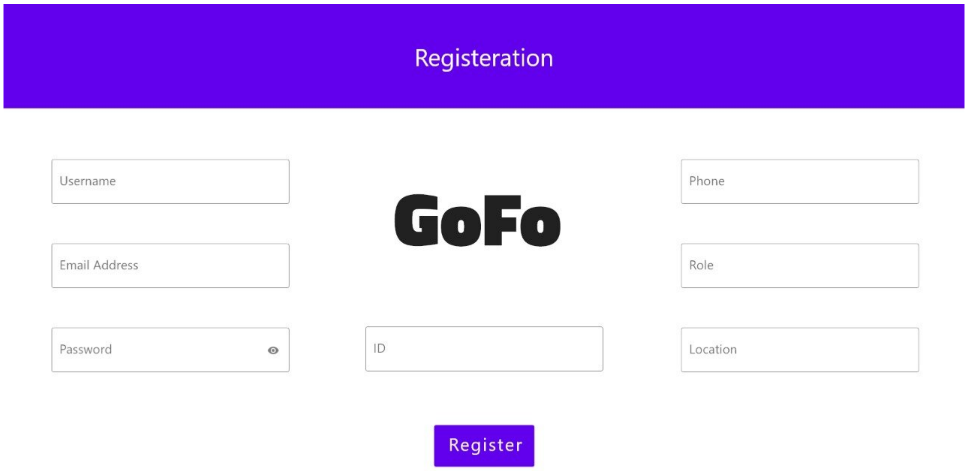
## Class - Sequence Usage Table

| All Methods Used | Classes Used | Sequence Diagram |
|---|---|---|
| 1.Regestir():void<br>2.Registration(Name,Id,Password,mail,Location,ph,0):void | 1-Class player<br>2-Class user<br>3-Class administrator<br>4-class<br>5-PlaygroundOwner | Register |
| 1. Read_From_File():void<br>2. print():void | 1-Class player<br>2-Class<br>3-PlaygroundOwner | display playground |
| 1.addPlayGround():void | 1-PlaygroundOwner | Add playground and change time |
| 1. static Creating_Team():void<br>2. print_Team():void | 1-Class Player | Create a team |
| 1.AddMoney(int):void<br>2. transfer_money(int):void | 1-Class<br>2-PlaygroundOwner<br>3-Class player | transfer money |
| 1.static approvePlayground(String Name,String Location,int Size,int PricePerHour,int cancellationPeriod,boolean Status,ArrayList<String>availableHours) :boolean | 1- Administrator | Suspend or delete a stadium |

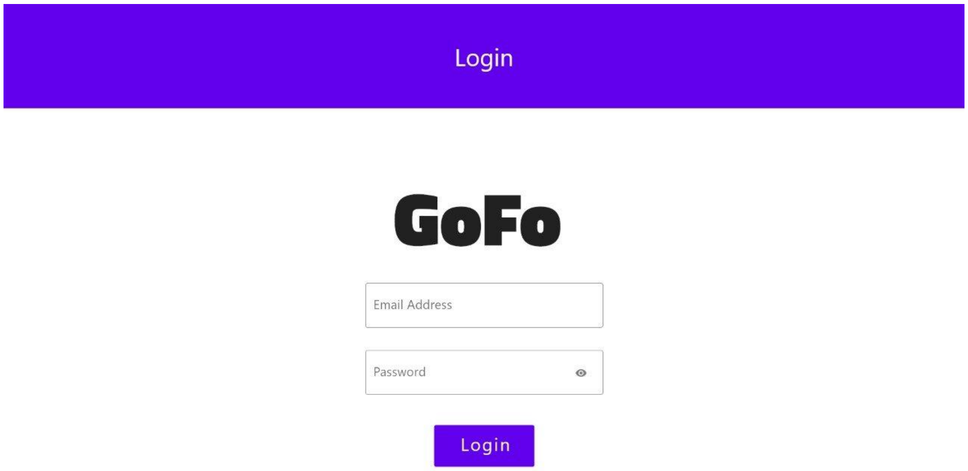
# Software Design Specification

## User Interface Design

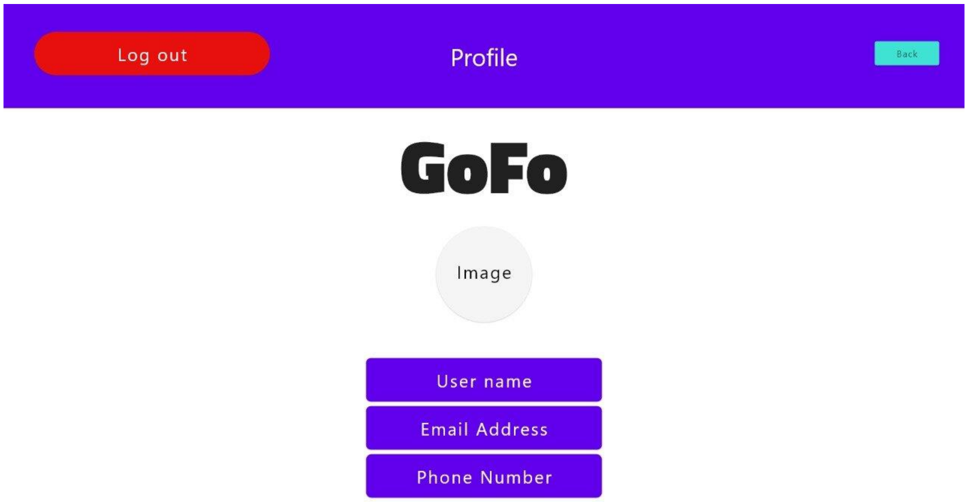
### 1. Registration:



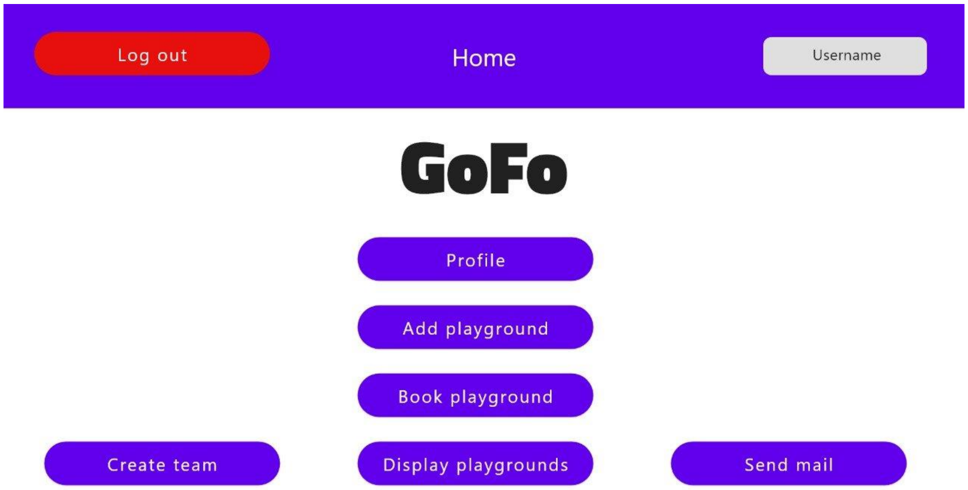
### 2. Login:



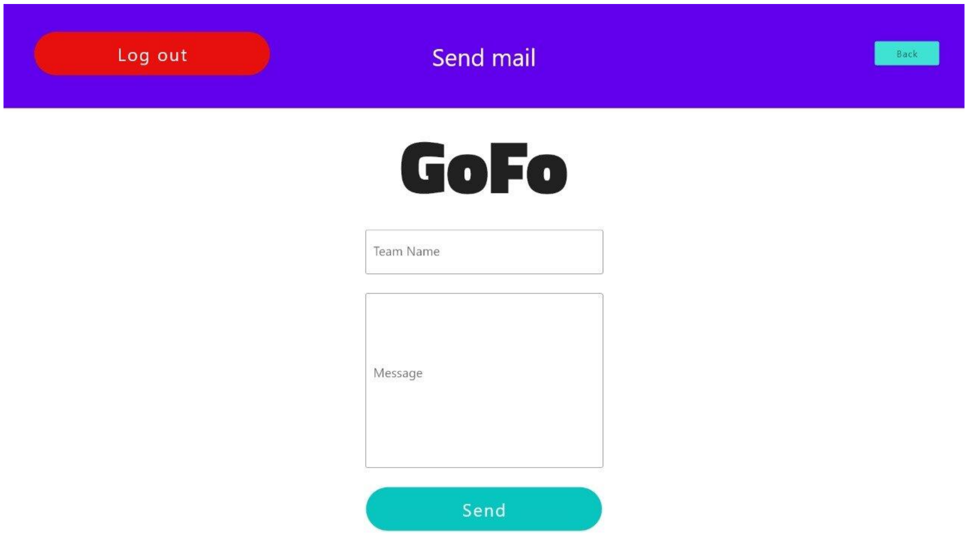
## 3. Profile:



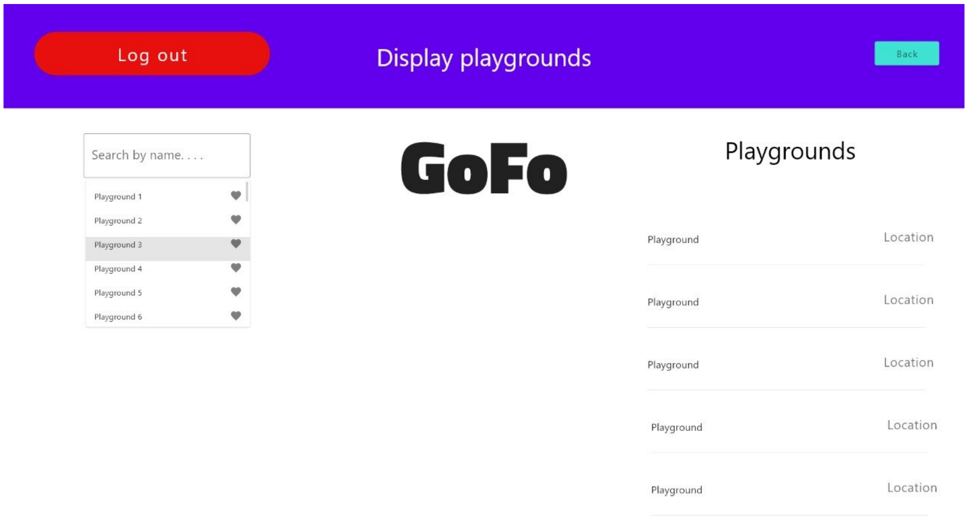
## 4. Home:



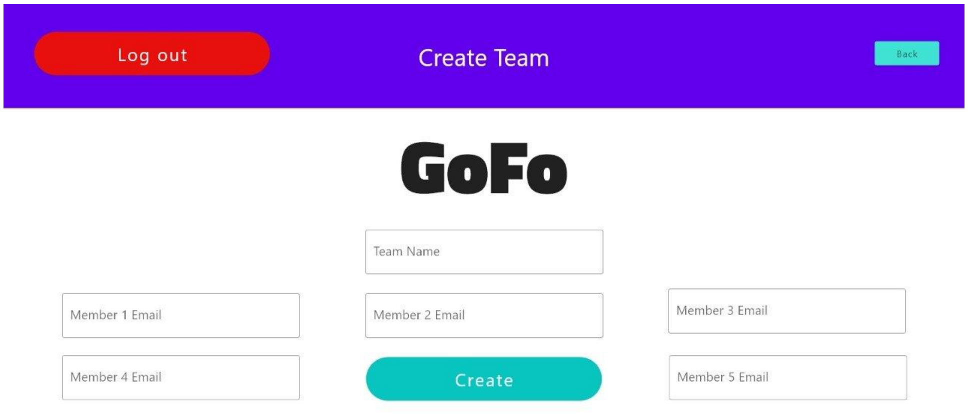
### 5. Send Mail:



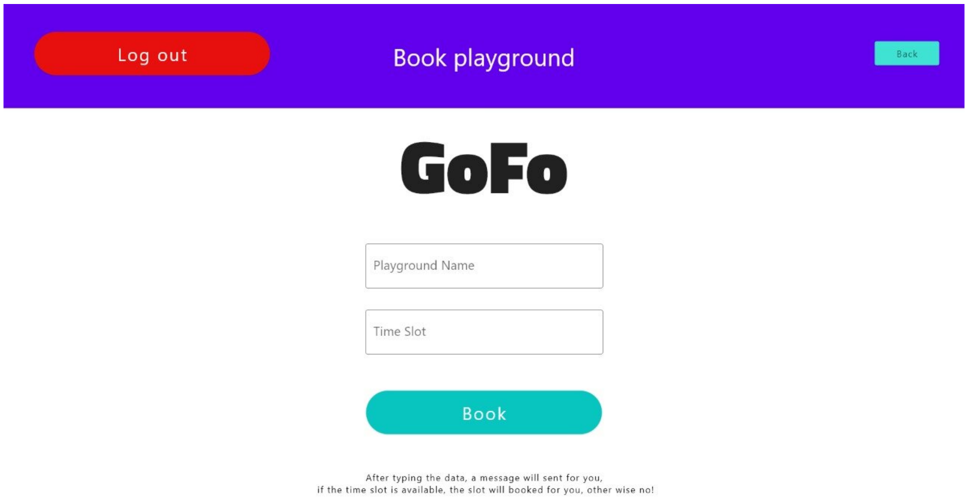
### 6. Display Playgrounds:



## 7. Create team:



## 8. Book Playground:



After typing the data, a message will sent for you,
if the time slot is available, the slot will booked for you, other wise no!

CS251: Phase 2 – < PM-132 >
Project: <GoFo>

# Software Design Specification

| Screen ID | Screen Name | Screen / Wireframe Description |
|---|---|---|
| 1 | registration | Where the user first registers here And if he wants to have another account |
| 2 | login | If he has an account, he can enter his email address and password, and enter his account |
| 3 | profile | The account status is displayed where the image or name can be changed |
| 4 | home | The user can see all the previous options and go smoothly other than the first two |
| 5 | Send email | The user can send messages to a group or friends by setting the friend's name |
| 6 | Display playgrounds | The user can view the stadium, its location and book it if desired |
| 7 | Create team | A user can add five of his friends to the team and a suit will be sent to them via e-mail and name of team |
| 8 | Book playground | Through it you can enter the name of the stadium and book it |

## Tools

**StarUML**: We solved the class diagram and sequence diagrams by Staruml.

**Adobe XD**: We solved the user Interface Design by Adobe XD.

**Eclipse:** Code. And to communicate with version control system.

## Ownership Report

| Item | Owners |
|---|---|
| User Interface Design and Hosting (GitHub and Google drive) and Documentation (JavaDoc.). | Tawfik Yasser Tawfik |
| Part of Development and Screenshots | Abd el-rhman Mohamed Ramadan |
| Part of class diagram and sequence diagrams and Final file order. | Abdulaziz Jmal mousa |

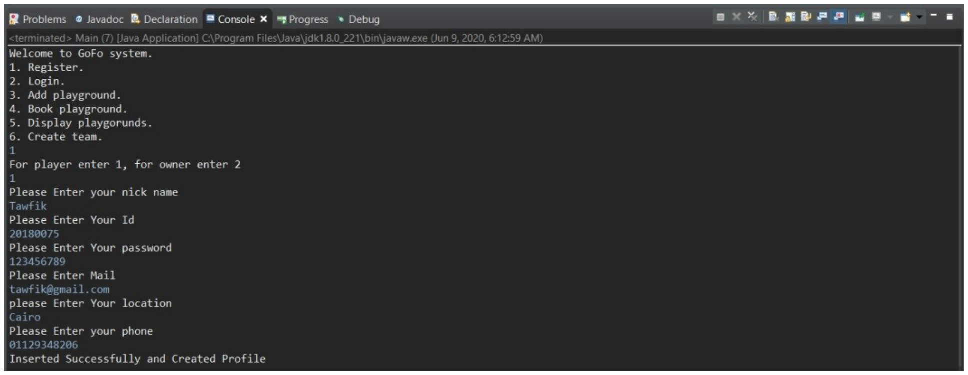
# Software Design Specification

## References

| Source Link | Source Name |
|---|---|
| http://staruml.io/ | Staruml |
| https://www.adobe.com/mena_ar/products/xd.html | Adobe XD |
| https://github.com/TawfikYasser/CS251FinalProjec | GitHub |
| https://drive.google.com/file/d/1IE5JFexhJ0941O2R434UUBo37-lL_vJJ/view?usp=sharing | Code on google drive |
| https://drive.google.com/file/d/1LVHZk-Ymc2wBkVlctzJDPojPu4ii_Nhq/view?usp=sharing | Javadoc on google drive |

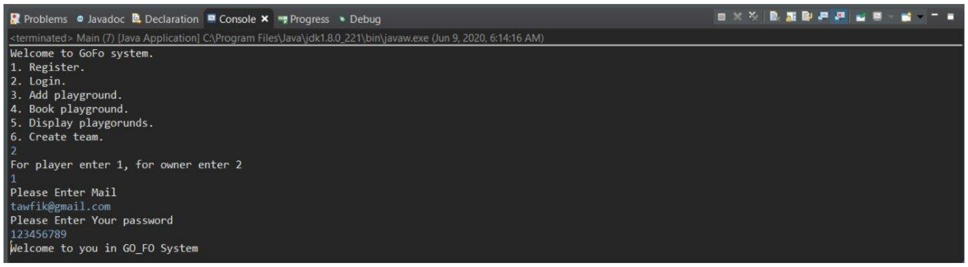
## Screen Snapshots:

### Register:



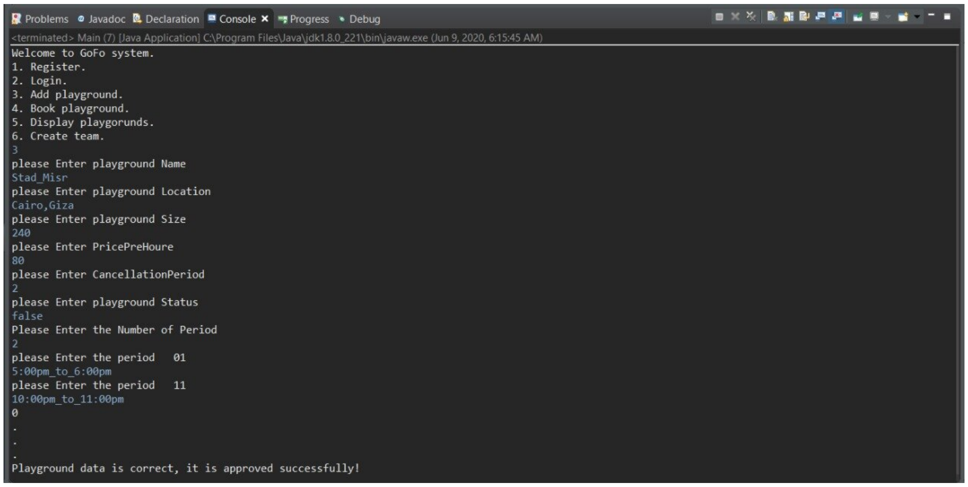
```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:12:59 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
1
For player enter 1, for owner enter 2
1
Please Enter your nick name
Tawfik
Please Enter Your Id
20180075
Please Enter Your password
123456789
Please Enter Mail
tawfik@gmail.com
please Enter Your location
Cairo
Please Enter your phone
01129348206
Inserted Successfully and Created Profile
```

### Login:



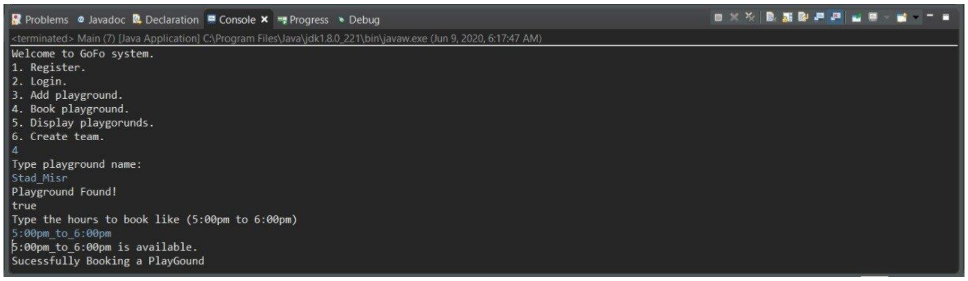
```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:14:16 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
2
For player enter 1, for owner enter 2
1
Please Enter Mail
tawfik@gmail.com
Please Enter Your password
123456789
Welcome to you in GO_FO System
```

## Add playground:



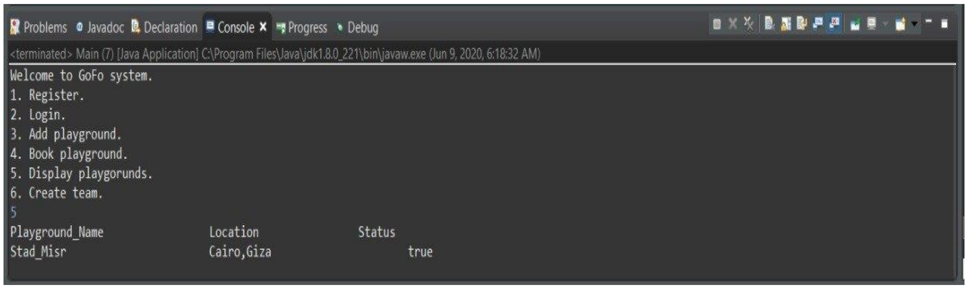
```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:15:45 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
3
please Enter playground Name
Stad_Misr
please Enter playground Location
Cairo,Giza
please Enter playground Size
240
please Enter PricePreHoure
80
please Enter CancellationPeriod
2
please Enter playground Status
false
Please Enter the Number of Period
2
please Enter the period   01
5:00pm_to_6:00pm
please Enter the period   11
10:00pm_to_11:00pm
0
.
.
.
Playground data is correct, it is approved successfully!
```

## Book playground:



```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:17:47 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
4
Type playground name:
Stad_Misr
Playground Found!
true
Type the hours to book like (5:00pm to 6:00pm)
5:00pm_to_6:00pm
5:00pm_to_6:00pm is available.
Sucessfully Booking a PlayGound
```

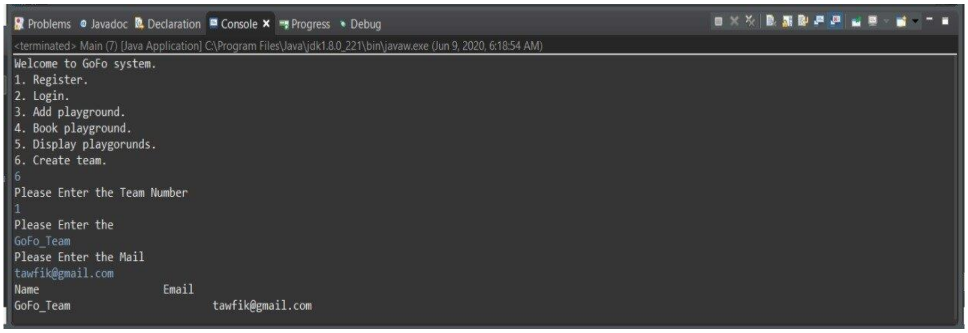
## Display playgrounds:



```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:18:32 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
5
Playground_Name                 Location                Status
Stad_Misr                       Cairo,Giza                       true
```

# Software Design Specification

## Create team with name and email:



```
<terminated> Main (7) [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (Jun 9, 2020, 6:18:54 AM)
Welcome to GoFo system.
1. Register.
2. Login.
3. Add playground.
4. Book playground.
5. Display playgorunds.
6. Create team.
6
Please Enter the Team Number
1
Please Enter the
GoFo_Team
Please Enter the Mail
tawfik@gmail.com
Name                        Email
GoFo_Team                           tawfik@gmail.com
```

## Code Listing:

## Class main:-

```
import java.io.IOException;

import java.util.ArrayList;

import java.util.Scanner;



public class Main {

    static Scanner scanner = new Scanner(System.in);

    static Player object = new Player();

    static PlayGround_Owner object2 = new PlayGround_Owner();

    static Playground object3 = new Playground();



    public static void main(String[] args) {

        // TODO Auto-generated method stub

        start();
```

```
}

private static void start() {
    // TODO Auto-generated method stub

    System.out.println("Welcome to GoFo system.");
    System.out.println("1. Register.");
    System.out.println("2. Login.");
    System.out.println("3. Add playground.");
    System.out.println("4. Book playground.");
    System.out.println("5. Display playgorunds.");
    System.out.println("6. Create team.");

    int choise = scanner.nextInt();
    choosed(choise);
}

private static void choosed(int choise) {
    // TODO Auto-generated method stub

    switch (choise) {
    case 1:
        // register
        System.out.println("For player enter 1, for owner enter 2");
        int rp = scanner.nextInt();
        if (rp == 1) {
            object.Register();

        } else if (rp == 2) {
            object2.Register();
        } else {
            start();
        }
        break;

    case 2:
        // login
        System.out.println("For player enter 1, for owner enter 2");
        int lp = scanner.nextInt();
        if (lp == 1) {
            object.login();
        } else if (lp == 2) {
            object2.login();
        } else {
            start();
        }
        break;

    case 3:
        // add playground
        object3.addPlayGround();
        break;

    case 4:
        // booking
        object3.Read_From_File();
        object.Booking();
        break;

    case 5:
        // display playgrounds
        object3.Read_From_File();
        object3.print();
        break;

    case 6:
        // create team
        object.Creating_Team();
```

```
            object.print_Team();

            break;

        }

    }

}
```

===================================================================================

## Class Administrator:-

```
import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

public class Administrator {

    public void open_file() throws IOException

    {

        File My_object=new File("PlayGround.txt");

        if(My_object.createNewFile())

        {

            System.out.println("Created");

            System.out.println(My_object.getName());

        }

        else
```

```
                {

                   System.out.println("there are some problem");

                }

             }

             public static boolean approvePlayground(String Name,String Location,int Size,int
PricePerHour,int cancellationPeriod,boolean Status,ArrayList<String>availableHours) throws
IOException {

                // FileWriter My_Writer= new FileWriter("PlayGround.txt",true);


if(!Name.isEmpty()&&!Location.isEmpty()&&Size!=0&&PricePerHour!=0&&cancellationPeriod!=0&&S
tatus==false&&availableHours.size()+1>0) {

                    System.out.println(".");

                    System.out.println(".");

                    System.out.println(".");

                    System.out.println("Playground data is correct, it is approved successfully!");

                    return true;

                }else {

                    return false;

                }

             }

          }
```

# Software Design Specification

## Class player:-

```java
import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Scanner;

public class Player extends User{

    static ArrayList<String>TeamMails=new ArrayList<String>();

    static ArrayList<String>TeamNames=new ArrayList<String>();

    public void Register(){

        Scanner Scan=new Scanner(System.in);

        System.out.println("Please Enter your nick name");

        String Name;

        Name=Scan.next();

        System.out.println("Please Enter Your Id");

        int Id=Scan.nextInt();

        System.out.println("Please Enter Your password");

        String Password=Scan.next();

        System.out.println("Please Enter Mail");

        String mail=Scan.next();

        System.out.println("please Enter Your location");

        String Location=Scan.next();
```

# Software Design Specification

```
    System.out.println("Please Enter your phone");

    String ph=Scan.next();

    Registration(Name,Id,Password,mail,Location,ph,0);

    Scan.close();

}


public void login()

{

    Scanner Scan=new Scanner(System.in);

    System.out.println("Please Enter Mail");

    String mail=Scan.next();

    System.out.println("Please Enter Your password");

    String Password=Scan.next();

    login(mail,Password);

    Scan.close();

}

Playground object=new Playground();

public void Booking()

{

    System.out.println("Type playground name: ");

    Scanner scanner = new Scanner(System.in);

    String name = scanner.next();

    for(int i=0;i<Playground.name.size();i++) {
            if(Playground.name.get(i).equals(name)) {
              System.out.println("Playground Found!");
              System.out.println(Playground.status.get(i));
                if(Playground.status.get(i)) {
                    System.out.println("Type the hours to book like (5:00pm to 6:00pm)");
                    Scanner scanner2 = new Scanner(System.in);
                    String hour = scanner2.nextLine();
                    for(int j =0;j<Playground.available_hours.size();j++) {
                      if(Playground.check(hour)) {
                        System.out.println(hour + " is available.");
                        FileWriter My_Writer = null;
                        try {
                          My_Writer = new FileWriter("Booking.txt",true);
                          My_Writer.write("Current User booked: "+Playground.name.get(i)+"
"+hour+" ");
                          My_Writer.write("\n");
                          My_Writer.close();
                          System.out.print("Sucessfully Booking a PlayGound");
                          break;
                        } catch (IOException e1) {
                          e1.printStackTrace();
                          break;
                        }
```

```
                                //break;
                            }
                            else {
                            System.out.println(hour + " is not available.");
                            break;
                            }
                        }
                }else {
                    System.out.println("This playground is not available now!");
                    break;
                }
        }else {
            System.out.println("Playground not found!");
        }
        break;
    }
    scanner.close();
}
public static void Creating_Team()
{
    System.out.println("Please Enter the Team Number");
    Scanner Scan=new Scanner(System.in);
    int num=Scan.nextInt();
    for(int i=0;i<num;i++) {
        System.out.println("Please Enter the ");
        TeamNames.add(Scan.next());
        System.out.println("Please Enter the Mail");
        TeamMails.add(Scan.next());
        FileWriter My_Writer = null;
        try {
            My_Writer = new FileWriter("Team&Mails.txt",true);
        } catch (IOException e1) {
            e1.printStackTrace();
        }
        try {
            My_Writer.write(TeamNames.get(i)+"            "+TeamMails.get(i)+"\n");
            My_Writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    Scan.close();
}
public static void print_Team() {
    System.out.println("Name\t\t\t"+"Email\t\t\t");
    for(int i=0;i<TeamMails.size();i++)
```

```
          {
            System.out.println(TeamNames.get(i)+"\t\t\t"+TeamMails.get(i)+"\t\t\t");
          }
        }
      }
```

==========================================================================

### class playground:-

```
      import java.io.File;

      import java.io.FileNotFoundException;

      import java.io.FileReader;

      import java.io.FileWriter;

      import java.io.IOException;

      import java.util.ArrayList;

      import java.util.Scanner;

      public class Playground {


        private String Name,Location;

        private int Size,PricePerHour,cancellationPeriod;

        private boolean Status;

        static int NumberOfPlayGround=0;

         public static ArrayList<String> name = new ArrayList<String>();

         public static ArrayList<String> location = new ArrayList<String>();
```

```
    public static ArrayList<Integer> size = new ArrayList<Integer>();

    public static ArrayList<Integer> price_per_hour = new ArrayList<Integer>();

    public static ArrayList<Integer> cancellation_peroid = new ArrayList<Integer>();

    public static ArrayList<Boolean> status = new ArrayList<Boolean>();

    public static ArrayList<String>available_hours=new ArrayList<String>();

    public static ArrayList<Integer>PlayNum=new ArrayList<Integer>();

    static int number_Of_PlayGround=0;

    int current=0;

    public Playground()

    {


    }

    public Playground(String name, String location, int size, int pricePerHour, int
cancellationPeriod, boolean status, String[] availableHours) {

        Name = name;

        Location = location;

        Size = size;

        PricePerHour = pricePerHour;

        this.cancellationPeriod = cancellationPeriod;

        Status = status;

        //this.availableHours = availableHours;

    }

    public void open_file() throws IOException
```

```
{
    File My_object=new File("PlayGround.txt");
    if(My_object.createNewFile())
    {
        System.out.println("Created");
        System.out.println(My_object.getName());
    }
    else
    {
        System.out.println("there are some problem");
    }
}
public void Read_From_File()
{
    File MyObjec=new File("PlayGround.txt");
    try {
        Scanner myRead = new Scanner( MyObjec);
        while (myRead.hasNextLine()) {
            String data = myRead.next();
            name.add(data);
            data=myRead.next();
            location.add(data);
            int Digi=Integer.parseInt(myRead.next());
```

```
          size.add(Digi);
          Digi=Integer.parseInt(myRead.next());
          price_per_hour.add(Digi);
          Digi=Integer.parseInt(myRead.next());
          cancellation_peroid.add(Digi);
          Boolean Stat=Boolean.parseBoolean(myRead.next());
          status.add(Stat);
           //int i=0;
          myRead.nextLine();
           available_hours.add(myRead.next());
           NumberOfPlayGround++;
           myRead.nextLine();
          //System.out.println(data);
         }
         myRead.close();
      } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
      }
    }
    public static boolean check(String a)
    {
      if(available_hours.contains(a))
    {
        return true;
    }
    return false;

}
public void print()
{
    System.out.println("Playground_Name"+"\t\t\t"+"Location"+"\t\t"+"Status");
    for(int i=0;i<name.size();i++)
    {
        System.out.print(name.get(i)+"\t\t\t");
        System.out.print(location.get(i)+"\t\t\t");
        System.out.println(status.get(i));

    }
}
public void addPlayGround()
{
    int num=0;
    try (Scanner Scan = new Scanner(System.in)) {
        System.out.println("please Enter playground Name");
        this.Name=Scan.next();
        System.out.println("please Enter playground Location");

        this.Location=Scan.next();

        System.out.println("please Enter playground Size");

        this.Size=Scan.nextInt();

        System.out.println("please Enter PricePreHoure");

        this.PricePerHour=Scan.nextInt();

        System.out.println("please Enter CancellationPeriod");

        this.cancellationPeriod=Scan.nextInt();

        System.out.println("please Enter playground Status");

        this.Status=Scan.nextBoolean();

        System.out.println("Please Enter the Number of Period");

        num=Scan.nextInt();

        PlayNum.add(num);

        for(int i=0;i<num;i++)

        {

           System.out.println("please Enter the period "+ "  "+ i+1);

           String Data;

           Data=Scan.next();

           available_hours.add(Data);

        }

        System.out.println(number_Of_PlayGround);

    FileWriter My_Writer = null;

    try {
        My_Writer = new FileWriter("PlayGround.txt",true);


      } catch (IOException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
      }
      try {
        Administrator Admin=new Administrator();
        if(Admin.approvePlayground(Name, Location, Size, PricePerHour, cancellationPeriod,
Status, available_hours))
    {
      this.Status=true;
      My_Writer.write(this.Name+" "+this.Location+" "+this.Size+" "+this.PricePerHour+"
"+this.cancellationPeriod+" "+this.Status+"\n");
      for(int i=0;i<num;i++)
      {
        My_Writer.write(available_hours.get(i)+" ");
      }
      My_Writer.write("\n");
      number_Of_PlayGround++;
    }
    else
    {
            System.out.println("Admin Refused This approvement");

            System.out.println("you should Enter the Correct info");

        }


        /*

        for(int i=0;i<num;i++)

        {

            My_Writer.write(available_hours[NumberOfPlayGround][i]+"  ");

        }

        My_Writer.write("\n");

        */

    } catch (IOException e) {

        // TODO Auto-generated catch block

        e.printStackTrace();

    }

    try {

        My_Writer.close();

    } catch (IOException e) {

        // TODO Auto-generated catch block

        e.printStackTrace();

    }

    NumberOfPlayGround++;

}
```

```
}

}
```

================================================================================

## class playground_Owner :-

```
import java.io.IOException;

import java.util.Scanner;

public class PlayGround_Owner extends User{

  static public Administrator administrator = new Administrator();

  public PlayGround_Owner()

  {

  }

  public void Register(){

    Scanner Scan=new Scanner(System.in);

    System.out.println("Please Enter your nick name");

    String Name;

    Name=Scan.next();

    System.out.println("Please Enter Your Id");

    int Id=Scan.nextInt();

    System.out.println("Please Enter Your password");

    String Password=Scan.next();

    System.out.println("Please Enter Mail");

    String mail=Scan.next();
```

```
        System.out.println("please Enter Your location");

        String Location=Scan.next();

        System.out.println("Please Enter your phone");

        String ph=Scan.next();

        Registration(Name,Id,Password,mail,Location,ph,1);

    }

    public void login()

    {

        Scanner Scan=new Scanner(System.in);

        System.out.println("Please Enter Mail");

        String mail=Scan.next();

        System.out.println("Please Enter Your password");

        String Password=Scan.next();

        login(mail,Password);

    }

}
```

==================================================================

## class User:-

```
import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;
import java.util.Scanner;

public class User {

    public String Name;

    private int ID;

    private String Password;

    private String Email;

    private String Location;

    private String Phone;

    private String Role;

    private ArrayList<String> Names=new ArrayList<String>();

    private ArrayList<String> Emails=new ArrayList<String>();

    private ArrayList<String> Passwords=new ArrayList<String>();

    ArrayList<Integer> ids = new ArrayList<>();

    static int numberOfUsers=0;

    public User()

    {


    }

    // this method used to open the file

    public void open_file() throws IOException

    {

        File My_object=new File("DataBase2.txt");

        if(My_object.createNewFile())
        {
            System.out.println("Created");
            System.out.println(My_object.getName());
        }
        else
        {
            System.out.println("there are some problem");
        }
    }

    public void Read_From_File()
    {
        File MyObject=new File("DataBase.txt");
        try {
            Scanner myReader = new Scanner( MyObject);
            while (myReader.hasNextLine()) {
                String data = myReader.next();
                Names.add(data);
                int data1;
                data1=Integer.parseInt(myReader.next());
                ids.add(data1);
                data=myReader.next();
                Passwords.add(data);
                data=myReader.next();
                Emails.add(data);
                data=myReader.nextLine();
                numberOfUsers++;
              }
              myReader.close();
        } catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
    public void print()
    {
        for(int i=0;i<Names.size();i++)
        {
            System.out.println(Names.get(i));
        }
    }

    protected void login(String Email,String Password)
    {
        Read_From_File();
        if(Emails.contains(Email))
```

```
{
    int index=Emails.indexOf(Email);
    if(Passwords.get(index).equals(Password))
    {
        System.out.print("Welcome to you in GO_FO System");
    }
    else
    {
        Scanner Scan=new Scanner(System.in);
        System.out.print("invalid authentication");
        int For_Pass=0;
        System.out.println("if you forget the password Enter 1");
        For_Pass=Scan.nextInt();
        if(For_Pass==1)
        {
            System.out.println("Please Enter Your Id");
            int Id=Scan.nextInt();
            if(Id==ids.get(index))
            {
                System.out.println("good job");
                System.out.println("Your Mail is:  "+Email);
                System.out.println("Your Passords is:  " +Passwords.get(index));
                System.out.println("Please Enter 1 if you want to change the password");
```

```
                    int change=Scan.nextInt();
                    if(change==1)
                    {
                        System.out.println("Please Enter your New Passwords");
                        String New_Password=Scan.next();
                        Passwords.add(index, New_Password);
                        System.out.println("good your Password is chagned");
                    }
                }

            }

        }
    }
    else
    {
        System.out.println("this mail is not exist");
        System.out.println("You should to Register First");
    }
}
protected void Registration(String Name,int Id,String Password,String mail,String loac,String phone,int num)
{
```

```
try {
  FileWriter My_Writer=new FileWriter("DataBase.txt",true);
  // Regular exp phone
  // that check that the phone number must have zero at the First
  // and contain 10 numbers from 0 to 9
  String regex1="^[0]+[0-9]{10}";
  // this is Regular Exp Mail
  // that check that the mail should have character from a(A) to z(Z) and number from 0 to 9
  // and . and - and must contain @
  String regex="^[A-Za-z0-9+_.-]+@[A-Za-z0-9.-]+$";
  // and if the both condirion is corrected then store user_info in DataBase_file
  if(phone.matches(regex1) && mail.matches(regex) && !Emails.contains(mail))
  {
    this.Name=Name;
    this.ID=Id;
    this.Password=Password;
    this.Location=loac;
    this.Phone=phone;
    this.Email=mail;
    // num this num if the num=0
    // the Role is 'Player'
    if(num==0)
```

```
        {
           this.Role="Player";
        }
        // else the num=1
        //the Role is 'PlayGround_owner'
        else
           this.Role="Owner";
        My_Writer.write(Name+" "+Id+" "+Password+" "+mail+" "+loac+" "+phone+" "+Role+" "+"\n");
        My_Writer.close();
        // store all names into Array_List
        Names.add(Name);
        // store all Mails into Array_List(Emails)
        Emails.add(mail);
        // store all Passwords into Array_List(Passwords)
        Passwords.add(Password);
        ids.add(Id);
        // i think this this is trivial Variable
        numberOfUsers++;
        // Profile is Created Profile data  Saved in DataBase File
        System.out.println("Inserted Successfully and Created Profile");
    }
    else
```

```
            {
                System.out.println("Invalid in Mail oR Phone number");
                System.out.println("Or this is already exits");
                My_Writer.close();
                System.out.println("Please Register again");
            }
        }
        catch (IOException e) {
            e.printStackTrace();
        }

    }
    public void Print() throws IOException {
        for(int i=0;i<numberOfUsers;i++) {
            System.out.println(Emails.get(i).toString());
    }
    }
}
```

# Software Design Specification

## Links:

### GitHub link:

https://github.com/TawfikYasser/CS251FinalProject

### Google Drive Code Link:

https://drive.google.com/file/d/1IE5JFexhJ0941O2R434UUBo37-lL_vJJ/view?usp=sharing

### Google Drive Javadoc. Link:

https://drive.google.com/file/d/1LVHZk-Ymc2wBkVlctzJDPojPu4ii_Nhq/view?usp=sharing

### Google Drive Video Link:

https://drive.google.com/file/d/13fgm8Q6053ijkdqCYRR5zcudwCCh6nfx/view?usp=sharing

Additional:

### Google Drive UI Video Link:

https://drive.google.com/file/d/1h2PkcacsTVuyvW2uhKT2s6s2yqobNK4X/view?usp=sharing

Thanks.

PS-132