**Cairo University**
**Faculty of Computers and Artificial Intelligence**


# CS251

# Software Engineering I

## GoFo

## Software Design Specifications

## Version 1.0

Paula Adel paulaadelkamal@gmail.com
Andrew Naseif andrewnasief3@gmail.com
Nader Atef Zaki nader_atef80@outlook.com

6/2021

CS251: Phase 2 – <Team Name>
Project: <Project Name>

# Software Design Specification

## Contents

CS251: Phase 2 – <Team Name>
Project: <Project Name>

# Software Design Specification

## Team

| ID | Name | Email | Mobile |
|---|---|---|---|
| 20190139 | Paula Adel Kamal | paulaadelkamal@gmail.com | 01277137930 |
| 20190114 | Andrew Naseif | andrewnasief3@gmail.com | 01111428868 |
| 20190575 | Nader Atef Zaki | Nader_atef80@outlook.com | 01274763696 |

## Document Purpose and Audience

This is a software design specification document (SDS) which contains diagrams and illustration for many software aspects.

The document is intended to be for a project owner who has technical knowledge about computer science in general and specially software engineering field and also know what are the user requirements.

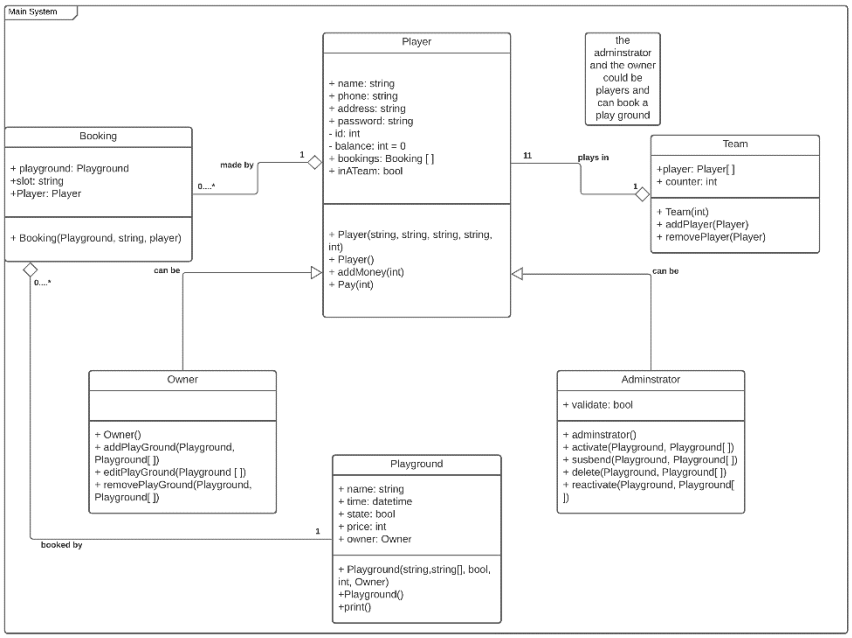
This project contains some UML diagrams as class diagrams, sequence diagrams and state diagrams.

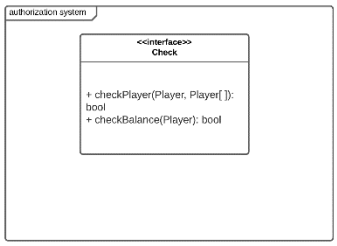
CS251: Phase 2 – <Team Name>
Project: <Project Name>

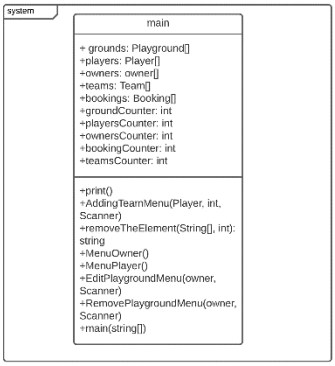
# Software Design Specification

## System Models

### I. Class Diagram(s)



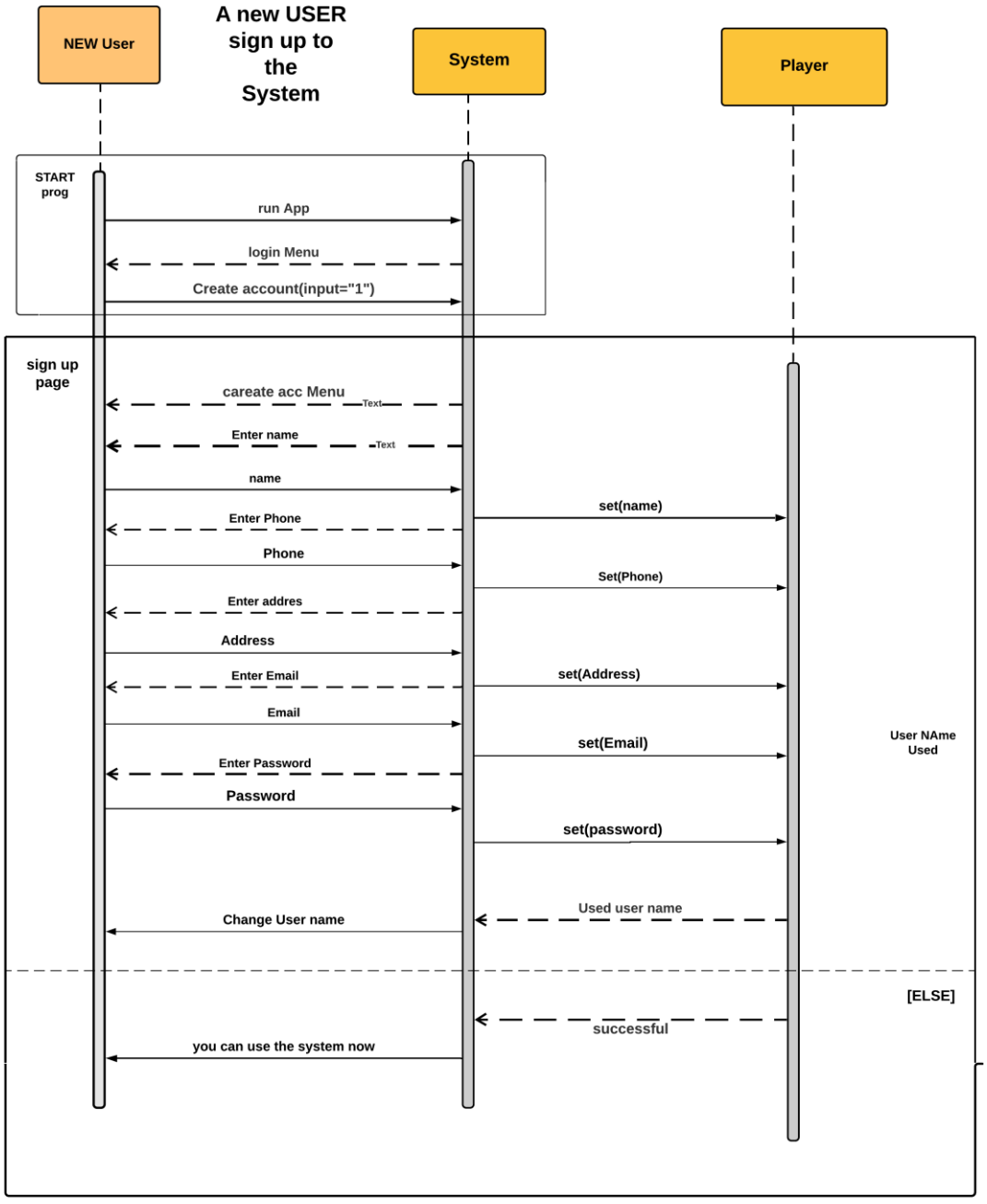
# Software Design Specification

## II. Class Descriptions

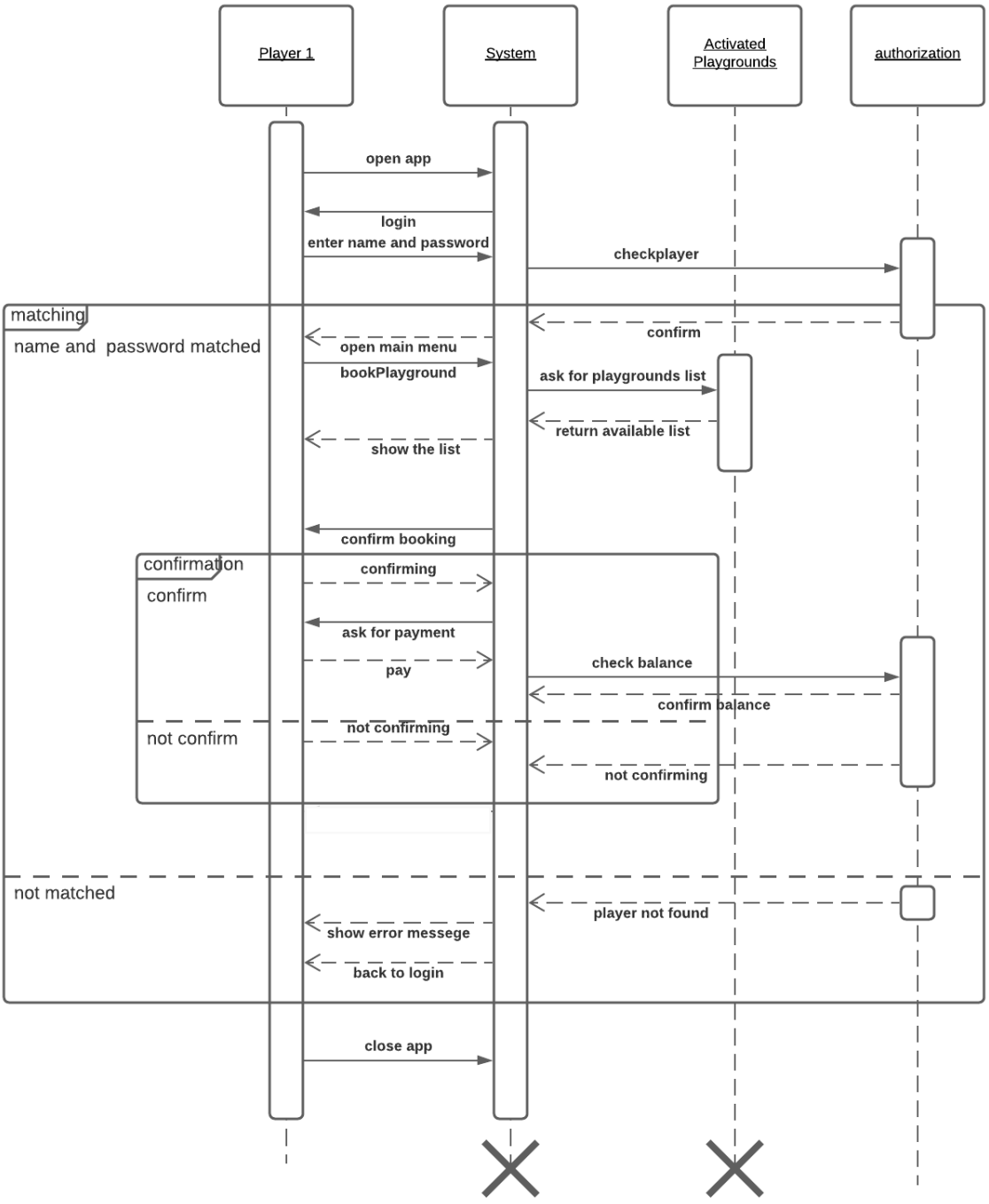
| Class ID | Class Name | Description & Responsibility |
|---|---|---|
| 1. | player | Player is the class responsible for players and their information.<br>It allows us to create player objects and enter its information and create a players array to carry all the players accounts.<br>Player class allow players to add money to their balance and pay for playgrounds. |
| 2. | owner | Owner class inherit from player class so it have all player class attributes and methods and have 3 special methods which are to add playground to the playgrounds array and to edit playground data and to remove the playground. |
| 3. | administrator | Owner class inherit from player class so it have all player class attributes and methods and have 4 special methods which are to activate playground, to suspend a playground, to delete playground and to reactivate it. |
| 4. | playground | Playground class is a class that allows us to make playground objects and playground arrays and it contains playground information as its name, time, state and price. |
| 5. | booking | Booking is the class that allows us to make bookings objects and bookings arrays. |
| 6. | team | Team is class that allows us to make teams of 11 players. |
| 7. | check | Check is an interface that allows to do most of the authorizations in the system |
| 8. | system | System is the class that acts as our main in which the one how use the system interact with and it may doesn't have many functions but it holds many things which are too small to be functions as showing messages take inputs from the user. |

# Software Design Specification

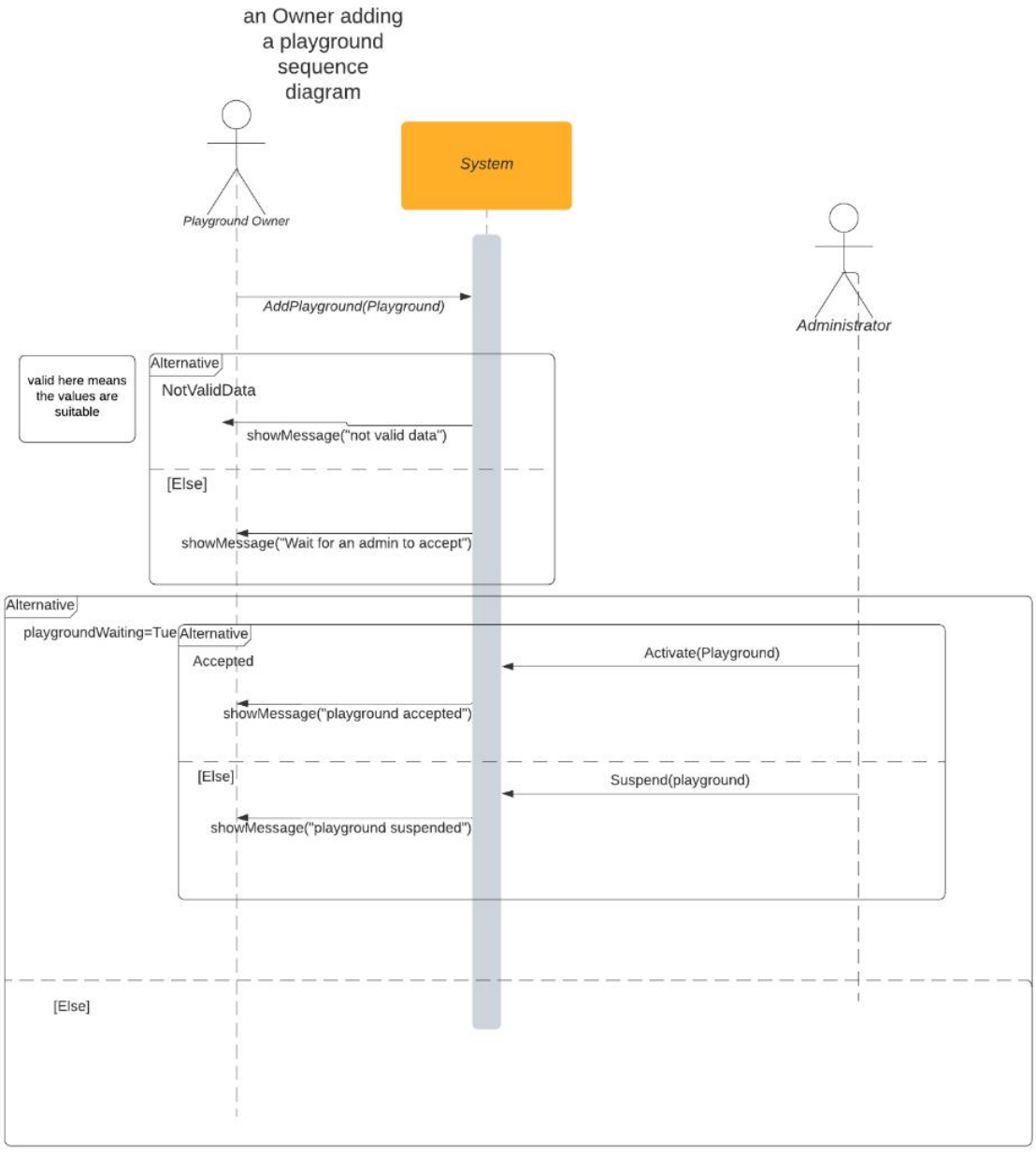
## III. Sequence diagrams



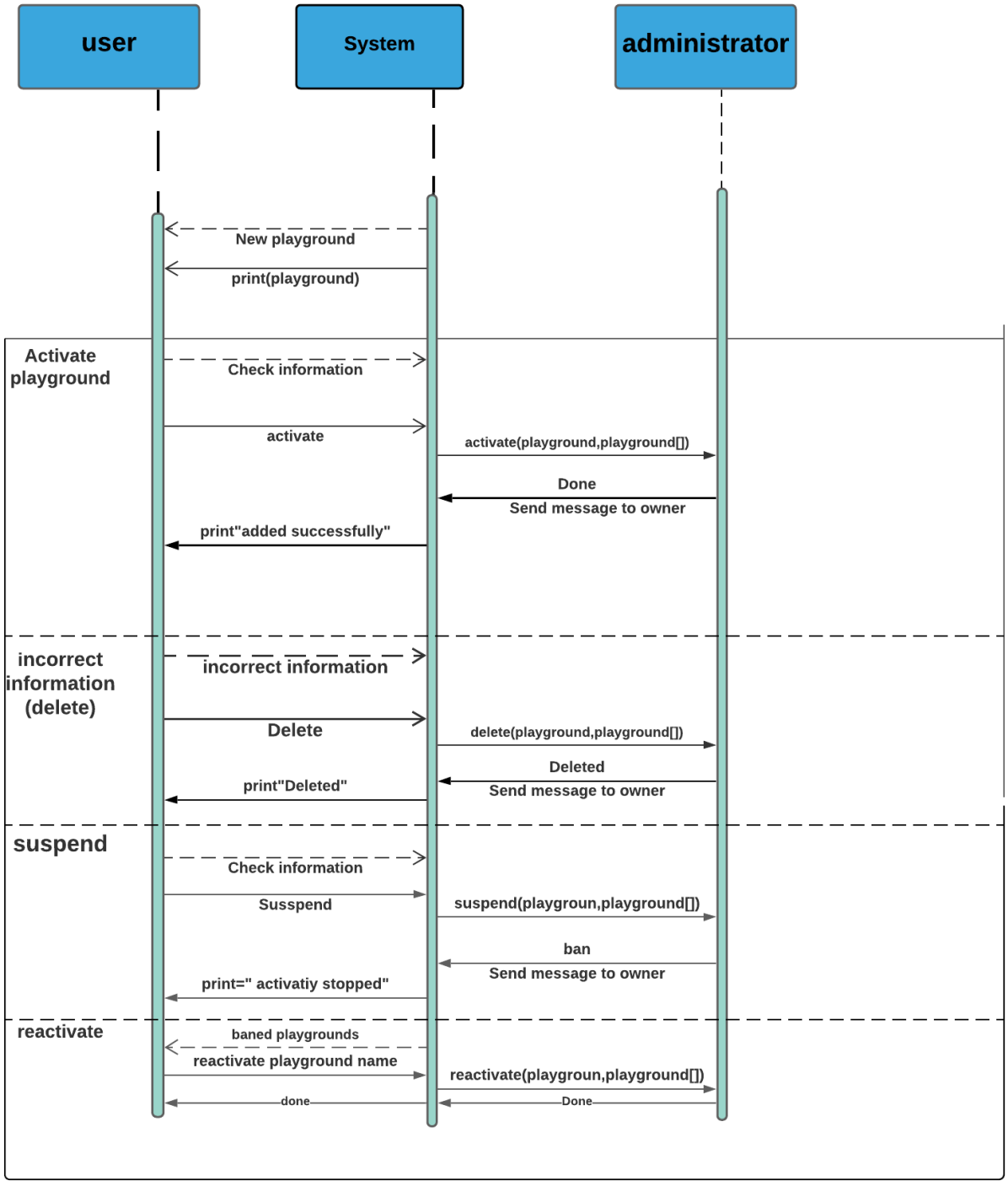
# Software Design Specification



# Software Design Specification



# Software Design Specification



# Software Design Specification

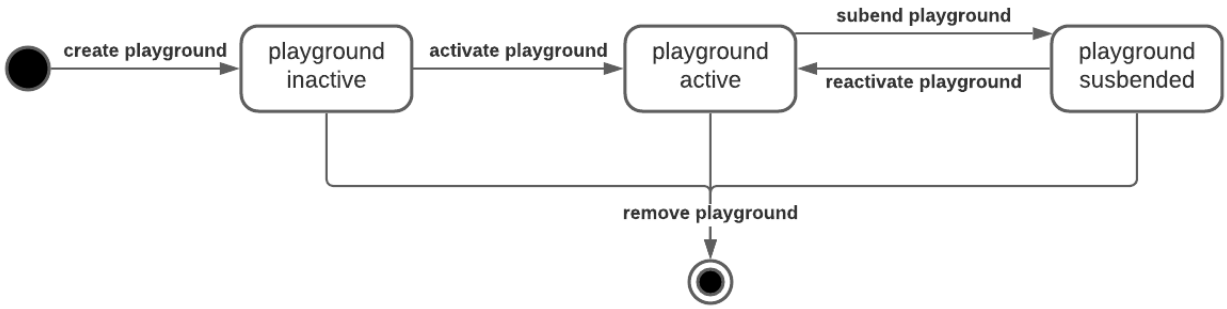
## Class - Sequence Usage Table

- **In this table, we will list the sequence diagrams you drew. For each one, list all the classes used in this sequence. For each class list all the methods you used in this class. Every method or object on a sequence diagram must belong to an existing class in the class diagram and be shown there. If sequence diagrams do not reflect actual classes and methods, they will be REJECTED.**

| Sequence Diagram | Classes Used | All Methods Used |
|---|---|---|
| 1. Sign up to system | Player class<br>System class<br>Check interface | createAccount()<br>setName()<br>setPhone()<br>setAddress()<br>setEmail()<br>setPassword()<br>checkPlayer() |
| 2. Booking a play ground | Player class<br>System class<br>Check interface<br>Playground class | Login()<br>bookPlayground()<br>checkPlayer()<br>checkBalance(player, player[ ])<br>Pay(int) |
| 3. Adding play ground | Owner class<br>Administrator class<br>Owner class<br>system | Addplayground()<br>activatePlayground()<br>susbendPlayground() |
| 4. Administrator's functions | System Class<br>Administrator Class<br>Playground Class | activate(playground,playground[ ])<br>delete(playground,playground[ ])<br>suspend(playground,playground[ ])<br>reactivate(playground,playground[ ]). |

# Software Design Specification

## IV. State Diagram




## Tools

**Lucid chart, Microsoft Word**

## Ownership Report

| owner | items |
|---|---|
| Paula Adel Kamal | **Part pf class diagram and sequence diagram 2 and state diagram** |
| Andrew Nasief | **Part pf class diagram and sequence diagram 1 and sequence diagram 4** |
| Nader Zaki | **Part pf class diagram and sequence diagram 3** |