PROJECT REPORT

On

APPLICATION FOR SHRUTHI SPORTS

By

GROUP 10

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)

(Affiliated to Osmania University; Accredited by NBA(AICTE) and NAAC(UGC), ISO Certified 9001:2015)

GANDIPET, HYDERABAD – 500 075

Website: www.cbit.ac.in

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)

DEPARTMENT OF INFORMATION TECHNOLOGY

(Affiliated to Osmania University)

GANDIPET, HYDERABAD – 500 075

CERTIFICATE

This is to certify that the project work entitled “**APPLICATION FOR SHRUTHI SPORTS**” submitted to **CBIT OPEN SOURCE COMMUNITY ,** in fulfilment of the COSC skill up series internship drive during the academic year 2020, is a record of original work done by the following students under our supervision and guidance.

S.NO	NAME	YEAR AND BRANCH
1	K.V.N Sai Ram Reddy	CSE 3/4
2	Dharani V	CSE 2/4
3	Shaik Abdullah Adnan	IT 2/4
4	Rahul pittala	CSE 2/4
5	Sanan Mouinuddin	CSE 1/4
6	Soma Pavani	CSE 1/4
7	Thoutam Dileep	CSE 1/4
8	J.S.S Chaitanyanand	CSE 1/4

Student Mentor
Mohammed Ameer Uddin
IT 3/4

FacultyMentor
Smt. M Trupthi
Asst. Professor
Dept. of IT

TABLE OF CONTENTS

DECLARATION

This report has been completed in accordance with the terms of reference prescribed by the mentor/guide of COSC , and the accomplished as part of project under cosc skill up series internship drive, prescribed by Chaitanya Bharathi Institute of Technology, Gandipet .

The content enclosed in this report is not fully endorsed by the author’s and alterations or corrections have not been made to suit the interest of Chaitanya Bharathi Institute of Technology. The recommendations and conclusion attained in this report is entirely a product of the authors’ standpoint and shall be treated only as a source of information.

ABSTRACT

Online application is one of the easiest methods for applying for anything. You are just few clicks away registering for a sport. This project has user and admin interface where the users (students) can register for their team using a mobile application and the admin accepts them through the web page.

This project is done using python, Java programming language and we have used Django, Flask ,HTML ,CSS ,Bootstrap and Android studio . The mobile app provides a platform for the students to register for a sport and the web app enables the sport’s faculty to schedule the matches online and to accept the registrations.

ACKNOWLEDGEMENTS

Our project report is a brainchild of the combined efforts of all members of the group and accomplishing this would not have been easy without the consistent guidance, inspiring words and relentless support received from the following, throughout the execution of this project work.

We would fall short of words to express our heartfelt gratitude and thanks to Mohammed Ameer Uddin, Mentor , Who has been a friend ,a guide and power house of inspiration to complete our project

We would like to express our sincere gratitude to Smt. M Trupthi, Asst professor, Department of Information Technology , Chaitanya Bharathi Institute of Technology, Gandipet, for providing us with stringent terms of reference and matchless guidance for pursuit of this project.

We would also like to thank the entire COSC team for providing us an internship opportunity during this pandemic

We are indeed grateful to the above mentioned distinguished men/women.

LIST OF FIGURES

1. INTRODUCTION

1.1 Problem Statement: Application For Shruthi Sports

Users:

- Login-captain/individual
- Register for sports with team details
- Registration status(accept/ pending)
- View registered sports details/updates (updates in form of notification)
- View reporting times etc
- View sports categories(indoor/outdoor) date scheduled and details(tentative timings)

Admin:

- Login
- View sports/modify sport details
- View registrations and accept them
- Schedule sports timing/dates/matches between etc.
- Set reporting time
- Edit schedule

1.2 Basic Definition:

Shruthi is the inter-department 3-day cultural fest organized at the end of each even semester, which acts as a host for many cultural events. The inter-department fest, an inter-department sports competition is held. Sports in the fest are basketball, football, cricket, badminton, volleyball, kabaddi, throwball and track and field events along with various other indoor games.

As we all know how difficult and confusing the applications get every year, So with help of our project the task gets simple. Every student is allowed to sign-up for the app ,where he can get all the updates regarding the sports fest. He/She can also register for sports through this app. We have also created a web application for the sports faculty , where the faculty can handle the registrations and schedule the matches.

2.SYSTEM REQUIREMENT SPECIFICATION

a. Hardware Requirements

- Microsoft® Windows® 7/8/10 (64-bit)

- 4GB **RAM minimum**, 8 GB **RAM** recommended.

- 2 GB of available disk space **minimum**, 4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)

-1280 x 800 **minimum** screen resolution.

-1.8 GHz or faster processor. Quad-core or better recommended

b. Software Requirements

- Operating systems: Windows® 10, macOS*, and Linux*
- Android 6 or above

c. Tools and Technologies Used

- MySQL Workbench
- Flask-restful
- Python
- Django
- HTML , CSS, Bootstrap
- Java
- Android Studio
- Visual Studio Code

3. SYSTEM DESIGN

a. Use case diagram for admin and user

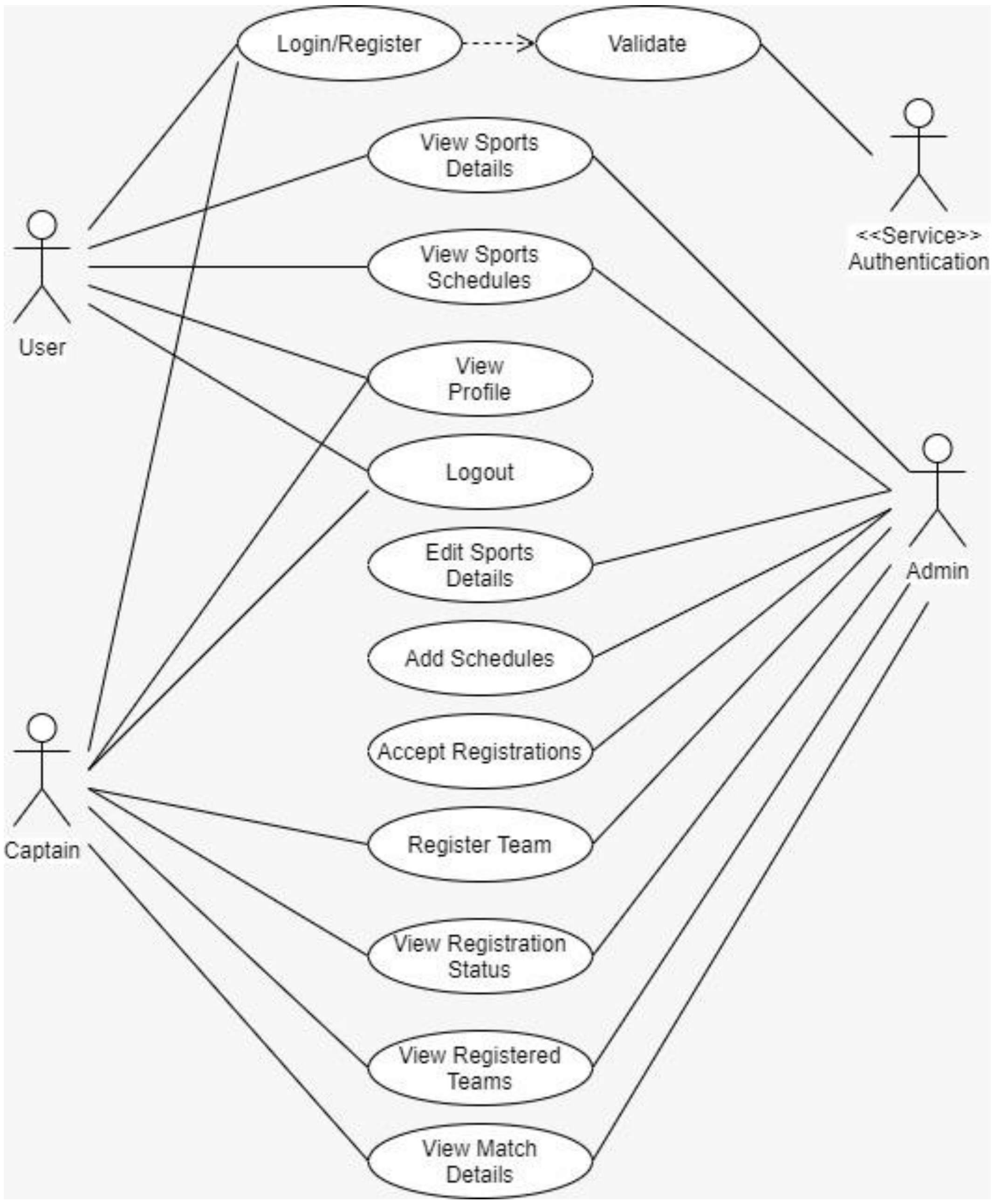


Fig 3.1: Use case diagram for admin and user

There are 2 primary actors for the use case diagram one is user and one is captain user can access login/register, sport details, sports schedules, profile and logout. captain can access login/register, profile, register team, registration status,registered teams, match details etc. And there is a authentication service for authentication of user input data and finally admin is the secondary actor who has access to editing sports details, add schedules, accept registrations, team registrations and match details etc.

b. Activity diagram for the users

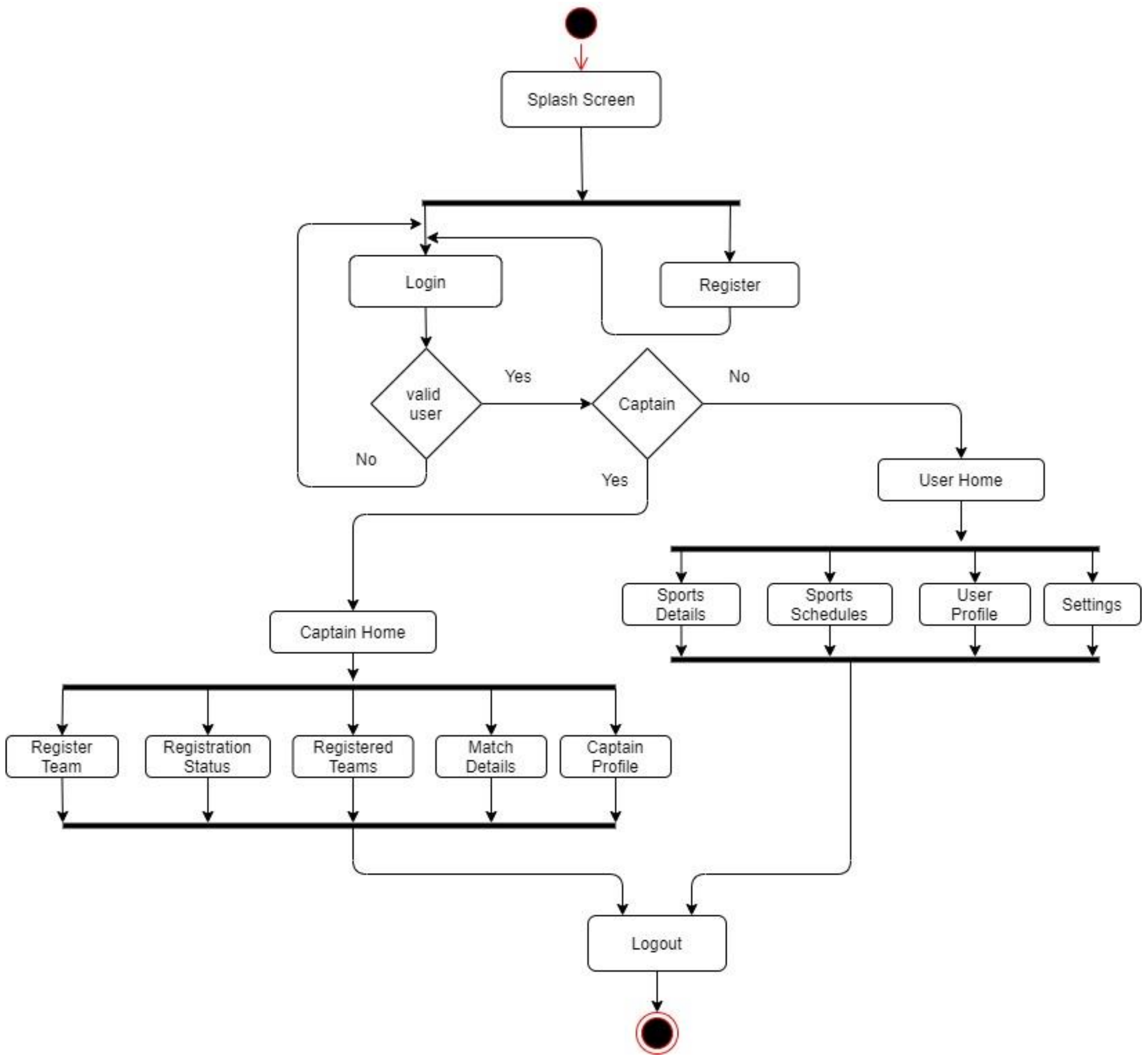


Fig 3.2: Activity diagram for the users

Whenever the user opens the app the splash screen activity where intializations takes place, from the splash screen user can go to either login or register page then after authorization he is redirected to user's or captain's home screen activities from where they can access various other activities like
register team registration status, registered teams, match details, sports details,sports schedules, profile, settings etc. Finally when they try to goto logout activity all the users on device stored details are erased.

c. Data Flow diagram for users

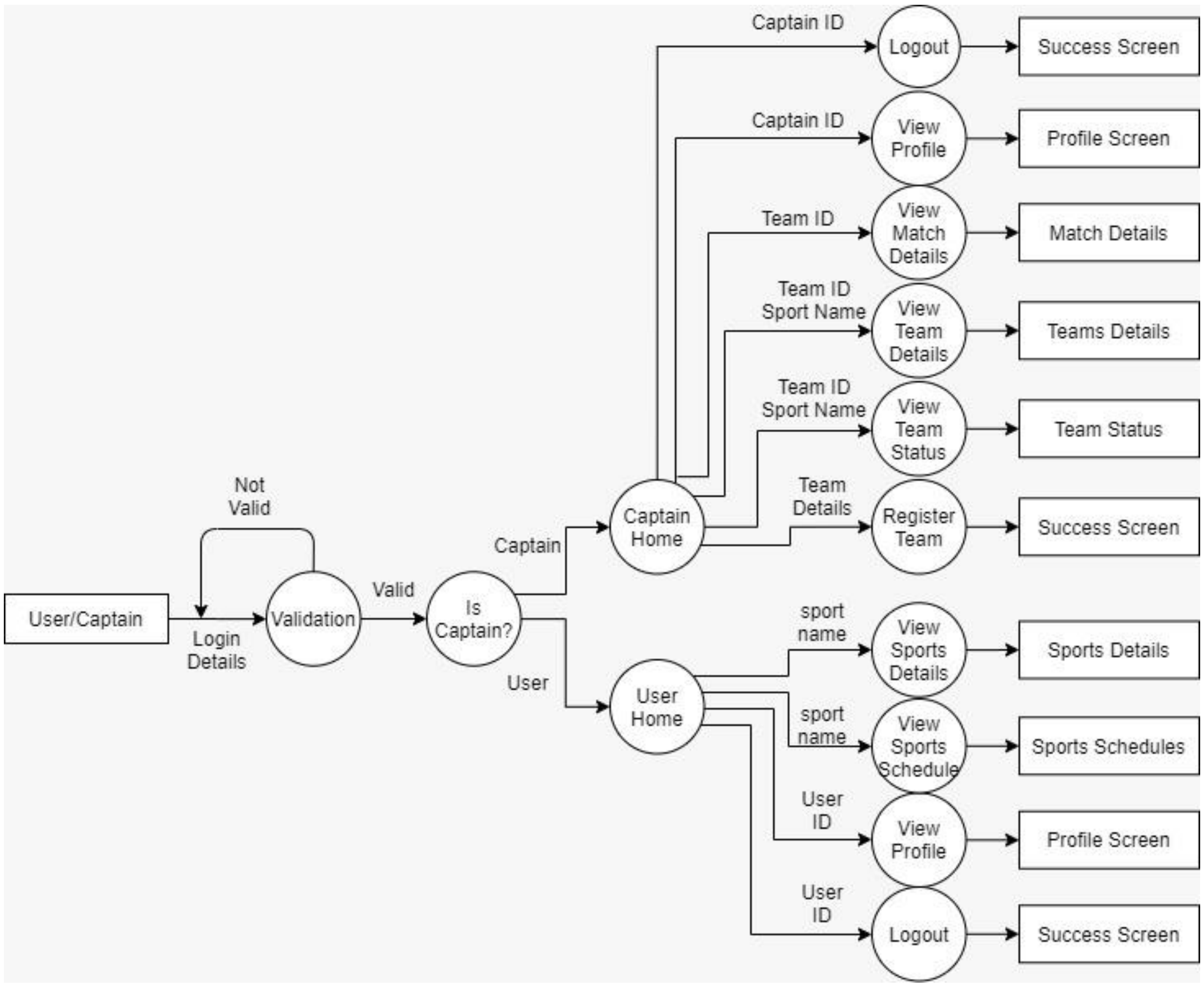


Fig 3.3: Data Flow diagram for users

When the user enters he goes to validation activity where he gives his details and validation of details takes place and these details are sent to next screens. Based on whether he is captain or not he goes to captain home or user home. from there he can goto many options available, for captain captain's ID and team details are passed and for user user's ID and sports details will be passed and finally the corresponding output is shown.

d. Data Flow diagram for admin

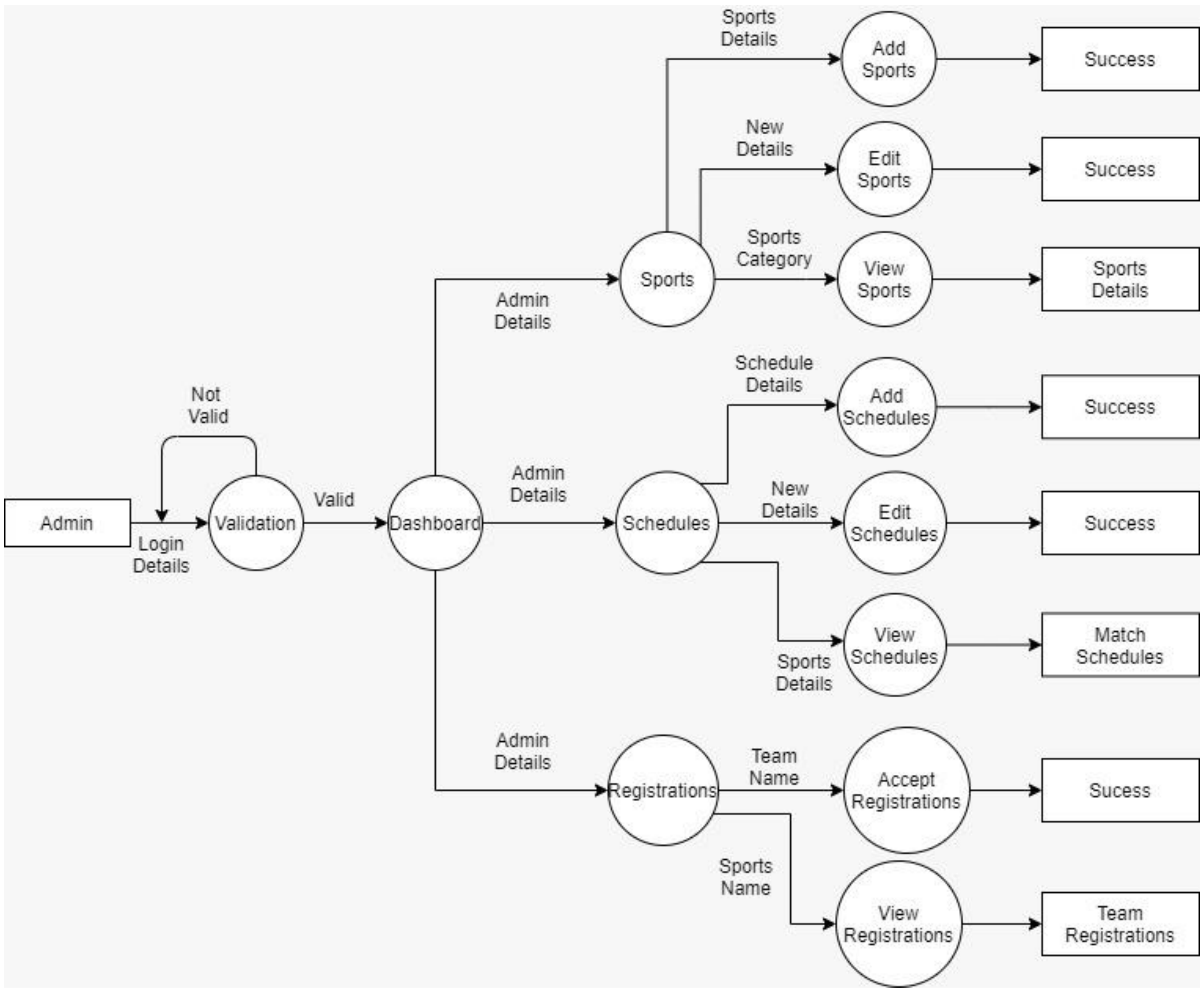


Fig 3.4: Data Flow diagram for admin

When the admin opens the application first he must login by providing their credentials if these are verified, these details will be forwarded to next pages. From there admin goes to three main activites where his details are forwarded. From those screen the required information like sports details, categories, schedules, team etc are forwarded to next activities to show the result screens.

4. IMPLEMENTATION

Our work is divided into three parts

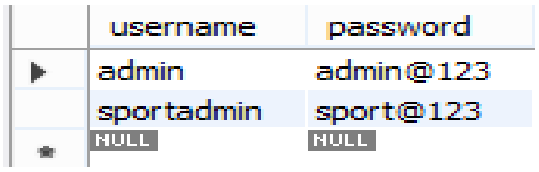
a. Database and rest API's
b. Web app
c. Android app

a. Database

Databases are the heart of mobile and web apps. For an application to be fully functional and interactive , we need database. Database collect and store user information during a one-time registration process and after that, it neglects the need to repeatedly log into the app until the user manually logs out. For our project we have used MYSQL to create the database tables.This project five tables according to our requirements.

i. Admin

We have an admin table to store the user name and pass word of the admin. Using the details present in the table the admin will login to the web app.

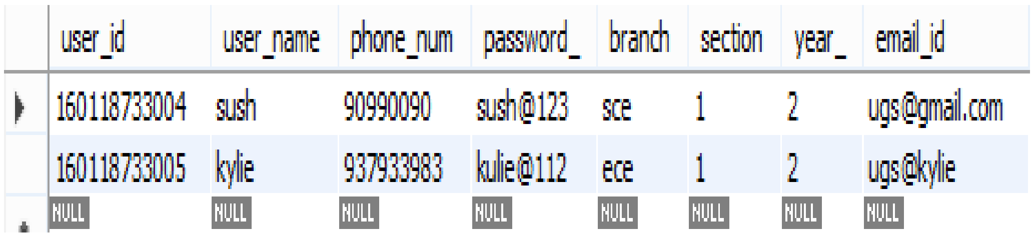


username	password
admin	admin@123
sportadmin	sport@123
NULL	NULL

Fig 4.1: Admin table

ii. Users

When a user signs up for the mobile application . His/her credentials will be stored in this table.



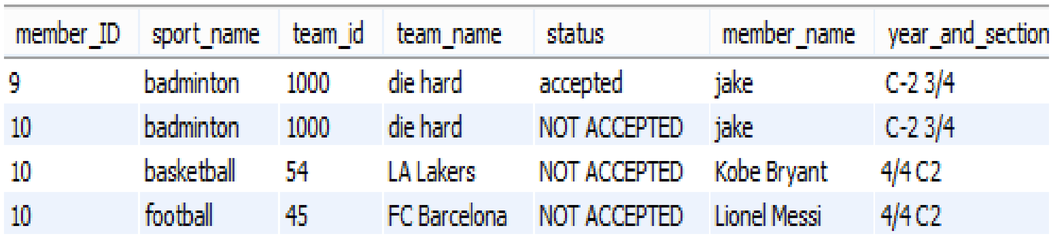
user_id	user_name	phone_num	password_	branch	section	year_	email_id
160118733004	sush	90990090	sush@123	sce	1	2	ugs@gmail.com
160118733005	kylie	937933983	kulie@112	ece	1	2	ugs@kylie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig 4.2: Users table

iii. Team details

Whenever the captain registers for his team, Details of the team members are stored in this table.

Member ID and sport name are primary key to ensure that a single person doesn't register for the same sport twice

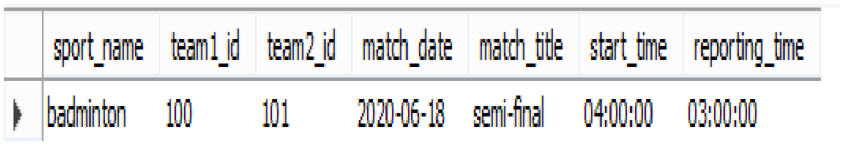


member_ID	sport_name	team_id	team_name	status	member_name	year_and_section
9	badminton	1000	die hard	accepted	jake	C-2 3/4
10	badminton	1000	die hard	NOT ACCEPTED	jake	C-2 3/4
10	basketball	54	LA Lakers	NOT ACCEPTED	Kobe Bryant	4/4 C2
10	football	45	FC Barcelona	NOT ACCEPTED	Lionel Messi	4/4 C2

Fig 4.3: Team details table

iv. Schedule

This is the table where the admin will be storing the data regarding the scheduled matches.

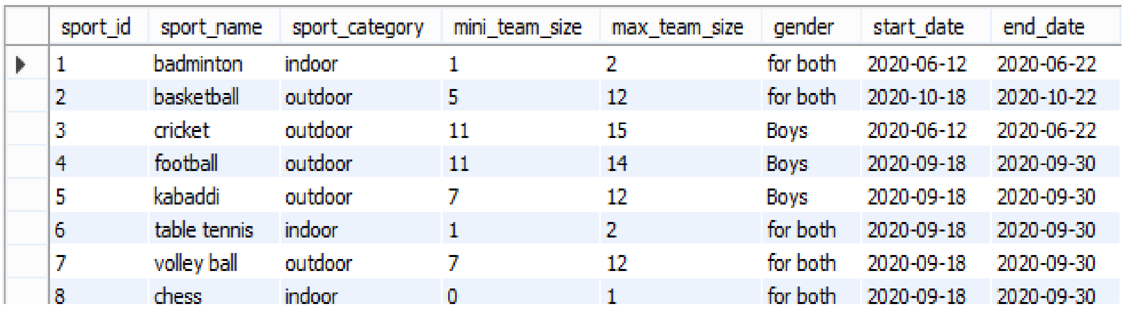


sport_name	team1_id	team2_id	match_date	match_title	start_time	reporting_time
badminton	100	101	2020-06-18	semi-final	04:00:00	03:00:00

Fig 4.4: Schedule table

v. Sports

This table consists the details of all the sports for which the students can register .



sport_id	sport_name	sport_category	mini_team_size	max_team_size	gender	start_date	end_date
1	badminton	indoor	1	2	for both	2020-06-12	2020-06-22
2	basketball	outdoor	5	12	for both	2020-10-18	2020-10-22
3	cricket	outdoor	11	15	Boys	2020-06-12	2020-06-22
4	football	outdoor	11	14	Boys	2020-09-18	2020-09-30
5	kabaddi	outdoor	7	12	Boys	2020-09-18	2020-09-30
6	table tennis	indoor	1	2	for both	2020-09-18	2020-09-30
7	volley ball	outdoor	7	12	for both	2020-09-18	2020-09-30
8	chess	indoor	0	1	for both	2020-09-18	2020-09-30

Fig 4.5: Sports table

b. APIs

Application program interfaces, are vital tools for projects. They allow the capabilities of one program to be used by another. They are a means by which two different programs are able to communicate.

In our project we have used Flask-restful extension to create the apis. Flask Restful is an extension for Flask that adds support for building REST APIs in Python using Flask as the back-end.

Before creating the api we have installed the following packages using the commands below

1) Flask:

Pip install flask

2) flask-restful:

Pip install flask-restful

3) Pymysql:

Pip install pymysql

4)Flask JWT

Pip install Flask-JWT

Pip install flask-jwt-extended

API For The Web Page(Admin):

Our api folder is divided into the following files

i) App.py

This is the flask app where all the endpoints are present .

Code :

```
from flask import Flask,jsonify
from flask_restful import Api
from flask_jwt_extended import JWTManager
from resources.admin import Adminlogin,Sport,Schedule,Team_details,Team_members,Modify_schedul
e,Add_schedule,Add_dates,Sport_category
```

```
app=Flask(__name__)
app.config['PROPAGATE_EXCEPTIONS']=True
app.config['JWT_SECRET_KEY']='group10'
app.config['PREFERRED_URL_SCHEME']='http'
api=Api(app)
jwt=JWTManager(app)
@jwt.unauthorized_loader
def missing_token_callback(error):
    return jsonify({
        'error': 'authorization_required',
        "description": "Request does not contain an access token."
    }), 401

@jwt.invalid_token_loader
def invalid_token_callback(error):
    return jsonify({
        'error': 'invalid_token',
        'message': 'Signature verification failed.'
    }), 401
api.add_resource(Adminlogin,'/Adminlogin')
api.add_resource(Sport,'/sportdetails')
api.add_resource(Schedule,'/schedule')
api.add_resource(Modify_schedule,'/Modify_schedule')
api.add_resource(Team_details,'/team_details')
api.add_resource(Team_members,'/team_members')
api.add_resource(Add_schedule,'/add_schedule')
api.add_resource(Add_dates,'/add_dates')

api.add_resource(Sport_category,'/sport_category')
```

ii) db.py

This is the file where we link our api to the database

Code:

```
from flask import jsonify
from decimal import Decimal
import pymysql
def query(querystr,return_json=True):
    connection=pymysql.connect( host='skillup-team-10.cxgok3weok8n.ap-south-
1.rds.amazonaws.com',
                                user='admin',
                                password='coscskillup',
                                db='group10',
                                cursorclass=pymysql.cursors.DictCursor )
    connection.begin()
    cursor=connection.cursor()
    cursor.execute(querystr)
    result=encode(cursor.fetchall())
    connection.commit()
    cursor.close()
    connection.close()
    if return_json:
        return jsonify(result)
    else:
        return result

def encode(data):
    for row in data:
        for key,value in row.items():
            if isinstance(value,Decimal):
                row[key]=str(value)
    return data
```

iii) Resources folder

We have two files :

a)__init__,py: this file is used to import the python package

b)Admin.py: in this file all the end points are defined and are tested using the postman tool.

We have used two methods i.e Get and Post . Get method is used to get the data from the database and post method is used to add the data in the database

According to the problem statement this project has nine endpoints for admins, out of which we have choosen important ones for showing the demo

- Post method for admin login :

The admin has to provide the user name and password to get the access token

Code:

```
from flask_restful import Resource,reqparse
from werkzeug.security import safe_str_cmp
from flask_jwt_extended import create_access_token,jwt_required
from db import query
class Adminlogin(Resource):#admin login
    parser=reqparse.RequestParser()
    parser.add_argument('username',type=str,required=True,help="Admin username cannot be left
blank!")
    parser.add_argument('password',type=str,required=True,help="Password cannot be left blank!
")

    def post(self):
        data=self.parser.parse_args()
        user=User.getUserById(data['username'])
        if user and safe_str_cmp(user.password,data['password']):
            access_token=create_access_token(identity=user.username,expires_delta=False)
            return {'access_token':access_token},200
        return {"message":"Invalid Credentials!"}, 401
class User():
    def __init__(self,username,password):
        self.username=username
        self.password=password
    @classmethod
    def getUserById(cls,username):
        result=query(f"""select username,password from admin where username='{username}'""",re
turn_json=False)
        if len(result)>0:
            return User(result[0]['username'],result[0]['password'])
        else:
            return None
```

- Get method for viewing sports:

Whenever the admin enters the sport category all the sports which belong to that particular sport category will be displayed

Code:

```
class Sport_category(Resource):
    @jwt_required
    def get(self):
        parser=reqparse.RequestParser()
        parser.add_argument('sport_category',type=str,required=True,help="Sport Category canno
t be left blank!")
        data=parser.parse_args()
        try:
            return query(f"""SELECT * FROM group10.sports WHERE sport_category='{data['sport_c
ategory']}'; """)
        except:
            return {"message":"There has been an error retrieving sports details"},500
        return {"message":"Sports details retrieved succesfully."}
```

API For Android App (Users)

All the files which are mentioned in the api for the admin are also present in the users. The only file which changes is the app.py where in all the end points are present. According to the requirements we have 11 endpoints.

i) app.py

```
from flask import Flask,jsonify
from flask_restful import Api
from flask_jwt_extended import JWTManager
from resources.user import Userlogin,User_reg,Sport_Registration,Team_status,Spor
tdetails,Reporting_time,Sport_category,Schedules,User_info,Registered_sports,Chan
ge_password
```

```
app=Flask(__name__)
app.config['PROPAGATE_EXCEPTIONS']=True
app.config['JWT_SECRET_KEY']='group10'
app.config['PREFERRED_URL_SCHEME']='http'
api=Api(app)
jwt=JWTManager(app)
@jwt.unauthorized_loader
def missing_token_callback(error):
    return jsonify({
        'error': 'authorization_required',
        "description": "Request does not contain an access token."
    }), 401

@jwt.invalid_token_loader
def invalid_token_callback(error):
    return jsonify({
        'error': 'invalid_token',
        'message': 'Signature verification failed.'
    }), 401
api.add_resource(Userlogin,'/Userlogin')
api.add_resource(User_reg,'/User_reg')
api.add_resource(Sport_Registration,'/registration')
api.add_resource(Team_status,'/team_status')
api.add_resource(Sportdetails,'/sportdetails')
api.add_resource(Reporting_time,'/reporting_time')
api.add_resource(Sport_category,'/sportcategory')
api.add_resource(Schedules,'/schedule')
api.add_resource(User_info,'/user_info')
api.add_resource(Registered_sports,'/team_member_details')
api.add_resource(Change_password,'/change_password')




if __name__ == "__main__":
    app.run(debug=True)
```

- Post method for Sign-up for app: The user has to provide the credentials to signup for the app.

```
class User_reg(Resource):#user registration

    def post(self):
        parser=reqparse.RequestParser()
        parser.add_argument('user_id',type=int,required=True,help="user_id cannot be left blan
k!")
```

```python
        parser.add_argument('user_name',type=str,required=True,help="user_name cannot be left blank!")
        parser.add_argument('phone_num',type=int,required=True,help="phone_num cannot be left blank!")
        parser.add_argument('password_',type=str,required=True,help="password_ cannot be left blank!")
        parser.add_argument('branch',type=str,required=True,help="branch cannot be left blank!")
        parser.add_argument('section',type=int,required=True,help="section cannot be left blank!")
        parser.add_argument('year_',type=int,required=True,help="year cannot be left blank!")
        parser.add_argument('email_id',type=str,required=True,help="branch cannot be left blank!")
        data=parser.parse_args()
        try:
            x=query(f"""SELECT * FROM group10.users WHERE user_id={data['user_id']}""",return_json=False)
            if len(x)>0: return {"message":"A user with that user_id already exists."},400
        except:
            return {"message":"There was an error inserting into users table."},500
        try:
            query(f"""INSERT INTO group10.users (user_id, user_name, phone_num, password_, branch, section, year_, email_id) VALUES ({data['user_id']},
            '{data['user_name']}',{data['phone_num']},'{data['password_']}','{data['branch']}',{data['section']},{data['year_']},'{data['email_id']}');""")
        except:
            return {"message":"There was an error inserting into users table."},500
        return {"message":"Successfully Inserted."},201
```

- Get method for reporting time: The captain has to provide his id and the sport in which he registered, to get the information regarding schedule

```python
class Reporting_time(Resource):
    @jwt_required
    def get(self):
        parser=reqparse.RequestParser()
        parser.add_argument('team_id',type=int,required=True,help="Team id cannot be left blank!")
        parser.add_argument('sport_name',type=str,required=True,help="sport name cannot be left blank!")
        data=parser.parse_args()
        try:
            return query(f"""SELECT * FROM group10.schedule1 WHERE (team1_id={data['team_id']} OR team2_id={data['team_id']}) AND sport_name='{data['sport_name']}'; """)
        except:
            return {"message":"There has been an error retrieving schedule details"},500
        return {"message":"Schedule retrieved succesfully."}
```

Heroku links:

Below are the links which we got after deploying our apis

For users:https://group-10-user-api.herokuapp.com/

For Admins: https://group-10-api.herokuapp.com/

Website for admins:

Let's see a bit of coding part involved in creating the web pages. There are three important pages where the functionality takes place i. Urls.py ii.Views.py iii.html pages

i)Urls. Py

This is the file where you define the mapping between **URLs** and views. A mapping is a tuple in **URL** patterns like − from **django**. conf. **urls** import patterns, include, **url** from **django**.

Code:

```
from django.urls import path

from . import views

urlpatterns = [

    path('',views.login, name="login"),
    path("index",views.index, name="index"),

    path("add_sport",views.add_sport, name="add_sport"),
    path("edit_sport",views.edit_sport, name="edit_sport"),
    path("view_sport",views.view_sport, name="view_sport"),
    path("add_schedule",views.add_schedule, name="add_schedule"),
    path("edit_schedule",views.edit_schedule, name="edit_schedule"),
    path("edit_registration",views.edit_registration, name="edit_registration"),
    path("view_registration",views.view_registration, name="view_registration"),
    path("view_schedule",views.view_schedule, name="view_schedule"),
    path("add_registration",views.add_registration, name="add_registration"),

]
```

ii)View.py

A view function, or *view* for short, is a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it's on your Python path. There's no other requirement–no "magic," so to speak. For the sake of putting the code *somewhere*, the convention is to put views in a file called **views.py**, placed in your project or application directory.

Let us have a look at the code of a function named **add_sport**within the **views.py** file:

```
def add_sport(request) :
  if(p!=None):
    if (request.method)=="POST":
         Sport_id=request.POST.get('Sport_id')
         Sport_name=request.POST.get('Sport_name')
         cat=request.POST.get('Sport_category')
         mini=request.POST.get('min_team_size')
         maxi=request.POST.get('max_team_size')
         gender=request.POST.get('gender')

         print(Sport_id,mini)
         a = requests.post('https://group-10-
api.herokuapp.com/sportdetails',headers={'Authorization':f'Bearer {p}'},data={'sport_id':Sport_id,'sport_
name':Sport_name,'sport_category':cat,'mini_team_size':mini,'max_team_size':maxi,'gender':gender})
         print(a)
         return render(request,'add_sport.html')
    else:
        return render(request,'add_sport.html')
  else:
    return render(request,"login.html")
```

iii)Html page

The coding for the web page was done with HTML, CSS and JS. With a combination of all three languages and other such language concepts as well, the website was successfully made possible.
Let us have a look at the code of one of the HTML pages within our project, i.e. **add_sport.html**:

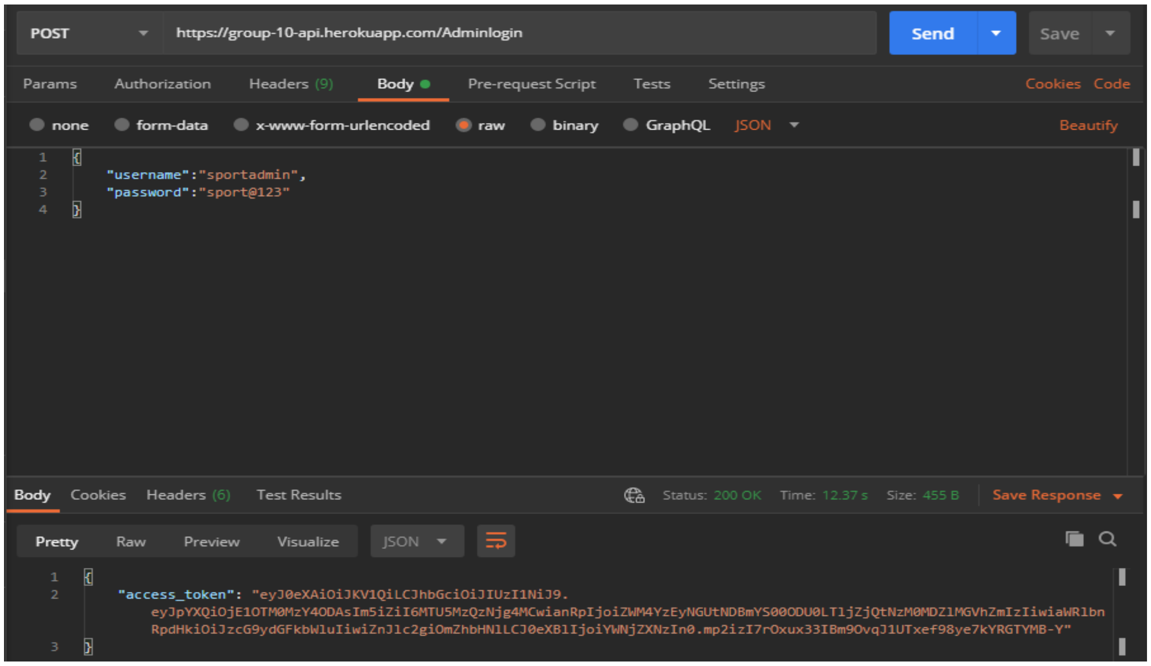
```
{%extends 'base.html'%}
{% load static %}
{%block title%}
Add Sport
{%endblock%}

{%block main%}

  <!-- Content Wrapper -->
  <div id="content-wrapper" class="d-flex flex-column">

    <!-- Main Content -->
    <div id="content p-0">
```

```
     <!-- Topbar -->
     <nav class="navbar navbar-expand navbar-light bg-white topbar mb-4 static-top shadow pb-2">

       <div class="navbar-header">
         <h1 class=" px-2  text-primary">ADD SPORT</h1>
       </div>

      </nav>


</div>

<form method="POST" id="myForm" action="{%url 'add_sport'%}">
                              {% csrf_token %}
                 <div class="user">


     <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="Sport_id" placeholder="Enter Sport ID..." required>
                   </div>
                   <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="Sport_name" placeholder="Enter Sport Name..." required>
                   </div>
                   <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="Sport_category" placeholder="Enter Sport Category..." required>
                   </div>
                   <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="min_team_size" placeholder="Enter min team size..." required>
                   </div>
                   <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="max_team_size" placeholder="Enter max_team_size..." required>
                   </div>
                   <div class="form-group">
                     <input type="text" class="form-control form-control-
user" name="gender" placeholder="Enter gender..." required>
                   </div>
                   <center>
                   <div class="form-group ">
                  <button class="btn btn-primary " type="submit" value="submit" >Add Sport</button>
                 </div>
                  </center>
                 </div>
               </form>
               {%endblock%}
```

5. RESULTS

a. Testing in postman:

For this project,we have used jwt access tokens for security purpose. JWT Access Tokens provide a way to create and validate access tokens without requiring a central storage such as a database.



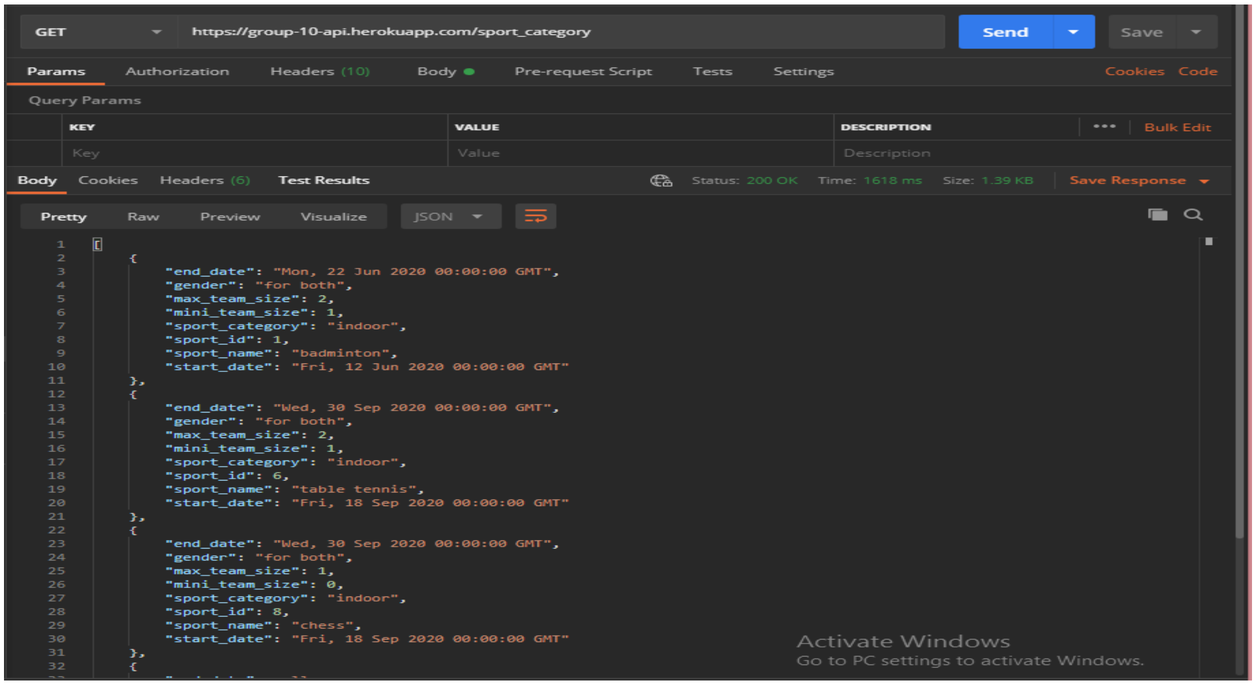
POST https://group-10-api.herokuapp.com/Adminlogin

```
{
    "username":"sportadmin",
    "password":"sport@123"
}
```

Status: 200 OK Time: 12.37 s Size: 455 B

```
{
    "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
        eyJpYXQiOjE1OTM0MzY4ODAsIm5iZiI6MTU5MzQzNjg4MCwianRpIjoiZWM4YzEyNGUtNDBmYS00ODU0LTljZjQtNzM0MDZlMGVhZmIzIiwiaWRlbn
        RpdHkiOiJzcG9ydGFkbWluIiwiZnJlc2giOmZhbHNlLCJ0eXBlIjoiYWNjZXNzIn0.mp2izI7rOxux33IBm9OvqJ1UTxef98ye7kYRGTYMB-Y"
}
```

Fig 5.1:Demonstration of admin login



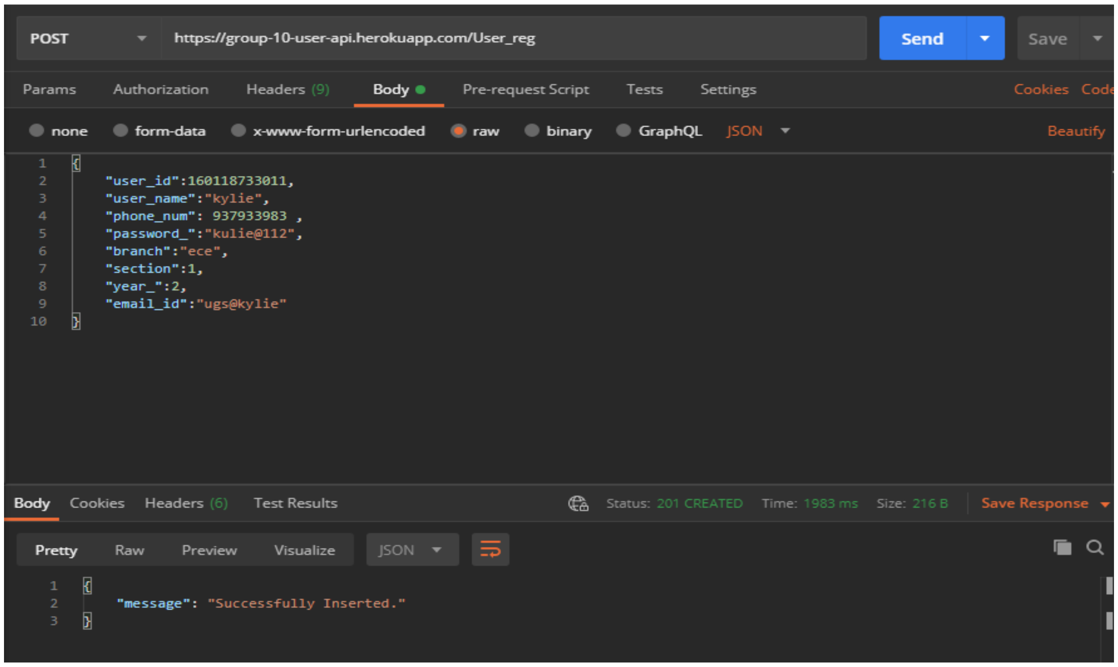
GET https://group-10-api.herokuapp.com/sport_category

Status: 200 OK Time: 1618 ms Size: 1.39 KB

```
[
    {
        "end_date": "Mon, 22 Jun 2020 00:00:00 GMT",
        "gender": "for both",
        "max_team_size": 2,
        "mini_team_size": 1,
        "sport_category": "indoor",
        "sport_id": 1,
        "sport_name": "badminton",
        "start_date": "Fri, 12 Jun 2020 00:00:00 GMT"
    },
    {
        "end_date": "Wed, 30 Sep 2020 00:00:00 GMT",
        "gender": "for both",
        "max_team_size": 2,
        "mini_team_size": 1,
        "sport_category": "indoor",
        "sport_id": 6,
        "sport_name": "table tennis",
        "start_date": "Fri, 18 Sep 2020 00:00:00 GMT"
    },
    {
        "end_date": "Wed, 30 Sep 2020 00:00:00 GMT",
        "gender": "for both",
        "max_team_size": 1,
        "mini_team_size": 0,
        "sport_category": "indoor",
        "sport_id": 8,
        "sport_name": "chess",
        "start_date": "Fri, 18 Sep 2020 00:00:00 GMT"
    },
    {
```

Fig 5.2:Viewing sports

Fig 5.2 depicts the get method for /sport_category end point which shows all the sports under that category

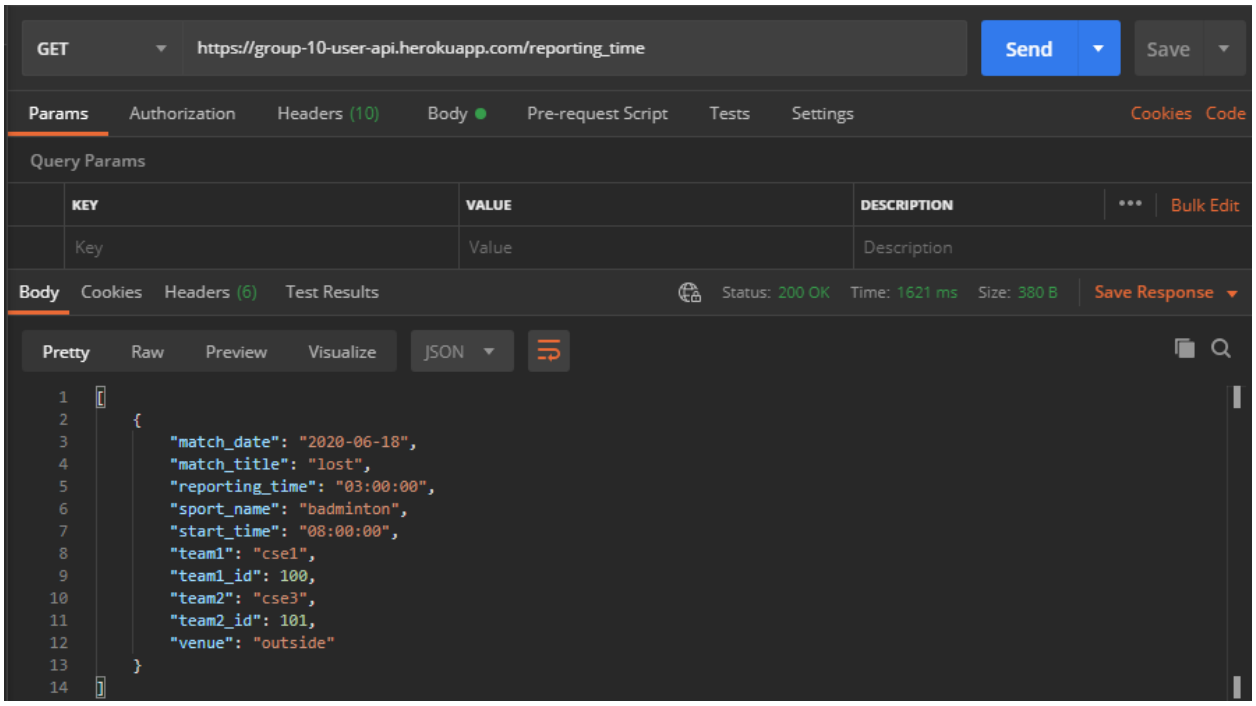


```
{
    "user_id":160118733011,
    "user_name":"kylie",
    "phone_num": 937933983 ,
    "password_":"kulie@112",
    "branch":"ece",
    "section":1,
    "year_":2,
    "email_id":"ugs@kylie"
}
```

```
{
    "message": "Successfully Inserted."
}
```

Fig 5.3: Demonstartion of user registration

Fig 5.3 depicts the post method for signing up of user using /User_reg endpoint



```
[
    {
        "match_date": "2020-06-18",
        "match_title": "lost",
        "reporting_time": "03:00:00",
        "sport_name": "badminton",
        "start_time": "08:00:00",
        "team1": "cse1",
        "team1_id": 100,
        "team2": "cse3",
        "team2_id": 101,
        "venue": "outside"
    }
]
```

Fig 5.4:Viewing reporting time

b. WEBSITE FOR ADMIN

The usage of the application is done by the users in the Android Application. But, when it comes to handling the input information and doing all behind the scenes work lies in the hand of Admin.
As a result, a website has been designed for the admins to care of the operations made with the application.

All the user input information and other such related activities provided in the user application is reflected in the website
This website is solely designed for the admin and a user cannot access this website.
The website can be thought of as a place where all the behind the scene activities take place.

Let us have a brief overview of the website including the concepts and technologies used inside it along with the operations.

Website Pages :

i. Login Page :

The login page is the first and foremost page which is visible to the admin once he opens the website.Login is done with the help of correct user admin ID and password. If the user ID and password are correct i.e. it matches with the details entered in the database earlier for the admin, the admin gets redirected to the main index page of the website. Fig 5.5 depicts the admin login page:

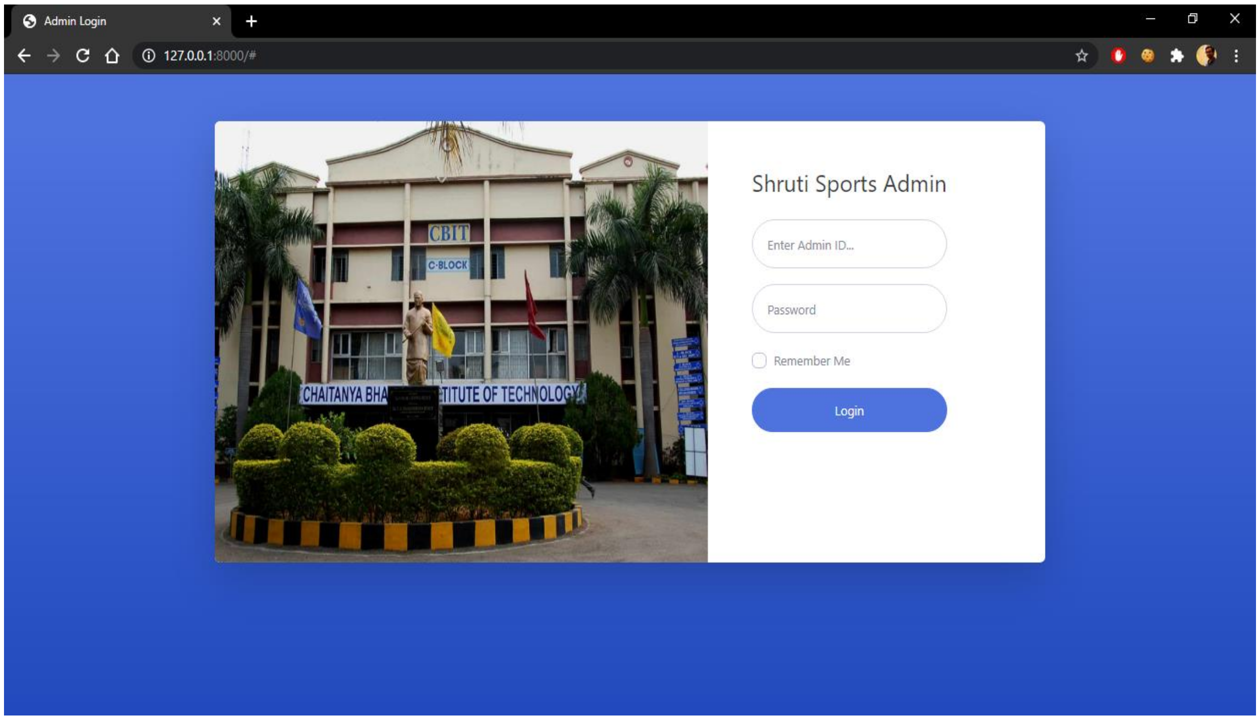


Fig 5.5: Admin’s login page

The concept used behind the login and accessing the website is JWT (JSON Web Tokens). If the information entered by the admin is authentic, the admin will be granted with a token with the help of which the admin can access the website as long as the life of the token lasts.
With the help of this feature, good security of website can be realized.

Once the user clicks the login button with correct information, the user is redirected to the main page of the website.

ii.Main Index Page :

This is the main index page of the website. Here all the functionality of the website is listed in proper form. The user can access any feature of the website by clicking the appropriate option provided to him/her.

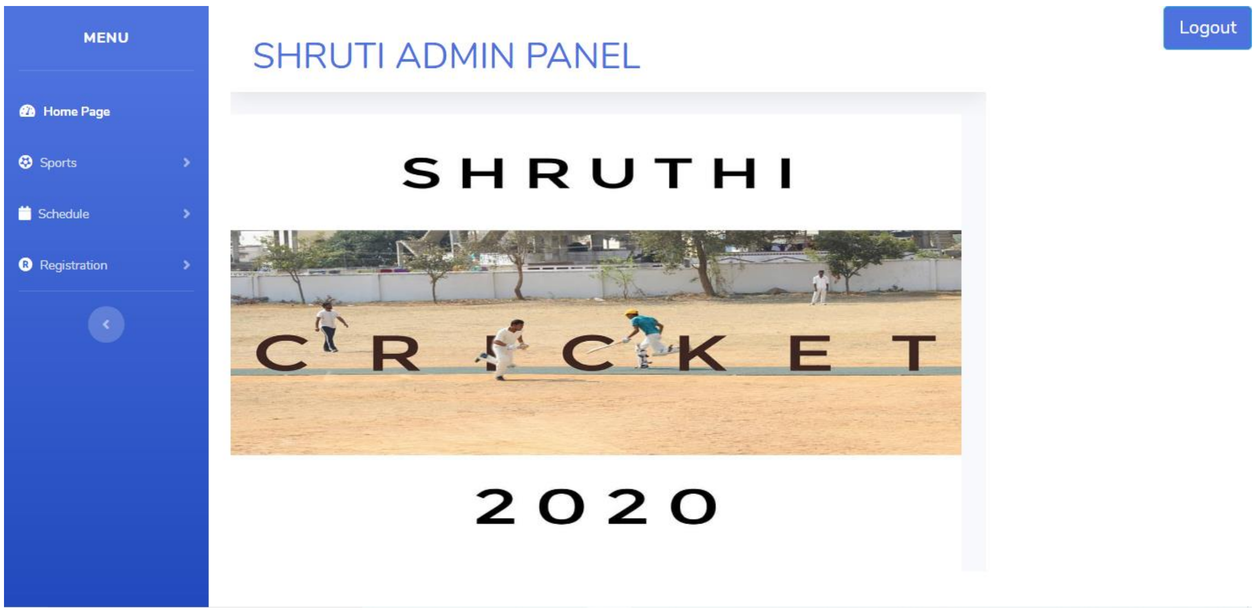


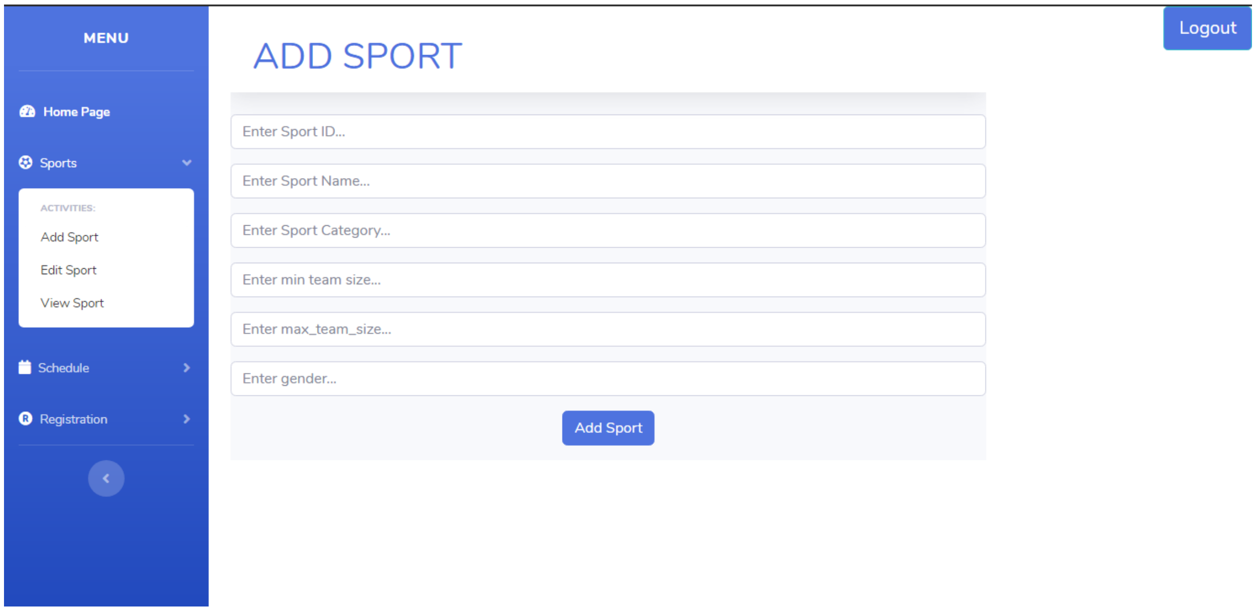
Fig 5.6 : Admin’s Dashboard

It represents the different categories in a side panel clicking on which the admins can get their work done.
A slideshow of pictures is also made available.
On top right corner is a button meant for logout. Once the admin clicks the button he will be again redirected to the login page.

As the problem statement suggests, all the required operations to be done by the admin are made available in this website. For each operation a separate page is designed. Let us have a look at all of them in brief.

iii. Sports Page :

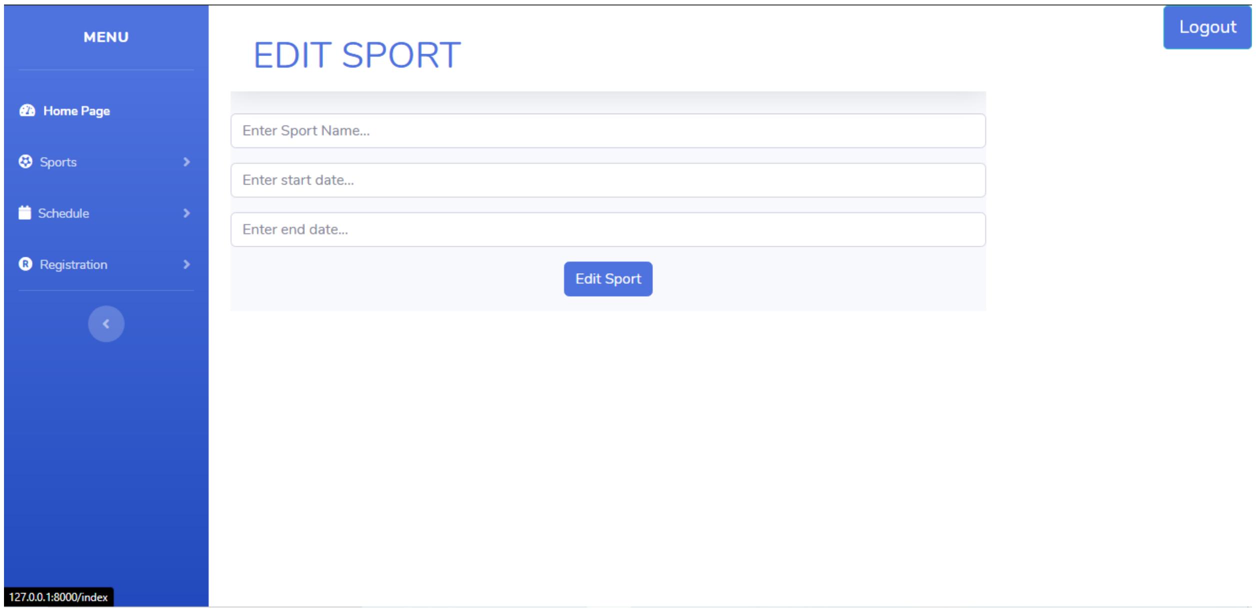
Among the categories include a sport category which has three operations inside it namely: Add sport, View sport and view sport.The admin can do the following additions and changes with the help of the features.



ADD SPORT

Enter Sport ID...
Enter Sport Name...
Enter Sport Category...
Enter min team size...
Enter max_team_size...
Enter gender...

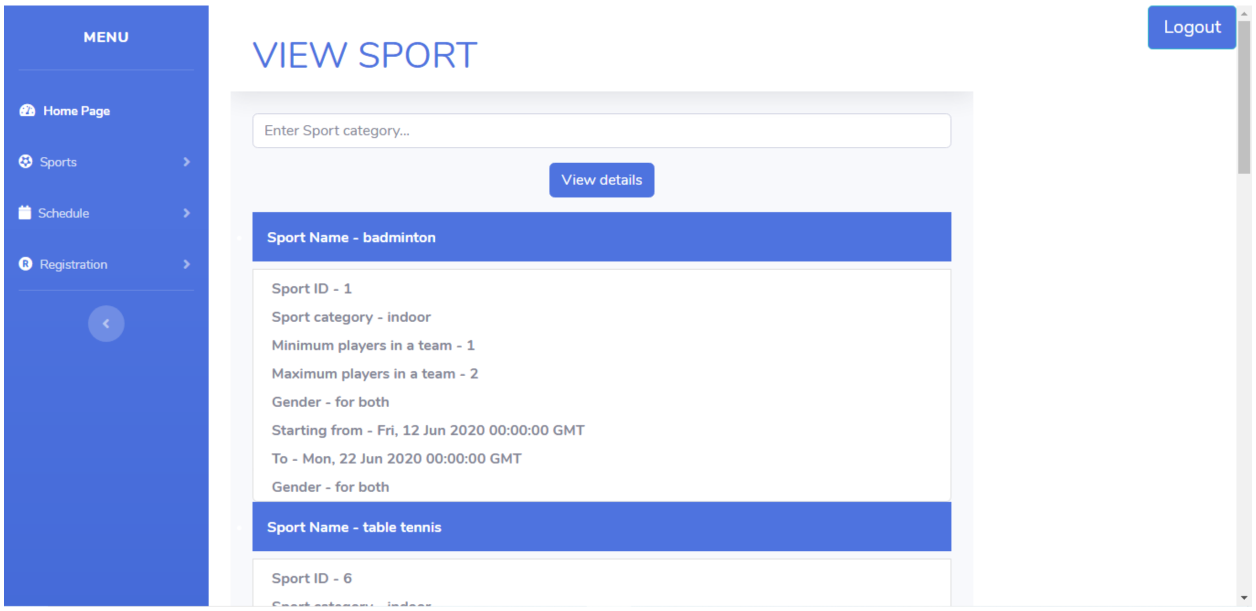
Fig 5.7: Add sports screen



EDIT SPORT

Enter Sport Name...
Enter start date...
Enter end date...

Fig 5.8 edit sports screen



Sport Name - badminton

Sport ID - 1
Sport category - indoor
Minimum players in a team - 1
Maximum players in a team - 2
Gender - for both
Starting from - Fri, 12 Jun 2020 00:00:00 GMT
To - Mon, 22 Jun 2020 00:00:00 GMT
Gender - for both

Sport Name - table tennis

Sport ID - 6

Fig 5.9 Viewing sports screen

As shown above the admin can carry out the three operations within the sports category.
All the changes and info given as the input will be reflected in the database as well.

iv .Schedule Page :

The schedule page is same like the Sport page when the operations are taken into account. Within the Schedule page, the admin can do some operations namely: Add schedule, View schedule and Edit schedule.
All the changes made here are reflected in the database as well.
A screen shot for the same is provided below showing the operations which are made available to the admin within the scheduling page:

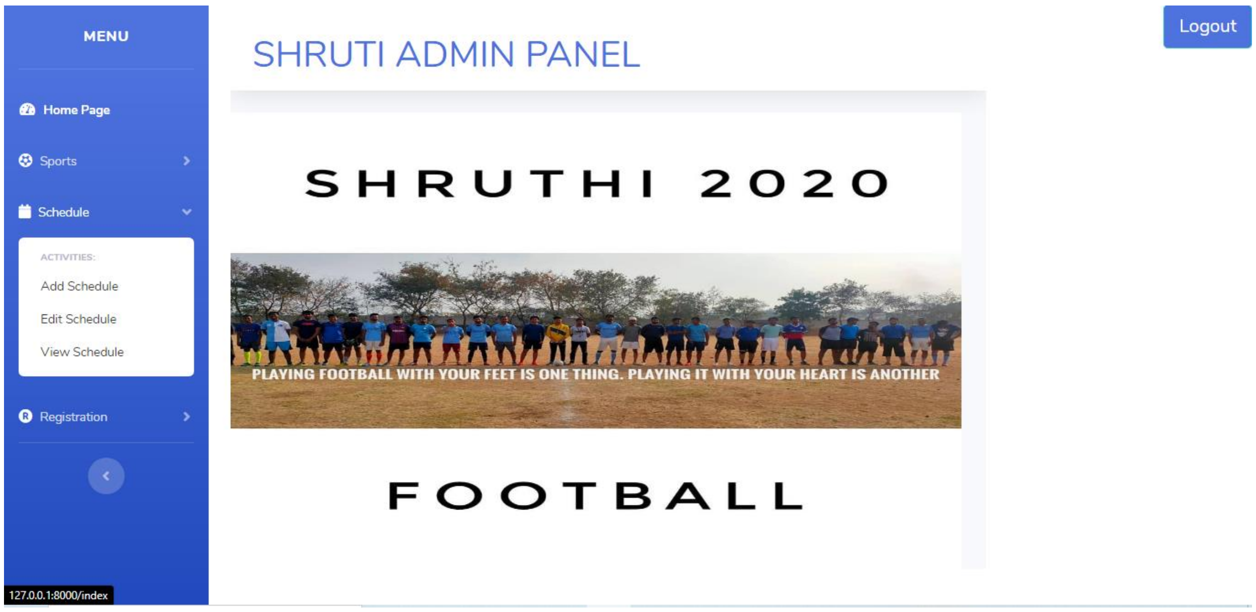
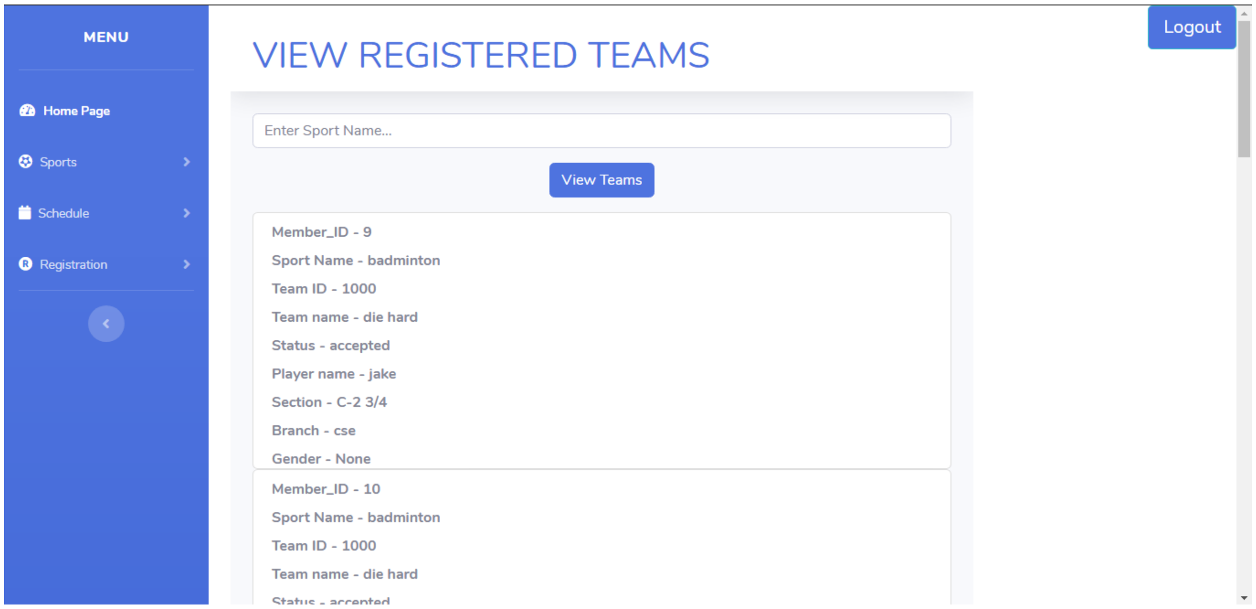


Fig 5.10: Schedule screen

v. Registration Page :

The registration page consists the part for where the admins can accept for the registrations which were made by the users in the android application.
Once these registrations are accepted by the admin, the user gets notified with a notification on their android device.

Some of the features which are included in the registrations page are as follows: View registered teams, Accept registrations and View team players.
Some screen shots depicting the same are displayed below :



Member_ID - 9
Sport Name - badminton
Team ID - 1000
Team name - die hard
Status - accepted
Player name - jake
Section - C-2 3/4
Branch - cse
Gender - None

Member_ID - 10
Sport Name - badminton
Team ID - 1000
Team name - die hard
Status - accepted

Fig 5.11 : Viewing registered teams screen

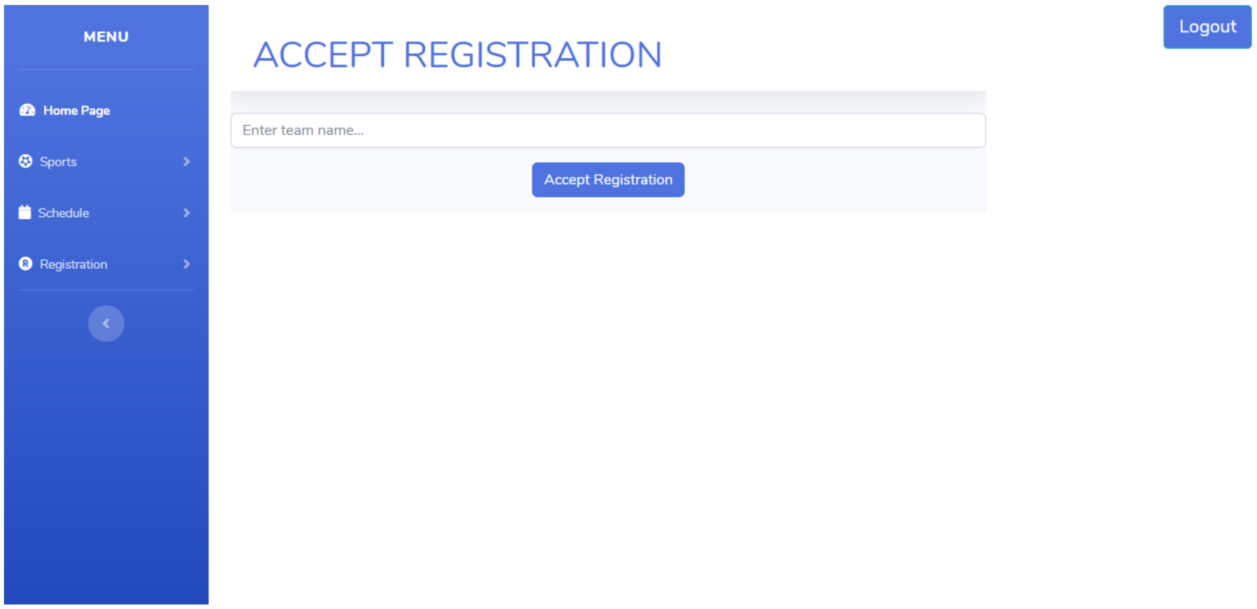


Fig 5.12 : Accepting registrations screen

Thus the admin can do all the registration related work and stuff under these registration pages.

Last but not the least, within the website and on every page is a logout button for the admin to logout anytime. Pressing on this button would redirect the admin to login page where he needs to login again if he wants to access the website.

This was all the front end for the admin.

c. Android Application for Captains/Users

The android application is mainly used by the captains and the users (team members / sports enthusiasts). The captains can perform various actions like registering their team, viewing their registration status, viewing team details and viewing their match schedules. The users on the other hand can view the sports details, when and where the matches are going to be held etc. All the information will be provided by the admin and verified by the admin using the admin portal and shown to the user. Let us see the application in more detailed view.

Sample Screens:

i. Splash Screen:

The Splash screen is first shown to the user when the app is launched, this is necessary for the app to work as all the permission will be check here and all the essential details will be loaded from the device memory like if the user selected remember me the previous time, his user ID and password will be loaded and also checks if he is a captain or not.

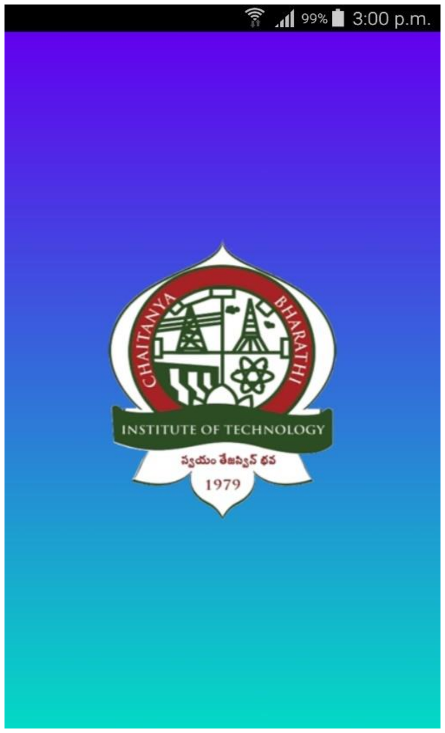


Fig 5.13 Splash screen

ii. Login screen:

In the login screen the user will enter his user ID and valid password which are verified by the server and an access token is returned for future uses and also the user ID is stored for use in the next screens. The user also has a captain checkbox if he wants to login as captain. There is also a remember me checkbox so that the application can store the user details and skip the login for next time.

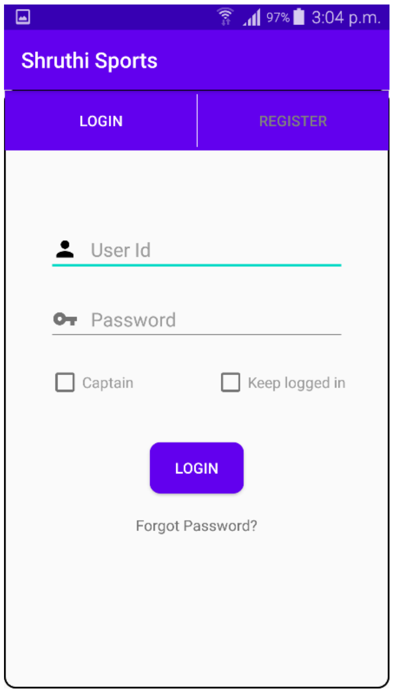


Fig 5.14: Login screen

iii. Register Screen:

In the register screen the user enters all his/her details to create their accounts. The entered information is validated at the user side by comparing the user entered details like name, ID, official email, branch etc. Only after validation of input details an OTP is sent to the users' official mail ID and by entering the correct OTP the registration will be successful.

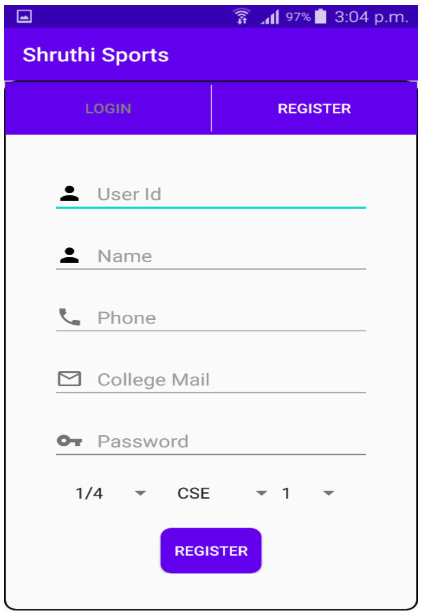
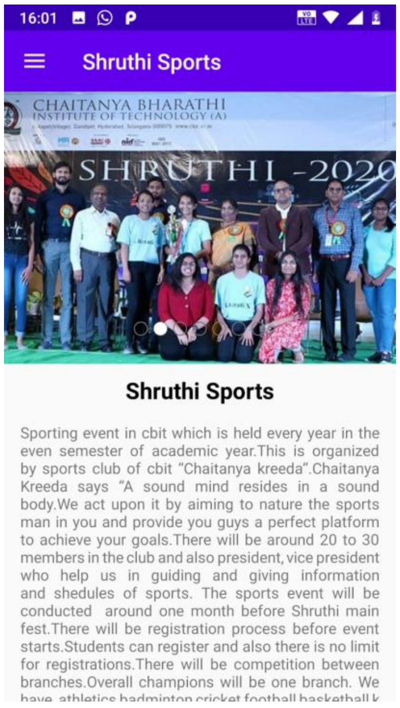


Fig 5.15 : Register screen

iv. **User Home Screen:**

Users' home screen is where the user is redirected once after he successfully logs in. In this screen the photos and details of previous years events will be displayed so that user can know more about the events before he participates and from this screen, he can access various other features of the application.



Shruthi Sports

Sporting event in cbit which is held every year in the even semester of academic year.This is organized by sports club of cbit "Chaitanya kreeda".Chaitanya Kreeda says "A sound mind resides in a sound body.We act upon it by aiming to nature the sports man in you and provide you guys a perfect platform to achieve your goals.There will be around 20 to 30 members in the club and also president, vice president who help us in guiding and giving information and shedules of sports. The sports event will be conducted around one month before Shruthi main fest.There will be registration process before event starts.Students can register and also there is no limit for registrations.There will be competition between branches.Overall champions will be one branch. We have athletics badminton cricket football basketball k

Fig 5.16 : user's home screen

v. User Navigation Drawer:

Navigation drawer is the widget available in every screen for the user to navigate from one screen to another. The user navigation drawer can redirect to all the features that user has access to.

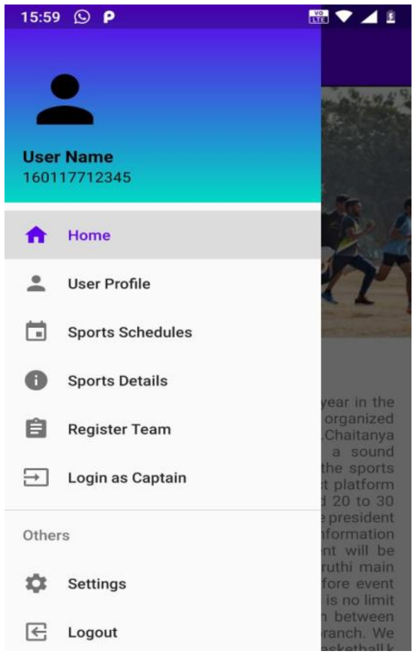
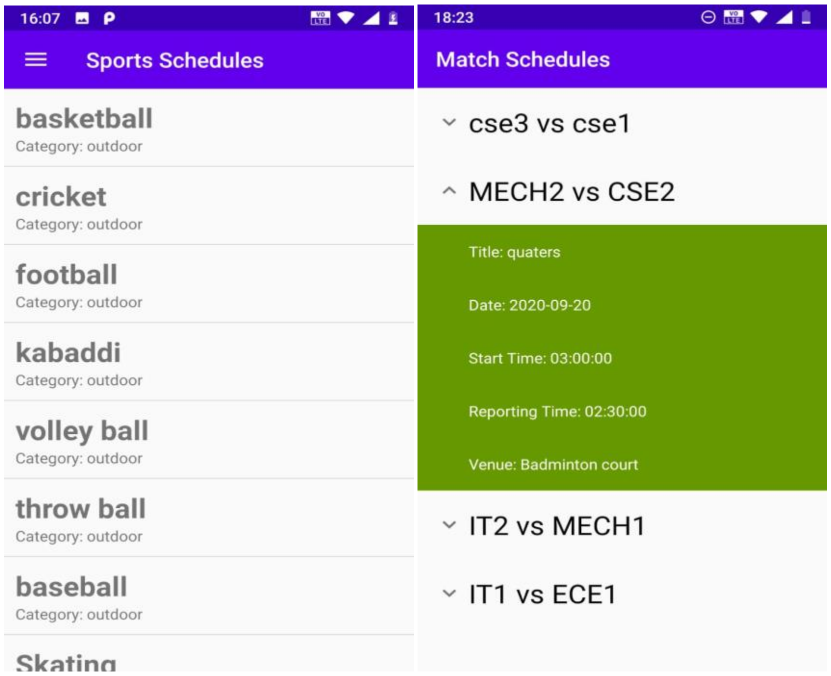


Fig 5.17 : User’s navigation drawer

vi. Sports Schedule Screen:

Sports Schedule Screen is the screen where a user can view the schedules for the matches of a sport by selecting a sport from list of sports. Once the user clicks on one of the sports, he is redirected to another page with all the matches related to that sport.


16:07

Sports Schedules

basketball
Category: outdoor

cricket
Category: outdoor

football
Category: outdoor

kabaddi
Category: outdoor

volley ball
Category: outdoor

throw ball
Category: outdoor

baseball
Category: outdoor

Skating

18:23

Match Schedules

cse3 vs cse1

MECH2 vs CSE2

Title: quaters

Date: 2020-09-20

Start Time: 03:00:00

Reporting Time: 02:30:00

Venue: Badminton court

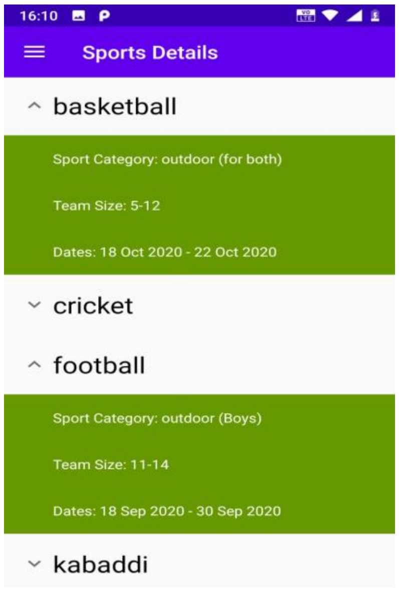
IT2 vs MECH1

IT1 vs ECE1

Fig 5.18: Sports schedule screen

vii. Sports Details Screen:

Sports Details Screen is the screen where a user can view various details of a sport like the category of the sport, minimum and maximum teams' size for that sport, start and end dates of the sport, venue for the sport etc.


16:10

Sports Details

basketball

Sport Category: outdoor (for both)

Team Size: 5-12

Dates: 18 Oct 2020 - 22 Oct 2020

cricket

football

Sport Category: outdoor (Boys)

Team Size: 11-14

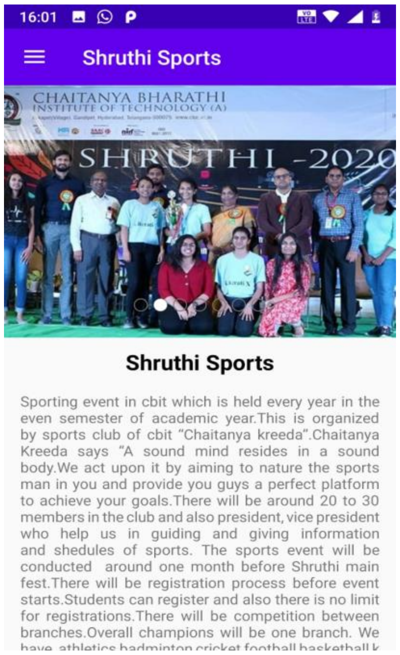
Dates: 18 Sep 2020 - 30 Sep 2020

kabaddi

Fig 5.19: Sport details screen

viii. Captain Home Screen:

Captains' home screen is where the captain is redirected once after he successfully logs in. In this screen the photos and details of previous years events will be displayed so that user can know more about the events before he participates and from this home screen, he can access various other features of the application.



Shruthi Sports

Sporting event in cbit which is held every year in the even semester of academic year.This is organized by sports club of cbit "Chaitanya kreeda".Chaitanya Kreeda says "A sound mind resides in a sound body.We act upon it by aiming to nature the sports man in you and provide you guys a perfect platform to achieve your goals.There will be around 20 to 30 members in the club and also president, vice president who help us in guiding and giving information and shedules of sports. The sports event will be conducted around one month before Shruthi main fest.There will be registration process before event starts.Students can register and also there is no limit for registrations.There will be competition between branches.Overall champions will be one branch. We have athletics badminton cricket football basketball

Fig 5.20: Captain's home screen

ix. Captain Navigation Drawer:

Navigation drawer is the widget available in every screen for the captain to navigate from one screen to another. The captain navigation drawer can redirect to all the features that captain has access to.

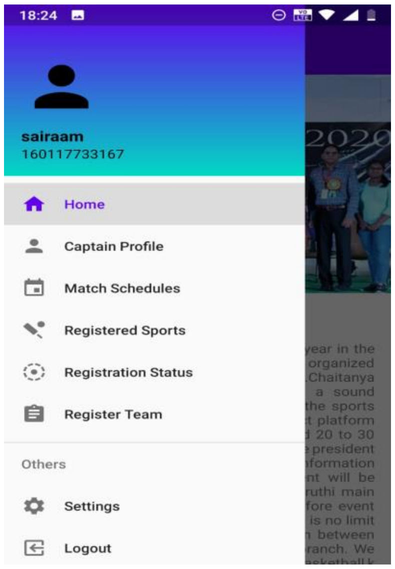


Fig 5.21: Captain's navigation drawer

x. Register Team Screen:

Register Teams Screen is the place where the captain can register a team to participate in the tournament. First the user must enter all the necessary information like the section, branch, sport name and size etc. then he must enter the details of the team members like their name and ID, once all these details are submitted they'll be checked by the server and added to the database with registration status as "NOT ACCEPTED YET" waiting for the admin's approval.

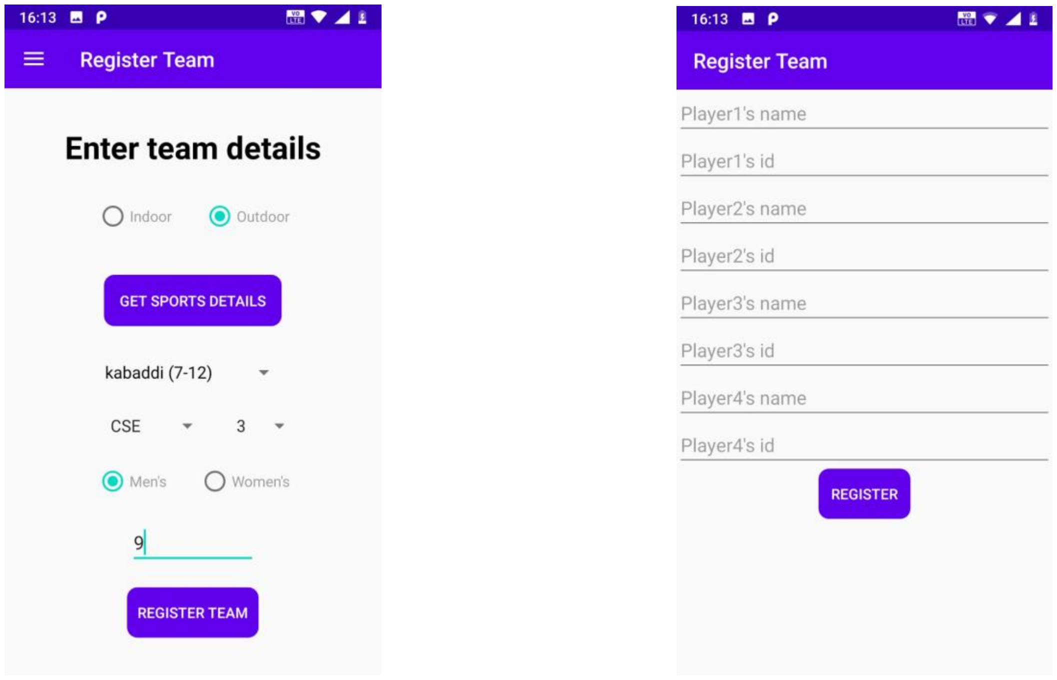


Fig 5.22: Register team screen

xi. Registered Sports Screen:

Registered Sports Screen is the screen where the captain can see the details of the sports, he has registered to along with the team details like team members and their IDs. Once the teams are accepted by the admin their details will be shown in this screen.

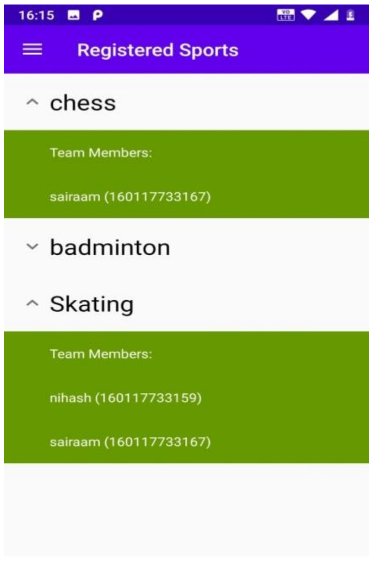


Fig 5.23: Registered sports screen

xii. Team Status Screen:

Team Status Screen is the screen where the captain can query for the registration status of the team he has registered to. Here the user should specify the sport name, based on his ID and name of sport the team status will be displayed if the team exists.

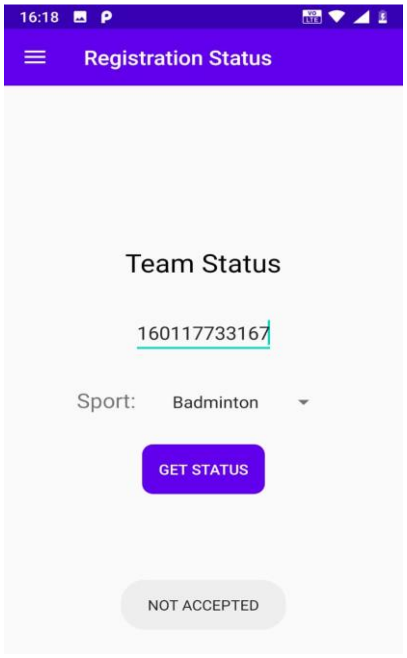
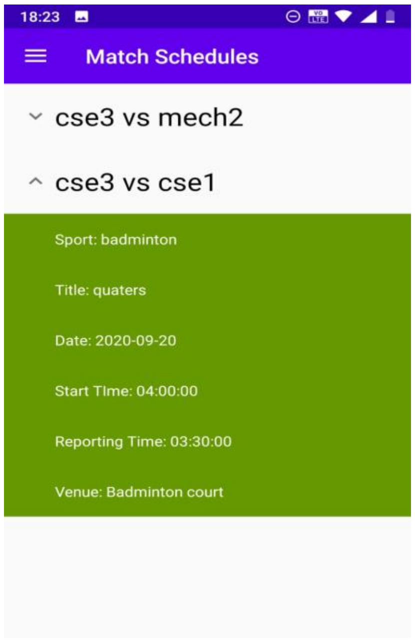


Fig 5.24: Team status screen

xiii. Match Schedules Screen:

Match Schedules Screen is the screen where all the captains' matches will be shown. In the background using the captains' ID all his registered teams are acquired and all their playing matches will be fetched and shown to the captain so that the captain can keep track of the matches the teams will be playing in the future.



Sport: badminton

Title: quaters

Date: 2020-09-20

Start TIme: 04:00:00

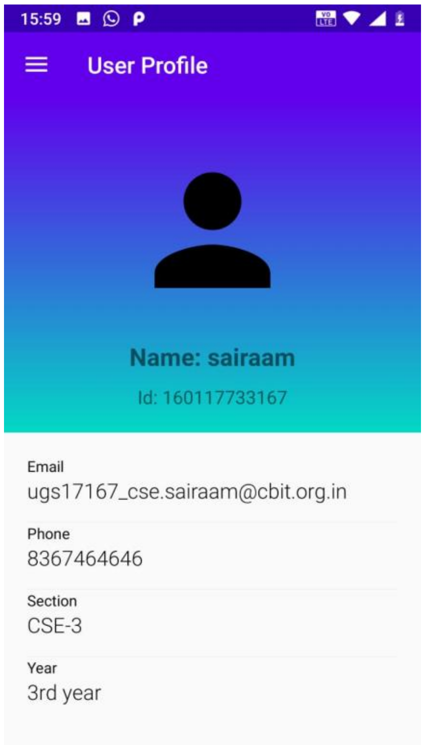
Reporting Time: 03:30:00

Venue: Badminton court

Fig 5.25: Schedule screen

xiv. Profile Screen:

Profile screen is where one can see their details whether a captain or a normal student all their details are displayed here.



Name: sairaam

Id: 160117733167

Email
ugs17167_cse.sairaam@cbit.org.in

Phone
8367464646

Section
CSE-3

Year
3rd year

Fig 5.26: Profile screen

xv. Change Password Screen:

Change password screen can be helpful when user wants to change his password for security reason every now and then. In this screen user can change his password easily.

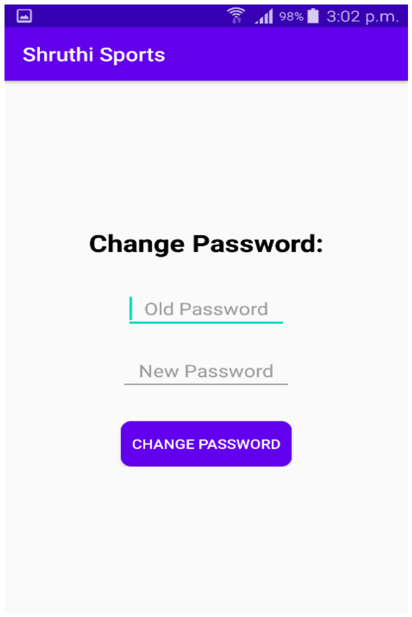


Fig 5.27: Change password screen

xvi. Logout Screen:

Logout screen is also as important as login screen because this is the place where all the user related information that is stored in the device should be erased so that no other person can access or make changes to the data.

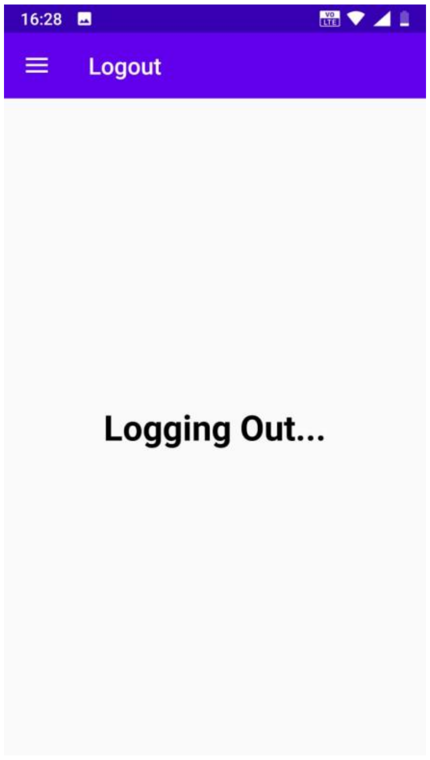


Fig 5.28: Logout screen

6. CONCLUSION

To conclude we have successfully created an android application where user can receive data regarding sports events, captains can register and manage team and a web application where the admin manage sports details, match schedules and registration to digitalize the whole process of registration and enquiry of Shruthi Sports. Using this application user, captains and also admins can effectively manage the events without wasting hours of time waiting and going from one end of college to another, they can manage everything through devices in their hands**.**

Future Scope

As applying for sports manually, every year, is a tedious task, this project helps the students as well as the faculty to handle the registrations smoothly. The application which is made for the students is very useful and handy as all we need is a smart phone and good internet connection .

BIBLIOGRAPHY

1) https://www.w3schools.com/
2) https://stackoverflow.com/
3) https://www.geeksforgeeks.org/
4) https://www.tutorialspoint.com/index.htm