

A  
Project Report  
on  
**ONLINE SPORTS TURF PLAYGROUND BOOKING SYSTEM**

Submitted for partial fulfilment of the requirements for the award of the degree of  
**BACHELOR OF TECHNOLOGY**  
in

**COMPUTER SCIENCE AND ENGINEERING**  
By

**P. Sai Kishore Reddy (19N81A0535)**  
**M. ShivaKethan (19N81A0539)**  
**D. Manikanth Reddy (19N81A0558)**

Under the guidance of  
**Mr T Shravan Kumar** M.TECH (CSE)

Assistant Professor  
Department of CSE



**B. Tech \* MBA**

**SPHOORTHY ENGINEERING COLLEGE**  
Department of Computer Science and Engineering  
(Affiliated to JNTUH & Recognized by AICTE)  
Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510  
Academic Year: 2022-23



**B. Tech \* MBA**

## **CERTIFICATE**

This is to certify that this Project Seminar Report entitled “**Online Sports Turf Playground Booking System**” is a bonafide work carried out by **P.SaiKishoreReddy (19N81A0535), M.ShivaKethan (19N81A0539), D.ManikanthReddy (19N81A0558)**, in partial fulfilment of the requirements for the award of degree of **Bachelor of Technology** in **Computer Science And Engineering** from **Sphoorthy Engineering College**, affiliated to **Jawaharlal Nehru Technological University Hyderabad, Hyderabad**, during the Academic Year 2022-23 under our guidance and supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

**Internal Guide**

Mr. T. Shravan Kumar  
Assistant Professor  
Department of CSE  
SPHN.

**Head of the department**

Mr. P. Rammohan Rao  
Head  
Department of CSE  
SPHN.

**External Examiner**

## **DECLARATION**

We the undersigned, declare that the mini project title “**ONLINE SPORTS TURF PLAYGROUND BOOKING SYSYEM**” carried out at “**SPHOORTHY ENGINEERING COLLEGE**” is original and is being submitted to the Department of **COMPUTER SCIENCE AND ENGINEERING**, Sphoorthy Engineering College, and Hyderabad towards partial fulfilment for the award of Bachelor of Technology.

We declare that the result embodied in the Mini Project work has not been submitted to any other University or Institute for the award of any Degree or Diploma.

**Date:**

**Place: Hyderabad**

**Mr.P.SAI KISHORE REDDY**  
**(19N81A0535)**

**Mr.M.SHIVAKETHAN**  
**(19N81A0539)**

**Mr.D.MANIKANTH REDDY**  
**(19N81A0558)**

## ACKNOWLEDGEMENT

We express our deep sense of gratitude to our project Co-Ordinator **Dr. T. Venkata Ramana** for his constant and continuous support throughout the project, we sincerely thank our Project Guide **Mr.T.Shravan Kumar**, Asst. Professor, Department of Computer Science & Engineering, Sphoorthy Engineering College, Jawaharlal Nehru Technological University Hyderabad (JNTUH), Hyderabad for his inspiring guidance, consistent encouragement, constructive criticism and helpful suggestions during the entire course of my research work.

We express our sincere thanks to **Mr. P. Ram Mohan Rao**, Associate Professor & Head of the Department, Department of Computer Science & Engineering, Sphoorthy Engineering College, Nadargul (V), Balapur (M), Rangareddy (D) for his encouragement which helped me to complete my Project work.

We deem it a great privilege to express our profound gratitude and sincere thanks to **Mr. S. Chalama Reddy**, Chairman, **Mr. S. Jagan Mohan Reddy**, Secretary, **Prof. J.B.V. Subrahmanyam**, Principal, **Prof.M.V.S. Ram Prasad**, Director, Sphoorthy Engineering College, Nadargul (V), Balapur (M), Rangareddy (D), for their moral support and help in the completion of my research work.

We express our heartfelt thanks to Professors, Associate Professors, Assistant Professor and other professional non-teaching staff of Department of Computer Science and Engineering, Sphoorthy Engineering College, Nadargul (V), Balapur (M), Rangareddy (D) for providing the necessary information pertaining to our project work.

BATCH: 2019 - 2023

## **ABSTRACT**

Turf Playground is used to play various sports like football, tennis, cricket, etc. Many school teams and clubs prefer turf playgrounds for practice and training purposes. Sometimes it becomes difficult to book turf playgrounds because of timing issues or the slot getting booked previously.

This sports ground booking website is proposed for booking the turf easily and efficiently. It has three modules namely, Admin, Manager, and User. Admin can log in and can add turf locations, assign manager by creating login credentials for the manager, add price details for the particular turf, manages turf and view the details of sports venues booking for all locations.

Online Sports Turf Playground Booking System sends message reminders to managers and users whenever slots are booked, canceled or rescheduled. And your users can easily and securely authenticate themselves by linking their existing service by using a password.

**P.SAI KISHORE REDDY (19N81A0535)**

**M.SHIVAKETHAN (19N81A0539)**

**D.MANIKANTH REDDY (19N81A0558)**

# INDEX

Contents		Page No.
Abstract		V
Index		VI
List of Figures		VII
<b>Chapters</b>		
<b>1.</b>	<b>Introduction</b>	1
1.1.	Problem Statement	1
1.2.	Objective	1
1.3.	Motivation	2
1.4.	Existing System	2
1.5.	Proposed System	3
1.6.	Scope	3
1.7.	Software & Hardware Requirements	4
<b>2.</b>	<b>Literature Survey</b>	5
2.1.	Survey of Major Area Relevant to Project	5
2.2.	Techniques and Algorithms	5-6
2.3.	Applications	7
<b>3.</b>	<b>System Design</b>	8
3.1.	System Architecture	8
3.2.	System Flow	9-12
3.3.	Module Description	13-14
<b>4.</b>	<b>Implementation</b>	15
4.1.	Environmental Setup	15
4.2.	Implementation of Modules	16-19
4.3.	Integration and Development	20
<b>5.</b>	<b>Evaluation</b>	21
5.1.	Datasets	21-24
5.2.	Evaluation Metrics	25-27
5.3.	Tests Cases	28
5.4.	Results	29-30
<b>6.</b>	<b>Conclusion and Future Enhancement</b>	31
	<b>References</b>	32
	<b>Appendix</b>	33
A.	Sample Code	34-47

## LIST OF FIGURES

<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
3.1	System Architecture	8
3.2.1	Use Case Diagram	10
3.2.2	Admin Sequence Diagram	11
3.2.3	User Sequence Diagram	12
3.2.4	Manager Sequence Diagram	12
4.2.1	Login	16
4.2.2	Admin	17
4.2.3	Manager	18
4.2.4	User	19
4.3	Integration and Development	20
5.1.1	Turf Registration	21
5.1.2	Admin Login	21
5.1.3	Adding Turfs	22
5.1.4	Available Turfs	22
5.1.5	Manager Login	22
5.1.6	User Registration	23
5.1.7	User Login	23
5.1.8	Book Turf	23
5.1.9	Booking Confirmation	24
5.1.10	Booking History	24
5.1.11	Support	24
5.2.1	Turf Locations	25
5.2.2	Turf Registrations	26
5.2.3	Age Group	27
5.4.1	Day Wise Bookings	29
5.4.2	Annual Report	30

# CHAPTER-1

## INTRODUCTION

### 1.1 PROBLEM STATEMENT

TurfPlayground is used to play various sports like football, tennis, cricket etc. Many school teams and clubs prefer turf playgrounds for practice and training purposes. Sometimes it becomes difficult to book turf playgrounds because of timing issues or the slot getting booked previously.

### 1.2 OBJECTIVE

It has three modules namely Admin, Manager and User.

**Admin** can login and can add turf locations, assign manager by creating login credentials, add price details for the particular turf, manages turf and view the details of sports venues booking for all locations.

**Managers** assigned by the Admin are different for various Turf playground locations. Managers will get login credentials from admin and then managers can login using their credentials, check the rates, view the request of turf booking for the respective location, accept bookings, generate bill and view the booking history.

**Users** can check the availability of the turf, select timings, fill personal details, can pay using bank details or card details and view previous turf playground booking history.



### **1.3 MOTIVATION**

This sports turf playground booking website is proposed for booking the turf Easily and Efficiently. Online Sports Turf Playground Booking System website is designed for booking the turf of various sports like Football, Rugby, Tennis, Cricket, Basketball etc in an easy and efficient manner. This Sports Turf Playground Booking System is carefully designed with project requirement specification which describes the design functionalites and constraints

Online Sports Turf Playground Booking System Sends message reminders to managers and users whenever slots are booked, canceled or rescheduled. Admin can log in and can add turf locations, assign manager by creating login credentials for the manager, add price details for the particular turf, manages turf and view the details of sports venues booking for all locations and users can securely authenticate themselves by linking their existing service by using a password.

### **1.4 EXISTING SYSTEM**

- The existing turf playground booking system is Offline Based.
- Booking a turf playground in offline is Exhausting and Time-consuming.
- Sometimes Booking a turf offline can be a nightmare due to the Queues and Overcrowding.
- Discounts and Offers are given only for a short period of time in offline booking system.

## **1.5 PROPOSED SYSTEM**

- Our online sports Turf playground booking website provides a streamlined booking and reservation procedure that reduce long queues and overcrowding.
- Online Booking System is at your service 24/7.
- User have access to book wide range of Turf locations based on availability.
- Our online system handles instant payments for booking turfs through Net-Banking, Credit cards, Debit cards or UPI.
- Online Booking System also sends Message reminders to managers and users whenever slots are booked, canceled or rescheduled.
- User also get exclusive Offers and Deals in online Booking System.

## **1.6 SCOPE**

Upgrading to an online system can have many advantages to help you stand out from competitors as well as proving a hassle-free booking process to customers.

Users can easily and securely authenticate themselves by linking their existing service by using a password.

Manager can view booking requests and confirm the turf booking which user requested, later he will generate the bills.

Admin provides login credentials to manager and can also add turf locations based on their availability.

## **1.7 SOFTWARE REQUIREMENTS**

- WINDOWS 7 or higher
- PYTHON
- FLASK Framework
- SQL Server
- Any Text Editor

## **1.8 HARDWARE REQUIREMENTS**

- PROCESSOR – Core i3
- HARD DISK – 64 GB
- MEMORY – 1 GB RAM
- INTERNET – 10 MBPS

## **CHAPTER-2**

### **LITERATURE SURVEY**

#### **2.1 SURVEY OF MAJOR AREA RELEVANT TO PROJECT**

The internet has dramatically changed the way we do business. As we move processes away from standard pen and paper and into the digital world, the integration of an Online booking platform is a necessity for any sports and leisure provider. Customers are no longer waiting in line to secure a booking and are instead using the internet in the convenience of their own home. Upgrading to an online system can have many advantages to help you stand out from competitors as well as proving a hassle-free booking process to your customers.

#### **2.2 TECHNIQUES AND ALGORITHMS**

The following techniques provides Reliability and Security for using our Turf Playground Booking System.

- Offering online Payment Options.
- Offering Deals and Discounts.
- Using Social Media platforms effectively.
- Running Email and SMS marketing campaigns.
- Giving the best experience to our customers.
- Sending Message Reminders.

The Algorithms used to develop our Online Sports Turf Playground Booking System website in which the users can search for different varieties of Turf Playgrounds and are able to sort them according to the price, type of sport, turf availability or the turf location in the city.

### **DIJKSTRA ALGORITHM**

Dijkstra Algorithm is used for design and implementation of a framework to find nearest Turf Locations in our Playground booking system. This algorithm helps explore turf playgrounds and book slots during peak times.

### **BREADTH FIRST SEARCH ALGORITHM**

Breadth first search is a traversal algorithm that starts traversing the slots from the turf playgrounds and explores all the neighboring turfs. Then, it selects the nearest turf and explores all the unexplored turfs.

### **QUICKSORT ALGORITHM**

Quicksort algorithm is used for information searching and it is the fastest algorithm that is widely used as a better way of searching. It is used for all the searching aspects in the booking system. Quicksort is a cache-friendly algorithm as it has a good locality of reference when used for searching.

## **2.3 APPLICATIONS**

Online Sports Turf Playground Booking System provides applications such as

- Cloud Services
- Real-time Booking
- User friendly Interface
- Customer Support
- Secure Payment
- Business Marketing

## CHAPTER-3

### SYSTEM DESIGN

#### 3.1 SYSTEM ARCHITECTURE

- System Architecture is the process of designing the architecture, components, and interfaces for a system so that it meets the end-user requirements.
- The goal of system architecture is to allocate the requirements of a large system to hardware and software components.

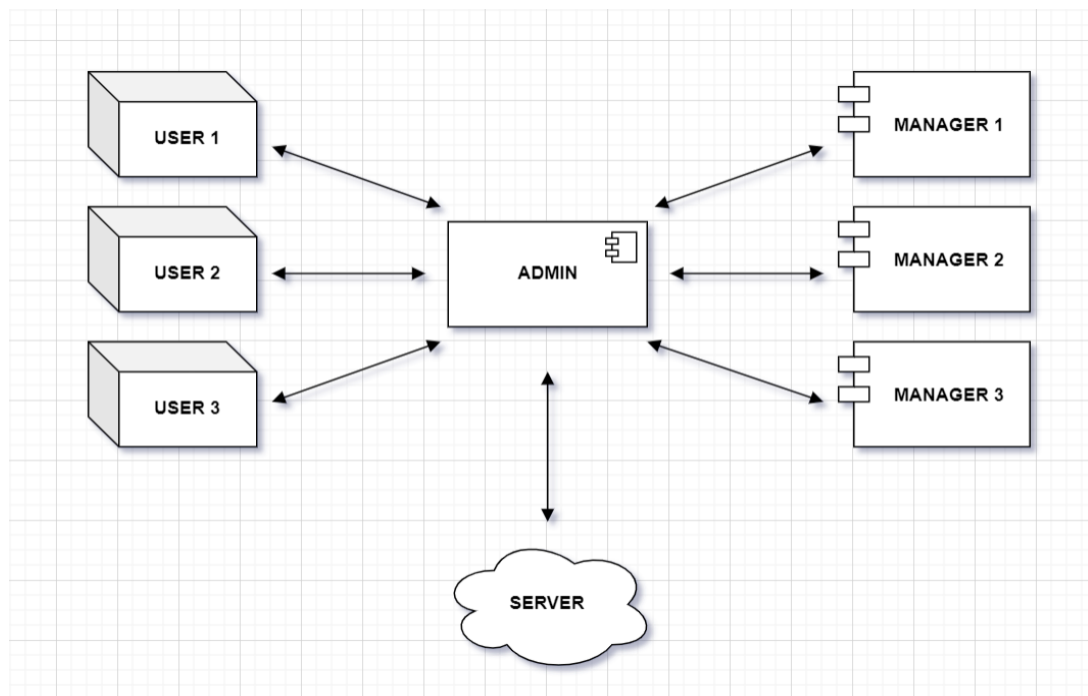


Fig.3.1

## **3.2 SYSTEM FLOW**

System Flows are system models that show the activities and decisions that system execute. They are useful for understanding complex system interactions because they visually show the back-and-forth interactions between systems and complex branching.

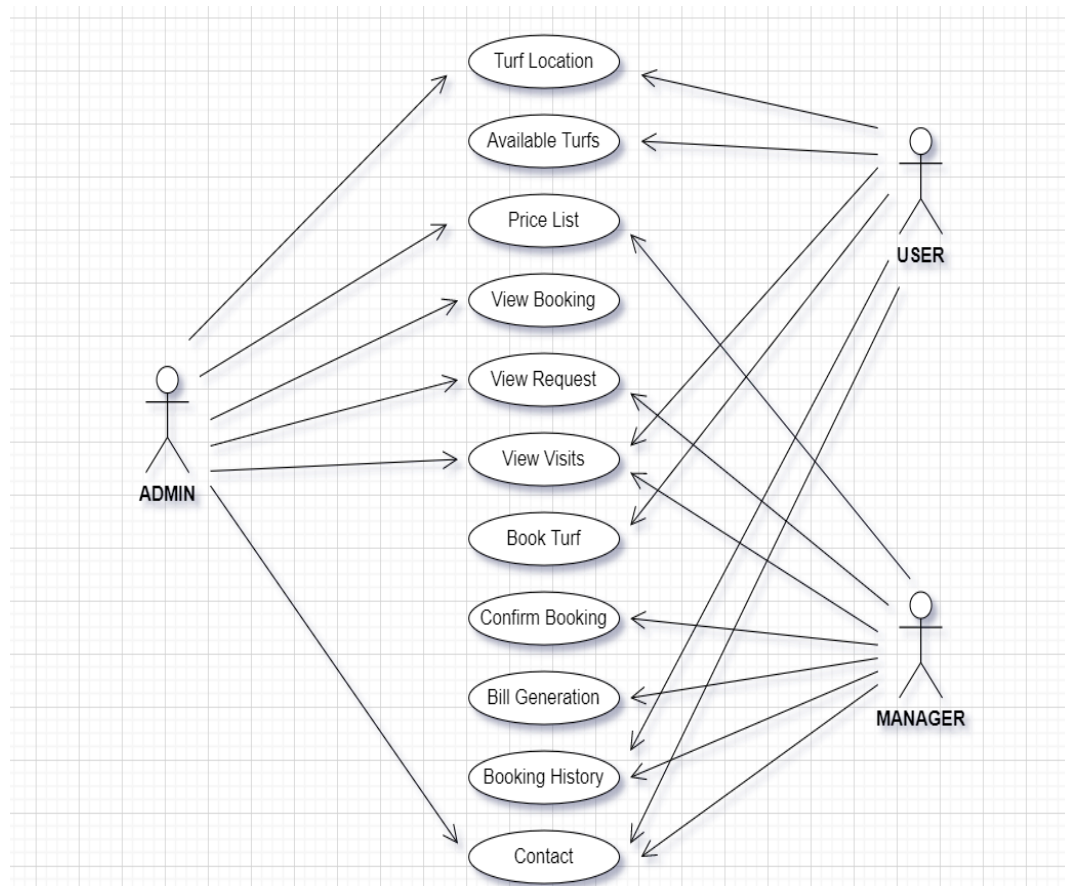
### **A) USE CASE**

A use case diagram is a list of actions or event steps typically defining the interactions between a actor and a system to achieve a goal. Use Case is shown as an ellipse containing the name of use case. An Actor is shown as a stick figure with the name below it. Use Case diagram is a graph of actors.

- The Actors in Use Case diagram are Admin, Manager and User.
- Actors use arrows to denote Use Cases of the project.
- Turf Location, Price List, View Booking, View Request, View Visits and Contact are the use cases of Admin.
- Turf Location, Available Turf, View Visits, Book Turf, Booking History and Contact are the use cases of User.
- Price List, View Requests, View Visits, Confirm Booking, Bill Generation, Booking History and Contact are the use cases of Manager.
- Turf Location use case shows the turf locations of various sports like Football, Rugby, Tennis, Cricket, Basketball etc.
- Price List use case lists the prices of different turf playgrounds.
- View Booking use case shows the Bookings of the turf playgrounds of various sports.
- Book Turf use case is used to book the required turf playgrounds by the user.



- Bill Generation use case is used by the manager to generate the bill receipts of the booked Turf Playgrounds.



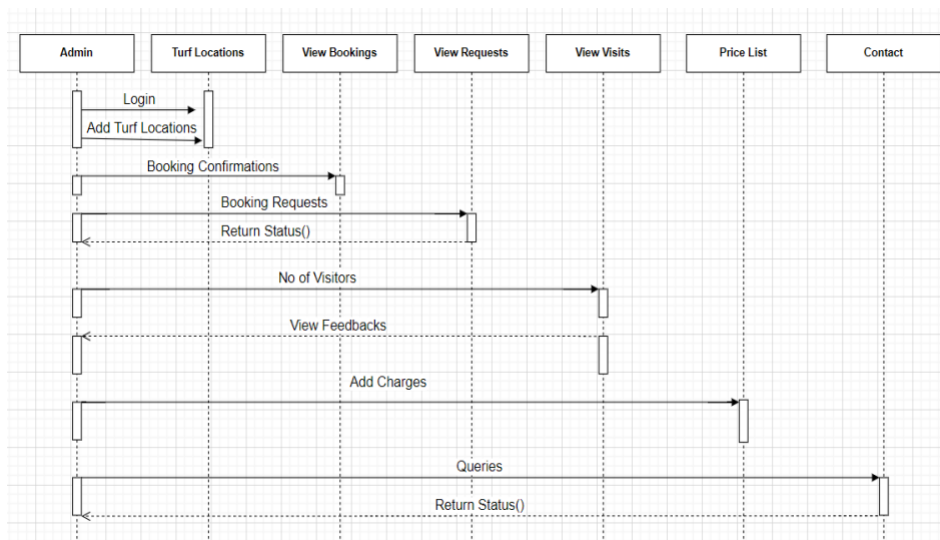
**Fig.3.2.1**

## **B) SEQUENCE DIAGRAM**

- Sequence Diagrams are used for the documentation of various system's requirements and to flush out a system's design.
- Sequence diagram is so useful because it shows the interaction logic between the objects in the system in the time order at which interactions take place.

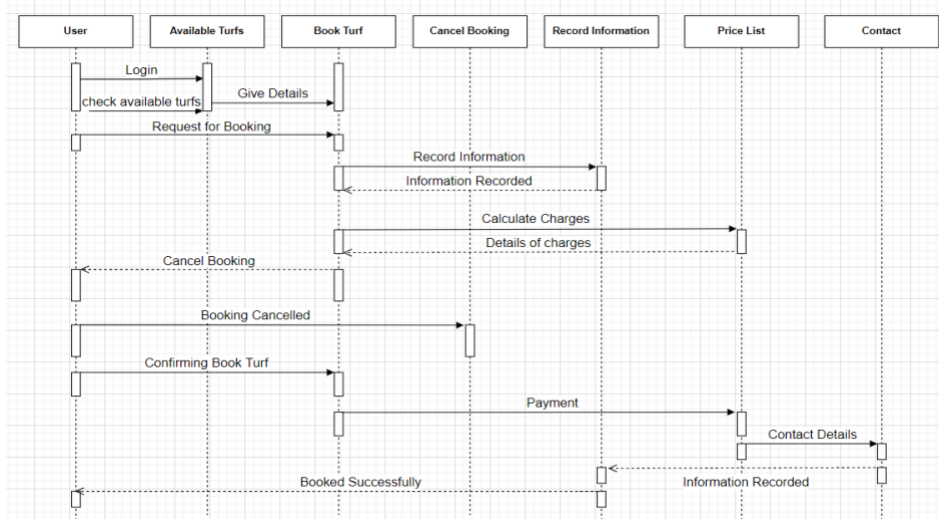
- A sequence diagram shows the sequence of messages passed between objects and also show the control structures between objects.
- These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.
- Admin provides login credentials to manager and can also add turf locations based on their availability.
- Users can easily and securely authenticate themselves by linking their existing service by using a password.
- Manager can view booking requests and confirm the turf booking which user requested, later he will generate the bills.
- Users have 24/7 Customer support in case of any queries.

### **ADMIN**



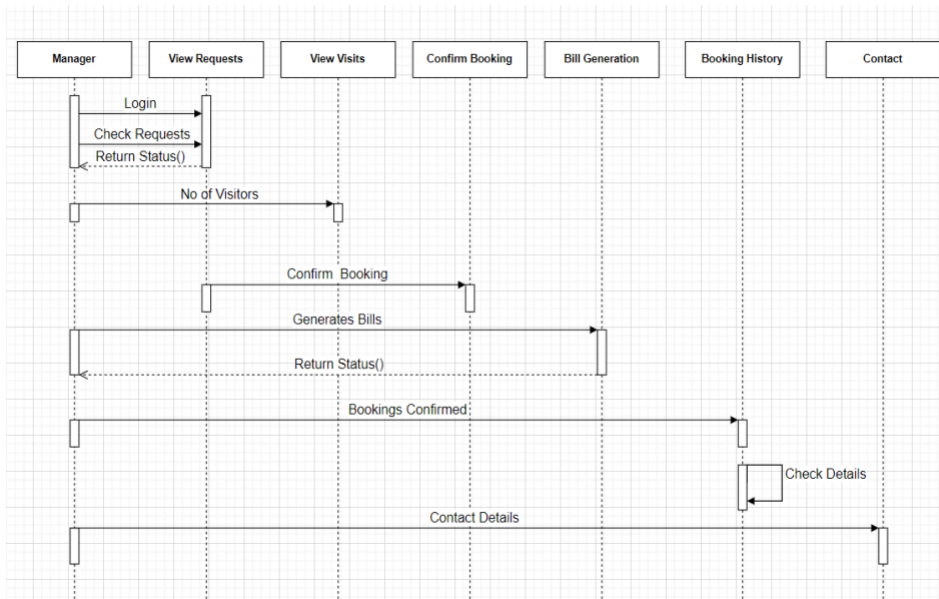
**Fig.3.2.2**

## USER



**Fig.3.2.3**

## MANAGER



**Fig.3.2.4**

### 3.3 MODULE DESCRIPTION

#### ADMIN

Admin can login and can add turf locations, assign manager by creating login credentials, add price details for the particular turf, manages turf and view the details of sports venues booking for all locations.

- **Add Manager:** Admin can add manager of the respective turf location.
- **Add Location:** Admin can add various turf locations based on their availability.
- **Add Price List:** Admin can add price for the respective turfs.
- **Manage Turf:** Admin can manage turf by allocating turf
- **View Booking:** Admin can view booking done and the user details.

#### MANAGER

Managers assigned by the Admin are different for various Turf playground locations. Managers will get login credentials from admin and then managers can login using their credentials, check the rates, view the request of turf booking for the respective location, accept bookings, generate bill and view the booking history.

- **Login:** The manager can log in with the credentials provided by the Admin.
- **Check Rates:** The manager can check rates for the respective location turf.
- **View Request:** The manager can view the request for turf bookings.
- **Confirm Booking:** Manager can confirm the booking of the turf.

- **Bill Generation:** The manager can generate bills as per the rates.
- **Bookings History:** Manager can check previous booking history.

## **USER**

Users can check the availability of the turf, select timings, fill personal details, can pay using bank details or card details and view previous turf playground booking history.

- **Check Turf:** Users can check for turf of nearby locations and prices.
- **Check Availability:** User can see the availability of the respective turf.
- **Book Turf:**User can provide the personal details and do the payment.
- **Bookings History:** User can check booking history.

## **CHAPTER 4**

### **IMPLEMENTATION**

#### **4.1 ENVIRONMENTAL SETUP**

##### **A) VISUAL STUDIO CODE**

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs such as Visual Studio IDE..

Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software.

Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.

##### **B) FLASK**

Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file.

## 4.2 IMPLEMENTATION OF MODULES

### A) LOGIN

The login page allows Admin, Manager and User to gain access to website by entering their username and password or by authenticating using a social media login credentials.

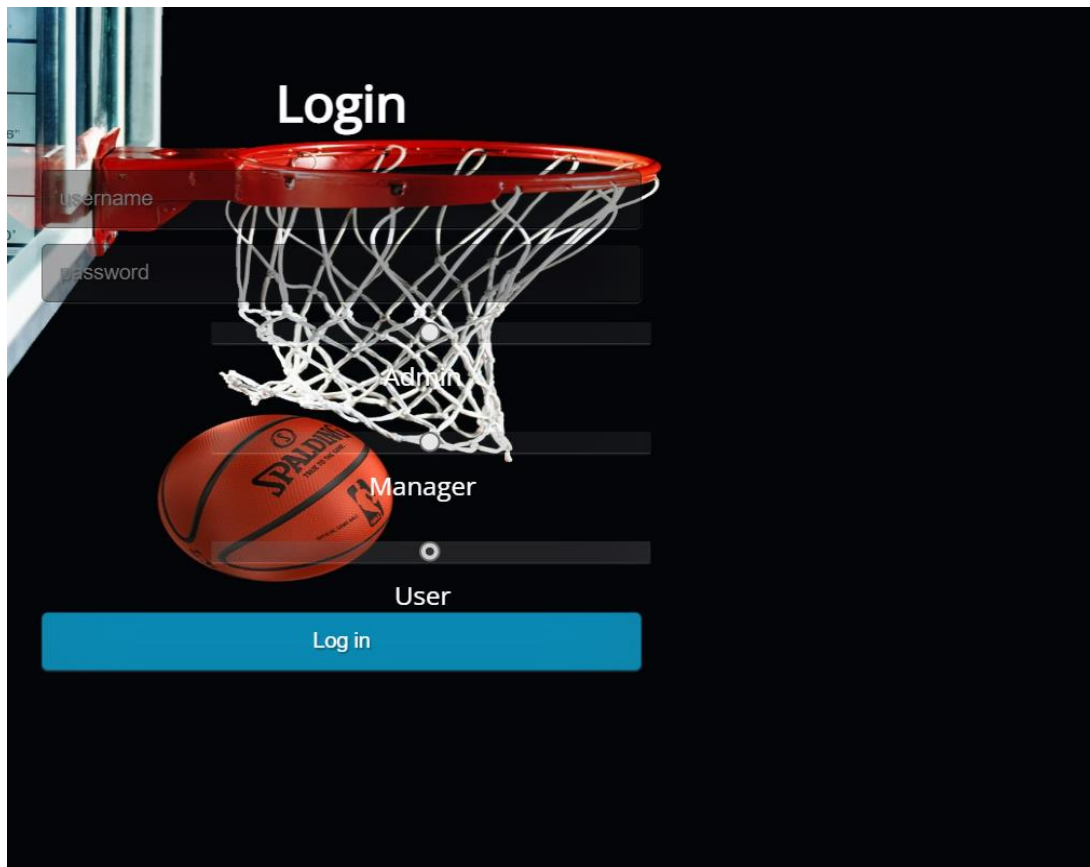


Fig.4.2.1

## B) ADMIN

Admin can login and can add turf locations, assign manager by creating login credentials, add price details for the particular turf, manages turf and view the details of sports venues booking for all locations.

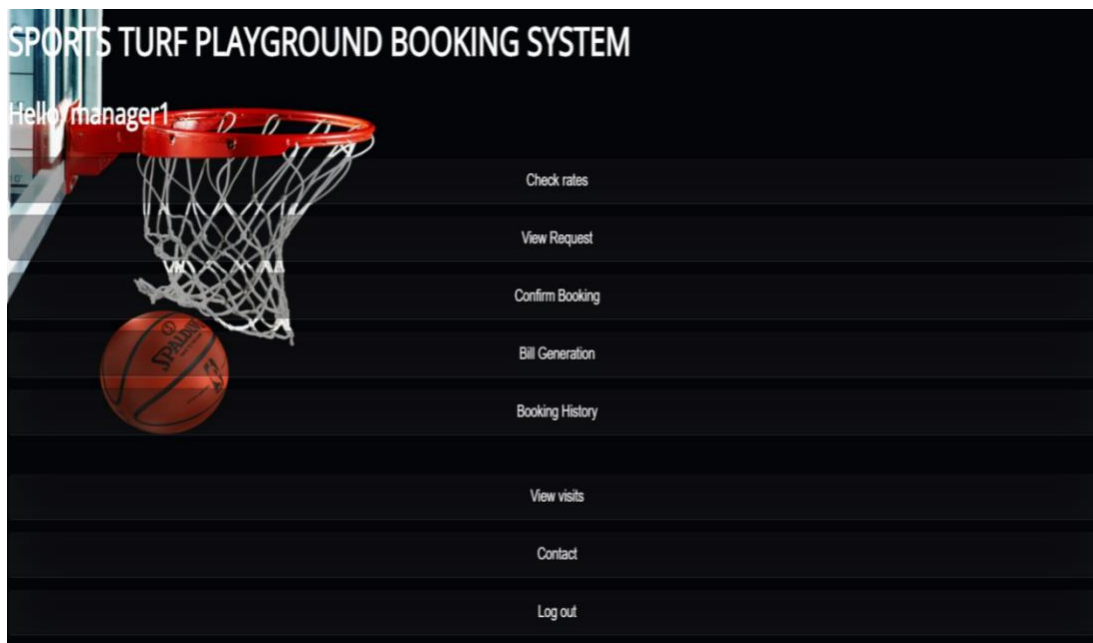


**Fig.4.2.2**



### **C) MANAGER**

Managers assigned by the Admin are different for various Turf playground locations. Managers will get login credentials from admin and then managers can login using their credentials, check the rates, view the request of turf booking for the respective location, accept bookings, generate bill and view the booking history.



**Fig.4.2.3**

## D) USER

Users can check the availability of the turf, select timings, fill personal details, can pay using bank details or card details and view previous turf playground booking history.



**Fig.4.2.4**

## 4.3 INTEGRATION AND DEVELOPMENT

Integration and Development services support the implementation and rollout of new network infrastructure, including consolidation of established network infrastructure. Activities may include hardware or software procurement, staging configuration, tuning, installation and interoperability testing.

The screenshot shows a web browser window titled "Turf Playground" with the address bar displaying "https://Turf Playground/Booking Details/Receipt". The page content is as follows:


TURF PLAYGROUND	
Payment Receipt	
<b>Booking Details</b>	
📄	Booking ID : TPB99235976
⚽	Sport : Box Cricket
🌿	Turf Name : Synthetic Turf 1
📅	Date : 2 , July 2022
🕒	TIME : 7:00 PM to 8:00 PM
💰	PRICE : INR 550.00
₹	PRICE : INR 22.00
₹	<b>Total Amount</b>
	<b>INR 572.00</b>
📞	98765432100
✉	sportsturfplayground@gmail.com
?	Help
	

Fig.4.3

## CHAPTER 5

### EVALUATION

#### 5.1 DATASETS

A dataset is a structured collection of data generally associated with a unique body of work. A database is an organized collection of data stored as multiple datasets. It is a collection of numbers or values that relate to a particular subject. Following are the datasets of our Online sports Turf Playground Booking System.

##### A) Turf Registration

	Column Name	Data Type	Allow Nulls
🔑	TurfID	int	<input type="checkbox"/>
	TurfName	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Location	varchar(50)	<input checked="" type="checkbox"/>
	PinCode	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.1**

##### B) Admin Login

	Column Name	Data Type	Allow Nulls
🔑	AdminID	int	<input type="checkbox"/>
	Password	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.2**

### C) Adding Turfs

	Column Name	Data Type	Allow Nulls
🔑	TurfID	int	<input type="checkbox"/>
	TurfName	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Location	varchar(50)	<input checked="" type="checkbox"/>
	PinCode	varchar(50)	<input checked="" type="checkbox"/>
	Description	varchar(MAX)	<input checked="" type="checkbox"/>

**Fig.5.1.3**

### D) Available Turfs

	Column Name	Data Type	Allow Nulls
🔑	TurfID	int	<input type="checkbox"/>
	TurfName	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Location	varchar(50)	<input checked="" type="checkbox"/>
	Price	varchar(50)	<input checked="" type="checkbox"/>
	Contact	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.4**

### E) Manager Login

	Column Name	Data Type	Allow Nulls
🔑	ManagerID	int	<input type="checkbox"/>
	Password	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.5**

### F) User Registration

	Column Name	Data Type	Allow Nulls
🔑	UserID	int	<input type="checkbox"/>
	FirstName	varchar(50)	<input checked="" type="checkbox"/>
	LastName	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Password	varchar(50)	<input checked="" type="checkbox"/>
	Contact	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.6**

### G) User Login

	Column Name	Data Type	Allow Nulls
🔑	UserID	int	<input type="checkbox"/>
	Password	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.7**

### H) Book Turf

	Column Name	Data Type	Allow Nulls
🔑	BookingID	int	<input type="checkbox"/>
	TurfID	varchar(50)	<input checked="" type="checkbox"/>
	TurfName	varchar(50)	<input checked="" type="checkbox"/>
	Location	varchar(50)	<input checked="" type="checkbox"/>
	PinCode	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.8**

### I) Booking Confirmation

	Column Name	Data Type	Allow Nulls
🔑	BookingID	int	<input type="checkbox"/>
	TurfID	varchar(50)	<input checked="" type="checkbox"/>
	PaymentStatus	varchar(50)	<input checked="" type="checkbox"/>
	GenerateBills	varchar(50)	<input checked="" type="checkbox"/>
	Contact	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.9**

### J) Booking History

	Column Name	Data Type	Allow Nulls
🔑	BookingID	int	<input type="checkbox"/>
	TurfID	varchar(50)	<input checked="" type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Status	varchar(50)	<input checked="" type="checkbox"/>

**Fig.5.1.10**

### K) Support

	Column Name	Data Type	Allow Nulls
🔑	UserID	int	<input type="checkbox"/>
	Email	varchar(50)	<input checked="" type="checkbox"/>
	Message	varchar(50)	<input checked="" type="checkbox"/>
	Reply	varchar(MAX)	<input checked="" type="checkbox"/>

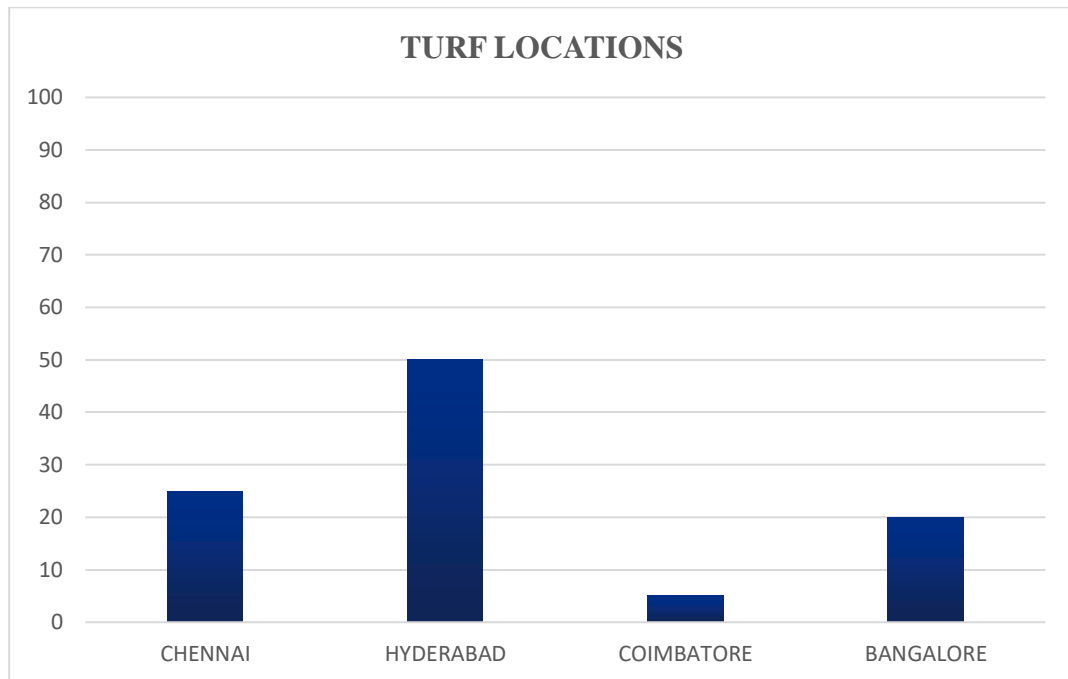
**Fig.5.1.11**

## 5.2 EVALUATION METRICS

An evaluation metric quantifies the performance of a predictive model. This typically involves training a model on a dataset, using the model to make predictions on a holdout dataset not used during training, then comparing the predictions to the expected values in the holdout dataset.

### TURF LOCATIONS

Turf Locations are specified based on the availability of turf playgrounds in various places. The below graph gives the information about the turf users from different locations.



**Fig.5.2.1**



## TURF REGISTRATIONS

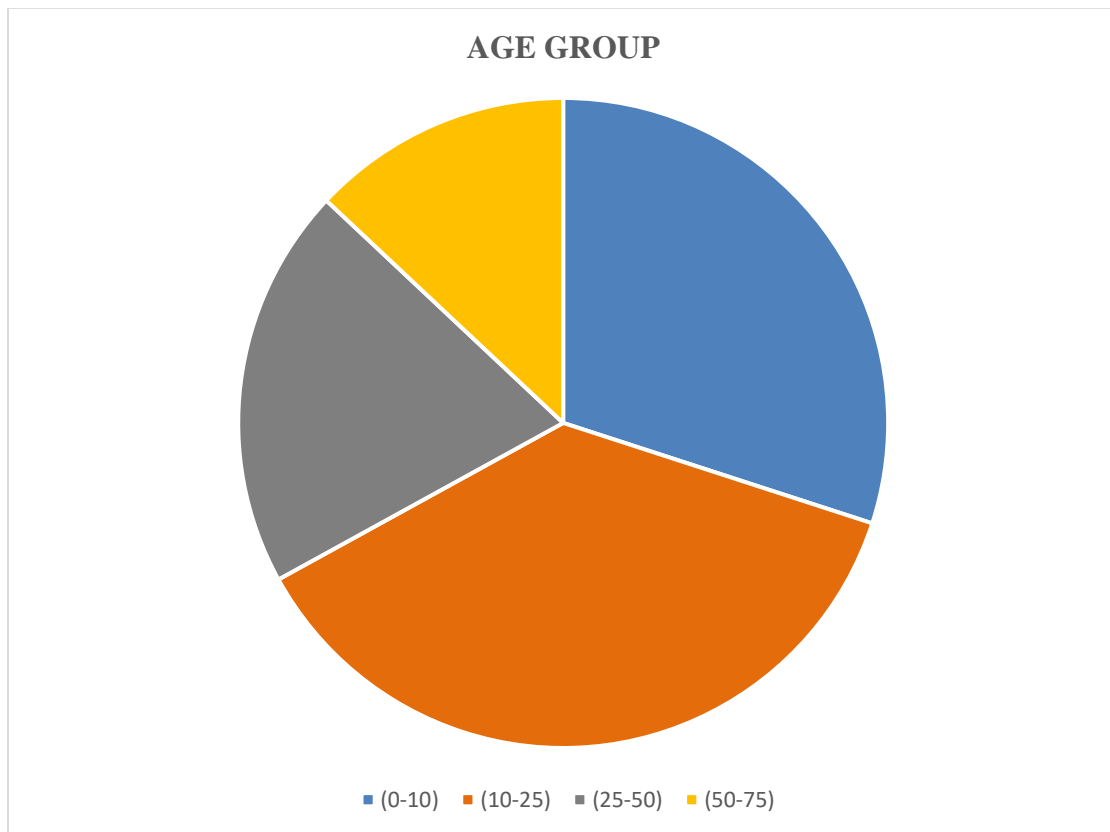
The below Bar Graph is composed with the turf registrations based on particular sport. It helps to recognize the popularity of certain sport among the Turf users.



**Fig.5.2.2**

## AGE GROUP

The below pie chart holds the data of Age Group based on the registrations towards the turf playgrounds. This data assists us to recognize about which specific age group has bounteous interest.



**Fig.5.2.3**

### 5.3 TESTS CASES

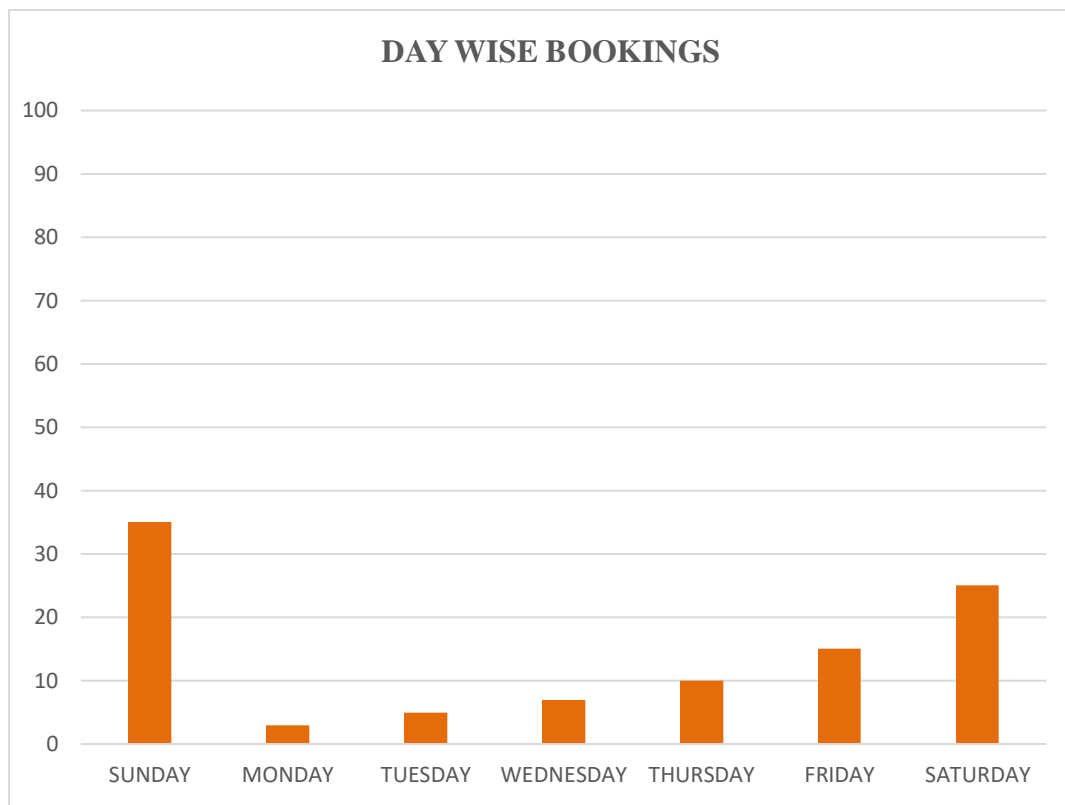
Test cases define how to test a system, software or an application. A test case is a singular set of actions or instructions for a tester to perform that validates a specific aspect of a product or application functionality. If the test fails, the result might be a software defect that the organization can triage.

S.NO	TEST CASE	STATUS
1	Verify Online Sports Turf Playground Booking Website is loading properly or not.	PASSED
2	Verify on searching turf details like available turfs, select timings, price details, booking requests, payment status are displayed.	PASSED
3	Verify that invalid login credentials can't access to the Turf Playground Booking website.	PASSED
4	Verify that the User is able to see slots availability in a turf.	PASSED
5	Verify that the pricing of different types of slots in a turf is displayed to the users.	PASSED
6	Verify User is able to select one or more than one slot.	PASSED
7	Verify if the slot is booked then users should not be able to book the same slot of that turf.	PASSED
8	Verify when user select slot, enters turf details and does the payment then the selected slot should get booked.	PASSED
9	Verify after successful booking of turf user should be able to see his turf booking history.	PASSED
10	Verify that once the user has booked the turf then he can download invoice of it.	PASSED
11	Verify status of the turf gets change to booked after the user booked it.	PASSED
12	Verify user should receive SMS or Mail after successfully booking of the turf.	PASSED
13	Verify when user cancels the booked turf then his amount should get refunded.	PASSED
14	Verify payment from different types of payment mode.	PASSED

## 5.4 RESULTS

### DAY WISE BOOKINGS

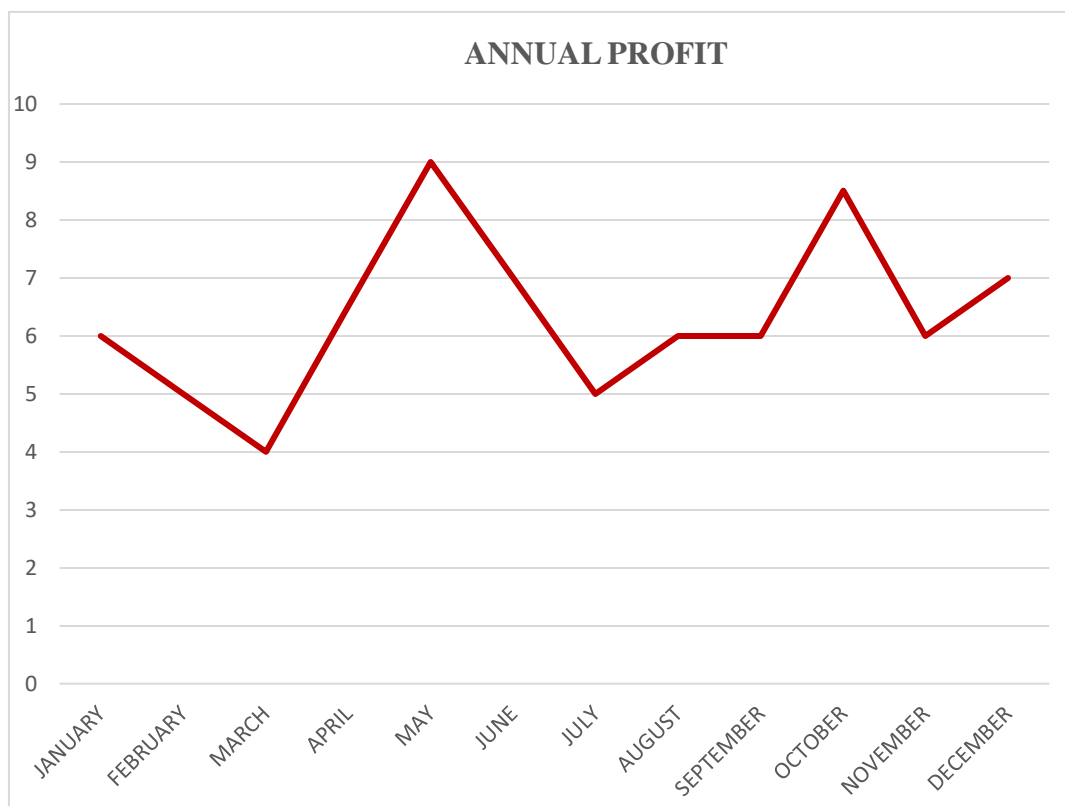
Day Wise bookings report is helpful to compute the booking information on a particular day. The Below bar graph comprises the day wise bookings of our Online sports turf playground booking website.



**Fig.5.4.1**

## ANNUAL PROFIT

Annual Profit is the net income of the online sports turf playground booking system website. Below graph represents the annual profit of our website. It is used to determine the success ratio of our website.



**Fig.5.4.2**

## **CHAPTER 6**

### **CONCLUSION AND FUTURE ENHANCEMENT**

- Our online sports Turf playground booking website provides a streamlined booking and reservation procedure that reduce long queues and overcrowding.
- Online Sports Turf Playground Booking System Sends message reminders to managers and users whenever slots are booked, canceled or rescheduled.
- Users can easily and securely authenticate themselves by linking their existing service by using a password.
- Manager can view booking requests and confirm the turf booking which user requested, later he will generate the bills.
- Admin provides login credentials to manager and can also add turf locations based on their availability.
- Upgrading to an online system can have many advantages to help you stand out from competitors as well as proving a hassle-free booking process to your customers.
- Our online system handles instant payments for booking turfs through Net-Banking, Credit cards, Debit cards or UPI.

## REFERENCES

- **Github** - Overview of the project

<https://github.com/gayu-thri/Online-Sports-Turf-Playground-Booking-System>.

- **PlayO** - for Website reference

<https://playo.co/>

- **Flask** - Python Web framework

<https://www.tutorialspoint.com/flask/index.htm>

- **Python** - Web development

<https://www.w3schools.com/python/>

- **Different Tools used for designing Turf Playground Booking System**

<https://draw.io>

<https://www.microsoft.com/en-in/sql-server/sql-server-downloads>

<https://balsamiq.com/givingback/free/>

## APPENDIX

### DEFINITIONS, ACRONYMS, ABBREVIATIONS

**TURF PLAYGROUND:** Turf is a surface of synthetic fibers made to look like natural grass. It is most often used in areas for sports that were normally played on grass.

**ONLINE BOOKING SYSTEM:** An online booking system is a software solution that allows potential guests to self-book and pay through your website, and other channels, while giving you the best tools to run and scale your operation, all in one place.

**SERVER:** A network server, today is a powerful computer that provides various shared resources to workstations and other servers on a network. The shared resources can include disk space, hardware access and email services.

**FLASK:** Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier.

**DATASETS:** A dataset is a structured collection of data generally associated with a unique body of work. A database is an organized collection of data stored as multiple datasets.



## **SAMPLE CODE**

```
from flask import Flask

from flask import Flask, flash, redirect, render_template, request, session, abort

import os

user = {}

admin = {}

manager = {}

user = {

    'user1' : 'password',

    'user2' : 'password',

    'user3' : 'password'

}

admin['admin'] = 'password'

manager = {

    'manager1' : 'password',

    'manager2' : 'password',

    'manager3' : 'password'

}

app = Flask(__name__)
```

```

global active

global email

global visit

global location,viewreqs,booking_hist,viewbook

global am  ##allocation of manager to a turf  manager:turf

global au,avail_turf  ##allocation of user to a turf  user:turf

global price,reqs  ##request - user:turf

viewreqs = {}

viewbook = {}

location=['Chennai','Bangalore','Coimbatore']

price={'Chennai':500}

am = {'manager1':'Chennai'}

au = {'user1':'Chennai'}

avail_turf = []

reqs = {'user2':'Bangalore'}

booking_hist = []

active = None

visit = 0    #for calculating the number of visits for a web page.

@app.route('/')

```

```

def home():

    return render_template('login.html')

@app.route('/login', methods=['POST'])

def do_admin_login():

    global active

    global email

    if request.form['choice'] == 'user':

        if request.form['username'] in user and user[request.form['username']] == request.form['password']:

            active = request.form['username']

            session['logged_in'] = True

        else:

            return redirect('/error')

    return render_template('home_user.html',email=active)

    elif request.form['choice'] == 'admin':

        if request.form['username'] in admin and admin[request.form['username']] == request.form['password']:

            active = request.form['username']

            session['logged_in'] = True

```

```

else:

return redirect('/error')

return render_template('home_admin.html',email=active)

elif request.form['choice'] == 'manager':

if request.form['username'] in manager and manager[request.form['username']] ==
request.form['password']:

active = request.form['username']

session['logged_in'] = True

else:

return redirect('/error')

return render_template('home_manager.html',email=active)

@app.route('/home_admin', methods=["POST"])

def home_admin():

global visit

visit = visit + 1

global location

if request.form['submit_button'] == 'Add turf location':

return render_template('add_location.html',l = ",".join(location))

global manager

```

```

if request.form['submit_button'] == 'Provide credentials & Add a manager':

    return render_template('add_manager.html',mavail = ",".join(manager.keys()))

    global am

    l = list(am.items())

    if request.form['submit_button'] == 'Allocate a manager':

        return render_template('allocate_manager.html',a = l)

        global price

        if request.form['submit_button'] == 'Add price list':

            return render_template('add_price.html',p = list(price.items()))

            global au,reqs

            if request.form['submit_button'] == 'View Booking':

                return render_template('view_booking.html',b= list(au.items()),r=list(reqs.items()))

            if request.form['submit_button'] == 'View visits':

                return render_template('visitors.html', v=visit)

            if request.form['submit_button'] == 'Log out':

                return logout()

            if request.form['submit_button'] == 'Contact':

                return render_template('contact.html')

            @app.route('/add_location',methods=["POST"])

```

```

def add_location():

    global location

    if request.form['submit_button'] == 'Add':

        location.append(request.form['location'])

        return "After addition, the available locations are: " + ",".join(location) #list passing

@app.route('/add_manager',methods=["POST"])

def add_manager():

    global manager

    if request.form['submit_button'] == 'Add the manager':

        manager[request.form['muname']] = request.form['mpass']

        return "Successfully added manager with credentials: " + "Username-"
        "+request.form['muname'] + "Password-" + request.form['mpass']

@app.route('/allocate_manager',methods=["POST"])

def allocate_manager():

    global am

    if request.form['submit_button'] == 'Add':

        am[request.form['man']] = request.form['loc']

        return "After addition, the available locations are: " + str(list(am.items())) #dict passing

@app.route('/add_price',methods=["POST"])

```

```

def add_price():

    global price

    if request.form['submit_button'] == 'Add':

        price[request.form['loc']] = request.form['price']

        return "After addition, the available locations are: " + str(list(price.items()))

@app.route('/home_user', methods=["POST"])

def home_user():

    global visit

    visit = visit + 1

    global price

    if request.form['submit_button'] == 'Check turf':

        return render_template('check_turf.html', p=str(list(price.items())))

    global au, avail_turf

    for ch in location: ## checks from locations & not in allocated!!

        if ch not in au.values():

            avail_turf.append(ch)

    if request.form['submit_button'] == 'Check availability':

        return render_template('check_availability.html', u=", ".join(avail_turf))

    if request.form['submit_button'] == 'Book a turf':

```

```

return render_template('book_turf.html')

if active in au.keys():

    rl = au[active]

else:

    rl = None

if active in reqs.keys():

    re = reqs[active]

else:

    re = None

if request.form['submit_button'] == 'My history':

    return render_template('my_history.html',out=rl,req=re) ##out- booked , req-requested

if request.form['submit_button'] == 'View visits':

    return render_template('visitors.html', v=visit)

if request.form['submit_button'] == 'Log out':

    return logout()

if request.form['submit_button'] == 'Contact':

    return render_template('contact.html')

@app.route('/book_turf',methods=["POST"])

def book_turf():

```



```

global au,active,reqs

if request.form['submit_button'] == 'Book':

    temp = request.form['loc']

    if active in au.keys():

        var = au[active]

        return "Only 1 booking at a time !! Already booked- "+ active +": "+var

    if temp not in au.values() and temp in location:

        reqs[active] = temp

        return "Requested for: "+ active + ":" + temp

    else:

        return temp + " location not available !! Sorry "

@app.route('/home_manager', methods=["POST"])

def home_manager():

    global visit

    visit = visit + 1

    if request.form['submit_button'] == 'Check rates':

        return render_template('check_rates.html')

    global reqs, am, viewreqs

    if request.form['submit_button'] == 'View Request':

```

```

if active not in am.keys():

    return "You don't have a turf location allotted to you!! Hence no requests will be visible to
    you.. Contact admin"

else:

    assigned_loc = am[active]

    for user,loc in reqs.items():

        if loc == assigned_loc:

            viewreqs[user] = assigned_loc

    return render_template('view_request.html', v = list(viewreqs.items()))

if request.form['submit_button'] == 'Confirm Booking':

    if active not in am.keys():

        return "You don't have a turf location allotted to you!! Hence no requests will be visible to
        you.. Contact admin"

    else:

        assigned_loc = am[active]

        for user,loc in reqs.items():

            if loc == assigned_loc:

                viewreqs[user] = assigned_loc

```

```

return
render_template('confirm_booking.html',general=list(reqs.items()),mine=list(viewreqs.items()))

global price

if request.form['submit_button'] == 'Bill Generation':

return render_template('bill_generation.html')

global viewbook

if request.form['submit_button'] == 'Booking History':    ###WORK - NOT YET FINISHED

assigned_loc = am[active]

for user,loc in au.items():

if loc == assigned_loc:

viewbook[user] = assigned_loc

return render_template('booking_history.html',l=assigned_loc,u = list(viewbook.keys()))

if request.form['submit_button'] == 'View visits':

return render_template('visitors.html', v=visit)

if request.form['submit_button'] == 'Log out':

return logout()

if request.form['submit_button'] == 'Contact':

return render_template('contact.html')

```

```

@app.route('/generate_bill',methods=["POST"])

def generate_bill():

    usr = request.form['usr']

    if request.form['submit_button'] == 'Generate':

        if usr not in au.keys():

            return "The entered user (" +usr+" ) has no booking confirmed."

        else:

            lo = au[usr]

            pr = price[lo]

            return render_template('generate_bill.html',u = usr,l = lo, p = pr)

@app.route('/confirm',methods=["POST"])

def confirm():

    global au

    if request.form['submit_button'] == 'Confirm':

        usr = request.form['usr']

        loc = request.form['loc']

        if loc in au.values():

            return "Sorry!! Location " + loc + " is already booked."

        else:

```

```

au[usr] = loc

return "Successfully booked "+ loc + " for the user: "+usr

@app.route('/rate', methods=["POST"])

def rate():

    global price

    if request.form['loc'] in price.keys():

        var = price[request.form['loc']]

        return "The price for the requested location "+ request.form['loc'] + " is :"+ str(var)

    else:

        return "Sorry!! The requested location is not available"

@app.route('/contactexp', methods=["GET"])

def contactexp():

    return render_template('contactexp.html')

@app.route('/error')

def err():

    flash("Wrong password entered")

    return home()

@app.route("/logout")

def logout():

```

```
session['logged_in'] = False

global active

active = None

return home()

if __name__ == "__main__":

    app.secret_key = os.urandom(12)

    app.run(debug=True,host='127.0.0.1', port=4000)
```

