

## **ABSTRACT**

This project is entitled as “WEB BASED SPORTS PORTAL” .The goal of the sports management system is to offer a way to oversee the operations of numerous sports simultaneously. When using the automated system instead of doing manual paperwork, users will save more time. All of the service tasks will be handled quickly by the system. It is simpler to store data. Any report can be checked by it at any moment. There is less manual labor and paperwork. The technology is simple to use and intuitive. The student can use his or her username and password to register and log in to the system for this project. Following the registration process, the user can examine the admin-updated sports (games) details. The student can use the specific sport online for this project. All student requests are visible to the administrator on this website. It streamlines the manual search, gathering, and joining procedure for sports information.

# **1. PROJECT DESCRIPTION**

## **1.1 INTRODUCTION**

In today's cutthroat world, sports are viewed as one of the lucrative markets. The need for highly qualified professionals to oversee sport and related fields is being driven by the profession's increasing competitiveness. The demand for skilled workers, particularly from schools, colleges, and universities, has increased due to the rise in competitive sport participation and sport-related enterprises. The impact and influence on sports management, traditional sports management concepts, organization structure, and management methods are becoming increasingly difficult due to the ongoing development of sports information and the ability of information technology to fundamentally alter sports management and systems. Application of information technology throughout the entire sports management process, the requirements of sports management to implement and perfect the use of information technology, and the mutual promotion of information technology and sports management.

Sports management information systems are a focus of research on sports information and are a crucial component of information building when it comes to the interaction of information technology with sports management. It is the regularization of particular business processes in sports management through the specific application and execution of sports management information system processes. The Online Portal for College Sports aims to provide a system that allows many sports to be coordinated simultaneously. It also manages the selection of students at the state and collegiate levels. Users of the automated system will save time compared to manual paper. The system will handle all servicing-related duties in a timely manner. Data storage is easier. It is capable of checking any report at any time. Paperwork and manual labor are reduced. The technology is easy to use and intuitive.

## **1.2 EXISTING SYSTEM**

The current method is manual and requires users to keep records in order to store information such as application and sports-related data. Maintaining historical data is exceedingly challenging. The current system is a manual that is difficult to track and does not keep details with appropriate in charge method. To join the sports committee, students must get in touch with the sports coordinator. Students will not be able to access effective knowledge sources from this. Pupils are not given comprehensive sports information. Since managing all the information and details takes a lot of time, the current method seems inaccurate, making maintenance extremely difficult.

### **EXISTING SYSTEM DISADVANTAGES:**

- It's challenging to respond to the questions right away.
- To obtain the sports information, manual labor is required.
- Don't get a close-up look at how college athletics are run. Joining the sports committee is challenging.

## **1.3 PROPOSED SYSTEM**

College sports can be managed with the help of a web-based sports portal. This program was created to monitor the various athletic events that took place on campus. The website's administrator could add the sport's details. Students ought to sign up using this application. Only registered students may apply for sports. Students can view the sports offered by their college after registering. After that, people can choose the sport in which they are interested and submit a request to join the sports committee. The administrator can then see the student information that has been submitted for sports.

### **Benefits include:**

- Saves a great deal of time;
- Makes it simple to inform pupils about sports.
- Lessen the amount of manual labor required of students to join the committee.
- Lessen the administrative load of keeping track of sports information.

### **1.2.1 HARDWARE REQUIREMENTS**

**Processor** : Dual core processor 2.6.0 GHZ

**RAM** : 4GB

**Hard disk** : 320 GB

**Compact Disk** : 650 Mb

**Keyboard** : Standard keyboard

**Monitor** : 15 inch color monitor

### **1.2.2 SOFTWARE REQUIREMENTS**

**Operating system** : Windows OS

**Front End** : Python

**Back End** : MySQL SERVER

**IDLE** : Python 2.7 IDLE

## **2. LOGICAL DEVELOPMENT**

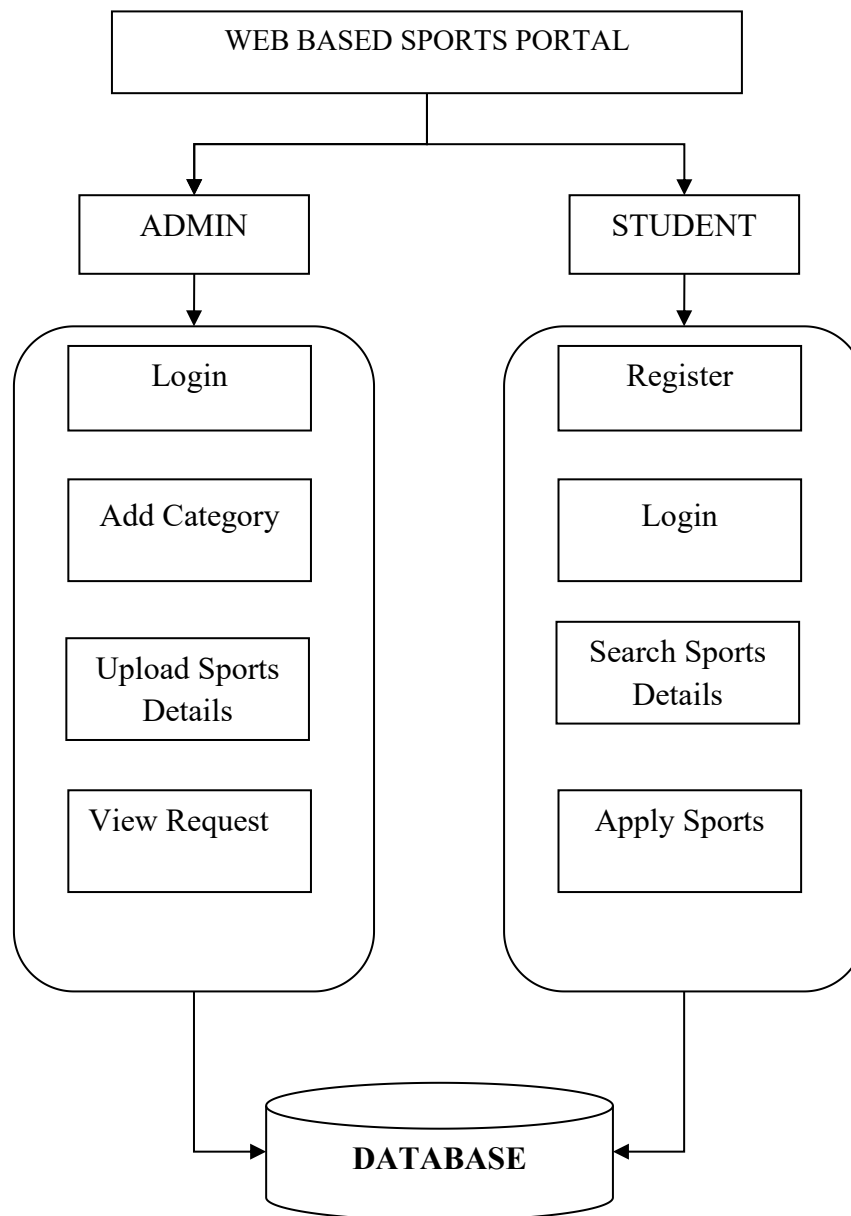
### **2.1 DESIGN OF ARCHITECTURE**

An allotted configuration of tangible components that offers a customer the design solution. The conceptual model that outlines a system's behavior, structure, and other aspects is called a systems architecture. A formal description and representation of a system that is structured to facilitate inference about its behaviors and structures is called an architecture description. System components, their outwardly evident characteristics, and the connections (such as behavior) among them can all be included in the system architecture. It can offer a blueprint from which systems and products can be created that will cooperate to carry out the system as a whole. Architecture description languages (ADLs) are a group of formalized languages that have been developed to describe system architecture.

**Systems architecture is defined differently by different organizations, including:**

- Product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
- Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behaviour.
- If recorded, it could contain details like a thorough inventory of the hardware, software, and networking capabilities that are currently in use; a breakdown of long-term goals and objectives for upcoming purchases; and a strategy for updating and/or replacing outdated hardware and software.

The combination of product life-cycle procedures and design architectures.


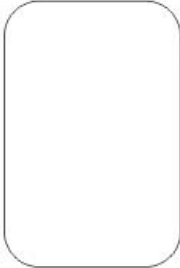




**Figure 2.1 Overall Architecture**

## 2.2 DIAGRAM OF DATA FLOW

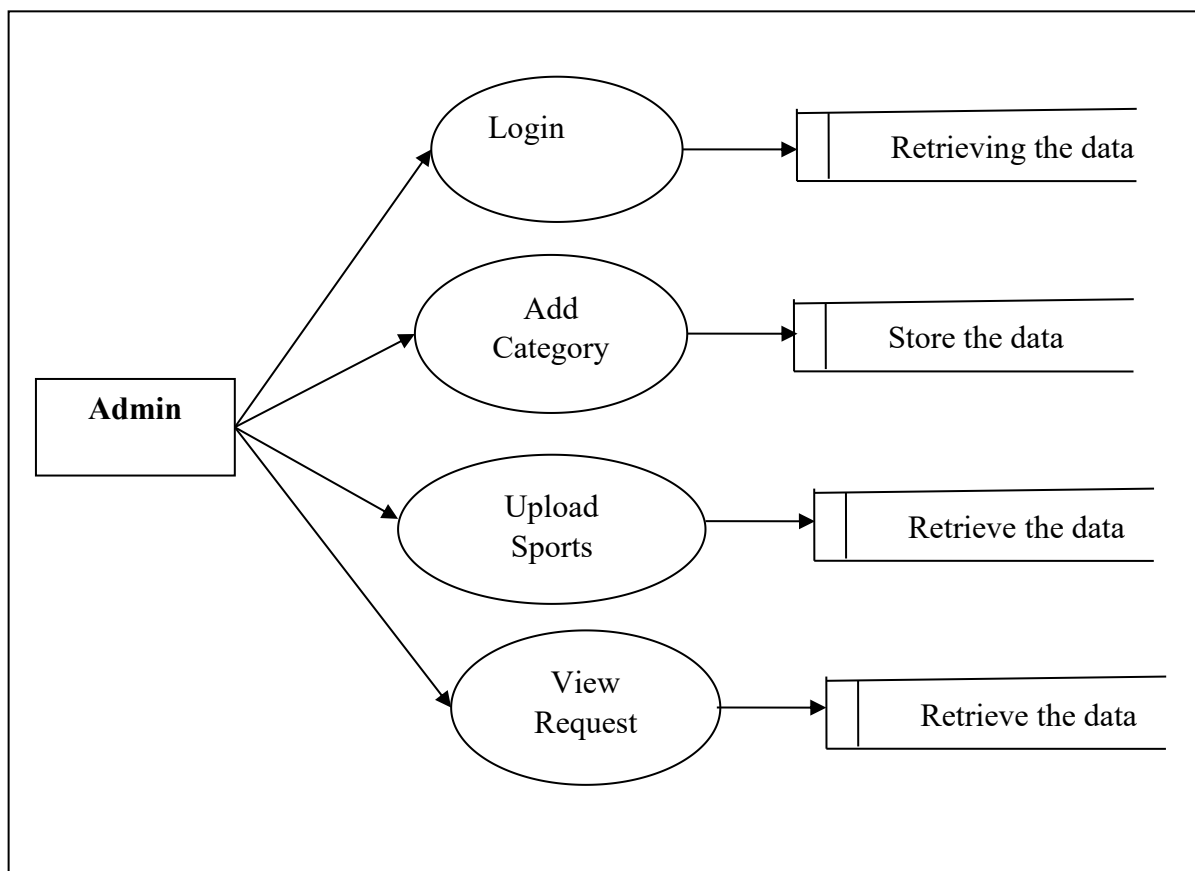
A two-dimensional diagram illustrates the system's data processing and transfer mechanisms. Each data source and how it interacts with other data sources to produce a common result are identified graphically. Finding external sources and outputs, figuring out how the inputs and outputs link to one another, and using graphics to illustrate the relationships and outcomes are all necessary steps for anyone attempting to create a data flow diagram. Business development and design teams can discover or enhance specific parts of data processing by using this kind of diagram to depict the process.

### Data flow Symbols:

Symbol	Description
	An <b>entity</b> . A source of data or a destination for data.
	A <b>process</b> or task that is performed by the system.
	A <b>data store</b> , a place where data is held between processes.
	A <b>data flow</b> .

## LEVEL 0

In the Level 0 DFD, the system is broken down into 'sub-systems' (processes), each of which handles one or more data flows to or from an external agent and collectively provide the system's overall functionality. It also illustrates the data flow between the different components of the system and indicates internal data stores that are necessary for the system to function.

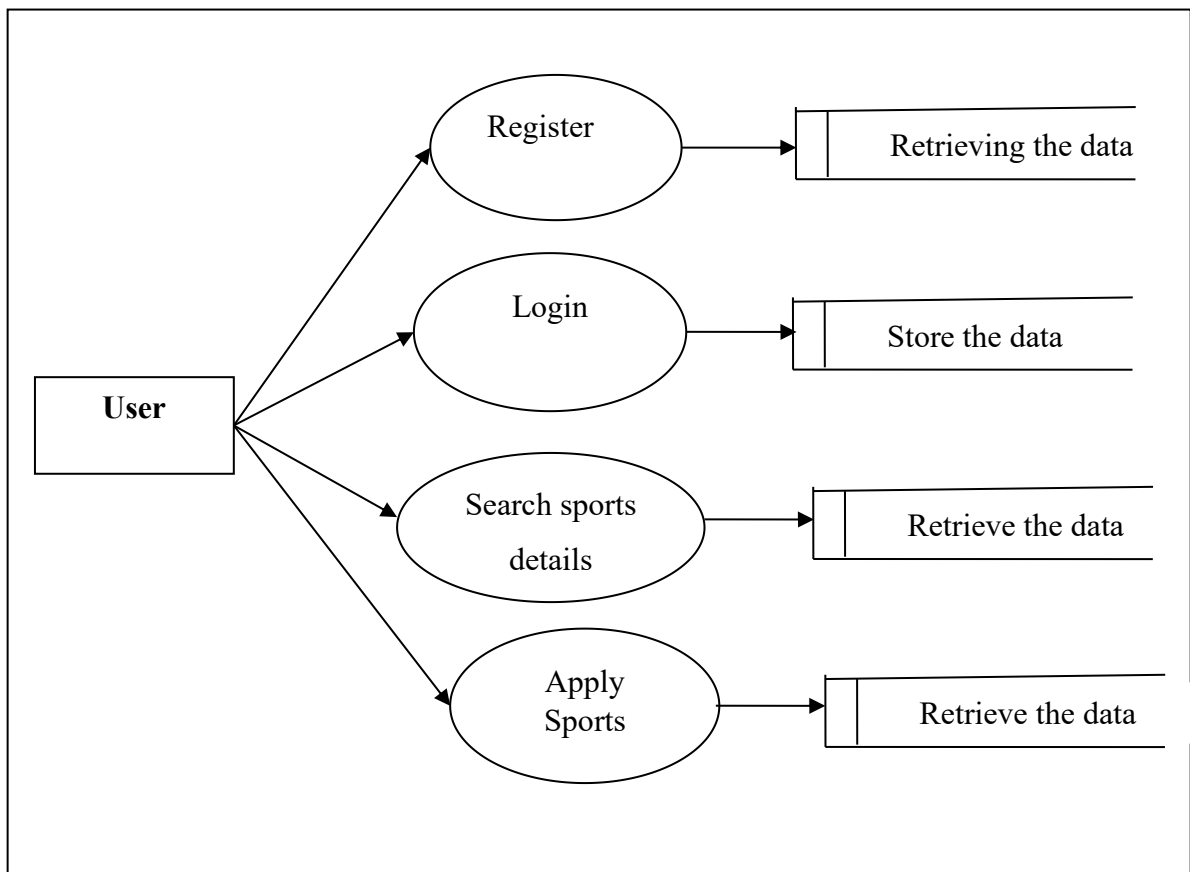


**Figure 2.2 Level 0 Diagram**



## LEVEL 1

The Level 1 Data Flow Diagram is the next step. This demonstrates the primary tasks that the system performs. Generally, two to seven functions were used to characterize the system, with two denoting a simple system and seven denoting a complex one. This allows us to maintain the model's manageability on paper or a screen.



**Figure 2.2 Level 1 Diagram**

### 3. DATABASE DESIGN

#### 3.1 TABLE DESIGN

Information is arranged into rows and columns in a table, a type of data structure. Data in a structured format can be displayed and stored using it. Databases, for instance, store information in tables so that certain rows can be rapidly accessed. Tables are frequently used on websites to show several rows of data. Spreadsheets store and present data in an organized manner, combining the two functions of a table.

Various tables, each with a distinct function, are frequently found in databases. For instance, distinct columns for clients, suppliers, and workers might be found in a corporation database. Depending on what information the table must hold, each table may have a unique collection of fields. Each entry (or record) in a database table is referred to as a row, and each field is referred to as a column. Data from a single column and row can be requested to retrieve a particular value from the table.

**Table Name: applytb**

Field	Type	Null	Default
aid	bigint(250)	Yes	NULL
Ename	varchar(250)	Yes	NULL
Place	varchar(250)	Yes	NULL
Name	varchar(250)	Yes	NULL
Gender	varchar(250)	Yes	NULL
Dob	varchar(250)	Yes	NULL
Mobile	varchar(250)	Yes	NULL
Email	varchar(250)	Yes	NULL
Adress	varchar(250)	Yes	NULL
Pincode	varchar(250)	Yes	NULL
Height	varchar(250)	Yes	NULL
Weight	varchar(250)	Yes	NULL

Profile	varchar(250)	Yes	NULL
Proof	varchar(250)	Yes	NULL
Status	varchar(250)	Yes	NULL
UserName	varchar(250)	Yes	NULL
cat	varchar(250)	Yes	NULL

**Table Name: categorytb**

Field	Type	Null	Default
<i>cid</i>	bigint(250)	Yes	NULL
category	varchar(250)	Yes	NULL

**Table Name: eventtb**

Field	Type	Null	Default
<i>eid</i>	bigint(250)	Yes	NULL
Ename	varchar(250)	Yes	NULL
Category	varchar(250)	Yes	NULL
Place	varchar(250)	Yes	NULL
Date	varchar(250)	Yes	NULL
Time	varchar(250)	Yes	NULL
Date1	varchar(250)	Yes	NULL
Time1	varchar(250)	Yes	NULL
Image	varchar(250)	Yes	NULL

**Table Name: regtb**

Field	Type	Null	Default
<i>uid</i>	bigint(250)	Yes	NULL
Name	varchar(250)	Yes	NULL
Gender	varchar(250)	Yes	NULL
Mobile	varchar(250)	Yes	NULL
Email	varchar(250)	Yes	NULL
Adress	varchar(250)	Yes	NULL

UserName	varchar(250)	Yes	NULL
Password	varchar(250)	Yes	NULL

**Table Name: winnerth**

Field	Type	Null	Default
<i><b>wid</b></i>	bigint(250)	Yes	NULL
PlayerName	varchar(250)	Yes	NULL
Category	varchar(250)	Yes	NULL
Pos	varchar(250)	Yes	NULL

### 3.2 DATA DICTIONARY

FIELD NAME	TYPE	DESCRIPTION	SAMPLE VALUES
Ename	varchar(250)	Specify the event name	Football
Place	varchar(250)	Specify the place	Trichy
Name	varchar(250)	Specify the name	Hari
Gender	varchar(250)	Specify the gender	Male
Dob	varchar(250)	Specify the date of birth	15.3.2002
Mobile	varchar(250)	Specify the mobile	9874120365
Email	varchar(250)	Specify the email id	hari@gmail.com
Adress	varchar(250)	Specify the address	chennai
Pincode	varchar(250)	Specify the pincode	620001
Height	varchar(250)	Specify the height	152.2
Weight	varchar(250)	Specify the weight	65
Status	varchar(250)	Specify the status	Booked
UserName	varchar(250)	Specify the user name	hari
cat	varchar(250)	Specify the category	sports
Date	varchar(250)	Specify the date	25.4.2025
Time	varchar(250)	Specify the time	11.00AM
Image	varchar(250)	Specify the image	Image.jpg
Password	varchar(250)	Specify the password	*****

### 3.3 RELATIONSHIP DIAGRAM

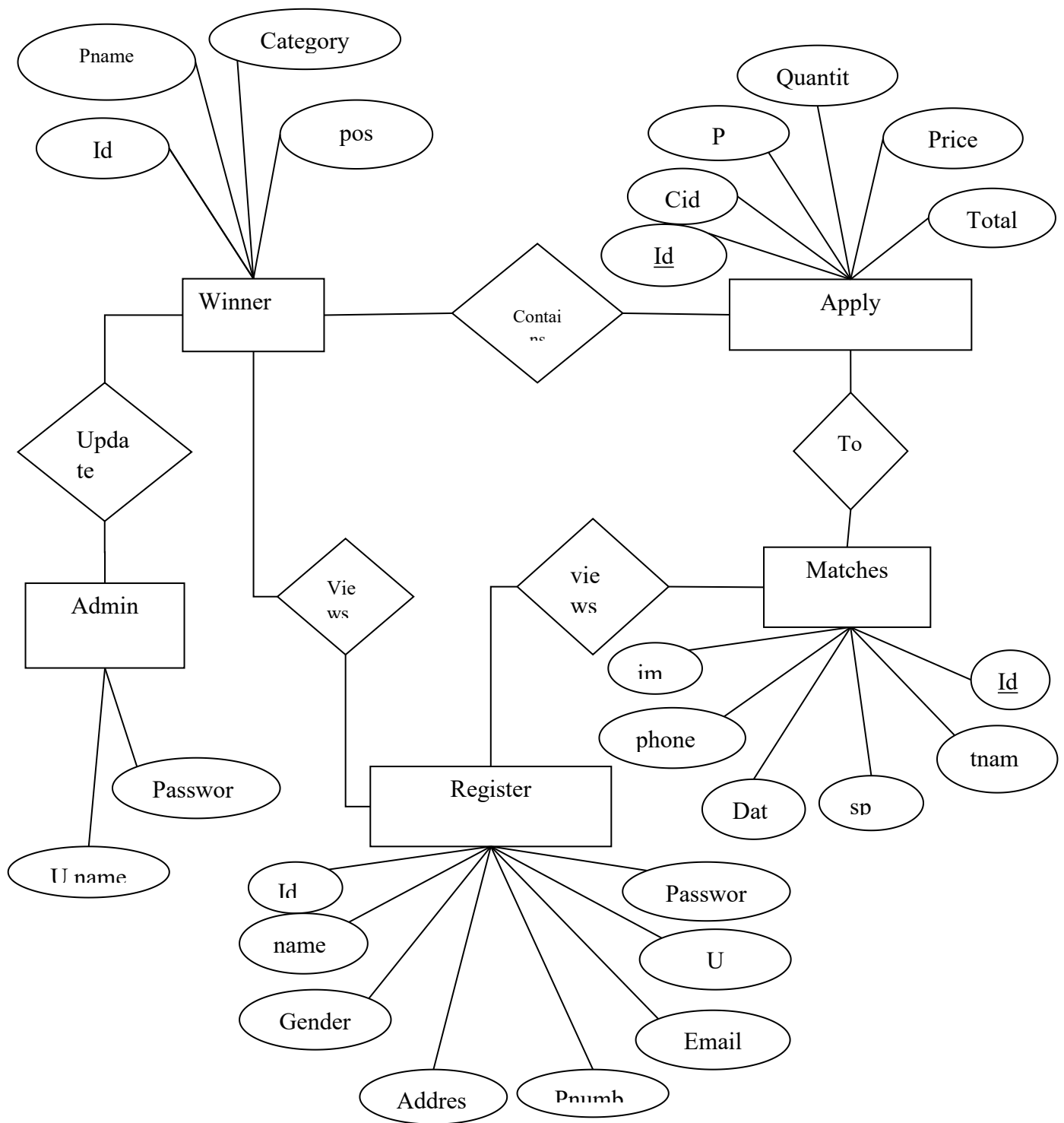


Figure 3.3 relationship diagram

## **4. PROGRAM DESIGN**

### **4.1 MODULES**

#### **Admin**

- Login
- Add Category
- Upload sports Details
- View request

#### **Student**

- Register
- Login
- Search sports details
- Apply sports

### **4.2 MODULE DESCRIPTION**

#### **4.2.1 Administrator:**

- **Log in**

In this module, the administrator enters their username and password to access the system. The administrator may be in charge of keeping all information on a single page.

- **Include a Category**

The administrator enters the tournament's name, sport name, location, eligibility, date, and player count in this module.

- **Upload the sports information.**

Upload sports information in this module, including the sport's name, player information, student information, and department information. View request

- **View request**

The administrator can view the student request details in this module, including the student's name, sports name, and date.

#### **4.2.2 Student**

- **Register**

Students provide their information in this module, including their name, gender, date of birth, phone number, email address, department, year, and registration number.

- **Login**

Students that successfully register log in to the system to examine all of their personal, athletic, and other information.

- **Search sports details**

Students can search for sports that the administrator has included in this module. Sports information is displayed according to the time and date.

- **Apply sports**

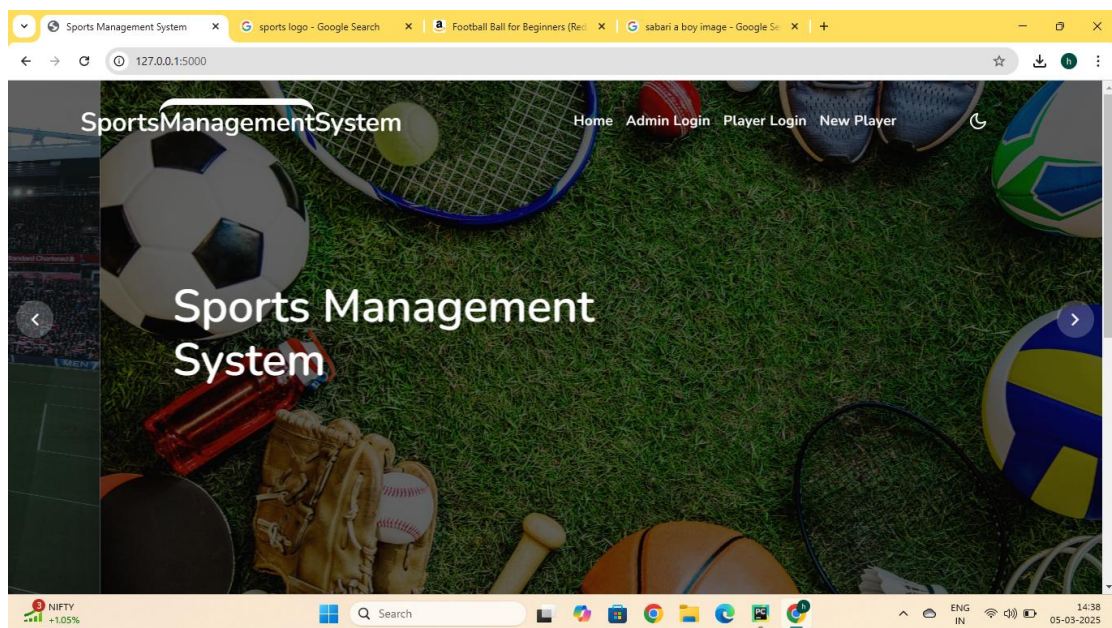
Students use sports in this module according to their needs. The admin page receives these student request details.



## 5. TESTING

### 5.1 SYSTEM TESTING

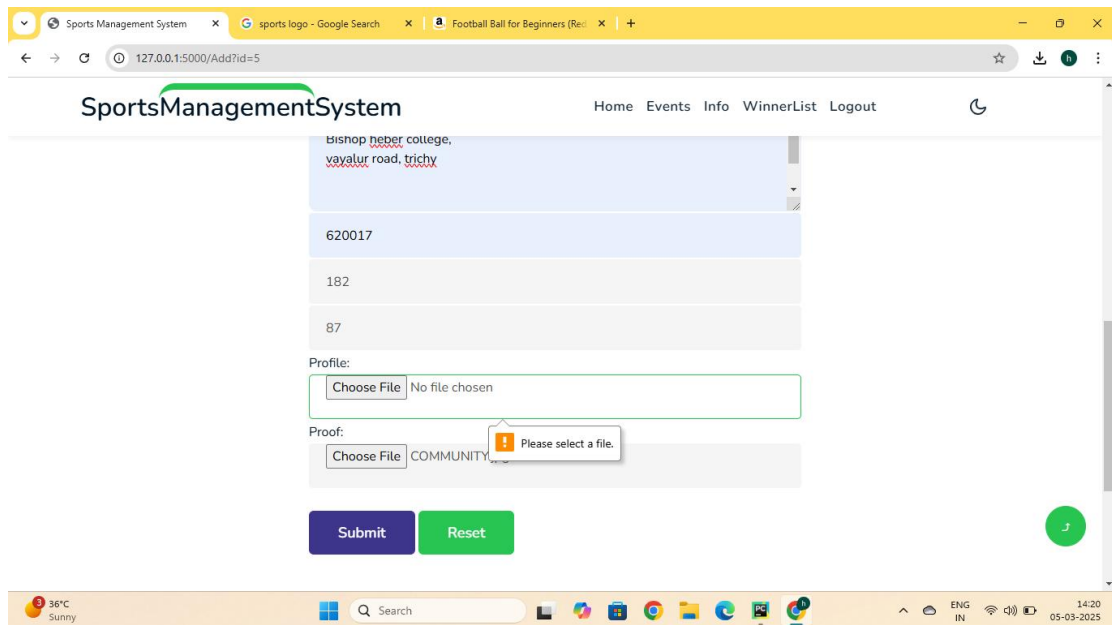
System testing is the process of testing a software product that is completely integrated and comprehensive. This type of testing, known as "black-box" testing, is carried out by the testing team and does not require knowledge of the inner workings of the code. Verification that the supplied product satisfies the requirements given in the requirement document is the last test. Both functional and non-functional criteria should be examined.



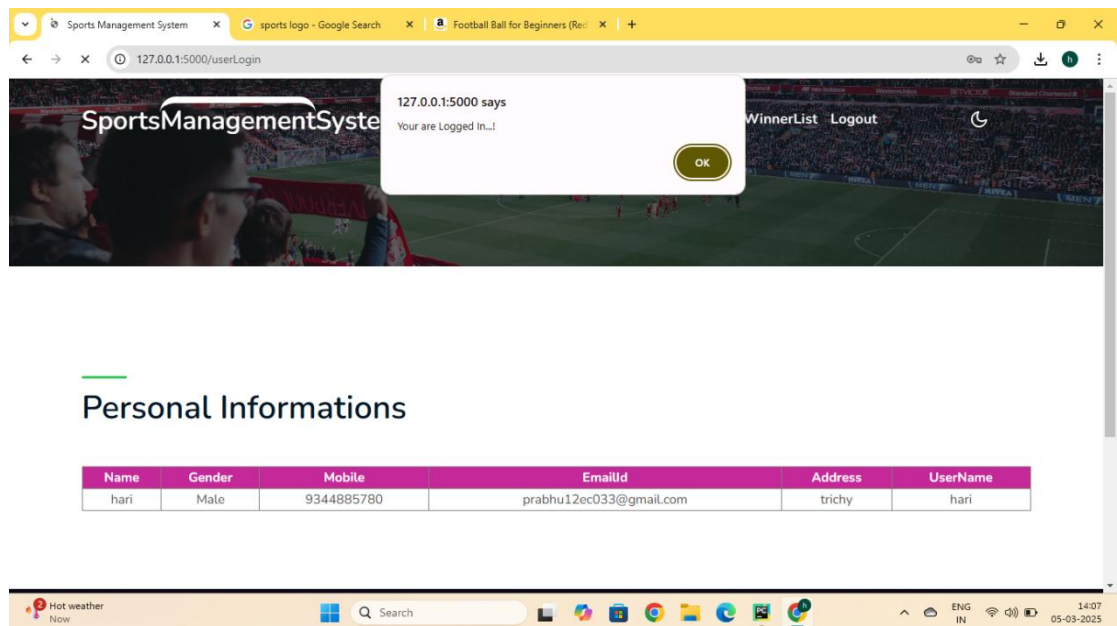
**Figure 5.1 Sytem Testing in Home Page**

### 5.2 UNIT TESTING

The unit test is the initial test in the development process. Typically, the source code is separated into modules, which are further subdivided into units, which are smaller units. These units behave in particular ways. Unit tests are the tests performed on these code units. The language used to design the project affects the unit test. Unit tests guarantee that every distinct project path operates precisely in accordance with the stated specifications and has inputs and expected outcomes that are well-defined.



**Figure 5.2.1 Unit Testing in Personal Information**

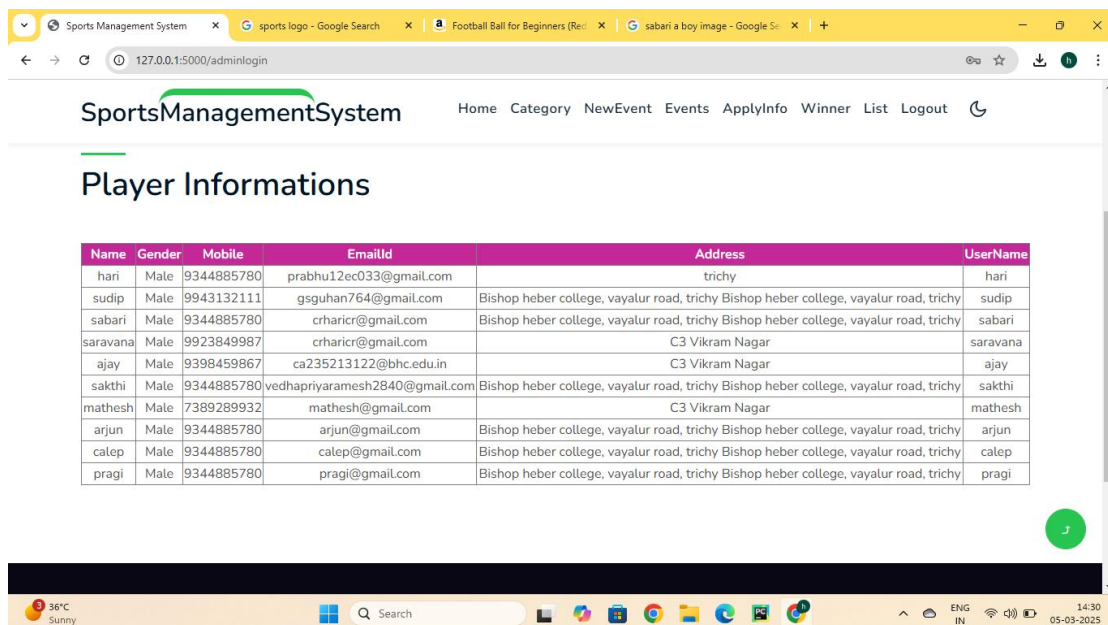


**Figure 5.2.2 Unit Testing**

**5.3 INTEGRATION TESTING** Modules are merged and tested collectively in integration testing. Code modules, standalone apps, source and destination apps over a network, etc. are examples of modules. Integration testing comes before system testing and after unit testing, testing following the completion of the product's code. In the hopes that people will purchase the finished product when it is published, betas are sometimes disseminated extensively or even to the general public.



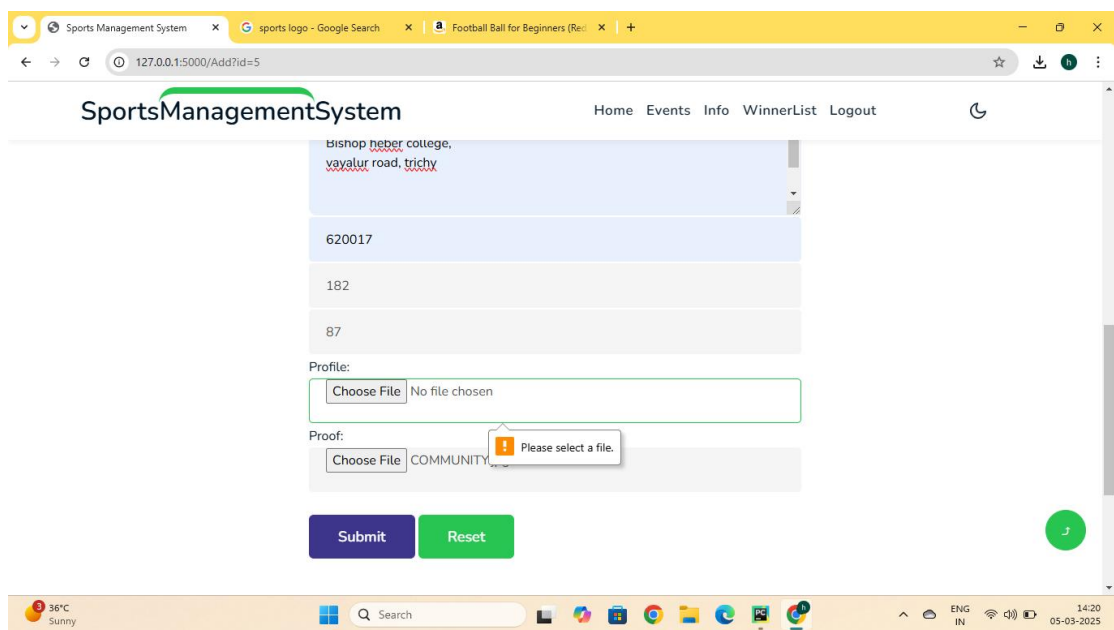
**Figured 5.1.1 Integration Testing**



**Figured 5.1.2 Integration Testing in player information**

## 5.4 VALIDATION

The process of assessing the finished product to see whether the software satisfies the needs and expectations of the client is known as validation. It is a dynamic process for testing and validating the real product. The process of validating a system involves assessing if it fits the demands of users, the organization's objectives, and the requirements. Validation aids in creating the ideal product to meet the demands and requirements of the client.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/Add?id=5'. The page title is 'SportsManagementSystem'. The navigation bar includes links for 'Home', 'Events', 'Info', 'WinnerList', and 'Logout'. The main content area contains a form with the following fields and values:

- Address:** Bishop heber college, vayalur road, trichy (with red squiggly lines under 'vayalur' and 'trichy')
- Pincode:** 620017
- Phone:** 182
- Mobile:** 87
- Profile:** A file upload button labeled 'Choose File' with the text 'No file chosen'.
- Proof:** A file upload button labeled 'Choose File' with the text 'COMMUNITY'.

At the bottom of the form are two buttons: 'Submit' (purple) and 'Reset' (green). A green circular button with a right arrow is located on the right side of the form. A red error message box is visible near the 'Proof' field, stating 'Please select a file.' The browser's taskbar at the bottom shows the system clock as 14:20 on 05-03-2025, with a temperature of 36°C and 'Sunny' weather.

**Figure 5.4 validation testing**

## **6. CONCLUSION**

The goal of the web-based sports portal is to offer a way to manage multiple sports' activities simultaneously. With the automated method, the learner will spend less time than with manual paper work. All of the service tasks will be handled quickly by the system. It is simpler to store data. Any report can be checked by it at any moment. The goal of this request is to give specifics on how this system handles multiple sports at once. The serving action will be provided by this system quickly and easily. Because it is an automated system, it will take less time. In addition to being gratifying, web-based sports portals assist programmers in efficiently organizing sports events and listings. It will always be able to check anything that has to do with sports. With this technique, less manual labor and paper work is required.

## **7. REFERENCES**

### **BOOK REFERENCES**

1. Heinold, Brian. "A practical introduction to Python programming." (2021).
2. Kneusel, Ronald T. Practical deep learning: A Python-based introduction. No Starch Press, 2021.
3. Dhruv, Akshit J., Reema Patel, and Nishant Doshi. "Python: the most advanced programming language for computer science applications." Science and Technology Publications, Lda (2021): 292-299.
4. Sundnes, Joakim. Introduction to scientific programming with Python. Springer Nature, 2020.
5. Hill, Christian. Learning scientific programming with Python. Cambridge University Press, 2020.

### **WEBSITE REFERENCES**

1. <https://medium.com/javarevisited/10-free-python-tutorials-and-courses-from-google-microsoft-and-coursera-for-beginners-96b9ad20b4e6>
2. <https://www.bestcolleges.com/bootcamps/guides/learn-python-free/>
3. <https://www.programiz.com/python-programming>
4. <https://realpython.com/>
5. <https://www.codecademy.com/learn/learn-python>

## 8. APPENDIX

### 8.1 SOURCE CODE

```
from flask import Flask, render_template, flash, request, session, send_file
from datetime import datetime
import mysql.connector
app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = 'sdssdsds15154545'
@app.route("/")
def home():
    return render_template('index.html')
@app.route("/AdminLogin")
def AdminLogin():
    return render_template('AdminLogin.html')
@app.route("/UserLogin")
def UserLogin():
    return render_template('UserLogin.html')
@app.route("/NewUser")
def NewUser():
    return render_template('NewUser.html')
@app.route("/AdminHome")
def AdminHome():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l1sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM regtb ")
    data = cur.fetchall()
    return render_template('AdminHome.html',data=data)
@app.route("/UserHome")
def UserHome():
    uname = session['uname']
    conn = mysql.connector.connect(user='root', password="", host='localhost',
```

```

database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM regtb where UserName='"+ uname +"' ")
    data = cur.fetchall()
    return render_template('UserHome.html',data=data)
@app.route("/adminlogin", methods=['GET', 'POST'])
def adminlogin():
    if request.method == 'POST':
        if request.form['uname'] == 'admin' and request.form['pas'] == 'admin':
            conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
            cur = conn.cursor()
            cur.execute("SELECT * FROM regtb ")
            data = cur.fetchall()
            flash("Your are Logged In...!")
            return render_template('AdminHome.html', data=data)

        else:
            flash("Username or Password is wrong")
            return render_template('AdminLogin.html')
@app.route("/newuser", methods=['GET', 'POST'])
def newuser():
    if request.method == 'POST':
        name = request.form['name']
        gender= request.form['gender']
        mobile = request.form['mobile']
        email = request.form['email']
        address = request.form['address']
        uname = request.form['uname']
        pas = request.form['pas']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where UserName='"+ uname +"' and

```



```

Password=" + pas + "")
    data = cursor.fetchone()
    if data is None:
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO regtb VALUES ('" + name + "','" + gender + "','" +
mobile + "','" + email + "','" + address + "','" + uname + "','" + pas + "')"
        conn.commit()
        conn.close()
        flash('New User Registered successfully')
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cur = conn.cursor()
        cur.execute("SELECT * FROM regtb ")
        data = cur.fetchall()
        return render_template('UserLogin.html', data=data)
    else:
        flash('Already registered')
        return render_template('NewUser.html')
@app.route("/userLogin", methods=['GET', 'POST'])
def userlogin():
    if request.method == 'POST':
        uname = request.form['uname']
        pas = request.form['pas']
        session['uname'] = request.form['uname']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute("SELECT * from regtb where UserName=" + uname + " and
Password=" + pas + "")
        data = cursor.fetchone()
        if data is None:

```

```

        flash("RegisterNo or Name is wrong...!")
        return render_template('UserLogin.html')
    else:
        session['mob'] = data[2]
        session['add'] = data[4]
        session['cname'] = data[2]
        session['department'] = data[7]
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cur = conn.cursor()
        cur.execute("SELECT * FROM regtb where UserName='" + uname + "' and
Password='" + pas + "'")
        data = cur.fetchall()
        flash("Your are Logged In...!")
        return render_template('UserHome.html', data=data)
@app.route("/AddCategory")
def AddCategory():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM categorytb ")
    data = cur.fetchall()
    return render_template('AddCategory.html', data=data)
    return render_template('AddCategory.html')
@app.route("/newcategory", methods=['GET', 'POST'])
def newcategory():
    if request.method == 'POST':
        cat = request.form['cat']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute("SELECT * from categorytb where category='" + cat + "'")
        data = cursor.fetchone()
        if data is None:

```

```

        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO categorytb VALUES ('" + cat + "')"
        )
        conn.commit()
        conn.close()
        flash('New Category Registered successfully')
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cur = conn.cursor()
        cur.execute("SELECT * FROM categorytb ")
        data = cur.fetchall()
        return render_template('AddCategory.html', data=data)
    else:
        flash('Already Added')
        return render_template('AddCategory.html')

@app.route("/ARemove")
def ARemove():
    id = request.args.get('id')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cursor = conn.cursor()
    cursor.execute(
        "delete from categorytb where cid='" + id + "'"
    )
    conn.commit()
    conn.close()
    flash('Category Removed Successfully!')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM categorytb ")
    data = cur.fetchall()
    return render_template('AddCategory.html', data=data)

```

```

@app.route("/NewEvent")
def NewEvent():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l1sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct category FROM categorytb")
    data = cur.fetchall()
    return render_template('NewEvent.html',data=data)
@app.route("/newevent", methods=['GET', 'POST'])
def newevent():
    if request.method == 'POST':
        ename = request.form['ename']
        cat = request.form['cat']
        place = request.form['place']
        date = request.form['date']
        time = request.form['time']
        date1 = request.form['date1']
        time1 = request.form['time1']
        file = request.files['file']
        file.save("static/uploads/" + file.filename)

        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l1sportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO eventtb VALUES ('" + ename + "','" + cat + "','" + place +
            "','" + date + "','" + time + "','"
            + date1 + "','" + time1 + "','" + file.filename + "')"
        )
        conn.commit()
        conn.close()
        flash('New Event Registered successfully')
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l1sportsmanagement')
        cur = conn.cursor()

```

```

        cur.execute("SELECT * FROM eventtb ")
        data = cur.fetchall()
        return render_template('AViewEvent.html', data=data)
@app.route("/AviewEvent")
def AviewEvent():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM eventtb ")
    data = cur.fetchall()
    return render_template('AviewEvent.html',data=data)
@app.route("/AERemove")
def AERemove():
    id = request.args.get('id')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cursor = conn.cursor()
    cursor.execute(
        "delete from eventtb where cid='" + id + "'"")
    conn.commit()
    conn.close()
    flash('eventtb Removed Successfully!')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM eventtb ")
    data = cur.fetchall()
    return render_template('AviewEvent.html', data=data)
@app.route("/UviewEvent")
def UviewEvent():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM eventtb ")

```

```

        data = cur.fetchall()
        return render_template('UviewEvent.html',data=data)
@app.route("/Add")
def Add():
    id = request.args.get('id')
    session['pid'] = id
    uname = session['uname']
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM eventtb where eid='" + id + "' ")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM regtb where UserName='" + uname + "'")
    data1 = cursor.fetchone()
    if data1:
        name = data1[1]
        gender = data1[2]
        mobile = data1[3]
        email = data1[4]
        address = data1[5]
    else:
        return 'No Record Found!'
    return render_template('UApply.html',
data=data,name=name,gender=gender,mobile=mobile,email=email,address=address)
@app.route("/uapply", methods=['GET', 'POST'])
def uapply():
    if request.method == 'POST':
        pid = session['pid']
        name = request.form['name']
        gender = request.form['gender']
        dob = request.form['dob']

```

```

mobile = request.form['mobile']
email = request.form['email']
address = request.form['address']
pin = request.form['pin']
height = request.form['height']
weight = request.form['weight']
file = request.files['profile']
file.save("static/uploads/" + file.filename)
file1 = request.files['proof']
file1.save("static/uploads/" + file1.filename)
uname = session['uname']
conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
cursor = conn.cursor()
cursor.execute("SELECT * FROM eventtb where eid='" + pid + "'")
data = cursor.fetchone()
if data:
    EName = data[1]
    cat = data[2]
    Place = data[3]
else:
    return 'No Record Found!'
conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
cursor = conn.cursor()
cursor.execute(
    "INSERT INTO applytb VALUES ('" + EName + "','" + Place + "','" + name
+ "','" + gender + "','" + dob + "','" + mobile + "','" + email + "','"
+ "','" + address + "','" + pin + "','" + height + "','" + weight +
,'" + file.filename + "','" + file1.filename + "','waiting','" + uname + "','" + cat + "')"
)
conn.commit()
conn.close()
flash('Registered successfully')
conn = mysql.connector.connect(user='root', password="", host='localhost',

```

```

database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM eventtb ")
    data = cur.fetchall()
    return render_template('UApply.html', data=data)

@app.route("/AApplyInfo")
def AApplyInfo():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where Status='waiting'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('AApplyInfo.html', data=data, data1=data1)

@app.route("/VAdd")
def VAdd():
    id = request.args.get('id')
    session['pid'] = id
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb  where aid='" + id + "' ")
    data = cur.fetchall()
    return render_template('APersonal.html', data=data)

@app.route("/VAdd1")
def VAdd1():
    id = request.args.get('id')
    session['pid'] = id

```



```

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where aid='" + id + "' ")
    data = cur.fetchall()
    return render_template('APersonal1.html', data=data)
@app.route('/download')
def download():
    id = request.args.get('id')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM applytb where aid = '" + str(id) + "'")
    data = cursor.fetchone()
    if data:
        filename = "static\\uploads\\"+data[13]
        return send_file(filename, as_attachment=True)
    else:
        return 'Incorrect username / password !'
@app.route("/Reject")
def Reject():
    id = request.args.get('id')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cursor = conn.cursor()
    cursor.execute(
        "update applytb set Status='Rejected' where aid='" + id + "'")
    conn.commit()
    conn.close()
    flash('Player Application is Rejected...!')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where Status='waiting'")

```

```

    data = cur.fetchall()

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')

    cur = conn.cursor()

    cur.execute("SELECT * FROM applytb where Status='Accepted'")

    data1 = cur.fetchall()

    return render_template('AApplyInfo.html', data=data, data1=data1)

@app.route("/Accept")
def Accept():
    id = request.args.get('id')

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')

    cursor = conn.cursor()

    cursor.execute(
        "update applytb set Status='Accepted' where aid='" + id + "'")

    conn.commit()

    conn.close()

    flash('Player Application is Accepted...!')

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')

    cur = conn.cursor()

    cur.execute("SELECT * FROM applytb where Status='waiting'")

    data = cur.fetchall()

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')

    cur = conn.cursor()

    cur.execute("SELECT * FROM applytb where Status='Accepted'")

    data1 = cur.fetchall()

    return render_template('AApplyInfo.html', data=data, data1=data1)

@app.route("/UApplyinfo")
def UApplyinfo():
    uname = session['uname']

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')

```

```

cur = conn.cursor()
cur.execute("SELECT * FROM applytb where UserName='"+ uname +"' ")
data = cur.fetchall()
return render_template('UApplyinfo.html',data=data)
@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html',data=data,data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO winnertb VALUES ('" + player + "','" + cat + "','" + pos +
            "')")
        conn.commit()
        conn.close()
        flash('New List Added successfully')
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

```

```

        cur = conn.cursor()
        cur.execute("SELECT * FROM winnertb ")
        data = cur.fetchall()
        return render_template('NewWinner.html', data=data)
@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html',data=data,data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO winnertb VALUES ('" + player + "','" + cat + "','" + pos +
            "')")
        conn.commit()
        conn.close()
        flash('New List Added successfully')
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

```

```

        cur = conn.cursor()
        cur.execute("SELECT * FROM winnertb ")
        data = cur.fetchall()
        return render_template('NewWinner.html', data=data)
@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('AWinnerList.html',data=data)
@app.route("/UWinnerList")
def UWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('UWinnerList.html',data=data)
if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('AWinnerList.html',data=data)

@app.route("/UWinnerList")
def UWinnerList():

```

```

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('UWinnerList.html',data=data)
if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

@app.route("/Accept")
def Accept():
    id = request.args.get('id')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cursor = conn.cursor()
    cursor.execute(
        "update applytb set Status='Accepted' where aid='" + id + "'"
    )
    conn.commit()
    conn.close()
    flash('Player Application is Accepted...!')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where Status='waiting'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('AApplyInfo.html', data=data, data1=data1)
@app.route("/UApplyinfo")
def UApplyinfo():
    uname = session['uname']

```

```

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM applytb where UserName='"+ uname +" ")
    data = cur.fetchall()
    return render_template('UApplyinfo.html',data=data)
@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html',data=data,data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO winnertb VALUES ('" + player + "','" + cat + "','" + pos +
            "')")
        conn.commit()
        conn.close()
        flash('New List Added successfully')

```

```

        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cur = conn.cursor()
        cur.execute("SELECT * FROM winnertb ")
        data = cur.fetchall()
        return render_template('NewWinner.html', data=data)
@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html',data=data,data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cursor = conn.cursor()
        cursor.execute(
            "INSERT INTO winnertb VALUES ('" + player + "'," + cat + "'," + pos +
            "'")
        conn.commit()
        conn.close()
        flash('New List Added successfully')

```



```

        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
        cur = conn.cursor()
        cur.execute("SELECT * FROM winnertb ")
        data = cur.fetchall()
        return render_template('NewWinner.html', data=data)
@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('AWinnerList.html',data=data)
@app.route("/UWinnerList")
def UWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('UWinnerList.html',data=data)
if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('AWinnerList.html',data=data)

```

```

@app.route("/UWinnerList")
def UWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('UWinnerList.html',data=data)
if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html',data=data,data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')
        cursor = conn.cursor()
        cursor.execute(

```

```

        "INSERT INTO winnertb VALUES ('" + player + "','" + cat + "','" + pos +
        "')")
    conn.commit()
    conn.close()
    flash('New List Added successfully')
    conn = mysql.connector.connect(user='root', password="", host='localhost',
    database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb ")
    data = cur.fetchall()
    return render_template('NewWinner.html', data=data)
@app.route("/NewWinner")
def NewWinner():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
    database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Name FROM applytb where Status='Accepted'")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password="", host='localhost',
    database='l sportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT distinct Ename FROM applytb where Status='Accepted'")
    data1 = cur.fetchall()
    return render_template('NewWinner.html', data=data, data1=data1)
@app.route("/newwinner", methods=['GET', 'POST'])
def newwinner():
    if request.method == 'POST':
        player = request.form['player']
        cat = request.form['cat']
        pos = request.form['pose']
        conn = mysql.connector.connect(user='root', password="", host='localhost',
    database='l sportsmanagement')
        cursor = conn.cursor()
        cursor.execute(

```

```

        "INSERT INTO winnertb VALUES ('" + player + "','" + cat + "','" + pos +
    """)")

    conn.commit()
    conn.close()

    flash('New List Added successfully')

    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb ")
    data = cur.fetchall()

    return render_template('NewWinner.html', data=data)

@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()

    return render_template('AWinnerList.html',data=data)

@app.route("/UWinnerList")
def UWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()

    return render_template('UWinnerList.html',data=data)

if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

@app.route("/AWinnerList")
def AWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='l sportsmanagement')

```

```

    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('AWinnerList.html',data=data)

@app.route("/UWinnerList")
def UWinnerList():
    conn = mysql.connector.connect(user='root', password="", host='localhost',
database='lsportsmanagement')
    cur = conn.cursor()
    cur.execute("SELECT * FROM winnertb")
    data = cur.fetchall()
    return render_template('UWinnerList.html',data=data)
if __name__ == '__main__':
    app.run(debug=True, use_reloader=True)

```

8.2 O/P SCREENS

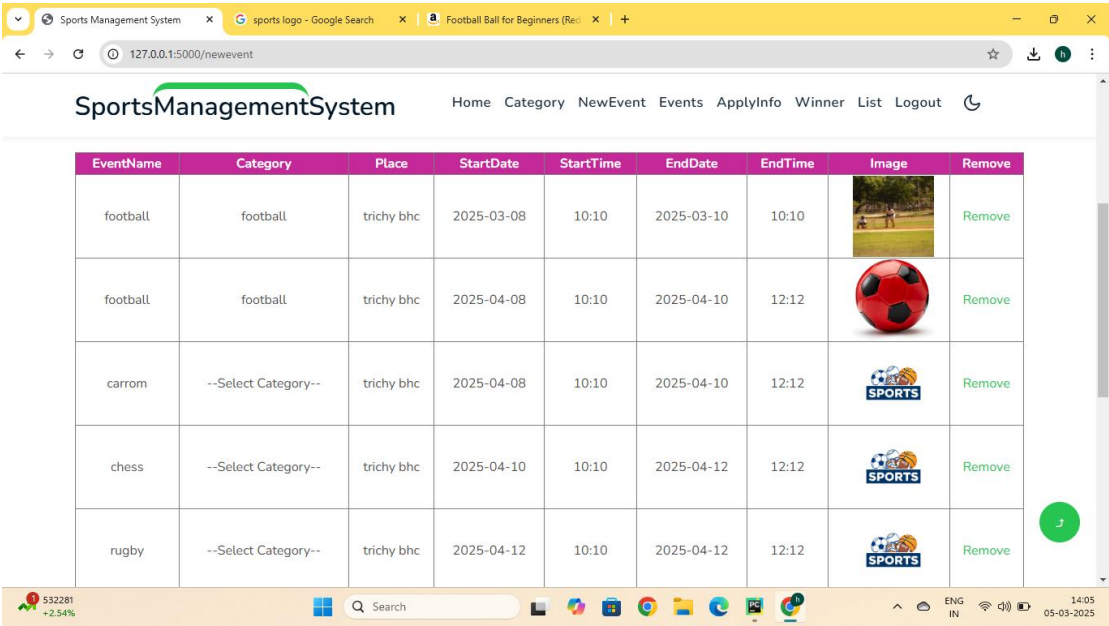


Figure 8.2.1 HomePage Screen



Figure 8.2.2 Events Lists

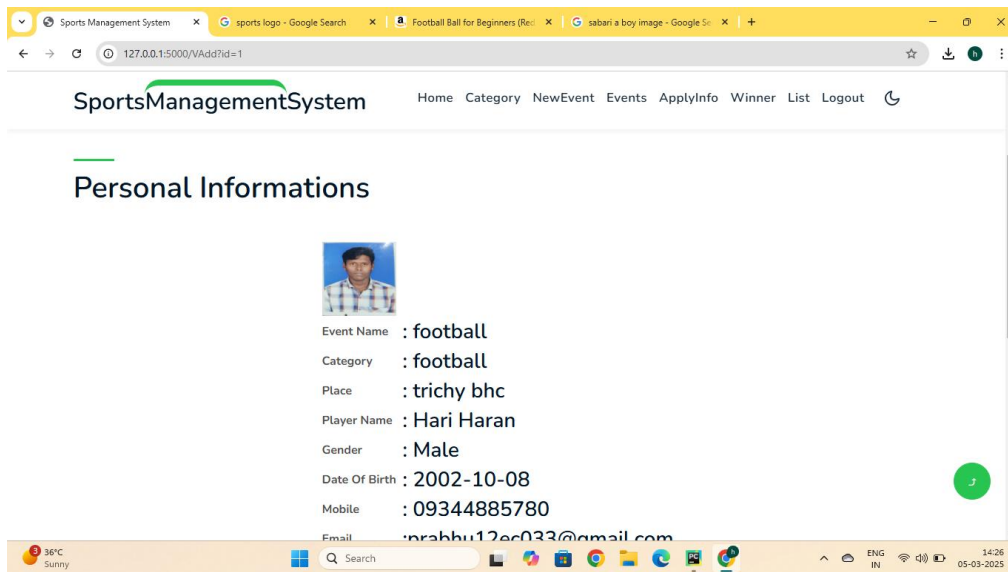


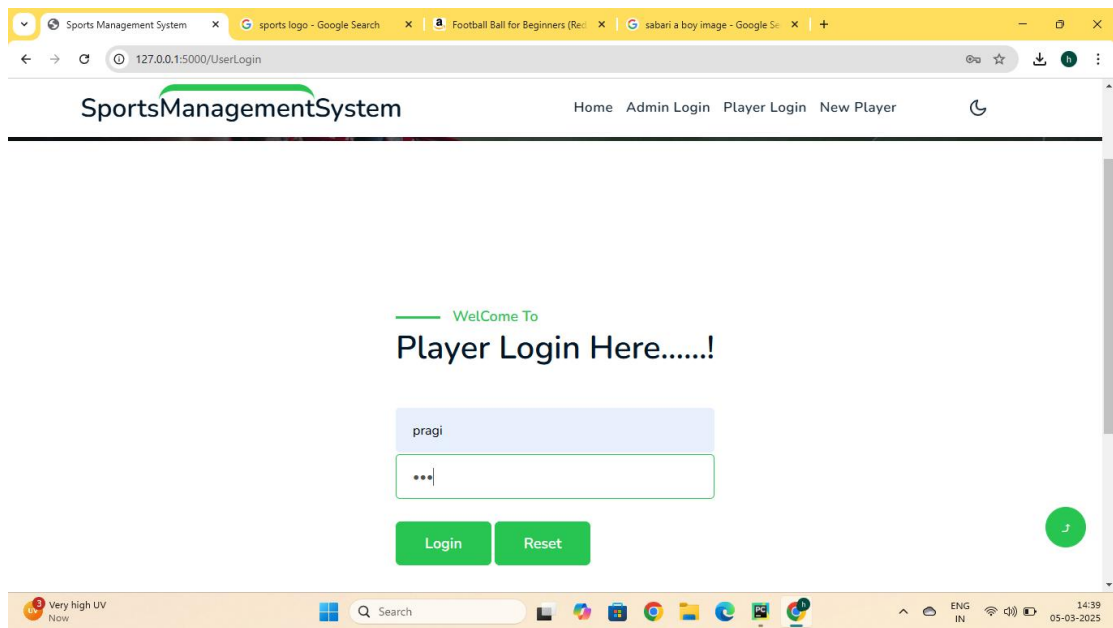
Figure 8.2.3 Personal information

The screenshot displays the 'Player Informations' page of the Sports Management System, showing a table with player details. The table has columns for Name, Gender, Mobile, EmailId, Address, and UserName.

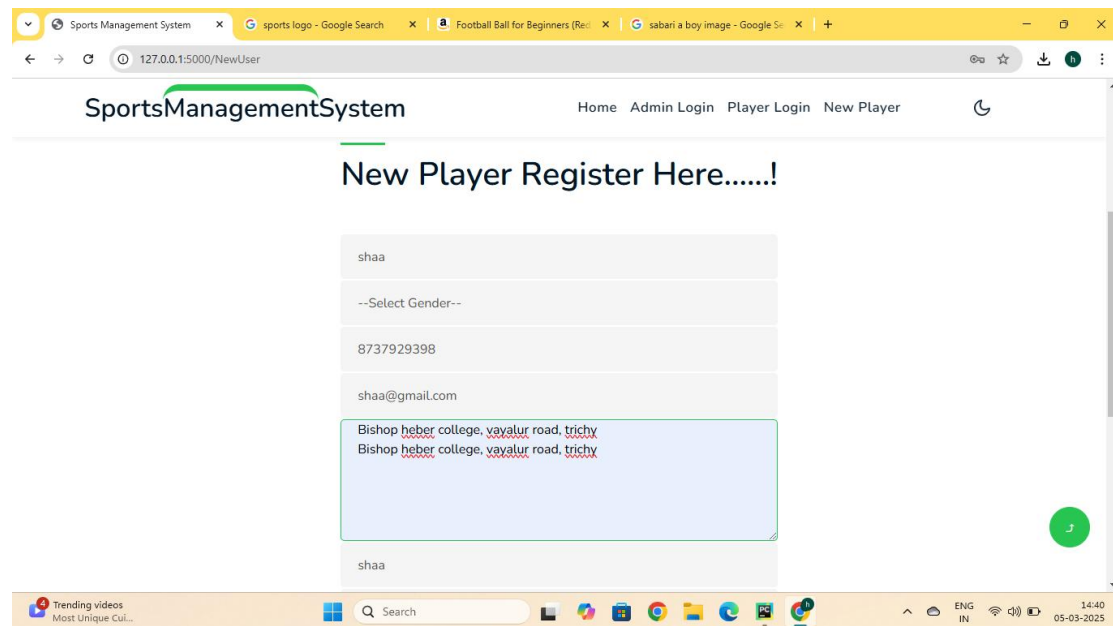
Name	Gender	Mobile	EmailId	Address	UserName
hari	Male	9344885780	prabhu12ec033@gmail.com	trichy	hari
sudip	Male	9943132111	gsguhan764@gmail.com	Bishop heber college, vayalur road, trichy	sudip
sabari	Male	9344885780	crharicr@gmail.com	Bishop heber college, vayalur road, trichy	sabari
saravana	Male	9923849987	crharicr@gmail.com	C3 Vikram Nagar	saravana
ajay	Male	9398459867	ca235213122@bhc.edu.in	C3 Vikram Nagar	ajay
sakthi	Male	9344885780	vedhapriyaramesh2840@gmail.com	Bishop heber college, vayalur road, trichy	sakthi
mathesh	Male	7389289932	mathesh@gmail.com	C3 Vikram Nagar	mathesh
arjun	Male	9344885780	arjun@gmail.com	Bishop heber college, vayalur road, trichy	arjun
calep	Male	9344885780	calep@gmail.com	Bishop heber college, vayalur road, trichy	calep
pragi	Male	9344885780	pragi@gmail.com	Bishop heber college, vayalur road, trichy	pragi

The system's navigation bar includes links for Home, Category, NewEvent, Events, ApplyInfo, Winner, List, and Logout. The browser's address bar shows the URL 127.0.0.1:5000/adminlogin.

Figure 8.2.4 Players Information List



**Figure 8.2.5 Player Login Page**



**Figure 8.2.6 Player Registration Page**



**SportsManagementSystem** Home Category NewEvent Events ApplyInfo Winner List Logout

## Player Informations

Name	Gender	Mobile	EmailId	Address	UserName
hari	Male	9344885780	prabhu12ec033@gmail.com	trichy	hari
sudip	Male	9943132111	gsguhan764@gmail.com	Bishop heber college, vayalur road, trichy	sudip
sabari	Male	9344885780	crharicr@gmail.com	Bishop heber college, vayalur road, trichy	sabari
saravana	Male	9923849987	crharicr@gmail.com	C3 Vikram Nagar	saravana
ajay	Male	9398459867	ca235213122@bhc.edu.in	C3 Vikram Nagar	ajay
sakthi	Male	9344885780	vedhapriyamesh2840@gmail.com	Bishop heber college, vayalur road, trichy	sakthi
mathesh	Male	7389289932	mathesh@gmail.com	C3 Vikram Nagar	mathesh
arjun	Male	9344885780	arjun@gmail.com	Bishop heber college, vayalur road, trichy	arjun
calep	Male	9344885780	calep@gmail.com	Bishop heber college, vayalur road, trichy	calep
pragi	Male	9344885780	pragi@gmail.com	Bishop heber college, vayalur road, trichy	pragi

**Figure 8.2.7 Player Information**

**SportsManagementSystem** Home Category NewEvent Events ApplyInfo Winner List Logout

## Winner List

PlayerName	Event	Position
Hari Haran	football	Winner
sabari	rugby	Winner
pragi	atheletics	Winner
ajay	carrom	Winner
ajay	kabddi	Winner
Hari Haran	kabddi	Winner
pragi	chess	Runner
sabari	carrom	Runner
ajay	football	Runner

**Figure 8.2.8 Winners List**

Sports Management System x sports logo - Google Search x Football Ball for Beginners (Re: x sabari a boy image - Google S: x +

127.0.0.1:5000/NewEvent

SportsManagementSystem Home Category NewEvent Events ApplyInfo Winner List Logout

## New Event Register Here.....!

football

football

place

Start

10-04-2025

10:10

End

10-04-2025

12:12

36°C Sunny

Search

ENG IN 14:42 05-03-2025

**Figure 8.2.9 Event Registration Page**