**SPORTSHUB**

**[Details]**

*Mini Project Report*

*Submitted by*

**Vidhu Krishnan Vinod**

**Reg. No.: AJC19MCA-000**

*In Partial fulfillment for the Award of the Degree of*

**INTEGRATED MASTER OF COMPUTER APPLICATIONS**

**(INMCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING**

**KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with ‘A’ grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

# 2023-2024

## DEPARTMENT OF COMPUTER APPLICATIONS

### AMAL JYOTHI COLLEGE OF ENGINEERING

**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**SPORTSHUB”** is the bona fide work of **VIDHU KRISHNAN VINOD (Regno: AJC00MCA-0000)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

**Guide Name Coordinator Name**

**Internal Guide Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**

**Head of the Department**

**DECLARATION**

I hereby declare that the project report **“SPORTSHUB”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

**Date: STUDENT NAME**

**KANJIRAPPALLY Reg: AJC00MCA-0000**

# ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Coordinator Name** for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Guide Name** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

VIDHU KRISHNAN VINOD

# ABSTRACT

.

**CONTENT**

|  |  |  |
| --- | --- | --- |
| **SL. NO** | **TOPIC** | **PAGE NO** |
| **1** | **INTRODUCTION** |  |
| **1.1** | **PROJECT OVERVIEW** |  |
| **1.2** | **PROJECT SPECIFICATION** |  |
| **2** | **SYSTEM STUDY** |  |
| **2.1** | **INTRODUCTION** |  |
| **2.2** | **EXISTING SYSTEM** |  |
| **2.3** | **DRAWBACKS OF EXISTING SYSTEM** |  |
| **2.4** | **PROPOSED SYSTEM** |  |
| **2.5** | **ADVANTAGES OF PROPOSED SYSTEM** |  |
| **3** | **REQUIREMENT ANALYSIS** |  |
| **3.1** | **FEASIBILITY STUDY** |  |
| **3.1.1** | **ECONOMICAL FEASIBILITY** |  |
| **3.1.2** | **TECHNICAL FEASIBILITY** |  |
| **3.1.3** | **BEHAVIORAL FEASIBILITY** |  |
| **3.1.4** | **FEASIBILITY STUDY QUESTIONNAIRE** |  |
| **3.2** | **SYSTEM SPECIFICATION** |  |
| **3.2.1** | **HARDWARE SPECIFICATION** |  |
| **3.2.2** | **SOFTWARE SPECIFICATION** |  |
| **3.3** | **SOFTWARE DESCRIPTION** |  |
| **3.3.1** | **PHP** |  |
| **3.3.2** | **MYSQL** |  |
| **4** | **SYSTEM DESIGN** |  |
| **4.1** | **INTRODUCTION** |  |
| **4.2** | **UML DIAGRAM** |  |
| **4.2.1** | **USE CASE DIAGRAM** |  |
| **4.2.2** | **SEQUENCE DIAGRAM** |  |
| **4.2.3** | **STATE CHART DIAGRAM** |  |
| **4.2.4** | **ACTIVITY DIAGRAM** |  |
| **4.2.5** | **CLASS DIAGRAM** |  |
| **4.2.6** | **OBJECT DIAGRAM** |  |
| **4.2.7** | **COMPONENT DIAGRAM** |  |
| **4.2.8** | **DEPLOYMENT DIAGRAM** |  |
| **4.2.9** | **COLLABORATION DIAGRAM** |  |
| **4.3** | **USER INTERFACE DESIGN USING FIGMA** |  |
| **4.4** | **DATABASE DESIGN** |  |
| **5** | **SYSTEM TESTING** |  |
| **5.1** | **INTRODUCTION** |  |
| **5.2** | **TEST PLAN** |  |
| **5.2.1** | **UNIT TESTING** |  |
| **5.2.2** | **INTEGRATION TESTING** |  |
| **5.2.3** | **VALIDATION TESTING** |  |
| **5.2.4** | **USER ACCEPTANCE TESTING** |  |
| **5.2.5** | **AUTOMATION TESTING** |  |
| **5.2.6** | **SELENIUM TESTING** |  |
| **6** | **IMPLEMENTATION** |  |
| **6.1** | **INTRODUCTION** |  |
| **6.2** | **IMPLEMENTATION PROCEDURE** |  |
| **6.2.1** | **USER TRAINING** |  |
| **6.2.2** | **TRAINING ON APPLICATION SOFTWARE** |  |
| **6.2.3** | **SYSTEM MAINTENANCE** |  |
| **7** | **CONCLUSION & FUTURE SCOPE** |  |
| **7.1** | **CONCLUSION** |  |
| **7.2** | **FUTURE SCOPE** |  |
| **8** | **BIBLIOGRAPHY** |  |
| **9** | **APPENDIX** |  |
| **9.1** | **SAMPLE CODE** |  |
| **9.2** | **SCREEN SHOTS** |  |

## List of Abbreviation

# CHAPTER 1

# INTRODUCTION

### PROJECT OVERVIEW

SportsHub is a revolutionary platform designed to innovate the sports and fitness experience for enthusiasts and those dedicated to their health. This comprehensive system seamlessly integrates professional sports facility bookings, equipment reservations, personalized fitness training and advanced AI-enabled fitness tracking. Of particular importance is the implementation of MovNet technology, an advanced advancement in precise posture detection that ensures real-time feedback during fitness activities. SportsHub provides users with easy access to booking sports facilities and equipment, including options for football, basketball, cricket, tennis and volleyball, all with unsurpassed convenience. . At the same time, health-conscious people can create personalized workout programs and workout routines, under the supervision of certified trainers. MovNet technology fundamentally transforms fitness monitoring, allowing users to partake in virtual training sessions where their workout posture is meticulously analyzed, offering instant feedback, posture violation alerts, and detailed progress reports. By merging the power of MovNet with tailored user experiences, SportsHub aims to redefine the standard for sports and fitness platforms, setting a new benchmark for accessibility, efficiency, and personalization.

### PROJECT SPECIFICATION

Technology stack:

SportsHub uses an advanced technology stack, including Django as the core framework, HTML for site building, CSS for layout and style and SQLite as a relational database. for effective data management. The integration of MovNet technology enhances posture detection, ensuring accurate analysis when monitoring fitness.

Module details:

Administration module:

The administration module serves as the central platform for system administration.

Administrators have authority to manage sports facilities and equipment, update product catalogs, and oversee user management and system configuration.

Registered Sports Users Module:

Suitable for sports enthusiasts, this module simplifies the booking of sports facilities and equipment.

's secure payment options enable a smooth booking process, enhancing the sports experience.

Registered Gym User Module:

Designed for those interested in fitness, this module provides access to a variety of fitness resources.

users can choose from personalized workout programs and secure payment methods for their fitness journey.

Registered Instructors Module:

Certified

Trainers use this module to interact with users, provide personalized fitness plans, and closely monitor progress user program.

Real-time feedback and guidance during training sessions optimizes user performance. Project

Goals and Future Developments:

Project

Goals:

SportsHub's main mission is to democratize sports and fitness activities, ensuring accessibility, efficiency and personalization. Leveraging MovNet technology, it aims to provide exceptional user experience and expert advice through real-time fitness tracking, thereby setting new industry standards.

Future expansion and growth:

The initial implementation of SportsHub is the basis for future expansion. Future phases will focus on improving the capabilities of all modules, increasing user engagement, and introducing additional features including improved MovNet integration for detection more comprehensive posture.

SportsHub, with its advanced MovNet posture detection technology, is poised to redefine the sports and fitness landscape, delivering unparalleled user experience, unmatched convenience and expert guidance. This project continues to serve as a solid foundation for future expansion, enhancing the sports and fitness journey of all users.

# CHAPTER 2

# SYSTEM STUDY

# 

### INTRODUCTION

Transforming sports and fitness with SportsHub

In an era where technology and personal health are intertwined, SportsHub is emerging as a pioneering platform poised to redefine the sports and fitness experience. Fitness. This groundbreaking project seamlessly integrates professional sports facility bookings, equipment bookings, personal fitness training and advanced AI-integrated fitness tracking, powered by cutting-edge MovNet technology to automatically detect correct postures. SportsHub's goal is to democratize sports and fitness, ensuring accessibility, effectiveness and personalization, setting new industry standards. Through an innovative combination of technology and personalized user experience, SportsHub is not only growing but revolutionizing the sports and fitness landscape, delivering convenience, personalization and advice Great expertise.

### EXISTING SYSTEM

**2.2.1 NATURAL SYSTEM STUDIED**

The existing system for sports and fitness activities typically relies on traditional methods, often involving manual booking processes for sports facilities and equipment. Users may have limited options for personalized fitness training, and progress monitoring primarily occurs through personal records and sporadic trainer interactions. Advanced AI technology, such as MovNet, is not integrated into the existing system, limiting real-time fitness monitoring capabilities. Access to sports and fitness resources might not be user-friendly, and there may be a lack of centralized, comprehensive platforms that cater to both sports and fitness needs.

**2.2.2 DESIGNED SYSTEM STUDIED**

The designed system, SportsHub, is meticulously crafted to overcome the limitations of the existing system by introducing automated online booking, personalized fitness programs, real-time AI-integrated monitoring, enhanced safety through precise posture detection, the consolidation of sports and fitness resources, user-friendly digital platforms, and improved accessibility. SportsHub revolutionizes the user experience, making it more efficient and accessible, while also providing comprehensive solutions for sports enthusiasts and fitness-conscious individuals.

### DRAWBACKS OF EXISTING SYSTEM

### Manual booking processes for sports facilities and equipment.

### Limited personalization options for fitness training.

### Lack of real-time fitness monitoring capabilities.

### Safety concerns due to a lack of immediate feedback.

### Fragmented resources across different platforms or locations.

### Lack of user-friendly digital platforms for booking.

### Limited accessibility for a wide range of users.

### PROPOSED SYSTEM

SportsHub is a forward-thinking solution that addresses the limitations of the existing system. It streamlines the process of booking sports facilities and equipment, offering a wide range of options for sports enthusiasts. Additionally, it provides personalized fitness training, enabling users to tailor fitness programs to their specific needs. The integration of MovNet technology allows for precise and real-time pose detection during fitness activities, enhancing safety and performance optimization. The proposed system creates an innovative, user-friendly ecosystem that combines sports and fitness resources, making them accessible, efficient, and tailored to individual needs. It aims to set new industry standards for the sports and fitness experience by offering convenience, personalization, and expert guidance.

In summary, SportsHub represents a significant leap forward from the limitations of the existing system, utilizing technology, personalization, and advanced fitness monitoring to create a comprehensive and transformative platform for sports and fitness enthusiasts.

### ADVANTAGES OF PROPOSED SYSTEM

* **Efficient online booking.**
* **Personalized fitness training.**
* **Real-time fitness monitoring.**
* **Enhanced safety features.**
* **Comprehensive resource access.**
* **User-friendly digital interface.**
* **Inclusivity for a wider user base.**
* **Scalability and future expansion.**

# CHAPTER 3

# REQUIREMENT ANALYSIS

## FEASIBILITY STUDY

Feasibility refers to the practicality of successfully executing a project. To evaluate the feasibility

of a project, a feasibility study is conducted, which assesses the practicality and workability of the

proposed solution to meet the requirements of the software. The feasibility study takes into account

various factors such as resource availability, cost estimation for software development, the benefits

of the software to the organization after its development, and the cost of maintaining the software.

The findings of the feasibility study should be presented in a report that recommends whether or

not to proceed with the requirements engineering and system development process. The primary

objective of the feasibility study is to determine the reasons for developing software that is

acceptable to users, adaptable to change, and compliant with established standards.

Other objectives of the feasibility study are listed below:

• To analyse whether the software will meet organizational requirements

• To determine whether the software can be implemented using the current technology and within

the specified budget and schedule

• To determine whether the software can be integrated with other existing software.

### Economical Feasibility

Economic feasibility for the SportsHub project is a fundamental assessment of its financial viability, encompassing a comprehensive cost-benefit analysis, revenue projections, break-even analysis, and return on investment calculation. This analysis aims to determine whether the project's expected financial benefits, including potential revenue streams from user subscriptions, equipment bookings, and trainer fees, outweigh the initial development and ongoing operational costs. By evaluating potential cost reduction benefits, conducting risk assessments, and performing sensitivity analyses, the project's economic feasibility is meticulously examined to inform stakeholders about its financial prospects and to guide sound decision-making regarding its implementation.

### Technical Feasibility

Technical feasibility analysis for the SportsHub project is a critical examination of the system's technical viability, encompassing an evaluation of the chosen technology stack, the integration of MovNet technology for precise pose detection, the availability of development expertise, data security measures, scalability considerations, and potential technical risks. This assessment aims to ensure that the project can be efficiently developed, operated, and maintained, utilizing the selected technologies and technical resources. By addressing compatibility, security, scalability, and potential challenges, the technical feasibility analysis provides stakeholders with a clear understanding of the project's readiness for implementation and any necessary technical adaptations to ensure its success.

### Behavioral Feasibility

The SportsHub Project's behavioral feasibility analysis is a critical assessment of the acceptability and adoption of the system within the user community, including considerations of users' willingness to adopt the system. new platform adoption, behavior change management strategy, user experience, feedback integration, cultural alignment, training . and stakeholder engagement. This analysis ensures that the project aligns with user needs and preferences, facilitates a smooth transition, and provides a positive user experience. By addressing both the human and organizational aspects, the Behavioral Feasibility Assessment informs stakeholders about user and organizational readiness to successfully adopt and integrate the SportsHub system into their exercise and sports routines.

**3.1.4 Feasibility Study Questionnaire**

1. Do you have access to outdoor fields, courts, or tracks for practicing sports, or are they limited?

Ans: Yes, good quality is also rare

2. How easy or difficult is it to book or reserve sports facilities for personal training or practice

sessions?

Ans: booking not hard, but others are difficult.

3. Are there any local clubs or teams you can join to participate in organized sports competitions or

leagues?

Ans: yes, but there also they can’t afford to access professional facilities are rare.

4. Have you ever faced challenges in accessing sports or fitness facilities due to factors like

distance, cost, or availability?

Yes, mostly distance is prominent for good quality facility.

5. Are there any initiatives or programs in your community that promote sports and fitness for

teenagers?

None

6. How do you cope with the frustration of not having access to specific gym equipment during

your workout sessions?

Yes, overcrowding is there

4

7. Do you believe that the fitness center should take measures to address equipment shortages,

and if so, what suggestions do you have for improvement?

yes

8. Have you ever discussed the issue of equipment shortage with other fitness enthusiasts at the

gym, and what are their thoughts on the matter?

Proper management can overcome that problem.

9. If there were a system for reserving gym equipment during busy hours, would you find it helpful,

or do you prefer a first-come, first-serve approach?

Yes would be helpful.

10. Do you think the shortage of gym equipment affects the overall quality of your fitness center

experience? How could this be improved?

Yes it can be improved

## SYSTEM SPECIFICATION

### Hardware Specification

Processor - intel i5

RAM - 8gb ram

Hard disk - 1Tb

### Software Specification

Front End - HTML, CSS, JavaScript

Back End - Python (Django framework)

Database - SQLite

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, CSS,MoveNet

## SOFTWARE DESCRIPTION

## 3.3.1 HTML

## HTML, short for Hypertext Markup Language, serves as the basic language for creating web pages. It provides a set of essential elements that structure and give meaning to the document's content. As part of the SportsHub project, HTML5 is used to design websites. HTML not only helps organize content into logical sections, but also allows for the inclusion of multimedia elements, such as images and videos. Additionally, HTML validation is used to maintain code quality, ensuring proper compliance with web standards and guidelines.

## 3.3.2 CSS

## CSS, or cascading style sheets, play a central role in defining the layout and visual style of website content. While HTML defines the structure and semantic meaning of content, CSS manages its appearance. It allows customization of various aspects, including fonts, colors, sizes, spacing, and visual effects. In the SportsHub project, CSS3 is integrated to style web pages and create animations that enhance the visual appeal of the website. CSS is an essential technology that ensures a consistent and visually appealing user experience.

## 3.3.3 Python (Django Framework)

## Python, a flexible and powerful programming language, is used as the back-end language for the SportsHub project. Specifically, the Django web framework is used to support the development of robust, scalable, and secure web applications. Django simplifies various aspects of web development, including user authentication, database management, and URL routing. It follows the Model-View-Controller (MVC) architectural pattern, which prioritizes clean and maintainable code. Python, combined with Django, forms the backbone of the system, allowing for efficient data processing and seamless execution of business logic.

## 3.3.4 SQLite

## Although not used in this project, it is worth mentioning that SQLite is a lightweight and efficient "RDBMS" relational database management system; Widely used for small and medium applications. It excels in situations that require compact, self-contained databases. However, in the SportsHub project, the more powerful MySQL is used as the RDBMS. MySQL offers scalability, speed, and reliability, making it suitable for managing system data effectively.

## 3.3.5 MovNet

## The integration of MovNet technology represents an advanced addition to the SportsHub project. MovNet, an advanced solution, specializes in precise posture detection. It enables real-time analysis of users' exercise postures and provides valuable feedback when tracking physical activity. This technology improves user experience by providing instant information and guidance, thereby promoting consistent and effective training techniques. The system leverages MovNet to ensure accurate progress tracking and physical activity tracking, which distinguishes it as a modern and innovative platform.

# 

# CHAPTER 4

# SYSTEM DESIGN

* 1. **INTRODUCTION**

Any engineered system or product's development process begins with design. A creative process

is design. The secret to an efficient system is a decent design. The process of using different

methodologies and concepts to specify a process or a system in enough detail to allow for its

physical realization is referred to as "design." One way to describe it is as the process of using

different methodologies and concepts to specify a device, a process, or a system in enough detail

to allow for its physical realization. Regardless of the development paradigm that is employed,

software design forms the technical core of the software engineering process. The architectural details required to build a system or product are developed during the system design process. This

program has also gone through the best possible design phase, fine-tuning all levels of efficiency, performance,

and accuracy, just like any system engineering. A user-facing document is converted into a document for programmers or database staff during the design phase. The two stages of system design development are logical design and physical design.

### 4.2 UML DIAGRAM

The standardised language known as UML, which is used for the purposes of designing, developing, visualising, and describing software system products, was created by the Object Management Group (OMG). The UML 1.0 standard's initial draught was published in January 1997, and since then it has been widely utilised as a visual language for describing and developing software systems. In contrast to programming languages like Java, C++, and COBOL, UML is used to depict the flow and interactions of a system's components through diagrams rather than writing code. Although UML is most frequently used for software system design, it can also be applied to non-software systems, such as the process flow in a manufacturing facility. With the use of accessible tools, UML diagrams may be translated into code for a number of computer languages. UML is now recognized as a standard by OMG due to its successful standardisation and tight relationship to the analysis and design of object-oriented systems.

The nine different UML diagrams are:

* Class Diagram
* Object Diagram
* Use Case Diagram
* Sequence Diagram
* Activity Diagram
* State chart Diagram
* Deployment Diagram
* Component Diagram

#### 4.2.1 USE CASE DIAGRAM

A use case diagram is a graphical representation that demonstrates the relationships between the functions of various system components. It is a method for determining, outlining, and organising the system's requirements, such as the necessity for a website for ordering goods and services for mailing. Use case diagrams are one of the tools used by the popular modelling language UML to describe the construction and functioning of systems and objects seen in the real world. The tasks that make up a system's objectives could range from defining the general needs to approving hardware designs, testing and troubleshooting currently under development software products, creating online help resources, and carrying out customer service-related tasks. For instance, in the context of a scenario involving the selling of goods, use cases can include responding to customer service requests, processing product orders, maintaining product catalogues, and controlling payment processing. Typically, a use case diagram has four essential components.

* The wall separating the studied system from its surroundings.
* The actors are often participants in the system who are identified by their roles.
* The actors within and outside of the system carry out the use cases, which are specialist roles.
* The conversations and interactions between the participants and the use cases.

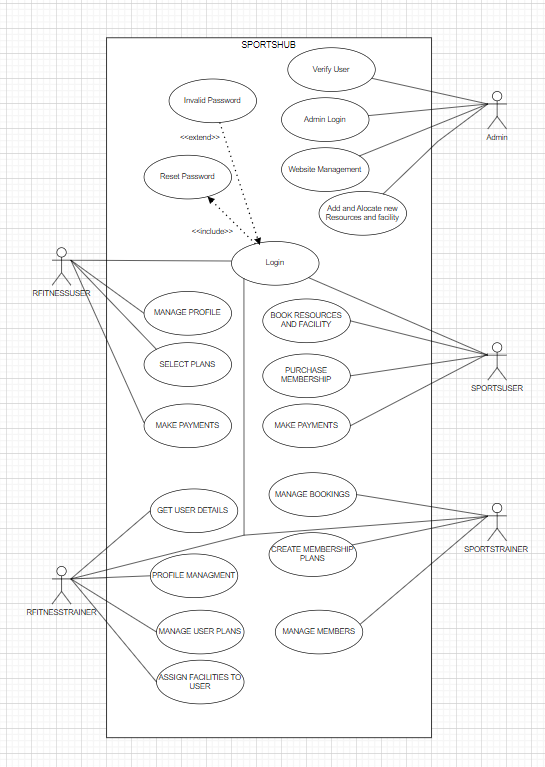


Figure 1: Use Case Diagram

#### 4.2.2 SEQUENCE DIAGRAM

A sequence diagram is a visual representation of the interaction between objects in a specific order. It can also be referred to as an event diagram or event scenario. The purpose of a sequence diagram is to illustrate the order in which objects in a system perform their functions. These diagrams are frequently utilized by software developers and business professionals to document and comprehend the requirements for both new and existing systems.

##### Sequence Diagram Notations –

1. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.
2. **Lifelines –** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.
3. **Messages –** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

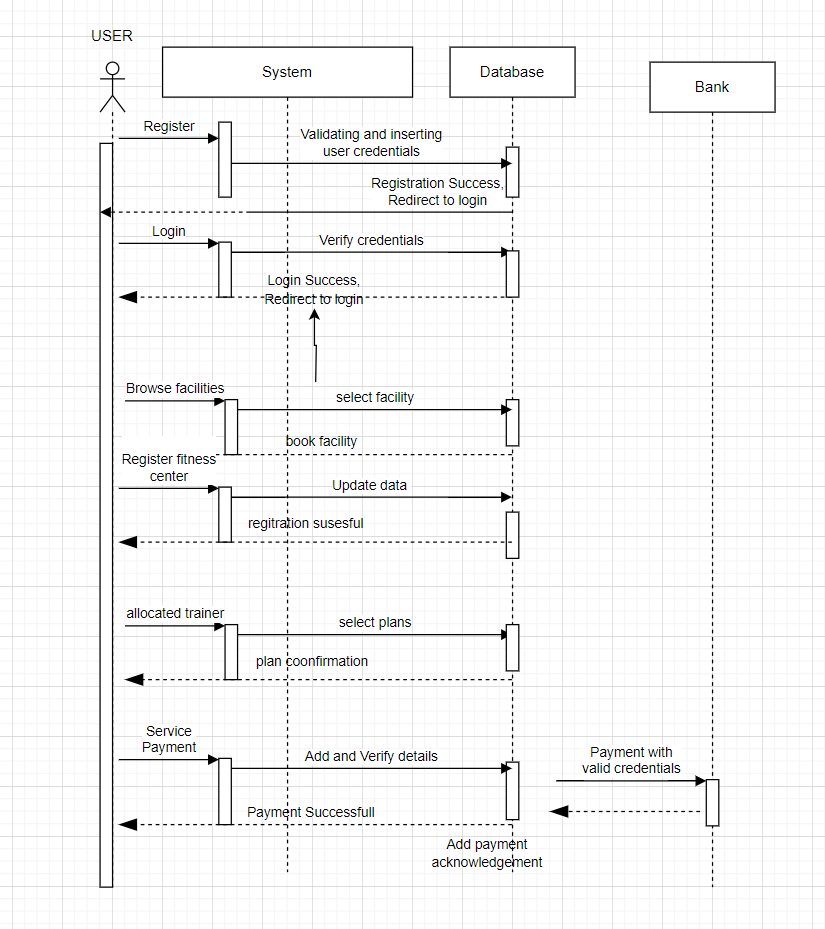
Messages can be broadly classified into the following categories:

* + Synchronous messages
  + Asynchronous Messages
  + Create message
  + Delete Message
  + Self-Message
  + Reply Message
  + Found Message
  + Lost Message

1. **Guards –** To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

##### Uses of sequence diagrams –

* Used to model and visualize the logic behind a sophisticated function, operation or procedure.
* They are also used to show details of UML use case diagrams.
* Used to understand the detailed functionality of current or future systems.
* Visualize how messages and tasks move between objects or components in a system.



##### 4.2.3 State Chart Diagram

The state machine diagram is also called the State chart or State Transition diagram, which shows the order of states underwent by an object within the system. It captures the software system's behavior. It models the behavior of a class, a subsystem, a package, and a complete system. It tends out to be an efficient way of modeling the interactions and collaborations in the external entities and the system. It models event-based systems to handle the state of an object. It also defines several distinct states of a component within the system. Each object/component has a specific state.

###### Notation of a State Machine Diagram

Following are the notations of a state machine diagram enlisted below:

1. **Initial state:** It defines the initial state (beginning) of a system, and it is represented by a black filled circle.
2. **Final state:** It represents the final state (end) of a system. It is denoted by a filled circle present within a circle.
3. **Decision box:** It is of diamond shape that represents the decisions to be made on the basis of an evaluated guard.
4. **Transition:** A change of control from one state to another due to the occurrence of some event is termed as a transition. It is represented by an arrow labeled with an event due to which the change has ensued.
5. **State box:** It depicts the conditions or circumstances of a particular object of a class at a specific point of time. A rectangle with round corners is used to represent the state box.

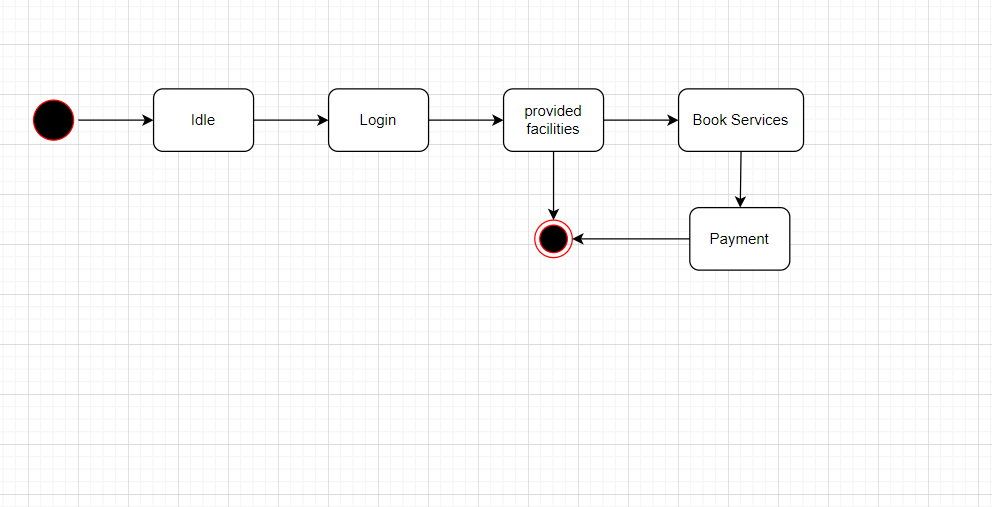


Figure 3: State Chart Diagram

##### 4.2.4 Activity Diagram

An activity diagram in UML is used to illustrate the flow of control within a system, focusing on the workflow between activities rather than the implementation. It can depict both concurrent and sequential activities, and emphasizes the order and conditions of the flow. The diagram includes features such as fork and join to handle different types of flow. It is often referred to as an object- oriented flowchart and is composed of a set of actions or operations to model the behavioral diagram.

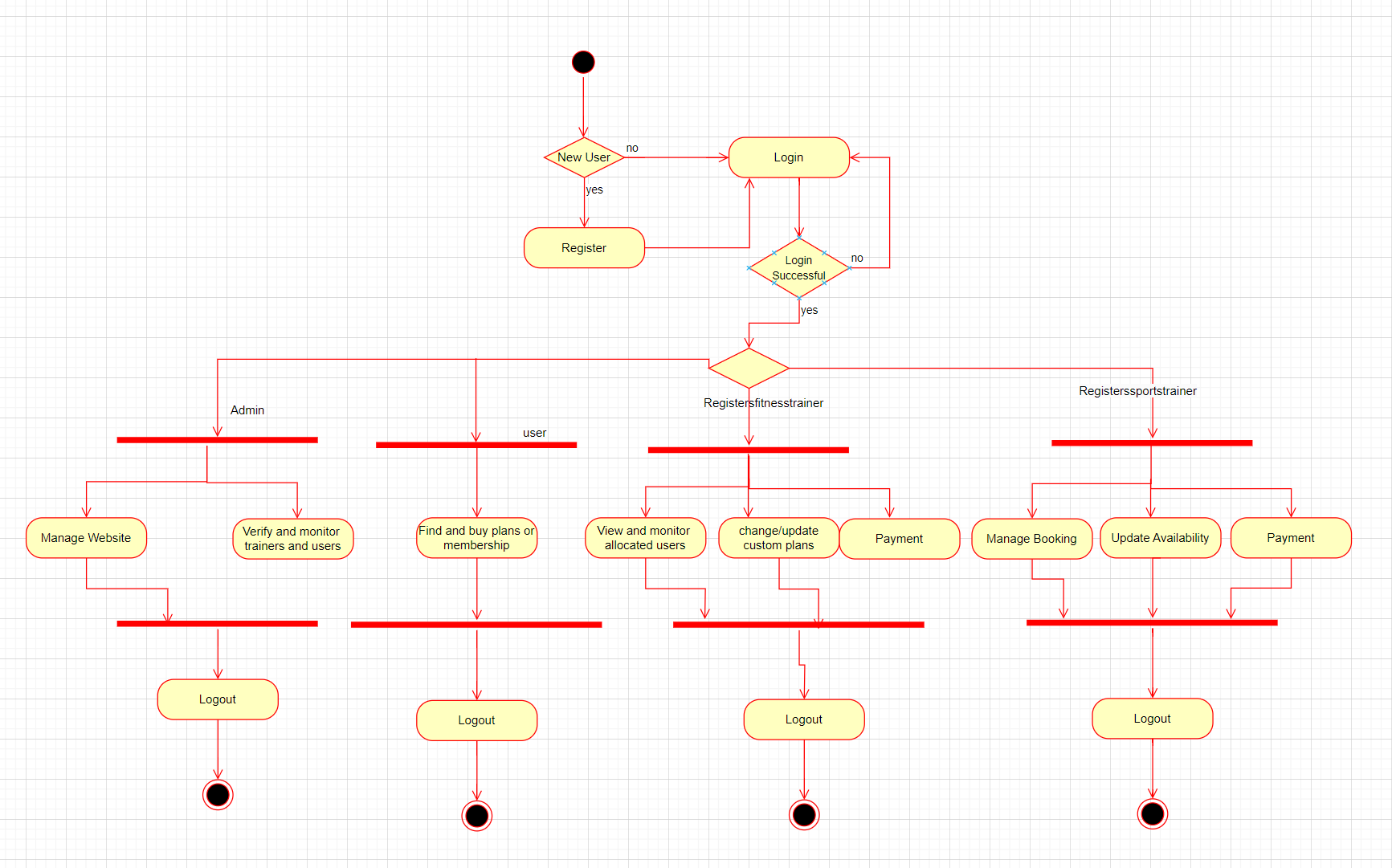
###### Components of an Activity Diagram

1. **Initial Node:** This node marks the starting point of the activity diagram.
2. **Activity States:** These states represent the various activities involved in a process. They are depicted by rounded rectangles.
3. **Decision Nodes:** Decision nodes represent a point in the activity diagram where the flow can take different paths depending on some condition.
4. **Fork Nodes:** Fork nodes indicate parallel processing of activities.
5. **Join Nodes:** Join nodes are used to merge parallel processing flows back into a single flow.
6. **Final Nodes:** Final nodes indicate the end of an activity diagram.
7. **Control Flows:** Control flows represent the order in which the activities are performed and are represented by arrows connecting the various nodes.
8. **Action States:** Action states represent the actions or operations performed during an activity and are represented by rectangles with rounded edges.
9. **Swimlanes:** Swimlanes are used to group activities according to their responsible parties or departments, making it easier to understand the flow of work.

###### Notation of an Activity diagram

Activity diagram constitutes following notations:

1. **Initial State:** It depicts the initial stage or beginning of the set of actions.
2. **Final State:** It is the stage where all the control flows and object flows end.
3. **Decision Box:** It makes sure that the control flow or object flow will follow only one path.
4. **Action Box:** It represents the set of actions that are to be performed.



##### 4.2.5 Class Diagram

A static view of a programme is shown in the class diagram. The many item types that the system contains, as well as any links between them, are displayed. Despite the fact that a class can also derive from other classes, a class is only as good as its objects. Class diagrams are used to depict, describe, and document a variety of system components. They are also employed to create executable software code.

A structural diagram gives an overall picture of the architecture of a software system by displaying the classes, relationships, properties, and functionalities of the system. In order to support the creation of software applications in a given domain, it helps to collect information about class names, properties, and capabilities.

###### Components of a Class Diagram

The class diagram is made up of three sections:

**Upper Section:** The upper section encompasses the name of the class. A class is representation of similar objects that shares the same relationships, attributes, operations, and semantics.Some of the following rules that should be taken into account while representing a class are given below:

1. Capitalize the initial letter of the class name.
2. Place the class name in the center of the upper section.
3. A class name must be written in bold format.
4. The name of the abstract class should be written in italics format.

**Middle Section:** The middle section constitutes the attributes, which describe the quality of class. • The attributes have the following characteristics:

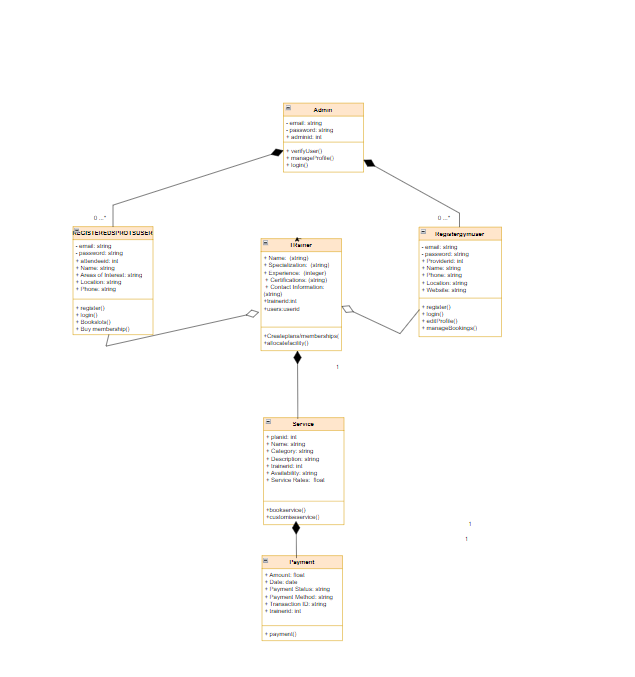
* The attributes are written along with its visibility factors, which are public (+), private (-), protected (#), and package (~).
* The accessibility of an attribute class is illustrated by the visibility factors.
* A meaningful name should be assigned to the attribute, which will explain its usage inside the class.

**Lower Section:** The lower section contain methods or operations. The methods are represented in the form of a list, where each method is written in a single line. It demonstrates how a class interacts with data.

###### Relationships

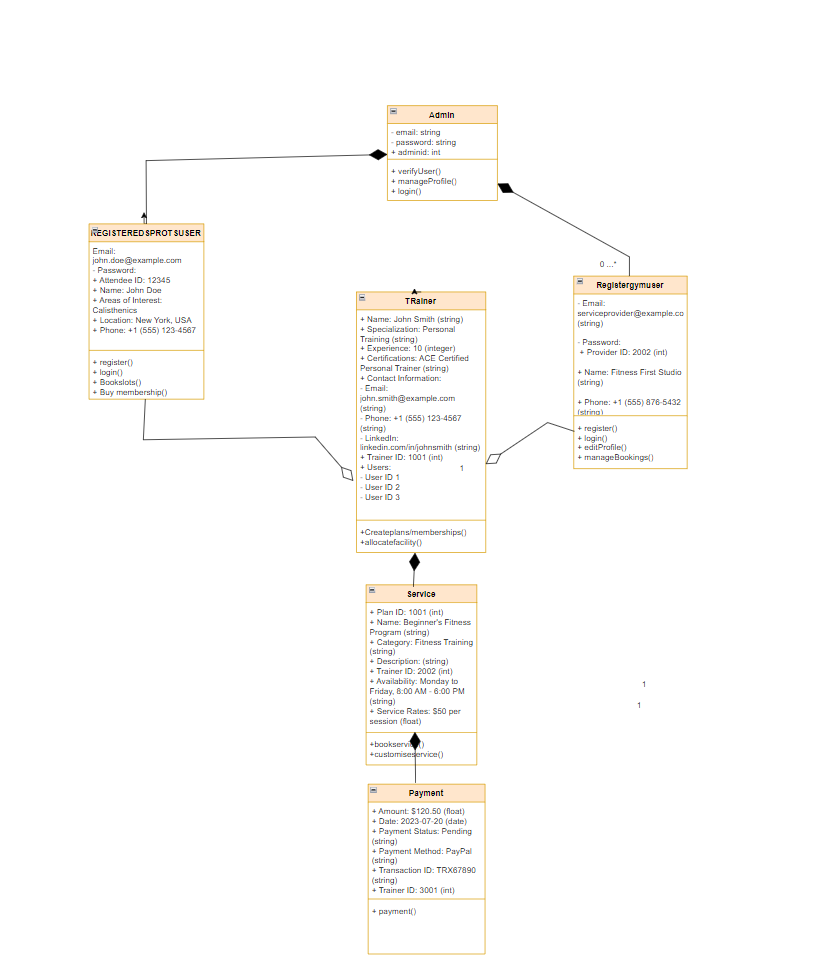
In UML, relationships are of three types:

* **Dependency:** A dependency is a semantic relationship between two or more classes where a change in one class cause changes in another class. It forms a weaker relationship.
* **Generalization:** A generalization is a relationship between a parent class (superclass) and a child class (subclass). In this, the child class is inherited from the parent class.
* **Association:** It describes a static or physical connection between two or more objects. It depicts how many objects are there in the relationship.



##### 4.2.6 Object Diagram

Object diagrams rely on the class diagram as they are based on the class diagram. They illustrate an occurrence of a class diagram by representing an object. Objects provide a static representation of an object-oriented system at a particular moment. The object and class diagrams are similar in some ways, but the class diagram presents a conceptual view of a system, giving a visual representation of a specific function of a system.

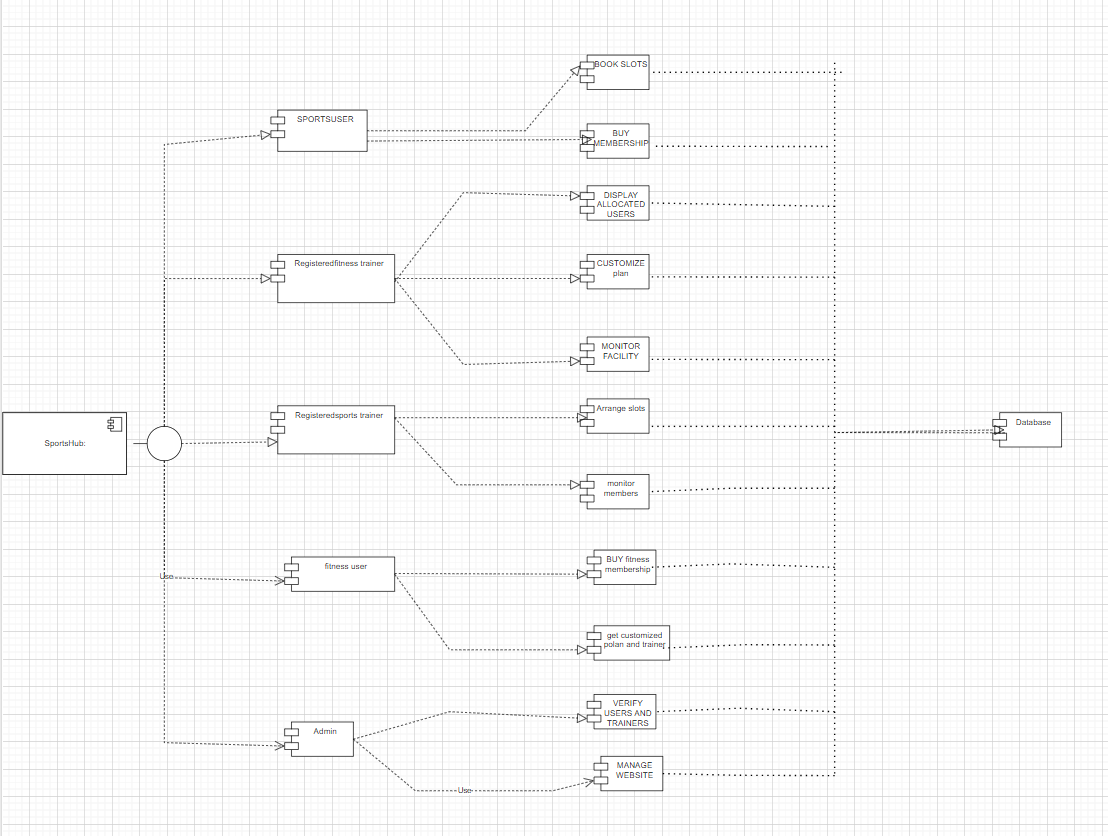


##### 4.2.7 Component Diagram

To make a complex object-oriented system more understandable, the smaller components are separated out using a component diagram. It simulates the physical view of a system, including its internal node's executables, files, libraries, etc. It depicts the connections and hierarchies that exist between the system's components. It aids in creating an operational system. An individual, replaceable, and executable system unit is referred to as a component. A component's implementation details are concealed, therefore an interface is required to carry out a function. It functions like a "black box," with the provided and necessary interfaces explaining its behaviour.

###### Notation of a Component Diagram

1. A component
2. A node



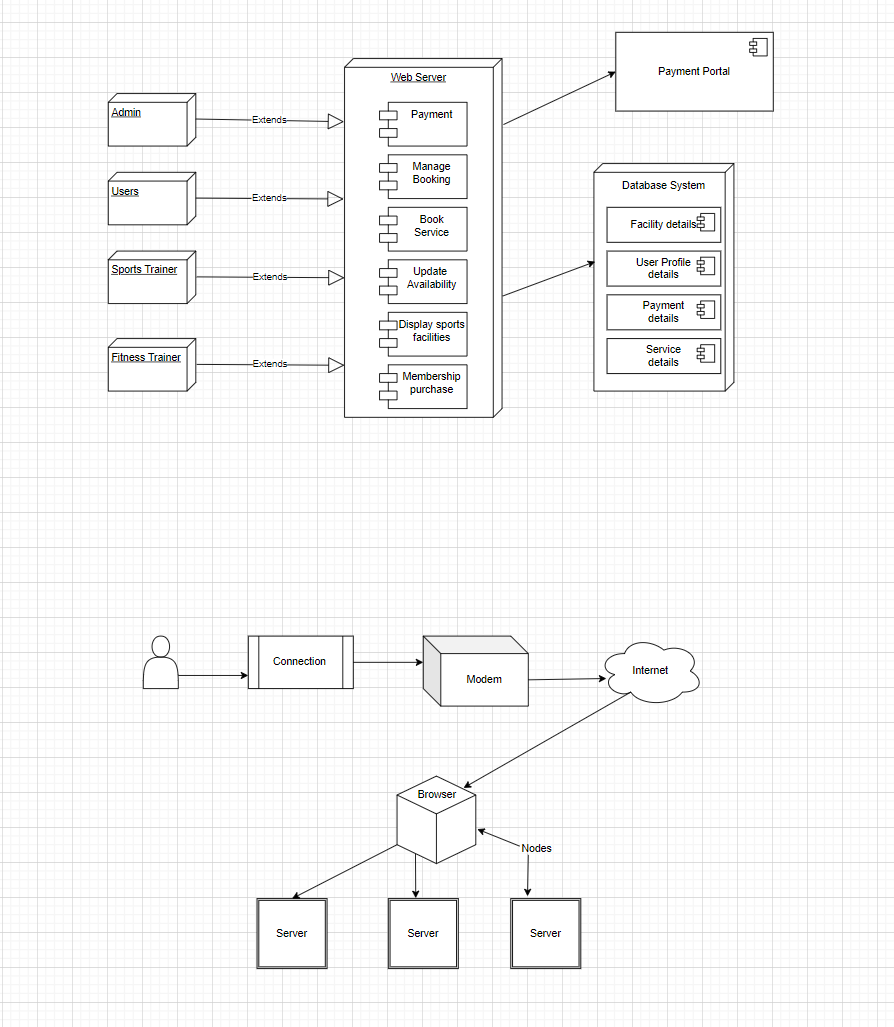
##### 4.2.8 Deployment Diagram

A deployment diagram is a type of UML diagram that shows the physical hardware or system architecture on which software applications will be deployed. It provides a static view of the system deployment and includes nodes and their relationships. The diagram illustrates how the software components are distributed and deployed on the physical hardware or nodes. The diagram is useful for mapping the software architecture designed to the physical system architecture, where the software will be executed. Communication channels are used to show the relationships between the nodes involved.

###### Notation of Deployment diagram

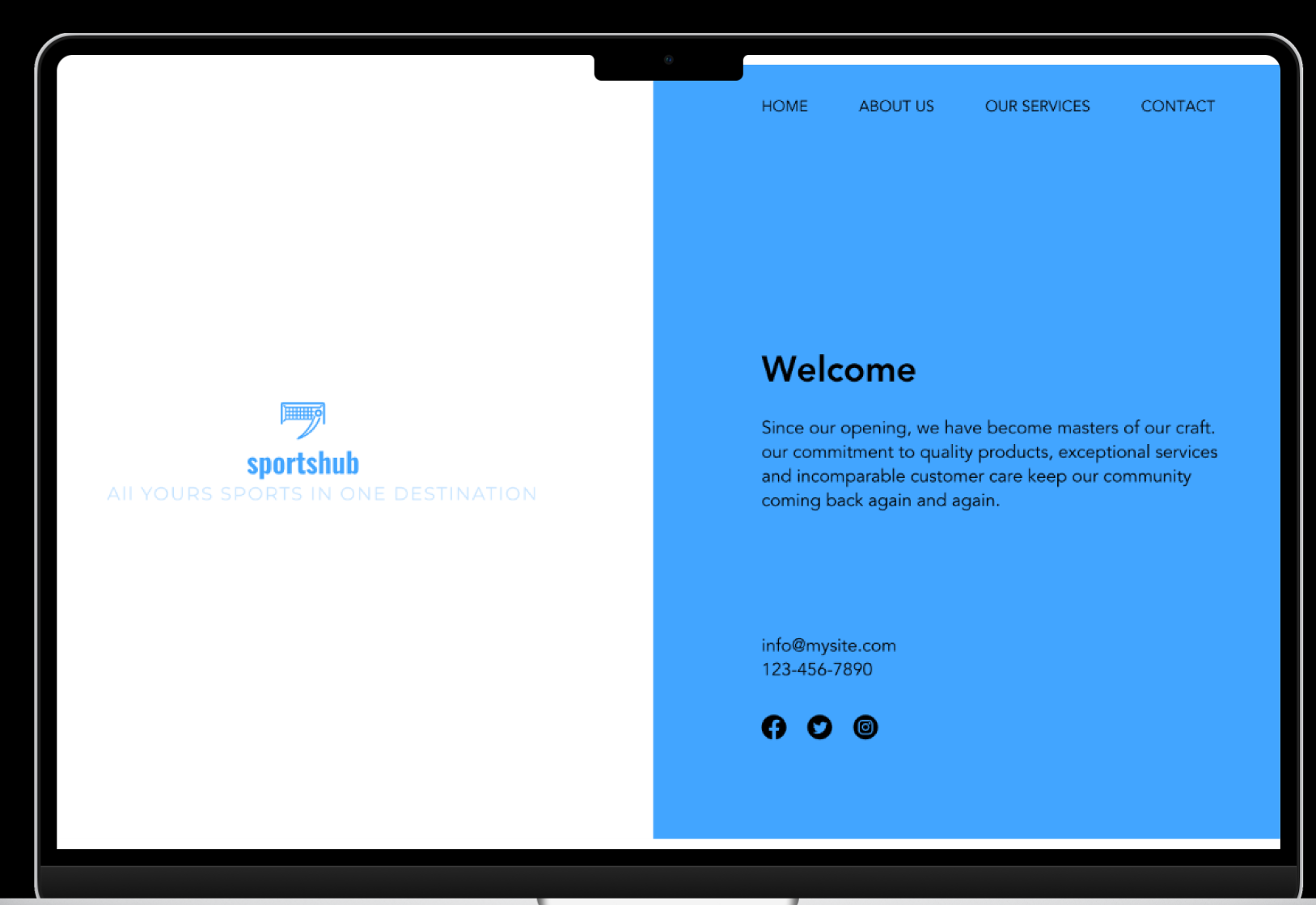
The deployment diagram consists of the following notations:

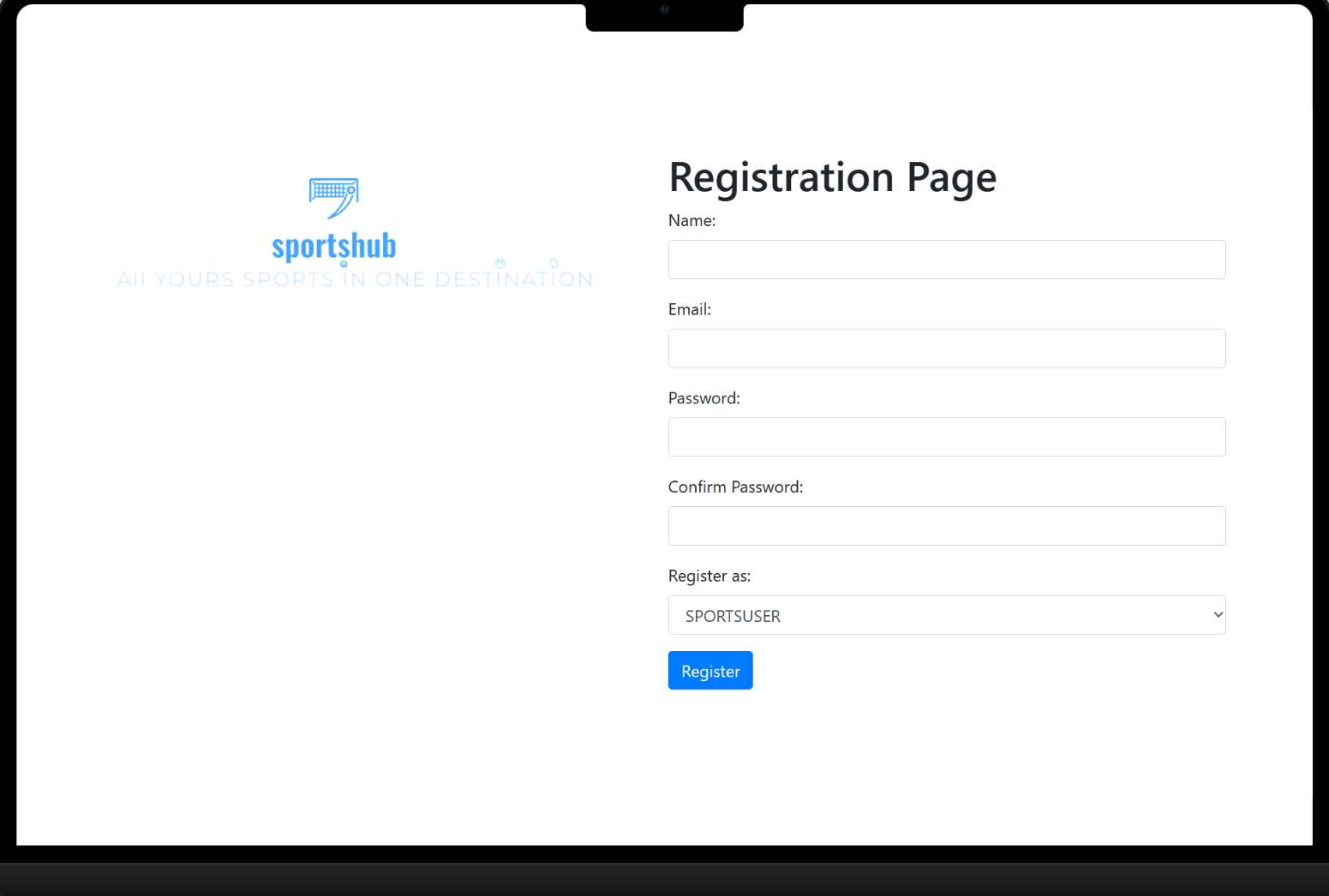
1. A component
2. An artifact
3. An interface
4. A node

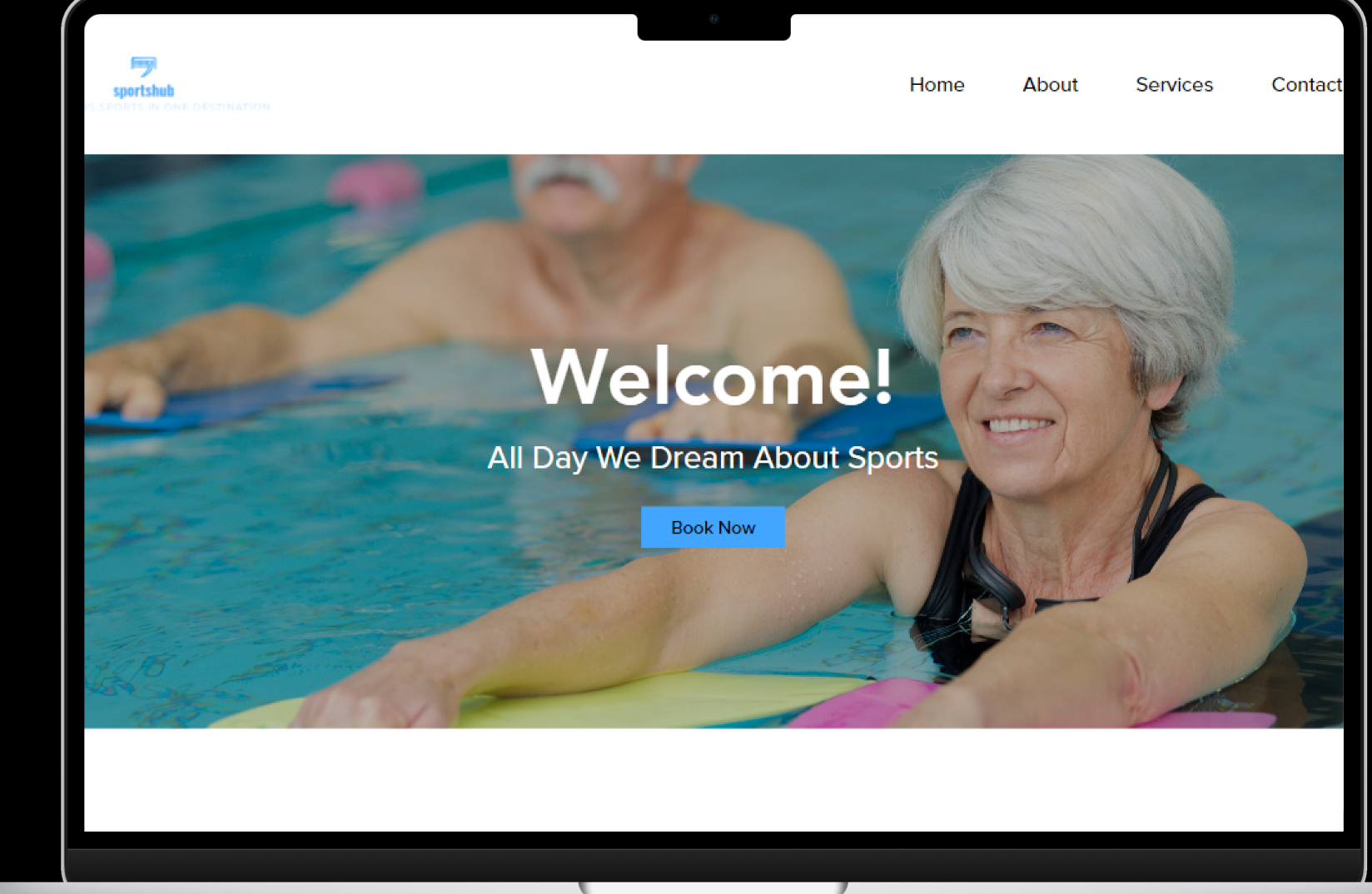
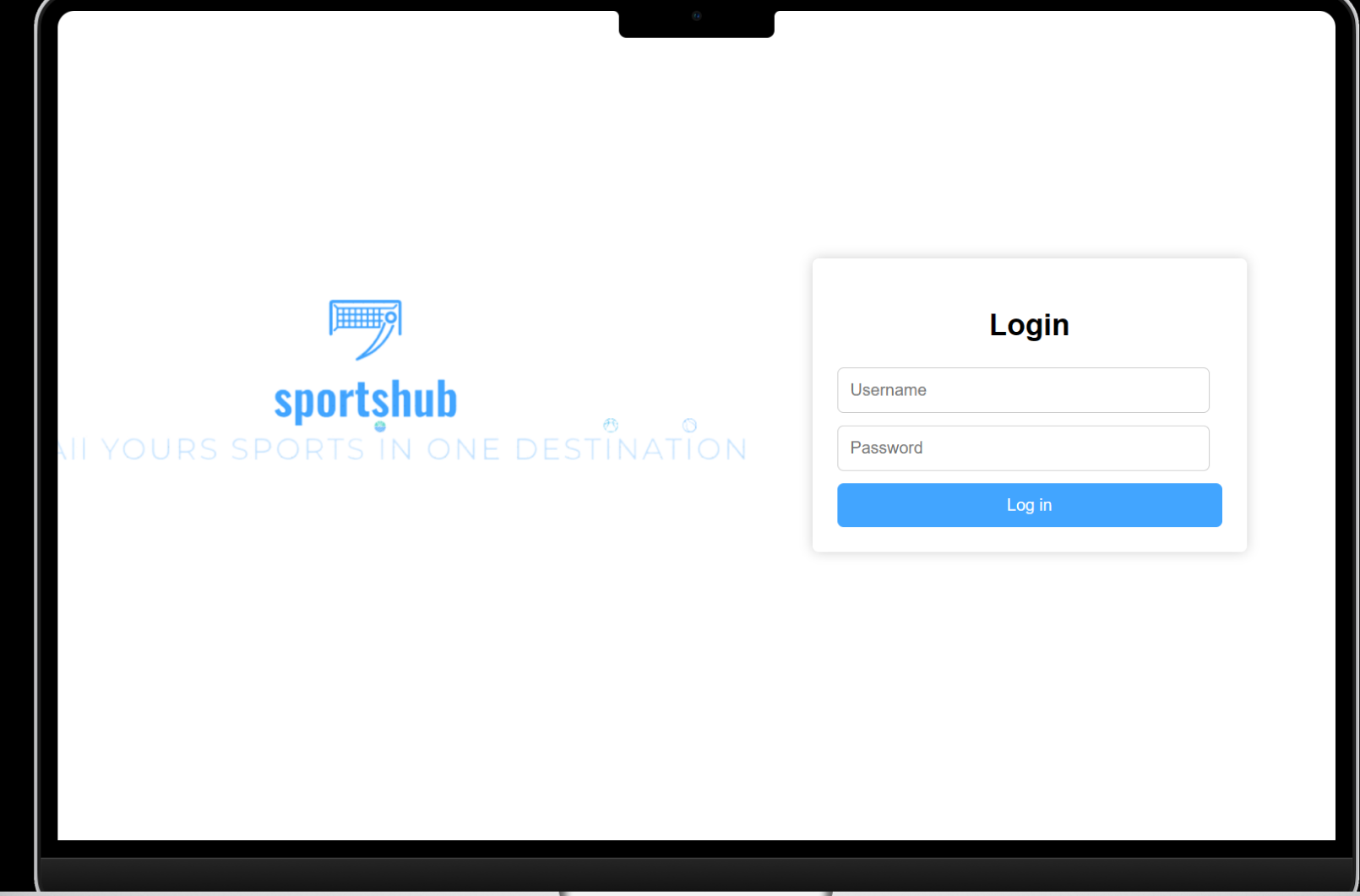


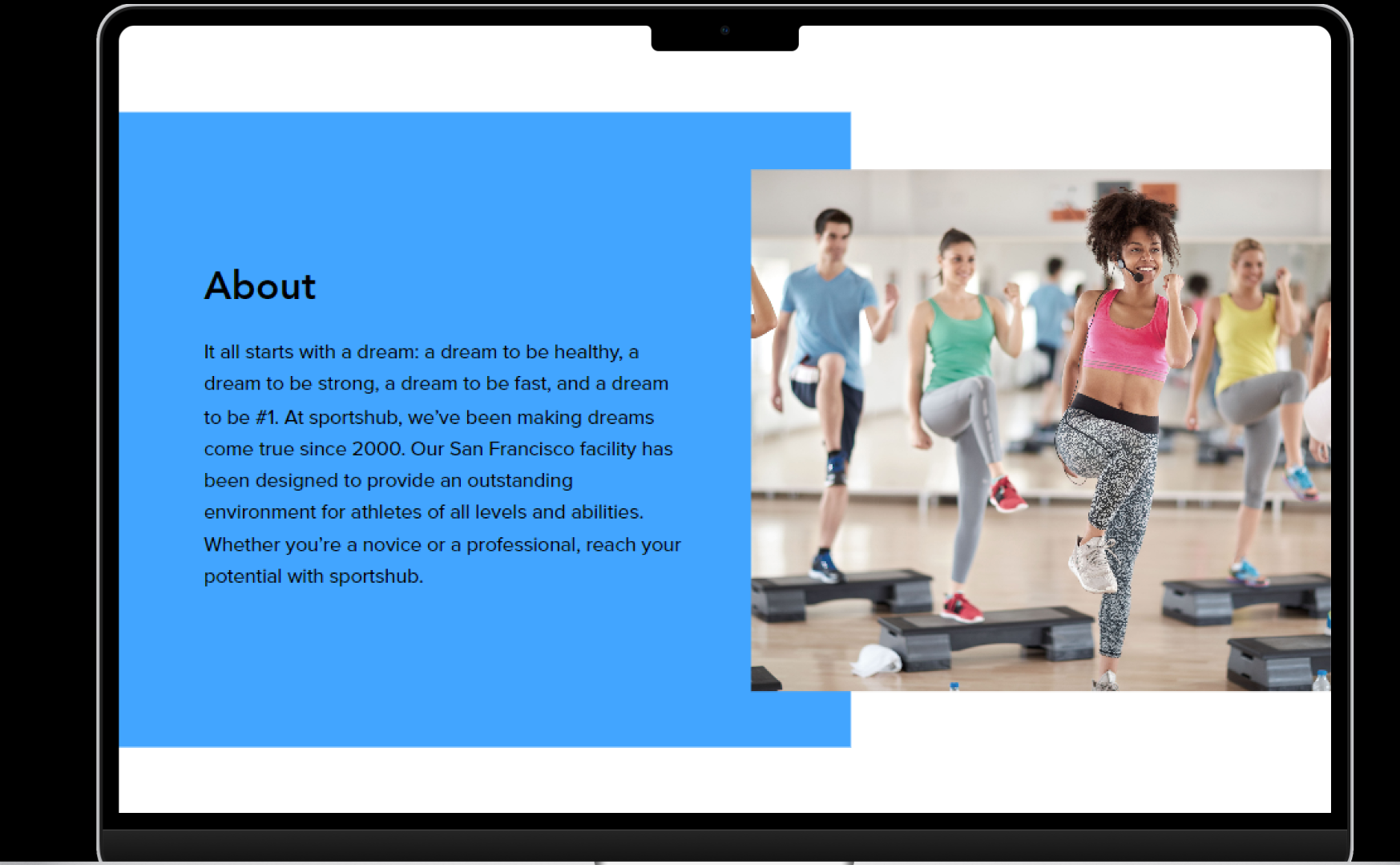
## 4.3 USER INTERFACE DESIGN USING FIGMA

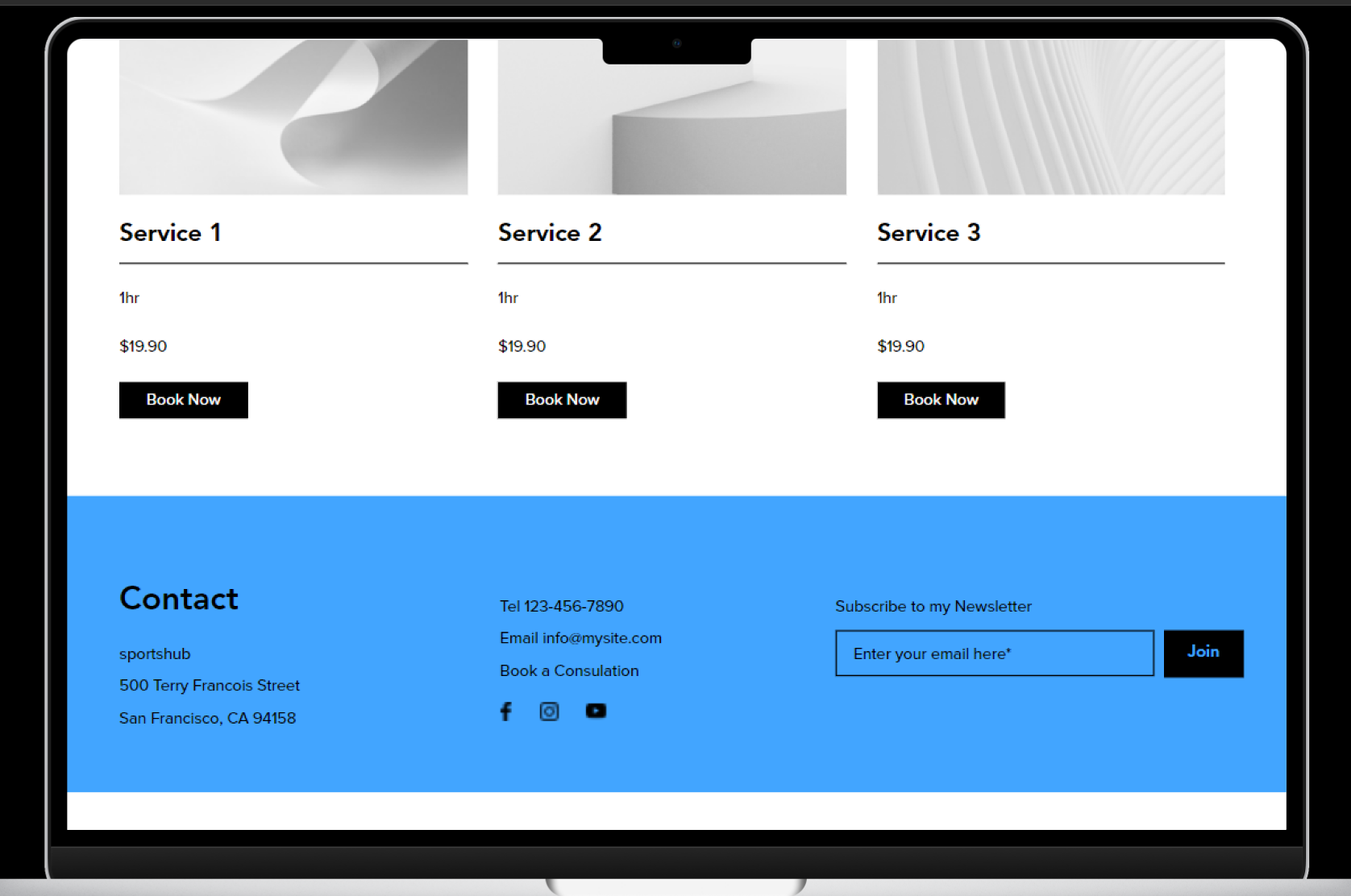
**Form Name: abcc**





**Form Name: abcc**

****

****

Screenshot

All Forms

### 4.4 DATABASE DESIGN

A organized system called a database allows users to retrieve information quickly and easily and has the capacity to retain information. The main objective of any database is its data, which need protection. The database design process is divided into two steps. In order to create a database that as clearly as possible satisfies these objectives, the user requests must first be identified. This stage, called as Information Level Design, is carried out independently from any specific DBMS.

UML is a popular language utilized for modeling systems and objects found in the real world, and one of the tools it employs is use case diagrams to represent their creation and operation. The system architecture is paralleled by a database design.

The database's data layout seeks to achieve the two objectives listed below

* Data independence
* Data Integrity

#### **4.4.1 Relational Database Management System (RDBMS)**

RDBMS stands for Relational Database Management System. All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, My-SQL, and Microsoft Access are based on RDBMS. It is called Relational Database Management System (RDBMS) because it is based on the relational model introduced by E.F. Codd. Data is represented in terms of tuples (rows) in RDBMS.A relational database is the most commonly used database. It contains several tables, and each table has its primary key. Due to a collection of an organized set of tables, data can be accessed easily in RDBMS.

##### Relations, Domains & Attributes

A table is a data structure representing a relation between entities, where the rows in a table, commonly referred to as tuples, are composed of an ordered set of n elements. These elements are organized into columns, also known as attributes, which define the characteristics of the entities. Relationships between tables in a database are established to ensure both referential and entity relationship integrity. To define the domain of a data attribute, a set of atomic values is specified, typically drawn from a particular data type. Assigning a name to the domain aids in interpreting its values and is a common practice.

##### Relationships

* Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
* Entity Integrity enforces that no Primary Key can have null values.
* Referential Integrity enforces that no Primary Key can have null values.
* Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

#### **4.4.2 Normalization**

During the initial phase, data is grouped together in a basic manner to minimize the impact of future changes on data structures. The process of normalizing data structures formally aims to enhance data integrity and reduce duplication by organizing it into logical and efficient groupings. Using the normalization technique, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modelling. In a table, In a table, each row is distinguished by a unique key, which may either be a primary key or a foreign key. A primary key is a single or multiple elements that act as an exclusive identifier for the particular row in the table, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized. It means placing things in their natural form, as the name suggests. By using normalization, the application developer aims to establish a coherent arrangement of the data into appropriate tables and columns, where names may be quickly related to the data by the user. By removing recurring groups from the data, normalization prevents data redundancy, which puts a heavy strain on the computer's resources.

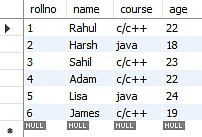
These consist of

* Select appropriate table and column names.
* Normalize the data.
* Choose the proper name for the data.

##### First Normal Form(1NF)

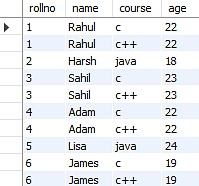
* A table is considered to be in the First Normal Form if it satisfies the requirement of atomicity, which means that the table's values cannot be divided into smaller, more granular parts, and each attribute or column contains only a single value. In other words, each column should hold only one piece of information, and there should be no repeating groups or arrays of values within a single row.
* The concept of atomicity in this context denotes that a cell within a database cannot contain more than one value, and must exclusively store a single-valued attribute.
* The first normal form in database normalization places limitations on attributes that have multiple values, are composed of multiple sub-attributes, or contain a combination of both. Therefore, in order to satisfy the conditions of the first normal form., these attributes need to be modified or removed, a relation must not have multi-valued or composite attributes, and any such attributes must be split into individual attributes to form atomic values.

E.g. The table contains information pertaining to students, including their roll number, name, course of study, and age.



###### Figure 12 : Table not in 1NF

The table containing the records of students displays a violation of the First Normal Form due to the presence of two distinct values in the course column. To ensure compliance with the First Normal Form, the table has been modified resulting in the formation of a new table.



###### Figure 13 : Table in 1NF

The First Normal Form is applied to achieve atomicity in the system, which ensures that each column has unique values. By following this normalization process, the data is organized into individual atomic values, eliminating any redundancies or repeating groups. As a result, data integrity is maintained, and the system can efficiently handle complex data manipulations and updates.

##### Second Normal Form(2NF)

To meet requirements of Second Normal Form, a table must satisfy the criteria of First Normal Form as a prerequisite. Furthermore, the table must not exhibit partial dependency, which refers to a scenario where a non-prime attribute is dependent on a proper subset of the candidate key. In other words, the table should have no attributes that are determined by only a portion of the primary key.

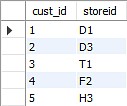
Now understand the Second Normal Form with the help of an example.

Consider the table Location:



###### Figure 14 : Table not in 2NF

The Location table currently has a composite primary key consisting of cust id and storied, and its non-key attribute is store location. However, the store location attribute is directly determined by the storied attribute, which is part of the primary key. As a result, this table does not meet the requirements of second normal form. To address this issue and ensure second normal form is met, it is necessary to separate the Location table into two separate tables. This will result in the creation of two distinct tables that accurately represent the relevant data and relationships: one for customer IDs and store IDs, and another for store IDs and their respective locations.:



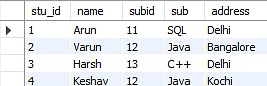
###### Figure 15 : Table in 2NF

After eliminating the partial functional dependency in the location table, it can be observed that the column storing the location is now entirely dependent on the primary key of the same table. In other words, the location column is fully determined by the primary key, thereby ensuring greater data consistency and integrity in the table.

##### Third Normal Form

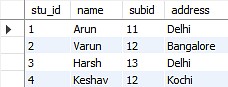
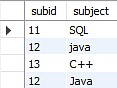
A table must meet the requirements of Second Normal Form in order to be considered in Third Normal Form, and also fulfill two additional conditions. The second condition states that non- prime attributes should not rely on non-prime characteristics that are not a part of the candidate key within the same table, thus avoiding transitive dependencies. A transitive dependency arises when A → C indirectly, due to A → B and B → C, where B is not functionally dependent on A. The main objective of achieving Third Normal Form is to reduce data redundancy and guarantee data integrity.

For instance, let's consider a student table that includes columns like student ID, student name, subject ID, subject name, and address of the student. To comply with the requirements for Third Normal Form, this table must first meet the standards of Second Normal Form and then ensure that there are no transitive dependencies between non-prime attributes.



###### Figure 16 : Table not in 3NF

Now to change the table to the third normal form, you need to divide the table as shown below: Based on the given student table, it can be observed that the stu\_id attribute determines the sub\_id attribute, and the sub\_id attribute determines the subject (sub). This implies that there is a transitive functional dependency between stu\_id and sub. As a result, the table does not satisfy the criteria for the third normal form. To adhere to the third normal form, the table must be divided into separate tables where each table represents a unique entity or relationship. In this case, the table can be divided into two tables: one for the student-subject relationship (stu\_id and sub\_id), and another for the subject information (sub\_id and sub):

###### Figure 17 : Table in 3NF

The two tables illustrate that the non-key attributes are completely determined by and reliant on the primary key. In the first table, the columns for name, sub\_id, and addresses are all exclusively linked to the stu\_id. Likewise, the second table demonstrates that the sub column is entirely dependent on the sub\_id.

#### **4.4.3 Sanitization**

Data Sanitization involves the secure and permanent erasure of sensitive data from datasets and media to guarantee that no residual data can be recovered even through extensive forensic analysis. Data sanitization has a wide range of applications but is mainly used for clearing out end-of-life electronic devices or for the sharing and use of large datasets that contain sensitive information. The main strategies for erasing personal data from devices are physical destruction, cryptographic erasure, and data erasure. While the term data sanitization may lead some to believe that it only includes data on electronic media, the term also broadly covers physical media, such as paper copies. These data types are termed soft for electronic files and hard for physical media paper copies. Data sanitization methods are also applied for the cleaning of sensitive data, such as through heuristic-based methods, machine-learning based methods, and k-source anonymity.

#### **4.4.4 Indexing**

Indexing is used to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed. The index is a type of data structure. It is used to locate and access the data in a database table quickly.

* Primary Index − Primary index is defined on an ordered data file. The data file is ordered on a key field. The key field is generally the primary key of the relation.
* Secondary Index − Secondary index may be generated from a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values.

Clustering Index − Clustering index is defined on an ordered data file. The data file is ordered on a non-key field.

### 4.5 TABLE DESIGN

#### **1.tbl\_login**

Primary key: **login**\_**id**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | username | varchar(100) | Unique | Username |
| 3 | email | varchar(254) | Unique | Email of the user |
| 4 | first\_name | varchar(100) | Not Null | First name |
| 5 | last\_name | varchar(100) | Not Null | Last name |
| 6 | role | varchar(20) | Not Null | User role (e.g., "FitnessUser", "FitnessTrainer", "SportsTrainer") |
| 7 | description | text | Blank | Description (optional) |
| 8 | profile\_picture | varchar(100) | Blank | Profile picture filename (optional) |
| 9 | address | text | Blank | Address (optional) |
| 10 | contact\_number | varchar(15) | Blank | Contact number (optional) |
| 11 | second\_contact\_number | varchar(15) | Blank | Second contact number (optional) |
| 12 | is\_active | boolean | Default=True | User is active or not (default is active) |
| 13 | is\_staff | boolean | Default=False | User is a staff member or not (default is not staff) |
| 14 | last\_login | datetime | Blank | Last login timestamp |
| 15 | date\_joined | datetime | Not Null | Date joined (registration date) |

#### **2. tbl\_role**

Primary Key: id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | name | varchar(20) | Unique | Role name |

**3. tbl\_commonchoice**

Primary Key: id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | name | varchar(255) |  | Display name of the choice |
| 3 | choice\_type | varchar(25) |  | Type of choice (e.g., "Fitness Goal", "Trainer Specialization") |

**4. tbl\_roleapplication**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

Foreign Key: role\_id references table tbl\_role

Foreign Key: fitness\_goal\_id references table tbl\_commonchoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | role\_id | int(11) | Foreign Key | References table tbl\_role |
| 4 | specialization\_details | text | Blank | Specialization details (optional) |
| 5 | is\_approved | boolean | Default=False | Application approval status |
| 6 | fitness\_goal\_id | int(11) | Foreign Key | References table tbl\_commonchoice |
| 7 | height | float | Blank | Height (optional) |
| 8 | weight | float | Blank | Weight (optional) |
| 9 | experience | int(11) | Blank | Experience (optional) |
| 10 | certification | varchar(255) |  | Certification |
| 11 | certification\_link | varchar(255) | Blank | Certification link (optional) |

**5. tbl\_sportscenter**

Primary Key: id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | name | varchar(100) | Not Null | Sports Center name |
| 3 | location | varchar(200) | Not Null | Location of the sports center |
| 4 | description | text | Blank | Description of the sports center (optional) |
| 5 | image | varchar(100) | Blank, Null | Image filename of the sports center (optional) |
| 6 | price\_per\_slot | decimal(10, 2) | Blank, Null | Price per slot (optional) |

**2. tbl\_sportscenterslot**

**Primary Key: id**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | start\_time | time | Not Null | Start time |
| 3 | end\_time | time | Not Null | End time |

**3. tbl\_reservation**

Primary Key: id

Foreign Key: reserver\_id references table tbl\_customuser

Foreign Key: sport\_id references table tbl\_sportscenter

Foreign Key: slot\_id references table tbl\_sportscenterslot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | reserver\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | sport\_id | int(11) | Foreign Key | References table tbl\_sportscenter |
| 4 | slot\_id | int(11) | Foreign Key | References table tbl\_sportscenterslot |
| 5 | reservation\_date | date | Not Null | Reservation date |
| 6 | reservation\_time | datetime | Auto-Generated | Reservation timestamp |
| 7 | status | boolean | Default=False | Reservation status |

**4. tbl\_payment**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

Foreign Key: reservation\_id references table tbl\_reservation

Foreign Key: trainingplanassignment\_id references table tbl\_trainingplanassignment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | razorpay\_order\_id | varchar(255) | Not Null | Razorpay order ID |
| 4 | payment\_id | varchar(255) | Not Null | Razorpay payment ID |
| 5 | amount | decimal(8, 2) | Not Null | Amount paid |
| 6 | currency | varchar(3) | Not Null | Currency code (e.g., "INR") |
| 7 | timestamp | datetime | Auto-Generated | Timestamp of the payment |
| 8 | payment\_status | varchar(20) | Default=Pending | Payment status |
| 9 | reservation\_id | int(11) | Foreign Key | References table tbl\_reservation (optional) |
| 10 | trainingplanassignment\_id | int(11) | Foreign Key | References table tbl\_trainingplanassignment (optional) |

**4. tbl\_roleapplication**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

Foreign Key: role\_id references table tbl\_role

Foreign Key: fitness\_goal\_id references table tbl\_commonchoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | role\_id | int(11) | Foreign Key | References table tbl\_role |
| 4 | specialization\_details | text | Blank | Specialization details (optional) |
| 5 | is\_approved | boolean | Default=False | Application approval status |
| 6 | fitness\_goal\_id | int(11) | Foreign Key | References table tbl\_commonchoice |
| 7 | height | float | Blank | Height (optional) |
| 8 | weight | float | Blank | Weight (optional) |
| 9 | experience | int(11) | Blank | Experience (optional) |
| 10 | certification | varchar(255) |  | Certification |
| 11 | certification\_link | varchar(255) | Blank | Certification link (optional) |

**1. tbl\_usertrainerconnection**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

Foreign Key: trainer\_id references table tbl\_fitnesstrainer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | trainer\_id | int(11) | Foreign Key | References table tbl\_fitnesstrainer |
| 4 | start\_date | date | Not Null | Start date of the connection |
| 5 | end\_date | date | Not Null | End date of the connection |
| 6 | (Other fields) | (Datatype) | (Constraints) | (Description) |

**2. tbl\_trainingplan**

Primary Key: id

Foreign Key: connection\_id references table tbl\_usertrainerconnection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | connection\_id | int(11) | Foreign Key | References table tbl\_usertrainerconnection |
| 3 | name | varchar(255) | Not Null | Training plan name |
| 4 | description | text |  | Description of the plan |
| 5 | hours\_per\_day | int(11) |  | Hours per day for training |
| 6 | days\_per\_week | int(11) |  | Days per week for training |
| 7 | pace | varchar(255) |  | Pace of training plan |

**1. tbl\_fitnessuser**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | fitness\_goal | varchar(255) | Not Null | Fitness goal of the user |
| 4 | height | float | Not Null | Height of the user |
| 5 | weight | float | Not Null | Weight of the user |

**1. tbl\_fitnessuser**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | fitness\_goal | varchar(255) | Not Null | Fitness goal of the user |
| 4 | height | float | Not Null | Height of the user |
| 5 | weight | float | Not Null | Weight of the user |

**1. tbl\_fitnessuser**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | fitness\_goal | varchar(255) | Not Null | Fitness goal of the user |
| 4 | height | float | Not Null | Height of the user |
| 5 | weight | float | Not Null | Weight of the user |

**1. tbl\_sportstrainer**

**Primary Key: id**

**Foreign Key: user\_id references table tbl\_customuser**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | specialization | varchar(255) | Not Null | Specialization of the trainer |
| 4 | height | float | Blank, Null | Height of the trainer (optional) |
| 5 | weight | float | Blank, Null | Weight of the trainer (optional) |
| 6 | (Other fields) | (Datatype) | (Constraints) | (Description) |

**2. tbl\_fitnesstrainer**

**Primary Key: id**

**Foreign Key: user\_id references table tbl\_customuser**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | experience | int(11) | Not Null | Experience of the trainer |
| 4 | certification | varchar(255) |  | Trainer's certification |
| 5 | training\_goal | varchar(255) | Not Null | Training goal of the trainer |
| 6 | certification\_link | varchar(255) | Blank, Null | Certification link (optional) |
| 7 | height | float | Blank, Null | Height of the trainer (optional) |
| 8 | weight | float | Blank, Null | Weight of the trainer (optional) |

**2. tbl\_traineruserconnection**

Primary Key: id

Foreign Key: fitness\_trainer\_id references table tbl\_fitnesstrainer

Foreign Key: fitness\_user\_id references table tbl\_fitnessuser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | fitness\_trainer\_id | int(11) | Foreign Key | References table tbl\_fitnesstrainer |
| 3 | fitness\_user\_id | int(11) | Foreign Key | References table tbl\_fitnessuser |
| 4 | status | varchar(10) | Default=pending | Connection status (pending, approved, rejected) |
| 5 | certification\_link | varchar(255) | Blank, Null | Certification link (optional) |

**2. tbl\_fitnesstrainer**

Primary Key: id

Foreign Key: user\_id references table tbl\_customuser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | user\_id | int(11) | Foreign Key | References table tbl\_customuser |
| 3 | experience | int(11) | Not Null | Experience of the trainer |
| 4 | certification | varchar(255) |  | Trainer's certification |
| 5 | training\_goal | varchar(255) | Not Null | Training goal of the trainer |
| 6 | certification\_link | varchar(255) | Blank, Null | Certification link (optional) |
| 7 | height | float | Blank, Null | Height of the trainer (optional) |
| 8 | weight | float | Blank, Null | Weight of the trainer (optional) |

**2. . tbl\_trainingplan**

Primary Key: id

Foreign Key: created\_by\_trainer\_id references table tbl\_fitnesstrainer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | plan\_name | varchar(255) | Not Null | Training plan name |
| 3 | description | text |  | Description of the plan |
| 4 | duration | int(11) |  | Duration of the plan (in days) |
| 5 | created\_by\_trainer\_id | int(11) | Foreign Key | References table tbl\_fitnesstrainer |
| 6 | amount | decimal(10, 2) | Blank, Null | Amount (optional) |

**12. tbl\_equipment**

Primary Key: id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | name | varchar(255) | Not Null | Equipment name |
| 3 | description | text |  | Equipment description |

**3. tbl\_timeslot**

Primary Key: id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | slot\_number | int(11) | Unique | Slot number (unique) |
| 3 | start\_time | time | Not Null | Start time of the slot |
| 4 | end\_time | time | Not Null | End time of the slot |

**14. tbl\_equipmentreservation**

Primary Key: id

Foreign Key: trainer\_id references table tbl\_fitnesstrainer

Foreign Key: fitness\_user\_id references table tbl\_fitnessuser

Foreign Key: equipment\_id references table tbl\_equipment

Foreign Key: timeslot\_id references table tbl\_timeslot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Fieldname** | **Datatype** | **Key Constraints** | **Description** |
| 1 | id | int(11) | Primary Key | Primary Key |
| 2 | trainer\_id | int(11) | Foreign Key | References table tbl\_fitnesstrainer |
| 3 | fitness\_user\_id | int(11) | Foreign Key | References table tbl\_fitnessuser |
| 4 | equipment\_id | int(11) | Foreign Key | References table tbl\_equipment |
| 5 | timeslot\_id | int(11) | Foreign Key | References table tbl\_timeslot |
| 6 | date | date | Default=Today | Reservation date |

# CHAPTER 5

# SYSTEM TESTING

### 5.1 INTRODUCTION

Software testing is a thorough method of analysing a piece of software's performance to see if it performs as planned. This procedure, which is also known as verification and validation, entails evaluating a product to make sure it complies with relevant standards and meets user needs. Software testing is accomplished using a variety of techniques, including reviews, analyses, inspections, and walkthroughs, as well as associated processes like static analysis. Static analysis, which evaluates source code without running it, and dynamic analysis, which keeps track of programme behaviour while it is in use and produces information like execution traces, timing profiles, and test coverage statistics. The various tasks that make up testing can be pre-organized and completed in a methodical manner. The testing process for computer-based systems begins with separate modules and moves forward through system integration. There are many laws that can be used as testing objectives, and testing is necessary for the system testing objectives to be successful.

### 5.2 TEST PLAN

A test plan defines the steps necessary to carry out different testing approaches, as well as the tasks that must be accomplished. The task of generating computer programmes, documentation, and data structures falls under the purview of software developers. Individual testing is required to make sure that every part of the programme operates as planned. An independent test group (ITG) is ready to give developers feedback and spot any potential problems. The test strategy must include a quantification of the testing objectives. These goals may include measurements like the mean time to failure, the cost of fixing errors, the density of remaining errors, the frequency of recurrence, and the number of hours needed for regression testing. The testing levels could include:

• Unit testing

* Integration Testing
* Validation Testing or System Testing
* Output Testing or User Acceptance Testing
* Automation Testing
* Selenium Testing

#### 5.2.1 Unit Testing

Unit testing is a crucial stage of software verification that concentrates on testing individual components or modules, which are the fundamental building blocks of software design. This testing process entails reviewing the component-level design specifications to detect any errors within the module's boundaries by analyzing key control paths. The complexity of the test and the untested areas are determined during the process of unit testing, which is designed to be white-box focused, and multiple components can be tested simultaneously. To ensure proper functionality of the program unit being tested, the modular interface undergoes checks to ensure correct data flow in and out. Additionally, the integrity of temporarily stored data in the local data structure is examined throughout the algorithm's execution. The evaluation of boundary conditions confirms that every statement within the module has been executed at least once. Finally, inspection of each error management path is conducted to ensure proper error handling.

Performing data flow tests across module interfaces prior to any other testing is crucial for ensuring the effectiveness of the testing process. This is because if data cannot flow in and out of the system correctly, all other tests will be useless. Error handling channels must be established, and fault scenarios must be anticipated during the design stage to ensure that the system can redirect or stop working when an error occurs. The final stage of unit testing is boundary testing, which involves testing the software at its boundaries. This is because software often fails at its boundaries.

#### 5.2.2 Integration Testing

Integration testing is a rigorous procedure that entails creating the structure of a program and executing tests to identify any problems related to the connections between different components. The objective is to establish a program structure based on design, utilizing components that have undergone unit testing. The overall program is then tested as a whole. However, correcting any issues can be challenging since the program's size makes it difficult to isolate the root causes. Once these errors are corrected, new ones may emerge, and the process may seem to repeat itself endlessly. After completing unit testing on all modules, they are integrated into the system to verify that there are no interface inconsistencies. This integration process also leads to the development of a distinctive program structure, as any discrepancies in the program structures are eliminated.

#### 5.2.3 Validation Testing or System Testing

After conducting the testing phase, the entire system, including its code, modules, and class modules, was thoroughly examined using a process known as system tests or black box testing. Black box testing specifically aims to ensure that the software's functional requirements are met by creating input conditions that replicate all possible scenarios. The objective of this testing procedure is to detect and address a range of issues spanning from inadequate or inaccurate functionalities, interface discrepancies, errors in data arrangement or external data retrieval, performance drawbacks, initialization malfunctions, to termination glitches.

#### 5.2.4 Output Testing or User Acceptance Testing

The system under consideration is tested for user acceptance; in this case, it must satisfy the business' requirements. The programme should consult the user and the perspective system while it is being developed in order to make any necessary adjustments. This was accomplished in regards to the following areas : • Input Screen Designs

• Output Screen Designs

A variety of test data are used to conduct the aforementioned tests. The process of system testing requires the preparation of test data. After the preparation of sample data, the system being analyzed is put to the test using that data. Test data errors are discovered once more and resolved using the testing techniques mentioned above when the system is tested. Additionally, the fixes are noted for future use.

#### 5.2.5 Automation Testing

Automation testing is a software testing process that verifies whether a software product meets specific requirements by automatically executing tests. The main purpose of automation testing is to detect bugs, defects, and other issues in software products and ensure that they perform as designed.

Benefits of using automation testing for software development:

* Detailed reporting capabilities
* Improved bug detection
* Simplifies testing
* Speeds up the testing process
* Reduces human intervention

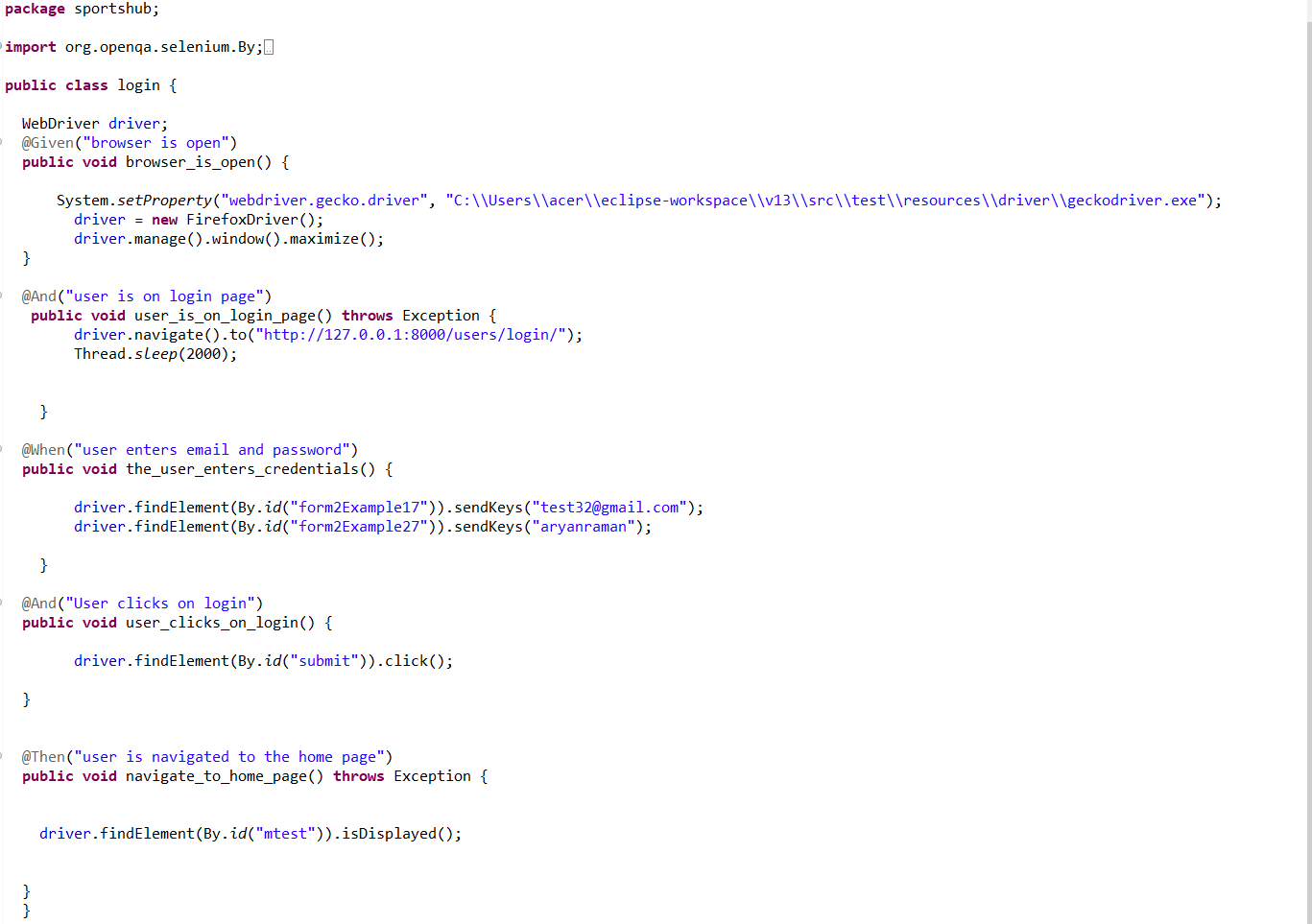
#### 5.2.6 Selenium Testing

Selenium is a widely used open-source automation testing suite for web UI. Originally developed by Jason Huggins in 2004, Selenium supports automation across different browsers, platforms, and programming languages. It can be deployed on Windows, Linux, Solaris, Macintosh, and even supports mobile OS such as iOS, Windows Mobile, and Android. Selenium supports various programming languages through drivers specific to each language including C#, Java, Perl, PHP, Python, and Ruby. Selenium Web Driver is the most popular with Java and C#. Test scripts can be coded in any supported language and run directly in modern web browsers such as Internet Explorer, Mozilla Firefox, Google Chrome, and Safari. Selenium is not only used for functional tests but can also be integrated with automation test tools like Maven, Jenkins, and Docker for continuous testing. Furthermore, it can be integrated with TestNG and JUnit for managing test cases and generating reports.

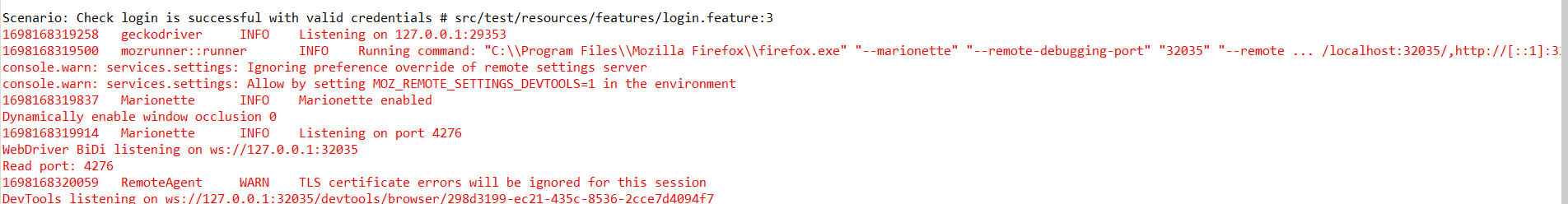
**Example:**

**Test Case 1**

**Code**



**Eg.Screenshot**



**Eg.Test Report**

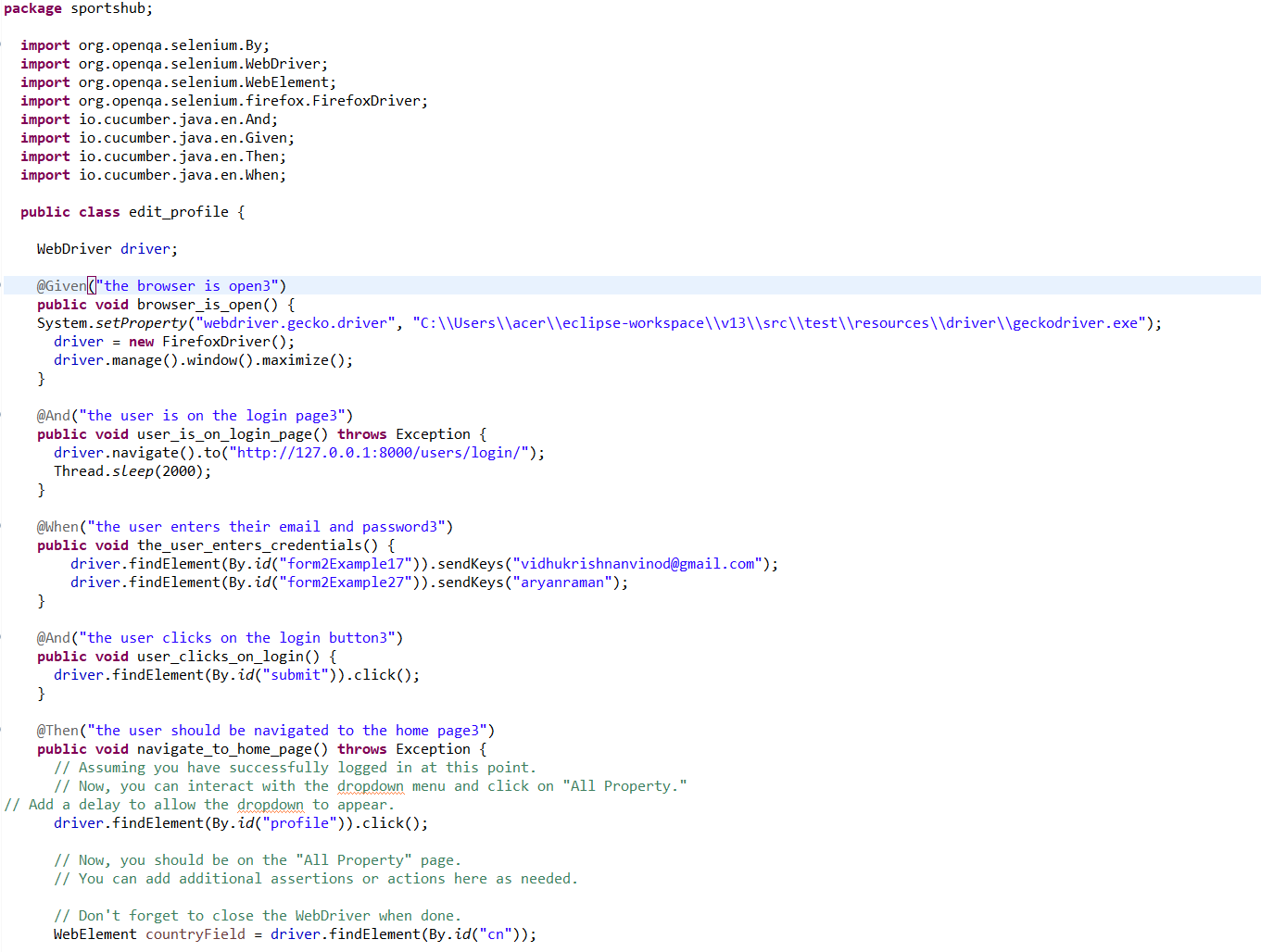
**Test Report**

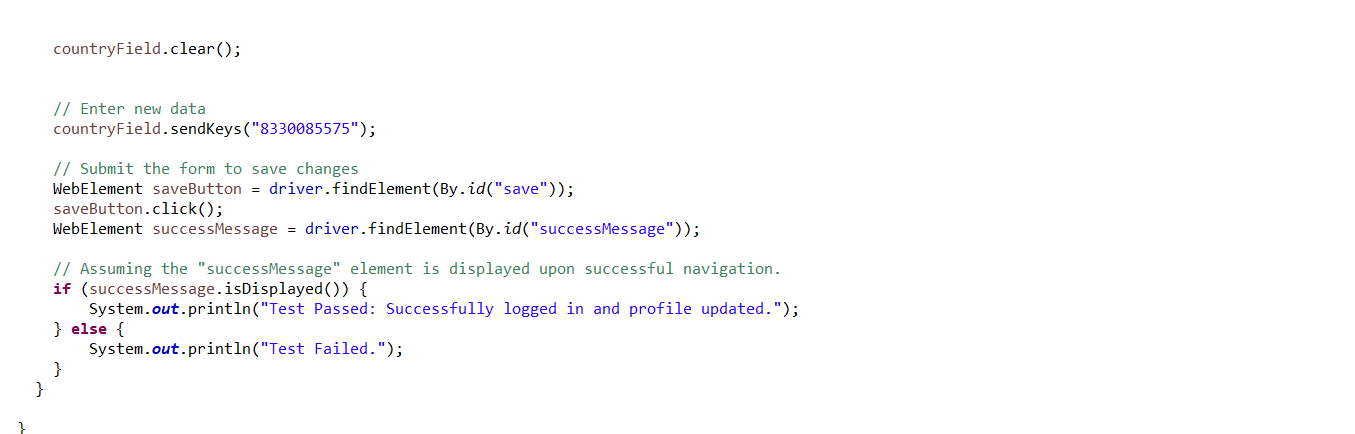
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name : SoulCure** | | | | | |
| **Login Test Case** | | | | | |
| **Test Case ID:** 1 | | | **Test Designed By: Amal Raj** | | |
| **Test Priority**  **(Low/Medium/High):** High | | | **Test Designed Date: 23**-10-2023 | | |
| **Module Name**: Login Page | | | **Test Executed By: Dr.Paulin Paul** | | |
| **Test Title:** Verify login with valid email and password | | | **Test Execution Date: 23**-10-2023 | | |
| **Description:** Test the Login Page | | |  | | |
| **Pre-Condition:** User has valid email id and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/Fail**  **)** |
| 1 | Navigation to Login Page |  | Login Page should be displayed | Login page displayed | Pass |
| 2 | Provide Valid email | User  Name: test32@gmail.com" | User should be able to Login | User Logged in and navigated to the dashboard with records | Pass |
| 3 | Provide  Valid  Password | Password:  aryanraman |
| 4 | Click on Sign In button |  |
| **Post-Condition:** User is validated with database and successfully login into account. The Account session details are logged in database | | | | | |

**Test Case 2:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name : SoulCure** | | | | | |
| **Login Test Case** | | | | | |
| **Test Case ID:** 1 | | | **Test Designed By: Amal Raj** | | |
| **Test Priority**  **(Low/Medium/High):** High | | | **Test Designed Date: 23**-10-2023 | | |
| **Module Name**: Login Page | | | **Test Executed By: Dr.Paulin Paul** | | |
| **Test Title:** Verify login with valid email and password | | | **Test Execution Date: 23**-10-2023 | | |
| **Description:** Test the Login Page | | |  | | |
| **Pre-Condition:** User has valid email id and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/Fail**  **)** |
| 1 | Navigation to Login Page |  | Login Page should be displayed | Login page displayed | Pass |
| 2 | Provide Valid email | User  Name: test32@gmail.com" | User should be able to Login | User Logged in and navigated to the dashboard with records | Pass |
| 3 | Provide  Valid  Password | Password:  aryanraman |
| 4 | Click on Sign In button |  |
| **Post-Condition:** User is validated with database and successfully login into account. The Account session details are logged in database | | | | | |

**Code**

****

****

**Screenshot**

**Test report**

# CHAPTER 6

# IMPLEMENTATION

## INTRODUCTION

During the implementation stage of a project, the theoretical design is transformed into a functional system, which is essential in achieving a successful outcome and user confidence in the system's effectiveness and accuracy. User training and documentation are the primary focus during this stage, and conversion typically takes place around the same time as user training or afterwards. Implementation involves converting the new system design into an operational one and can create chaos and confusion if not carefully planned or controlled. The implementation process encompasses all activities necessary to convert from the existing system to the new system, whether it be a totally new or modified system. It is crucial to provide a reliable system that meets organizational requirements. The system can only be implemented after thorough testing is done, and it is found to be working according to specifications. Feasibility checks are performed by system personnel, and the more complex the system being implemented, the more involved the system analysis and design effort required for education and training, system testing, and changeover.

The implementation state involves the following tasks:

• Careful planning.

• Investigation of system and constraints.

• Design of methods to achieve the changeover.

## IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

• The active user must be aware of the benefits of using the new system.

• Their confidence in the software is built up.

• Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won’t take place.

### User Training

The goal of user training is to give users the confidence to test and alter computer-based systems in order to finally accomplish the intended goals. Training becomes more important as systems get more complicated. As part of the user training process, participants are exposed to a variety of crucial tasks like data entering, handling error alerts, querying databases, calling up routines to generate reports, among others.

### Training on the Application Software

The new application software requires users to first complete a basic computer literacy training course before utilizing it. They should be shown how to access help resources, traverse the software panels, correct entry errors, and update data that has already been entered. Specific topics pertaining to the user group and their function in using the system or its components should also be covered in the training. The training for the programme should be adapted to the requirements of various user groups and hierarchical levels.

### System Maintenance

The software maintenance phase is an essential part of the software development life cycle and occurs after a software product has been successfully implemented and is performing useful work. Maintenance is necessary to ensure that the system remains adaptable to changes in the system environment. While finding mistakes is a part of software maintenance, it is not the only task involved. There are many other activities involved in maintenance, such as updating documentation, fixing bugs, enhancing existing features, and adding new features. Effective maintenance can help ensure that the software remains functional, reliable, and efficient over time.

# CHAPTER 7

# CONCLUSION AND FUTURE SCOPE

## CONCLUSION

## 

.

* 1. **FUTURE SCOPE**

.

# CHAPTER 8

# BIBLIOGRAPHY

### REFERENCES:

* Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
* Roger S Pressman, “*Software Engineering*”, 1994.
* Pankaj Jalote, “So*ftware engineering*: a precise approach”, 2006.
* IEEE Std 1016 Recommended Practice for Software Design Descriptions.

### WEBSITES:

* [www.w3schools.com](http://www.w3schools.com/)
* https:/stackoverflow.com/
* https://www.youtube.com/brototype/
* <https://app.diagrams.net/>

# CHAPTER 9

# APPENDIX

## Sample Code

Main functionalities

## Screen Shots