VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", MACHHE, BELAGAVI – 590018

Mini Project Report
on

CRICKET DATABASE MANAGEMENT SYSTEM

Submitted in partial fulfillment of the requirements for the V semester

Bachelor of Engineering

in

Computer Science and Engineering

of

Visvesvaraya Technological University, Belagavi

by

Mr. AMARJEET KUMAR (1CD20CS401)
Mr. AKASH M (1CD20CS400)

Under the Guidance of

Ms. Priyadarshini	Ms. Reshma S. Dsouza
Assistant Professor,	Assistant Professor,
Dept of CSE, CITech	Dept of CSE, CITech

Department of Computer Science and Engineering
CAMBRIDGE INSTITUTE OF TECHNOLOGY, BENGALURU – 560036
2021-2022

CAMBRIDGE INSTITUTE OF TECHNOLOGY

K.R. Puram, Bengaluru – 560036

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

Certified that **Mr. Amarjeet Kumar and Mr. Akash M** bearing USN **1CD20CS401, 1CD20CS400,** respectively are bonafide students of **Cambridge Institute of Technology,** has successfully completed Mini Project entitled "**Cricket Database Management System**" in partial fulfillment of the requirements for V semester **Bachelor of Engineering** in **Computer Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year 2021 - 2022. It is certified that all Corrections / Suggestions indicated for Internal Assessment have been incorporated in the report. The Database Project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the said semester.

Internal Guides:

Head of the Dept.

1) Ms. Priyadarshini
Dept. of CSE., CiTech.

Dr. Shashikumar D. R.
Dept. of CSE., CiTech.

2) Ms. Reshma S. Dsouza
Dept. of CSE., CiTech.

Examiners:

1)

2)

ACKNOWLEDGEMENT

We are extremely thankful to **Dr. Yogesh Velankar.,** Principal, CITech., Bengaluru, for providing us the academic ambience and everlasting motivation to carry out this work and shaping our careers.

We express our sincere gratitude to **Dr. Shashikumar D. R.,** HOD, Dept. of Computer Science and Engineering, CITech., Bengaluru, for his stimulating guidance, continuous encouragement and motivation throughout the course of present work.

We also wish to extend our thanks to **Priyadarshini,** Assistant Professor Dept. of Computer Science and Engineering, CITech., Bengaluru and **Reshma S. Dsouza.,** Assistant Professor, Dept. of Computer Science and Engineering, CITech., Bengaluru, for their expert guidance and constructive suggestions to improve the quality of this work.

We would also like to thank all other teaching and technical staffs of Department of Computer Science and Engineering, who have directly or indirectly helped us in the completion of this Project Work.

And lastly we would hereby acknowledge and thank our parents who have been a source of inspiration and also instrumental in the successful completion of this project.

Amarjeet Kumar (1CD20CS401)

Akash M (1CD20CS400)

ABSTRACT

Cricket Database Management System is a software application to manage all the activities concern to the cricket. This project will help the cricket organization to store all cricket related information. This application is useful store all the information related to the players, teams, matches, score and stadium ground, etc., Cricket Database Management is a cricket scheduling-based application exclusively for the game of cricket. The Application features schedules, information about teams, records of batting and bowling, creating new schedules, can search about players. The admin has all authorities to make changes for the database so admin can add players, can add schedules, can add stadiums, can add scores and also have permission to removing of them from the database. It features with printing the details of player, Teams, Matches, Scores, Stadium. It can fetch the schedules with their venue and squad available by the team, players selected for the current match. The admin has authorized to perform all CRUD operation on the database. The main objective of the application is to automate the existing method of managing data manually to systematic storage of data, which can be searched any time as and when required. The other main objective of this application is to store all the player, teams, match, ground details.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO DATABASE MANAGEMENT SYSTEM

Databases and database technology have a major impact on the growing use of computers. It is fair to say that databases play a critical role in almost all areas where computers are used, including business, electronic commerce, engineering, medicine, genetics, law, education, and library science. The word database is so commonly used that User must begin by defining what a database is. Our initial definition is quite general. A database is a collection of related data.1 By data, User mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know. You may have recorded this data in an indexed address book or you may have stored it on a hard drive, using a personal computer and software such as Microsoft Access or Excel. This collection of related data with an implicit meaning is a database. The preceding definition of database is quite general; for example, User may consider the collection of words that make up this page of text to be related data and hence to constitute a database. However, the common use of the term database is usually more restricted.

A database has the following implicit properties:

- A database represents some aspect of the real world, sometimes called the mini world or the universe of discourse. Changes to the inworld are reflected in the database.
- A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database
- A database is designed, built, and populated with data for a specific purpose. It has an intended. group of users and some preconceived applications in which these users are interested.

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications. Defining a database involves specifying the data types, structures, and constraints of the data to be stored in the database.

The database definition or descriptive information is also stored by the DBMS in the form of a database catalogue or dictionary; it is called meta-data. Constructing the database is the process of storing the data on some storage medium that is controlled by the DBMS. Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the inworld, and generating reports from the data. Sharing a database allows multiple users and programs to access the database simultaneously.

1.2 History of DBMS

In 1959, the TX-2 computer was developed at MIT's Lincoln Laboratory. The TX-2 integrated a number of new man-machine interfaces. A light pen could be used to draw sketches on the computer using Ivan Sutherland's revolutionary Sketchpad software. Using a light pen, Sketchpad allowed one to draw simple shapes on the computer screen, save them and even recall them later. The light pen itself had a small photoelectric cell in its tip. This cell emitted an electronic pulse whenever it was placed in front of a computer screen and the screen's electron gun fired directly at it. By simply timing the electronic pulse with the current location of the electron gun, it was easy to pinpoint exactly where the pen was on the screen at any given moment. Once that was determined, the computer could then draw a cursor at that location. Also, in 1961 another student at MIT, Steve Russell, created the first video game, E. E. Zajac, a scientist at Bell Telephone Laboratory (BTL), created a film called "Simulation of a two-gravity attitude control system" in 1963. During 1970s, the first major advance in 3D computer graphics was created at UU by these early pioneers, the hidden-surface algorithm. In order to draw a representation of a 3D object on the screen, the computer must determine which surfaces are "behind" the object from the viewer's perspective, and thus should be "hidden" when the computer creates (or renders) the image. In the 1980s, artists and graphic designers began to see the personal computer, particularly the Commodore Amiga and Macintosh, as a serious design tool, one that could save time and draw more accurately than other methods. In the late 1980s, SGI computers were used to create some of the first fully computer-generated short films at Pixar. The Macintosh remains a highly popular tool for computer graphics among graphic design studios and businesses. Modern computers, dating from the 1980s often use graphical user interfaces (GUI) to present data and information with symbols, icons and pictures, rather than text. Graphics are one of the five key elements of multimedia technology. 3D graphics became more popular in the 1990s in gaming, multimedia and animation. In 1996, Quake, one of the first fully 3D games, was released. In 1995, Toy Story, the first full-length computer-generated animation film, was released in cinemas worldwide. Since then, computer graphics have only become more detailed and realistic, due to more powerful graphics hardware and 3D modeling software.

1.3 APPLICATIONS OF DBMS

Applications where we use Database Management Systems are:

Telecom: There is a database to keeps track of the information regarding calls made, network usage, customer details etc. Without the database systems it is hard to maintain that huge amount of data that keeps updating every millisecond.

Industry: Where it is a manufacturing unit, warehouse or distribution Centre, each one needs a database to keep the records of ins and outs. For example, distribution Centre should keep a track of the product units that supplied into the Centre as well as the products that got delivered out from the distribution Centre on each day; this is where DBMS comes into picture.

Banking System: For storing customer info, tracking day to day credit and debit transactions, generating bank statements etc. All this work has been done with the help of Database management systems.

Education sector: Database systems are frequently used in schools and colleges to store and retrieve the data regarding student details, staff details, course details, exam details, payroll data, attendance details, fees detail etc. There is a hell lot amount of inter-related data that needs to be stored and retrieved in an efficient manner.

Online shopping: You must be aware of the online shopping websites such as Amazon, Flipkart etc. These sites store the product information, your addresses and preferences, credit details and provide you the relevant list of products based on your query. All this involves a Database management system.

1.4 Overview of the project

Cricket Database Management System is a user-friendly Application which is based on JAVA which helps members to schedule and manage various Cricket Matches and also allows us to manage the records of various players. The application uses JAVA (JFrame) as a front end for interacting with the user and also for connection to the database. At backend we used MySQL for database.

1.5 Theory and Concepts

Inheritance: In object-oriented programming, inheritance is when an object or class is based on another object or class, using the same implementation (inheriting from an object or class) or specifying a new implementation to maintain the same behaviour (realizing an interface). Such an inherited class is called a subclass of its parent class or super class.

Encapsulation: In object-oriented programming, encapsulation is a mechanism of binding the data, and the functions together in a class and use them by creating an object of that class.

Data Abstraction: Data abstraction refers to, providing only essential information to the outside world and hiding their background details, i.e., to represent the needed information in program without presenting the implementation details. Data abstraction is a programming (and design) technique that relies on the separation of interface and implementation.

1.6 Xampp server

XAMPP is a free and open-source cross platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, Maria DB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform, Apache, Maria DB, PHP and Perl. It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well.

CHAPTER 2

REQUIREMENTS

2.1 Software Requirement Specifications

A major element in building a system is the section of compatible software since the software in the market is experiencing in geometric progression. Selected software should be acceptable by the firm and one user as well as it should be feasible for the system.

This document gives a detailed description of the software requirement specification. The study of requirement specification is focused specially on the functioning of the system. It allows the developer or analyst to understand the system, function to be carried out the performance level to be obtained and corresponding interfaces to be established.

Front End	: JAVA (Swings)
Back End	: XAMPP server, MySQL
Operation System	: Windows 10
IDE	: NetBeans/ Eclipse

2.2 Hardware Requirement Specifications

The section of hardware configuration is an important task related to the software development insufficient random-access memory may affect adversely on the speed and efficiency of the entire system. The process should be powerful to handle the entire operations. The hard disk should have sufficient capacity to store the file and application

Processor	: Intel PentiumT4200/ Intel Core Duo 2.0 GHz / more
RAM	: Minimum 1 GB RAM capacity
Hard disk	: Minimum 40 GB ROM capacity
Cache Memory	: L2-1 MB
GPU	: Intel HD Graphics

CHAPTER 3

ENTITY RELATIONSHIP DIAGRAM

3.1 ER Diagram

An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored.

The cardinality or fundamental principle of one data aspect with respect to another is a critical feature. The relationship of one to the other must be precise and exact between each other in order to explain how each aspect links together. In simple words Cardinality is a way to define the relationship between two entities.

The following are the notations of the ER diagram:

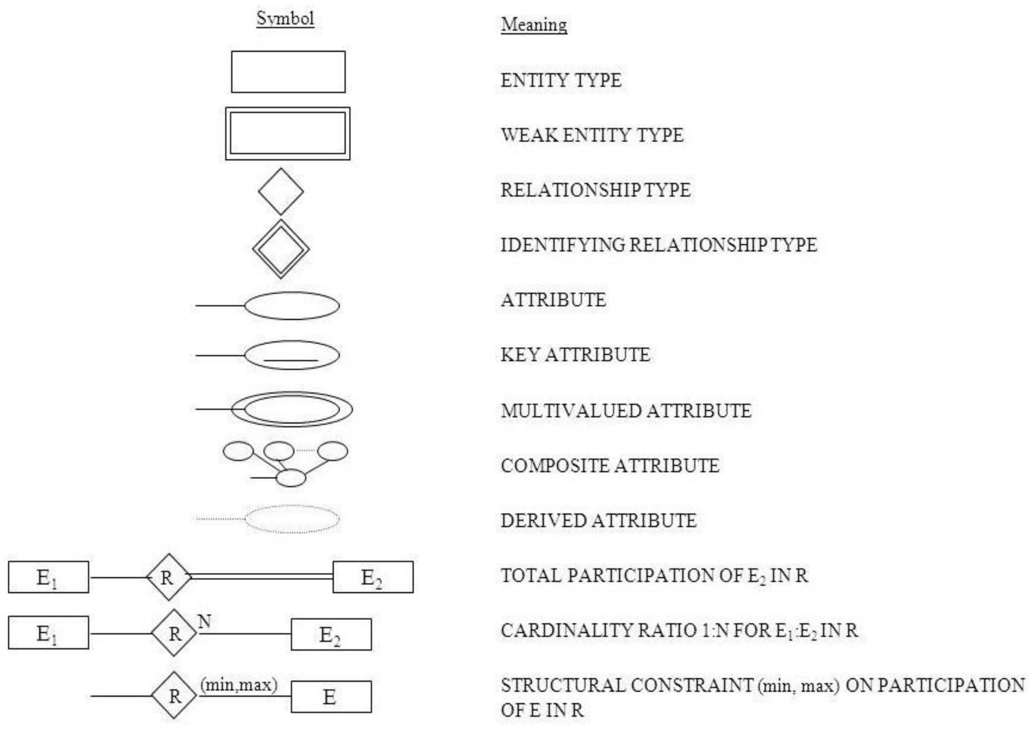


Figure 3.2 Notation for ER Diagram

Cardinalities can denote that an entity is optional (for example, an employee rep could have no customers or could have many) or mandatory (for example, there must be at least one product listed in an order.

The four main cardinal relationships are:

- One-to-one (1: 1) - For example, each customer in a database is associated with one mailing address.
- One-to-many (1: N) - For example, a single customer might place an order for multiple products. The customer is associated with multiple entities, but all those entities have a single connection back to the same customer.
- Many-to-one (N: 1) – For example, many employees will have only one manager above them but one manager can have many employees below him.
- Many-to-many (M: N) - For example, at a company where all call center agents work with multiple customers, each agent is associated with multiple customers, and multiple customers might also be associated with multiple agents.

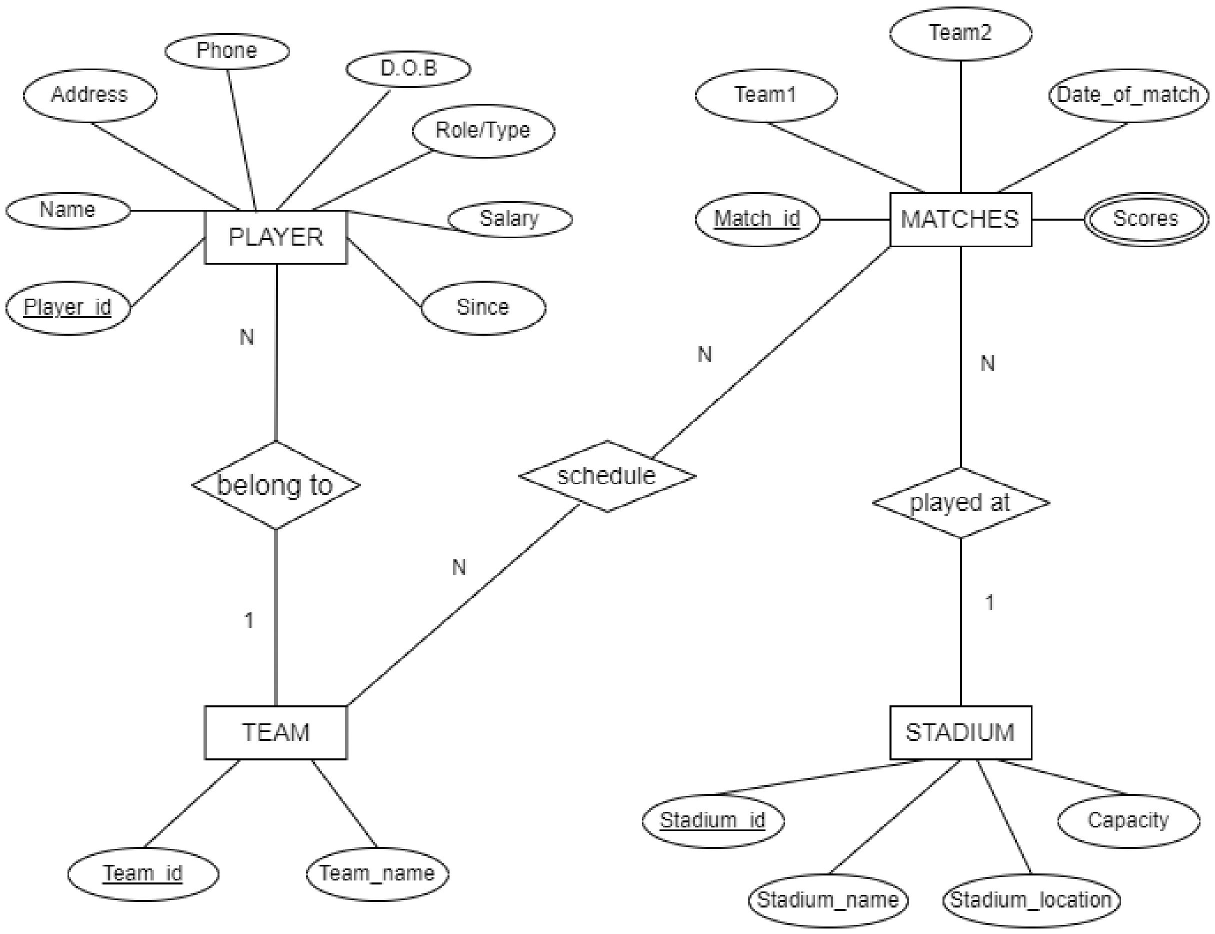


Figure 3.2 ER-Diagram

CHAPTER 4

SCHEMA DIAGRAM

4.1 SCHEMA DIAGRAM

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful. It is important that we distinguish these two terms individually. Database schema is the skeleton of database. It is designed when the database doesn't exist at all. Once the database is operational, it is very difficult to make any changes to it. A database schema does not contain any data or information. A database instance is a state of operational database with data at any given time. It contains a snapshot of the database. Database instances tend to change with time.

A DBMS ensures that its every instance (state) is in a valid state, by diligently following all the validations, constraints, and conditions that the database designers have imposed. A database schema can be divided broadly into two categories:

- Physical Database Schema - This schema pertains to the actual storage of data and its form of storage like files. indices, etc. It defines how the data will be stored in a secondary storage.
- Logical Database Schema – This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views and integrity constraints.

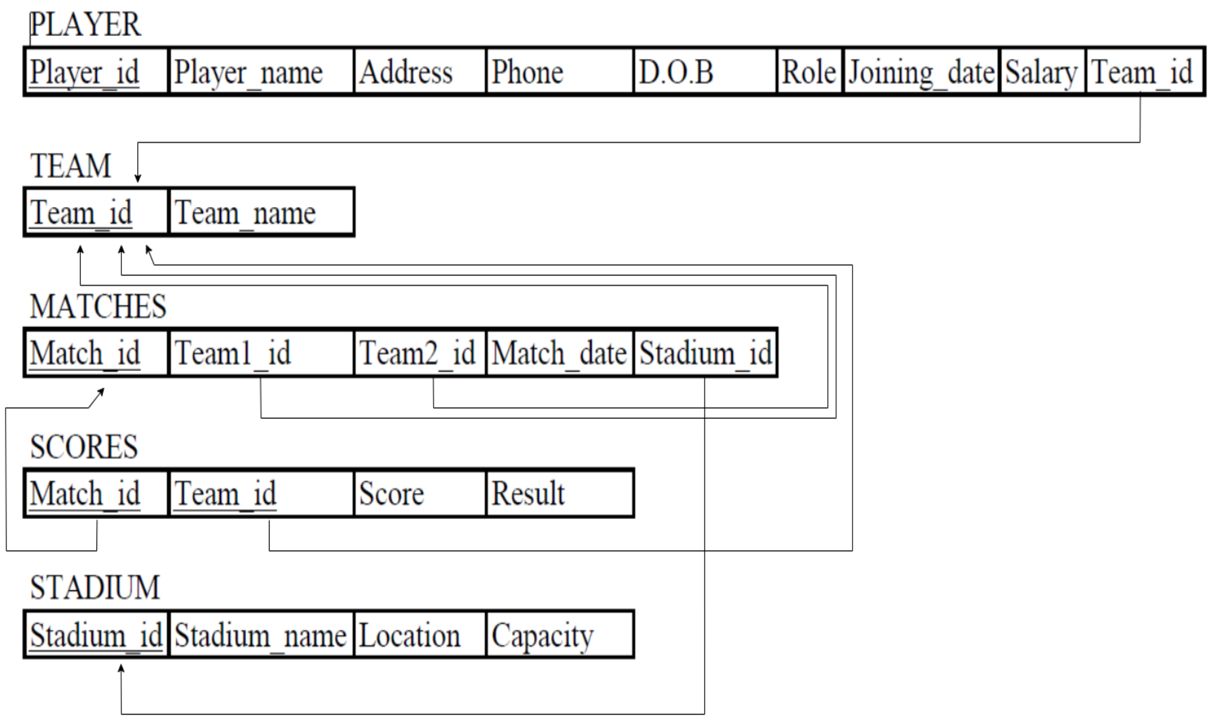


Figure 4.1 Schema diagram

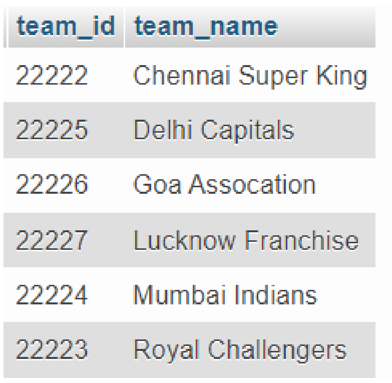
CHAPTER 5

IMPLEMENTATION

5.1 Backend Implementation (database)

Team Table:

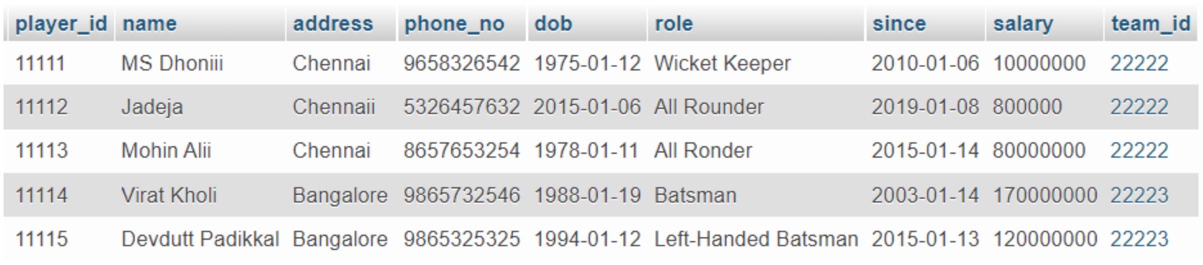
```
create table team(
        team_id int primary key,
        team_name varchar(255) unique
);
```



team_id	team_name
22222	Chennai Super King
22225	Delhi Capitals
22226	Goa Assocation
22227	Lucknow Franchise
22224	Mumbai Indians
22223	Royal Challengers

Player Table:

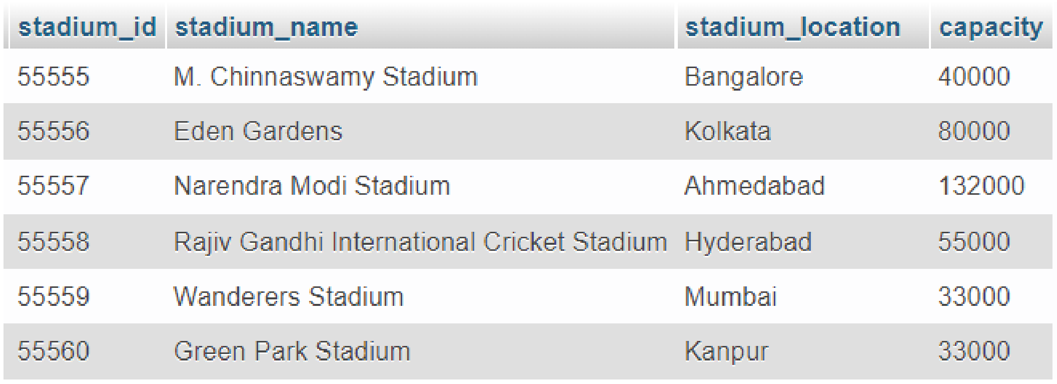
```
create table player(
        player_id int primary key,
        name varchar(255),
        address varchar(255) not null,
        phone_no int,
        dob date,
        role varchar(20),
        since date,
        salary int,
        team_id int,
        foreign key(team_id) references team(team_id) on delete set null
);
```



player_id	name	address	phone_no	dob	role	since	salary	team_id
11111	MS Dhoniii	Chennai	9658326542	1975-01-12	Wicket Keeper	2010-01-06	10000000	22222
11112	Jadeja	Chennaii	5326457632	2015-01-06	All Rounder	2019-01-08	800000	22222
11113	Mohin Alii	Chennai	8657653254	1978-01-11	All Ronder	2015-01-14	80000000	22222
11114	Virat Kholi	Bangalore	9865732546	1988-01-19	Batsman	2003-01-14	170000000	22223
11115	Devdutt Padikkal	Bangalore	9865325325	1994-01-12	Left-Handed Batsman	2015-01-13	120000000	22223

Stadium Table:

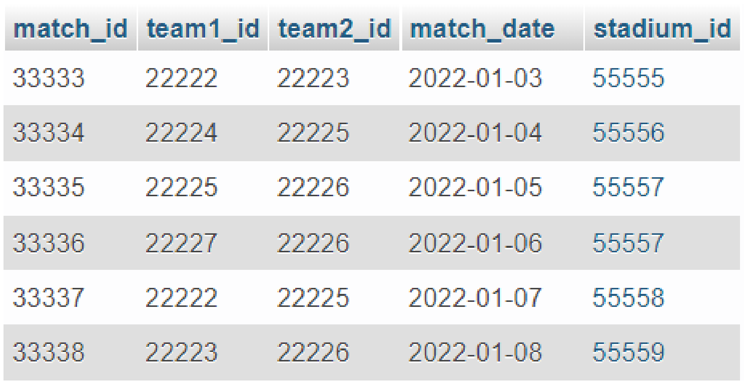
```
create table stadium(
        stadium_id int primary key,
        stadium_name varchar(255) not null,
        stadium_location varchar(255),
        capacity int
);
```



stadium_id	stadium_name	stadium_location	capacity
55555	M. Chinnaswamy Stadium	Bangalore	40000
55556	Eden Gardens	Kolkata	80000
55557	Narendra Modi Stadium	Ahmedabad	132000
55558	Rajiv Gandhi International Cricket Stadium	Hyderabad	55000
55559	Wanderers Stadium	Mumbai	33000
55560	Green Park Stadium	Kanpur	33000

Matches Table:

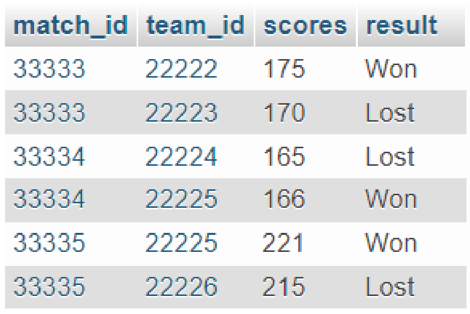
```
create table matches(
        match_id int primary key,
        team1_id int not null,
        team2_id int not null,
        match_date date,
        stadium_id int,
        foreign key(stadium_id) references stadium(stadium_id) on delete set null
);
```



match_id	team1_id	team2_id	match_date	stadium_id
33333	22222	22223	2022-01-03	55555
33334	22224	22225	2022-01-04	55556
33335	22225	22226	2022-01-05	55557
33336	22227	22226	2022-01-06	55557
33337	22222	22225	2022-01-07	55558
33338	22223	22226	2022-01-08	55559

Scores Table:

```
create table scores(
        match_id int,
        team_id int
        primary key(match_id, team_id),
        scores int,
        result varchar(20) not null,
        foreign key(match_id) references matches(match_id) on delete cascade,
        foreign key(team_id) references team(team_id) on delete cascade
);
```



match_id	team_id	scores	result
33333	22222	175	Won
33333	22223	170	Lost
33334	22224	165	Lost
33334	22225	166	Won
33335	22225	221	Won
33335	22226	215	Lost

5.2 Frontend Implementation

The backend and frontend are connected through JAVA programming language. The entire Front End is implemented using the Swing framework of Java. And the backend used in the project is MySQL.

5.2.1 Source Code

Login.java

```
private void loginActionPerformed(java.awt.event.ActionEvent evt) {
        if (username.getText().equals("admin") && password.getText().equals("pass")) {
                setVisible(false);
                new Home().setVisible(true);
        } else {
                    JOptionPane.showMessageDialog(null, "Incorrect Username or
        Password");
        }
}
private void exitActionPerformed(java.awt.event.ActionEvent evt) {
        int a = JOptionPane.showConfirmDialog(null, "Do you want to close Application",
        "Select", JOptionPane.YES_NO_OPTION);
        if (a == 0)
                    System.exit(0);
}
private void LogoutActionPerformed(java.awt.event.ActionEvent evt) {
        int a = JOptionPane.showConfirmDialog(null, "Do you want to Logout", "Select",
        JOptionPane.YES_NO_OPTION);
        if (a == 0) {
                setVisible(false);
                new Login().setVisible(true);
        }
}
```

ConnectionProvider.java

```
package CricketManagementSystem;
import java.sql.Connection;
import java.sql.DriverManager;
//Connect to the MySQL database
public class ConnectionProvider {
   public static Connection getConnection() {
         try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/cricket_database", "root", "");
            return con;
         } catch (Exception e) {}
         return null;
   }
}
```

AddUpdateDeletePlayer.java

```
private void saveActionPerformed(java.awt.event.ActionEvent evt) {
         int playerid = Integer.parseInt(player_id.getText());
         String playerName = player_name.getText();
         String addr = address.getText();
         BigInteger phon = new BigInteger(phone.getText());
         SimpleDateFormat dFormat = new SimpleDateFormat("yyyy-MM-dd");
         String birthDate = dFormat.format(dob.getDate());
         String rol = role.getText();
         String joiningDate = dFormat.format(joining_date.getDate());
         String sal = salary.getText();
         String teamid = team_id.getText();
         try {
            Connection con = ConnectionProvider.getConnection();
            Statement st = con.createStatement();
            st.executeUpdate("insert into player values('" + playerid + "', '" + playerName + "',
'" + addr + "', '" + phon + "', '" + birthDate + "', '" + rol + "', '" + joiningDate + "', '" + sal + "',
'" + teamid + "')");
```

```
        JOptionPane.showMessageDialog(this, "Successfully Saved");
        setVisible(false);
        new AddUpdateDeletePlayer().setVisible(true);
        } catch (SQLException ex) {
                Logger.getLogger(AddUpdateDeletePlayer.class.getName()).log(Level.SEVE
                RE, null, ex);
        }
}
private void updateActionPerformed(java.awt.event.ActionEvent evt) {
        int playerid = Integer.parseInt(player_id.getText());
        String playerName = player_name.getText();
        String addr = address.getText();
        BigInteger phon = new BigInteger(phone.getText());
        SimpleDateFormat dFormat = new SimpleDateFormat("yyyy-MM-dd");
        String birthDate = dFormat.format(dob.getDate());
        String rol = role.getText();
        String joiningDate = dFormat.format(joining_date.getDate());
        String sal = salary.getText();
        String teamid = team_id.getText();
        try {
           Connection con = ConnectionProvider.getConnection();
           //Query to insert data to student table
           pst = con.prepareStatement("update player set name = ?, address = ? , phone_no =
?, dob = ?, role = ?, since = ?, salary= ?, team_id = ? where player_id = ?");
           pst.setString(1, playerName);
           pst.setString(2, addr);
           pst.setBigDecimal(3, new BigDecimal(phon));
           pst.setString(4, birthDate);
           pst.setString(5, rol);
           pst.setString(6, joiningDate);
           pst.setString(7, sal);
           pst.setString(8, teamid);
           pst.setInt(9, playerid);
           pst.executeUpdate();
```

```
            JOptionPane.showMessageDialog(this, "Succesfully Updated");
            this.dispose();
            new AddUpdateDeletePlayer().setVisible(true);
        } catch (SQLException ex) {
                Logger.getLogger(AddUpdateDeletePlayer.class.getName()).log(Level.SEVE
                RE, null, ex);
        }
}
private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
        int playerid = Integer.parseInt(player_id.getText());
        try {
            Connection con = ConnectionProvider.getConnection();
            PreparedStatement pst = con.prepareStatement("delete from player where player_id
            = ?");
            pst.setInt(1, playerid);
            pst.executeUpdate();
            JOptionPane.showMessageDialog(this, "Succesfully Deleted");
            this.dispose();
            new AddUpdateDeletePlayer().setVisible(true);
        } catch (SQLException ex) {
            Logger.getLogger(AddUpdateDeletePlayer.class.getName()).log(Level.SEVERE,
            null, ex);
        }
}
```

ShowPlayerDetails.java

```
PreparedStatement pst;
ResultSet rs;
DefaultTableModel d;
public void userLoad() {
        int count;
        try {
                Connection con = ConnectionProvider.getConnection();
                pst = con.prepareStatement("select * from player");
                rs = pst.executeQuery();
                ResultSetMetaData rsd = rs.getMetaData();
                count = rsd.getColumnCount();
                d = (DefaultTableModel) tblPlayer.getModel();
                d.setRowCount(0);
                while (rs.next()) {
                        Vector v2 = new Vector();
                        for (int i = 1; i <= count; i++) {
                           v2.add(rs.getString("player_id"));
                           v2.add(rs.getString("name"));
                           v2.add(rs.getString("address"));
                           v2.add(rs.getString("phone_no"));
                           v2.add(rs.getString("dob"));
                           v2.add(rs.getString("role"));
                           v2.add(rs.getString("since"));
                           v2.add(rs.getString("salary"));
                           v2.add(rs.getString("team_id"));
                        }
                        d.addRow(v2);
                }
        } catch (SQLException ex) {
                Logger.getLogger(ShowPlayerDetails.class.getName()).log(Level.SEVERE,
                null, ex);
        }
}
```

CHAPTER 6

SNAPSHOTS

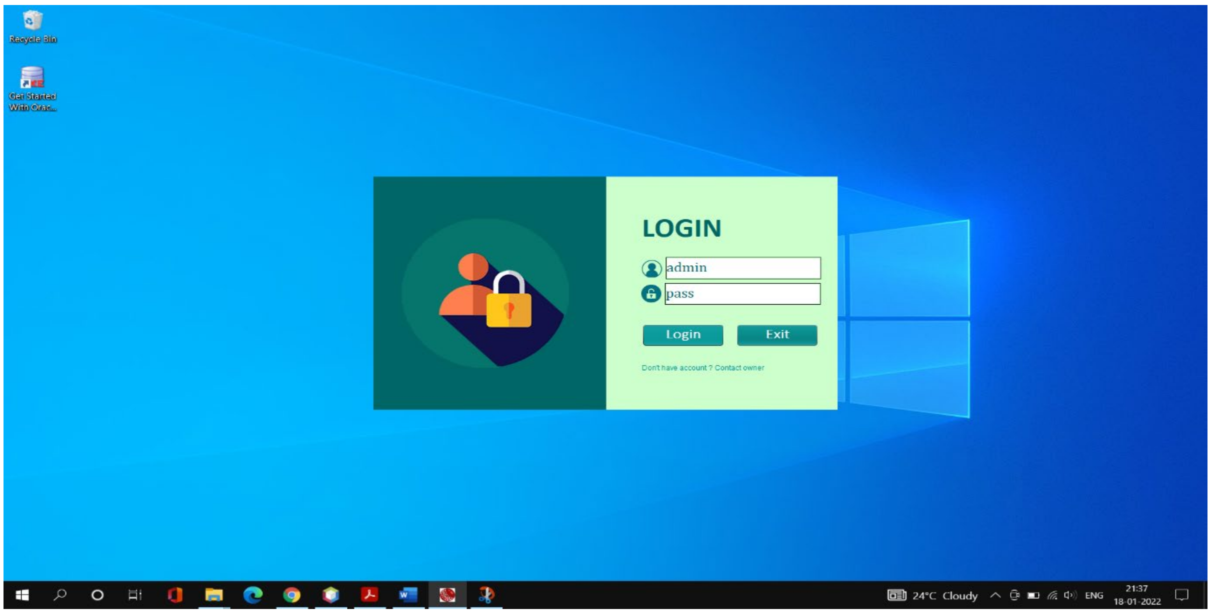


Figure 6.1 Admin Login Page

It shows admin login page which gives authentication to enter into the Admin page.

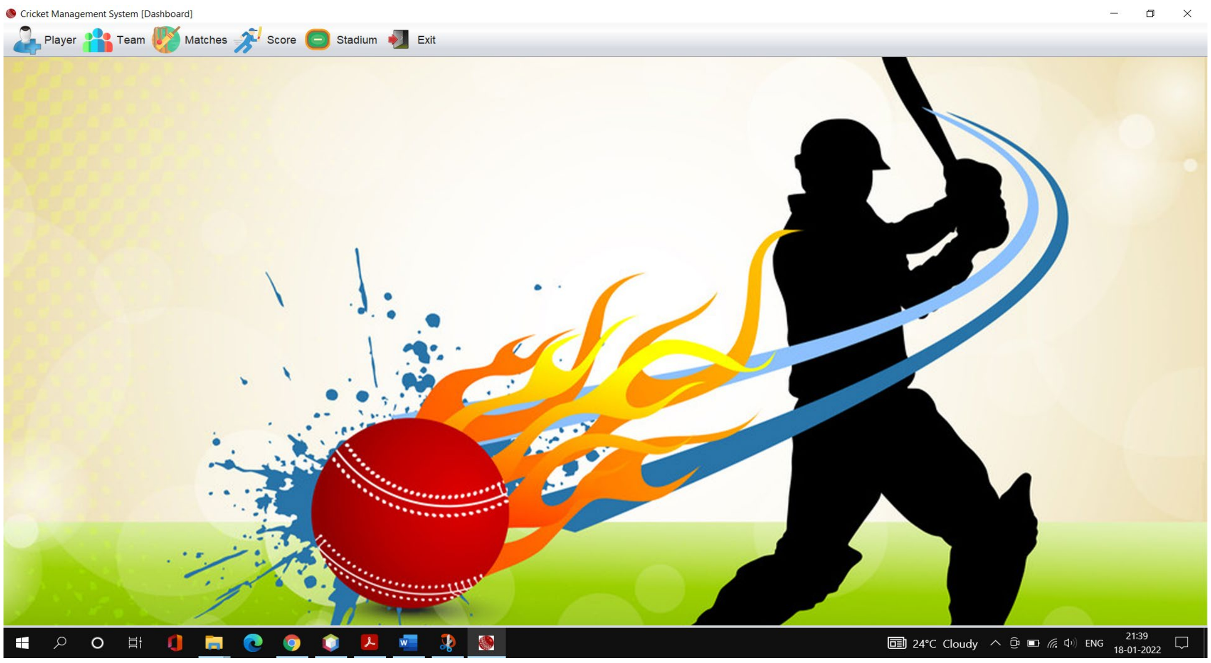


Figure 6.2 Home page

In the above Figure 6.2, shows which allows to add, update and delete of players, teams, matches, score, stadiums and about page.

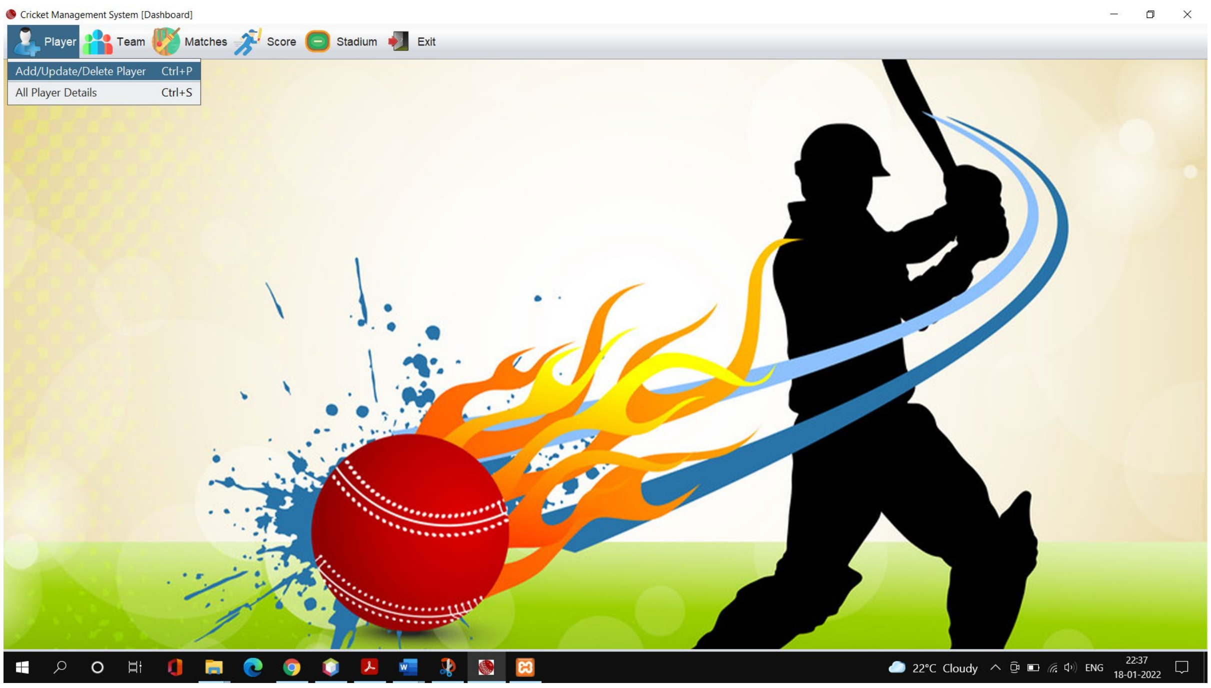
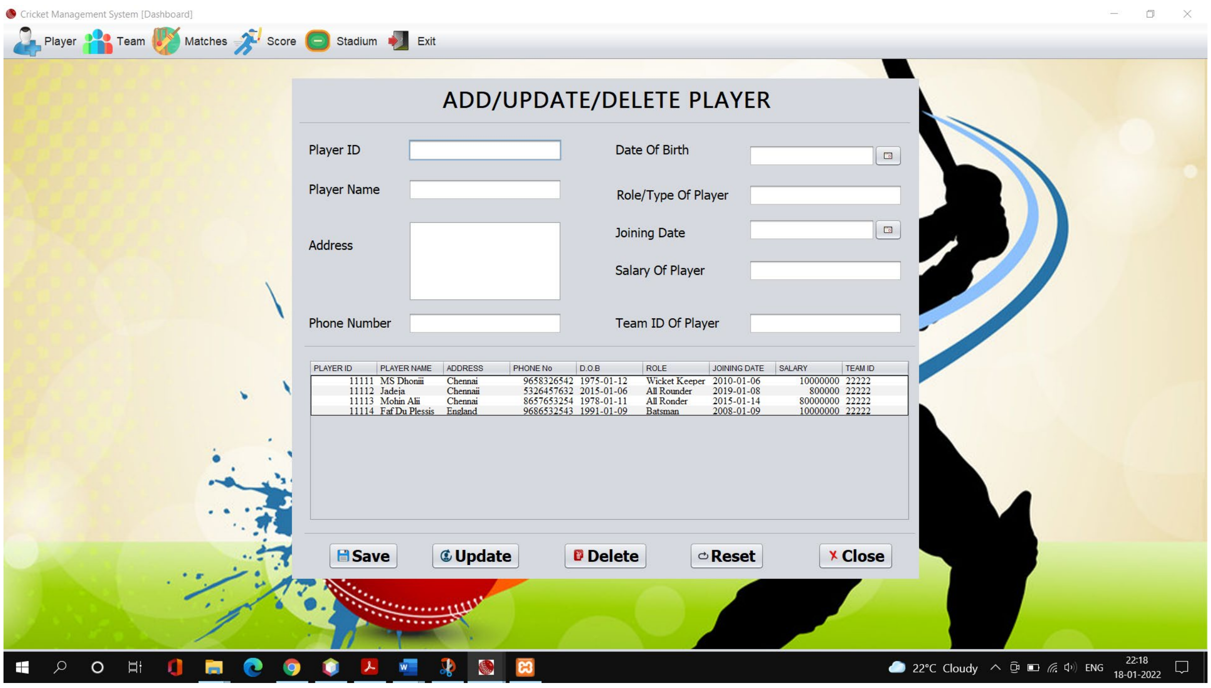

Cricket Management System [Dashboard]
Player
Team
Matches
Score
Stadium
Exit
Add/Update/Delete Player Ctrl+P
All Player Details Ctrl+S
22°C Cloudy
ENG
22:37
18-01-2022

Figure 6.3 Player menu



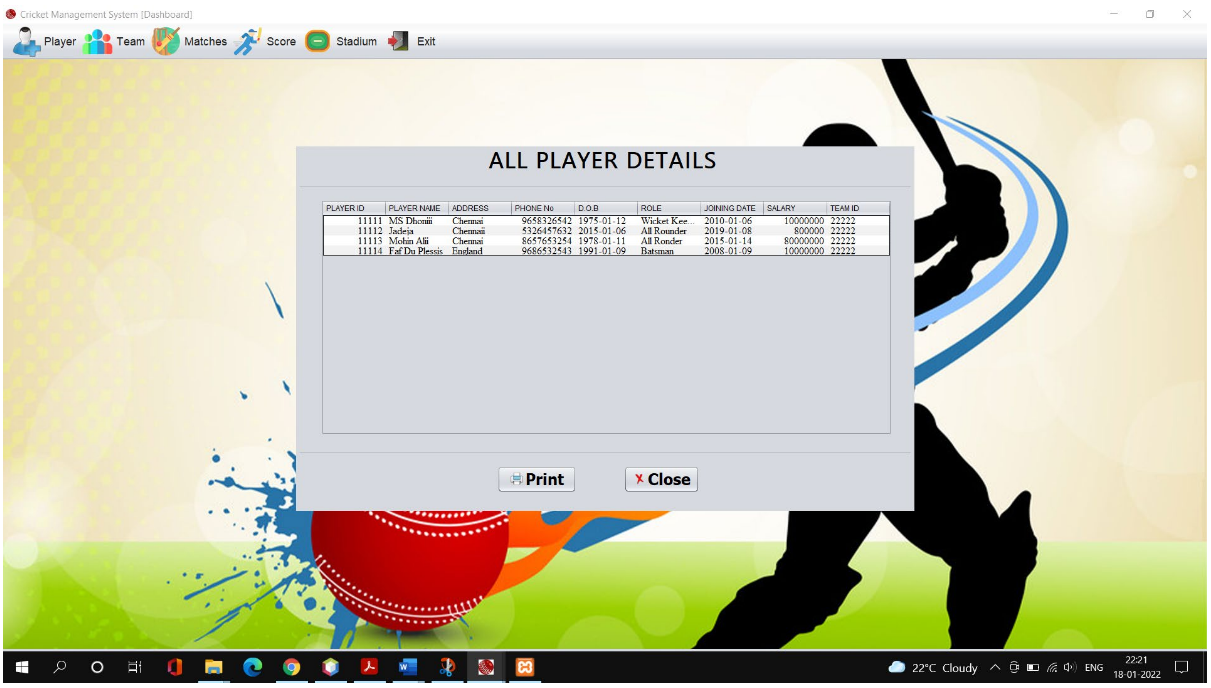
ADD/UPDATE/DELETE PLAYER

Player ID	
Player Name	
Address	
Phone Number	
Date Of Birth	
Role/Type Of Player	
Joining Date	
Salary Of Player	
Team ID Of Player	

PLAYER ID	PLAYER NAME	ADDRESS	PHONE No	D.O.B	ROLE	JOINING DATE	SALARY	TEAM ID
11111	MS Dhoniii	Chennai	9658326542	1975-01-12	Wicket Keeper	2010-01-06	10000000	22222
11112	Jadeja	Chennaii	5326457632	2015-01-06	All Rounder	2019-01-08	800000	22222
11113	Mohin Alii	Chennai	8657653254	1978-01-11	All Ronder	2015-01-14	80000000	22222
11114	Faf Du Plessis	England	9686532543	1991-01-09	Batsman	2008-01-09	10000000	22222

Figure 6.4 Add/ Update/ Delete player

In the above Figure 6.4, It show player window which allows admin to add, update and delete the Player details

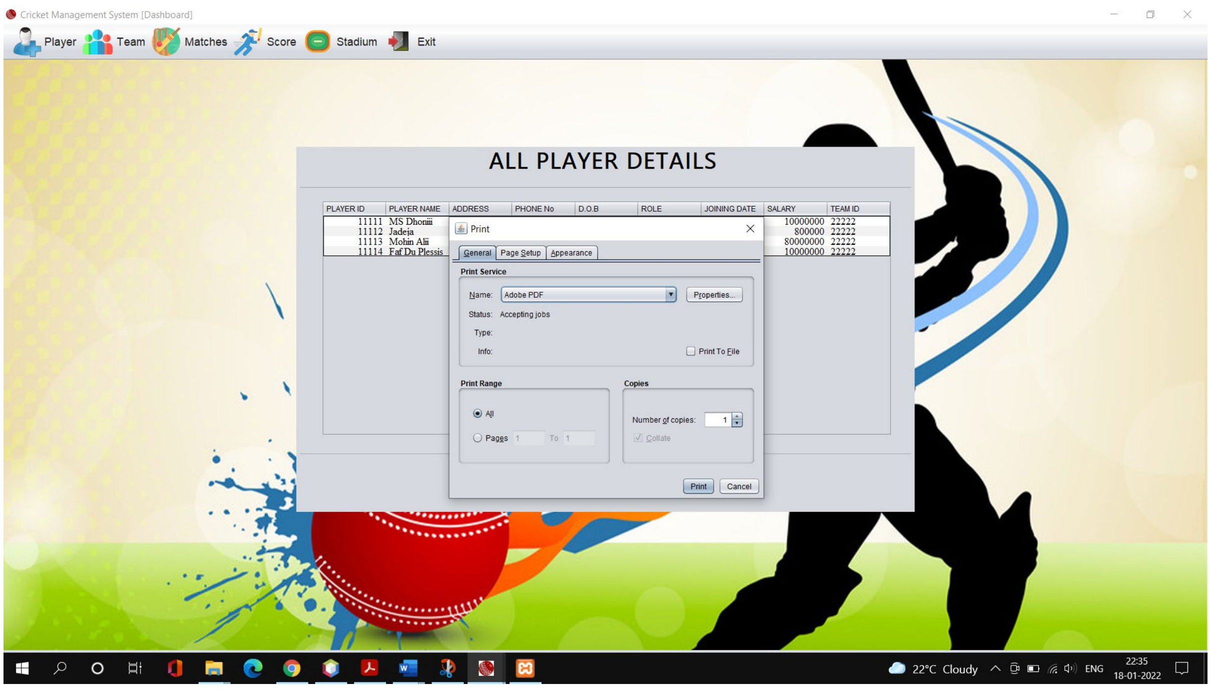


ALL PLAYER DETAILS

PLAYER ID	PLAYER NAME	ADDRESS	PHONE No	D.O.B	ROLE	JOINING DATE	SALARY	TEAM ID
11111	MS Dhoniii	Chennai	9658326542	1975-01-12	Wicket Kee...	2010-01-06	10000000	22222
11112	Jadeja	Chennaii	5326457632	2015-01-06	All Rounder	2019-01-08	800000	22222
11113	Mohin Alii	Chennai	8657653254	1978-01-11	All Ronder	2015-01-14	80000000	22222
11114	Faf Du Plessis	England	9686532543	1991-01-09	Batsman	2008-01-09	10000000	22222

Figure 6.5 Player details

In the above Figure 6.5, It shows player details window which allows admin to print the details of player from the connected printer or just can save as pdf document.

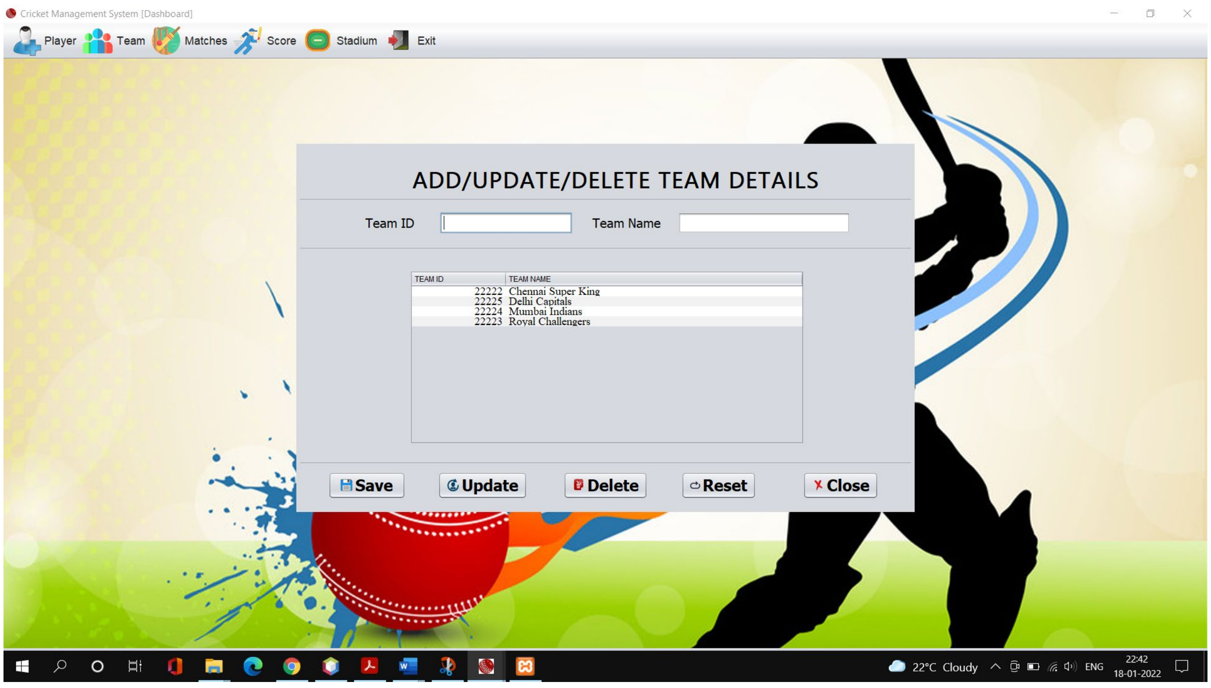


Print Service
Name: Adobe PDF
Status: Accepting jobs
Type:
Info:
☐ Print To File

Print Range
● All
○ Pages 1 To 1

Copies
Number of copies: 1
✓ Collate

Figure 6.6 Print player details



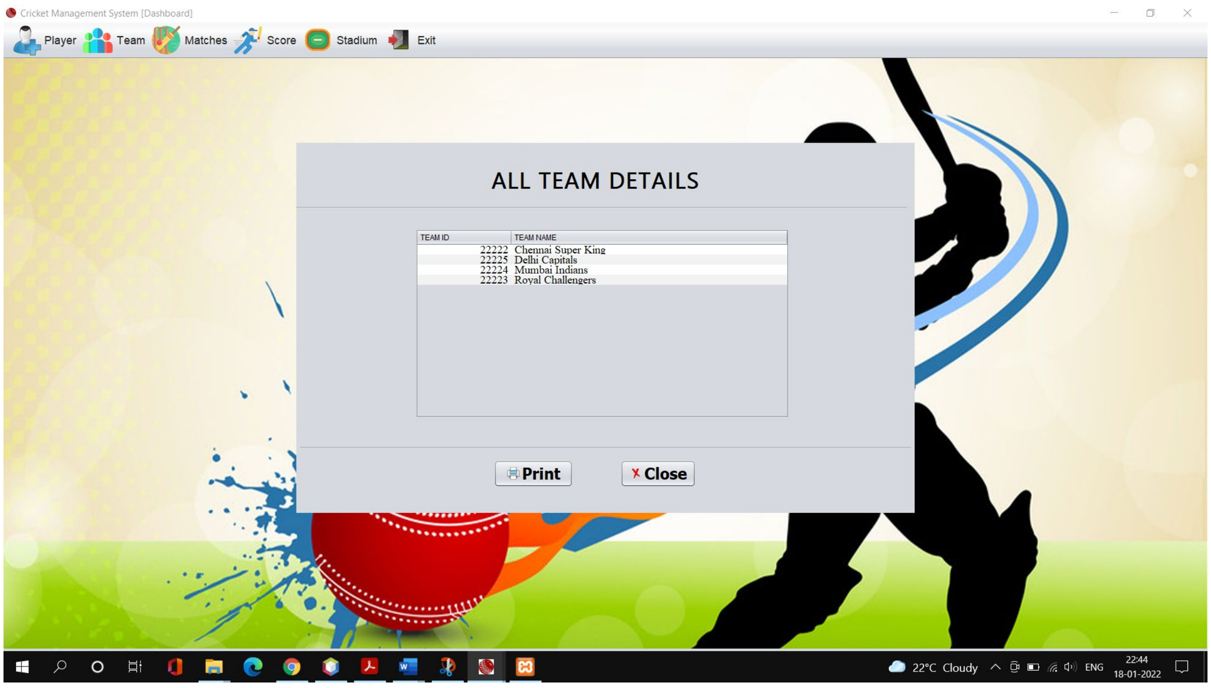
ADD/UPDATE/DELETE TEAM DETAILS

Team ID | Team Name

TEAM ID	TEAM NAME
22222	Chennai Super King
22225	Delhi Capitals
22224	Mumbai Indians
22223	Royal Challengers

Figure 6.7 Add/ Update/ Delete Team

In the above Figure 6.7, It show team window which allows admin to add, update and delete the team details

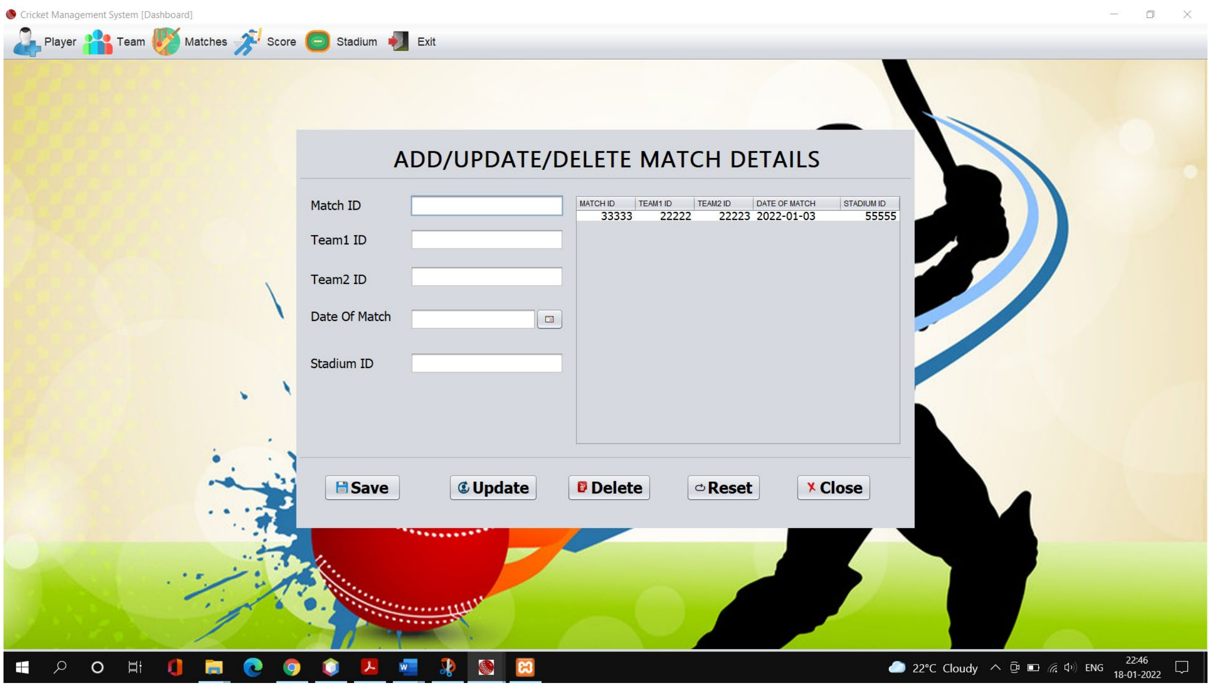


ALL TEAM DETAILS

TEAM ID	TEAM NAME
22222	Chennai Super King
22225	Delhi Capitals
22224	Mumbai Indians
22223	Royal Challengers

Figure 6.8 Team details

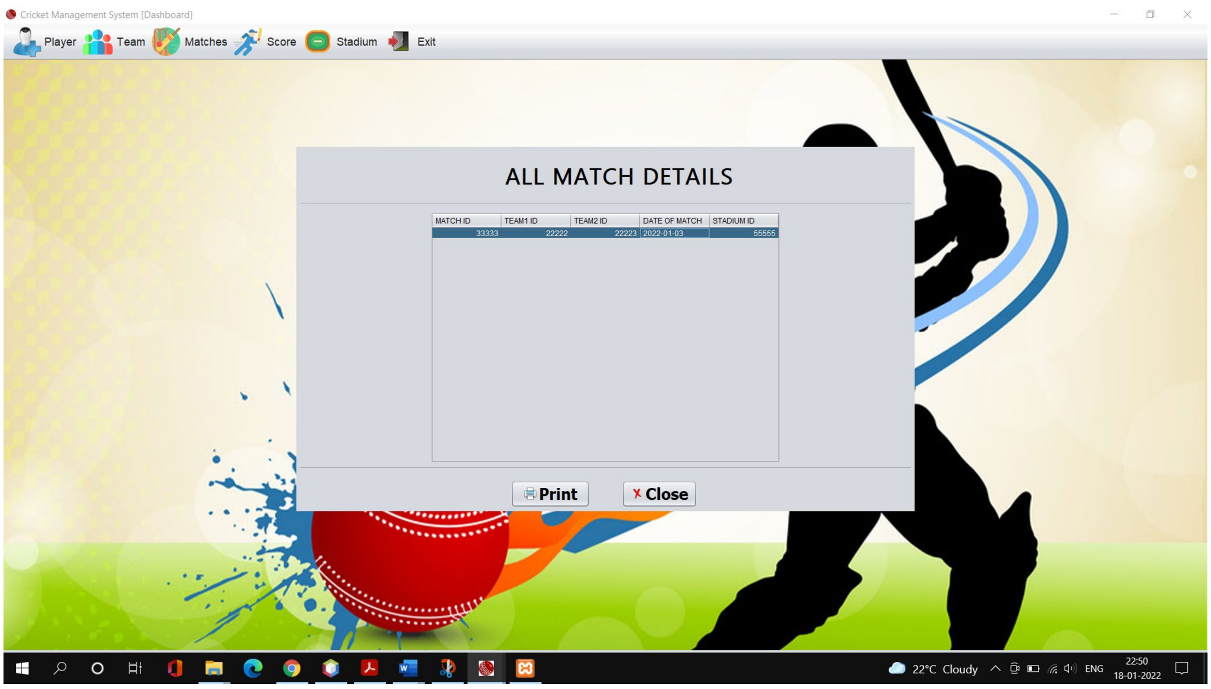
In the above Figure 6.8, It shows team details window which allows admin to print the details of team from the connected printer or just can save as pdf document.



MATCH ID	TEAM1 ID	TEAM2 ID	DATE OF MATCH	STADIUM ID
33333	22222	22223	2022-01-03	55555

Figure 6.9 Add/ Update/ Delete match

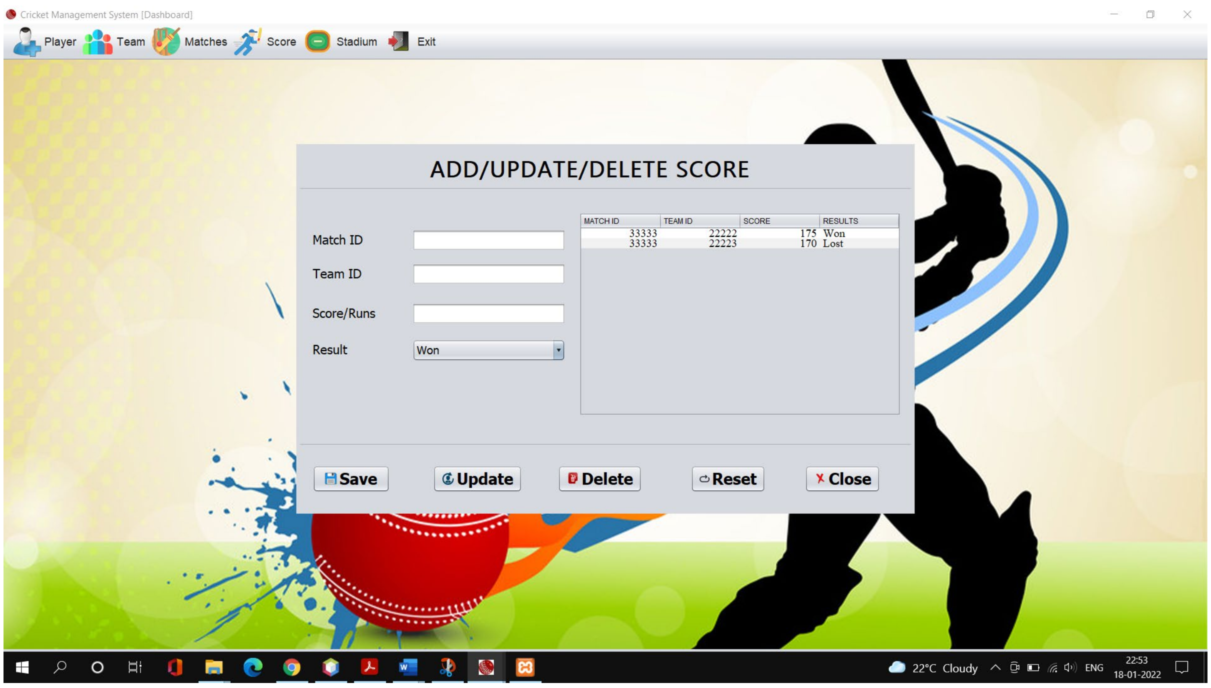
In the above Figure 6.9, It shows match details window which allows admin to schedules the matches between the teams and also can update or delete a scheduled match.



MATCH ID	TEAM1 ID	TEAM2 ID	DATE OF MATCH	STADIUM ID
33333	22222	22223	2022-01-03	55555

Figure 6.10 Match details

In the above Figure 6.10, It shows match details window which allows admin to print the details of match from the connected printer or just can save as pdf document.



ADD/UPDATE/DELETE SCORE

Match ID

Team ID

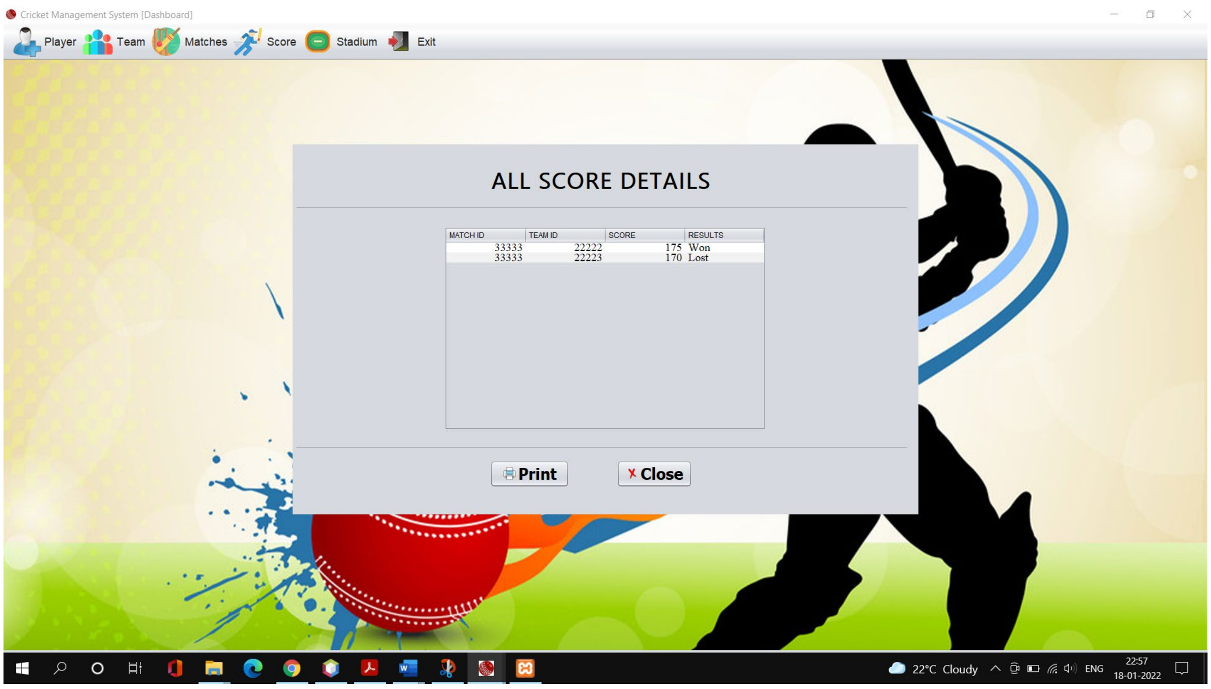
Score/Runs

Result: Won

MATCH ID	TEAM ID	SCORE	RESULTS
33333	22222	175	Won
33333	22223	170	Lost

Figure 6.11 Add/ Update/ Delete score

In the above Figure 6.11, It shows score window which allows admin to add score of a particular team and can also update or delete the score.

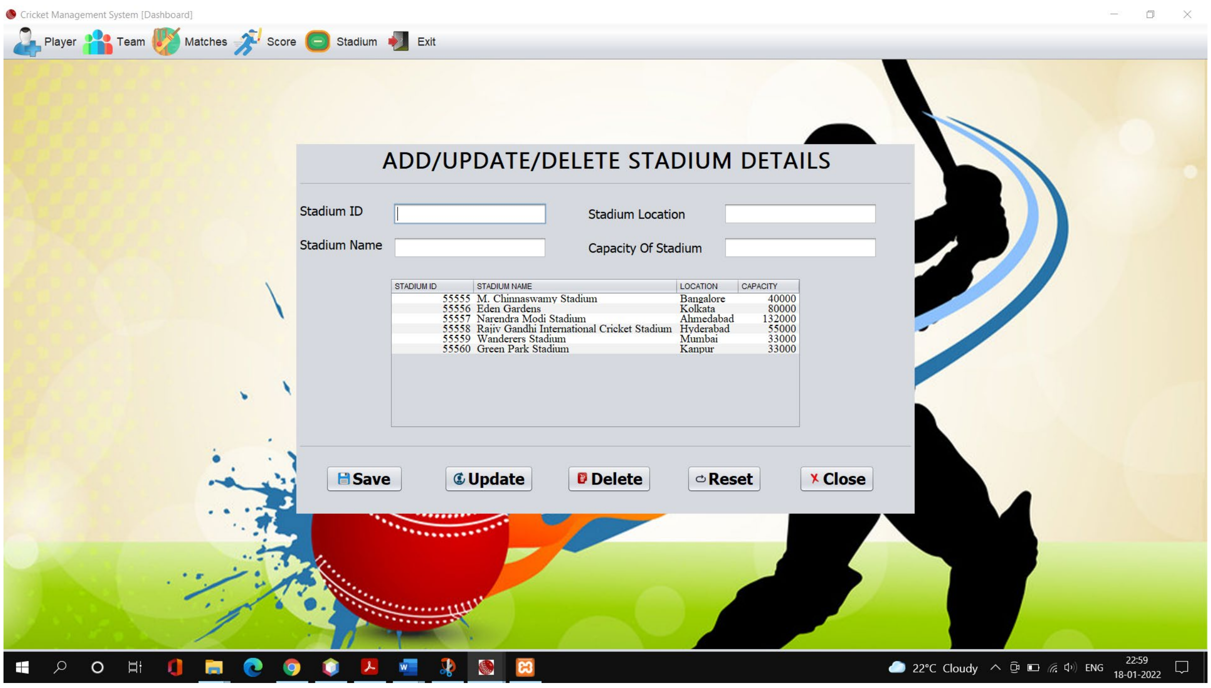


ALL SCORE DETAILS

MATCH ID	TEAM ID	SCORE	RESULTS
33333	22222	175	Won
33333	22223	170	Lost

Figure 6.12 Score details

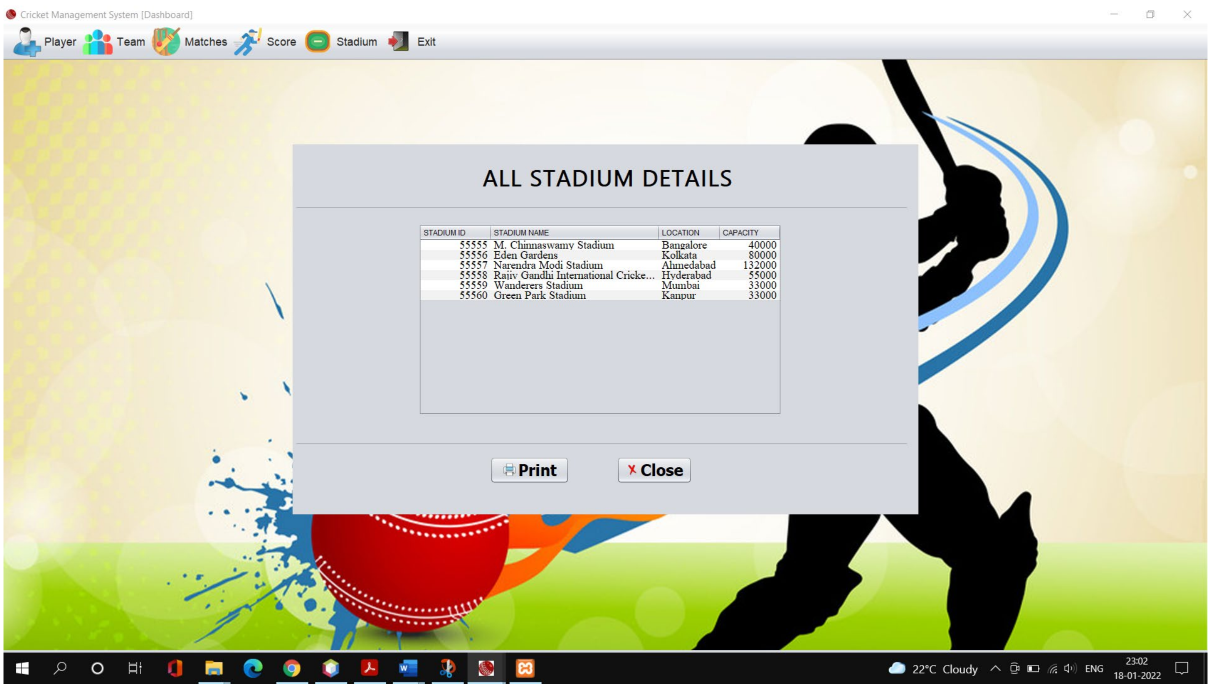
In the above Figure 6.12, It shows score details window which allows admin to print the details of score from the connected printer or just can save as pdf document.



STADIUM ID	STADIUM NAME	LOCATION	CAPACITY
55555	M. Chinnaswamy Stadium	Bangalore	40000
55556	Eden Gardens	Kolkata	80000
55557	Narendra Modi Stadium	Ahmedabad	132000
55558	Rajiv Gandhi International Cricket Stadium	Hyderabad	55000
55559	Wanderers Stadium	Mumbai	33000
55560	Green Park Stadium	Kanpur	33000

Figure 6.13 Add/ Update/ Delete Stadium details

In the above Figure 6.13, It shows stadium window which allows admin to add a new stadium and can also update or delete the stadium details.



STADIUM ID	STADIUM NAME	LOCATION	CAPACITY
55555	M. Chinnaswamy Stadium	Bangalore	40000
55556	Eden Gardens	Kolkata	80000
55557	Narendra Modi Stadium	Ahmedabad	132000
55558	Rajiv Gandhi International Cricke...	Hyderabad	55000
55559	Wanderers Stadium	Mumbai	33000
55560	Green Park Stadium	Kanpur	33000

Figure 6.14 Stadium details

In the above Figure 6.14, It shows stadium details window which allows admin to print the details of stadium from the connected printer or just can save as pdf document.

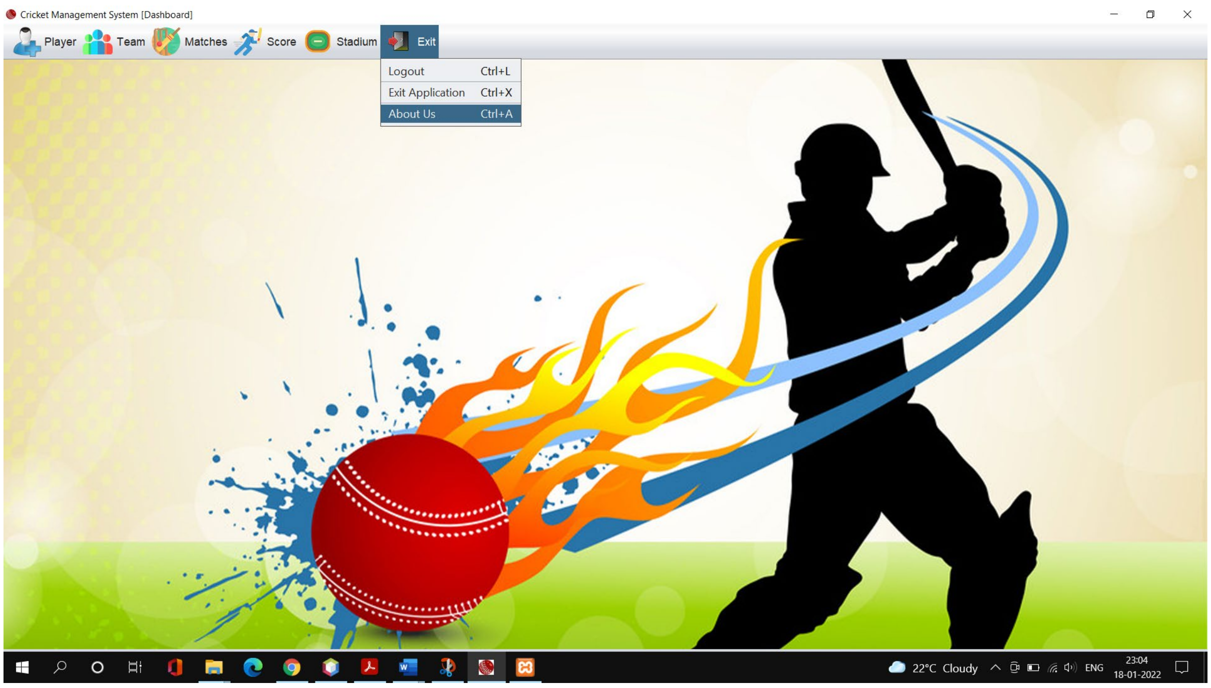
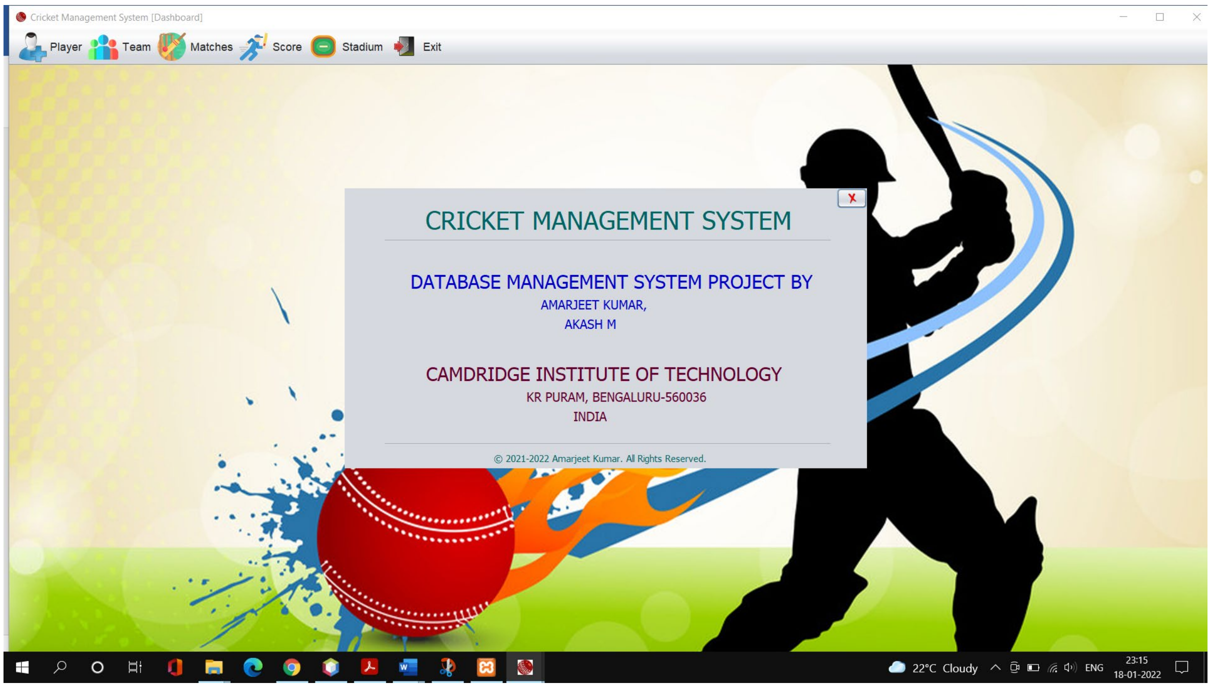


Figure 6.15 Exit menu

In the above Figure 6.15, It show the exit menu which has got Logout, Exit Application and About Us



CRICKET MANAGEMENT SYSTEM

DATABASE MANAGEMENT SYSTEM PROJECT BY
AMARJEET KUMAR,
AKASH M

CAMDRIDGE INSTITUTE OF TECHNOLOGY
KR PURAM, BENGALURU-560036
INDIA

© 2021-2022 Amarjeet Kumar. All Rights Reserved.

Figure 6.16 About Us

CONCLUSION

The project, developed using JAVA and MySQL is based on the requirement specification of the user and the analysis of the existing system, with flexibility for future enhancement. The expanded functionality of today’s software requires an appropriate approach towards software development. This Cricket database management software is designed for people who want to manage various particulars can be known by recording them in the database. Various records and particulars about match got increased rapidly. Thereby the numbers of matches and there is going to be increased day-by-day. And hence there is a lot of strain on the person who are watching the IPL to know about future matches and also to see the records done by various players and getting datils in fingertips. Identification of the drawbacks of the existing system leads to the designing of computerized system that will be compatible to the existing system with the system which is more user friendly and more GUI oriented.

REFERENCES

[1] Ramez Elmasri and Shamkant B. Navathe, “Database systems Models, Languages, Design and Application Programming”,6th Edition, Pearson,2017.

[2] Ramakrishnan and Gehrke, “Database management systems”, 3rd Edition McGraw Hill 2014

[3] Silberschatz Korth and Sudharshan, “Database System Concepts”, 6th Edition Mc-Graw Hill, 2013.

[4] Coronel, Morris,and Rob “Database Principles Fundamentals of Design, Implementation and Management” Cengage Learning 2012

[5] Abraham Silberchatz, Henry korth and S.Sudarshan, “Database System Concepts”, McGraw-Hill Education, 16 Jun 2010.

[6] Jeffery D.Ullman, “Principles of Data Base System”,Financial Times Prentice Hall 2nd Revised edition(1 December 1982).

[7] http://stackoverflow.com

[8] https://www.w3schools.com

[9] http://www.phptpoint.com

[10] https://www.bootply.com/

[11] https://www.tutorialspoint.com

[12] http://1000projects.org/

[13] https://erdplus.com/#/

Vision

To become a premier institute transforming our students to be global professionals.

Mission

M1: Develop competent Human Resources, and create state-of-the-art infrastructure to impart quality education and to support research.

M2: Adopt tertiary approach in teaching – learning pedagogy that transforms students to become professionally competent technocrats and entrepreneurs.

M3: Nurture and train students to develop the qualities of global professionals.

Department of Computer Science and Engineering

Vision

To impart quality education in the field of Computer Science and Engineering with emphasis on innovative thinking, communication and leadership skills to meet the global challenges in IT paradigm.

Mission

M1: Focus on student centric approach through experiential learning and necessary infrastructure.

M2: Develop innovative thinking, communication and leadership skills by creating conducive environment and relevant training.

M3: Enrich students by developing the traits of global professionals.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

CAMBRIDGE INSTITUTE OF TECHNOLOGY

K. R. PURAM, BENGALURU - 560036