

Go-Play

Project Report Submitted By

SILJA C K

Reg. No.: AJC20MCA-2065

In Partial fulfillment for the Award of the Degree of

**MASTER OF COMPUTER APPLICATIONS (2 Years)
(MCA)**

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY



**AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by
AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally,
Kottayam, Kerala – 686518]

2021-2022

DEPARTMENT OF COMPUTER APPLICATIONS
AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY



CERTIFICATE

This is to certify that the Project report, “**Go-Play**” is the bonafide work of **SILJA C K (Reg.No:AJC20MCA-2065)** in partial fulfillment of the requirements for the award of the Degree of Regular Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2021-22.

Dr. Bijimol T K
Internal Guide

Ms. Grace Joseph
Coordinator

Rev.Fr.Dr. Rubin Thottupurathu Jose
Head of the Department

DECLARATION

I hereby declare that the project report “**Go-Play**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications (2 Years) from APJ Abdul Kalam Technological University, during the academic year 2021- 2022.

Date:

SILJA C K

KANJIRAPPALLY

Reg. No: AJC20MCA-2065

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal Dr. Lilly kutty Jacob for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping me. I extend my wholehearted thanks to the project coordinator **Ms. Grace Joseph** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide, **Dr. Bijimol T K** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

SILJA C K

ABSTRACT

Go-Play is an online web application for managing Sports tournaments. In this project, there are three modules: the first one is admin, the second one is the captain, and the last one is the player. If we talk about admin, where admin work is to manage all the functions related to the sports, schedule, tournaments, booking requests, news, and team. The admin creates the team and selects a captain for the team. And the captain can log in and view game details; he can select players for the team and can register for sports tournaments. Now we come to the third one, which is a player who can view tournament results and score. Here, the admin is the most important part of this sports tournament system because he manages all the events and schedules the time of the sports tournament. He even chooses the playing team members.

CONTENT

Sl. No	Topic	Page No
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	3
2.1	INTRODUCTION	4
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	5
2.4	PROPOSED SYSTEM	5
2.5	ADVANTAGES OF PROPOSED SYSTEM	6
3	REQUIREMENT ANALYSIS	7
3.1	FEASIBILITY STUDY	8
3.1.1	ECONOMICAL FEASIBILITY	8
3.1.2	TECHNICAL FEASIBILITY	9
3.1.3	BEHAVIORAL FEASIBILITY	9
3.2	SYSTEM SPECIFICATION	10
3.2.1	HARDWARE SPECIFICATION	10
3.2.2	SOFTWARE SPECIFICATION	10
3.3	SOFTWARE DESCRIPTION	10
3.3.1	PHP	10
3.3.2	MYSQL	11
4	SYSTEM DESIGN	13
4.1	INTRODUCTION	14
4.2	UML DIAGRAM	14
4.2.1	USE CASE DIAGRAM	15
4.2.2	SEQUENCE DIAGRAM	18
4.2.3	STATE CHART DIAGRAM	20
4.2.4	ACTIVITY DIAGRAM	21
4.2.5	CLASS DIAGRAM	22
4.2.6	OBJECT DIAGRAM	23
4.2.7	COMPONENT DIAGRAM	24
4.2.8	DEPLOYMENT DIAGRAM	24

4.5	USER INTERFACE DESIGN	25
4.6	DATA BASE DESIGN	28
5	SYSTEM TESTING	36
5.1	INTRODUCTION	37
5.2	TEST PLAN	38
5.2.1	UNIT TESTING	38
5.2.2	INTEGRATION TESTING	39
5.2.3	VALIDATION TESTING	39
5.2.4	USER ACCEPTANCE TASTING	40
6	IMPLEMENTATION	41
6.1	INTRODUCTION	42
6.2	IMPLEMENTATION PROCEDURE	42
6.2.1	USER TRAINING	43
6.2.2	TRAINING ON APPLICATION SOFTWARE	43
6.2.3	SYSTEM MAINTENANCE	43
7	CONCLUSION & FUTURE SCOPE	44
7.1	CONCLUSION	45
7.2	FUTURE SCOPE	45
8	BIBLIOGRAPHY	46
9	APPENDIX	49
9.1	SAMPLE CODE	50
9.2	SCREEN SHOTS	61

List of Abbreviation

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1. PROJECT OVERVIEW

Go –Play system is to manage sports tournaments.. This system's primary purpose is to organize and run competitions. This project consists of three sections. The administrator comes in first, followed by the captain and then the player. Managing all the sports-related tasks, including the schedule, tournament, booking requests, news, and team, falls under the administrative category. The team is formed by the admin, who also chooses the captain. Additionally, the captain can log in to check game information, choose players for his squad, and register for sporting events. The third one, a player who can check tournament results and scores, is currently shown. Here, the administrator is the most crucial component of this sports tournament system since he oversees all events, plans the time for the competition, and even selects the members of the competing teams.

1.2. PROJECT SPECIFICATION

The proposed system is a website that manages sports tournaments and teams, schedule tournaments, Publish result of tournaments.

The system includes 3 modules. They are:

Admin Module

An admin must have a login into this system. He has overall control of the system. An Admin can add and manage tournaments. Admin creates a team and chooses a captain from among approved users. can also schedule tournaments and publish the results of tournaments.

Captain Module

The captain of the team is selected by admin. After the approval of the admin, the captain can add players to the team and register for available tournaments. The captain can view player information, tournament results, and scores.

Player Module

The captain can view player information, tournament results, scores and tournament related news and videos.

CHAPTER 2

SYSTEM STUDY

2.1. INTRODUCTION

The process of gathering and analysing data, finding problems, and using the data to recommend system modifications is known as system analysis. Tight collaboration between system engineers and users is required for this activity. A system analysis or research should be the first step in any system development process. The system is carefully inspected and evaluated. System analysts act as interrogators and examine the operation of the current system in great detail. The system's inputs are acknowledged, and the system is viewed as a whole. The numerous procedures can be connected to the organisations' outputs. with identifying the problem, finding the relevant and impacting variables, analysing and combining the various components, and choosing the best or, at the very least, suitable programme or solution.

The Preliminary research is the procedure of gathering and analysing data in order to use it for upcoming system investigations. Initial research requires strong collaboration between system users and developers since it involves problem-solving. It carries out several feasibility studies. These investigations give a rough idea of the system activities, and this information can be utilised to choose the methods to employ for effective system research and analysis.

2.2 EXISTINGSYSTEM

In the current tournament management system, a team registers for a tournament, but a person who is not a member of any team cannot play in the tournament. Sports tournament management system Admin manages various game, tournament, schedule, booking request, news and team.

2.3 DRAWBACKS OF EXISTINGSYSTEM

2.3.1 Only existing teams can play.

2.3.2 A person who is not a member of the team cannot participate in the tournament.

2.4 PROPOSED SYSTEM

The proposed system is defined to meet all the disadvantages of the existing system. It is necessary to have a system that is more user-friendly. In my proposed system, there is an admin who can create and manage tournaments and select the captain for the team. It allows the captain to select players for the team. The proposed system's users are admin, captain, and player. The captain and players of the team can view the tournament results and score. The Go-Play System is a tournament management system designed to facilitate team sports. The players of a team are selected by a system. The system is useful for a sportsman who is not affiliated with any sports team to play in a tournament

2.5 ADVANTAGES OF PROPOSED SYSTEM

In the proposed system, the user approved by the admin can be selected to the team and can play the tournament. It has got the following features:

- **The system selects the players in the team:-**After the team is added admin select captain to the team, then the captain can select the team players and register tournament.
- Admin can view team and team member details.
- Admin can schedule tournament and publish result
- Players will receive an email after player being added to the team.
- Players can view tournament result and players individual score
- Players can view tournament related news and videos.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

A feasibility study is conducted to determine whether the project will, upon completion, fulfill the objectives of the organization in relation to the work, effort, and time invested in it. A feasibility study enables the developer to predict the project's usefulness and potential future. A system proposal is evaluated for viability based on its influence on the organization, ability to satisfy user needs, and efficient use of resources. As a result, before a new application is accepted for development, it often undergoes a feasibility assessment.

The document assesses the project's viability and cites a number of factors that were carefully taken into account throughout the feasibility assessment, including Technical, Economic and Operational feasibilities. The following are its features: -

3.1.1 Economical Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

The cost of project, Go-Play was divided according to the system used, its development cost and cost for hosting the project. According to all the calculations the project was developed in a low cost. As it is completely developed using open source software.

3.1.2 Technical Feasibility

The system must first undergo a technical evaluation. This feasibility must be built on an outline design of the system's requirements for input, output, programmes, and procedures. Following the identification of an outline system, the inquiry must advise on the type of equipment, the required process for building the system, and the techniques of operating the system once it has been designed. The investigation's technical issues include:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be created in such a way that the required performance and functionality are met within the limitations. The system may still be used even though the technology may become outdated after a while because a newer version of the same software still works with an earlier version. Therefore, this project only has a few limitations. The system was created using PHP for the front end and a MySQL server for the back end; it is technically feasible to complete the project. The system was created using PHP for the front end and a MySQL server for the back end; it is technically feasible to complete the project. The system used had a strong processor, an Intel i3 core, 8GB of RAM, and a 1TB hard drive.

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

3.2 SYSTEMS PECIFICATION

3.2.1 Hardware Specification

Processor - Intel core i3

RAM - 8GB

Hard disk - 1TB

3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used-JS, HTML5, AJAX, J Query, PHP,CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server-side scripting language used for general-purpose programming as well as web development. More than 244 million websites and 2.1 million web servers currently use PHP. The PHP group now produces the reference implementation of PHP, which was first developed by Rasmus Ledford in 1995. PHP, a recursive acronym that once stood for personal home page, now stands for hypertext preprocessor. A web server's PHP processor module interprets PHP code to produce the final web page. In order to handle data, PHP commands can be directly inserted into an HTML source file rather than calling an external file. Additionally, it has developed to now provide a command-line interface and be used in standalone incompatible with the GNU General Public License(GPL)due to restrictions on the usage of the term PHP.PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

3.3.2 MySQL

MySQL, the Oracle Corporation created, distributed, and provided support for MySQL, the most widely used Open Source SQL database management system. The most recent information on MySQL software is available on the MySQL Website..

- **MySQL is a database management system.**

A systematic collection of data is called a database. It might be anything, such as a straightforward grocery list, a photo gallery, or the enormous amount of data in a business network. A database management system like MySQL Server is required in order to add, access, and process data that is stored in a computer database. Database management systems, whether used as stand-alone programmes or as a component of other applications, are essential to computing because computers are excellent at processing vast volumes of data. MySQL databases are relational.

Instead of placing all the data in one huge warehouse, a relational database stores the data in individual tables. Physical files that are optimised for speed contain the database structures. The logical model provides a flexible programming environment with objects like databases, tables, views, rows, and columns. One-to-one, one-to-many, unique, compulsory or optional, and "pointers" between distinct tables are a few examples of the rules you might build up to regulate the relationships between various data fields. These guidelines are upheld by the database, ensuring that your application never encounters inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of "MySQL" stands for "Structured Query Language". The most popular standard language for accessing databases is SQL. Depending on your programming environment, you might explicitly enter SQL (for example, to generate reports), incorporate SQL statements into other languages' code, or use a language-specific API that obscures the SQL syntax. By way of the ANSI/ISO SQL Standard, SQL is defined. Since its inception in 1986, the SQL standard has undergone multiple revisions. In this document, "SQL92" refers to the 1992 standard, "SQL: 1999" to the 1999 standard, and "SQL: 2003" to the most recent version of the standard. The SQL Standard as it exists at any one time is referred to as "the SQL standard."

- **MySQL software is OpenSource.**

Anyone can use and modify the software because it is open source. The MySQL software is available for free download and usage online by anyone. You are free to examine the source code and modify it as necessary. The GPL (GNU General Public License) is used by the MySQL software to specify what you are allowed to do and are not allowed to do with the software in certain circumstances. You can purchase a commercially licenced version from us if the GPL makes you uncomfortable or if you need to integrate MySQL code into a for-profit application. For further details, see the MySQL Licensing Overview.

- **The MySQL Database Server is very fast, reliable, scalable, and easy touse.**

If You ought to give it a shot if that is what you're after. In addition to your other apps, web servers, and other software, MySQL Server can function smoothly on a desktop or laptop while requiring little to no maintenance. You can modify the settings to utilise all the RAM, CPU power, and I/O capacity if you dedicate an entire machine to MySQL..

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that includes a multi-threaded SQL server that supports several client programmes and libraries, administration tools, and a broad variety of application programming interfaces (APIs). Additionally, we offer MySQL Server as a built-in multi-threaded library that you can connect into your application to obtain a smaller, faster, and simpler to administer standalone product.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Any engineered system or product's development process begins with design. A creative process is design. The secret to an efficient system is a decent design. The process of using different methodologies and concepts to specify a process or a system in enough detail to allow for its physical implementation is referred to as "design." One way to describe it is as the process of using different methodologies and concepts to specify a device, a process, or a system in enough detail to allow for its physical reality. Regardless of the development paradigm that is employed, software design forms the technical core of the software engineering process. The system design creates the necessary architectural detail. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 UMLDIAGRAM

A common language known as UML is used to specify, visualize, build, and document the software system artifacts. The Object Management Group (OMG) was responsible for developing UML, and a draught of the UML 1.0 definition was presented to the OMG in January 1997..

Unified Modeling Language is known as UML. Compared to other popular programming languages like C++, Java, COBOL, etc., UML is unique. A visual language called UML is used to create software blueprints. A general-purpose visual modeling language for software system visualization, specification, construction, and documentation is what UML is known as. UML is not just used to represent software systems, despite the fact that this is its most common application. It is also used to model systems that are not software-based. For instance, the manufacturing facility's process flow, etc. Although UML is not a programming language, tools can be used to generate code using UML diagrams in a variety of languages. The analysis and design of objects-oriented systems are directly related to UML. UML has been standardised to the point where it is now an

OMG standard. A comprehensive UML diagram that depicts a system is made up of all the elements and relationships. The most crucial aspect of the entire procedure is the UML diagram's aesthetic impact. It is completed by using all the additional components. The following nine diagrams are part of UML.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

A use case diagram is a visual representation of the interactions between system components. A approach for identifying, outlining, and organizing system requirements is called a use case. The word "system" here refers to a thing that is being created or run, like a website for mail-order product sales and services. UML (Unified Modeling Language), a standard language for the modeling of real-world objects and systems, uses use case diagrams.

A use case diagram contains four components.

- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which are the specific roles are played by the actors within and around the system.
- The relationships between and among the actors and the use cases.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient

use case diagram

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

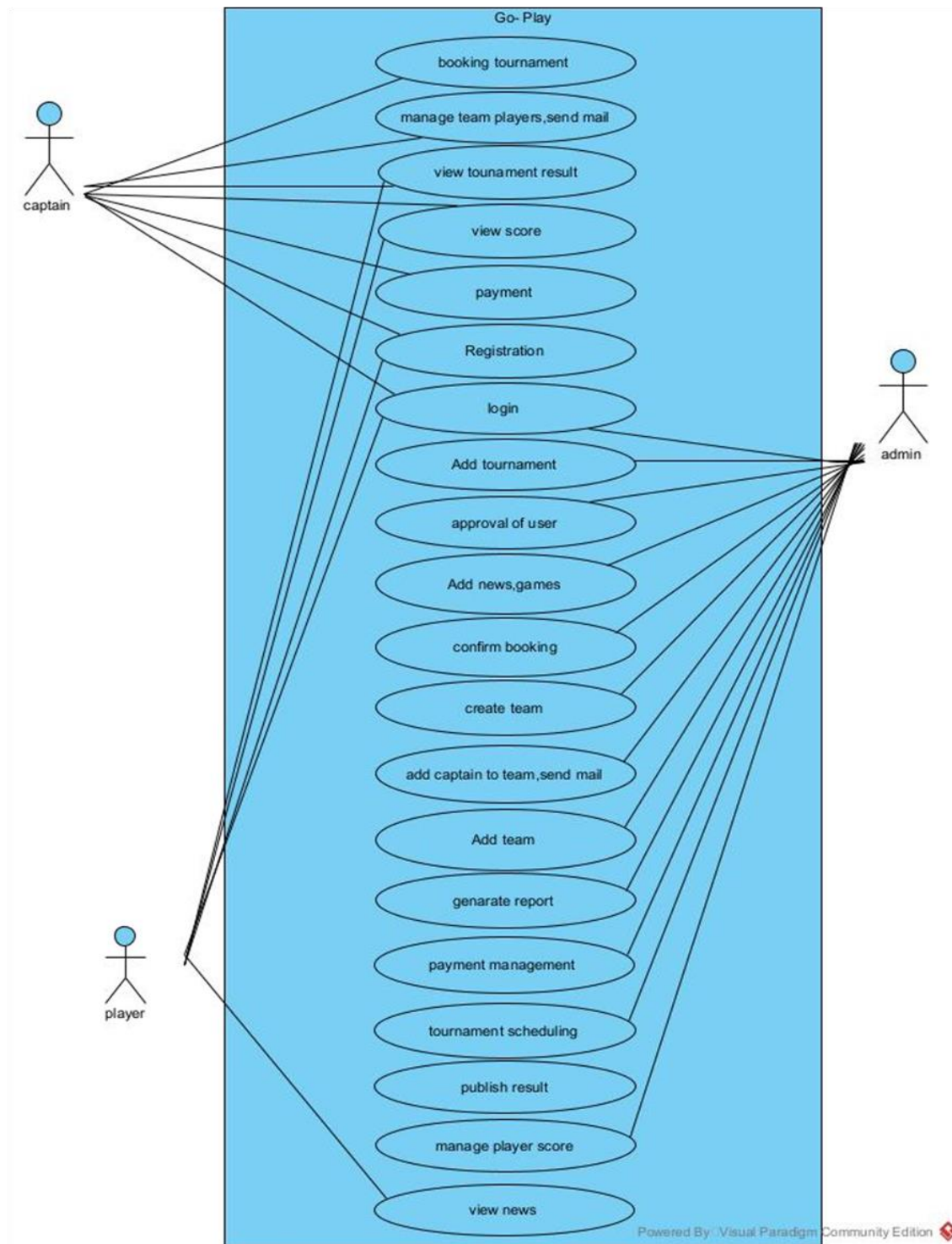


Fig 1 : Use case diagram for Go-Play

4.2.2 SEQUENCE DIAGRAM

A sequence diagram essentially shows how things interact with one another sequentially, or the order in which these interactions occur. A sequence diagram can also be referred to as event diagrams or event scenarios. Sequence diagrams show the actions taken by the components of a system in chronological order. Businesspeople and software engineers frequently use these diagrams to record and comprehend the requirements for new and current systems.

Sequence Diagram Notations –

- i. **Actors** – In a UML diagram, an actor represents a particular kind of role in which it communicates with the system's objects. An actor is always beyond the purview of the system that we want to use the UML diagram to represent. We employ actors to portray a variety of roles, including those of human users and other outside subjects. In a UML diagram, an actor is represented using a stick person notation. In a sequence diagram, there might be several actors.
- ii. **Lifelines** – A named piece that shows a specific participant in a sequence diagram is called a lifeline. In essence, a lifeline represents each incident in a sequence diagram. The lifeline components in a sequence diagram are at the top.
- iii. **Messages** – Using messages, communication between objects is demonstrated. The messages are displayed on the lifeline in chronological sequence. Arrows are how messages are represented. A sequence diagram's main components are lifelines and messages.

The following categories serve as general classifications for messages:

- **Synchronous messages**
- **Asynchronous Messages**
- **Create message**
- **Delete Message**
- **Self-Message**
- **Reply Message**

- **Found Message**
- **Lost Message**

iv. Guards – In UML, guards are used to model circumstances. They are employed when we need to impose a restriction on the flow of messages under the guise of a fulfilment of a condition. Guards are crucial in informing software developers of the limitations imposed by a system or specific procedure..

Uses of sequence diagrams –

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.

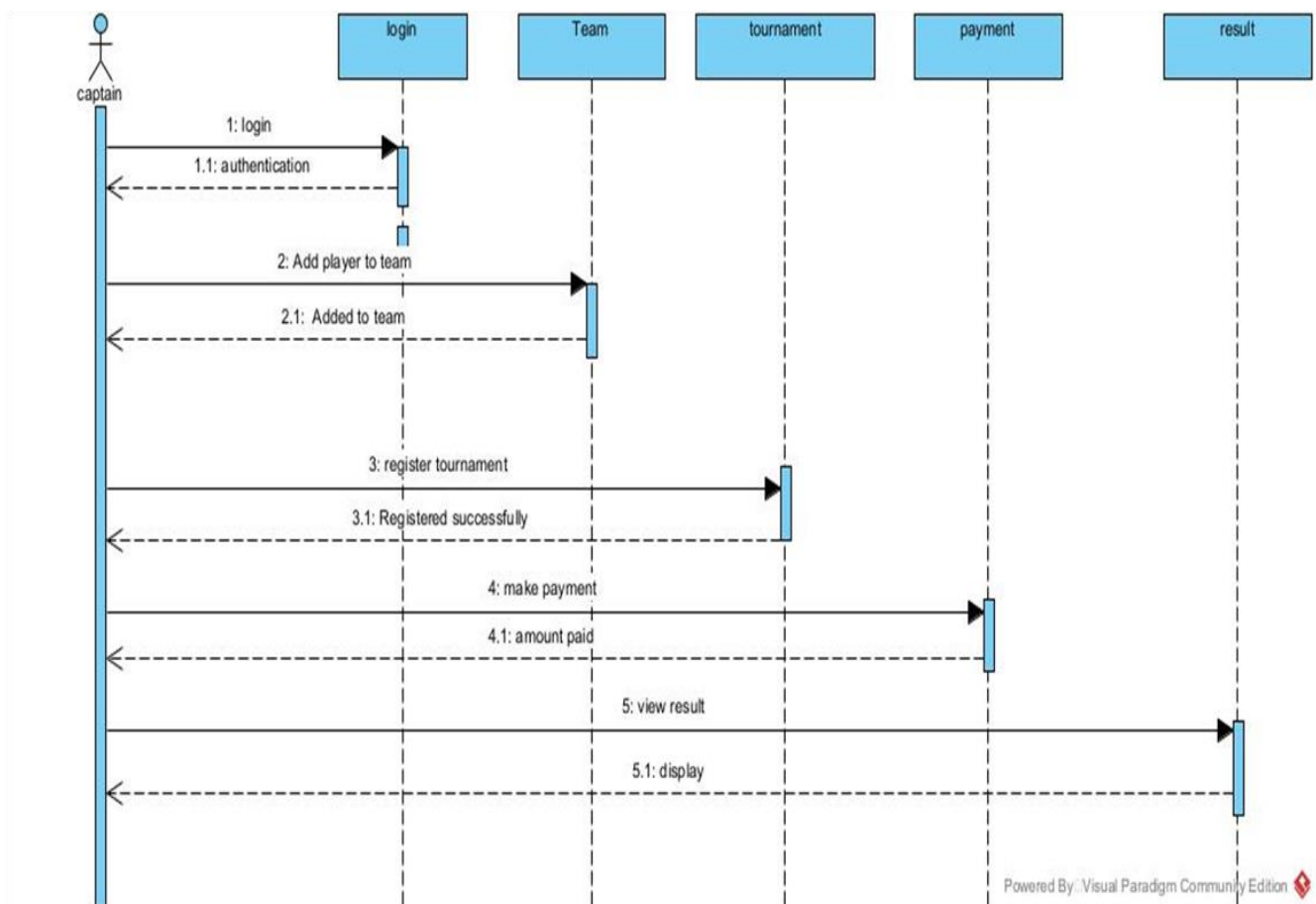


Fig 2 : Sequence diagram for Go-Play(captain)

4.2.3 STATE DIAGRAM

State chart diagrams are used to depict how a software system behaves. A class, a subsystem, a package, or even a complete system's behavior can be represented by a state machine diagram in a UML model. State charts and state transition diagrams are other names for it. State chart diagrams give us a useful approach to represent the communications or interactions that take place between external entities and a system. The event-based system is modeled using these diagrams. With the aid of an event, a state of an object can be managed. State chart diagrams are used to describe various states of an entity within the application

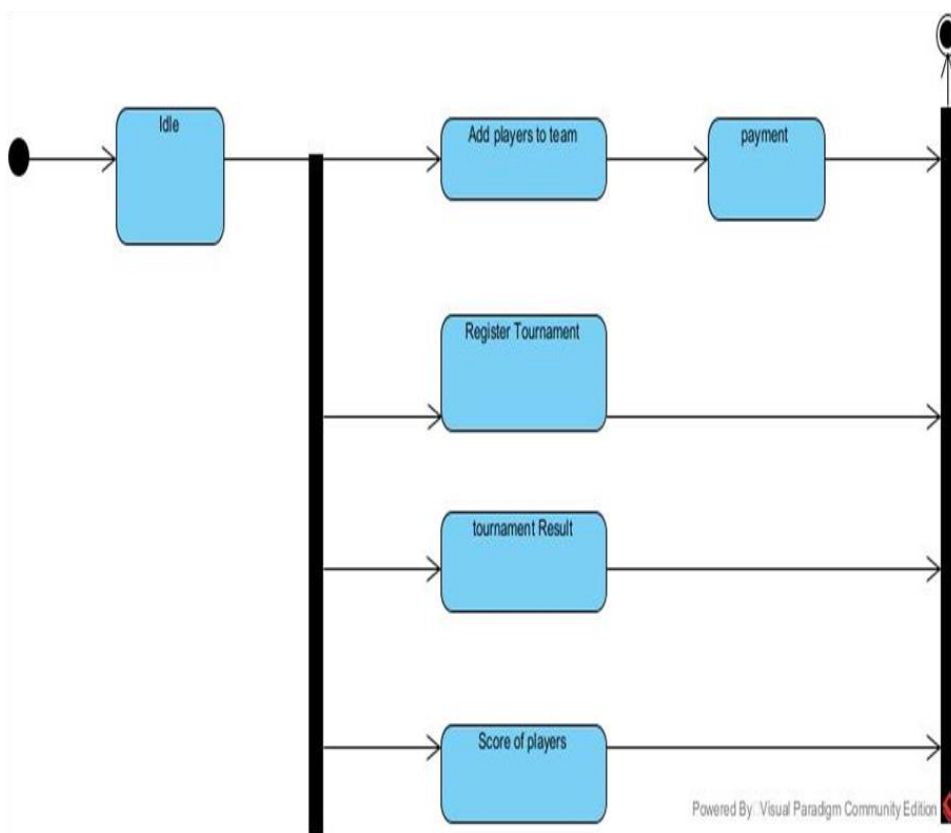


Fig 3 : State chart diagram for Go-Play(captain)

4.2.4 ACTIVITYDIAGRAM

Activity diagrams show how multiple levels of abstraction of activities are coordinated to produce a service. Typically, an event must be accomplished by some operations, especially when the operation is meant to accomplish several different things that call for coordination. Another common requirement is how the events in a single use case relate to one another, especially in use cases where activities may overlap and require coordination.

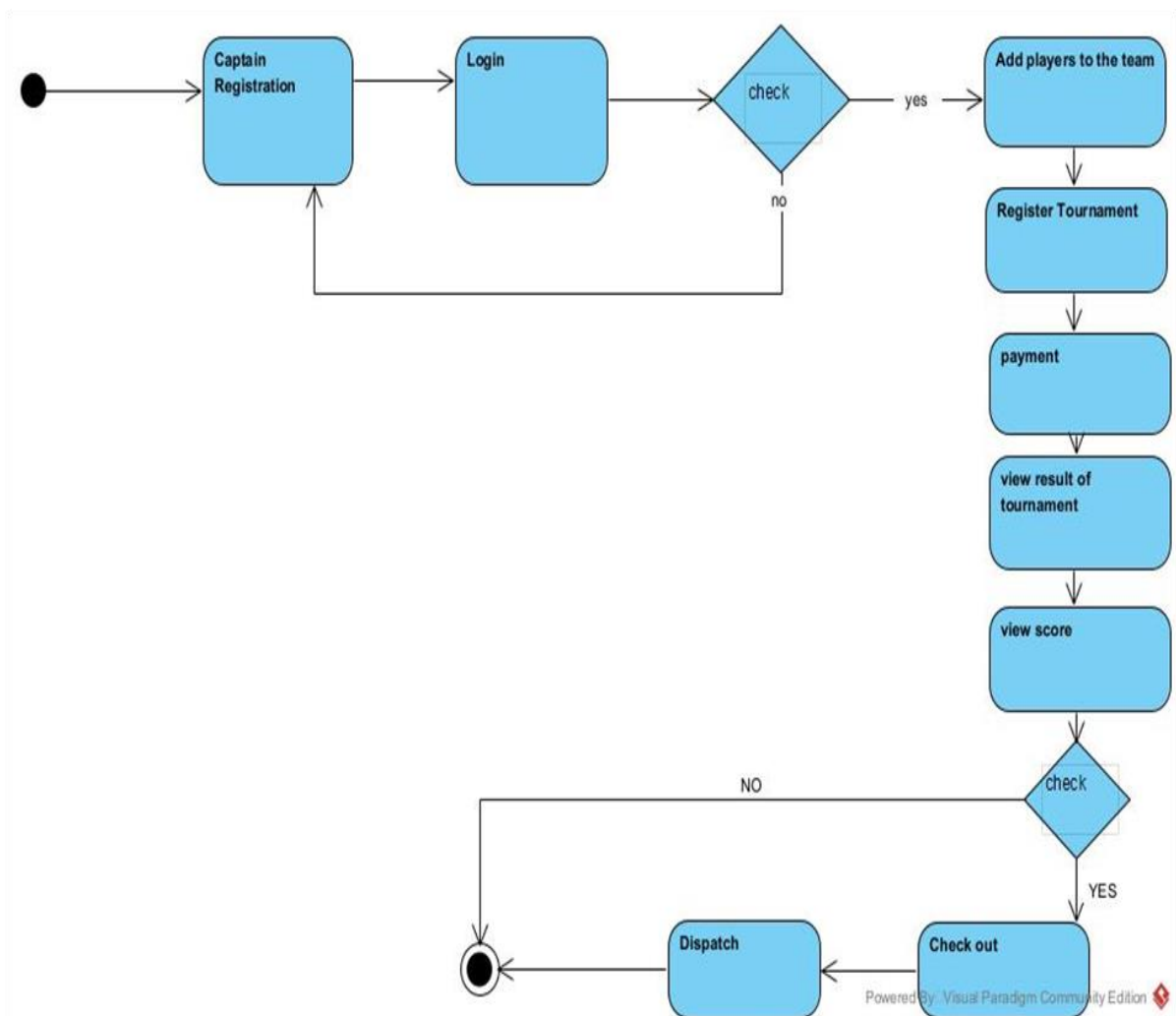


Fig 4:Activity diagram of captain for Go-Play(captain)

4.2.5 CLASSDIAGRAM

Static diagrams include class diagrams. It represents the application's static view. Class diagrams are used to create executable code for software applications as well as for visualising, explaining, and documenting various elements of systems. The characteristics and functions of a class are described in a class diagram, along with the restrictions placed on the system.

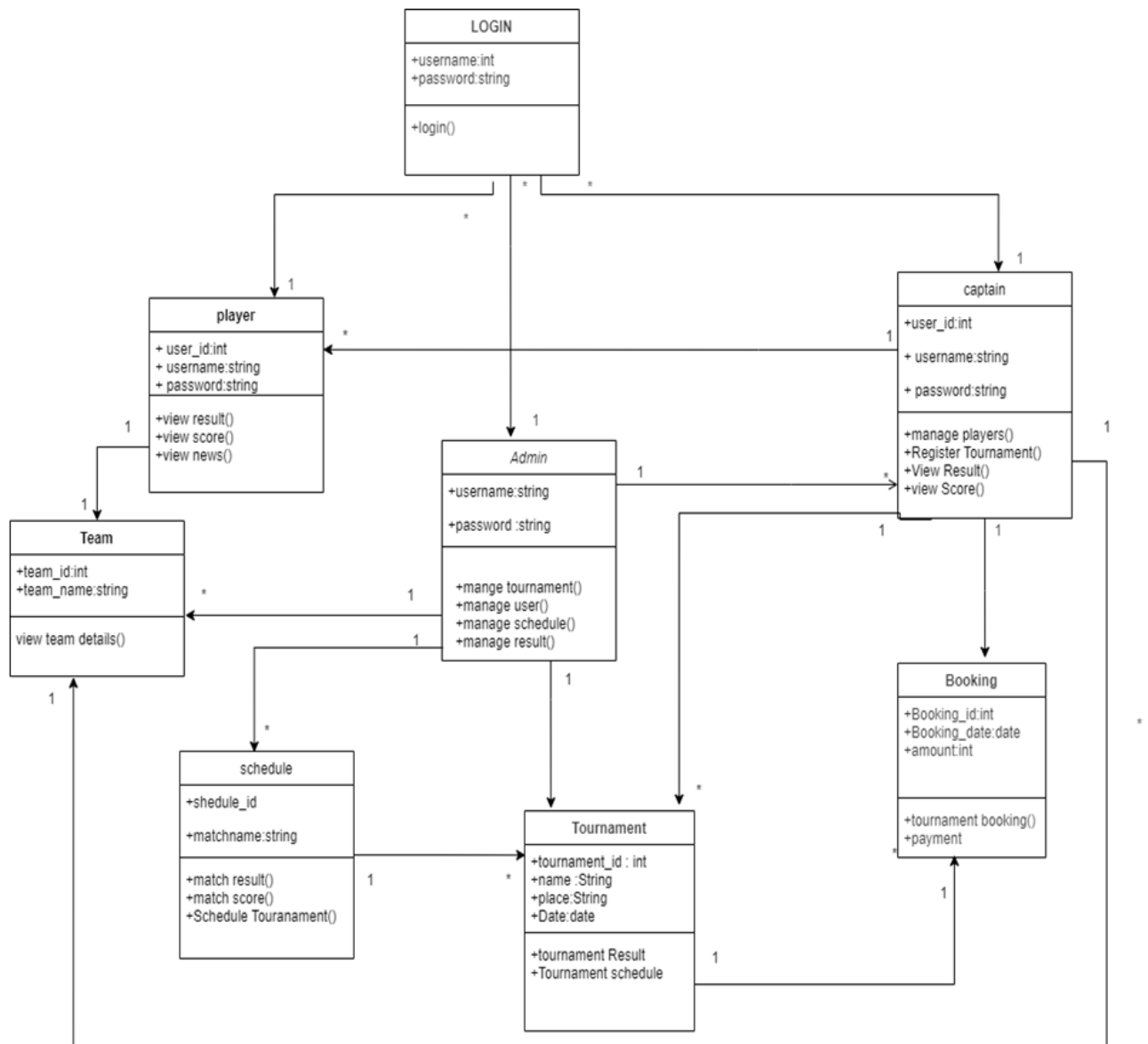


Fig 5: class diagram for Go-Play

4.2.6 OBJECT DIAGRAM

Since class diagrams are the source of object diagrams, they are dependent class diagrams. An instance of a class diagram is represented by an object diagram. Class and object diagrams both use the same fundamental ideas. The static view of a system is also represented by object diagrams, but this static view represents a momentary snapshot of the system. To represent a group of items and their connections as an instance, object diagrams are employed.

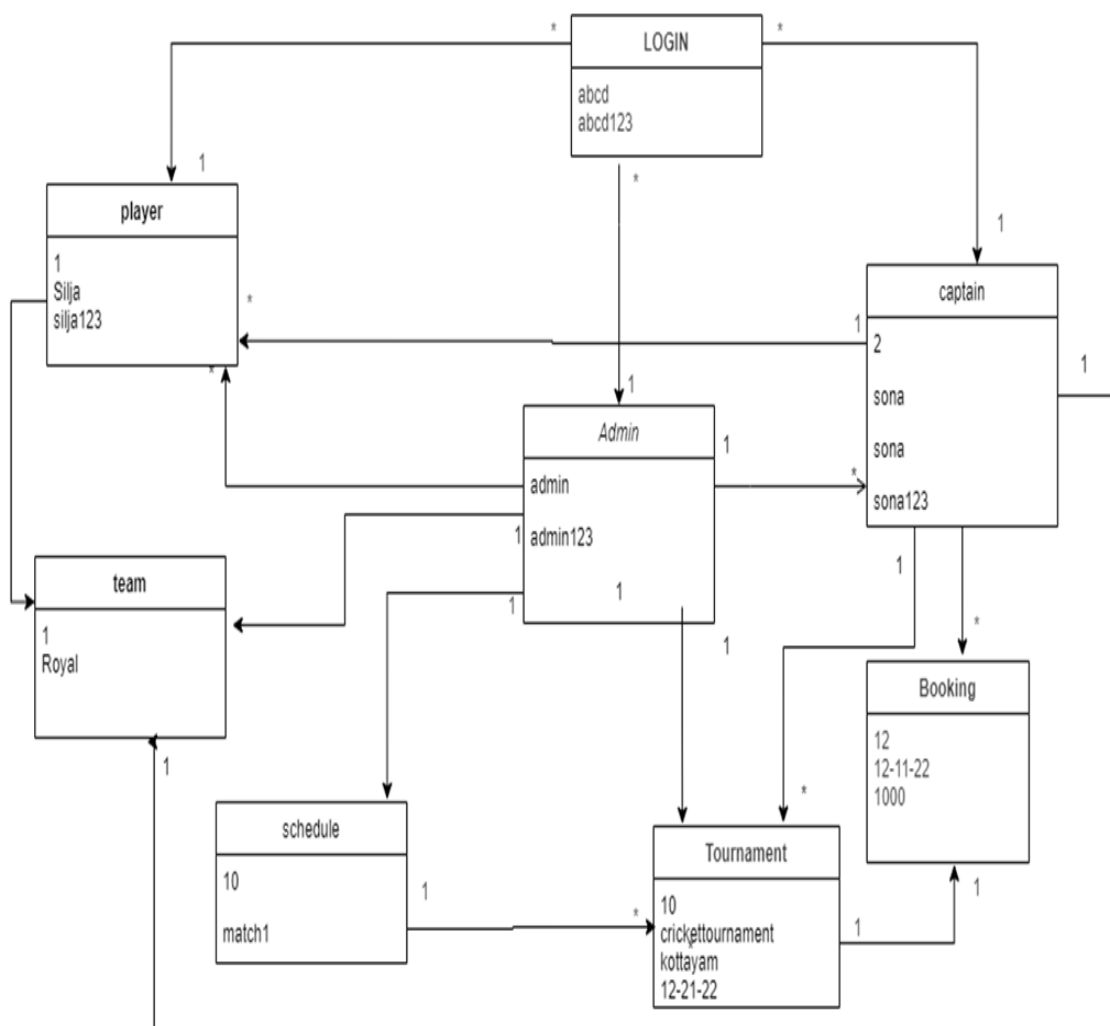


Fig 6: object diagram for Go-Play

4.2.7 COMPONENT DIAGRAM

A component diagram, sometimes called a UML component diagram, shows how the physical components of a system are wired together and organised. To model implementation specifics and ensure that all necessary functionalities of the system are covered by planned development, component diagrams are frequently created.

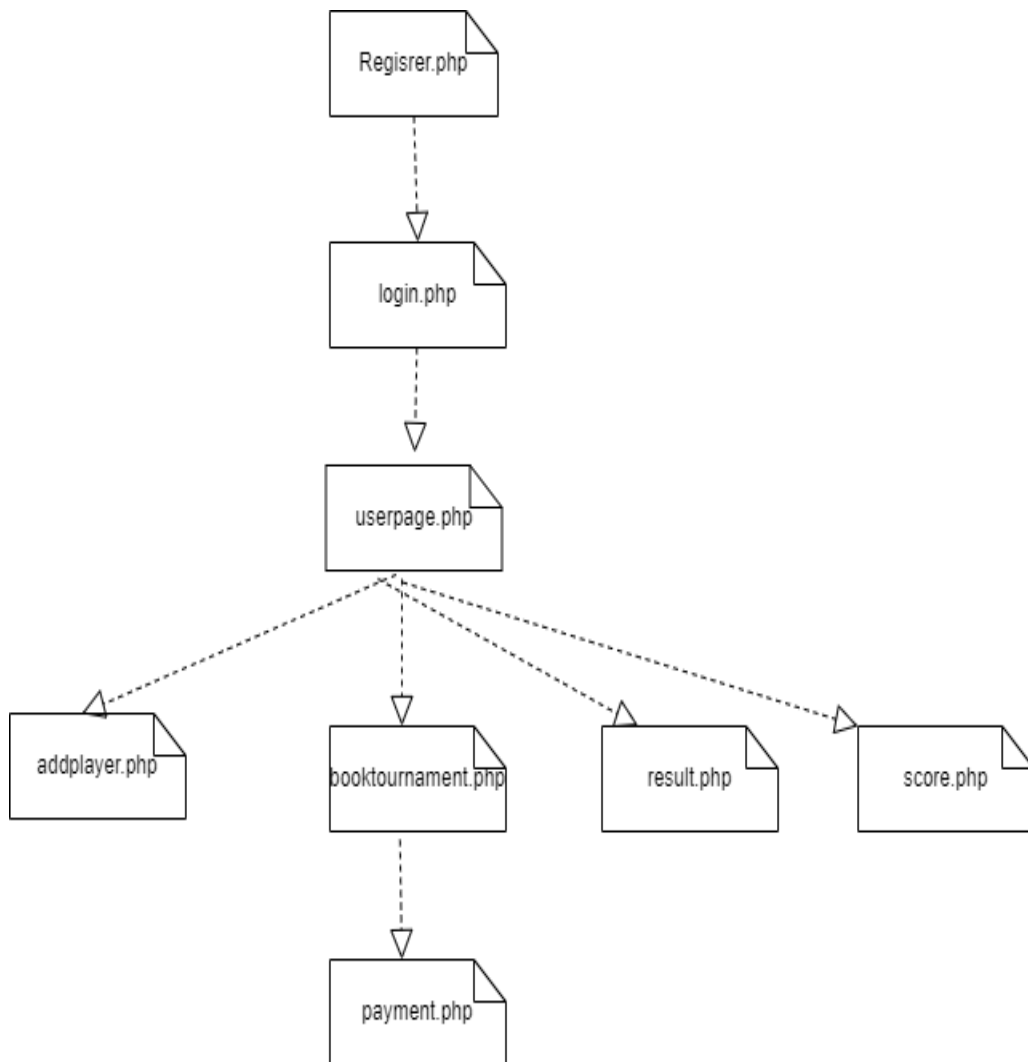


Fig 7: component diagram for Go-Play(captain)

4.2.8. DEPLOYMENT DIAGRAM

An execution architecture of a system, containing nodes like hardware or software execution environments, and the middleware linking them, is shown in a deployment diagram, a form of UML diagram. Typically, deployment diagrams are used to represent the actual hardware and software of a system.

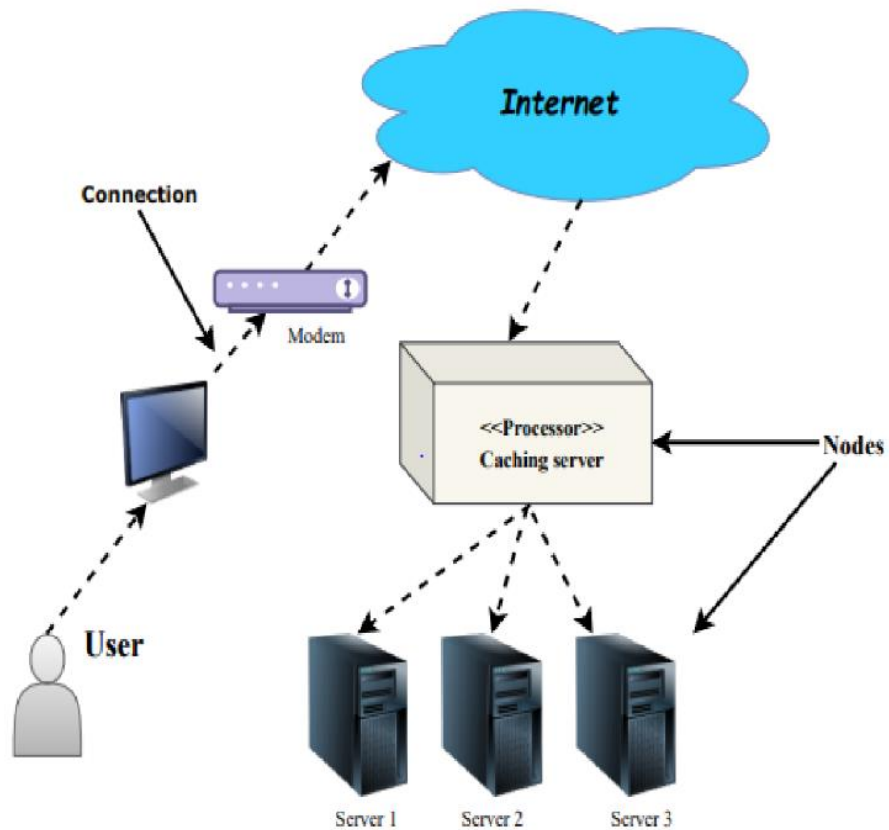
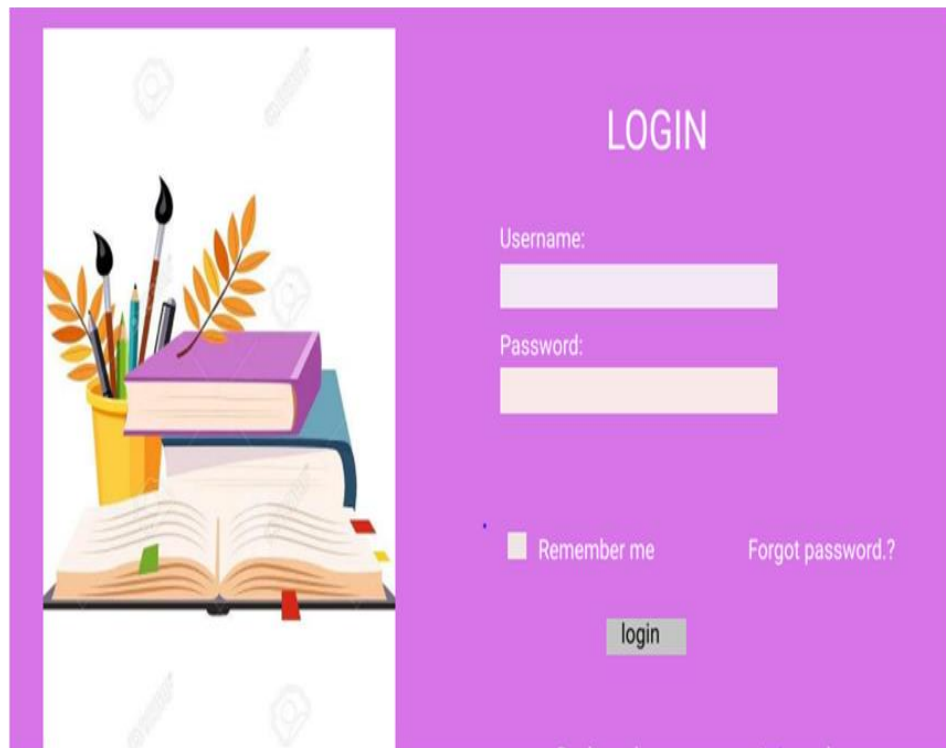


Fig 8: Deployment diagram for Go-Play

4.5 USER INTERFACE DESIGN

4.5.1-INPUT DESIGN

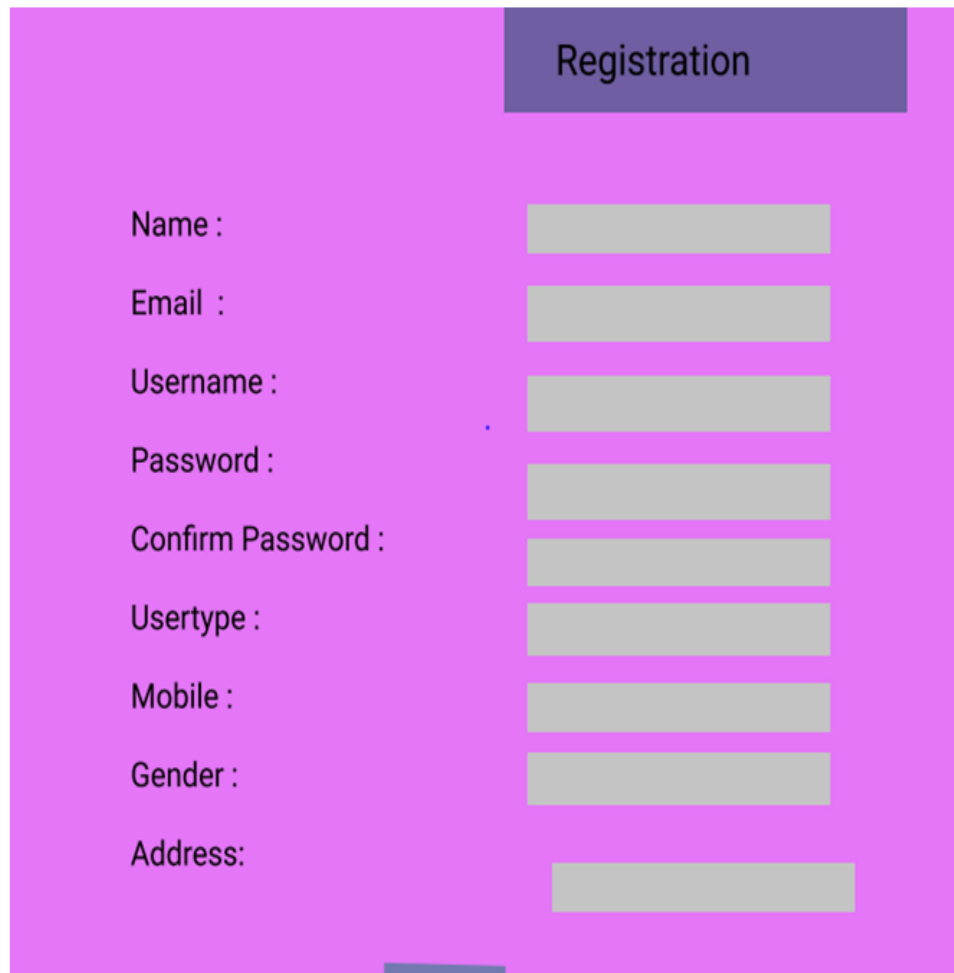
Form Name : User Login



The image displays a user login interface for 'Go-Play'. It is divided into two main sections by a vertical purple bar. The left section features a white background with a purple border, containing an illustration of a yellow pencil holder with pens and pencils, a stack of books, and an open book. The right section has a solid purple background. At the top right of this section, the word 'LOGIN' is written in white. Below it, there are two input fields: 'Username:' with a light purple box and 'Password:' with a light orange box. Under the password field, there is a checkbox labeled 'Remember me' and a link 'Forgot password?'. At the bottom center of the purple section is a grey button with the text 'login'.

Fig 9: login input design for Go-Play

Form Name : User Registration



The image shows a registration form titled "Registration" in a dark blue header. The form is set against a light blue background. It contains several input fields for user registration, each preceded by a label. The labels are: "Name :", "Email :", "Username :", "Password :", "Confirm Password :", "Usertype :", "Mobile :", "Gender :", and "Address:". The input fields are represented by light blue rectangles. The "Address:" label is followed by a wider input field. At the bottom right, there is a small dark blue button.

Registration	
Name :	<input type="text"/>
Email :	<input type="text"/>
Username :	<input type="text"/>
Password :	<input type="password"/>
Confirm Password :	<input type="password"/>
Usertype :	<input type="text"/>
Mobile :	<input type="text"/>
Gender :	<input type="text"/>
Address:	<input type="text"/>
<input type="button" value="Submit"/>	

Fig 10: Registration input design for Go-Play

4.5.2 OUTPUT DESIGN

User Login

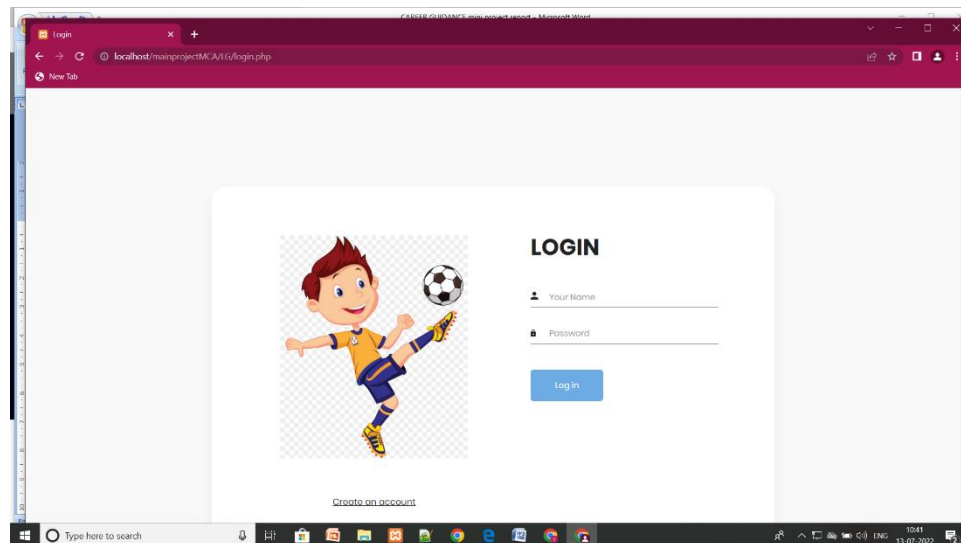


Fig 11: login output design for Go-Play

User Registration

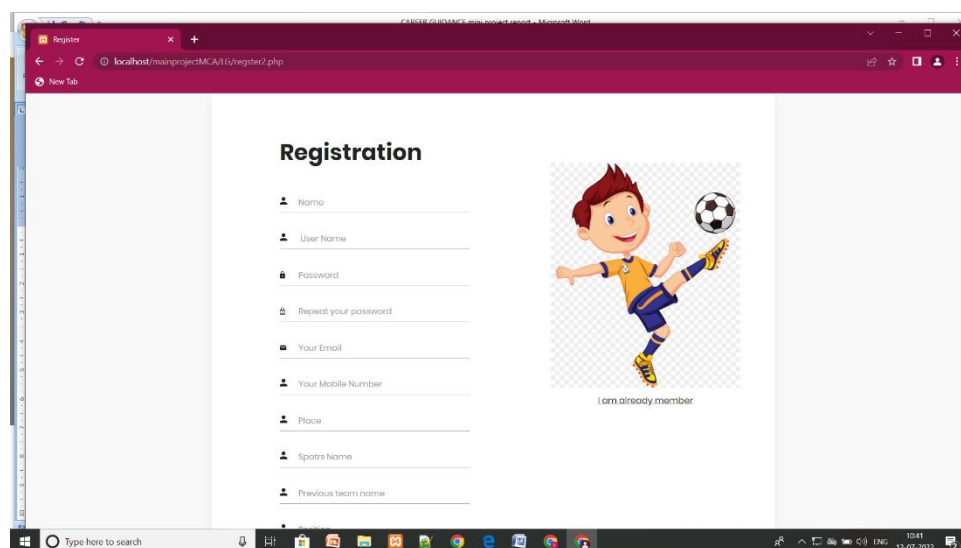


Fig 12: Registration output design for Go-Play

4.6. DATABASE DESIGN

A database is a structured system with the capacity to store information and allows users to retrieve stored information quickly and effectively. Any database's primary goal is its data, which demands protection.

There are two stages to the database design process. The user needs are obtained in the first step, and a database is created to as clearly as possible meet these criteria. Information Level Design is the name of this stage, which is carried out independently of any specific DBMS.

This information level design is converted into a design for the specific DBMS that will be used to implement the system in question in the second stage. This process, known as physical level design, is focused on the features of the particular DBMS that will be used. Parallel to the system design is a database design. The database's data arrangement aims to accomplish the two main goals listed below.

- Data Integrity
- Data independence

4.6.1 Relational Database Management System (RDBMS)

In a relational model, the database is shown as a set of relations. Each relation resembles a file or table of records with values. A row is referred to as a tuple, a column heading is referred to as an attribute, and the table is referred to as a relation in formal relational model language. A relational database is made up of a number of tables, each of which has a special name. In a story, each row represents a group of associated values.

Relations, Domains & Attributes

A relation is a table. Tuples are the units of a table's rows. An ordered group of n elements is a tuple. Attributes are referred to as columns. Every table in the database has relationships already established between them. This guarantees the integrity of both referential and entity relationships. A group of atomic values make up a domain D . Choosing a data type from which the domain's data values are derived is a typical way to define a domain. To make it easier to understand the values of the domain, it is also helpful to give it a name. Every value in a relation is atomic.

Relationships

- Key is used to create table relationships. Primary Key and Foreign Key are the two principal keys that are most crucial. With the use of these keys, relationships for entity integrity and referential integrity can be created..
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

4.6.2 Normalization

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modeling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are the two different kinds of keys. A primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized.

It means placing things in their natural form, as the name suggests. By using normalization, the application developer aims to establish a coherent arrangement of the data into appropriate tables and columns, where names may be quickly related to the data by the user. By removing recurring groups from the data, normalization prevents data redundancy, which puts a heavy strain on the computer's resources. These include:

- Normalize the data.
- Choose proper names for the tables and columns.
- Choose the proper name for the data.

First Normal Form

According to the First Normal Form, each attribute's domain must only include atomic values, and each attribute's value in a tuple must be a single value from that domain. In other words, 1NF forbids using relationships as attribute values within tuples or relations within relations. Single atomic or indivisible values are the only attribute values that are permitted under 1NF. The data must first be entered into First Normal Form. This can be accomplished by separating data into tables of a similar type in each table. Depending on the needs of the project, a Primary Key or Foreign Key is assigned to each table. For each nested relation or non-atomic attribute, new relations are formed in this process. This got rid of data groups that were repeated. If a relation solely meets the constraints that include the primary key, it is said to be in first normal form.

Second Normal Form

In relations when the primary key includes more than one attribute, no non-key attribute should be functionally dependent on a component of the main key. In this step, we dissect each partial key and create a fresh connection with its dependent attributes. All characteristics that are totally dependent on the primary key should still be connected to it. Data that only depends on a small amount of the key can be removed with the help of this approach. If and only if the relation satisfies all the first normal form criteria for the primary key, it is said to be in second normal form since every non-primary key feature of a relation is entirely dependent on its primary key alone.

Third Normal Form

In relations when the primary key includes more than one attribute, no non-key attribute should be functionally dependent on a component of the main key. In this step, we dissect each partial key and create a fresh connection with its dependent attributes. All characteristics that are totally dependent on the primary key should still be connected to it. Data that only depends on a small amount of the key can be removed with the help of this approach. If and only if the relation satisfies all the first normal form criteria for the primary key, it is said to be in second normal form since every non-primary key feature of a relation is entirely dependent on its primary key alone.

TABLE DESIGN

Table No **01**
Table Name **: tb_login**
Primary Key **: login_id**
Table Description : To store user Login information

Field Name	DataType	Constraints	Description
login_id	Bigint(20)	PrimaryKey	Login id
Username	Varchar(50)	Unique	User name
Password	Varchar(20)	Notnull	Password
UserType	Varchar(20)	Notnull	User type
Status	Int(5)	Notnull	Status
approve	Varchar(20)		Approve user

Table No **02**
Table Name **: tb_user**
Primary Key **: user_id**
Foreign Key **: log_id,**
Table Description: To store user registration information

Field Name	DataType	Constraints	Description
User_id	Int	PK	User id
Logid	Int	FK	Tb_login table id
Name	Varchar(100)		Name of user
Mobile	Int		Phone number
Email	Varchar(50)		Email id
Place	Varchar(100)		Place
Sp_type	Varchar(50)		Sports type
Pt_name	Varchar(100)		Previous team name
Position	Varchar(100)		Position of player
Exp	Varchar(100)		Experience
Eligibility	Varchar(100)		Eligibility of player

Table No **03**
Table Name **: tb_news**
Primary Key **: n_id**

Table Description: To store News related to tournament

Field Name	DataType	Constraints	Description
News_id	Int(5)	PK	News id
Title	Varchar(50)		Title of news
Description	Varchar(200)		News description

Table No **04**
Table Name **: tb_tournament**
Primary Key **: trmn_id**

Table Description: To store tournament information

Field Name	DataType	Constraints	Description
T_id	Int(5)	PK	Tournament id
T_name	Varchar(100)		Tournament name
Sports name	Varchar(50)		Name sports
Place	Varchar(50)		Place
Stadium	Varchar(50)		Name of stadium
Time	Time		Time
Date	Date		Date
price	Date		Registration Fee

Table No **05**
Table Name **:tb_teams**
Primary Key **: team_id**
Foreign Key **: user_id**
Table Description **: To store team details**

Field Name	DataType	Constraints	Description
team_id	Int(5)	PK	Team id
User_id	Int(5)	Fk	tb_user table id
Temname	Varchar(100)		Name of the team
Mobile	Int(15)		Mobile number

Email	Varchar(50)		Email id
Place	Varchar(50)		Place
Status	Int(5)		Status of team

Table No **06**

Table Name : **tb_Games**

Primary Key : **G_id**

Table Description: To store game details

Field Name	Data Type	Constraints	Description
g_id	Int(5)	PK	game id
Name	Varchar(50)		Name
No of ptc	Int(5)		No of participants
Rules	Varchar(100)		Rules of game

Table No **07**

Table Name : **tb_feedback**

Primary Key : **f_id**

Table Description: To store feedback details

Field Name	Data Type	Constraints	Description
F_id	Int	PK	Feedback id
User_id	Int	FK	tb_User table id
Feedback	Varchar(100)		User feedback

Table No **09**

Table Name : **tb_booking**

Primary Key : **booking_id**

Table Description: To store tournament booking details

Field Name	Data Type	Constraints	Description
Booking_id	Int(10)	PK	Booking id
trmn_id	Int(10)	FK	Tb_tournament table id
team_id	Int(10)	Fk	Tb_team table id

Status	Int(10)		Status
cancelreq	varchar(40)		Cancel request

Table No **10**
Table Name **: tb_teamplayers**
Primary Key **: tp_id**

Table Description: To store team players details

Field Name	Data Type	Constraints	Description
TP_id	Int(10)	PK	Team player id
Team_id	Int(10)	FK	Team id from tb_team
User_id	Int(10)	FK	User_id from tb_user
Status	Int(10)		Status

Table No **12**
Table Name **: tb_matchschedule**
Primary Key **: ms_id**

Table Description: To store match schedule details

Field Name	Data Type	Constraints	Description
ms_id	Int(10)	PK	Match schedule id
Trmn_id	Int(10)	FK	Tournament id
Match	Varchar(50)		Match name
Team1	Varchar(50)		First Team name
Team2	Varchar(50)		Second Team name

Table No **13**
Table Name **: tb_matchscore**
Primary Key **: tms_id**

Table Description: To store match score details

Field Name	Data Type	Constraints	Description
Tms_id	Int	PK	Match score id
Ms_id	Int (50)	FK	Match schedule

Team1score	Int (50)		First team score
Team2score	Int (50)		Second team score

Table No **14**
Table Name **: tb_score**
Primary Key **: score_id**

Table Description: To store player's score details

Field Name	Data Type	Constraints	Description
Score_id	Int	PK	Score id
Trmn_id	int(10)	FK	Tb_tournament table id
Team_id	int(10)	FK	Tb_teams table id
User_id	int(10)	FK	Tbuser table id
Score	Score(100)		Score of player

Table No **15**
Table Name **: tb_bookingpay**
Primary Key **: pb_id**

Table Description: To store payment details

Field Name	Data Type	Constraints	Description
Pb_id	Int(5)	PK	Booking payment id
log_id	Int(5)	FK	Login id from tb_login
Book_id	Int(5)	FK	Booking id from tb_booking
T_amount	Int(50)		Amount payment
Paystatus	Varchar(20)		Payment status
Date_added	date		Date of payment

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software testing is the practise of meticulously monitoring the way software is used to see if it works as intended. Two words that are frequently used in conjunction with the term "software testing" are verification and validation. A product, including software, is validated by being examined or evaluated to see if it conforms with all pertinent requirements. Software testing, one sort of verification, also makes use of reviews, analyses, inspections, and walkthroughs. Validation is the process of ensuring that what has been specified matches what the user truly desires.

The processes of static analysis and dynamic analysis are additional ones that are frequently related to software testing. Static analysis examines the software's source code, searching for issues and obtaining statistics without actually running the code. Dynamic analysis examines how software behaves while it is running in order to offer data like execution traces, timing profiles, and test coverage details.

Testing is a collection of activities that can be planned ahead of time and carried out in a methodical manner. Testing starts with individual modules and progresses to the integration of the full computer-based system. There are many rules that can be used as testing objectives, and testing is necessary for the system testing objectives to be successful. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a test is successfully carried out in accordance with the aforementioned aims, it will reveal software bugs. Additionally, testing shows that the software functions seem to be functioning in accordance with the specifications and that the performance requirements seem to have been satisfied.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The activity that is to be taken is outlined in the test plan. A computer programme, its documentation, and associated data structures are all created by software developers. It is always the responsibility of the software developers to test each of the program's separate components to make sure it fulfils the purpose for which it was intended. In order to solve the inherent issues with allowing the builder evaluate what they have developed, there is an independent test group (ITG). Testing's precise goals should be laid forth in quantifiable language. The test strategy should include information on the mean time to failure, cost to detect and correct problems, remaining defect density or frequency of occurrence, and test work hours per regression test. The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Unit testing concentrates verification work on the software component or module, which is the smallest unit of software design. The component level design description is used as a guide when testing significant control paths to find faults that exist inside the module's perimeter. the level of test complexity and the untested area determined for unit testing. Unit testing is white-box focused, and numerous components may be tested simultaneously. To guarantee that data enters and exits the software unit under test properly, the modular interface is tested. To make sure that data temporarily stored retains its integrity during all phases of an algorithm's execution, the local data structure is inspected. . A module's boundary conditions are checked to make sure that each statement has been run at least once. All methods for handling errors are then tested.

Before starting any other test, data flow tests across a module interface must be

completed. Error circumstances must be expected in good design, and methods for handling errors must be put up so that they can be cleanly terminated or rerouted when they do occur. The penultimate phase in the unit testing process is boundary testing. At its limits, software frequently fails.

5.2.2 Integration Testing

Integration testing is a methodical approach for creating the program's structure while also carrying out tests to find interface issues. The goal is to construct a programme structure that has been determined by design using unit tested components. The programme as a whole is tested. Correction is challenging since the size of the overall programme makes it challenging to isolate the causes. As soon as these mistakes are fixed, new ones arise, and the process repeats itself in an apparently unending cycle. All of the modules were integrated after unit testing was completed in the system to check for any interface inconsistencies.

The testing process comes to an end here. This involved testing the entire system in its entirety, including all forms, code, modules, and class modules. Popular names for this type of testing include system tests and black box testing.

The functional requirements of the software are the main emphasis of the black box testing approach. That example, using Black Box testing, a software engineer can create sets of input conditions that will fully test every programme requirement.

5.2.3 Output Testing or User Acceptance Testing

Testing is conducted as stated above using various test data types. In system testing, the preparation of test data is crucial. The system under investigation is tested after the test data has been prepared. Errors were discovered and fixed when testing the system by which test data were collected, and the fixes were also noted for future usage. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The aforementioned testing is carried out using a variety of test data. The preparation of test data is crucial for testing a system. The system under investigation is evaluated utilising the test data after the test data preparation. When testing the system that generates test data, errors are found and fixed using the above testing procedures, and the fixes are also recorded for later use.

Automation Testing

Automation testing is the process of testing software and other tech products to ensure it meets strict requirements. Essentially, it's a test to double-check that the equipment or software does exactly what it was designed to do. It tests for bugs, defects, and any other issues that can arise with product development. Automation testing can be run at any time of the day. It uses scripted sequences to examine the software. It then reports on what's been found, and this information can be compared with earlier test runs.

Benefits of Automation Testing

Detailed reporting capabilities - Automation testing uses well-crafted test cases for various scenarios. These scripted sequences can be incredibly in-depth, and provide detailed reports that simply wouldn't be possible when done by a human.

Improved bug detection - One of the main reasons to test a product is to detect bugs and other defects. Automation testing makes this process an easier one.

5.2.4 Selenium Testing

Selenium is an open-source tool that automates web browsers. It provides a single interface that lets you write test scripts in programming languages like Ruby, Java, NodeJS, PHP, Perl, Python, and C#, among others. The Selenium testing tool is used to automate tests across browsers for web applications. It's used to ensure high-quality web applications —whether they are responsive, progressive, or regular. Selenium is an open-source tool.

Test cases for a Login Page

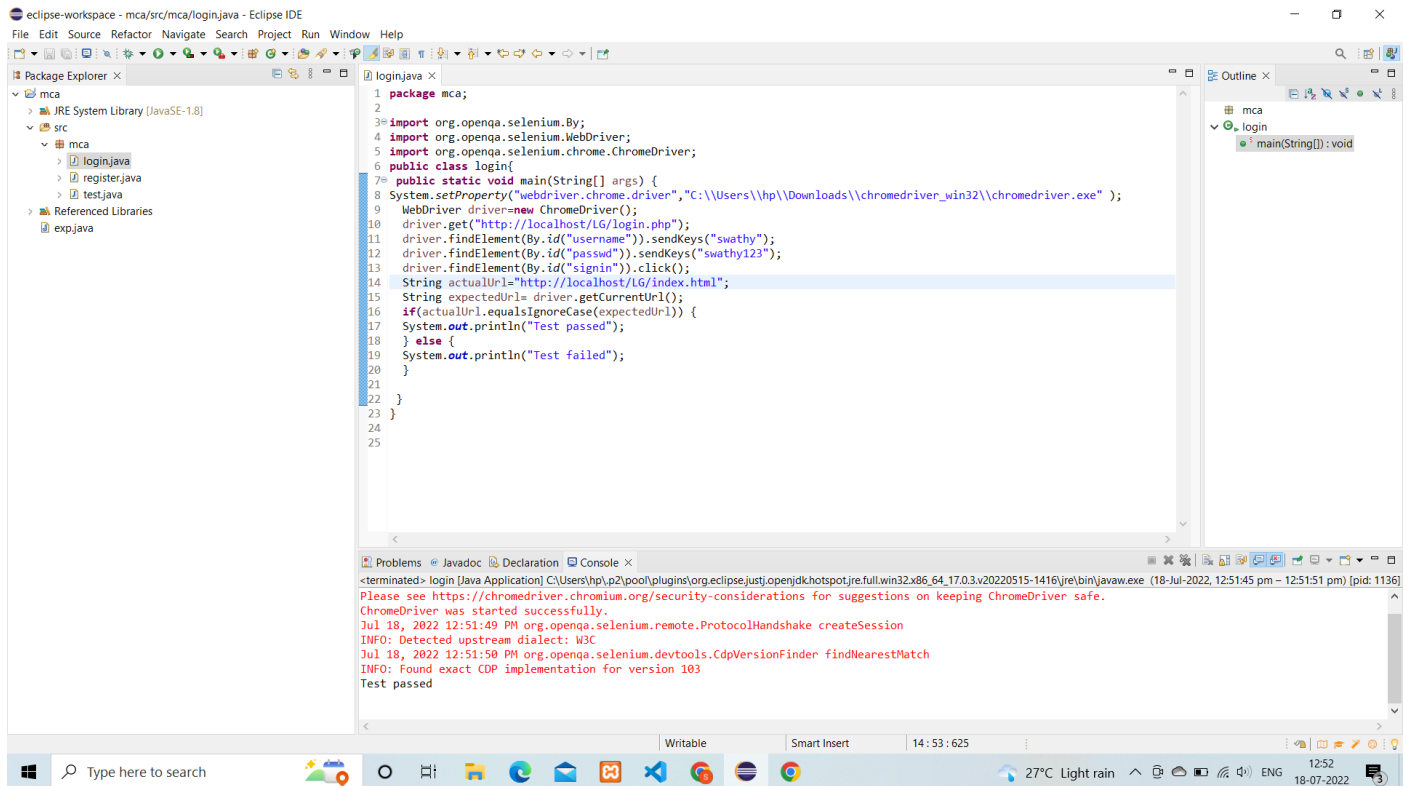
ProjectName:Go-play					
Login Test Case					
Test Case ID:Fun_1			Test Designed By:SILJA CK		
Test Priority (Low/Medium/High):High			Test Designed Date:18-07-2022		
Module Name:Login Screen			Test Executed By:Dr.BIJIMOL T K		
Test Title:Verify login with valid username and Password			Test Execution Date: 18-07-2022		
Description:Test the Login Page					
Pre-Condition:User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login page displayed	Pass
2	Provide Valid Username	Username:swathy	User should be able to Login	User Logged in and Navigated to User Dashboard	Pass
3	Provide Valid Password	Password:swathy123			
4	Click on Sign In Button				
5	Provide Invalid username or password	Username: swathy Password: swathy	User should not be Able to Login	Message for enter valid emailed or Password Displayed	Pass
6	Provide Null Username Or Password	Username: null Password: null			
7	Click on Sign In Button				

Post-Condition: User is validated with database and successfully login into account.
The Account session details are logged in database.

Codepackage

```
package mca;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class login{
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\\\Users\\hp\\Downloads\\chromedriver_wi
n32\\chromedriver.exe" );
        WebDriver driver=new ChromeDriver();
        driver.get("http://localhost/LG/login.php");
        driver.findElement(By.id("username")).sendKeys("swathy");
        driver.findElement(By.id("passwd")).sendKeys("swathy123");
        driver.findElement(By.id("signin")).click();
        String actualUrl="http://localhost/LG/index.html";
        String expectedUrl= driver.getCurrentUrl();
        if(actualUrl.equalsIgnoreCase(expectedUrl)) {
            System.out.println("Test passed");
        } else {
            System.out.println("Test failed");
        }
    }
}
```



Test cases for Go-play Registration

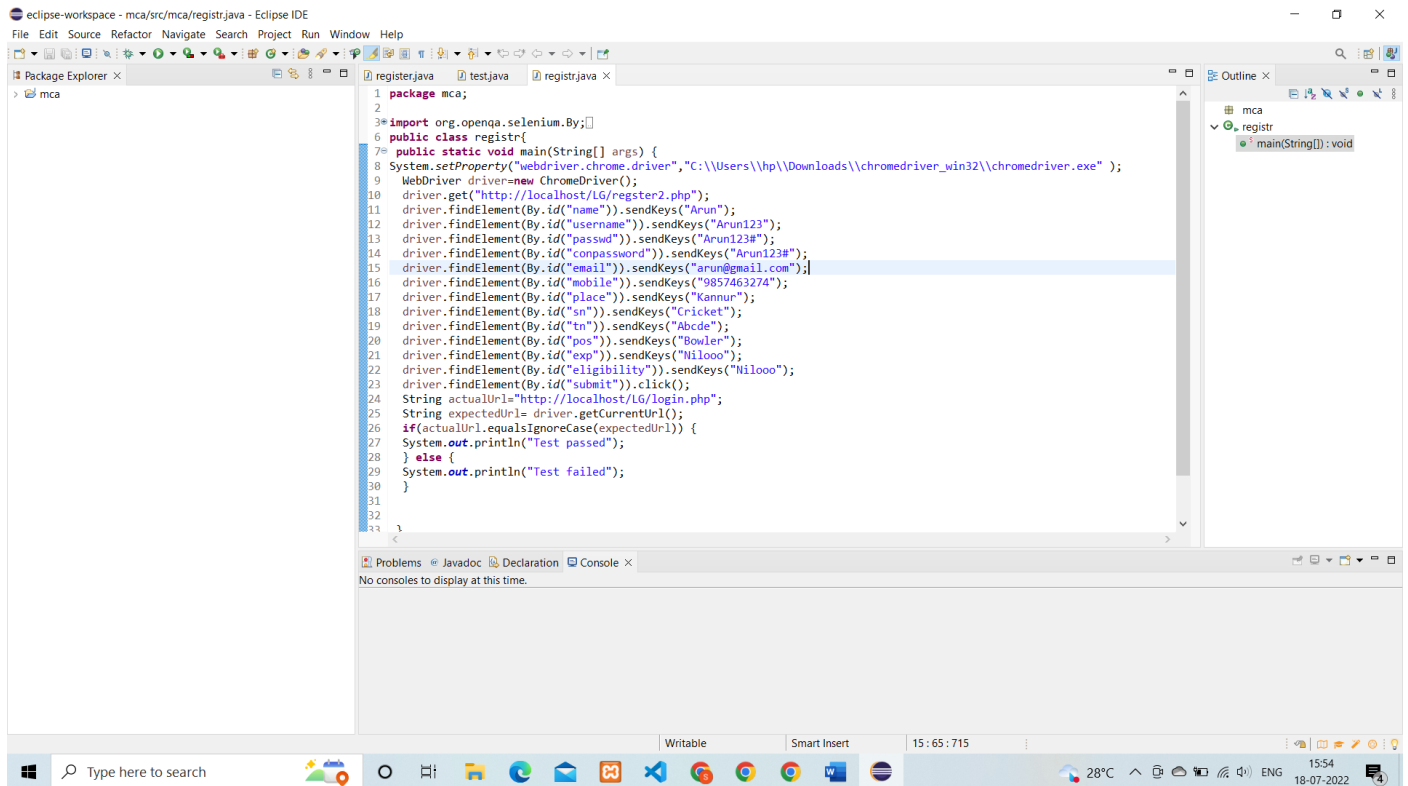
ProjectName:Go-play					
UpdationTestCase					
Test Case ID:registration			Test Designed By:SILJA C K		
Test Priority(Low/Medium/High): High			Test Designed Date:18-05-2022		
Module Name:Login Screen			Test Executed By:DR.BIJIMOL TK		
Test Title:User Registration Details			Test Execution Date:18-05-2022		
Description:Register to system and Registration is completed then login,if some error occurs, test will fail					
Pre-Condition: User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Register Page		Register Page Should be	Registrtrtion page displayed	Pass
2	Provide Valid Registration details	UserN ame: nimh	Users hould be able to Register	User registrion Completed after go to the login page	P a s s
3					
4	Click on Login button				
5	Provide profile details	Input Profile details	User will be redirected to Login page	Use will be redirected to Login page	P a s s
7	Click on register button				
8	Provide invalid information	Input invalid profile details.	User will be stay in register page	User will be stay on that page showing error message	P a s s
9	Click on register button				

Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database.

Codepackage

```
package mca;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class registr{
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\hp\\Downloads\\chromedriver_win32\\chromedriver.exe" );
        WebDriver driver=new ChromeDriver();
        driver.get("http://localhost/LG/register2.php");
        driver.findElement(By.id("name")).sendKeys("Arun");
        driver.findElement(By.id("username")).sendKeys("Arun123");
        driver.findElement(By.id("passwd")).sendKeys("Arun123#");
        driver.findElement(By.id("conpassword")).sendKeys("Arun123#");
        driver.findElement(By.id("email")).sendKeys("arun@gmail.com");
        driver.findElement(By.id("mobile")).sendKeys("9857463274");
        driver.findElement(By.id("place")).sendKeys("Kannur");
        driver.findElement(By.id("sn")).sendKeys("Cricket");
        driver.findElement(By.id("tn")).sendKeys("Abcde");
        driver.findElement(By.id("pos")).sendKeys("Bowler");
        driver.findElement(By.id("exp")).sendKeys("Nilooo");
        driver.findElement(By.id("eligibility")).sendKeys("Nilooo");
        driver.findElement(By.id("submit")).click();
        String actualUrl="http://localhost/LG/login.php";
        String expectedUrl= driver.getCurrentUrl();
        if(actualUrl.equalsIgnoreCase(expectedUrl)) {
            System.out.println("Test passed");
        } else {
            System.out.println("Test failed");
        }
    }
}
```



CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

The project's implementation phase is where the conceptual design is transformed into a functional system. It can be regarded as the most important stage in creating a successful new system since it gives users assurance that the system will operate as intended and be reliable and accurate. User documentation and training are its main concerns.

Usually, conversion happens either during or after the user's training. Implementation is the process of turning a newly revised system design into an operational one, and it simply refers to placing a new system design into operation. The majority of the workload, the majority of the interruption, and the majority of the impact on the current system are now handled by the user department. Confusion and mayhem may come from a poorly thought-out or managed implementation. Software testing is the practice of meticulously monitoring the way software is used to see if it works as intended. Two words that are frequently used in conjunction with the term "software testing" are verification and validation. A product, including software, is validated by being examined or evaluated to see if it conforms to all pertinent requirements. Software testing, one sort of verification, also makes use of reviews, analyses, inspections, and walkthroughs. Validation is the process of ensuring that what has been a specified match what the user truly desires.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

6.2 IMPLEMENTATION PROCEDURES

Software implementation refers to the complete installation of the package in its intended environment, as well as to the system's functionality and satisfaction of its intended applications. The software development project is frequently commissioned by someone who will not be using it. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place.

6.2.1 User Training

The purpose of user training is to get the user ready to test and modify the system. The people who will be involved must have faith in their ability to contribute to the goal and benefits anticipated from the computer-based system. Training is more necessary as systems get more complicated. The user learns how to enter data, handle error warnings, query the database, call up routines that will generate reports, and execute other important tasks through user training.

Training on the Application Software

The user must first receive the appropriate fundamental training in computer awareness prior to receiving instruction on the new application software. In addition to information on how the screens work, how they are made, what kinds of help are displayed, what errors are typically made when entering data, how to use validation procedures, and how to change the data that was entered, the fundamental principles of how the new system should be used are outlined here. allowing access to a certain user, group, or section of the system while doing application training. Different user groups and hierarchical levels may experience this differently.

System Maintenance

The mystery of system development is maintenance. When a software product is in the maintenance stage of its lifecycle, it is actively working. A system should be properly maintained after it has been effectively implemented. An essential part of the software development life cycle is system maintenance. In order for a system to be flexible to changes in the system environment, maintenance is required. Of course, software maintenance involves much more than just "Finding Mistakes".

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The Go-Play System is a tournament management system designed to facilitate team sports. The players of a team are selected by a system. The system is useful for a sportsman who is not affiliated with any sports team to play in a tournament. A team event such as volleyball or football is suitable to use this system. In conclusion, all of the objectives of this project have been achieved. This system gives a good impact for communities as it allows them to manage tournaments easily.

7.2 FUTURE SCOPE

- Data security can be enhanced
- Users can view live streaming of the tournament
- Users can view live score of the tournament
- An advertisement can be added
- In the future, we can provide coaching for players.
- This system can include sponsors for the tournament.
- User can provide complaints

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”,2009.
- Roger S Pressman, “*Software Engineering*”,1994.
- Pankaj Jalote, “*Software engineering: a precise approach*”,2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.jquery.com
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- www.agilemodeling.com/artifacts/useCaseDiagram.html

CHAPTER 9

APPENDIX

9.1 Sample Code

View_application.php

```
<?php
session_start();
include 'connection.php';
if(isset($_SESSION['user_id']))
    $user_id=$_SESSION['user_id'];

$result=mysqli_query($conn,"SELECT * FROM `tb_user` where user_id=$user_id") or
die(mysqli_error($conn));

include('pdf_mc_table.php');
$pdf = new PDF_MC_TABLE();
$pdf->AddPage();
$pdf->SetFont('Arial','B',15);
$pdf->Cell(176, 5, 'View Details', 0, 0, 'C');
$pdf->Ln();
$pdf->Ln();
$pdf->Ln();
$row=mysqli_fetch_array($result);
$pdf->SetFont('Arial','',12);
$pdf->Multicell(80,12,'name : '. $row['name'],1);
$pdf->Multicell(80,12,'email : '. $row['email'],1);
$pdf->Multicell(80,12,'moblie : '. $row['mobile'],1);
$pdf->Multicell(80,12,'place : '. $row['place'],1);
$pdf->Multicell(80,12,'sports: '. $row['sp_type'],1);

$pdf->Output();
?>
```

Verifypemail.php

```
<?php
//Import PHPMailer classes into the global namespace
//These must be at the top of your script, not inside a function
use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;
//$email=$_GET['email'];
$r=$_GET['uid'];
include('connection.php');
require 'Exception.php';
require 'PHPMailer.php';
require 'SMTP.php';
```

```
//Create an instance; passing `true` enables exceptions
$mail = new PHPMailer(true);

try {
    //Server settings
    $mail->SMTPDebug = SMTP::DEBUG_SERVER;           //Enable verbose debug output
    $mail->isSMTP();                               //Send using SMTP
    $mail->Host      = 'smtp.gmail.com';             //Set the SMTP server to send through
    $mail->SMTPAuth  = true;                         //Enable SMTP authentication
    $mail->Username   = 'tournamentgoplay9@gmail.com'; //SMTP username
    $mail->Password   = 'eacuelbxawbylgbw';          //SMTP password
    $mail->SMTPSecure = 'tls';                       //Enable implicit TLS encryption
    $mail->Port       = 587;                         //TCP port to connect to; use 587 if you have set
    `SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS`

    //Recipients
    $mail->setFrom('tournamentgoplay9@gmail.com', 'Go-Play');
    $mail->addAddress($r); //Add a recipient
    // $mail->addAddress('ellen@example.com'); //Name is optional
    // $mail->addReplyTo('info@example.com', 'Information');
    // $mail->addCC('cc@example.com');
    // $mail->addBCC('bcc@example.com');

    //Attachments
    /// $mail->addAttachment('/var/tmp/file.tar.gz'); //Add attachments
    // $mail->addAttachment('/tmp/image.jpg', 'new.jpg'); //Optional name

    //Content
    $mail->isHTML(true); //Set email format to HTML
    $mail->Subject = 'You Are An Team Captain Now';
    $mail->Body    = 'Your Application Is Approved you can login now as Captain';
    $mail->AltBody = 'Thanks For your Interest';

    $mail->send();
    echo "<script>alert('Message has been Sent');window.location.href='../viewteamdetails.php';</script>";

} catch (Exception $e) {
    //echo $e;
    echo "Message could not be sent. Mailer Error: {$mail->ErrorInfo}";
}
```

Paymentgateway.php

```
<?php
    $apiKey="rzp_test_c90wS0Xc4OOIWK";
    include 'connection.php';
    session_start();
    if(isset($_SESSION['id']))
    {
        include('connection.php');
        $r_id=$_GET['tid'];
        $tn=$_GET['uid'];
        $query2="select * from tb_teams where user_id='$r_id'";
```

```

    $res2 = mysqli_query($conn, $query2);
    $rr = mysqli_fetch_array($res2);
    $lid=$rr["team_id"];
    $sql=mysqli_query($conn,"insert into tb_booking(trmn_id,team_id,status,canselreq)values
    ('$tn','$lid','Requested','null')");
    $result=mysqli_query($conn,$sql);
    $id=$_SESSION['id'];
    $query="SELECT * FROM tb_login where log_id ='$id'";
    $res = mysqli_query($conn,$query);
    $r=mysqli_fetch_array($res);

    if($r[4]==1)
    {

        $query1="SELECT * FROM tb_user where log_id ='$id'";
        $res1 = mysqli_query($conn,$query1);
        $r1=mysqli_fetch_array($res1);

    }

?>
<!DOCTYPE html>
<html>
    <title>Payment Gateway</title>
    <head>
    <style>
    body {
        display: flex;
        justify-content: center;
        align-items: center;
        background: #242d60;
        font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto',
        'Helvetica Neue', 'Ubuntu', sans-serif;
        height: 100vh;
        margin: 0;
        -webkit-font-smoothing: antialiased;
        -moz-osx-font-smoothing: grayscale;
    }
    section {
        background: #ffffff;
        display: flex;
        flex-direction: column;
        width: 500px;
        height: 202px;
        border-radius: 6px;
        justify-content: space-between;
    }
    .product {
        display: flex;
        padding: 24px;
    }
    .description {
        display: flex;
        flex-direction: column;

```

```
    justify-content: center;
  }
  p {
    font-style: normal;
    font-weight: 500;
    font-size: 14px;
    line-height: 20px;
    letter-spacing: -0.154px;
    color: #242d60;
    height: 100%;
    width: 100%;
    padding: 0 20px;
    display: flex;
    align-items: center;
    justify-content: center;
    box-sizing: border-box;
  }
  h3,
  h5 {
    font-style: normal;
    font-weight: 500;
    font-size: 14px;
    line-height: 20px;
    letter-spacing: -0.154px;
    color: #242d60;
    margin: 0;
  }
  h5 {
    opacity: 0.5;
  }

  button {
    height: 36px;
    background: #556cd6;
    color: white;
    width: 100%;
    font-size: 14px;
    border: 0;
    font-weight: 500;
    cursor: pointer;
    letter-spacing: 0.6;
    border-radius: 0 0 6px 6px;
    transition: all 0.2s ease;
    box-shadow: 0px 4px 5.5px 0px rgba(0, 0, 0, 0.07);
  }
  .button
  {
    height: 36px;
    background: #556cd6;
    color: white;
    width: 100%;
    font-size: 14px;
    margin-top: 20px;
```

```

border: 0;
font-weight: 500;
cursor: pointer;
letter-spacing: 0.6;
border-radius: 0 0 6px 6px;
transition: all 0.2s ease;
box-shadow: 0px 4px 5.5px 0px rgba(0, 0, 0, 0.07);
}
button:hover {
  opacity: 1;
}

.detailsection{
  display: block;
}
.anim{
  width:25%;
  margin: 0% 4%;
}
</style>

</head>
<div class="anim" id="anim"></div>
<?php
    // $u=$_SESSION['id'];
    // $l=$_GET['id'];
    $query5 ="SELECT * FROM tb_booking where team_id='$lid' ";
    $res5 = mysqli_query($conn,$query5);
    $r5=mysqli_fetch_array($res5);
$vid=$r5['book_id'];
    $query6 ="SELECT * FROM tb_tournament where trmn_id='$tn' ";
    $res6 = mysqli_query($conn,$query6);
    $r6=mysqli_fetch_array($res6);
?>
<section>
    <div class="product">
        <div class="description">
            <h3>The AmountPay By: <?php echo $r1['name'];?> </h3>
            <h3>The Amount : <?php echo $r6['price'];?> </h3>
        </div>
    </div>

    <form action="payaction.php?id=<?php echo $vid;?>" method="post">
    <script
        src="https://checkout.razorpay.com/v1/checkout.js"
        data-key="<?php echo $apiKey; ?>" // Enter the Test API Key ID generated from Dashboard → Settings
        → API Keys
        data-amount="<?php echo $r6['price'] *100;?>" // Amount is in currency subunits. Hence, 29935 refers to
        29935 paise or ₹299.35.
        data-currency="INR"// You can accept international payments by changing the currency code. Contact our
        Support Team to enable International for your account
        data-order_id="<?php rand(100000, 999999);?>"// Replace with the order_id generated by you in the
        backend.

```

```

    data-buttontext="Procced to Pay"
    data-name="Go-Play"
    data-description="VBA Payment"
    data-image=""
    data-prefill.name=""
    data-prefill.email="<?php echo $r1['email'];?>"
    data-theme.color="darkblue"
</script>
<input type="hidden" name="bkid" value="<?php echo $vid; ?>">
<input type="hidden" custom="Hidden Element" name="hidden">
</form>

<!--style>
.razorpay-payment-button { display:none; }
</style>

<script type="text/javascript">
    $(document).ready(function(){
        $('.razorpay-payment-button').click();

    });
</script-->
<?php
}
else{
    echo '<script type="text/javascript">window.location.href="block.php"</script>';
}
}

    else
    {
        if(headers_sent())
        {
            die('<script type="text/javascript">window.location.href="log.php"</script>');
        }
        else
        {
            header("location:log.php");
            die();
        }
    }

?>

```

9.2 ScreenShots

Homepage

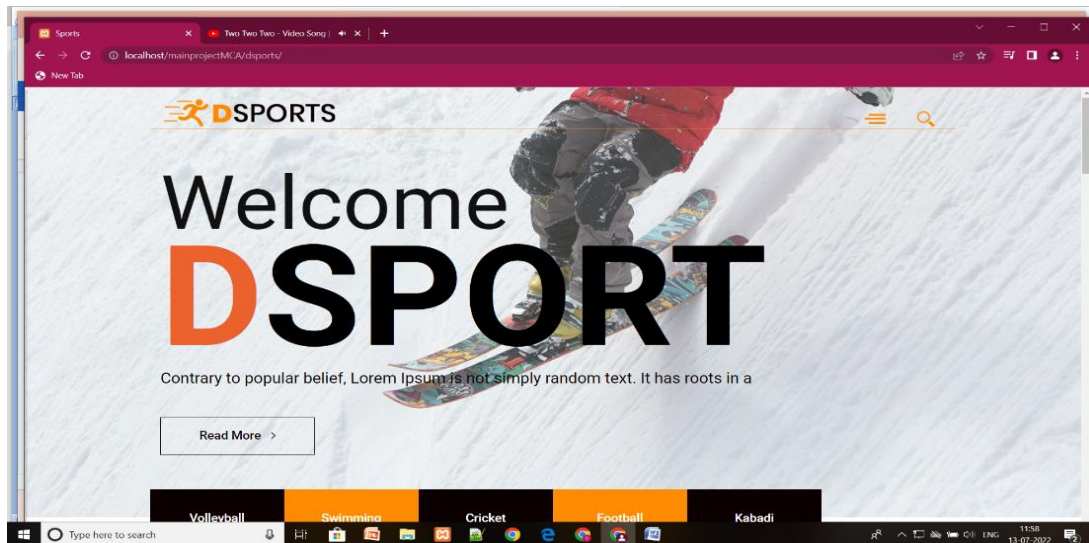


Fig 13:Home page Screen shot

Manage User

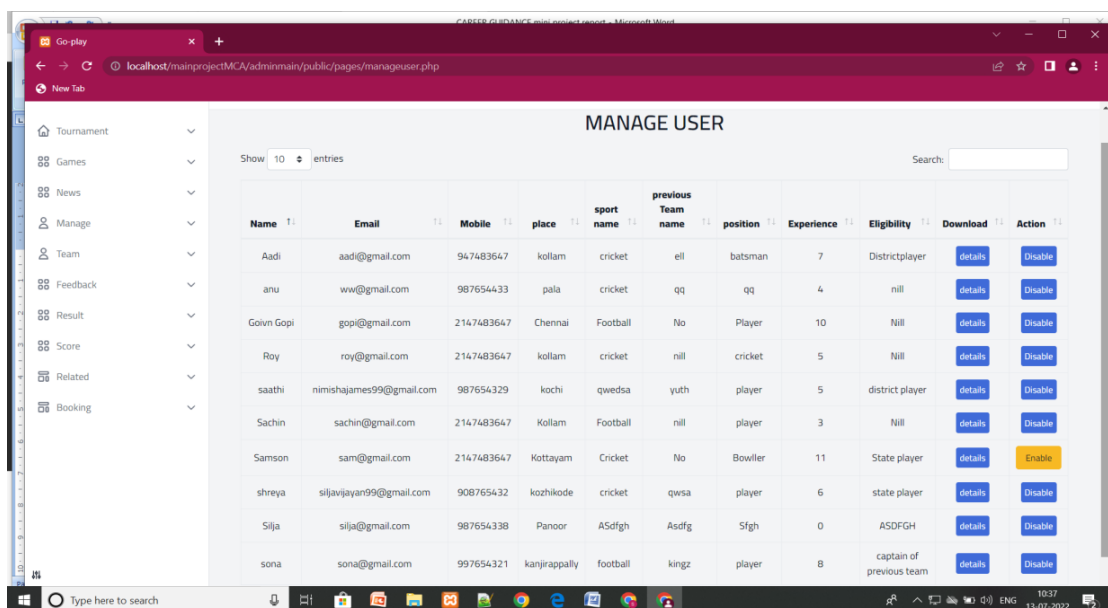


Fig 14:Manage user page Screen shot

Manage Team

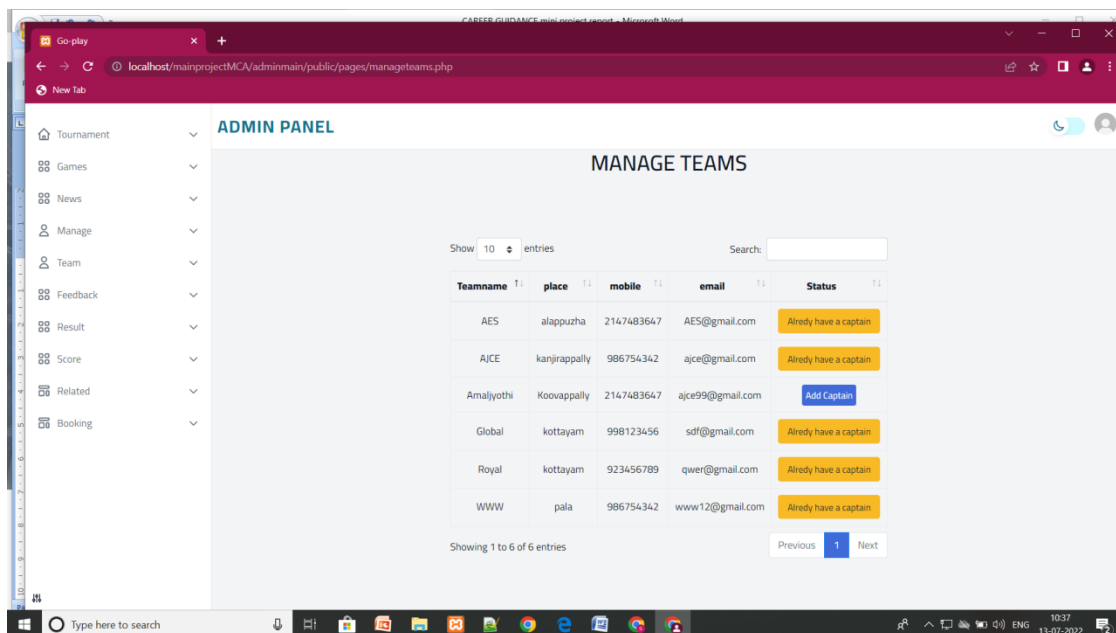


Fig 15: Add captain to a team page Screen shot

Schedule Tournament

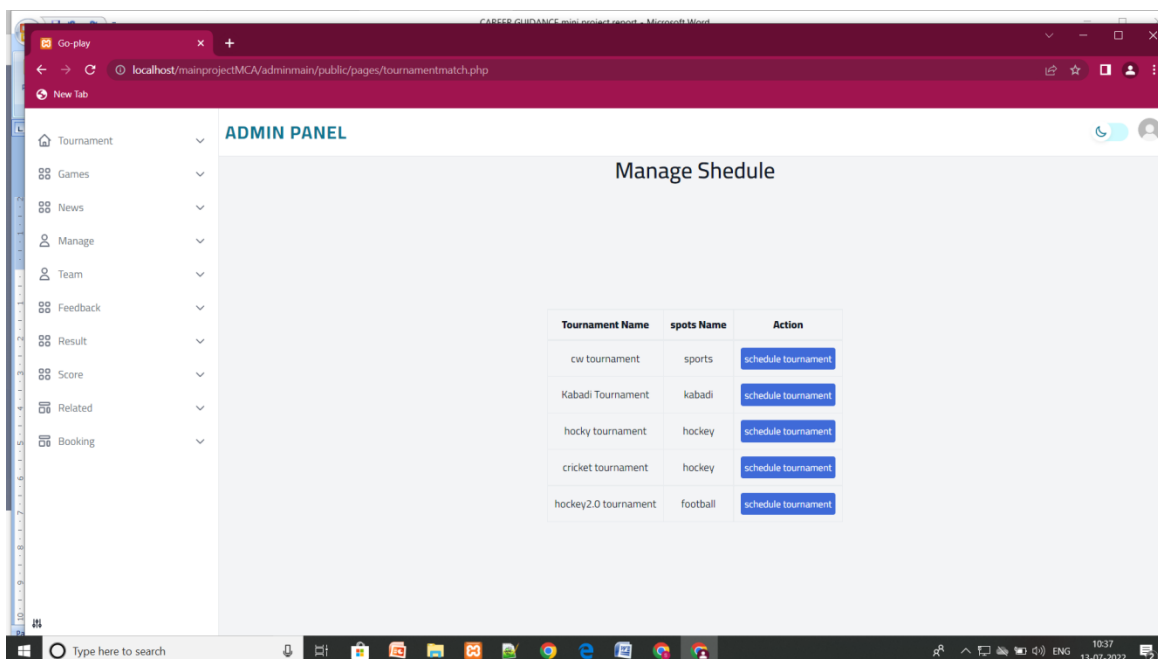


Fig 15: Schedule Tournament page Screen shot

Report Generate

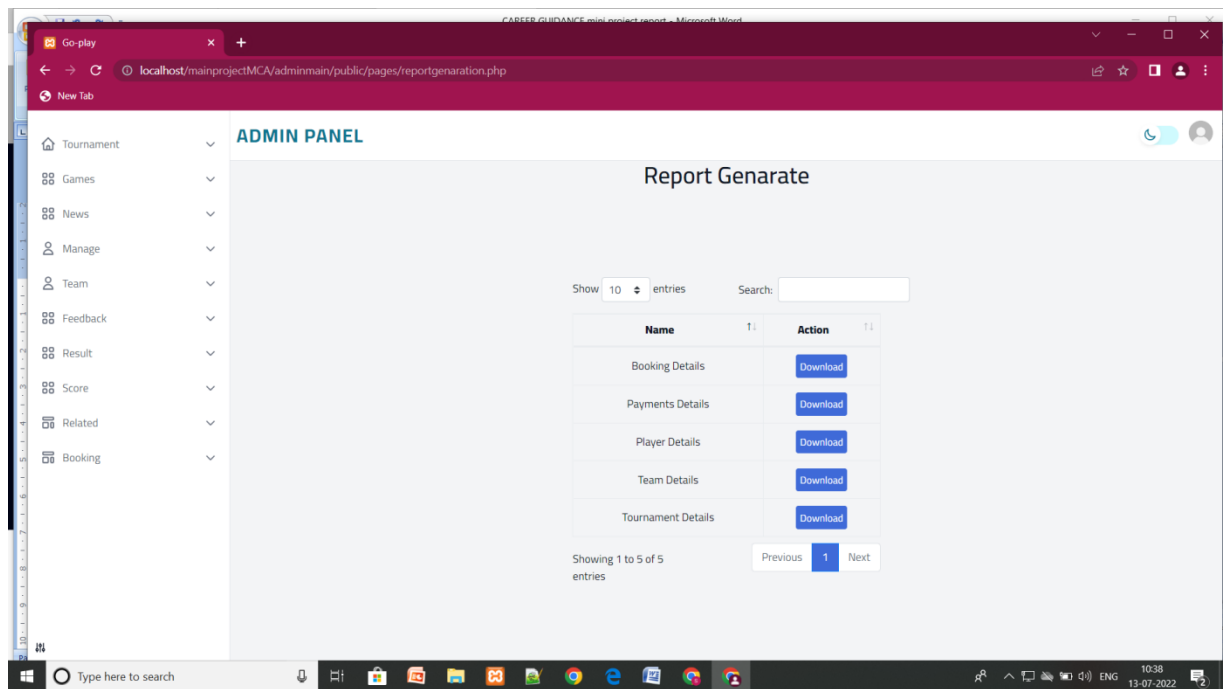


Fig 16:Report generation page Screen shot

View Score

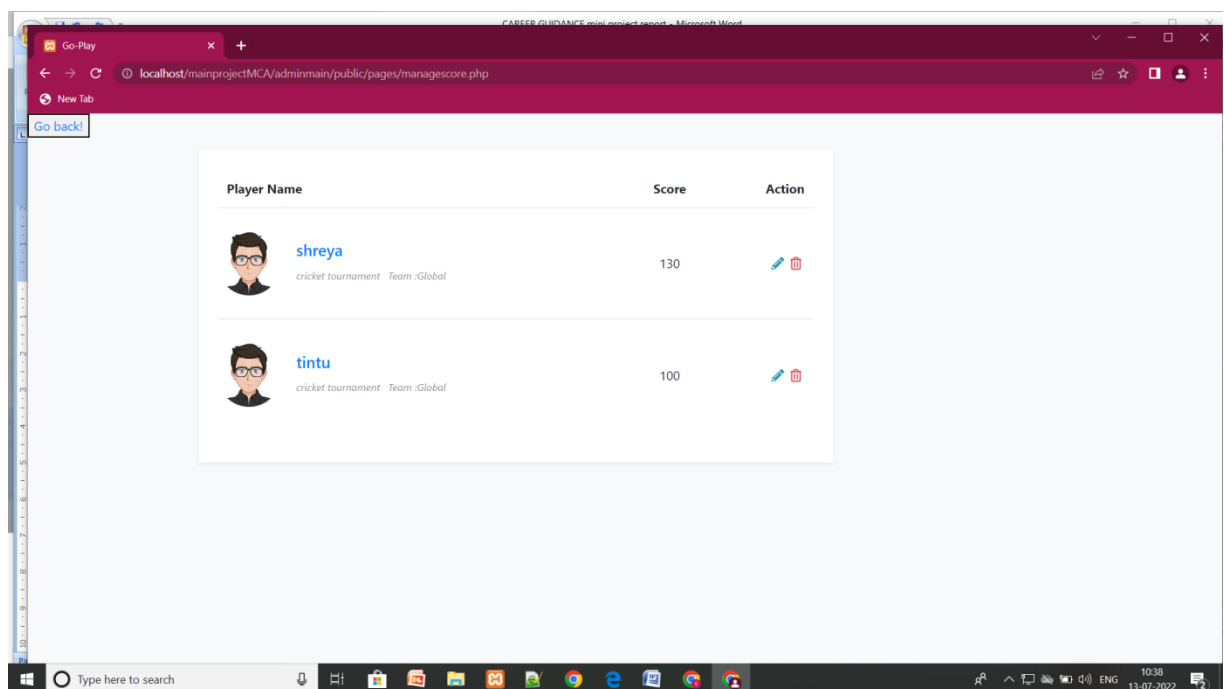


Fig 17:View Result page Screen shot

Add team players

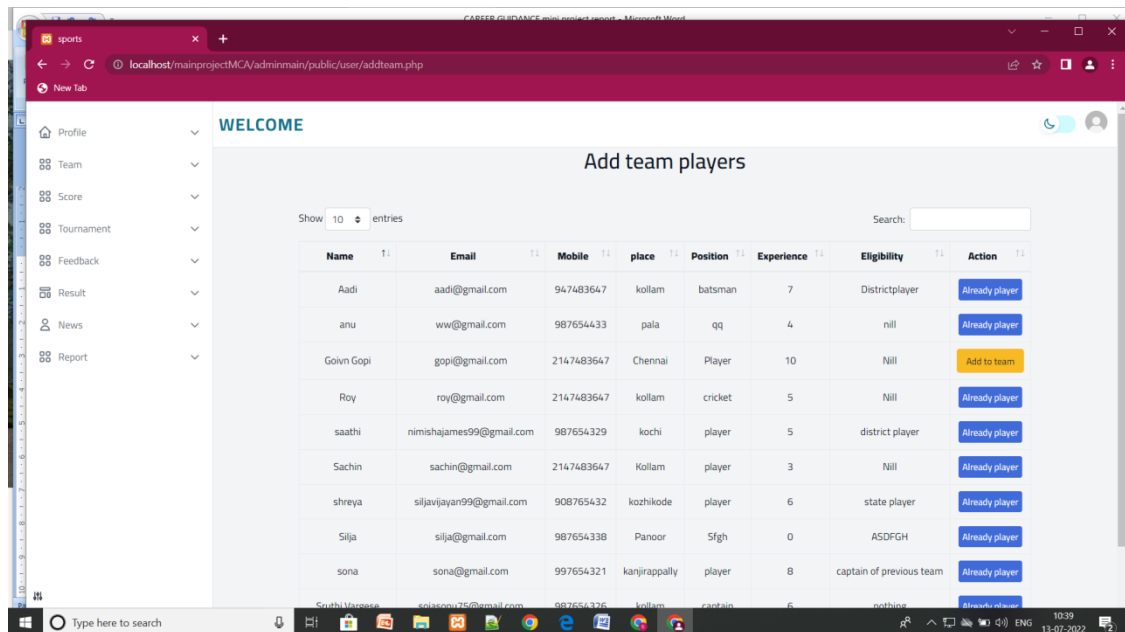


Fig 18: Adding players to team page Screen shot

Register Tournament

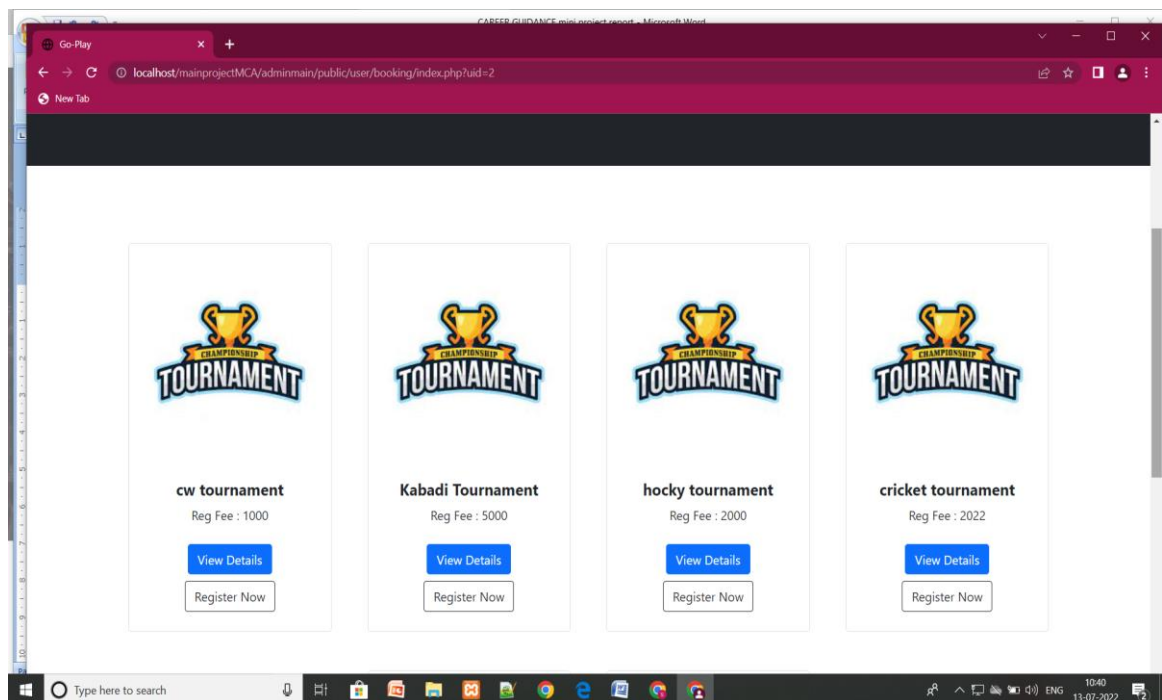


Fig 19: Register Tournament page Screen shot

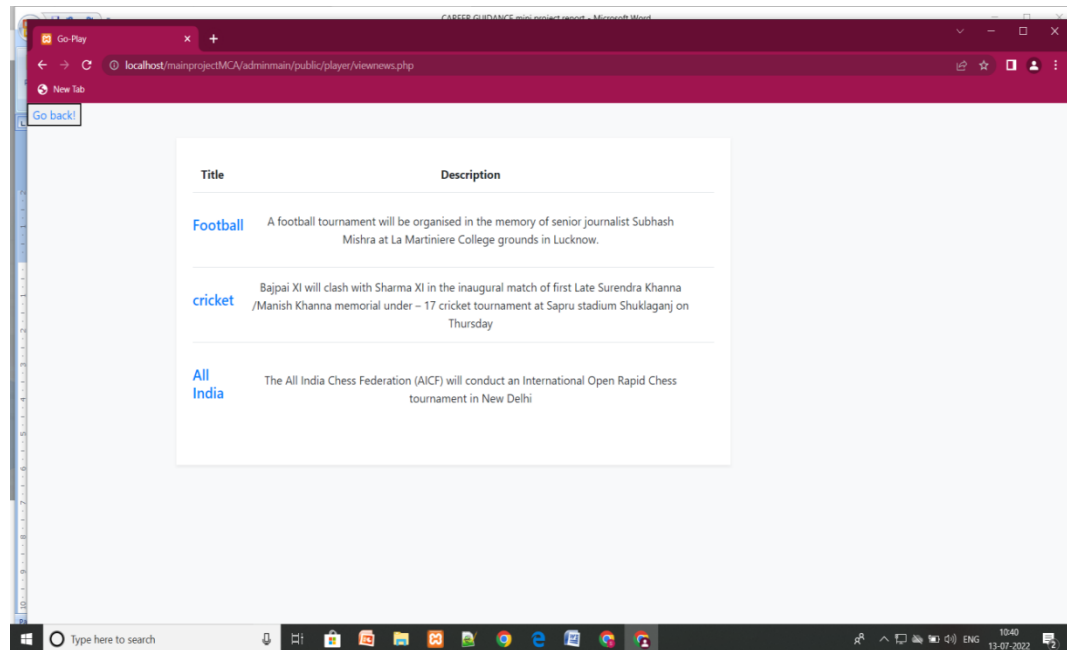


Fig 20:View News page Screen shot

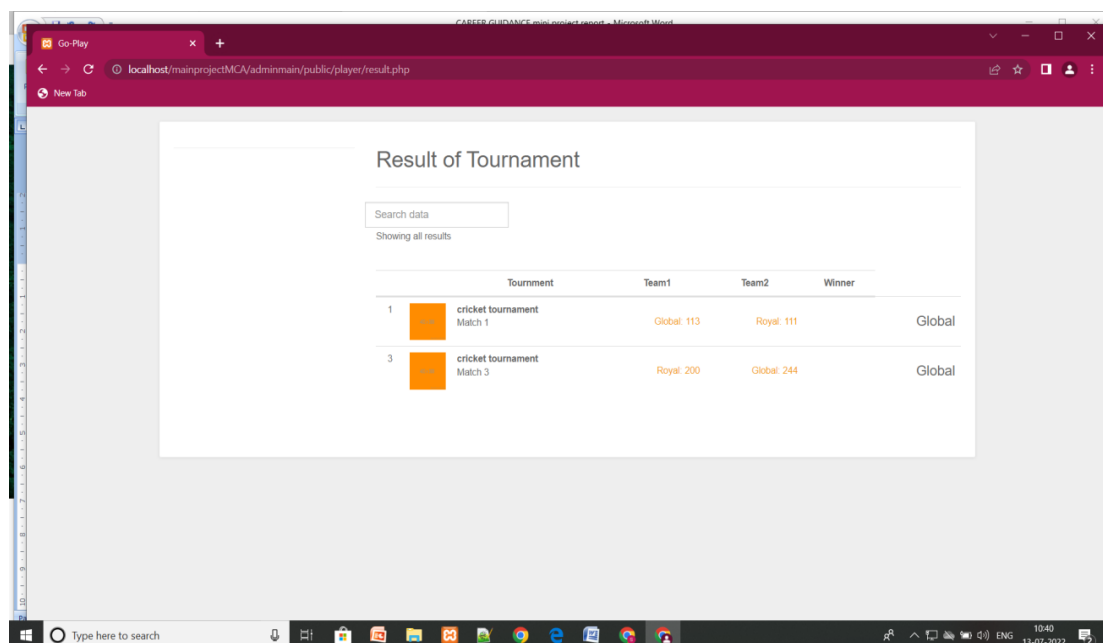


Fig 21:View Tournament Result page Screen shot