VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590018, KARNATAKA

A MINI PROJECT REPORT
ON
"FIFA 23 PLAYER MANAGEMENT SYSTEM"

Submitted in partial fulfillment of requirements for the award of 5th

semester,

BACHELOR OF ENGINEERING IN
COMPUTER SCIENCE & ENGINEERING

Submitted By:
Parmekar Shubham Rajendra
(USN: 1KT20CS052)
Inao Lakhong Singpho
(USN: 1KT21CS401)
Under the Guidance of
Mrs. Savita Patil &Ms. Sushma M,
Associate Professor,
Department of Computer Science & Engineering

DEPARTMENT OF COMPUTER SCIENCE &
ENGINEERING
SRI KRISHNA INSTITUTE OF TECHNOLOGY
BENGALURU-90
2022-23

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the mini project entitled "FIFA 23 PLAYER MANAGEMENT SYSTEM" as a part of 18CSL58 laboratory, is bonafide work carried out by PARMEKAR SHUBHAM RAJENDRA bearing USN:1KT20CS052 and INAO LAKHONG SINGPHO bearing USN:1KT21CS401 partial fulfillment for the award of degree in bachelor of Engineering in Computer Science Engineering from Visvesvaraya Technological University, Belagavi during the academic year 2021-22. It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in reports submitted in the department Library. This mini project report has been approved as it satisfies the academic requirements in respect of the mini project report prescribed for award of said degree.

Signature of Internal Guide	Signature of the HOD	Signature of Principal
Mrs. SAVITA PATIL &	**Dr. SHANTHARAM NAYAK**	**Dr. MAHESHA K**
Ms. SUSHMA M		
Assistant professor	Professor & head	Principal
Dept. of. CSE, SKIT	Dept. of. CSE, SKIT	SKIT, Bangalore

Name of the Examiner   Signature with date

ABSTRACT

Managing the ever increasing numbers of players in different parts of the world is a huge task. This project is aimed at developing a desktop-based application named 'FIFA 23 player management system' for managing players using a robust database at the backend and a Web based GUI at the frontend.

The application will allow users to track complete details about a player starting from his personal details, going through club and nationality information to right down to his technicalities at each position in footballing world. The software also allows users to view the whole list of players, teams and footballing statistics at once, thereby helping them build their perspective. Users have the privilege to add new players to a particular team, and to modify their records when the player decides to retire. FIFA 23 player management system also allows users to access players based on their rating other than their preferential position of playing thus guiding managers to build a strong positional team by selecting best rated player at each position. In conclusion, this application will come extremely handy in maintaining player spread across different teams and nations.

ACKNOWLEDGEMENT

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The project titled "**FIFA 23 Player management system**" is player management software for monitoring and accessing players based on their FIFA 23 PC/XBOX Game ratings. This project is developed using HTML, CSS and JavaScript for front-end and PHP, MySQL for back-end, which focuses on basic operation like adding a new player, new statistics, searching players with detailed information and edit as they grow their skills.

This project is a web based application designed and developed to help user's access players and organize teams. This software is easy to use, and it features a familiar and well-thought-out attractive user interface, combined with strong searching, insertion, and deletion with procedure capabilities.

Analysing players have been a huge task performed by professional scouting agents who are spread around the world. From personal details to football technicalities, FIFA 23 Management system allows easy maintenance record of such skilled youth talent.

Chapter 2

SYSTEM REQUIREMENT

One of the most difficult tasks is that, the selection of the software, once system requirement is known is determining whether a software package fits the requirements. After initial selection further security is needed to determine the desirability of software compared with other candidates. This section first summarizes the application requirement question and then suggests more detailed comparisons.

Hardware Requirement

1. 32/64-bit processor
2. i5 or greater intel processor chip
3. 2.1 or more GHz processor

Software Requirement

1. Windows 10 or higher version OS
2. WAMPP web server
3. Brackets web editor

2.1 Software and Executables memory size:

Fig 2.1 Memory consumption by brackets web editor

Fig 2.2 Memory consumption by Microsoft Edge
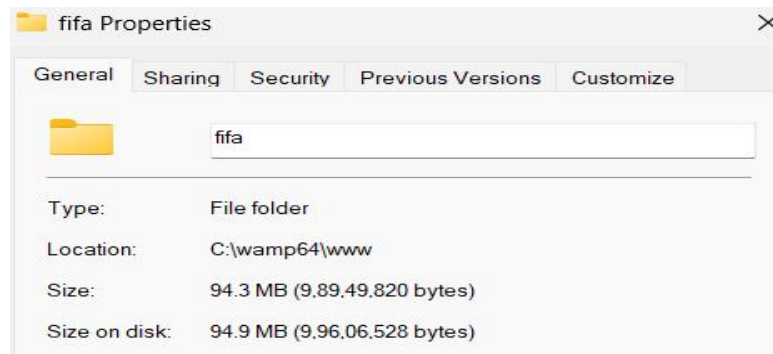
Fig 2.3 Project size on disk
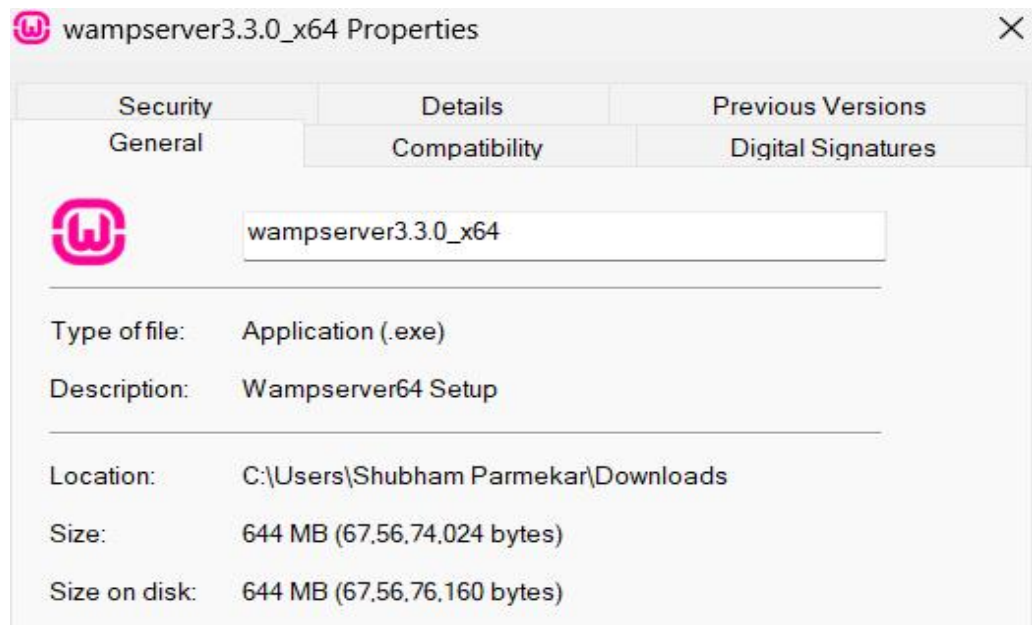
Fig 2.4 Wamp web server size on disk

Fig 2.5 Memory consumption by local server (Wamp)

Chapter 3

DESIGN OF THE PROJECT

To analyse and access players on a daily basis can be hectic and huge effort demanding task. To help users, managers, staffs and scouting agents in football world, FIFA 23 player management system provides effortless player management system to help users to analyse, improve, train and access plays on a daily basis.

Description of the Project:

This project consists of player details, which describes about player biodata such as age and nationality. It also consists of player stats which describes about players technical skills. It also consists of tables containing details such as player earnings, club information and preferred position of playing. It also provide a strong searching, updating, deleting and inserting operations with a user friendly web based UI.

The project also helps the users to keep track of the player details in a computerized way without any trouble. The project contains **7 stored procedures** and **3 triggers** per table. Stored procedures are used in search engine. Every time the user searches through the database, a procedure is called and the results is collected and displayed for the user in a structured manner. It also has 3 trigger namely "**Insert, Delete and Update**" triggers assigned separately to each table. Whenever operations such as insert or delete or update is performed on any table, these triggers are automatically called, and the logs are captured into 3 separate tables, individually for each trigger. Hence use of triggers provides users to trace back all the latest as well as the oldest changes into any table at any point of time.

This project is a simple prototype of managing larger numbers of players across different nations with different skill sets and attributes. It helps to access players and thus aids in building a strong positional team. It also helps in monitoring player growth.

3.1 ER Diagram:
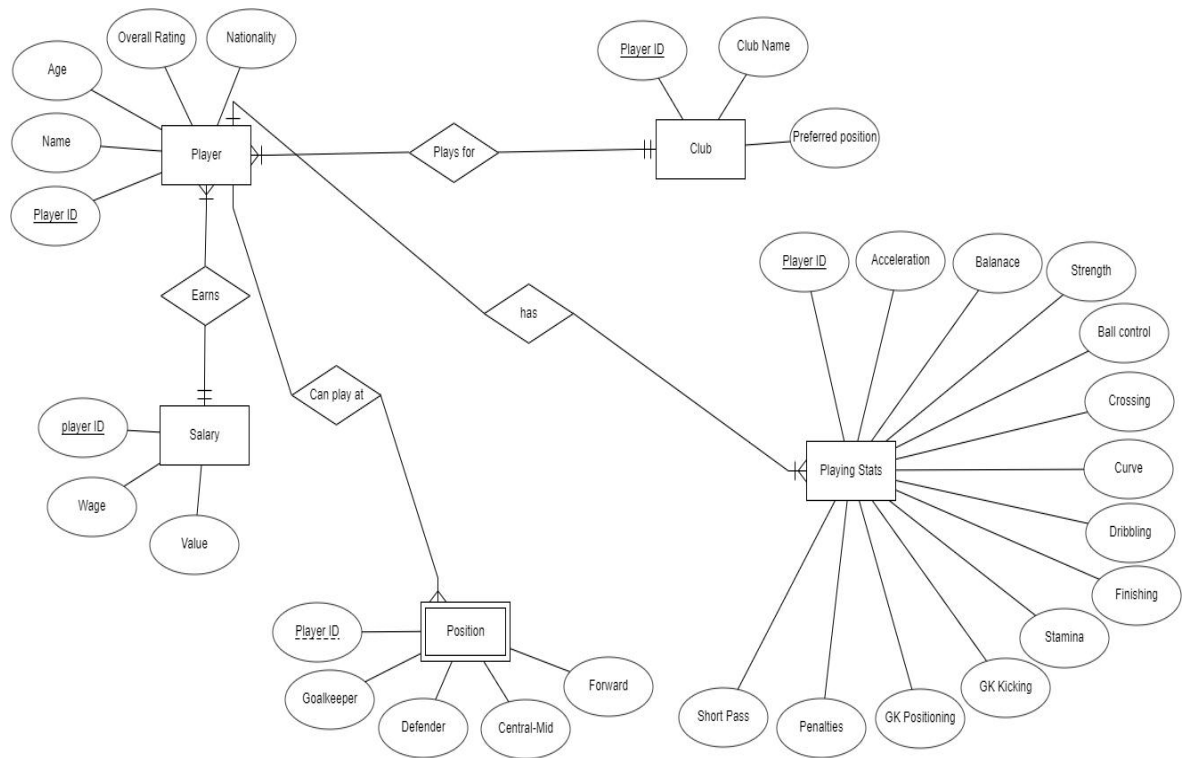
Fig 3.1 ER Diagram

3.2 Schema Diagram:

Player

Player ID	Name	Age	Overall rating	Nationality

Position

Player ID	Goalkeeper	Defender	Central-Mid	Forward

Salary

Player ID	Wage	Value

Playing Stats

Player ID	Acceleration	Balance	Ball control	Curve	Crossing	Dribbling	Finishing

Club

Player ID	Club	Preferred position
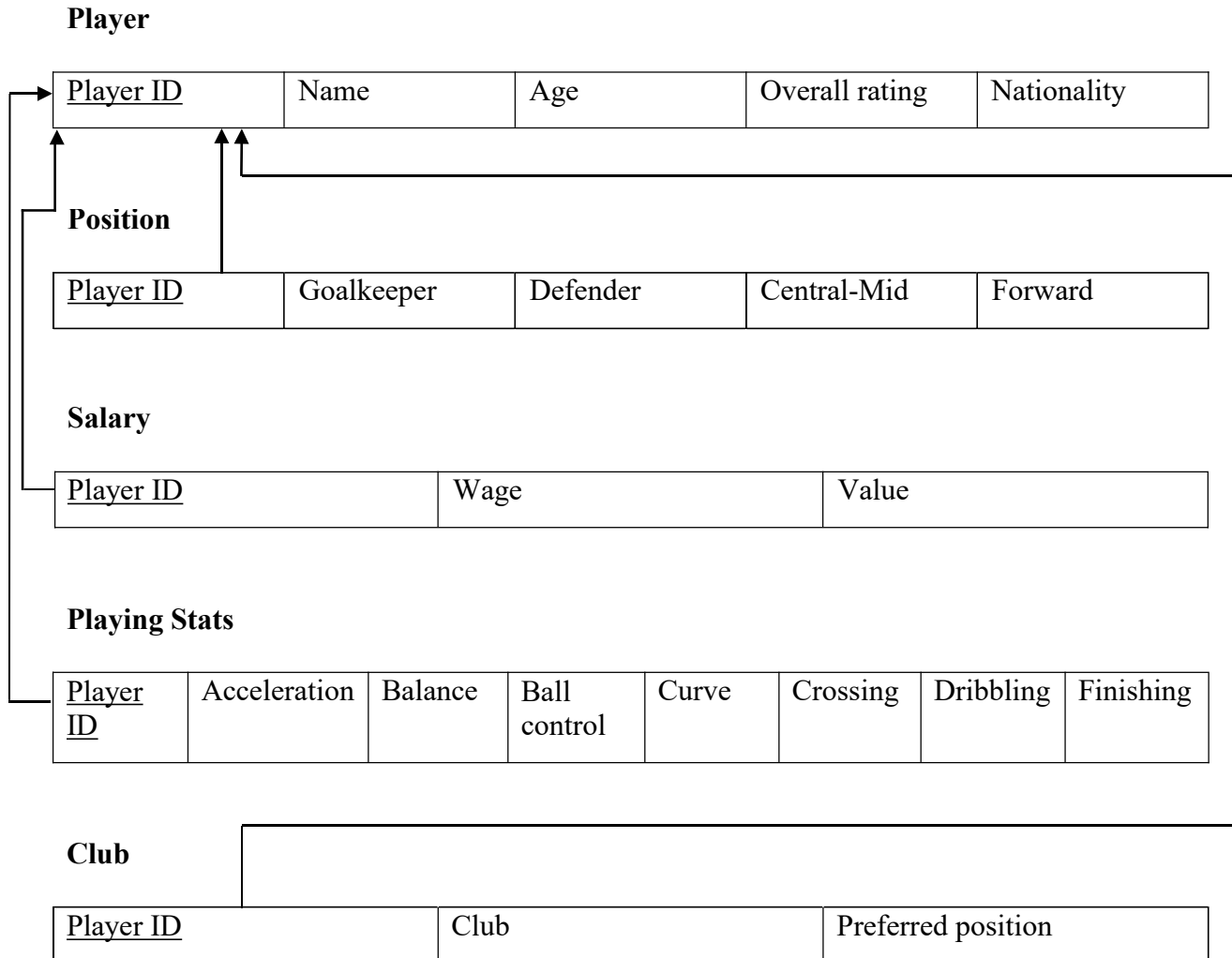
Fig 3.2 Relational schema

3.3 Table Structures:

3.3.1 Player details

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id 🔑	int(11)			No	None		AUTO_INCREMENT	🖉 Change ⊖ Drop ▾ More
☐	2	player_id 🔑	int(7)			No	None			🖉 Change ⊖ Drop ▾ More
☐	3	player_name 🔑	char(30)	latin1_swedish_ci		No	None			🖉 Change ⊖ Drop ▾ More
☐	4	age	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	5	overall_rating	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	6	nationality	char(30)	latin1_swedish_ci		Yes	NULL			🖉 Change ⊖ Drop ▾ More

Fig 3.3 Player table structure

The player table consists of 6 columns. Player_id and player_name are primary keys and player_id have references of other tables as well. To insert data, player_id should exist in this table before inserting into other tables.

3.3.2 Stats details

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id 🔑	int(3)			No	None		AUTO_INCREMENT	🖉 Change ⊖ Drop ▾ More
☐	2	player_id 🔑	int(7)			No	None			🖉 Change ⊖ Drop ▾ More
☐	3	acceleration	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	4	balance	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	5	ball_control	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	6	crossing	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	7	curve	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	8	dribbling	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	9	finishing	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	10	gk_kicking	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	11	gk_positioning	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	12	penalties	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	13	short_pass	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	14	stamina	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More
☐	15	strength	int(2)			Yes	NULL			🖉 Change ⊖ Drop ▾ More

Fig 3.4 Player stats table structure

Player stats table consists of 14 attributes, among which, player_id is primary key and also has a foreign key reference to "player" table. It is designed to contain all the football technicalities of a player.

3.3.3 Salary details

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(11)			No	None		AUTO_INCREMENT	🖊 Change	⊖ Drop	▽ More
☐	2	player_id 🔑	int(11)			No	None			🖊 Change	⊖ Drop	▽ More
☐	3	wage	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More
☐	4	value	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More

Fig 3.5 Salary table structure

The salary table consists player_id as primary key and also have a foreign key reference to "player" table. It is designed to store player weekly wage and his current value in the market.

3.3.4 Position details

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(11)			No	None		AUTO_INCREMENT	🖊 Change	⊖ Drop	▽ More
☐	2	player_id 🔑	int(11)			No	None			🖊 Change	⊖ Drop	▽ More
☐	3	gk	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More
☐	4	df	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More
☐	5	cm	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More
☐	6	fr	int(11)			Yes	NULL			🖊 Change	⊖ Drop	▽ More

Fig 3.6 Position table structure

The position table also have player_id as primary key and a foreign key reference to "player" table. It is designed to store the positions a player can play, if so, then how well he does on a scale of rating from 0 to 99. It helps user to access player based on positional play and decide the best position for a player.

3.3.5 Club details

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(11)			No	None		AUTO_INCREMENT	🖊 Change	⊖ Drop	▽ More
☐	2	player_id 🔑	int(11)			No	None			🖊 Change	⊖ Drop	▽ More
☐	3	club	char(30)	latin1_swedish_ci		Yes	NULL			🖊 Change	⊖ Drop	▽ More
☐	4	preferred_position	char(20)	latin1_swedish_ci		Yes	NULL			🖊 Change	⊖ Drop	▽ More

Fig 3.7 Club table structure

The club details table has club information and the preferred position of a player at that club. It also have player_id as primary key and also a foreign key reference on "player" table.

3.3.6 Delete logs

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(10)			No	*None*		AUTO_INCREMENT	🖉 Change	⊖ Drop	▼ More
☐	2	action	varchar(50)	utf8mb4_unicode_ci		No	*None*			🖉 Change	⊖ Drop	▼ More
☐	3	time	timestamp			No	*None*			🖉 Change	⊖ Drop	▼ More

Fig 3.8 Delete logs trigger structure

The delete logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

3.3.7 Update logs

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(10)			No	*None*		AUTO_INCREMENT	🖉 Change	⊖ Drop	▼ More
☐	2	action	varchar(50)	utf8mb4_unicode_ci		No	*None*			🖉 Change	⊖ Drop	▼ More
☐	3	time	timestamp			No	*None*			🖉 Change	⊖ Drop	▼ More

Fig 3.9 Update logs trigger structure

The Update logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

3.3.8 Insert logs

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	id 🔑	int(10)			No	*None*		AUTO_INCREMENT	🖉 Change	⊖ Drop	▼ More
☐	2	action	varchar(50)	utf8mb4_unicode_ci		No	*None*			🖉 Change	⊖ Drop	▼ More
☐	3	time	timestamp			No	*None*			🖉 Change	⊖ Drop	▼ More

Fig 3.10 Insert logs trigger structure

The Insert logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

3.3.9 Stored procedures

Fig 3.11 Stored procedures structure

There are 7 stored procedures present inside search page on the web application. These are called whenever any search instance occur on the web page. The results of the stored procedures are then displayed on the UI in a tabular structure.

Chapter 4
IMPLEMENTATION

4.1 Stored procedures:

The Stored procedures are executed using PHP and MySQL in the application file. The result of stored procedure is shown in Fig 5.2.

Use: Use of stored procedures allows modular programming, reduces network traffic, faster execution, and can be used as security mechanism.

Given below is a code snippet of implementation of all the 7 stored procedure present inside the web application.

#stored procedures

```
$spforage = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost` PROCEDURE
`SearchAge`(IN `page` INT(11)) NOT DETERMINISTIC CONTAINS SQL SQL
SECURITY DEFINER SELECT player_name,age,overall_rating,nationality FROM
personal_details WHERE personal_details.age = page;");


$spfornationality = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost`
PROCEDURE `SearchNationality`(IN `page` VARCHAR(30)) NOT DETERMINISTIC
CONTAINS SQL SQL SECURITY DEFINER SELECT * FROM personal_details WHERE
personal_details.nationality=page;");


$spforoverallrating = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost`
PROCEDURE `SearchOverallRating`(IN `page` INT(11)) NOT DETERMINISTIC
CONTAINS SQL SQL SECURITY DEFINER SELECT * FROM personal_details WHERE
personal_details.overall_rating = page;");


$spforteam = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost` PROCEDURE
`SearchTeam`(IN `page` VARCHAR(30)) NOT DETERMINISTIC CONTAINS SQL SQL
SECURITY DEFINER SELECT
pd.player_name,pd.overall_rating,pd.age,pd.nationality,od.club FROM personal_details
pd,other_details od WHERE od.club = page AND pd.player_id = od.player_id ORDER BY
pd.player_id;");
```

```
$spforname = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost` PROCEDURE
`SearchName`(IN `page` VARCHAR(30)) NOT DETERMINISTIC CONTAINS SQL SQL
SECURITY DEFINER SELECT * FROM personal_details WHERE player_name = page");
```

```
$spforplayerid = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost`
PROCEDURE `Searchplayerid`(IN `page` INT(11)) NOT DETERMINISTIC CONTAINS
SQL SQL SECURITY DEFINER SELECT * FROM personal_details WHERE player_id =
page;");
```

```
$spforposition = mysqli_query($conn, "CREATE DEFINER=`root`@`localhost`
PROCEDURE `SearchPosition`(IN `page` VARCHAR(11)) NOT DETERMINISTIC
CONTAINS SQL SQL SECURITY DEFINER SELECT pd.player_name, pd.overall_rating,
od.preferred_position, p.gk, p.df, p.cm, p.fr FROM personal_details pd, other_details od,
position p WHERE od.preferred_position = page AND p.player_id = od.player_id AND
p.player_id = pd.player_id GROUP BY pd.player_id;");
```

4.2 Inserting new records:

The insert page allows users to select the table to insert values into [Fig 5.4]. It then asks users to input required data columns for the particular table and upon successful insertion, a new page is displayed [Fig 5.5] with appropriate message. Upon failure, another page is displayed with appropriate message and a possible solution.

Given below is the code snippet of the insert page which is execute in the application using PHP and MySQL.

```
$sql = "INSERT INTO $dbname.personal_details (player_id, player_name,
age,overall_rating,nationality) VALUES
('$spid','$spname','$spage','$spoverallrating','$spnationality')";
```

```
if ($conn->query($sql) === TRUE) {
```

(Inside into_other_details.php)

```
<?php
```

```
$sql = "INSERT INTO $dbname.other_details (player_id, club, preferred_position) VALUES
('$spid','$spclub','$sppreferredposition')";
```

4.3 Updating existing records:

The update page allows users to look at a selected table and edit the table live on frontend. The user first selects the table he/she wants to modify [Fig 5.6]. Upon selection, a new page is created with respective table. This table consists of editable rows that can be modified and result can be seen real time [Fig 5.7].

Given below is a code snippet of implementation of update page. It is created using PHP, MySQL and JavaScript. Use of JavaScript makes the UI more flexible by providing real time data interaction.

```
if ($input['action'] === 'edit')

{
        $sql = "UPDATE personal_details SET player_name ='" .
  $input['player_name'] . "',age ='" . $input['age'] . "',overall_rating ='" .
  $input['overall_rating'] . "', nationality='" .
$input['nationality'] . "'" ." WHERE player_id='" . $input['player_id'] . "'";

    mysqli_query($con,$sql);
}
$choices = $_POST["choices-single-defaul"];

if($choices=='PERSONAL

DETAILS'){ header("Location:test_personal_details/inline-

table-edit.php");

}
else if($choices=='PLAYER

   EARNINGS'){ header("Location:test_player_salary/inline

   table-edit.php");}else if($choices=='PLAYER

   POSITION'){ header("Location:test_player_position/inline

   -table-edit.php");

}else if($choices=='PLAYER

CLUB'){ header("Location:test_player_club/inline-table

edit.php");
```

4.4 Deleting existing records:

The delete page allows users to delete data based on categories like age, nationality, player ID, name and overall rating [Fig 5.8]. Choosing from different categories provides better user-database interface. Upon deletion, a new page is displayed with appropriate message [Fig 5.9] and upon failing, a new page with error information and possible solutions.

Given below is a code snippet of implementation of delete page. It is created using PHP and MySQL.

```
$sql = "DELETE FROM personal_details WHERE player_name=\"$spname\"";
```

```
$sql = "DELETE FROM personal_details WHERE age=\"$spages\"";
```

```
$sql = "DELETE FROM personal_details WHERE nationality=\"$spnationality\"";
```

```
$sql = "DELETE FROM personal_details WHERE overall_rating=\"$spoverallrating\"";
```

```
$sql = "DELETE FROM personal_details WHERE player_id=\"$spid\"";
```

4.5 Triggers:

The trigger page allows users to look at the database log of operations such as delete, update and insert.

Use: To improve data integrity, trigger can be used. When an action is performed on data, it is possible to check if the manipulation of the data concurs with the underlying business rules, and thus avoids erroneous entries in a table.

Given below is a code snippet of implementation of trigger page. It is created using PHP, MySQL and HTML. Use of HTML makes the UI show database logs in a structured manner.

echo " <h1>INSERT TRIGGERS</h1>";

$sql = "SELECT * FROM insert_logs ORDER BY id";

echo " <h1>UPDATE TRIGGERS</h1>";

$sql = "SELECT * FROM update_logs ORDER BY id";

echo " <h1>DELETE TRIGGERS</h1>";

$sql = "SELECT * FROM delete_logs ORDER BY id";

Chapter 5

SCREENSHOTS

5.1 Stored procedures:
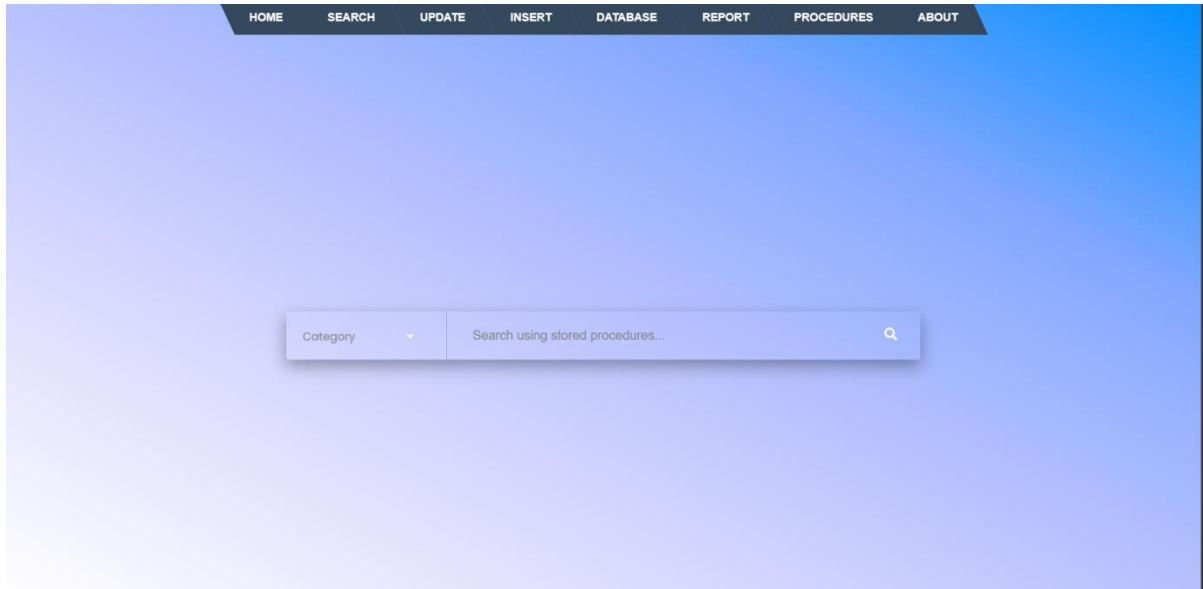
Fig 5.1 Search input for stored procedures.

Fig 5.2 Search result using stored procedure from frontend.

Fig 5.3 Search result using stored procedure from backend.

5.2 Inserting new records:

Fig 5.4 Selecting table for insertion
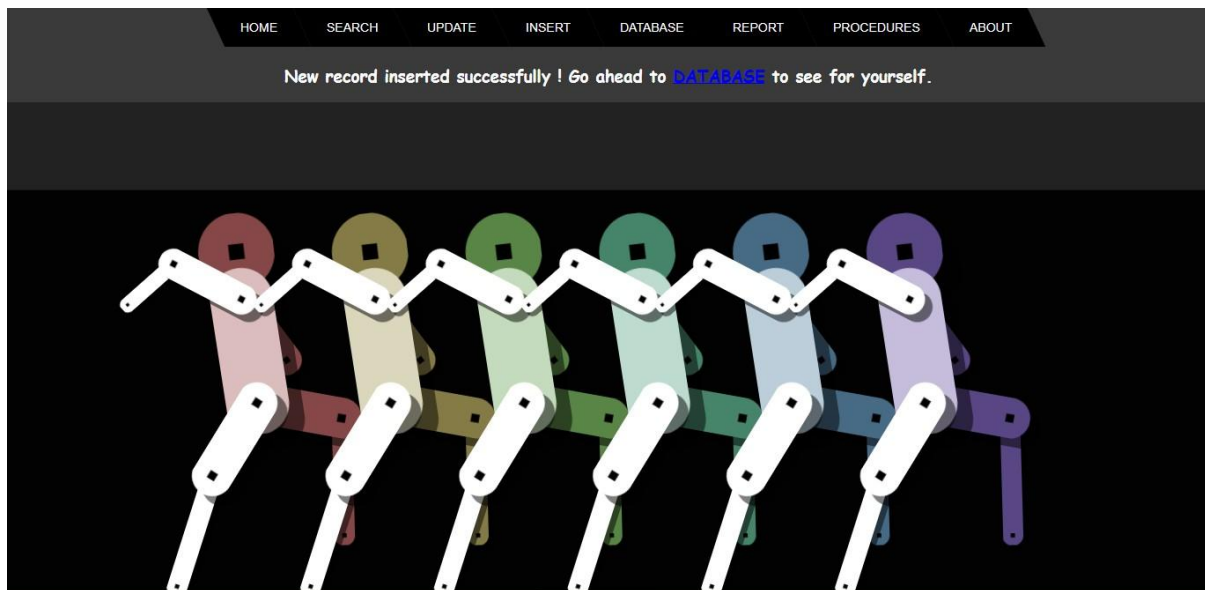
Fig 5.5 Successful insert instance page

5.3 Update existing records:
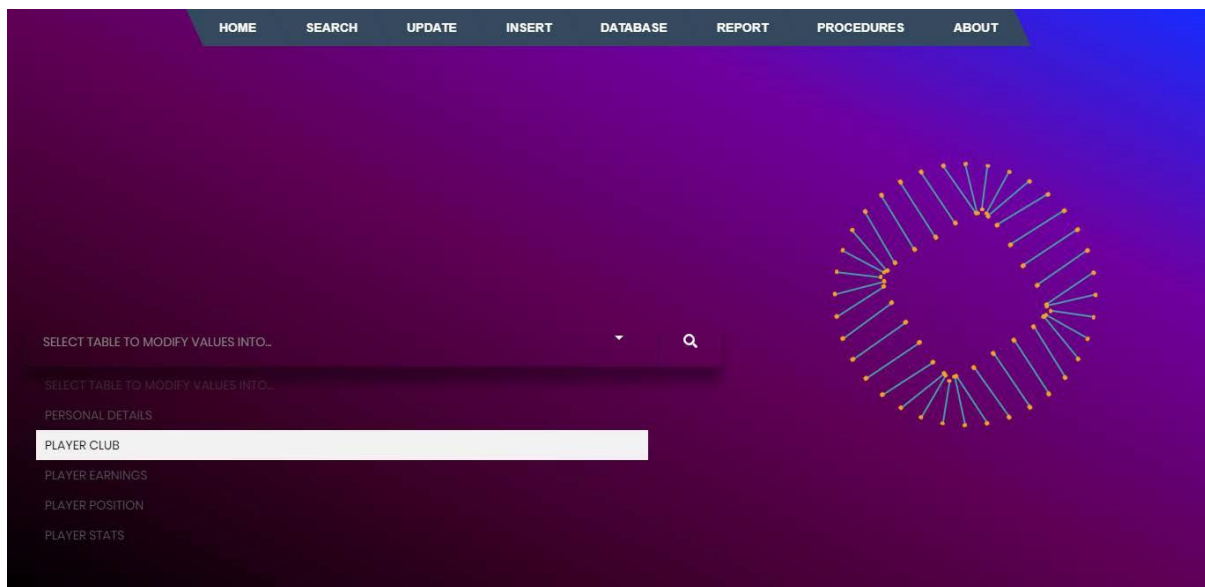
Fig 5.6 Selecting table to modify record

Fig 5.7 Modifying records in real time

5.4 Deleting records:
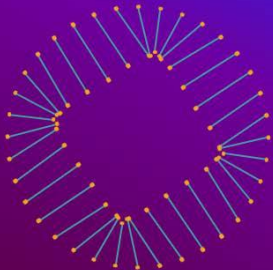
Fig 5.8 Selecting categories for deleting record
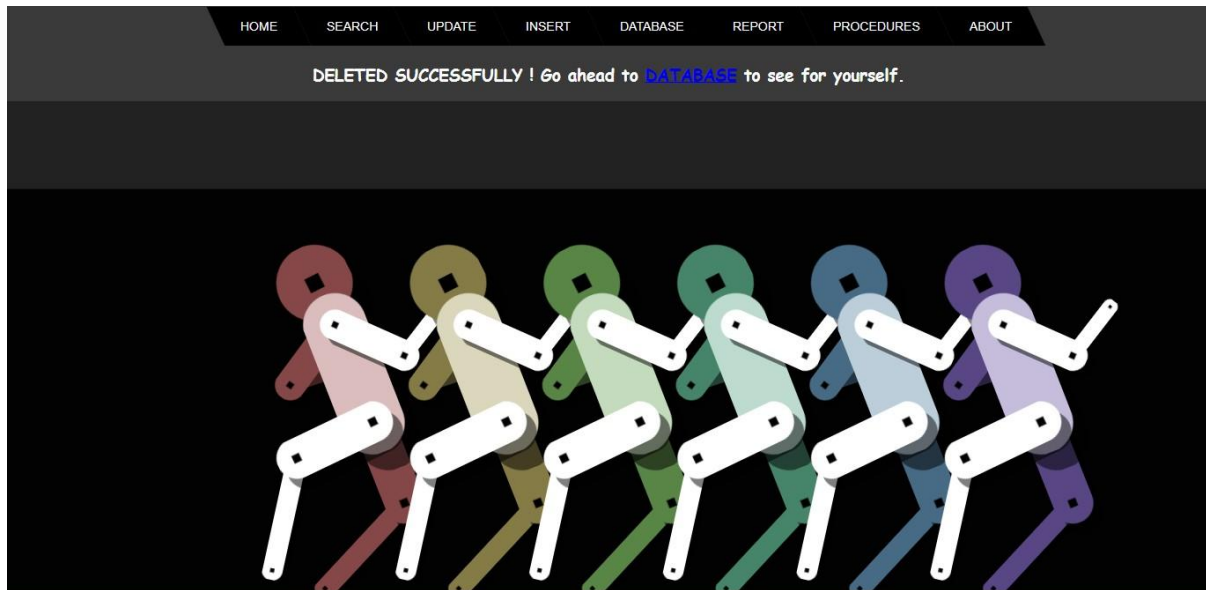
Fig 5.9 Successful deletion instance

5.5 Triggers:

Fig 5.10 Delete triggers table

Fig 5.11 Insert triggers table

Fig 5.12 Update triggers table

5.6 Database tables:

Fig 5.13 Player club table

Fig 5.14 Player salary table

PLAYER POSITIONS

PLAYER ID	GOALKEEPER	DEFENDER	CENTRAL-MID	FORWARD
1011	40	58	85	94
1034	40	56	82	91
1234	75	89	94	99
6969	57	60	69	21
2121	40	56	84	92

Fig 5.15 Player position table

PLAYER STATS

PLAYER ID	ACCELERATION	BALANCE	BALL CONTROL	CROSSING	CURVE	DRIBBLING	FINISHING	GK KICKING	GK POSITIONING	PENALTIES	SHORT PASS	STAMINA	STRENGTH
1011	91	90	92	89	88	93	94	40	40	95	88	90	78
1034	92	87	89	91	86	90	94	40	40	94	84	93	92
1234	95	90	80	91	95	90	95	75	75	99	98	99	99
6969	69	99	85	67	65	82	99	40	99	99	64	95	4
2121	92	90	83	86	87	90	88	40	40	86	78	78	69

Fig 5.16 Player stats table

PERSONAL DETAILS

PLAYER ID	NAME	AGE	OVERALL RATING	NATIONALITY
1011	LIONEL MESSI	35	94	ARGENTINA
1034	CRISTIANO RONALDO	37	91	PORTUGAL
1234	INAO SINGPHO	21	99	INDIA
6969	SHUBHAM PARMEKAR	19	69	INDIA
2121	NEYMAR JR.	30	92	BRAZIL

Fig 5.17 Player's table

CONCLUSION

This project is developed to nurture the needs of a user/scouting agent to monitor players and inspect their technicalities from every aspect on a football field. This is a computerized version of player management system which will benefit the players as well as the staff of a club.

In this entire process one can search player details, add new skilled players, Update ratings and view all the player statistics. The software takes care data and carefully stores all the player information. It provides security and encapsulation by the use of stored procedures.

FUTURE SCOPE

There is a future scope of this project is to help managers and club staffs to get out the best youth talent across the world. Features like predicting players rating based on their current performances and training sessions helps team staffs to judge players according to the club's needs.

BIBLIOGRAPHY

[1] Database System Model, Languages, Design and Application Programming, Ramez Elmasri and ShamKant B. Navathe, 7th edition, 2017, Pearson.

[2] Database Management System, Ramakrishnan, and Gehrke, 3rd edition, 2014, Mc-GrawHill.

Websites Referred:

- https://stackoverflow.com/search?q=insert+table
- https://stackoverflow.com/search?q=view+table
- https://stackoverflow.com/search?q=search+from+table
- www.quora.com
- https://www.codeproject.com/search.aspx?q=php+code+for+radio+buttons&x=0&y=0&sbo=qa
- https://www.codeproject.com/Articles/38808/Overview-of-SQL-Server-database-Triggers
- https://www.techrepublic.com/article/determine-when-to-use-stored-procedures-vs-sql-in-the-code/
- https://codepen.io/
- https://www.uplabs.com/
- https://www.ibm.com/support/knowledgecenter/en/SS6NHC/com.ibm.swg.im.dashdb.sql.ref.doc/doc/c0004100.html
- https://www.w3schools.com/html/default.asp
- https://www.w3schools.com/css/default.asp
- https://www.w3schools.com/php/default.asp
- https://www.ibm.com/support/knowledgecenter/en/SSEPEK_10.0.0/apsg/src/tpc/db2z_storedprocedure.html