

Software Design Document

Courts Managment System

V1.0

Hussam Eldin,Youssef Mohamed,Ahmed Mohamed,Omar Anas
Customer: Gamal Nour eldin

May 26, 2019

1 Introduction

1.1 Purpose of this document

The purpose of this document is to provide a full description of how the reservation system works.The reservation system is an online web-based application system to manage reservation of courts to ease the reservation process for players.This software design document (SDD) will describe the aim of the system and it's functionalities.In addition, the document will show all constraints on the system , all contraventional interfaces designs and all diagrams that were needed to build the system.

1.2 Scope of this document

The purpose of the online courts management system is to ease the reservation process and to create a convenient and easy-to-use application for players trying to reserve courts. The system is based on a relational database with its courts management and reservation functions. We will have a database server supporting major cities around Egypt as well as various courts .

1.3 Overview

This document describes most of the system diagrams and architectures. it also previews how the system main functionalities work and how the user views and interacts with the software. The sections in this document gives a detailed description for the diagrams that help the developer developing the system.It includes the class diagrams , sequence diagrams and the 4 architectures diagrams.

1.4 Definitions and Acronyms

Term	Definition
MVC	Model, View , Controller structure
EAV	Entity attribute value model.

2 System Overview

Provides stable courts reservation System based on web application. The project aims to ease the use of reservation of courts to all the people using the system including the Manager , Employee , Accountant, IT and the normal user. The system depends on web so it can be compatible on most of the modern devices. **System Overview Diagram :**

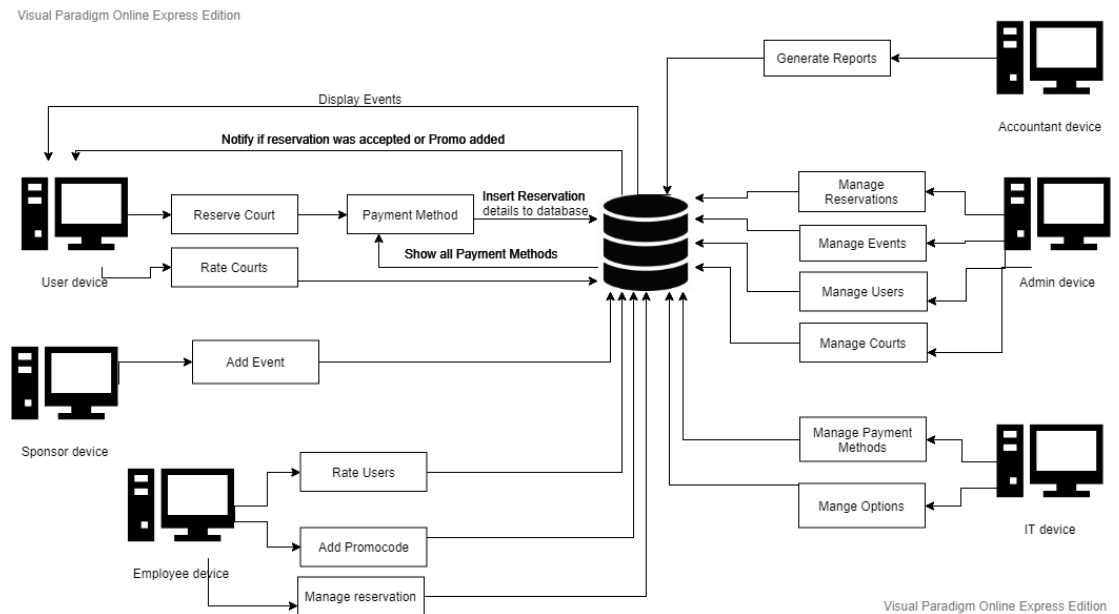


Figure 1: System Overview Diagram

3 System Architecture

3.1 Architectural Design

3.1.1 Logical Diagram

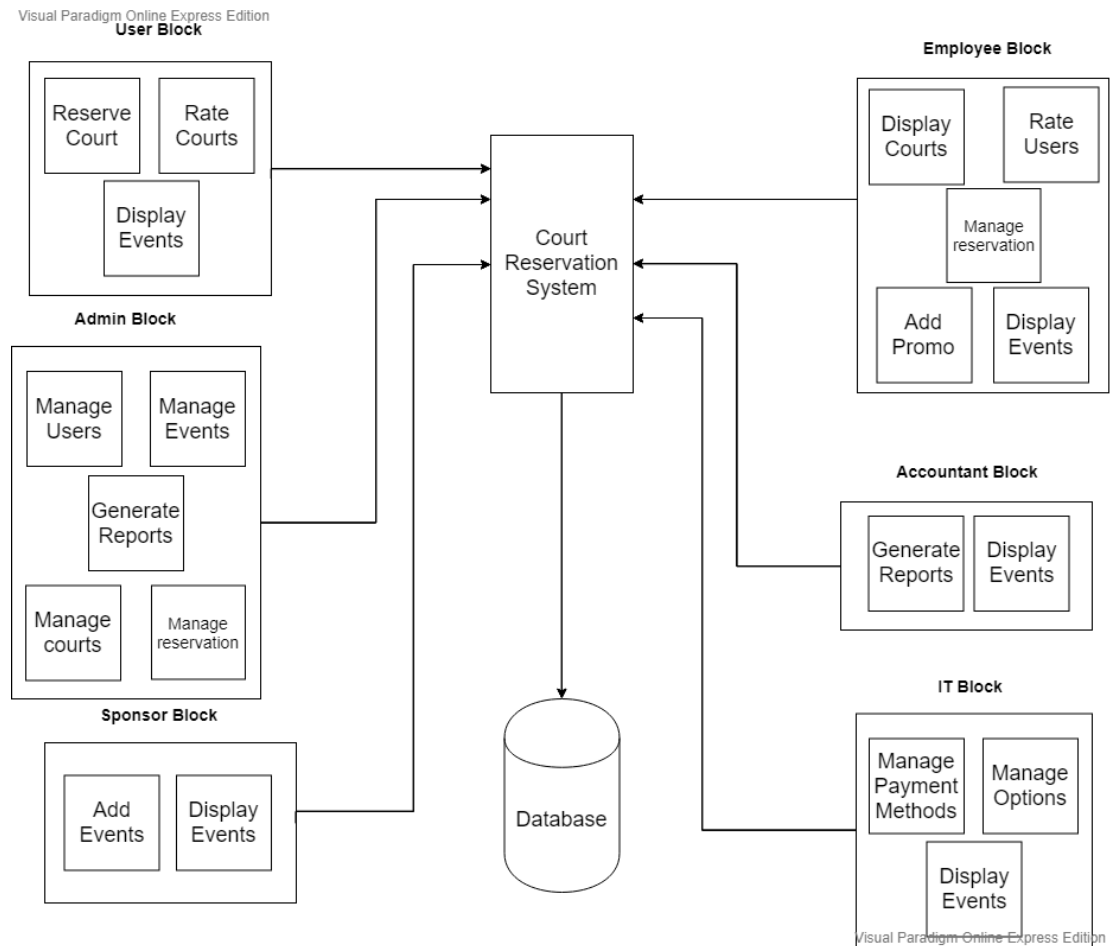


Figure 2: Logical Diagram

The above is a Logical Diagram that separates the system to some blocks, each block has a certain activity to do.

User Block: This is the end-user block where the user can reserve a court and then give it a rating according to his experience with the court.

Employee Block: Responsible for rating users and adding, deleting and updating any new information about reservation of courts. He can add promocode.

Accountant Block: Responsible for generating monthly reports for the reser-

vation system.

IT Block: Responsible for adding any new Payment methods, deleting payment methods and updating any one. Also, Manages the options for the payment methods.

Sponsor Block: Responsible for adding any events in the system.

Admin Block: Responsible for most of the system such as adding , deleting or updating any user , event , courts.

3.1.2 Process Diagram

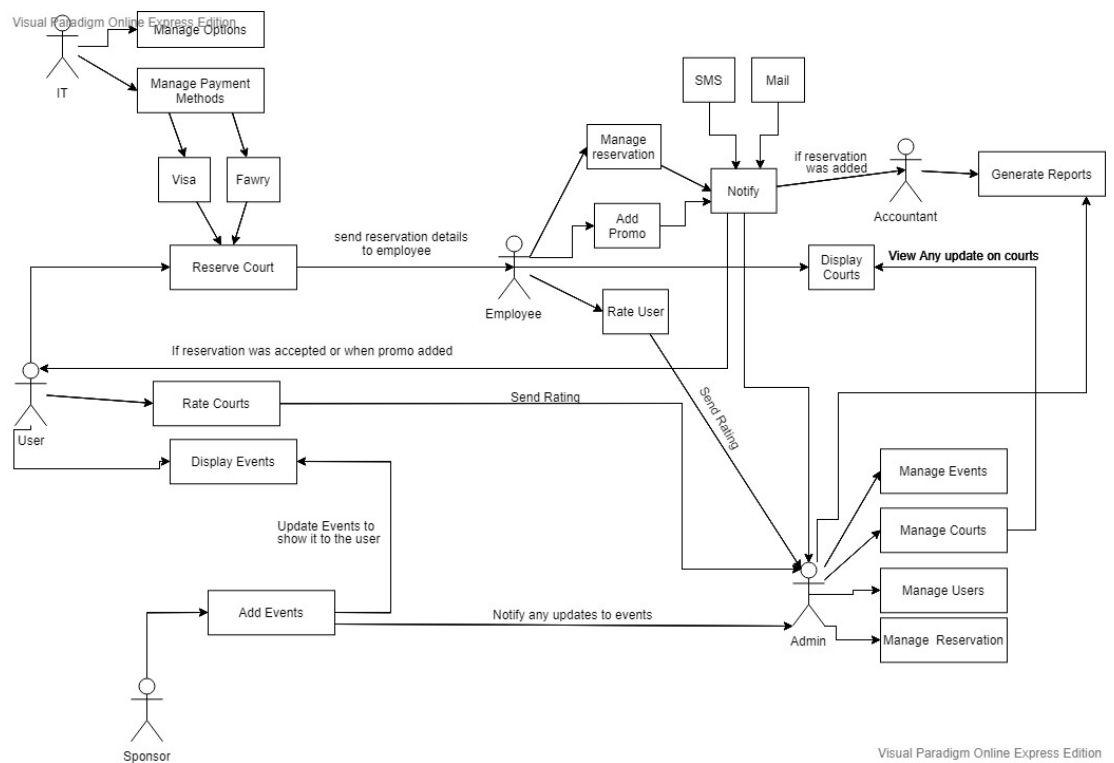


Figure 3: Process Diagram

This Diagram shows the process flow of many actions in the system.

The User can reserve a court , display events or rate courts.

If he chooses to reserve a court he then chooses a court and can pay with either Fawry or Visa which are managed by the IT. The reservation details is then sent to the employee to check whether it's real or fake reservation. The employee then notifies the user whether his reservation was approved or not and notifies the Accountant of this reservation so he can generate some statistics. After the

reservation process is completed the employee gives rating to the user and sends it to the Admin who can Manage users.

If the User chooses to rate courts he chooses a court to rate and this rating is sent to the admin who can Manage courts to improve it.

The sponsor can add, edit or delete events which are then displayed to the user and he notifies the admin about any changes made to the events so the admin can Manage events.

3.1.3 Software Diagram

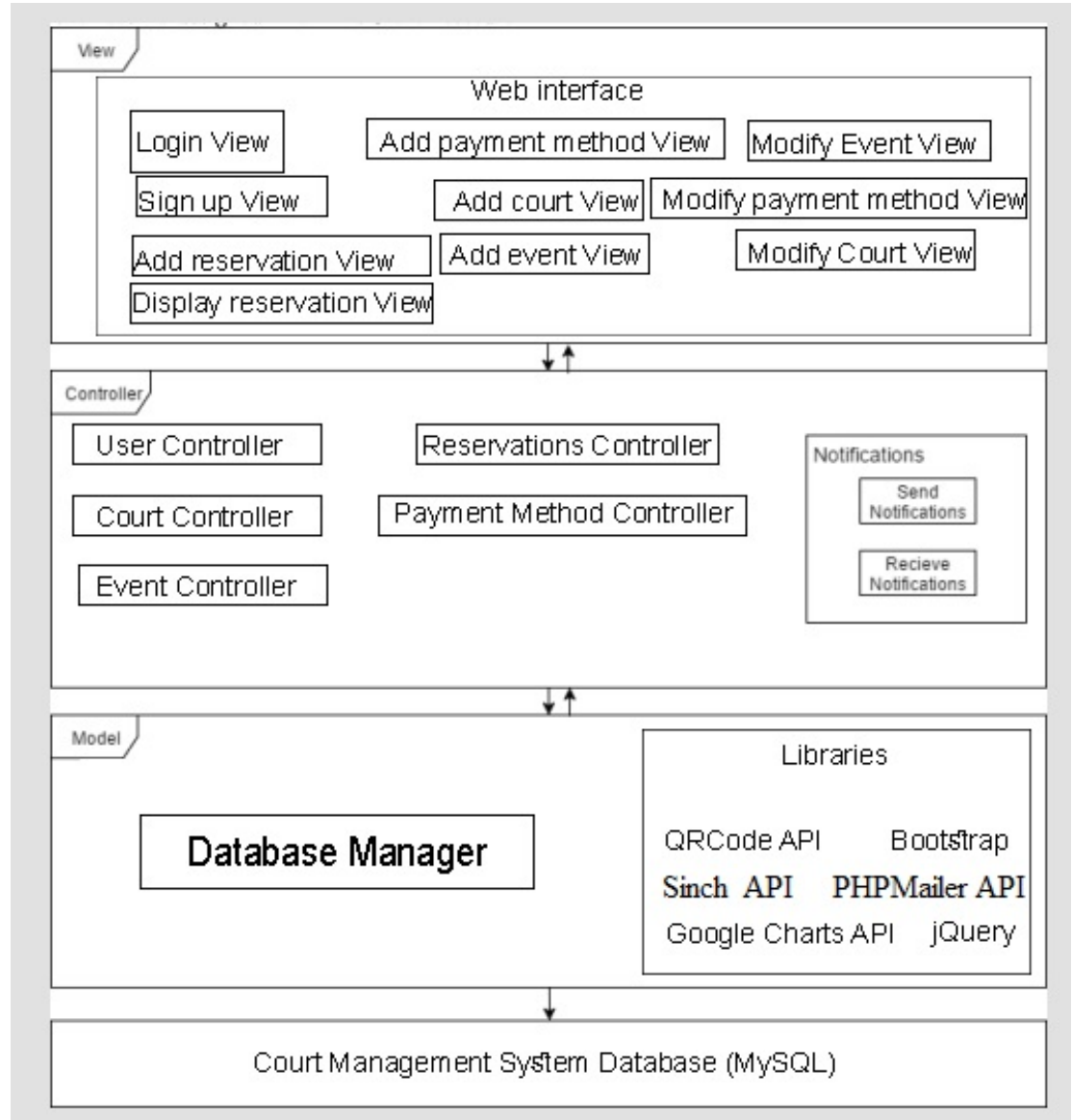


Figure 4: Software Diagram

MVC Architecture design is commonly used to seperate classes into 3 files to reduce testing costs:

Model : Deals with the database and includes all the used queries for a single

class

View : Contains all the user interface design to view it to the user

Controller : Connects the model to the view and contains any validations or conditions in the functions

3.1.4 Hardware Diagram

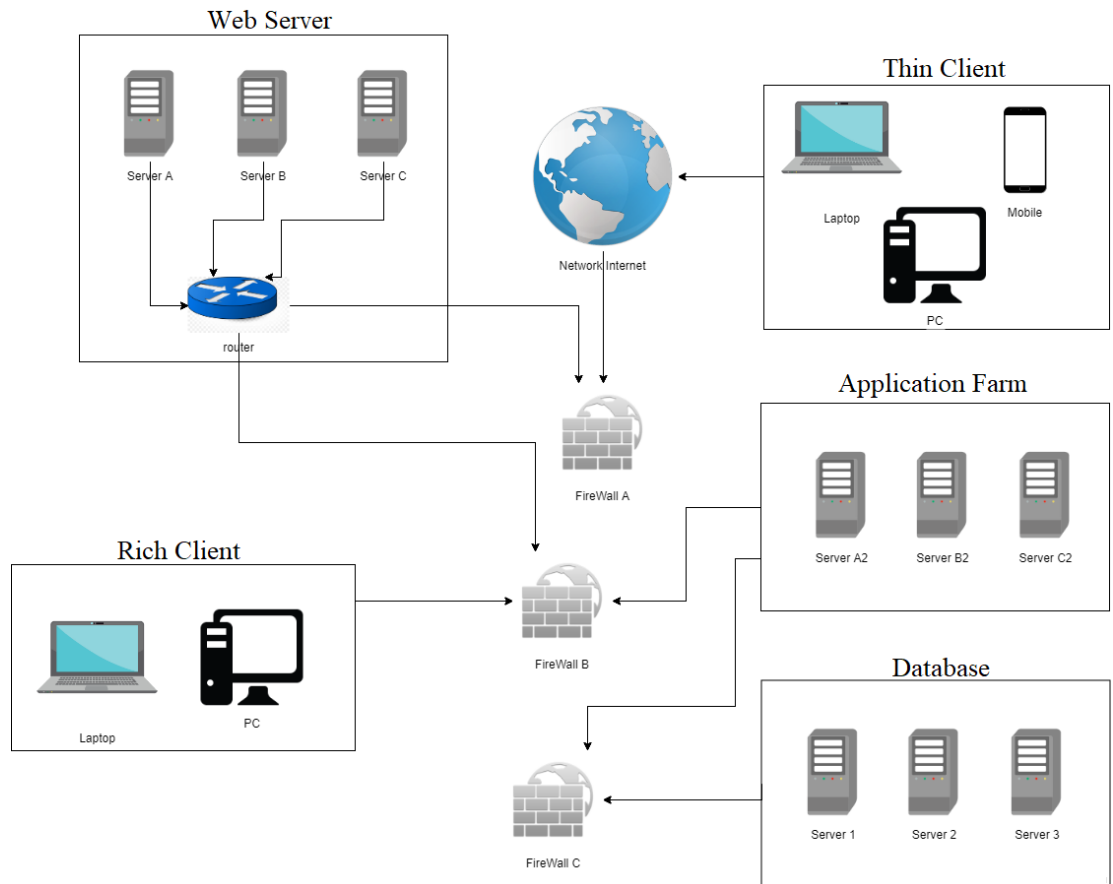


Figure 5: Hardware Diagram

This Hardware diagram is the estimated hardware for the system.

Thin Client : Is the client for server-client structure. Includes the software that's used by devices and they are connected to the internet.

Web Servers : Web servers are connected to the router which is connected to Firewall A and B.

Application Farm : A copy from the servers that give the same functionality in case one or more servers failed.

Database : Combining more than one server in single database.

3.2.1 Class Diagram



Figure 6: Class Diagram

3.2.2 Sequence Diagrams

1- Registration

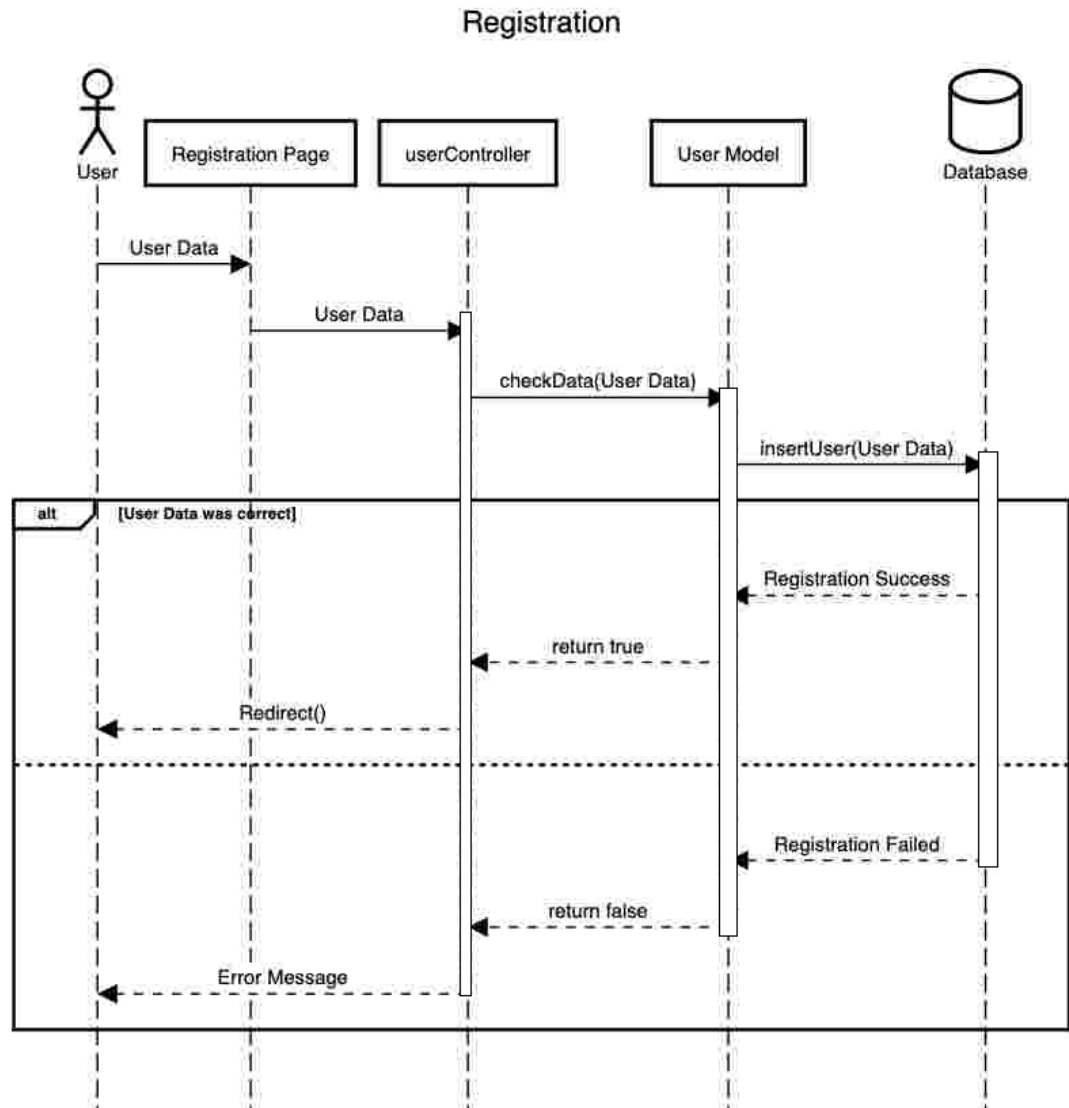


Figure 7: Registration sequence Diagram

A registraion form is displayed for the user, at first the user insert his data as example his first and last name, email and password. Validations are done using Javascript Functions so the validation is done while the user is entering his data, displaying him error messages if any of the fields are inserted with wrong

format other than needed. Then the data is tranfered to the userController so it can be checked for SQL Injections. Finally the User Data is tranfered to the User Model to be impelemented to the database, returning true if the data is inserted into the database succesfully to the userController so it can redirect the user to the index page so he can log in and use the system. Or returning false if the data isn't inserted into the database, and the userController displays an error message to the user.

2- Edit Account

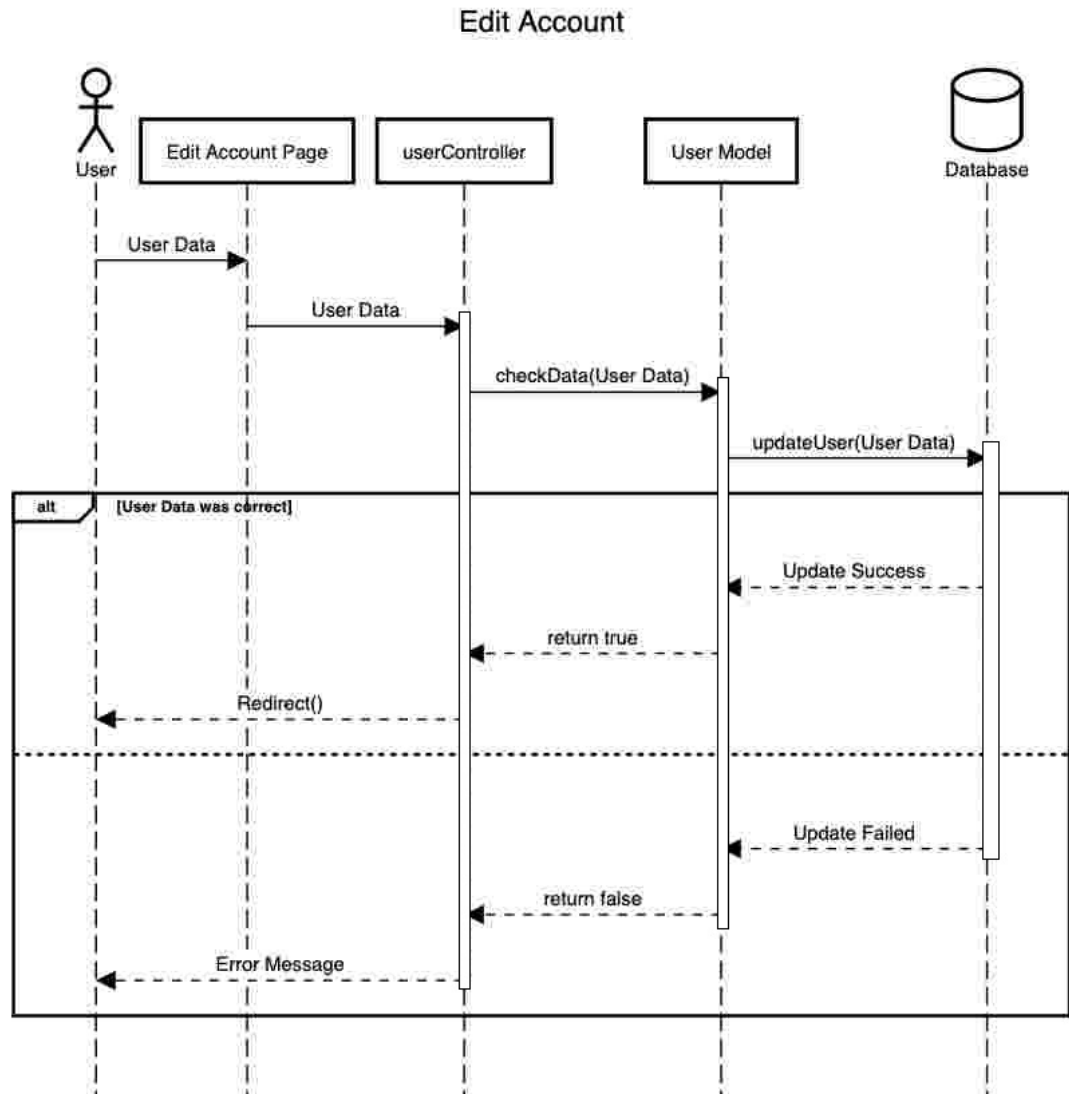


Figure 8: Registration sequence Diagram

An edit form is displayed for the user containing his first name, last name, telephone number and email. Of course form fields can't be left empty, while the user is editing his data a Javascript function is triggered to validate only the data format and shows an error if only the data format is wrong. After the user hit the submit button the data is transferred to the userController so it can be checked from SQL Injection, passing this new data to the user model so it

can be updated in the data base, returning true to the userController if the data is update in the database succesfully and the userController re-display the new data of the user. Or returning false to the userController if the data isn't updated in the database succesfully and the userController displays an error message.

3- Delete Account

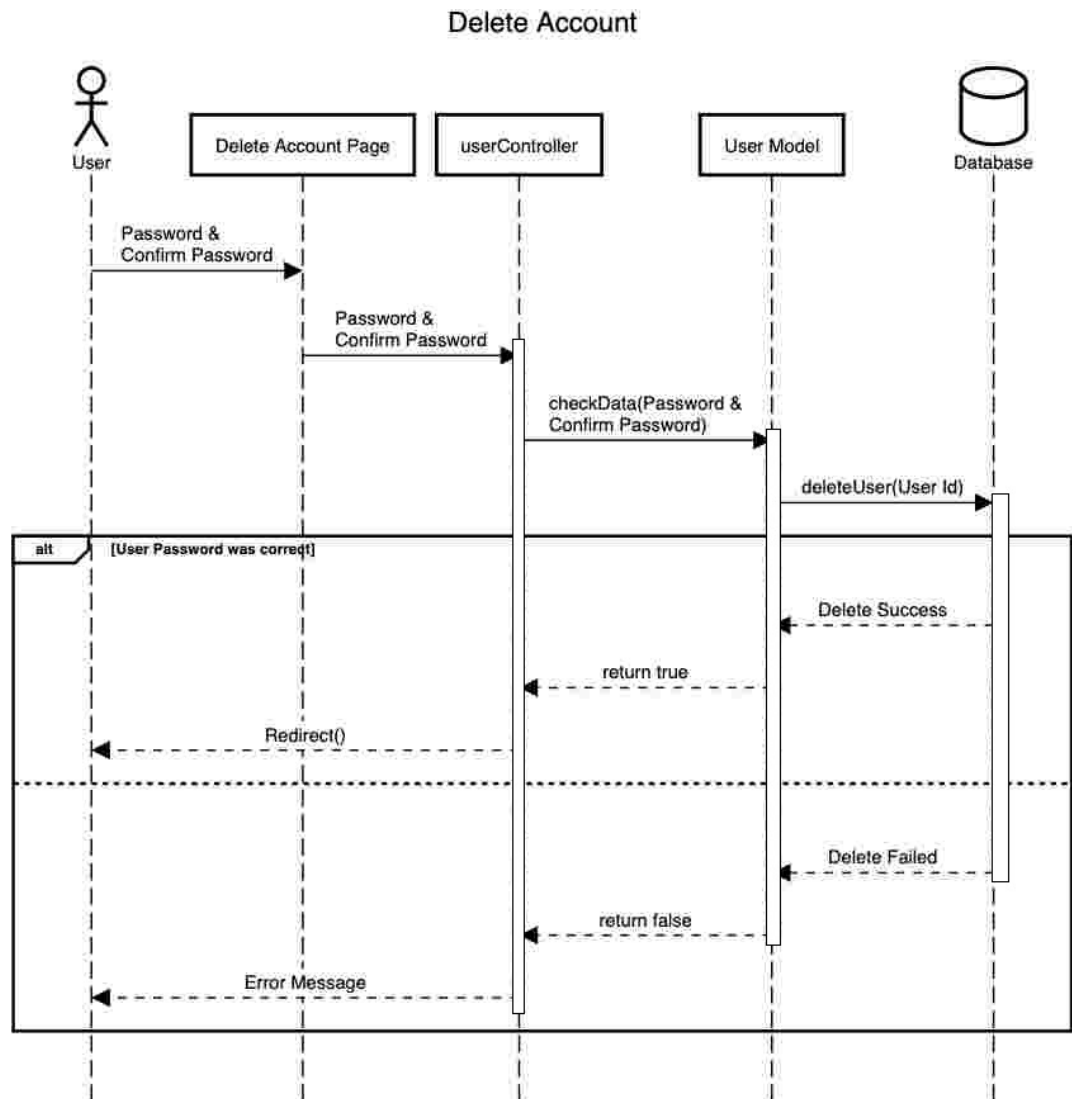


Figure 9: Registration sequence Diagram

A small form is displayed to the user composed of a field for the account

password, a field confirming the password. After the user hit the submit button the password is transfered to the userController checking for SQL Injection, then passed to a function in the user model called deleteUser, this function only sets the user isDeleted field from 0 to 1. If this proccess happened succesfully it returns true to the userController and the userController destroyes the session containing the user data and redirects him to the home page.

4- Log In

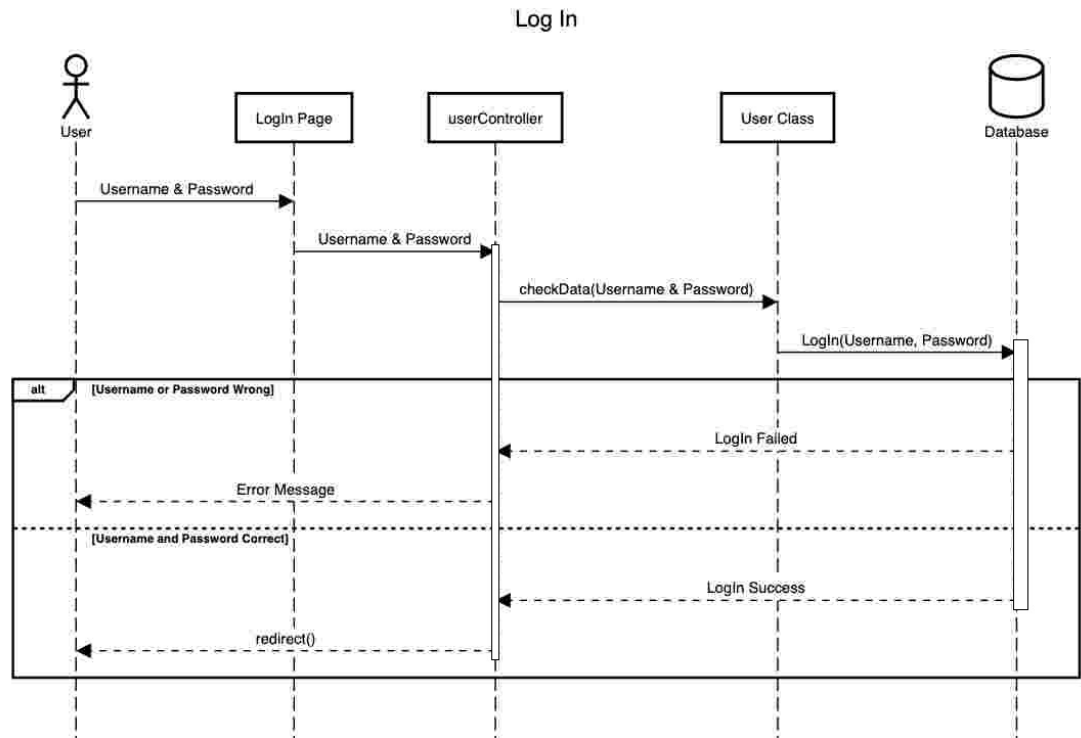


Figure 10: Log In sequence Diagram

A basic field is displayed to the user composed of a field for the email and a field for the password. After the user enters his email and password, these data are passed to the userController of course it must be checked for SQL Injection and passed to a function in the user model called `login`, checking if the email and password entered are correct in the database, if it's correct it returns true to the userController that gives access to the user to sign in and use the system. If the data entered are wrong the user model returns false to the userController that displays an error message to the user.

5- Reservation

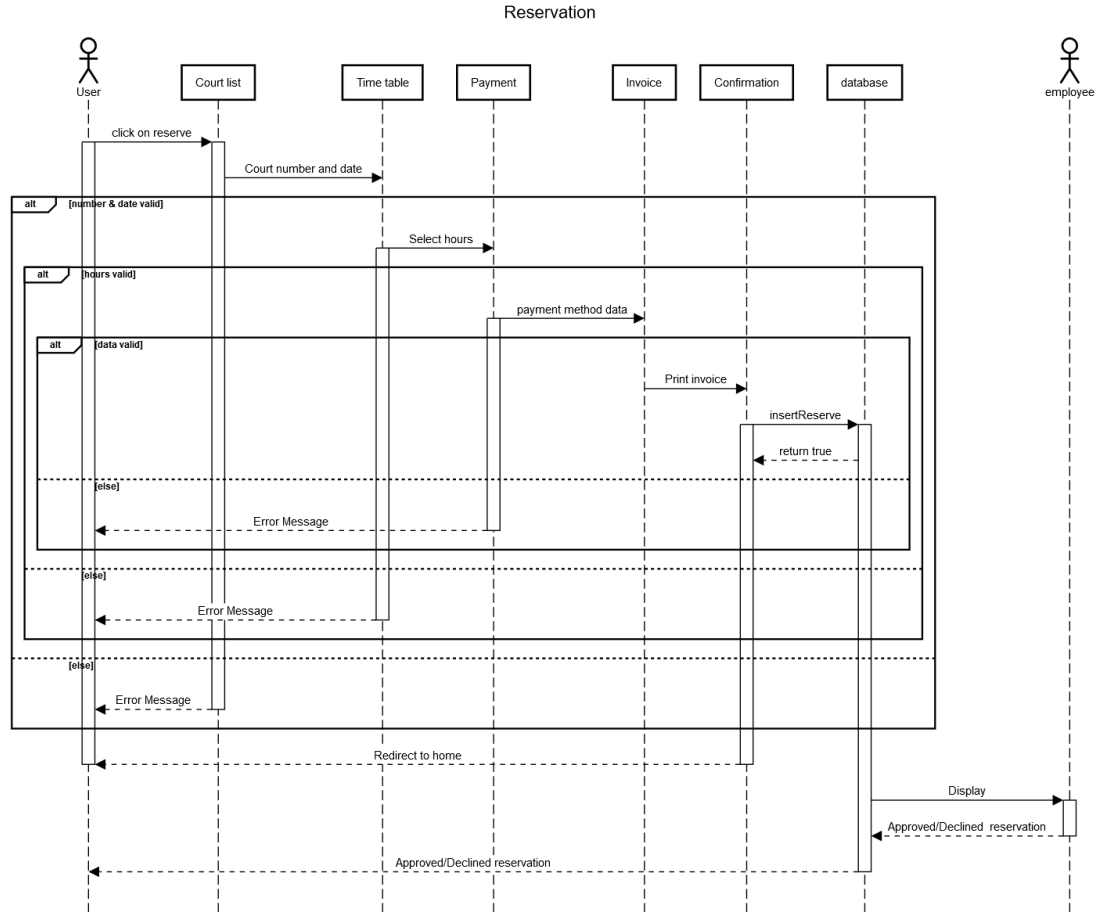


Figure 11: Reservation sequence Diagram

The guest user clicks on reserve button after that a form is shown where he choose a court number and date if he didnt choose one of them an error message appears, if he choose the court and date he then chooses from available hours if he didn't choose hours an error message appears, if he chooses from available hours. He then chooses which method to pay with if he didnt fill the form correctly or left it empty an error message appear , else he will confirm reservation and print invoice then he will be rediercted to homepage. on the other side Employee will approve/decline reservation which will be shown to him from the database.

6- Payment Method

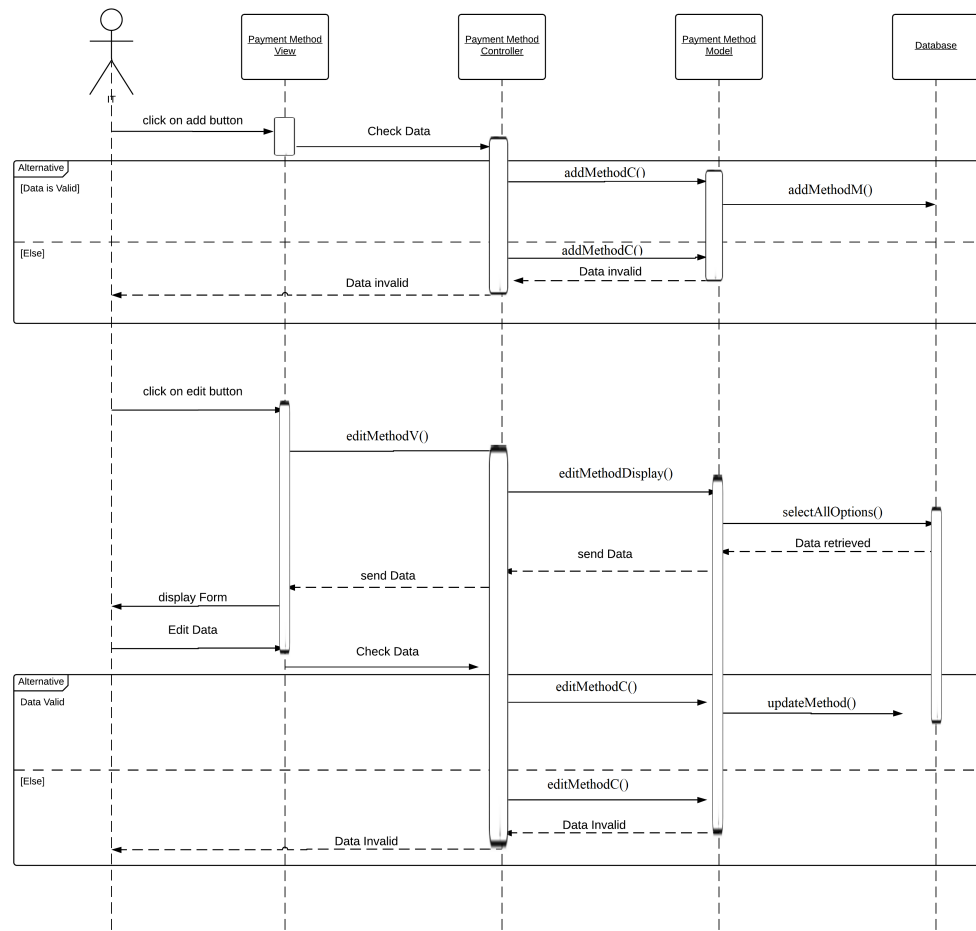


Figure 12: Payment Method sequence Diagram

IT member can add payment methods through the Payment method View, then the data he entered is checked in the Payment Method Controller, if the data is valid data is sent to the model which then adds new record to the database. IT can also edit method in this case when he clicks on the edit data is retrieved from database to show the editable data. When he submits edited data, it is checked and sent to database or an error message is shown to the user in case data is invalid. The above diagram is also applicable to the Options CRUD with the same sequence.

7- Courts

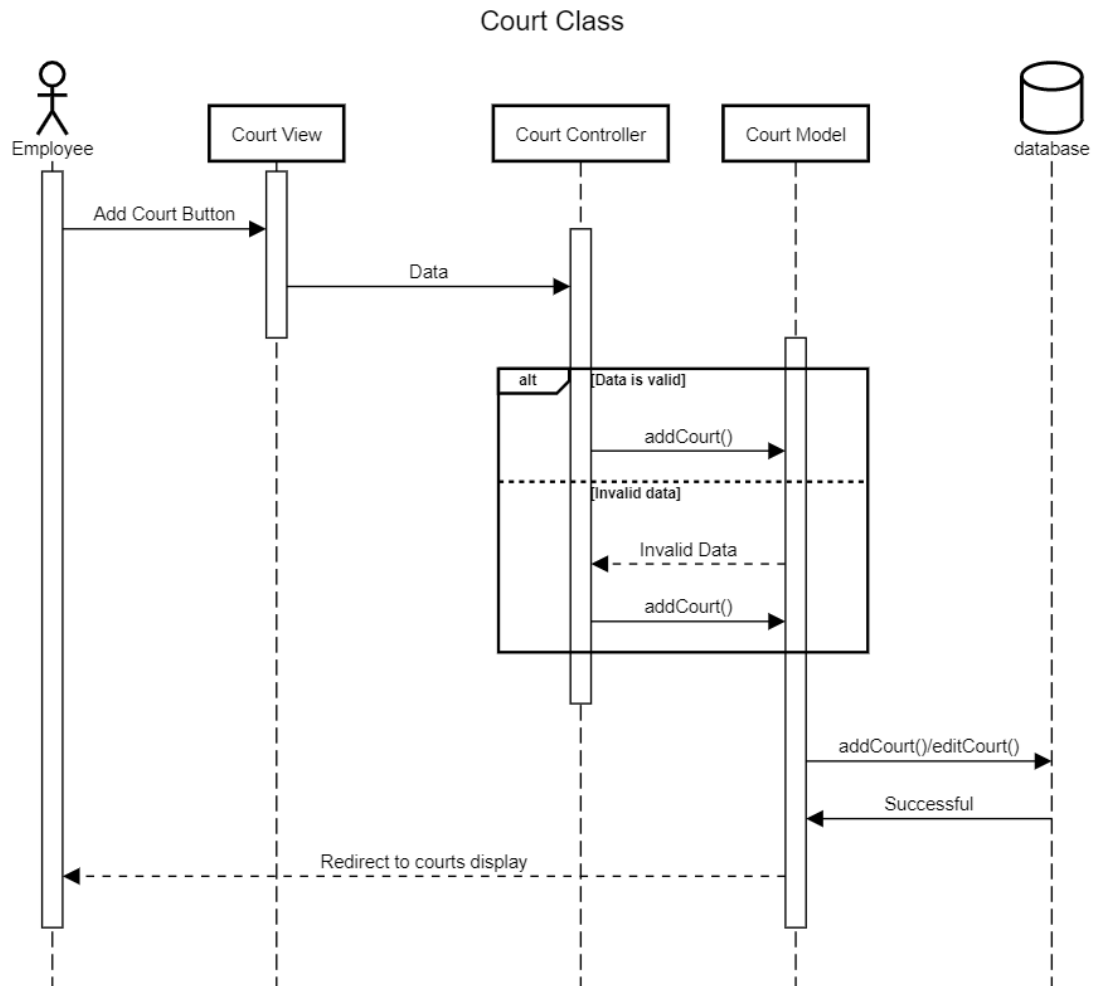


Figure 13: Courts sequence diagram

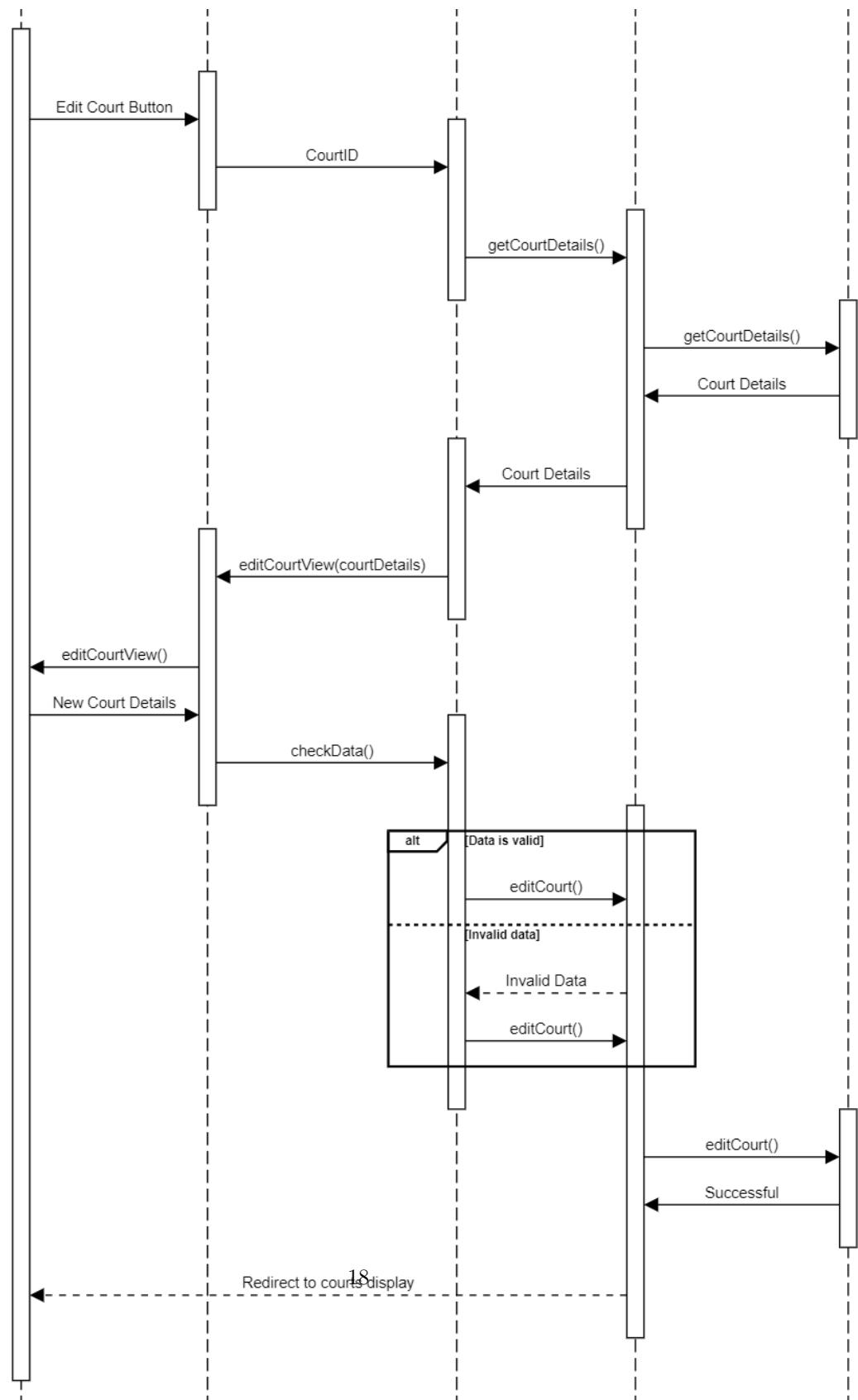


Figure 14: Courts sequence diagram continued

Employee can add a new court by going through the court view, the data he enters is checked and validated in the court controller, which is sent to the model to be added into the database if the data is valid, if it's not, the employee will have to resubmit data through the form. Employee can also edit existing courts. When the edit button is clicked, a form is shown in the court view similar to the add form but with the selected court data shown on the input fields. When the form is submitted, the controller checks if the data is valid and passes it to the model which then updates it into the database. The same diagram applies for the event CRUD.

3.3 Design Rationale

3.3.1 Model View Controller (MVC)

MVC architecture was used in this project as it's easier for the developer to understand and on the other hand easier to test and test costs are reduced. We divided the classes into 3 classes, Model includes all the queries from the database, View includes all HTML that's responsible for page interface and Controller includes all the logic and validations between model and view files.

3.3.2 Singleton Design Pattern

Singleton design pattern was used in the connection class to make a static variable for every user using the system to prevent the overwhelming of creating a new connection every time a sql is used to access the database to retrieve or edit or update or insert a data. Giving better performance to access the database and also prevent the overwhelming of the many useless connections created.

3.3.3 Strategy Design Pattern

Strategy design pattern was used in viewing reports as different shapes currently (Pie Chart and Bar chart). As Viewing the shapes requires different algorithms so we implement different classes with different algorithm and that's what the strategy pattern support.

3.3.4 Observer Design Pattern

Observer design pattern was used in making the notification system as It supports the principle of loose coupling between objects that interact with each other.

3.3.5 Decorative Design Pattern

Decorative design Pattern was used in calculating the Reservation total cost as it allow behavior modification at runtime rather than going back into existing code and making changes.

3.3.6 Facade Design Pattern

Facade design pattern was used to create a layer of abstraction between the classes with their complex implementation and the developer. It helps allow the developer to use many functions from different classes without having to look at their implementation and saves time when implementing new features with those classes.

3.3.7 Factory Design Pattern

Factory design pattern was used to create an object of a specific class in the system wanted by the developers only by specifying the type of the class (Model, View or Controller), then specifying the class name and finally passing parameters if the class constructor requires parameters. This design pattern prevents the struggle of editing the class name of every object in the code if the class name was edited during the development of the system, then the class name will only be edited one time in the factory class. Finally the factory class returns an object of the desired class.

4 Data Design

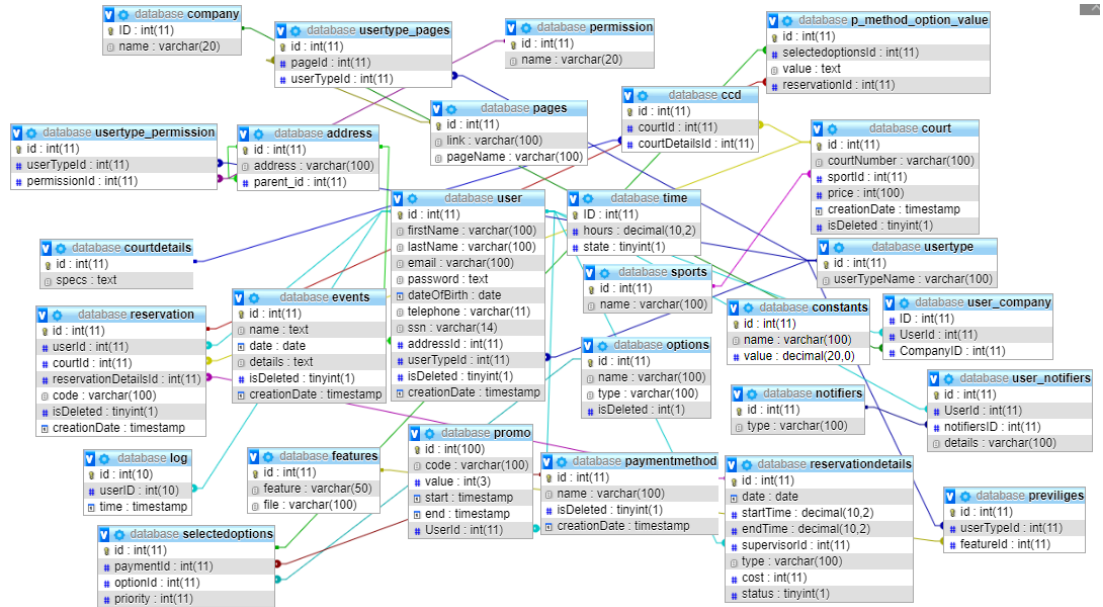
4.1 Data Description

Our Data will be stored in a database using MySQL phpmyadmin that's installed locally on the machine.

The database main tables are :

- 1. User :** User data is distributed on multiple tables each for user information, user roles
- 2.Reservation :** Connected with multiple tables to get reservation details, user that reserved, the court that was reserved
- 3.Payment Method :** Contains all payment methods that the user will pay with
- 4.Courts :** Contains all details about courts as its number , sport type and its price

Database Design:



4.2 Data Dictionary

Table	Column	Type
User	id	int(11)
	firstName	varchar(100)
	lastName	varchar(100)
	email	varchar(100)
	password	text
	dateofBirth	date
	telephone	varchar(11)
	ssn	varchar(14)
	addressId	int(11)
	userId	int(11)
	isDeleted	tinyint(1)
	creationDate	timestamp
Reservation	id	int(11)
	userId	int(11)
	courtId	int(11)
	reservationDetailsId	int(11)
	code	varchar(100)
	isDeleted	tinyint(1)
	creationDate	timestamp
Paymentmethod	id	int(11)
	name	varchar(100)
	isDeleted	tinyint(1)
	creationDate	timestamp
court	id	int(11)
	courtNumber	varchar(100)
	sportId	int(11)
	price	int(100)
	creationDate	timestamp
	isDeleted	tinyint(1)

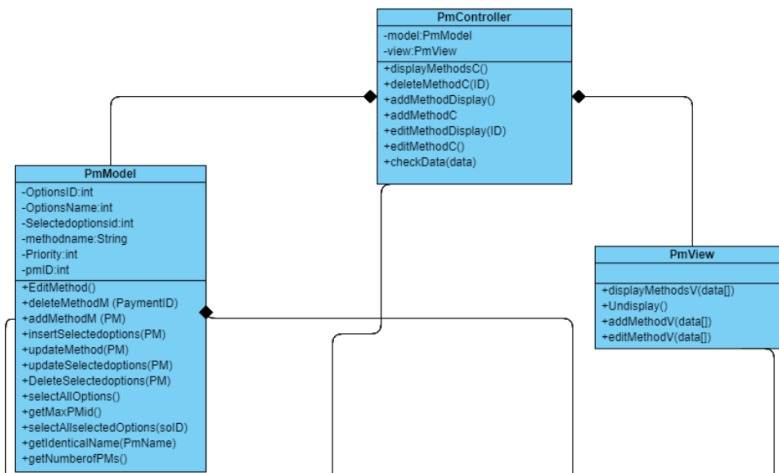
Table 1: Database core tables

5 Component Design

This section we will describe the main core classes in the class diagram and this includes :

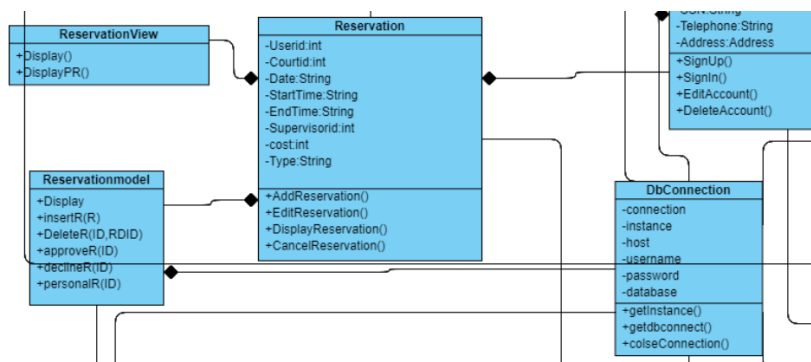
User Class , Reservation class, paymentMethod Class and court class.

5.0.1 User Class



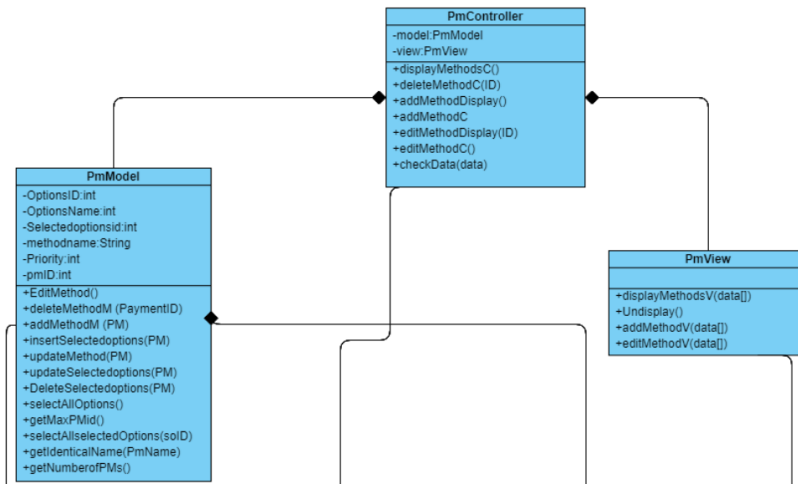
The user class contains all information about the users including (First name , last name , SSN ,etc..) and their roles in our system. The user class aggregates DbConnection class as it uses DbConnection to connect to the database to execute specific queris required for the user to login, Signup or edit his account.

5.0.2 Reservation Class



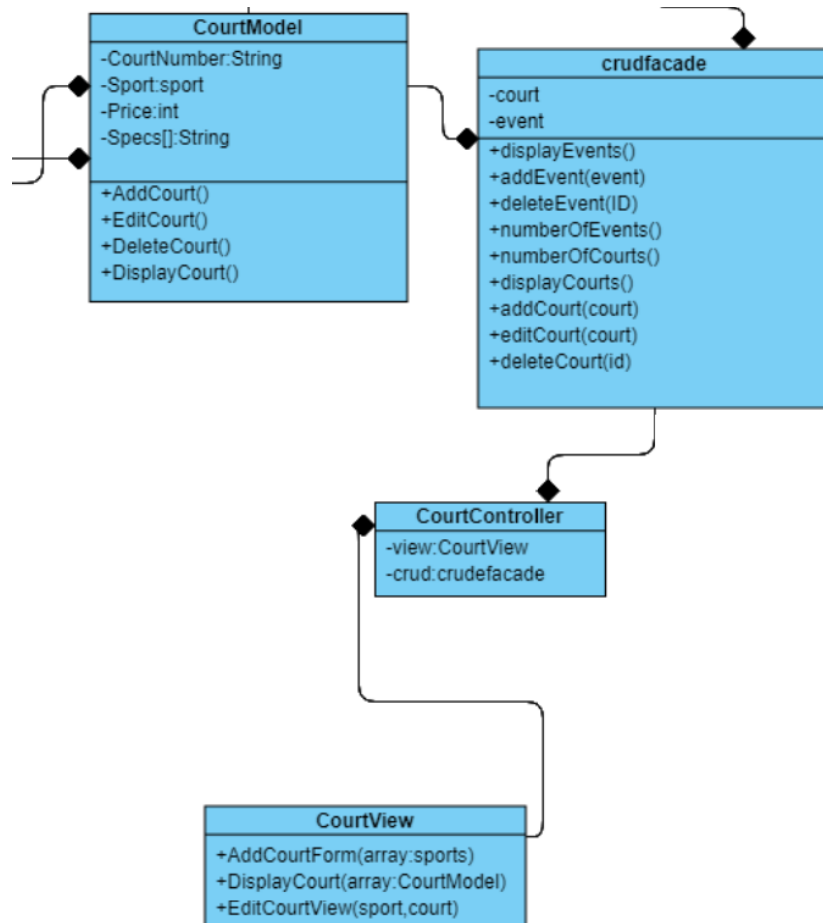
Reservation class responsibilities are divided into 3 classes due to the MVC architecture. The Reservation class aggregates reservation model and reservation view as it connects between them and show the interface from the view and execute queries from the model. It also associates the Pmcontroller class as it was needed to make a choice for payment method.

5.0.3 payment Method Class



paymentMethod class is also divide into model, view , controller classes . The contoller class aggregates both the view and the model. These classes are responsible for adding , deleting or editing any payment method. the model class aggregates from the DbConnection class so it can communicate with the database.

5.0.4 Court Class



the facade design pattern was included beside the 3 classes of the court. crud-facade aggregates from the court model to use the court model functions. court controller aggregates the crudfacade class to call functions of courtModel and execute queries also, it aggregates court view to call its functions to view the page for users.

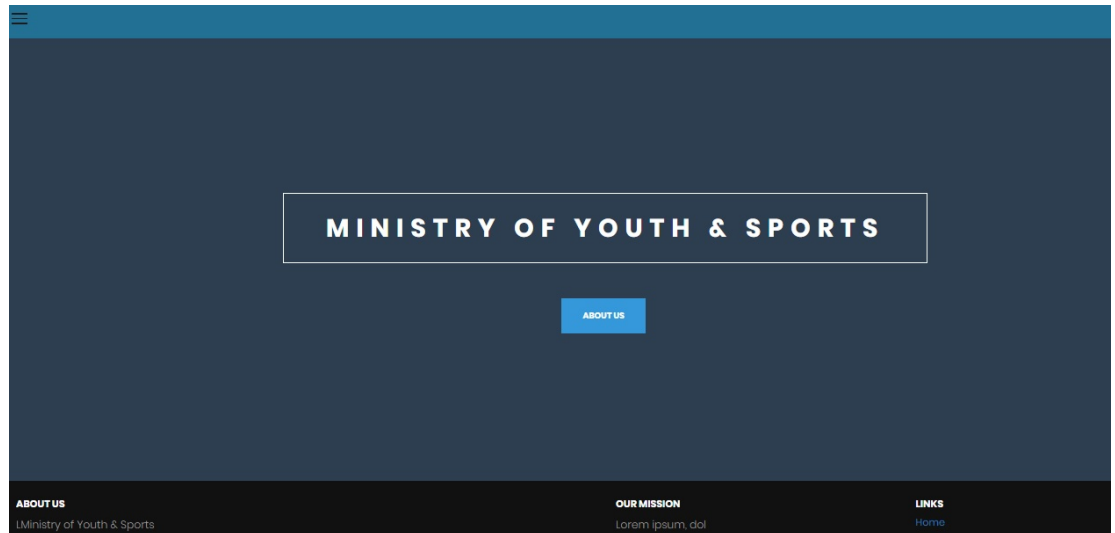
6 Humnan Interface Design

6.1 Overview of User Interface

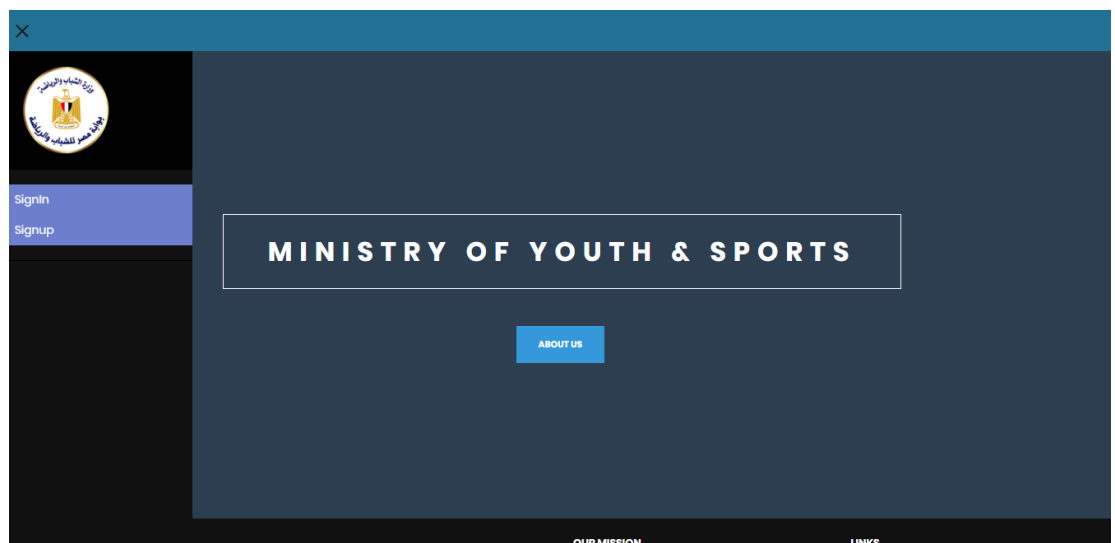
Guest: will encounter a Reservation button in the navigation bar to easily create a Reservation and later on he can open his reservation to see if it was approved or he will get notified , he will also be able to view events

6.2 Screen Images

6.2.1 Home Page



6.2.2 Before Login



6.2.3 Login Page

EMAIL ADDRESS

Email

PASSWORD

Password

SIGN IN

Don't have account ? [Sign Up Here](#)

6.2.4 Registration Page

Personal Information

FIRST NAME

First Name

LAST NAME

Last Name

DATE OF BIRTH

dd/mm/yyyy

PHONE NUMBER

Phone Number

SOCIAL SECURITY NUMBER

SSN

HOW WOULD YOU LIKE TO BE NOTIFIED ?

Choose ▼

Account Information

EMAIL ADDRESS

Email

PASSWORD

Password

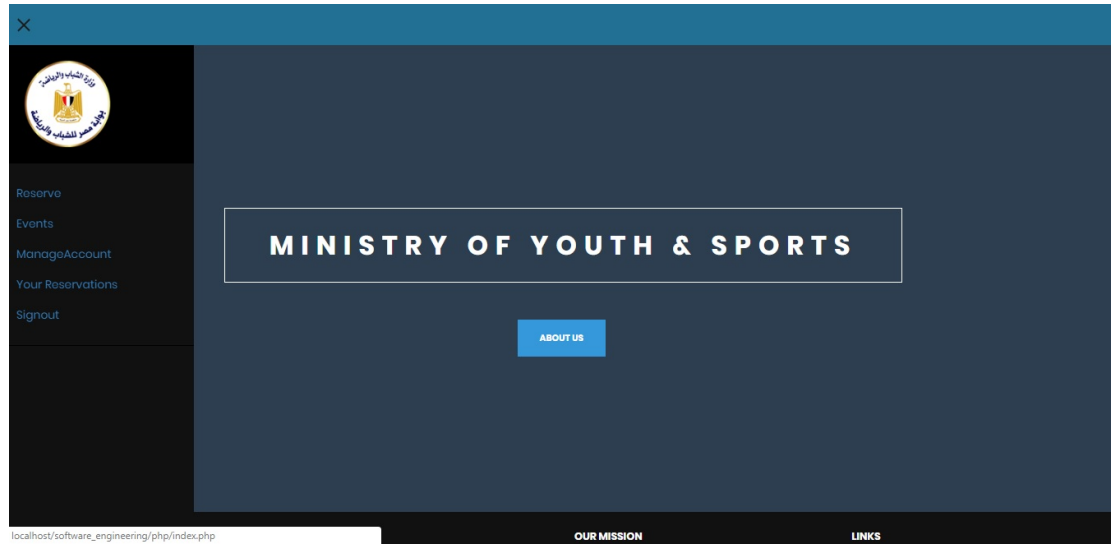
CONFIRM PASSWORD

Confirm Password

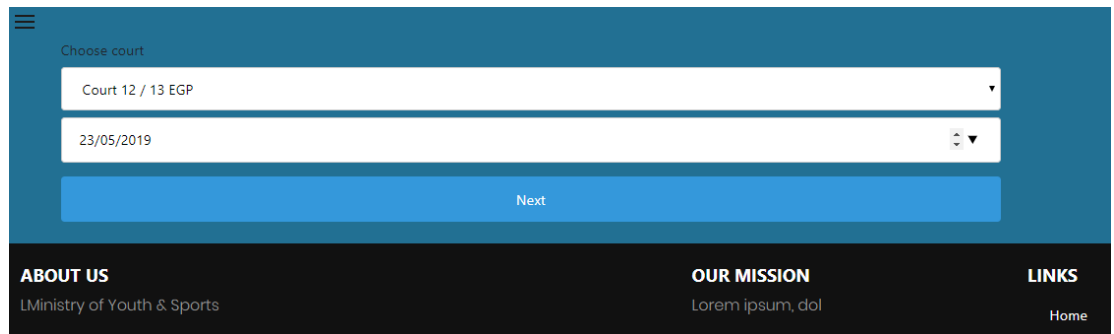
REGISTER

Already have account ? [Sign In](#)

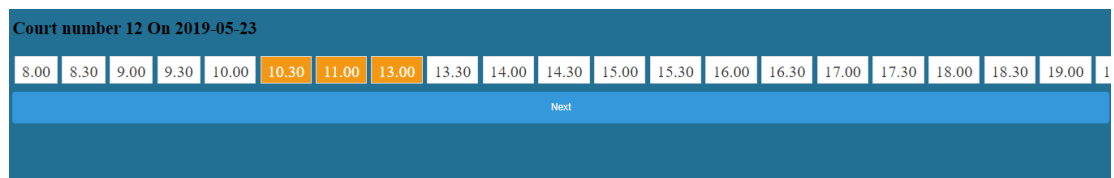
6.2.5 After Login/Signup



6.2.6 Reservation - Choosing a court and date




6.2.7 Reservation - Choosing time



6.2.8 Reservation - Choosing Payment Method

Payment Method
Visaa
Name
Guest
ExpiryDate
18/05/2019
number
133
promo code
Checkout

6.2.9 Reservation - Confirmation ticket

Checkout
Date:
2019-05-23
From:
10.30
To:
13.30
Reserved Court:
12
Total Cost:
total
19.5
total
19.5
Payment Method:
Visaa

confirm and download invoice cancel
ABOUT US Ministry of Youth & Sports
OUR MISSION carm@scrm.gov.il
LINKS Home

6.2.10 Display Events

Show 10 entries

Search:

Event Name	Event Date	Event Details
Football Match	2019-05-31	Ahly vs Zamalek
Music Concert	2019-05-28	3adel shakal will rock the stage

Showing 1 to 1 of 1 entries

Previous1Next

ABOUT US

OUR MISSION

LINKS

LMinistry of Youth & Sports

Lorem Ipsum, dol

Home

6.2.11 Display Reservations

Show 10 entries

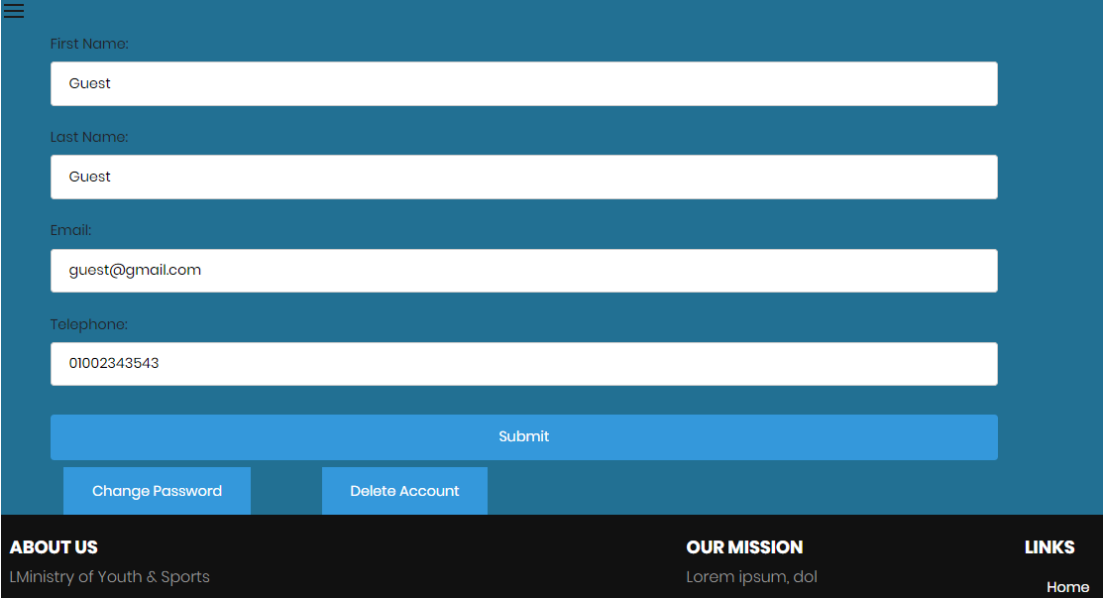
Search:

Court Number	Date	Reserver First name	Reserver Last name	Start time	End time	Suppervisor first name	Suppervisor Last name	Reservation Status
12	2019-05-24	Guest	Guest	10.30	11.30	test1	test1	Pending

Showing 1 to 1 of 1 entries

Previous1Next

6.2.12 Manage Account



The image shows a 'Manage Account' form with a dark blue header and a black footer. The form fields are white with blue borders. The fields are labeled 'First Name:', 'Last Name:', 'Email:', and 'Telephone:'. The 'First Name' and 'Last Name' fields contain the text 'Guest'. The 'Email' field contains 'guest@gmail.com'. The 'Telephone' field contains '01002343543'. Below the fields is a large blue 'Submit' button. Below the 'Submit' button are two smaller blue buttons: 'Change Password' and 'Delete Account'. The footer contains three sections: 'ABOUT US' with the text 'Ministry of Youth & Sports', 'OUR MISSION' with the text 'Lorem ipsum, dol', and 'LINKS' with the text 'Home'.

First Name:
Guest

Last Name:
Guest

Email:
guest@gmail.com

Telephone:
01002343543

Submit

Change Password

Delete Account

ABOUT US
Ministry of Youth & Sports

OUR MISSION
Lorem ipsum, dol

LINKS
Home

6.3 Screen Objects and Actions

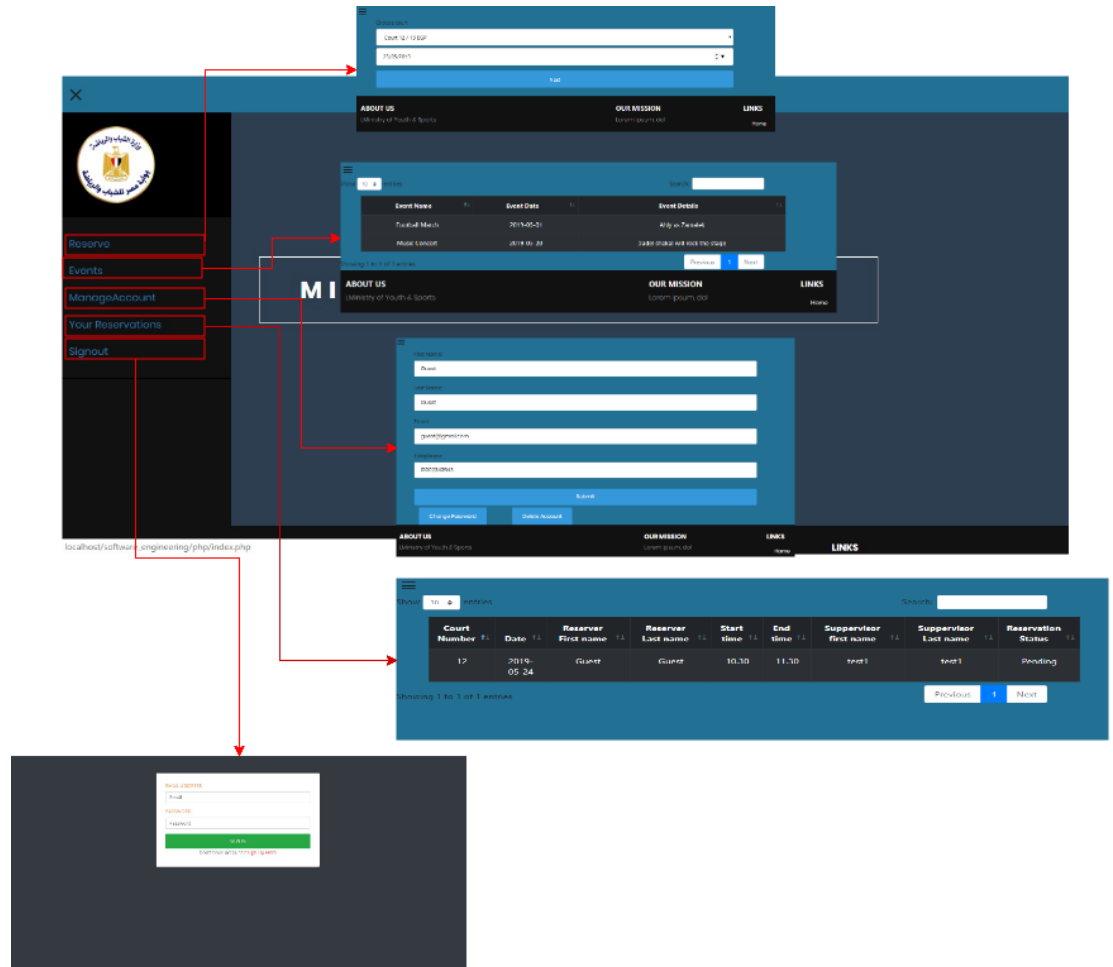
This section will discuss what happens when the user clicks on each component of the interfaces above. The user will Signup or Login then he can reserve a court , display events , display his reservation or manage his account.

6.3.1 Home Page- Before Login



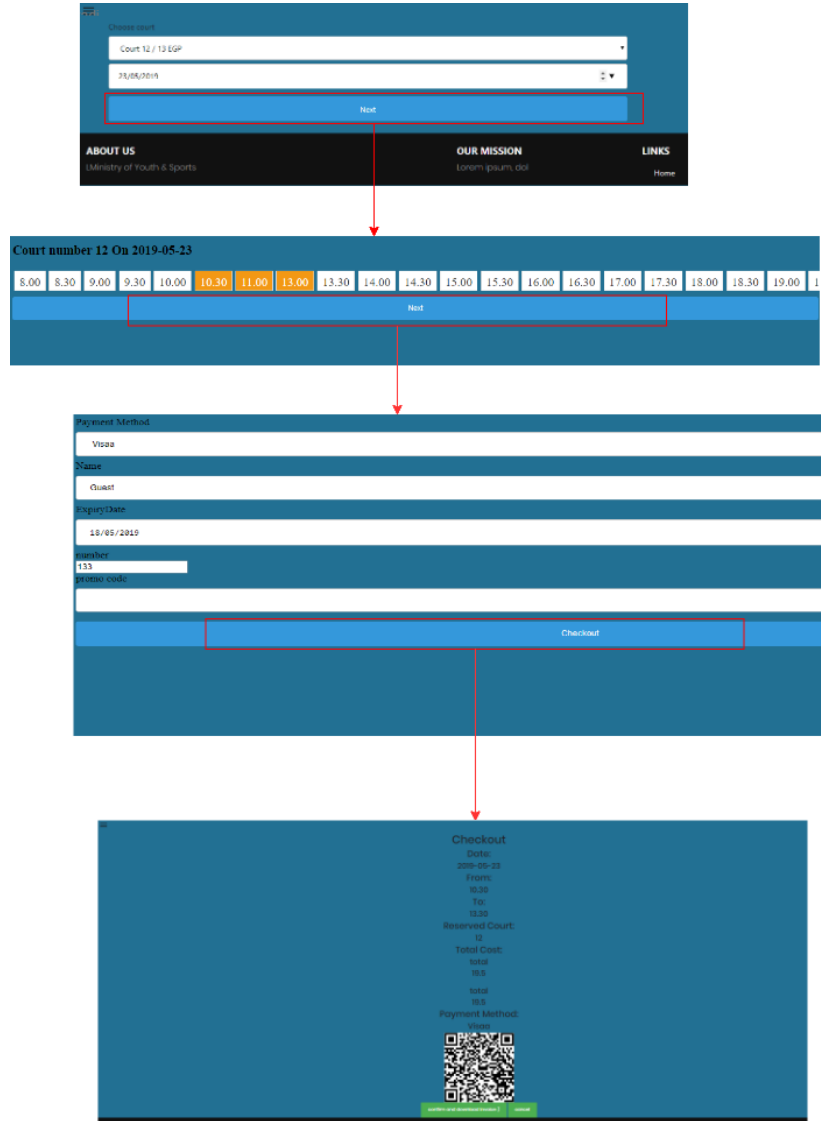
The guest user before signing up or logging into his account has 2 options either to signup if he dont have an account or to login to have access to many things.

6.3.2 Home Page- After Login



After the user LoggedIn/SignedUp he have access to reserve a court , display events that will happen in the future, manage his account so he can edit/delete his account and if he signs out he is rediercted to login page again.

6.3.3 Reservation process



When the user clicks on Reserve as shown above a form for choosing courts and date is shown to him. He choose courts from a dropdown list and date from a calendar. after clicking next he select number of available hours he desires. after clicking next he chooses number his payment method from a dropdown list and a form is shown according to the method he chose. after clicking checkout a confirmation ticket is shown to him and he can cancel or confirm and download a pdf file that have the ticket.

7 Testing Plan

7.1 Add Payment method

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Method Name 2. Select Options 3. Options Priority number	1. Enter Method Name, select one or more options, set priority number to them.	Method Name: "Visa" Selected Options: "Name", "Number" Priority: "1", "2"	Insert new data to database and redirect to Methods table.	As Expected.	Pass
	2. Leaving Method Name empty and not selecting option	Method Name: "" Selected options: ""	Error Message and redirecting to Methods table.	As Expected.	Pass
	3. Enter Method Name that's already taken.	Method Name: "Visa"	Error Message and redirecting to Methods Table.	As Expected.	Pass
	4. Giving two or more options a same priority	Method Name: "Mastercard" Selected Options: "Name", "Number", "Loan" Priority: "1", "1", "2"	Inserting only options with right priority and neglecting all options with conflicting priorities then redirecting to methods table	As Expected.	Pass

7.2 Edit Payment method

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Method Name 2. Delete options 3. Add options 4. Options Priority number	1. Edit method Name to valid name, add new option & set valid priority to it & delete some options	Method Name: "Visa Card" Add Option: "Loan" Priority: "4" Delete Option: "Name", "number"	Edit New data and update it to the database and redirect to methods table	As Expected.	Pass
	2. Delete Method Name & remove all options Priority	Method Name: " " Priority: " "	Error Message and redirecting to Methods table.	As Expected.	Pass
	3. Enter Method Name that's already taken.	Method Name: "Fawry"	Error Message and redirecting to Methods Table.	As Expected.	Pass
	4. Editing two or more options to same priority	Priority: "1", "1", "2"	Rejecting any conflicting priorities and give them their old priorities.	As Expected.	Pass

7.3 Add/Edit Options

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Option Name 2. Option Type	1. Enter valid Option Name & Type.	Option Name: "Number" Option Type: "text"	Insert new data to database and redirect to Options table.	As Expected.	Pass
	2. Leaving option Name or Type empty	Option Name: " " Option Type: " "	Error Message and redirecting to Options table.	As Expected.	Pass
	3. Enter option Name that's already taken.	Option Name: "Number" Option Type: "text"	Error Message and redirecting to Options Table.	As Expected.	Pass
	4. Entering option Type that includes numbers.	Option Name: "Number" Option Type: "text123"	Error Message and redirecting to Options Table.	As Expected.	Pass

7.4 Add/Edit Courts

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Sport 2. Court Number 3. Price Per Hour 4. Court Specs	1. Select a sport, enter court number and hourly price, and select specs.	Sport: "Football" Court Number: "2" Price Per Hour: "20" Court Specs: "Grass"	Court inserted successfully and redirected to courts table.	As Expected.	Pass
	2. Leave Court Number, Specs, Price or sports fields empty.	Sport: "" Court Number: "" Price Per Hour: "" Court Specs: ""	Error Message and page won't submit.	As Expected.	Pass
	3. Enter a court number or price in negative.	Sport: "Tennis" Court Number: "-3" Price Per Hour: "-20" Court Specs: "Clay"	Error Message and page won't submit.	As Expected.	Pass
	4. Enter a court number that exists in the same sport.	Sport: "Football" Court Number: "2" Price Per Hour: "20" Court Specs: "Grass"	Error message and redirected to form.	As Expected.	Pass

7.5 Add/Edit Events

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Event Name 2. Event Date 3. Event Details	1. Enter event name, choose date, and enter event details.	Event Name: "7afla" Event Date: "6/22/2019" Event Details: "MIU welcome party"	Data is inserted to database and redirected to events table.	As Expected.	Pass
	2. Leave any of event name, date, or details empty.	Event Name: " " Event Date: " " Event Details: " "	Error Message and page won't submit.	As Expected.	Pass
	3. Enter a date before the current day of submitting.	Event Name: "7afla" Event Date: "3/22/2019" Event Details: "MIU graduation party"	Error Message and page won't submit.	As Expected.	Pass
	4. Enter an event with same name and date as another.	Event Name: "7afla" Event Date: "6/22/2019" Event Details: "Welcome party"	Error message and redirected to events table.	As Expected.	Pass

7.6 Add Reservation

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Court number 2. Reservation date 3. Reservation time 4. Payment method	1. Choose court, Reservation day and time and payment method	Court number :12 Reservation date:12/5/2019 Reservation time:10-11 Payment method: visa	Insert new data to database , offer to download invoice and redirect to home	As Expected.	Pass
	2. Leaving court number or reservation day empty	Court number: " " Reservation day: " "	Error Message and prevent from redirecting	As Expected.	Pass
	3. Leave reservation time empty	Reservation Time:""	Error Message and prevent from redirecting	As Expected.	Pass
	4. Leave payment method empty	Payment method :""	Error Message and prevent from redirecting	As Expected.	Pass

7.7 Login

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Username 2. Password	1. Enter Username and Password correct.	Username: "ahmed@gmail.com" Password: "Ahmed"	Log the user in and redirect to the home page.	As Expected.	Pass
	2. Leaving the username or password empty.	Username: " " Password: " "	Error Message and won't log in the user and will refresh the log in page.	As Expected.	Pass
	3. Enter the email or password containing sql injection.	Username: " 1=1" Password: "Password"	Error Message and won't log in the user and will refresh the log in page.	As Expected.	Pass
	4. Enter the email or password wrong.	Username: "ahmed@gmail.com" Password: "Password"	Error Message and won't log in the user and will refresh the log in page.	As Expected.	Pass

7.8 Delete User

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. Password 2. Confirm Password	1. Enter Password and Password confirmation correct.	Password: "Ahmed" Confirm Password: "Ahmed"	Session will be destroyed and the user will be redirected to the home page.	As Expected.	Pass
	2. Leaving the Password or Confirm Password empty.	Password : " " Confirm Password : " "	Error message and the user's account won't be deleted.	As Expected.	Pass
	3. Enter the Password or confirm Password containing sql injection.	Password: " 1=1" Confirm Password : "Password"	Reload the page and won't delete the user's account.	As Expected.	Pass
	4. Enter the Password or Confirm Password wrong.	Password: " ahmed" Confirm Password : "Password"	Error Message and won't delete the user's account.	As Expected.	Pass

7.9 Edit User

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. First name 2. Last name 3. Email 4. Telephone	1. Enter all the data correct.	First name: "Ahmed" Last name: "Mohamed" Email: ahmed@gmail.com Telephone: "01234123123"	Data of the account will be updated in the database and the page will be reloaded displaying the new data.	As Expected.	Pass
	2. Leaving any of the data fields empty.	First name: " " Last name: " " Email: " " Telephone: " "	Error message and the user's account won't be updated.	As Expected.	Pass
	3. Enter any data containing sql injection.	First name: "1=1" Last name: "Mohamed" Email: ahmed@gmail.com Telephone: "01234123123"	Error message and the user's account won't be updated.	As Expected.	Pass
	4. Enter any field wrong.	First name: "Ahmed" Last name: "Mohamed" Email: " Ahmed" Telephone: "00"	Error message and the user's account won't be updated.	As Expected.	Pass

7.10 Registration

Input	Test Scenario	Input Values	Expected Result	Actual Result	Pass/Fail
1. First name 2. Last name 3. Date of Birth 4. Phone Number 5. SSN 6. Email 7. Password 8. Confirm password	1. Enter all the data correct.	First name: "Ahmed" Last name: "Mohamed" Date of birth: "16-01-1990" Phone number: "01234123123" SSN: "11234567891234" Email: ahmed@gmail.com Password: "Ahmed" Confirm password: "Ahmed"	User data will be registered in the database and the user will be redirected to home page to use the system.	As Expected.	Pass
	2. Leaving any of the data fields empty.	First name: " " Last name: " " Date of birth: " " Phone number: " " SSN: " " Email: " " Password: " " Confirm password: " "	Error message and the user's account won't be registered.	As Expected.	Pass
	3. Enter any data containing sql injection.	First name: "1=1" Last name: "Mohamed" Date of birth: "16-01-1990" Phone number: "01234123123" SSN: "11234567891234" Email: ahmed@gmail.com Password: "Ahmed" Confirm password: "Password"	Error message and the user's account won't be registered.	As Expected.	Pass
	4. Enter any field wrong.	First name: "Ahmed" Last name: "Mohamed" Date of birth: "16-01-1990" Phone number: "-12763176" SSN: "11234567891234" Email: "Ahmed" Password: "Password" Confirm password: "Ahmed"	Error message and the user's account won't be updated.	As Expected.	Pass

8 Requirements Matrix

Requirement ID	Requirement Description	Test Cases ID	Status
1.	User CRUD	TC8, TC9, TC10	TC8-Pass TC9-Pass TC10-Pass
2.	payment method CRUD	TC1, TC2	TC1-Pass TC2-Pass
3.	Courts CRUD	TC4	TC4-Pass
4.	Add Reservation	TC6	TC6-Pass
5.	Options CRUD	TC3	TC3-Pass
16.	Events CRUD	TC5	TC5-Pass