Subject Discussion Classroom

A Main Project report submitted in partial fulfillment
of requirements for the award of degree of VIII semester

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

V VAMSI KRISHNA (19131A05P6)
` **V AJAY KUMAR (19131A05Q5)**
B HARSHA VARDHAN (19131A0583)
V V V SAKETH (19131A05Q9)

Under the esteemed guidance of
Mrs. K. Swathi
M.Tech
Assistant Professor
Department of Computer Science and Engineering

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING
(AUTONOMOUS)
(Affiliated to JNTU, Kakinada, A.P)
VISAKHAPATNAM - 530048
2022 – 2023

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)

VISAKHAPATNAM-530048

CERTIFICATE

This report on

“Subject Classroom Discussion”

is a bonafide record of the main project work

submitted by

V VAMSI KRISHNA	**(19131A05P6)**
V AJAY KUMAR	**(19131A05Q5)**
B HARSHA VARDHAN	**(19131A0583)**
V V V SAKETH	**(19131A05Q9)**

in their VIII semester in partial fulfillment of the requirements for the Award of degree of **Bachelor of Technology in Computer Science and Engineering** during the academic year 2022-2023.

Signature of the Guide
Mrs. K. SWATHI
Assistant Professor
Department. of CSE
GVP College of Engineering(A)

Signature of the HOD
Dr. D.N.D. HARINI
Associate Professor & HOD
Department of CSE
GVP College of Engineering(A

DECLARATION

We hereby declare that this main project entitled "**Subject Discussion classroom**" is a bonafide work done by us and submitted to the **Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous),** Visakhapatnam, in partial fulfillment for the award of the degree of B. Tech is of our own and it is not submitted to any other university or has been published any time before.

PLACE: VISAKHAPATNAM
DATE:

By

V VAMSI KRISHNA (19131A05P6)

V AJAY KUMAR (19131A05Q5)

B HARSHA VARDHAN (19131A0583)

V V V SAKETH (19131A05Q9)

ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude to our esteemed institute **Gayatri Vidya Parishad College of Engineering (Autonomous)**, which has provided us an opportunity to fulfill our cherished desire.

We express our sincere thanks to our Principal **Dr. A.B. KOTESWARA RAO, Gayatri Vidya Parishad College of Engineering (Autonomous)** for his encouragement to us during this project, giving us a chance to explore and learn new technologies in the form of a main project.

We express our sincere thanks to **Dr. D.N.D. HARINI, Associate Professor and Head of the Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous)**, for giving us an opportunity to do the project in college.

We express our profound gratitude and deep indebtedness to our guide **Mrs. K. Swathi, M.Tech, Assistant Professor, Dept. of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous)**, whose valuable guidance, suggestions and comprehensive assistance helped us a lot in realizing the present project.

We also thank our coordinator, **Dr. CH. SITA KUMARI, Associate Professor, Department of Computer Science and Engineering**, for the kind suggestions and guidance for the successful completion of our project work.

Finally, we are indebted to the teaching and non-teaching staff of the Computer Science and Engineering Department for all their support in completion of our project.

V VAMSI KRISHNA (19131A05P6)
V AJAY KUMAR (19131A05Q5)
B HARSHA VARDHAN (19131A0583)
V V V SAKETH (19131A05Q9)

ABSTRACT

A custom classroom gives students the flexibility to connect with teachers from a distance. The primary purpose of a Classroom is to streamline the process of one-on-one communication between teachers and students and to share the subject materials with the students for the respective subjects. so, we came up with a solution to create a dynamic web app for subject discussion classrooms that creates forums and rooms for students according to section and branches. Forums in the application enable students and teachers to discuss a particular topic remotely. A custom classroom gives students the flexibility to connect with teachers from a distance.

The primary purpose of a Classroom is to streamline the process of one-on-one communication between teachers and students and to share the subject materials with the students for the respective subjects. so, we came up with a solution to create a dynamic web app for subject discussion classrooms that creates forums and rooms for students according to section and branches. Forums in the application enable students and teachers to discuss a particular topic remotely.

KEYWORDS:

- Axios, Localhost, Response, Local Storage, thetransactioncompany, tomcat, servlet.

INDEX

1. INTRODUCTION

Classroom for teachers to share lectures, interesting information, or study-related various technologies videos with students. General information or facts can be shared only during class. A lecturer can teach a lesson during class time and clears the doubts during class time itself. It is done in the workplace such that there is a stand period of time. Subject Discussion Classroom is an application which is useful for both students and teachers. This comprises many features such as one to one communication, Group chat and we have an application in which the teacher doesn't need to send the invitation request to the students such that when a student signs up based on details all he/she will directly join the class.

Subject Discussion Classroom, has a feature in such a way that even if a student logged in even a delay he extracts all the activities that happened until then.

1.1. OBJECTIVE

"Subject Discussion classroom" will be helpful for both students and teachers. The main objective of the project is to make teachers and students work simply for learning even in inconvenient times. It is comfortable to have a chat with the teacher directly and the teacher can be able to have a group conversation to clear the students doubts. So that the scope of learning for students is fulfilled.

1.2. PURPOSE

The main purpose of this project is to help schools and colleges. Where everyone can use it for their discussion purpose for either a group of people. It can be applicable to company people to create a discussion room for project wise, different sectors of a project also too.

1.3. SCOPE

Various organizations can use this system to filter the candidates. Assignment creation and distribution: Teachers can create assignments, distribute them to students, and collect completed work.

Allows teachers and students to communicate with each other through announcements and private messages and provides a centralized location for all class materials, assignments, and communication, making it easy for students to stay organized and access course materials.

2. SRS DOCUMENT

A software requirements specification (**SRS**) is a document that describes what the software will do and how it will be expected to perform

2.1 FUNCTIONAL REQUIREMENTS

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its components. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements are also called Functional Specification.

- Create a web app where the students and Teachers can need to create an account.
- When the student opens his account he is able to see all the classrooms he was enrolled in it and in each classroom there is discussion form, study materials and can talk directly to the teacher.
- In the teacher account when the teacher login the teacher will be displayed all the classes created by that teacher and in each classroom there is discussion form, materials tab where the teacher can add new materials regarding the subject and also the facility to talk to each student separately.

2.2. NON-FUNCTIONAL REQUIREMENTS

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, and Portability. Non-functional requirements are called qualities of a system, there are as follows:

- **Performance**-The average response time of the system is less.
- **Reliability -** The website is highly reliable.
- **Operability -** The interface of the website will be consistent.
- **Efficiency -** Once a user has learned about the system through his interaction, he can perform the task easily.
- **Understandability**-Because of user friendly interfaces, it is more understandable to the users.

2.3. MINIMUM HARDWARE REQUIREMENTS

- **Processor** - Intel Core i5
- **Hard Disk** – 1 TB
- **RAM** - 8GB

2.4. MINIMUM SOFTWARE REQUIREMENTS

- **Programming Languages –** Java 17, HTML, CSS, JavaScript.
- **Database Languages –** Postgres.
- **Operating System -** Windows 10 (64 bit)
- **IDE –** Visual Studio Code
- **Desktop applications –** postman,postgres shell, Any search engine(chrome)
- **Packages –** request,json

3. ANALYSIS

3.1. EXISTING SYSTEM

The traditional system of subject discussion is done offline, where the students can discuss their doubts regarding the subject when the teacher completes the lectures, or if the student gets doubts after class, he has to reach out to the faculty in their free time to get it clarified. In most cases, the materials are shared as a photocopy, the teacher dictates the notes, and the students must take them without any mistakes.

DRAWBACKS OF EXISTING SYSTEM

- It can be done only during standard period of time
- Can't clear doubt or can't share information after period
- Student have to reach the teacher during both of them are having free time
- Difficult to provides the document for every study with in time

3.2. PROPOSED SYSTEM

Proposed Solution consists of two modes of operation Student, Teacher. Every user can have an account based on their requirement it may be teacher or student and they can register

Teacher can have access to create a classroom but in this proposed solution once the classroom is created it takes section as also one of the parameter based on the section it directly add the student who have created account with this section

It allows you to have a group chat discussion along with the teacher and can also have a direct chat with the teacher. Data files like pdfs, youtube links can be stored in the same container to access directly to those files.

ADVANTAGES OF PROPOSED MODEL

- Contains a discussion form where students can discuss their doubts and help each other to clarify their doubts.
- Can post doubts regardless of the time and place and teacher can solve those doubts in their free time.
- Can access the study materials with ease.
- Teacher and student confidentiality is preserved.

3.3. FEASIBILITY STUDY

A feasibility study is an analysis that takes all a project's relevant factors into account including economic, technical, legal, and scheduling considerations to ascertain the likelihood of completing the project successfully. A feasibility study is important and essential to decide whether any proposed project is feasible or not. A feasibility study is simply an assessment of the practicality of a proposed plan or project.

The main objectives of feasibility are mentioned below:

To determine if the product is technically and financially feasible to develop, is the main aim of the feasibility study activity. A feasibility study should provide management with enough information to decide:

- Whether the project can be done.
- To determine how successful your proposed action will be.
- Whether the final product will benefit its intended users.
- To describe the nature and complexity of the project.
- What are the alternatives among which a solution will be chosen (During subsequent phases)
- To analyze if the software meets organizational requirements. There are various types of feasibility that can be determined. They are:

Operational: Define the urgency of the problem and the acceptability of any solution, including people-oriented and social issues: internal issues, such as manpower problems, labor objections, manager resistance, organizational conflicts, and policies; also, external issues, including social acceptability, legal aspects, and government regulations.

Technical: Is the feasibility within the limits of current technology? Does the technology exist at all? Is it available within a given resource?

Economic: Is the project possible with the given resource constraints? Are the benefits that will accrue from the new system worth the costs? What are the savings that will result from the system, including tangible and intangible ones? What are the development and operational costs?

Schedule: Constraints on the project schedule and whether they could be reasonably met.

3.3.1 Economic Feasibility:

Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system. In economic analysis the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. Economic feasibility study related to price, and all kinds of expenditure related to the scheme before the project starts. This study also improves project reliability. It is also helpful for the decision-makers to decide the planned scheme processed later or now, depending on the financial condition of the organization. This evaluation process also studies the price benefits of the proposed scheme.

Economic Feasibility also performs the following tasks.

- Cost of packaged software/ software development.
- Cost of doing full system study.
- Is the system cost Effective?

3.3.2 Technical Feasibility:

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements. The analyst must find out whether current technical resources can be where the expertise of system analysts is beneficial, since using their own experience and their contact with vendors they will be able to answer the question of technical feasibility.

Technical Feasibility also performs the following tasks.

- Is the technology available within the given resource constraints?
- Is the technology have the capacity to handle the solution
- Determines whether the relevant technology is stable and established.
- Is the technology chosen for software development to have a large number of users so that they can be consulted when problems arise, or improvements srequired?

3.3.3 Operational Feasibility:

Operational feasibility is a measure of how well a proposed system solves the problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility refers to the availability of the operational resources needed to extend research results beyond on which they were developed and for which all the operational requirements are minimal and easily accommodated. In addition, the operational feasibility would include any rational compromises farmers make in adjusting the technology to the limited operational resources available to them. The operational Feasibility also perform the tasks like:

- Does the current mode of operation provide adequate response time?
- Does the current of operation make maximum use of resources.
- Determines whether the solution suggested by the software development team is acceptable.
- Does the operation offer an effective way to control the data?

3.4 COST BENEFIT ANALYSIS

The financial and the economic questions during the preliminary investigation are verified to estimate the following:

- The cost of the hardware and software for the class of application being considered.
- The benefits in the form of reduced cost.
- The proposed system will give the minute information, as a result.
- Performance is improved which in turn may be expected to provide increased profits.
- This feasibility checks whether the system can be developed with the available funds.
- This can be done economically if planned judicially, so it is economically feasible.
- The cost of the project depends upon the number of man-hours required.

4. SOFTWARE DESCRIPTION

4.1 VISUAL STUDIO CODE

VS Code came up with several advantages:

- Intelligence for programming languages.
- Command Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET). Palette: Visual studio supports inbuilt Integrated terminal, initially starting at the root of your opened project.
- Integrated Version Control (Built-in Git): Visual Studio Code has Git integration built in, which makes it really easy to instantly see the changes you're making in your project.
- Debugging: One of the key features of Visual Studio Code is its great debugging support.
- Side by Side Editing on different files.
- Some Code management features: Vs code provides us language service features such as Peek Definition, Go to Definition, Find all References, and Rename Symbol. [1]

4.2 React Js

ReactJS is a popular open-source JavaScript library used for building user interfaces. It was developed by Facebook and released in 2013. React is based on the concept of components, which are reusable blocks of code that can be combined to create complex UIs.

React allows developers to build dynamic, high-performance, and scalable web applications with ease. It uses a virtual DOM (Document Object Model) which makes it very efficient in updating the UI without having to reload the entire page.

One of the key advantages of React is its popularity and large community of developers who contribute to the library and provide support. This has resulted in a vast ecosystem of libraries, tools, and resources for React development.

React is often used in conjunction with other libraries and frameworks, such as Redux for state management, Next.js for server-side rendering, and React Native for mobile app development. React has gained widespread adoption in the web development community, with many big companies such as Facebook, Netflix, and Airbnb using it for their web applications.

Axios is a popular JavaScript library that is used for making HTTP requests from a web browser or a Node.js server. It provides an easy-to-use API for performing asynchronous operations such as fetching data from a server, sending data to a server, or handling errors that occur during these operations

Axios can be used in both client-side and server-side applications. It is a lightweight library that can be easily installed via npm or included in a web page via a script tag. [2]

4.3 MAVEN

Maven is a popular build automation tool for Java-based projects. It was developed by the Apache Software Foundation and is used to manage project dependencies, build and package Java code, and automate the entire build process. Maven is an essential tool for Java developers and is widely used in the industry.

One of the key features of Maven is its ability to manage project dependencies. Maven uses a central repository where it stores all the dependencies required by a project. When a project is built, Maven automatically downloads the required dependencies from the repository and adds them to the project's classpath. This makes it easy to manage dependencies and ensure that all required libraries are available for a project to run.

Another important feature of Maven is its ability to automate the build process. Maven provides a standardized build lifecycle that includes different phases such as compile, test, package, install, and deploy. Each phase is executed in a specific order, and Maven automates the process of compiling the code, running tests, creating executable JAR files, and deploying the application to a server.

Maven also provides a standardized project structure that makes it easy to organize and maintain code. Maven uses a directory structure that separates source code, resources, and compiled code. This makes it easy to find and manage files, and helps to keep the code organized and maintainable.

Additionally, Maven supports plugins that can be used to extend its functionality. There are many plugins available for Maven that provide additional features such as generating documentation, analyzing code quality, and deploying applications to cloud-based platforms.

In summary, Maven is a powerful build automation tool that provides a standardized way to manage dependencies, build and package Java code, and automate the entire build process. Its ability to manage dependencies, automate builds, and provide a standardized project structure makes it an essential tool for Java developers. [3]

4.4. PostgreSql

PostgreSQL (often called "Postgres") is a powerful, open-source relational database management system (RDBMS) that is used by developers and organizations around the world. It was first released in 1995 and has since become one of the most popular databases for web applications and other types of software.

PostgreSQL is known for its reliability, stability, and robust feature set. It provides support for complex SQL queries, transactions, and concurrency control, and it can handle large volumes of data with ease. PostgreSQL also provides advanced features like full-text search, JSON support, and advanced indexing.

One of the key advantages of PostgreSQL is its scalability. It can handle large amounts of data and can be configured to run on a single server or a cluster of servers for increased performance and availability. Additionally, PostgreSQL is highly customizable, allowing developers to create their own data types, operators, and functions. [4]

4.5 Java

Java is a programming language and computing platform first released by Sun Microsystems in 1995. It has evolved from humble beginnings to power a large share of today's digital world, by providing the reliable platform upon which many services and applications are built. New, innovative products and digital services designed for the future continue to rely on Java,

Java is also an object-oriented language, which means that it organizes code into reusable components called objects. This makes code more modular, easier to maintain, and more scalable.

Another key feature of Java is its vast ecosystem of libraries and frameworks. Java has a large number of libraries and frameworks for everything from web development to machine learning, making it a versatile language for a wide range of applications.

Java is also heavily used in enterprise software development due to its ability to handle large-scale applications and distributed systems. It is used for building everything from desktop applications to web applications, mobile apps, and server-side applications. [5]

5. PROJECT DESCRIPTION

5.1 PROBLEM DEFINITION

A custom classroom gives students the flexibility to connect with teachers from a distance. The primary purpose of a Classroom is to streamline the process of one-on-one communication between teachers and students and to share the subject materials with the students for the respective subjects

5.2 PROJECT OVERVIEW

The main aim of this project is to create a dynamic web app for subject discussion classroom that creates forums and rooms for students according to section and branches. Forums in the application enable students and teachers to discuss on a particular topic remotely.Along with it the teacher can all the study materials required for that specific subject and the students can access it. There is also an additional feature for students to talk to the teacher separately and vice versa.

Framework of the web application:-

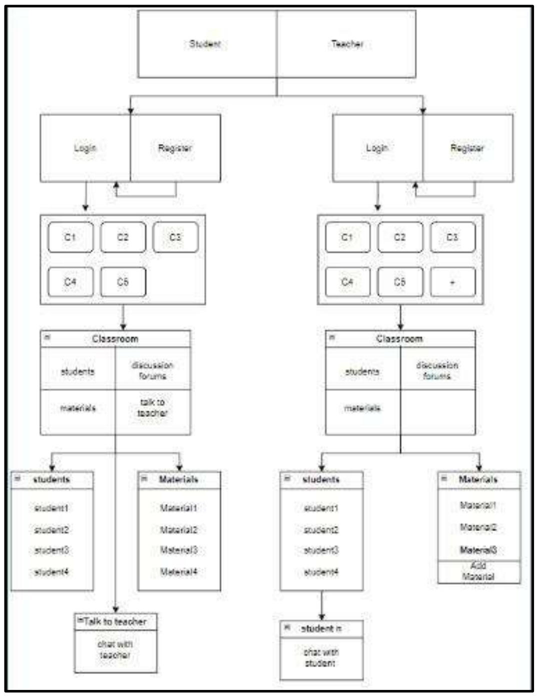


FIG 5.1 Website Design

5.3 MODULE DESCRIPTION

5.3.1. studentRoom.java :

It is one of the main java files where all the API regarding the student features, like sending a message in the discussion form and individually to the particular teacher. It also contains the APIs to display the materials that belong to that specific subject.

5.3.2. teacherRoom.java :

This java file contains code regarding all the API of the teacher features, like displaying the students of the classroom, sending a message in the discussion forum, displaying the discussion forum, and individually texting to the particular student of that classroom. It also contains the APIs to add the study materials and display the materials that belong to that specific subject.

5.3.3. sregister.java,slogin.java :

The sregister java file contains the code to take all the student's details and store them in the database, and the slogin java file contains code to log in to the account based on the details given by the student.

5.3.4. tregister.java,tlogin.java :

The tregister java file contains the code to take all the teacher's details and store them in the database, and the tlogin java file contains code to log in to the account based on the details given by the student.

5.3.5. index.html :

The Template folder contains this index.html module. Since this Subject Discussion classroom is implemented as a webapp, the way that webpage is designed, its styling e.t.c. is specified in this index.html module.

6. SYSTEM DESIGN

6.1 Introduction to UML

Unified Modeling Language **(UML)** is a general-purpose modeling language. The main aim of UML is to define a standard way to **visualize** the way a system has been designed. It is quite like blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system. UML helps software engineers, businessmen, and system architects with modeling, design, and analysis. The Object Management Group (OMG) adopted Unified Modelling Language as a standard in 1997. It's been managed by OMG ever since. The International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically. [6]

Why do we need UML

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So, UML becomes essential to communicate with non-programmers' essential requirements, functionalities, and processes of the system.
- A lot of time is saved down the line when teams can visualize processes, user interactions, and the static structure of the system.

UML is linked with **object-oriented** design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

- **Structural Diagrams –** Capture static aspects or structure of a system. Structural Diagrams include Component Diagrams, Object Diagrams, Class Diagrams, and Deployment Diagrams.
- **Behavior Diagrams –** Capture dynamic aspects or behavior of the system. Behavior Diagrams include Use Case Diagrams, State Diagrams, Activity Diagrams, and Interaction Diagrams.

Building Blocks of the UML Building Blocks of the UML Building Blocks of the UML

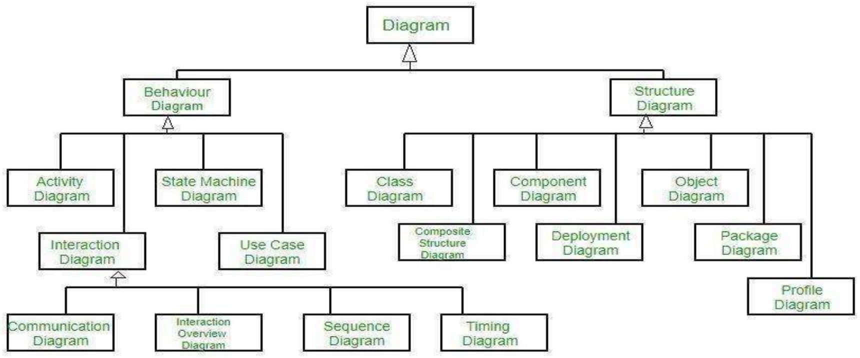


FIG 6.1 BUILDING BLOCKS IN UML

6.2 Building Blocks of the UML

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationships
- Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

6.2.1 Things in the UML

There are four kinds of things in the UML:

- Structural things
- Behavioural things
- Grouping things
- Annotational things

These things are the basic object-oriented building blocks of the UML. You use them to write well-formed models.

6.2.2 Structural Things

Structural things are the nouns of UML models. These are the mostly static parts of a model, representing either conceptual or physical elements. Collectively, the structural things are called classifiers.

A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces. Graphically, a class is rendered as a rectangle, usually including its name, attributes, and operations

Class - A Class is a set of identical things that outlines the functionality and properties of an object. It also represents the abstract class whose functionalities are not defined. Its notation is as follows

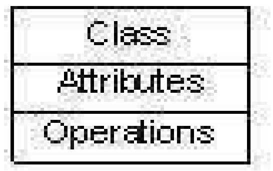


Fig.6.3 Class

Interface - A collection of functions that specify a service of a class or component, i.e.Externally visible behavior of that class.

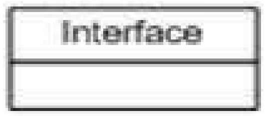


Fig.6.4 Interface

Collaboration - A larger pattern of behaviors and actions. Example: All classes and behaviors that create the modeling of a moving tank in simulation.

Fig.6.5 Collaboration

Use Case - A sequence of actions that a system performs that yields an observable result. Used to structure behavior in a model. Is realized by collaboration.

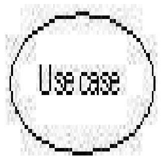


Fig.6.6 Use Case

Component - A physical and replaceable part of a system that implements several interfaces. Example: a set of classes, interfaces, and collaborations.

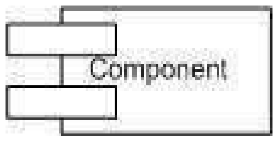


Fig.6.7 Component

Node - A physical element existing at run time and represents are the source

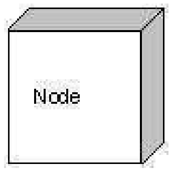


Fig.6.8 Node

6.2.3 Behavioral Things

Behavioral things are the dynamic parts of UML models. These are the verbs of a model, representing behavior over time and space. In all, there are three primary kinds of behavioral things

- Interaction
- State machine

Interaction

It is a behavior that comprises a set of messages exchanged among a set of objects or roles within a particular context to accomplish a specific purpose. The behavior of a society of objects or an individual operation may be specified with an interaction. An interaction involves several other elements, including messages, actions, and connectors (the connection between objects). Graphically, a message is rendered as a directed line, almost always including the name of its operation.

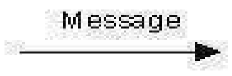


Fig.6.9 Message

State machine

The state machine is a behavior that specifies the sequences of states an object or an interaction goes through during its lifetime in response to events, together with its responses to those events. A state machine may specify the behavior of an individual class or a collaboration of classes. A state machine involves several other elements, including states, transitions (the flow from state to state), events (things that trigger a transition), and activities (the response to a transition). Graphically, a state is rendered as a rounded rectangle, usually including its name and its substrates.

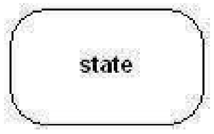


Fig.7.0 State

Grouping Things

Grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available.

Package – Package is the only grouping thing available for gathering structural and behavioral things.

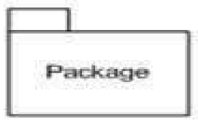


Fig.7.1 Package

6.2.4 Annotation Things

Annotation things are the explanatory parts of UML models. These are the comments you may apply to describe, illuminate, and remark about any element in a model. There is one primary kind of annotation thing, called a note. A note is simply a symbol for rendering constraints and comments attached to an element or a collection of elements.

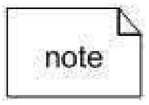


Fig.7.2 Note

6.2.5 Relationships in the UML

The relationship is another most important building block of UML. It shows how the elements are associated with each other and this association describes the functionality of an application.
There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

Dependency

It is an element (the independent one) that may affect the semantics of the other element (the dependent one). Graphically, a dependency is rendered as a dashed line, possibly directed, and occasionally including a label.

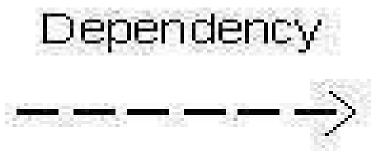


Fig.7.3Dependency

Association

Association is a set of links that connects the elements of a UML model. It also describes how many objects are taking part in that relationship.

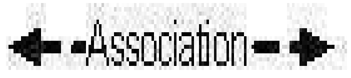


Fig.7.4 Association

Generalization

It is a specialization/generalization relationship in which the specialized element (the child) builds on the specification of the generalized element (the parent). The child shares the structure and the behavior of the parent. Graphically, a generalization relationship is rendered as a solid line with a hollow arrowhead pointing to the parent.

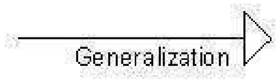
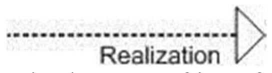


Fig.7.5 Generalization

Realization

Realization can be defined as a relationship in which two elements are connected. One element describes some responsibility, which is not implemented and the other one



implements them. This relationship exists in the case of interfaces.

Fig.7.6 Realization

6.3 UML DIAGRAMS

6.3.1 CLASS DIAGRAM:

The class diagram can be used to show the classes, relationships, interface, association, and collaboration. UML is standardized in class diagrams.

The main purpose to use class diagrams are:

• This is the only UML which can appropriately depict various aspects of OOPs concept.

• Proper design and analysis of application can be faster and efficient.

• Each class is represented by a rectangle having a subdivision of three compartments name, attributes and operation.

• There are three types of modifiers which are used to decide the visibility of attributes and operations.

• + is used for public visibility(for everyone)

• – is used for private visibility (for only me)

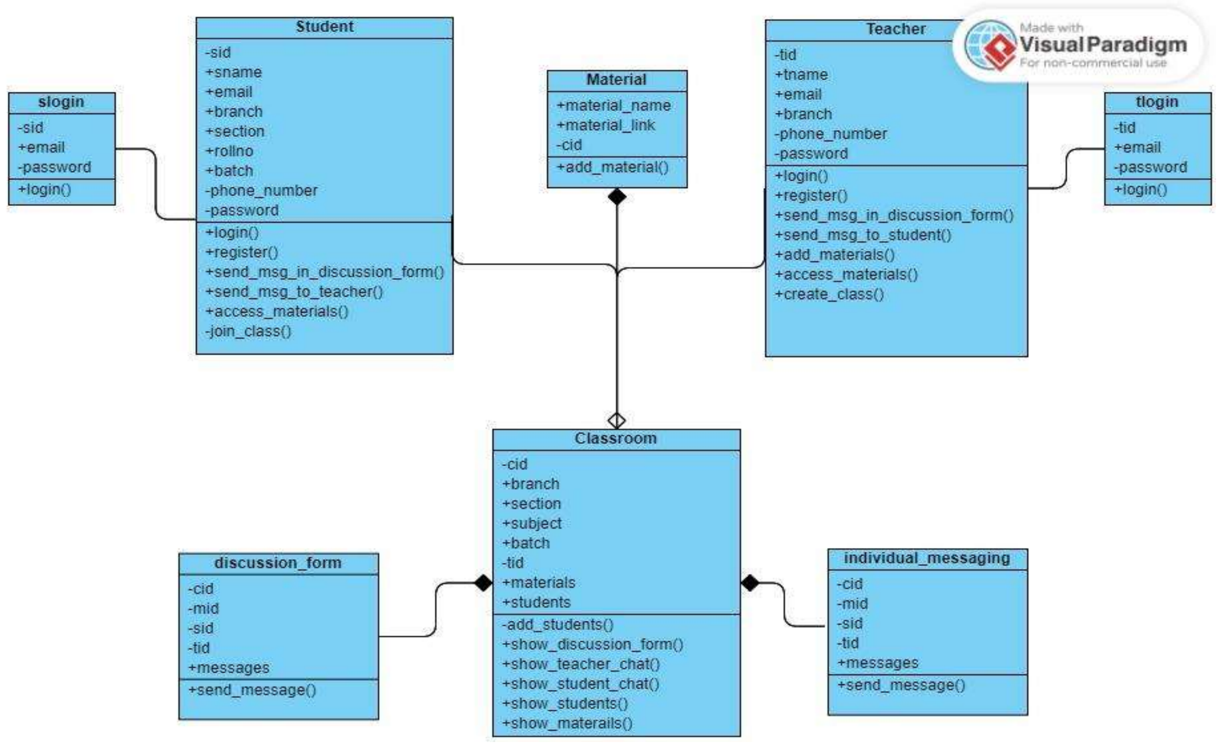


Fig.7.7 Class Diagram

6.3.2 INTERACTION DIAGRAMS:

From the term Interaction, it is clear that the diagram is used to describe some type of interactions among the different elements in the model. This interaction is a part of dynamic behavior of the system.

This interactive behavior is represented in UML by two diagrams known as Sequence diagram and Collaboration diagram. The basic purpose of both the diagrams are similar. Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages. The purpose of interaction diagrams is to visualize the interactive behavior of the system. Visualizing the interaction is a difficult task. Hence, the solution is to use different types of models to capture the different aspects of the interaction.

Sequence and collaboration diagrams are used to capture the dynamic nature but from a different angle.

The purpose of interaction diagram is –

• To capture the dynamic behaviour of a system.

• To describe the message flow in the system.

• To describe the structural organization of the objects.

• To describe the interaction among objects. The main purpose of both the diagrams are similar as they are used to capture the dynamic behavior of a system.

However, the specific purpose is more important to clarify and understand. Sequence diagrams are used to capture the order of messages flowing from one object to another. Collaboration diagrams are used to describe the structural organization of the objects taking part in the interaction. A single diagram is not sufficient to describe the dynamic aspect of an entire system, so a set of diagrams are used to capture it as a whole. Interaction diagrams are used when we want to understand the message flow and the structural organization. Message flow means the sequence of control flow from one object to another.

6.3.2.1 SEQUENCE DIAGRAM:

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

Purpose of Sequence Diagram:

- Model high-level interaction between active objects in a system.
- Model the interaction between objects within a collaboration that realizes an operation.
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of an interaction (showing just one path through the interaction).

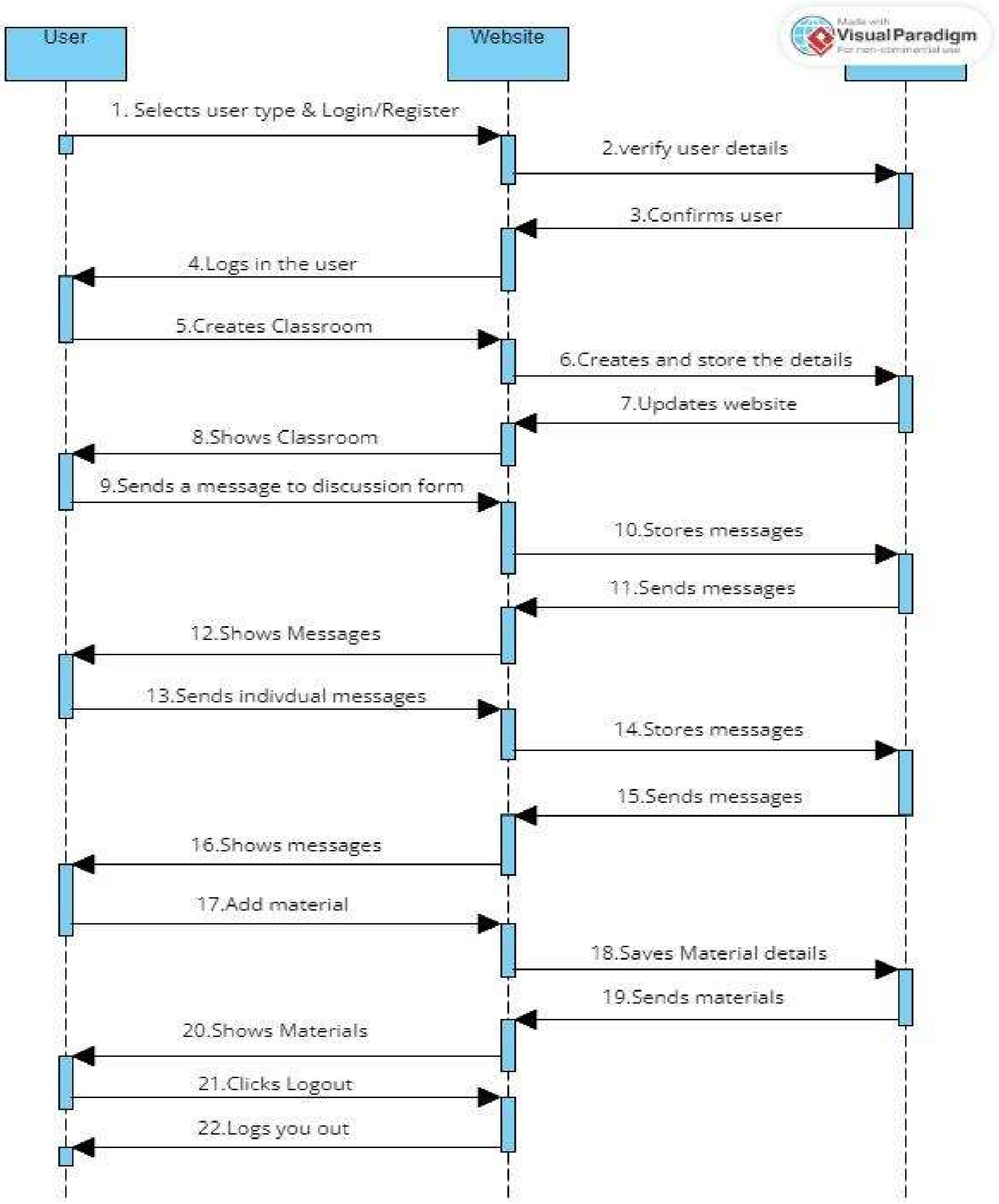


Fig.7.8 Sequence Diagram

6.3.3 USE CASE DIAGRAM

Use case diagrams are used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

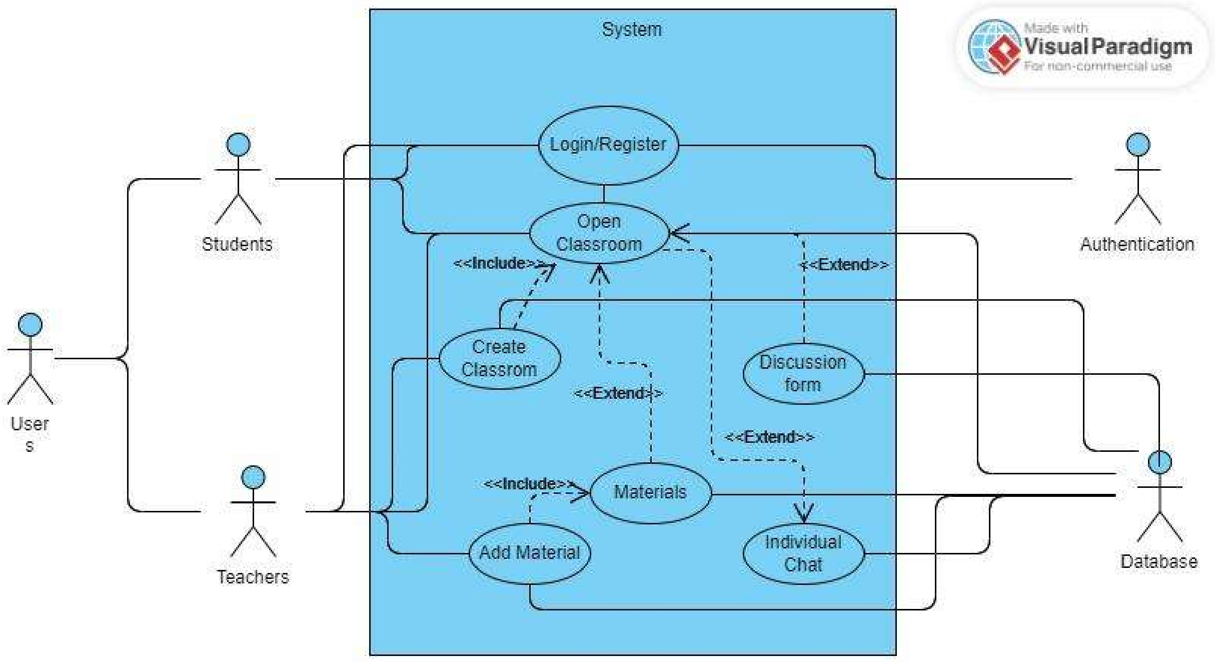


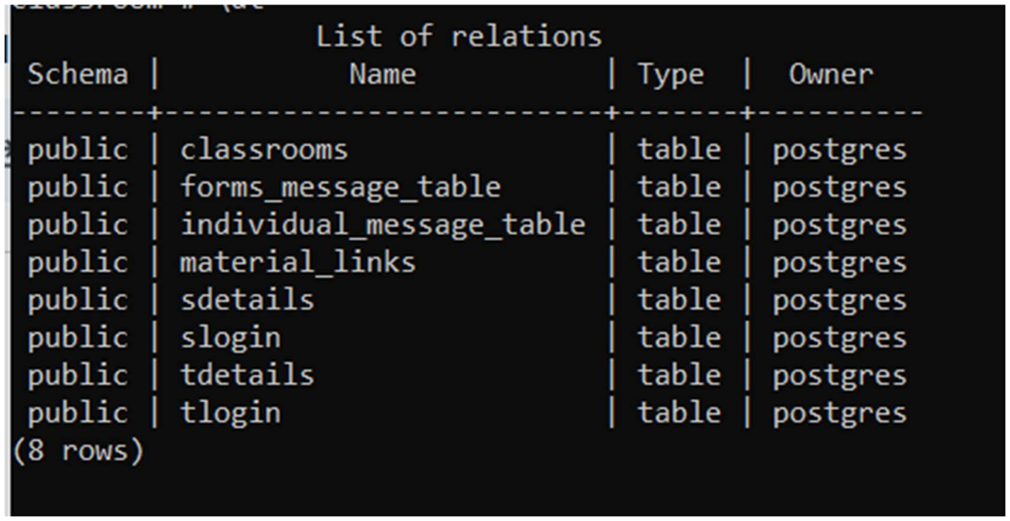
Fig.7.9 Use Case Diagram

7. DEVELOPMENT

7.1 Languages Used

7.1.1 PostgreSql :

This language PostgreSql is used to store the data in the Database and can be used to retrieve the data we have created multiple tables like register, login, classrooms and so on. The tables are used based on the functions that are used. The tables are as follows in the fig 7.1.1.



```
                  List of relations
 Schema |          Name          | Type  |  Owner
--------+------------------------+-------+----------
 public | classrooms             | table | postgres
 public | forms_message_table    | table | postgres
 public | individual_message_table | table | postgres
 public | material_links         | table | postgres
 public | sdetails               | table | postgres
 public | slogin                 | table | postgres
 public | tdetails               | table | postgres
 public | tlogin                 | table | postgres
(8 rows)
```

Fig 7.1 Tables

How to create an table :

```
create table sdetails (
   sid serial primary key,
   name text,
   email text,
   branch text,
   section text,
   rollno text,
   batch text,
   ph_no text
);
```

The above code represents the table creation for the student details table.

7.1.2 Java:

Java is used to create the apis using the existing data in the data base or to insert the data into the database which is linked to the frontend the below code represents the jdbc connection of the postgresql.

```
package com.example;
public class connections {
   public static final String driver = "org.postgresql.Driver";
   public static final String url = "jdbc:postgresql://localhost:5432/classroom";
   public static final String name = "postgres";
   public static final String password = "yourpassword";
}
Class.forName(connections.driver);
Connection con =
DriverManager.getConnection(connections.url,connections.name,connections.password);
Statement st = con.createStatement();
```

7.1.4 React Js:

React js is used to create the front end i.e, UI that is responsible to the user friendly actions and this is linked to the java servlet to perform the actions. The below code shows the start of the application.

```
import { useState } from 'react';
import './App.css';
import Login from './pages/Login';
import Register from './pages/Register';
import StudentRoom from './pages/StudentRoom';
import {BrowserRouter as Router,Route,Routes,Link} from 'react-router-dom'
import Home from './pages/Home';
import Navbar from './components/Navbar';
import TeacherRoom from './pages/TeacherRoom';
import ChatRoom from './pages/ChatRoom';
import TeacherChatRoom from './pages/TeacherChatRoom';
import StudentChatRoom from './pages/StudentChatRoom';
```

```
function App() {
  return (
    <div className='App'>
      <Router>
        <Routes>
          <Route
            path="/"
            element={<Home />}
            />
          <Route
           path="/register"
           element={<Register />}
           />
          <Route
            path="/login"
            element={<Login />}
          />
          <Route
            path="/StudentRoom"
            element={<StudentRoom />}
          />
          <Route
            path="/TeacherRoom"
            element={<TeacherRoom />}
          />
          <Route
            path="/ChatRoom/:id/:u"
            element={<ChatRoom />}
          />
          <Route
            path="/TeacherChatRoom/:id"
            element={<TeacherChatRoom />}
          />
          <Route
            path="/StudentChatRoom/:sid/:cid"
            element={<StudentChatRoom />}
          />
        </Routes>
      </Router>
    </div>
  );
}

export default App;
```

7.2 SAMPLE CODE

7.2.1 Registration Code :

```
package com.example;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.google.gson.JsonObject;
public class sregister extends HttpServlet{
   public void service(HttpServletRequest request , HttpServletResponse response)
throws IOException , ServletException{

        response.setContentType("JSON");
        PrintWriter out = response.getWriter();
        // Take parameters from user.
        String name = request.getParameter("name");
        String email = request.getParameter("email");
        String branch = request.getParameter("branch");
        String section = request.getParameter("section");
        String rollno = request.getParameter("rollno");
        String batch = request.getParameter("batch");
        String phno = request.getParameter("phno");
        String password = request.getParameter("password");
        JsonObject object = new JsonObject();

        int year = Integer.parseInt(batch.substring(2));
        batch = batch.substring(2)+Integer.toString(year+4);

        String query1 = "insert into sdetails
(name,email,branch,section,rollno,batch,ph_no) values
('"+name+"','"+email+"','"+branch+"','"+section+"','"+rollno+"','"+batch+"','"+phno+
"');";
        String query2 = "insert into slogin (email,password) values
('"+email+"','"+password+"');";

        try{

            Class.forName(connections.driver);
            Connection con =
DriverManager.getConnection(connections.url,connections.name,connections.passw
ord);
            Statement st = con.createStatement();
            int q1 = st.executeUpdate(query1);
            int q2 = st.executeUpdate(query2);
            if(q1>0 && q2>0){
                object.addProperty("statues", "success");
                object.addProperty("status code", "200");
                out.print(object);
            }else{
                object.addProperty("statues", "failed");
                object.addProperty("status code", "404");
                object.addProperty("message", "Invalid Details");
                out.print(object);
            }

        }catch(Exception e){
            object.addProperty("status", "failed");
            object.addProperty("status code", "500");
            object.addProperty("message", e.getMessage());
            out.print(object);
        }
    }

}
```

7.2.2 Login Code:

```
package com.example;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.google.gson.JsonObject;

public class slogin extends HttpServlet {
   public void service(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException {
      response.setContentType("JSON");
      PrintWriter out = response.getWriter();
      String email = request.getParameter("email");
      String password = request.getParameter("password");
      JsonObject object = new JsonObject();

      String query1 = "select * from slogin where email = '" + email + "' and
password = '" + password + "'";
      if (email.length() == 0 || password.length() == 0) {
         object.addProperty("status", "failed");
         object.addProperty("status code", "404");
         object.addProperty("message", "fill the fields.");
         out.print(object);
         return;
      }

      try {

         Class.forName(connections.driver);
         Connection con = DriverManager.getConnection(connections.url,
connections.name, connections.password);
         Statement st = con.createStatement();
         ResultSet rs = st.executeQuery(query1);
         if (rs.next()) {
            Statement st1 = con.createStatement();
            ResultSet rs1 = st1.executeQuery("select * from sdetails where sid =
'"+rs.getString("sid")+"';");
            rs1.next();
            object.addProperty("statues", "success");
            object.addProperty("status code", "200");
            object.addProperty("sid", rs.getString("sid"));
            object.addProperty("name", rs1.getString("name"));
            st1.close();
            st.close();
            out.print(object);
          } else {
            object.addProperty("statues", "failed");
            object.addProperty("status code", "404");
            object.addProperty("message", "Invalid Details");
            out.print(object);
          }
          con.close();

      } catch (Exception e) {
          object.addProperty("status", "failed");
          object.addProperty("status code", "500");
          object.addProperty("message", e.getMessage());
          out.print(object);
      }
   }
}
```

7.2.3 Send Mail:

```
package com.example;
import java.util.Properties;
import java.util.Random;
import javax.mail.Authenticator;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

public class Mail {

   static String getRandomNumber(){
      Random rand = new Random();
      int number  = rand.nextInt(999999);
      return String.format("%06d", number);
   }

   public static boolean sendEmail(String otp , String recipient) throws
MessagingException{

      Properties pro = new Properties();
      pro.put("mail.smtp.auth", "true");
      pro.put("mail.smtp.starttls.enable","true");
      pro.put("mail.smtp.host","smtp.gmail.com");
      pro.put("mail.smtp.port","587");

      final String mail = "saketh4532@gmail.com";
      final String pass = "zwqulwtcgbzmvcie";
      Session session = Session.getInstance(pro , new Authenticator() {

          @Override
          protected PasswordAuthentication getPasswordAuthentication(){
              return new PasswordAuthentication(mail, pass);
          }
      });

      Message message = prepareMessage(session,otp,mail,recipient);
      Transport.send(message);
      return true;
   }

   static Message prepareMessage(Session session, String otp, String mail ,String
recipient){
      Message message = new MimeMessage(session);
      try{
          message.setFrom(new InternetAddress(mail));
          message.setRecipient(Message.RecipientType.TO, new
InternetAddress(recipient));
          message.setSubject("User Email Verification");
          message.setContent("Register Successfully. Please verify the account using
this code : <h3>"+ otp +"</h3>" , "text/html");
          return message;
      }catch(Exception e){System.out.println(e.getMessage());}
      return null;
   }
}
```

7.2.4 Login Page:

```
import React from 'react'
import { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import axios from 'axios';
```

```
const Login = () => {
    const url = "http://localhost:8081/classroom_apis";
    const navigate = useNavigate()
    const [show,setShow] = useState("uil uil-eye-slash hide");
    const [type,setType] = useState("password")
    const [email,setEmail] = useState("");
    const [password,setPassword] = useState("")
    const [register_candidate,setRegister_candidate] = useState("student");
    const showPassword = (e) => {
        e.preventDefault();
        if (show==="uil uil-eye hide") {
        setShow("uil uil-eye-slash hide")
        setType("password")
        }
        else{
        setShow("uil uil-eye hide")
        setType("text")
        }
     }
    const changeType = (e) => {
        e.preventDefault();
        setRegister_candidate(e.target.value)
    }
    async function LoginUser(e) {
        e.preventDefault()
        if(register_candidate === 'student'){
            // student login
            axios.get(url+"/slogin",{
                params:{
                    email:email,
                    password:password
                }
            }).then((response)=>{
                if(response.data["status code"] == '200'){
                    localStorage.setItem("sid",response.data.sid)
                    localStorage.setItem("sname",response.data.name)
                    setEmail("")
                    setPassword("")
                    console.log(localStorage.getItem("sid"))
                    console.log(localStorage.getItem("sname"))
                    navigate('/StudentRoom')
                }else{
                    setPassword("")
                }
            }).catch((e)=>console.log(e))
        }else if(register_candidate=== 'teacher'){
            // teacher login
            axios.get(url+"/tlogin",{
                params:{
                    email:email,
                    password:password
                }
            }).then((response)=>{
                if(response.data["status code"] == '200'){
                    localStorage.setItem("tid",response.data.tid)
                    localStorage.setItem("tname",response.data.tname)
                    console.log(localStorage.getItem("tid"))
                    console.log(localStorage.getItem("tname"))
                    navigate('/TeacherRoom')
                }else{
                    setPassword("")
                }
            }).catch((e)=>console.log(e))
        }
    }
    return (
    <div style={{marginTop:100}} className='Register'>
    <div className="container">
    <div className="form login">
      <div className="tops">
        <span className="title">LOGIN</span>
          <select className="select-value" onChange={changeType}>
            <option value="student" selected>student</option>
            <option value="teacher">teacher</option>
          </select>
      </div>
            <form>
                <div className="input-field">
                    <input type="text" onChange={(e) => setEmail(e.target.value)}
name="name" placeholder="Enter your email" />
                    <i className="uil uil-envelope icon"></i>
                </div>
        <div className="input-field">
                    <input type={type} onChange={(e) => setPassword(e.target.value)}
name="password" placeholder="Enter your password" />
                    <i className="uil uil-lock icon"></i>
                    <button  onClick={showPassword}><i className={show}
style={{paddingLeft:10}}></i></button>
                </div>
            </form>
            <div className="button">
            <button onClick={LoginUser}>Login</button>
          </div>
          <div className="button">
            <button ><Link to="/register"
className='link'>Register</Link></button>
          </div>
        </div>
    </div>
    </div>
  )
}

export default Login;
```

7.2.5 Registration Page:

```
import React from 'react'
import { Link, useNavigate } from 'react-router-dom';
import { useState } from 'react';
import axios from 'axios';


const Register = () => {
    const url = "http://localhost:8081/classroom_apis";
    const navigate = useNavigate()
    const [register_type,setRegister_type] = useState(true);
    const [register_candidate,setRegister_candidate] = useState("student");
    const [name,setName] = useState("");
    const [email,setEmail] = useState("");
    const [mobile,setMobile] = useState("");
    const [branch,setBranch] = useState("cse")
    const [section,setSection] = useState("1")
    const [rollno,setRollno] = useState("")
    const [batch,setBatch] = useState("2019")
    const [password,setPassword] = useState("")
    const [cpassword,setCpassword] = useState("")
    const [show1,setShow1] = useState("uil uil-eye-slash hide");
    const [show2,setShow2] = useState("uil uil-eye-slash hide");
    const [type1,setType1] = useState("password");
    const [type2,setType2] = useState("password");
    const showPassword1 = (e) => {
        e.preventDefault();
        if (show1==="uil uil-eye hide") {
        setShow1("uil uil-eye-slash hide")
        setType1("password")
        }
        else{
```

```
    setShow1("uil uil-eye hide")
    setType1("text")
    }
}
const showPassword2 = (e) => {
    e.preventDefault();
    if (show2==="uil uil-eye hide") {
    setShow2("uil uil-eye-slash hide")
    setType2("password")
    }
    else{
    setShow2("uil uil-eye hide")
    setType2("text")
    }
}
const changeType = (e) => {
    e.preventDefault();
    if (register_type) {
        setRegister_type(false)
    }
    else{
        setRegister_type(true)
    }
    setRegister_candidate(e.target.value)
 }
async function RegisterUser(e) {
    e.preventDefault()
    console.log(name)
    console.log(branch)
    console.log(section)
    console.log(batch)
    if(password!==cpassword){
        window.location.reload()
    }
    if(register_candidate==="student"){
        axios.get(url+"/sregister",{
            params:{
                name: name,
                email: email,
                rollno: rollno,
                branch: branch,
                section: section,
                phno: mobile,
                batch: batch,
                password: password
            }
            }).then((response)=>{
                if(response.data["status code"] == '200'){
                    console.log("success")
                    navigate('/login')
                }
            }).catch((e)=>console.log(e))
        }
        else{
            axios.get(url+"/tregister",{
                params:{
                    name: name,
                    email: email,
                    branch: branch,
                    phno: mobile,
                    password: password
                }
            }).then((response)=>{
                if(response.data["status code"] == '200'){
                    console.log("success")
                    navigate('/login')
                }
            }).catch((e)=>console.log(e))
        }
    }
  return (
    <div className="Register">
        <div className="container">
            <div className="form login">
                <div class="tops">
                    <span className="title">REGISTER</span>
                    <select onChange={changeType} className="select-value">
                            <option value="student" selected>student</option>
                            <option value="teacher">teacher</option>
                    </select>
                </div>

                <form>
                    <div className="input-field">
                        <input type="text" name="name" onChange={(e) =>
setName(e.target.value)} placeholder="Enter your name" />
                        <i className="uil uil-user icon"></i>
                    </div>
                    <div className="input-field">
                        <input type="text" name="name" onChange={(e) =>
setEmail(e.target.value)} placeholder="Enter your email" />
                        <i className="uil uil-envelope icon"></i>
                    </div>

                    {   register_type ?
                        <div className="input-field">
                            <input type="text" name="name" onChange={(e) =>
setRollno(e.target.value)} placeholder="Enter your roll no" />
                            <i className="uil uil-dialpad-alt icon"></i>
                        </div>
                        :
                        <></>
                    }
                    <div className="input-field">
                        <input type="text" name="name" onChange={(e) =>
setMobile(e.target.value)} placeholder="Enter your phone no" />
                        <i className="uil uil-phone icon"></i>
                    </div>
                    { register_type?
                    <div className="input-field-selects">
                        <select onChange={(e) => setBranch(e.target.value)}
className="select-value branch">
                            <option value="cse" selected>cse</option>
                            <option value="it">it</option>
                            <option value="ece">ece</option>
                            <option value="eee">eee</option>
                            <option value="mech">mech</option>
                            <option value="civil">civil</option>
                            <option value="chem">chem</option>
                        </select>
                        <select onChange={(e) => setSection(e.target.value)}
className="select-value section">
                            <option value="1" selected>1</option>
                            <option value="2">2</option>
                            <option value="3">3</option>
                            <option value="4">4</option>
                            <option v-if="registervalues.branch == cse"
value="csd">csd</option>
                            <option v-if="registervalues.branch == cse"
value="csm">csm</option>
                            <option v-if="registervalues.branch == mech"
value="mrb">mrb</option>
                        </select>
                        <select onChange={(e) => setBatch(e.target.value)}
className="select-value section">
                            <option value="2019" selected>2019</option>
                            <option value="2020">2020</option>
                            <option value="2021">2021</option>
                            <option value="2022">2022</option>
                            <option value="2023">2023</option>
                            <option value="2024">2024</option>
                            <option value="2025">2025</option>
                        </select>
                    </div>
                    :<div className="input-field-selects">
                        <select onChange={(e) => setBranch(e.target.value)}
className="select-value branch">
                            <option value="cse" selected>cse</option>
                            <option value="it">it</option>
                            <option value="ece">ece</option>
                            <option value="eee">eee</option>
                            <option value="mech">mech</option>
                            <option value="civil">civil</option>
                            <option value="chem">chem</option>
                        </select>
                    </div>
                    }
                    <div className="input-field">
                        <input type={type1} name="password" onChange={(e) =>
setPassword(e.target.value)} placeholder="Enter your password" />
                        <i className="uil uil-lock icon"></i>
                        <button  onClick={showPassword1}><i className={show1}
style={{paddingLeft:10}}></i></button>
                    </div>
                    <div className="input-field">
                        <input  type={type2} name="cpassword" onChange={(e) =>
setCpassword(e.target.value)} placeholder="Enter password again" />
                        <i className="uil uil-lock icon"></i>
                        <button  onClick={showPassword2}><i className={show2}
style={{paddingLeft:10}}></i></button>
                    </div>
                  </form>

                  <div className="button">
                    <button onClick={RegisterUser}>Register</button>
                  </div>
                  <div className="button">
                    <button ><Link to="/login" className='link'>Login</Link></button>
                  </div>
                </div>
              </div>

          </div>
        )
```

```
}

export default Registion;
```

7.2.6 Chat Room:

```
import React,{ useEffect, useRef, useState } from 'react'
import Navbar from '../components/Navbar'
import sendicon from '../assets/send.png'
import teachericon from '../assets/teacher.png'
import groupicon from '../assets/group.png'
import axios from 'axios';
import { useNavigate, useParams } from 'react-router-dom'

const StudentChatRoom = () => {
  const url = "http://localhost:8081/classroom_apis";
  const params = useParams()
  const studentId = params.sid;
  const classId = params.cid;
  const u_id = params.u;
  const [user,setUser] = useState(null)
  const [pdfname,setPdfname] = useState("");
  const [pdflink,setPdflink] = useState("");
  const navigate = useNavigate()
  const [msgs,setMsgs] = useState(null)
  const [msg_num,setMsg_num] = useState(-1)
  const [isloaded,setIsloaded] = useState(false)
  const [click_state,setClick_state] = useState(true)
  const [msg,setMsg] = useState(null)
  const msgsEndRef = useRef(null);
  function goBack(e) {
   e.preventDefault()
   navigate('/ChatRoom/'+`${classId}`+'/t')
  }
  async function getMessages(e) {
    console.log(classId)
    console.log(studentId)
    console.log(localStorage.getItem('tid'))
    await
axios.get(url+"/TeacherRoom?cid="+classId+"&t_login_id="+localStorage.getItem(
'tid')+"&choice=6"+"&sid="+studentId)
    .then((response)=>{
     setMsgs(response.data.details)
     console.log(response.data.details)
     setMsg_num(0)
     msgsEndRef.current?.scrollIntoView();
```

```
        }).catch((e)=>console.log(e))
      }
      async function sendMsg(e) {

axios.get(url+"/TeacherRoom?cid="+classId+"&login_id="+localStorage.getItem('ti
d')+"&choice=5&new_message="+msg+"&new_sid="+studentId)
        .then((response)=>{
            if(response.data["status code"] == '200'){
                setMsg(null)
                getMessages()
                msgsEndRef.current?.scrollIntoView();
            }
        }).catch((e)=>console.log(e))
      }
      useEffect(() => {
        setUser(localStorage.getItem('sname'))
        console.log(user)
        getMessages()
      },[]);
    if(msgs==null && msg_num==-1){
        return (<h1 style={{color:"white"}}>Loading...</h1>)
    }
    else{
        return (
          <main >
          <Navbar />
          <div style={{display:"flex",height:"92%",width:"100%"}}>
          <div className="side-nav">
                      <div className="side-top">
                        <div className="nav-item"
style={{fontWeight:"500",fontSize:"23px",borderBottom:"1px solid
white"}}>Student Chat</div>
                          {console.log(msgs)}
                      </div>


                      <button style={{borderTop: "1px solid white"}} onClick={goBack}
className="side-bottom">
                          Go Back
                          <i className="uil uil-backward"></i>
                      </button>
                  </div>
            <div>

            <ul id="chat">
              {msgs&&msg_num==0 ? msgs.map((msg) =>
                {
                 return <li style={{marginTop:"17px"}}
className={`${msg.sender=="student" ? "me" : "you"}`} >
                 <div className="entete">
                   {console.log(user)}
                   {console.log(msg.message)}
                   <span className={`${msg.sender=="student" ? "status blue" : "status
green"}`}></span>
                 </div>
                 <div className="triangle"></div>
                 <div className="message" >
                   {msg.message}
                 </div>
                </li>;
                })
              :
              <p style={{color:"white",margin:"40px",fontSize:"20px"}}>No
messages</p>

            }
            <div style={{display:"hidden"}} ref={msgsEndRef} />
          </ul>
          <footer>
            <div>
            <input id='send_input' onChange={(e) => setMsg(e.target.value)}
placeholder="Type your message"></input>
            <p onClick={()=> {
              var send_input = document.getElementById('send_input')
              send_input.value=""
              sendMsg()
              }}><img src={sendicon} className="icon" /></p>
            <p><img onClick={()=>{navigate('/ChatRoom/'+`${classId}`+'/s/')}}
src={groupicon} className="icon" /></p>:<></>

            </div>
          </footer>
          </div>
          </div>

        </main>
     )
   }
}

export default StudentChatRoom
```

7.3 RESULTS

Fig 7.2 shows the screen which represents the home page of the web application that will be having two options like register and the login buttons so clicking on the buttons navigate to the register and login pages respectively.

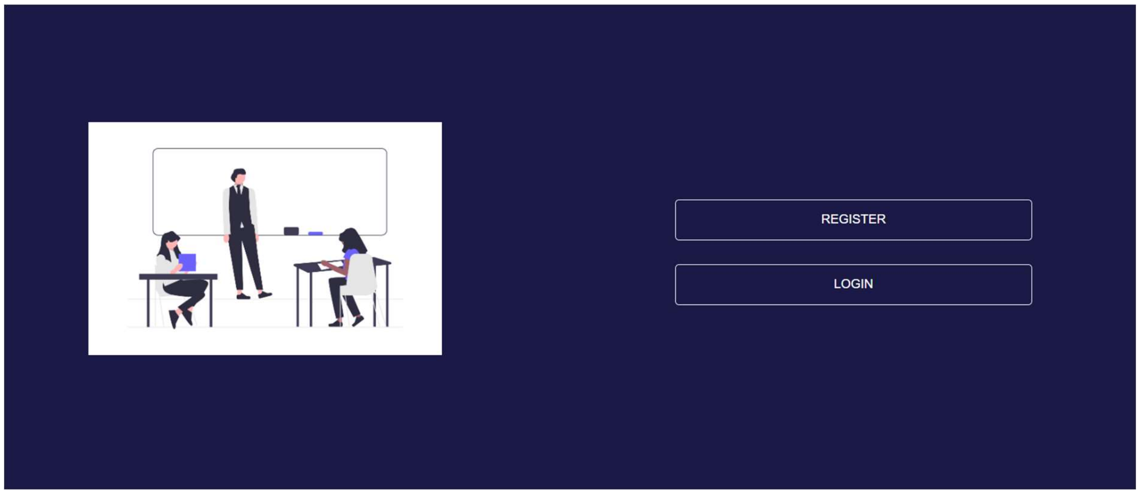


Fig 7.2 Home Page

Fig 7.3 shows the register page and having different options like student register and the teacher register page where the register page take the details like name , phone number, email id, roll no and the batch based on the select options.

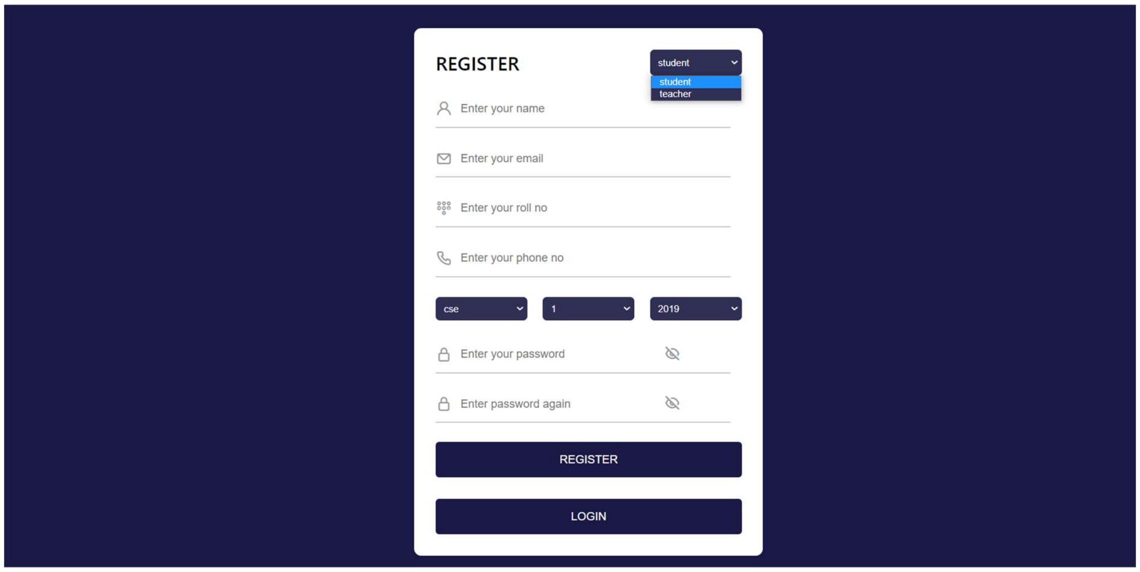


Fig 7.3 Register Page

Fig 7.4 shows the login page and this will also have two options like student login and teacher login based on the registered email and the password.

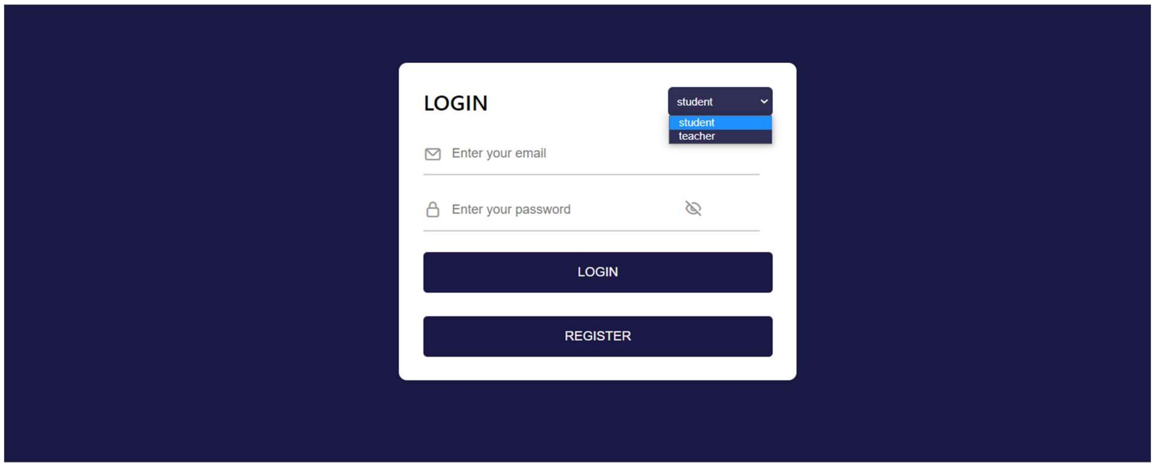


Fig 7.4 Login Page

Fig 7.5 show the student dashboard where the students can see all the classrooms that are created by the teacher respectively to their sections in the dashboard and in the side nav we Visit the students available in the classroom and the available classes.

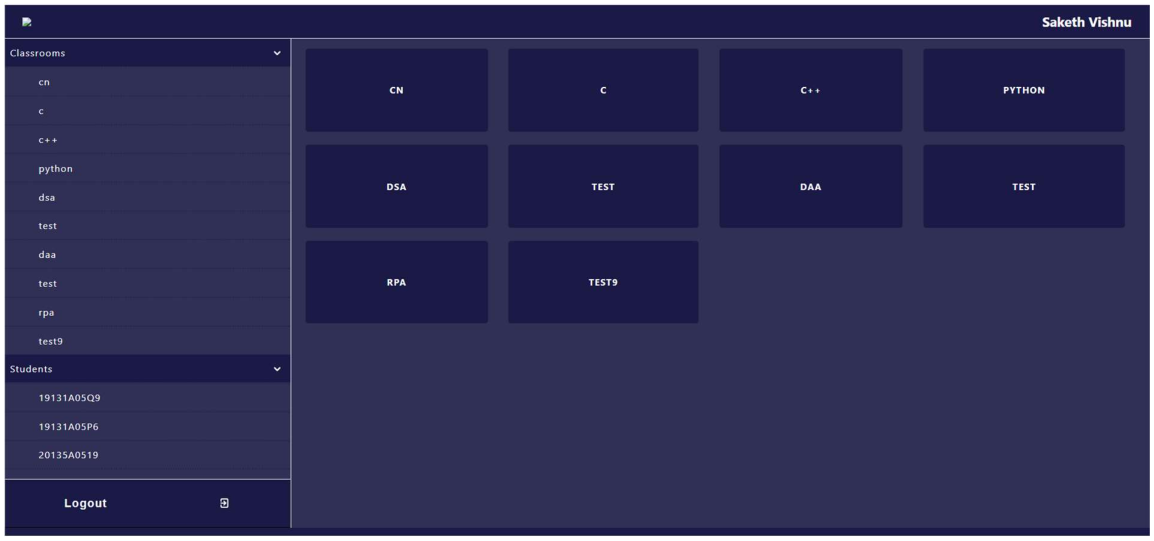


Fig 7.5 Student Dashboard

Fig 7.6 show the teacher dashboard where the can create the classroom and can see all the classrooms that are created by the teacher and can also have an option to create a new classroom and assign the batch to them.

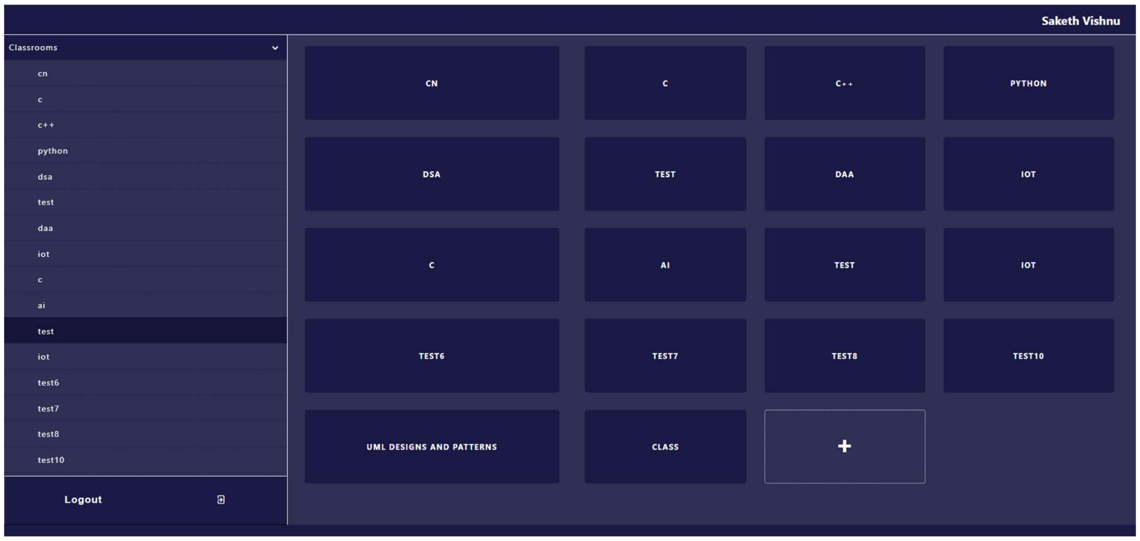


Fig 7.6 teacher Dashboard

Fig 7.7 shows the add classroom screen that is created by the teacher so that the teacher will assign the batch so that all the batch students can see the classroom in the dashboard so, not event a single student can miss the classroom by mistake or ignore it.

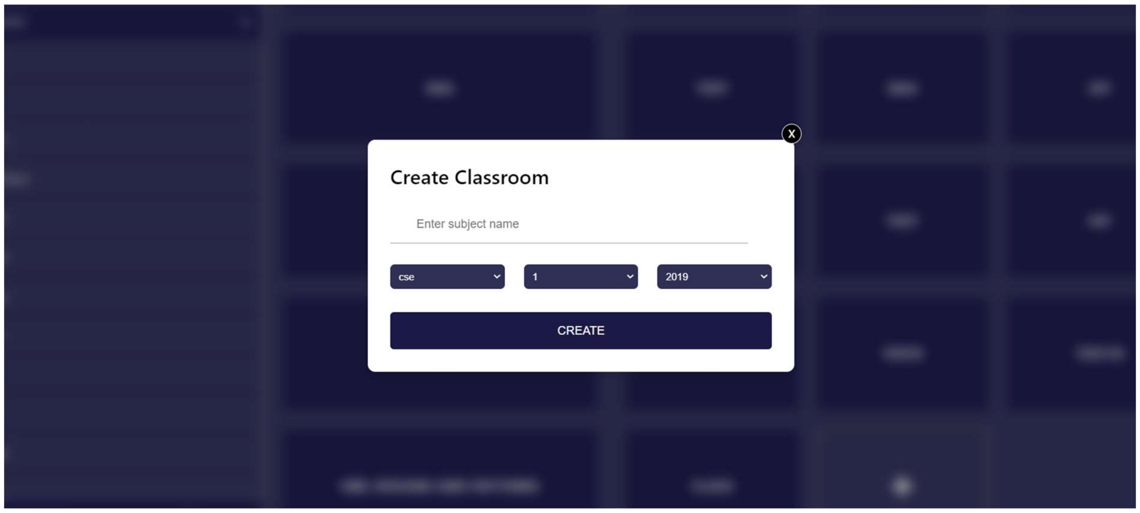
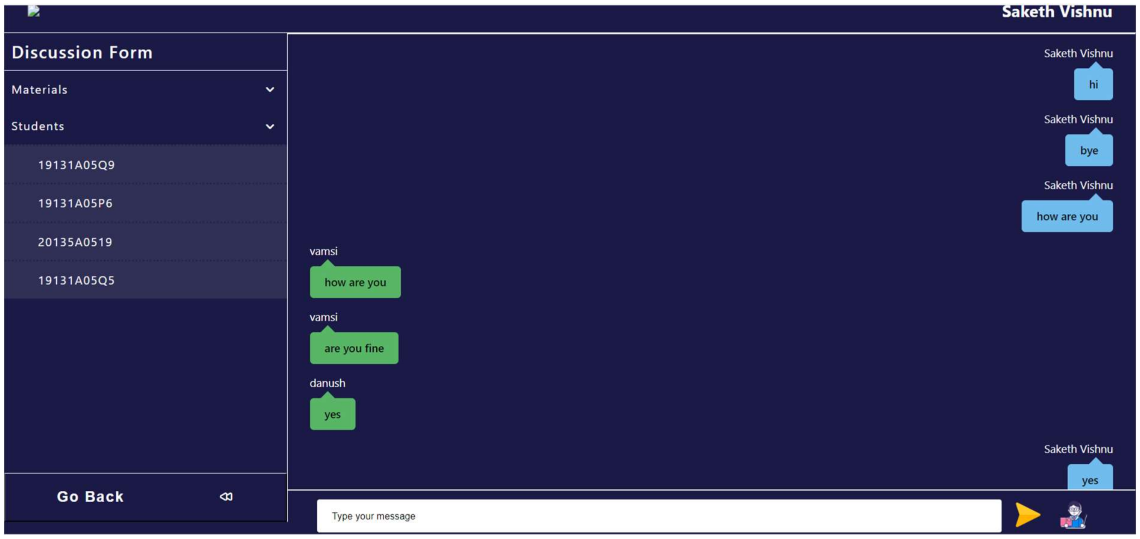


Fig 7.7 teacher Dashboard

Fig 7.8 shows the discussion rooms of the student so a student can add their doubts in the chat and can discuss on that problem and get the solution and the teacher can also be available in the chats to clarify the doubts.



Saketh Vishnu: hi
Saketh Vishnu: bye
Saketh Vishnu: how are you
vamsi: how are you
vamsi: are you fine
danush: yes
Saketh Vishnu: yes

Fig 7.8 Discussion Form

Fig 7.9 shows the add material form so the teacher can add the material name and can provide the link so that the link of the material so that the student can visit the links and the material links whenever they need to study.

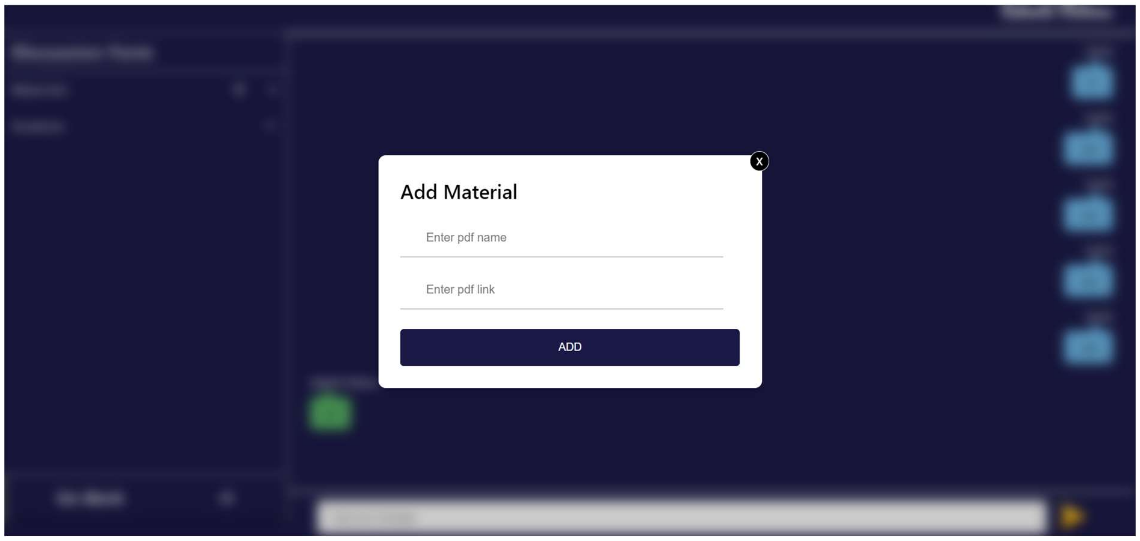


Fig 7.8 Discussion Form

8. TESTING

8.1 INTRODUCTION TO TESTING:

SOFTWARE TESTING is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. It is required for evaluating the system. This phase is the critical phase of software quality assurance and presents the ultimate view of coding.

Importance of Testing:

The importance of software testing is imperative. A lot of times this process is skipped, therefore, the product and business might suffer. To understand the importance of testing, here are some key points to explain

➢ Software Testing saves money

➢ Provides Security

➢ Improves Product Quality

➢ Customer satisfaction

Testing is of different ways The main idea behind the testing is to reduce the errors and do it with a minimum time and effort.

Benefits of Testing:

- **Cost-Effective:** It is one of the important advantages of software testing. Testing anyIT project on time helps you to save your money for the long term. In case if the bugs caught in the earlier stage of software testing, it costs less to fix.
- **Security:** It is the most vulnerable and sensitive benefit of software testing. People are looking for trusted products. It helps in removing risks and problems earlier.
- **Product quality:** It is an essential requirement of any software product. Testing ensures a quality product is delivered to customers.
- **Customer Satisfaction:** The main aim of any product is to give satisfaction to their customers. UI/UX Testing ensures the best user experience.

Different types of Testing:

Unit Testing: Unit tests are very low level, close to the source of your application. They consist in testing individual methods and functions of the classes, components or modules used by your software. Unit tests are in general quite cheap to automate and can be run very quickly by a Continuous integration server.

Integration Testing: Integration tests verify that different modules or services used by your application work well together. For example, it can be testing the interaction with the database or making sure that microservices work together as expected. These types of tests are more expensive to run as they require multiple parts of the application to be up and running.

Functional Tests: Functional tests focus on the business requirements of an application. They only verify the output of an action and do not check the intermediate states of the system when performing that action.

There is sometimes a confusion between integration tests and functional tests as they both require multiple components to interact with each other. The difference is that an integration test may simply verify that you can query the database while a functional test would expect to get a specific value from the database as defined by the product requirements.

Regression Testing: Regression testing is a crucial stage for the product & very useful for the developers to identify the stability of the product with the changing requirements. Regression testing is a testing that is done to verify that a code change in the software does not impact the existing functionality of the product.

System Testing: System testing of software or hardware is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements. System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system.

Performance Testing: It checks the speed, response time, reliability, resource usage, scalability of a software program under their expected workload. The purpose of Performance Testing is not to find functional defects but to eliminate performance bottlenecks in the software or device.

Alpha Testing: This is a form of internal acceptance testing performed mainly by the in-house software QA and testing teams. Alpha testing is the last testing done by the test teams at the development site after the acceptance testing and before releasing the software for the beta test. It can also be done by the potential users or customers of the application. But still, this is a form of in-house acceptance testing.

Beta Testing: This is a testing stage followed by the internal full alpha test cycle. This is the final testing phase where the companies release the software to a few external user groups outside the company test teams or employees. This initial software version is known as the beta version. Most companies gather user feedback in this release.

Black Box Testing: It is also known as Behavioural Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is **not** known to the tester. These tests can be functional or non-functional, though usually functional.

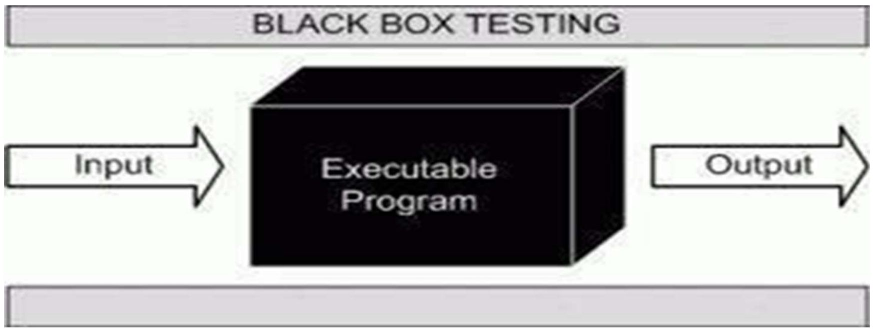


Fig. 8.1. Black Box testing

This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behaviour or performance errors
- Initialization and termination errors

White Box Testing: White box testing (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or

Structural Testing) is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system. This method is named so because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees.

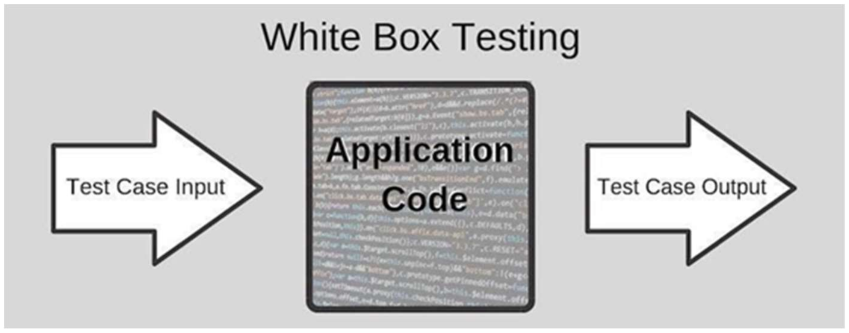
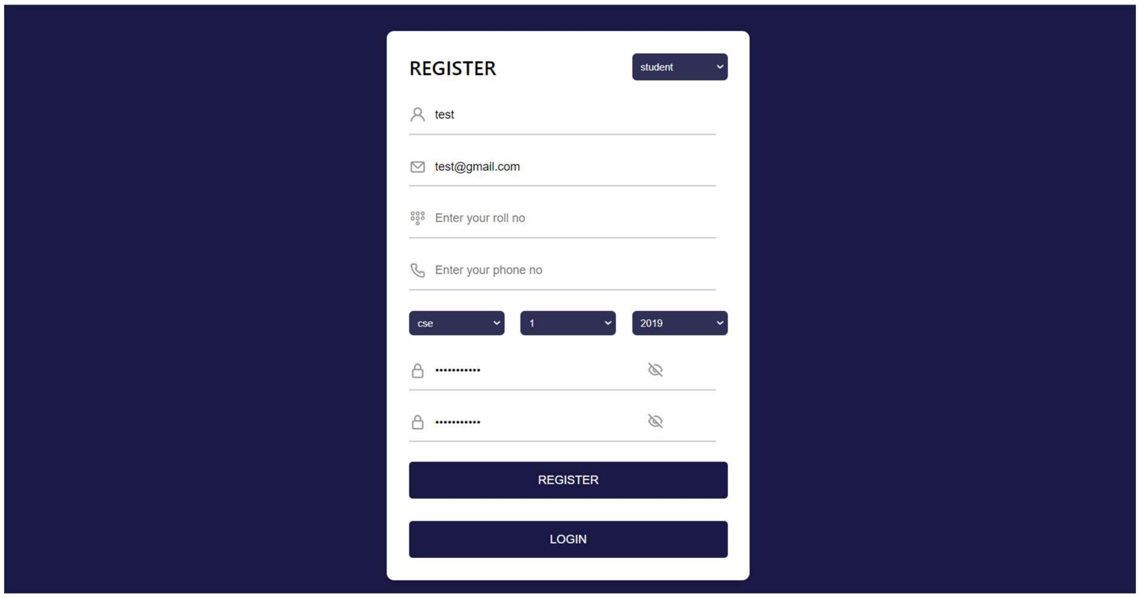


Fig. 8.2. White Box testing

Fig 8.3 shows the case that it wont be accepting the response if the field are not completely filled and if the password and confirm password dosen’t matches then it will not accept.


REGISTER

student

test

test@gmail.com

Enter your roll no

Enter your phone no

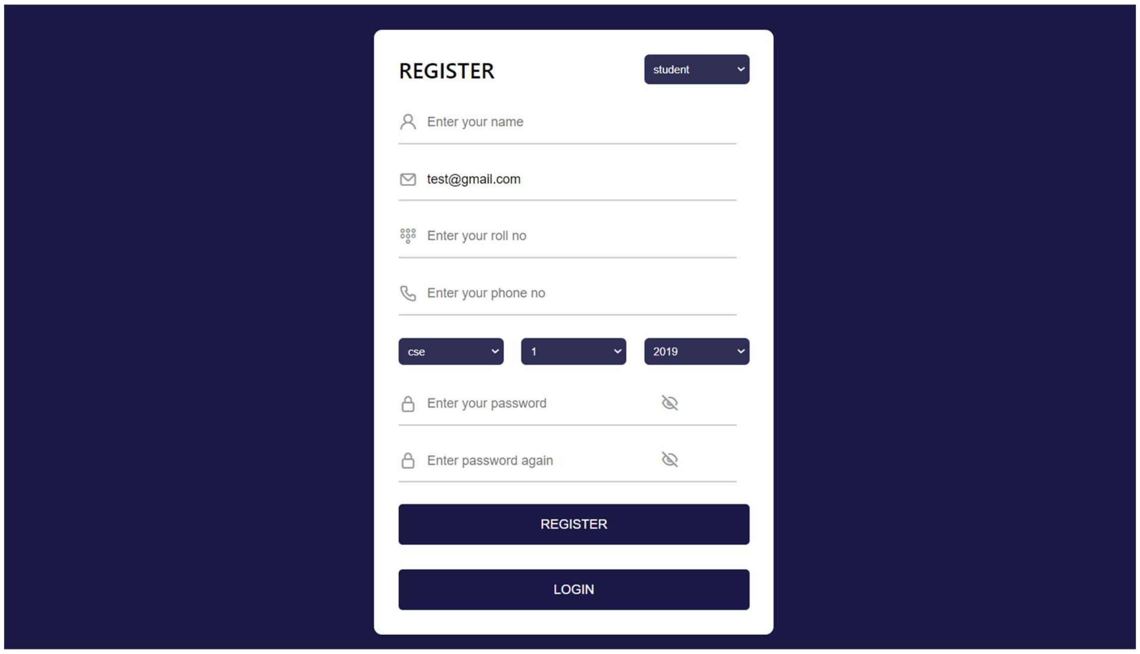
cse 1 2019

•••••••••••

•••••••••••

Fig. 8.2. Partially filled fields

Fig 8.4 shows the case that it wont be sending the opt if the mail given is not valid and if the mail id is not the domain of the college then it will not be accepting as the response and not sending the otp.


REGISTER

student

Enter your name

test@gmail.com

Enter your roll no

Enter your phone no

cse | 1 | 2019

Enter your password

Enter password again

Fig. 8.2. Register page with wrong mail id.

9. CONCLUSION

The Subject Discussion Classroom was developed using Java in the backend(servlets), React framework in the frontend and Postgres database to store the details of the students and teachers. The subject discussion classroom solves the problems in the traditional classroom. Like standard periods, students have to reach the teacher. In contrast, both have free time and difficulty in providing the materials. The proposed system allows a group chat discussion with the teacher and a direct conversation with the teacher. Data files like pdfs, youtube links can be stored in the same container to access those files directly.

10. FUTURE SCOPE

The Subject Discussion Classroom has a very great potential to be further improved in the future. We can eventually integrate any meeting platform into it and add features for downloading the attendance. Also we can maintain the minutes period of the class in a day wise manner so that it can be useful for students who are not able to attend the classes or if the student requires to memorize things that had happened in a specific class. We can also develop features for students to submit their assignments in it.

11. REFERENCES

[1] https://code.visualstudio.com/docs.

[2] https://legacy.reactjs.org/docs/getting-started.html.

[3] https://mvnrepository.com/

[4] https://www.postgresql.org/docs/

[5] https://www.java.com/en/download/help/index_using.html

[6] https://www.geeksforgeeks.org/unified-modeling-language-uml-in