

## **1. INTRODUCTION**

Project Management is the application of knowledge, skills, tools, and techniques to perform several tasks to meet project requirements, to make the final deliverables in a finite time and budget. The scope is a guaranteed set of deliverables, and the project is to be created, keeping in mind the scope.

Time is one of the most critical stakeholder considerations and a vital measure of project success. The time to complete the project must be estimated as accurately as possible. Quality is one such sphere that is almost always affected by the other constraints. If the time or cost reduces the quality of the project will be drastically affected. Cost is another important aspect that needs to be considered, which is an estimated factor of labour cost, factory, administration, software, equipment altogether.

The Project Manager must foresee the possible risks at every step of the project. The risk part involves many what-if scenarios and the solution for those scenarios. Every project has a combination of resources required to make it a success. Resources refer to the limitations to complete a particular job; they can be in terms of people, equipment, time, or other supplies.

## **2.SYSTEM STUDY**

### **2.1 EXISTING SYSTEM**

The Present system always facing time problem during their development work by which there exist extra investment and sometimes losing their markets due to criticism.

- All the work was maintained on using manual file by which it creates difficulties for the organization to prepare reports on projects.
- It's difficult for the managers to keep track of projects which are working indifferent teams and take immediate action to complete the projects within stipulated time.
- The manual systems are not able to provide alerts and notification for the important ones based on their maximum time involved and status along with predictions for their deadlines.

### **2.2 PROPOSED SYSTEM**

The important factor which will be considered in this new system is time. Before starting any new project, its manager can set the deadline time and date and through alert settings, the system will be able to notify its team members and managers before one week. With the help of this, they can check their project status and take necessary actions if required. The system will also be able to categorize the projects upon different modules and have look on them as and when required. This system will also be able to provide the list of employees who are involved project for particular module and time involved in development work. Through it's built in report builder section, it can prepare the clear picture of time invested on different level of projects.

## 2.3 PROBLEM DEFINITION AND PROJECT DESCRIPTION

The project titled “PROJECT MANAGEMENT SYSTEM” is a web based, React based application. It offers seamless project management, time handling, and members interaction capabilities. Designed for scalability and security, the system incorporates efficient database management and a resilient architecture. It features comprehensive administrative tools for site management and user-friendly modules for a streamlined shopping experience. With a focus on flexibility and expansion, it provide a efficient way for a company project managers to keep tracking the activities of their workers and provides in-depth analytics to optimize operations and drive informed decision-making.

- Admin
- User

### ADMIN MODULE

- ✓ View Projects and members
- ✓ Add projects and invite team members
- ✓ Update changes
- ✓ View feedback
- ✓ Remove users

### USER MODULE

- ✓ Login
- ✓ Checks project details
- ✓ Buy subscription for more features
- ✓ Join through invitation through UI Communicate with one another in project.

### 3.SYSTEM ANALYSIS

#### 3.1 REQUIREMENTS

##### SPECIFICATION                      HARDWARE

##### REQUIREMENTS

Processor	:	Intel(R) Core (TM) i5-7200U CPU
Speed	:	2.50 GHz -2.71GHz
Ram	:	8 GB
Hard disk	:	500 GB
Monitor	:	18.5” Color LED Monitor
Keyboard	:	108 keys Keyboard

##### SOFTWARE REQUIREMENTS

Operating System	:	WINDOWS 11, 64-bit OS
Database Connectivity	:	MYSQL
Front End	:	React
Back End	:	Spring Boot.

## 3.2 FEASIBILITY STUDY

Feasibility study is one of stage among important four stages of [Software Project Management Process](#). Hence, feasibility study is the feasibility analysis, or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyze whether software product will be right in terms of development, implementation, contribution of project to the organization etc. A feasibility analysis is conducted to decide if the solution considered to meet the criteria is feasible and workable in the software. During the feasibility study, information such as resource availability, cost estimates for software production, advantages of the software to the enterprise after its development, and cost to be expended on its maintenance is determined. The system has been tested for feasibility in the following points:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

### TECHNICAL FEASIBILITY

Technical feasibility is an assessment of whether a proposed project, product, or service can be successfully implemented using current or available technology. The following are the activities often performed by technical feasibility.

- ✓ Leverage widely adopted tools and frameworks known for their reliability and scalability.
- ✓ Well skilled developers in modern web development technologies facilitate to deal with the unexpected issues and helps with introduction of new update eventually.
- ✓ Existing integration solutions and middleware can mitigate potential challenges.
- ✓ Efficient data management, quick retrieval, and secure storage of sensitive information.

The current system developed is technically feasible as it provides the technical guarantee of efficiency, reliability, security and easy access to the users.

### 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 following are the operations carried out by operational feasibility:

- ✓ An Intuitive interface for managing products, orders, and payments efficiently
- ✓ Provides seamless, responsive managing experience with easy navigation, secure monitoring , and reliable employee support.
- ✓ Ensures proper working of the system if it is being developed and implemented.
- ✓ Regular updates and maintenance to ensure security and performance.

This Project Management System would ensure the optimal utilization of computer resources and would help in the improvement of performance status.

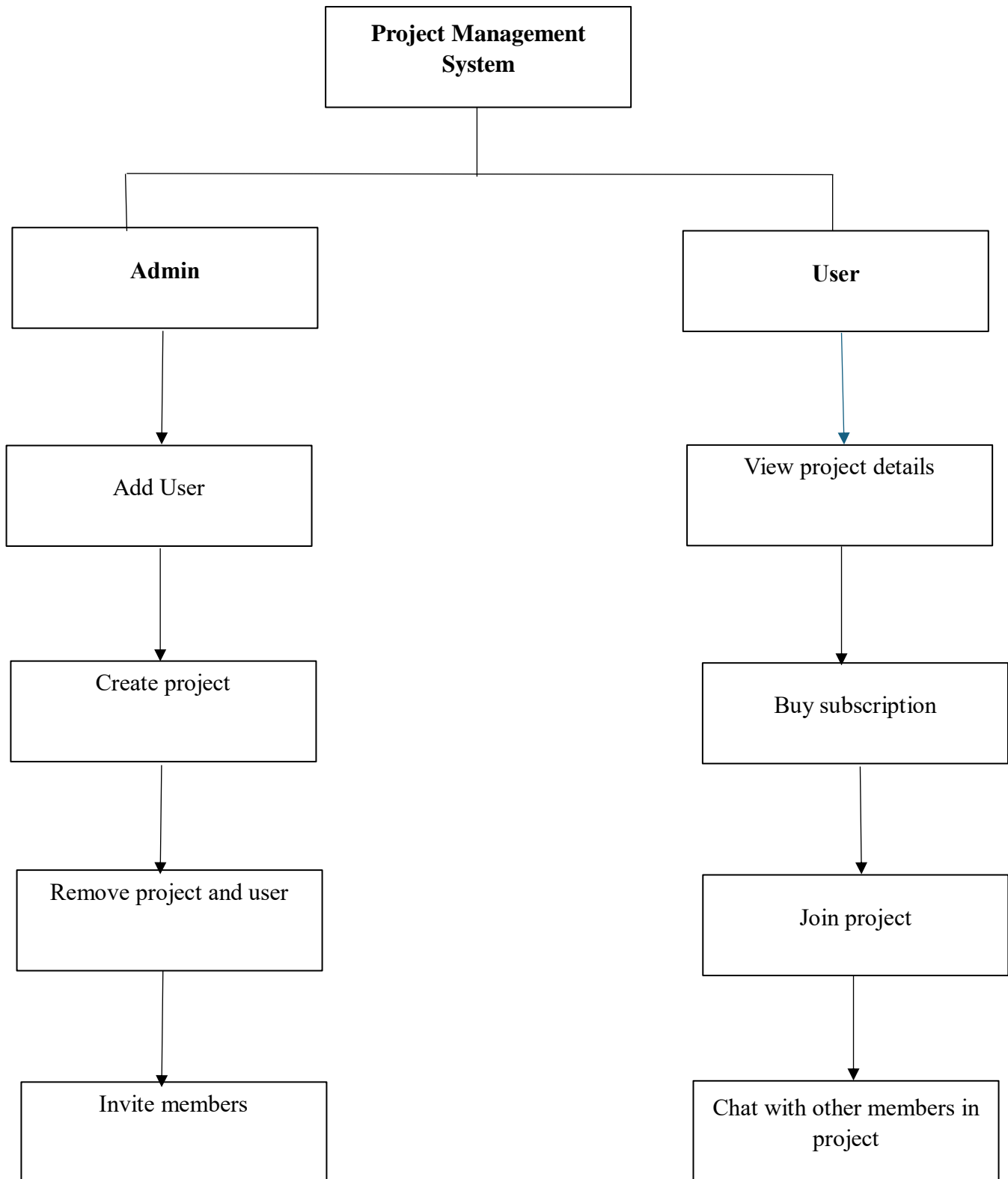
### **ECONOMIC FEASIBILITY**

Economic feasibility 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. If benefits outweigh costs, then the decision is made to design and implement the system. Software is economically feasible when it focuses on the issues listed below.

- ✓ Expense incurred on software development for achieving long-term gains for an organization.
- ✓ Expenses required to conduct elicitation and requirements analysis
- ✓ Income from direct sales of electrical and electronic products, potential premium memberships, and targeted advertisements.

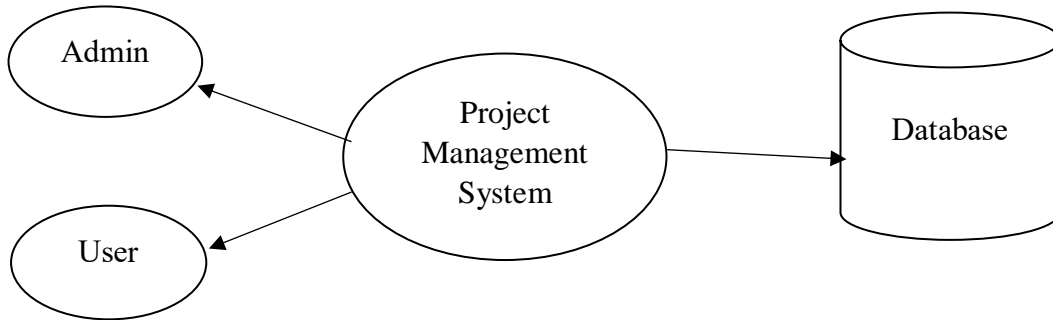
This system is economically feasible. Since this system is developed using the existing resources and technologies, there is nominal expenditure which ensures the economic feasibility of the system.s

#### 4.1 ARCHITECTURAL DESIGN:

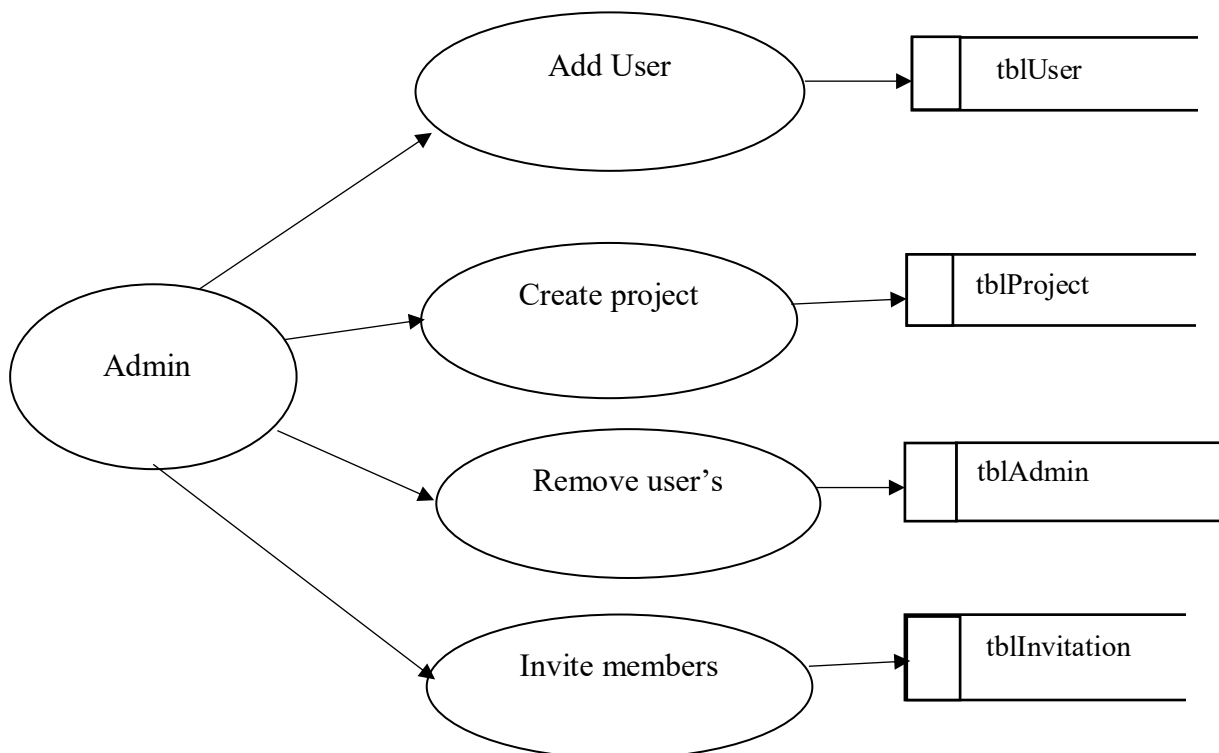


## 4.2 DATA FLOW DIAGRAM

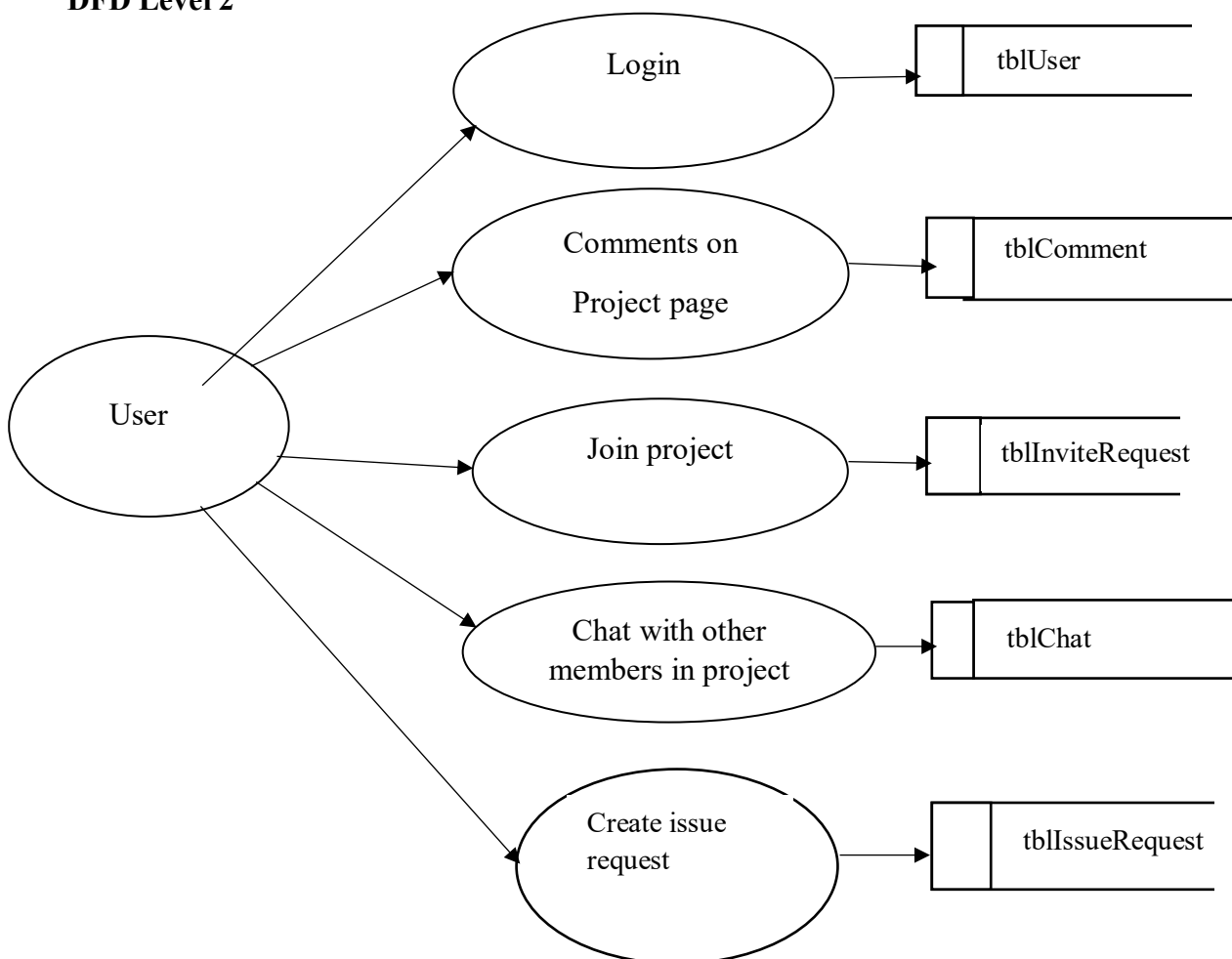
### DFD Level 0



### DFD Level 1





**DFD Level 2**

### 4.3 DATA DICTIONARY

**Table Name:** tblChat

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each chat message
2	message	varchar	225	Not Null	The content of the chat message
3	sender	varchar	100	Not Null	The name of the message sender
4	timestamp	timestamp	-	Not Null	The timestamp of when the message was sent

**Table Name:** tblComment

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each comment
2	text	varchar	225	Not Null	The content of the comment
3	issueId	bigint	-	Not Null	ID of the related issue
4	author	varchar	100	Not Null	The name of the comment's author
5	timestamp	timestamp	-	Not Null	The timestamp of when the comment was created or posted

**Table Name:** tblInvitation

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each invitation
2	token	varchar	225	Not Null	Unique token associated with the invitation
3	email	varchar	225	Not Null	Email address of the invited user
4	created_at	timestamp	-	Not Null	The timestamp of when the invitation was created
5	expires_at	timestamp	-	Null	The timestamp of when the invitation expires

**Table Name:** tblInviteRequest

S. No.	Field Name	Data type	Size	Constraint	Description
1	projectId	bigint	-	Not Null	ID of the project to which the invitation is related
2	email	varchar	225	Not Null	Email address of the person being invited

**Table Name:** tblIssue

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each issue
2	title	varchar	225	Not Null	Title or brief summary of the issue
3	description	text	-	Null	Detailed description of the issue
4	status	varchar	50	Not Null	Current status of the issue
5	priority	varchar	50	Null	Priority level of the issue (e.g., low, medium, high)
6	projectId	bigint	-	Not Null	ID of the project associated with this issue
7	created_at	timestamp	-	Not Null	The timestamp when the issue was created
8	updated_at	timestamp	-	Null	The timestamp when the issue was last updated

**Table Name:** tblMessage

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each message
2	content	text	-	Not Null	The content of the message
3	chatId	bigint	-	Not Null	ID of the chat session to which this message belongs
4	sender	varchar	100	Not Null	The name of the person who sent the message
5	createdAt	timestamp	-	Not Null	The timestamp when the message was created or sent

**Table Name:** tblProject

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each project
2	name	varchar	225	Not Null	Name of the project
3	description	text	-	Null	Detailed description of the project
4	ownerId	bigint	-	Not Null	ID of the user who owns the project
5	teamId	bigint	-	Null	Array of user IDs representing the team members
7	created_at	timestamp	-	Not Null	The timestamp when the project was created
8	updated_at	timestamp	-	Null	The timestamp when the project was last updated

**Table Name:** tblUser

S. No.	Field Name	Data type	Size	Constraint	Description
1	id	bigint	-	Primary Key	Unique identifier for each user
2	email	varchar	255	Not Null	Email address of the user
3	password	varchar	255	Not Null	Hashed password for the user
4	first_name	varchar	100	Null	First name of the user
5	last_name	varchar	100	Null	Last name of the user
7	created_at	timestamp	-	Not Null	The timestamp when the user was created
8	updated_at	timestamp	-	Null	The timestamp when the user was last updated

**Table Name:** tblAdmin

S. No	Field Name	Data type	Size	Constraint	Description
1	id	int	10	Primary Key	Unique identifier for admin
2	email	varchar	255	Not Null	Admin's email
3	password	varchar	16	Not Null	User's email

## 4.4 User Interface

### REGISTRATION:

Registration

Email

Fullname

password

register


### Login:

email

password

login

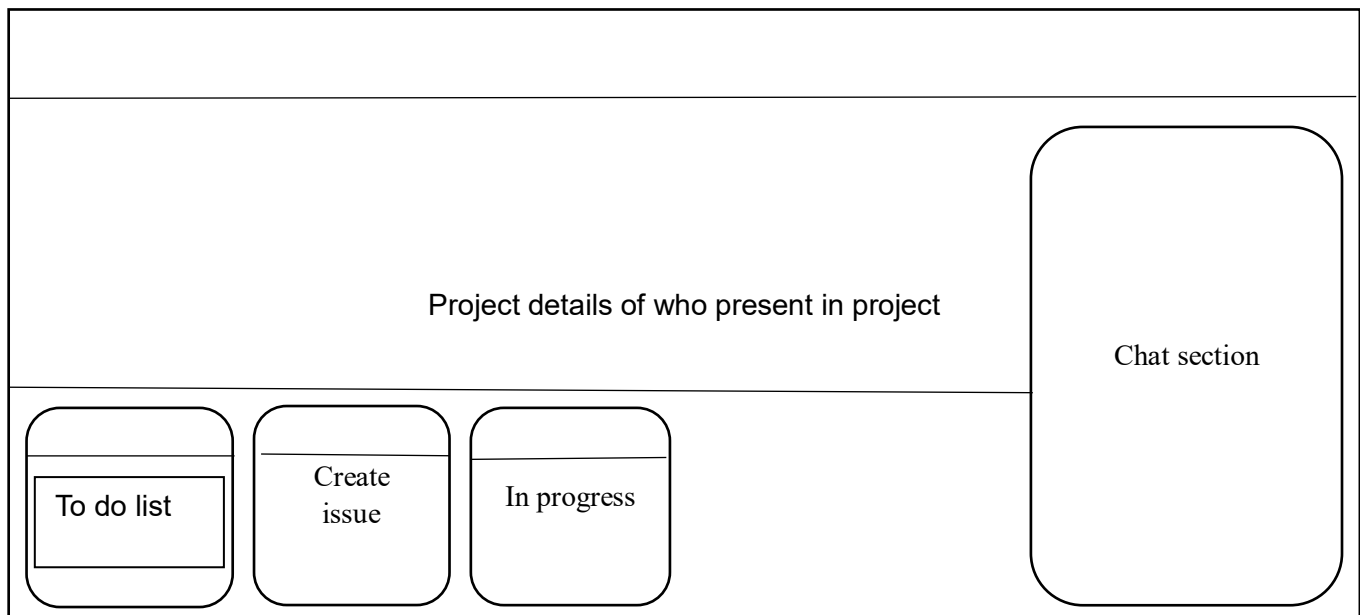
**Home page:**

Project management system		newproject	upgrade	
<div>Filter content</div>	Project info			
	Project info			
	Project info			

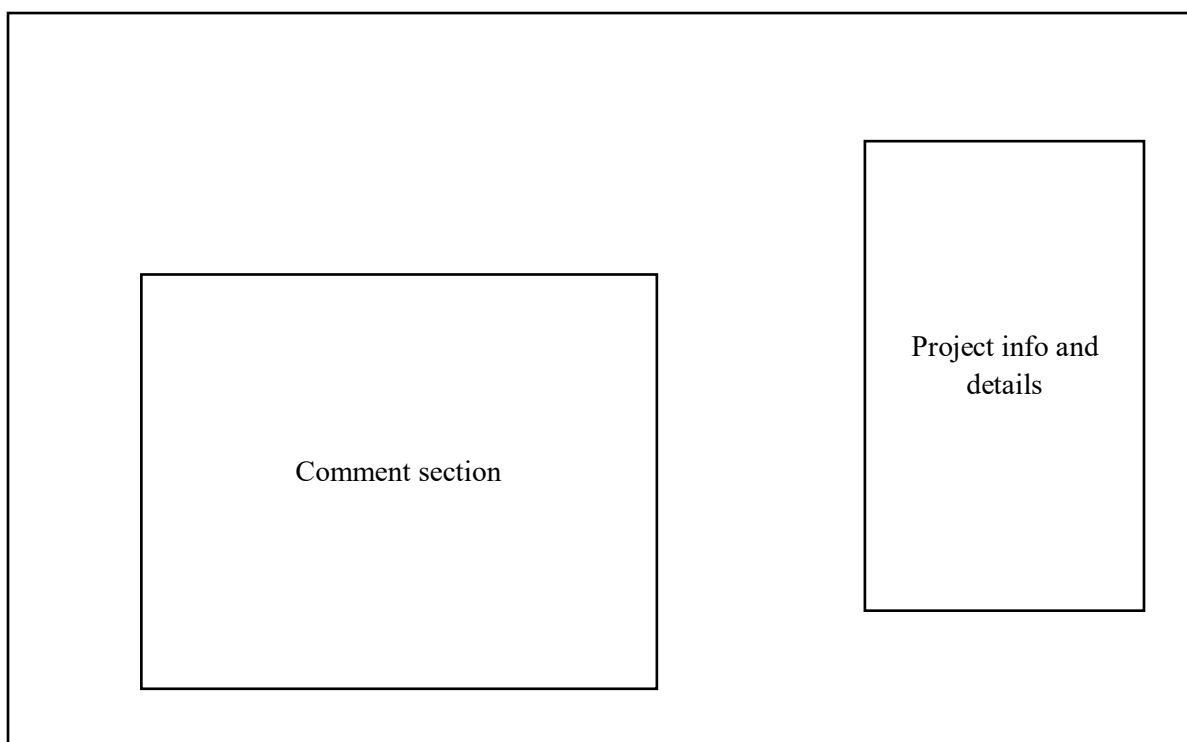
**Creating New Project:**

<div>Project name</div> <div>description</div> <div>category</div> <div>language</div> <div>ok</div>
--

**Project details page:**



**Comment page:**





## Invitation Page

[Click here to join!!!](#)

## 4.5 Normalization

Normalization is the process of organizing the data in the database. Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization divides the larger table into the smaller table and links them using relationship. The normal form is used to reduce redundancy from the database table.

### First Normal Form

For a table to be in the First Normal Form, it should follow the following 4 rules

- ✓ It should only have single(atomic) valued attributes/columns
- ✓ Values stored in a column should be of the same domain
- ✓ All the columns in a table should have unique names
- ✓ And the order in which data is stored, does not matter

### Second Normal Form

For a table to be in the Second Normal Form

- ✓ It should be in the First Normal form
- ✓ And, it should not have Partial Dependency. Partial Dependency occurs when a non prime attribute is functionally dependent on part of a candidate key

### Third Normal Form

A table is said to be in the Third Normal Form when,

- ✓ It is in the Second Normal form
- ✓ And, it doesn't have Transitive Dependency

### Boyce and Codd Normal Form (BCNF)

Boyce and Codd Normal Form is a higher version of the Third Normal form. This form deals with certain type of anomaly that is not handled by 3NF. A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF. For a table to be in BCNF, following conditions must be satisfied

- ✓ R must be in 3rd Normal Form
- ✓ For each functional dependency ( $X \rightarrow Y$ ), X should be a super Key

### Fourth Normal Form

A table is said to be in the Fourth Normal Form when

- ✓ It is in the Boyce-Codd Normal Form
- ✓ And, it doesn't have Multi-Valued Dependency.

## **5.1 TYPES OF TESTING**

Testing is the major quality measure technique employed during software development process. After the coding phase, computer programs are available that can be executed for testing purpose. Testing not only has to uncover errors introduced during coding, but also locates errors committed during the previous phase. Thus the aim of testing is to uncover requirements, design or coding errors in the program.

The basic types of testing are:

- Unit testing
- Integration testing
- Validation testing
- Output testing
- User Acceptance testing

### **UNIT TESTING**

This is the first level of testing. In this different modules are tested against the specification produced during the design of the modules. Unit testing is done for the verification of code produced during the coding of single program module in an isolated environment. Unit testing first focuses on the modules independently of one another to locate errors.

### **INTEGRATION TESTING**

After the modules are tested individually, they must be tested in combination with each other to be sure that the interfaces are correct. This is known as integration testing. Hence, we consider interfacing of various modules. Thus in the integration testing step, all the errors uncovered are corrected for the next testing steps.

## **VALIDATION TESTING**

Validation testing gives the final assurances that the software meets all functional, behavioral and performance requirements. The software is completely assembled as a package. Validation succeeds when the software functions in a manner in which the user expects. Validation refers to the process of using software in a live environment in order to find errors. If the password was given wrongly by user then it shows the check password error. Then if the username and password are not typed correct then it shows check username and password error. In the field, food quantity if the customers type any character other than numbers then it displays a warning message to give only numbers.

## **OUTPUT TESTING**

After performing the validation testing the next step is output testing of the proposed system since no system could be useful if it does not produce the required output generated or considered into two ways, one is on screen and another is printed format. The output formation the screen is found to be correct as the format was designed in the system design phase according to the user needs. If the user gives their correct username and password then it logs in to the corresponding page.

## **USER ACCEPTANCE TESTING**

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes where required.

## **5.2 TYPES OF VALIDATIONS**

A Validation control enables to validate an input and display an error message if necessary. Validation types are given below

### **REQUIRED FIELD VALIDATION**

The Required Field Validator is actually very simple, and yet very useful. One can use it to make sure that the user has entered something in a Text Box control. In every form required field validator is assigned to fulfil all the specification .

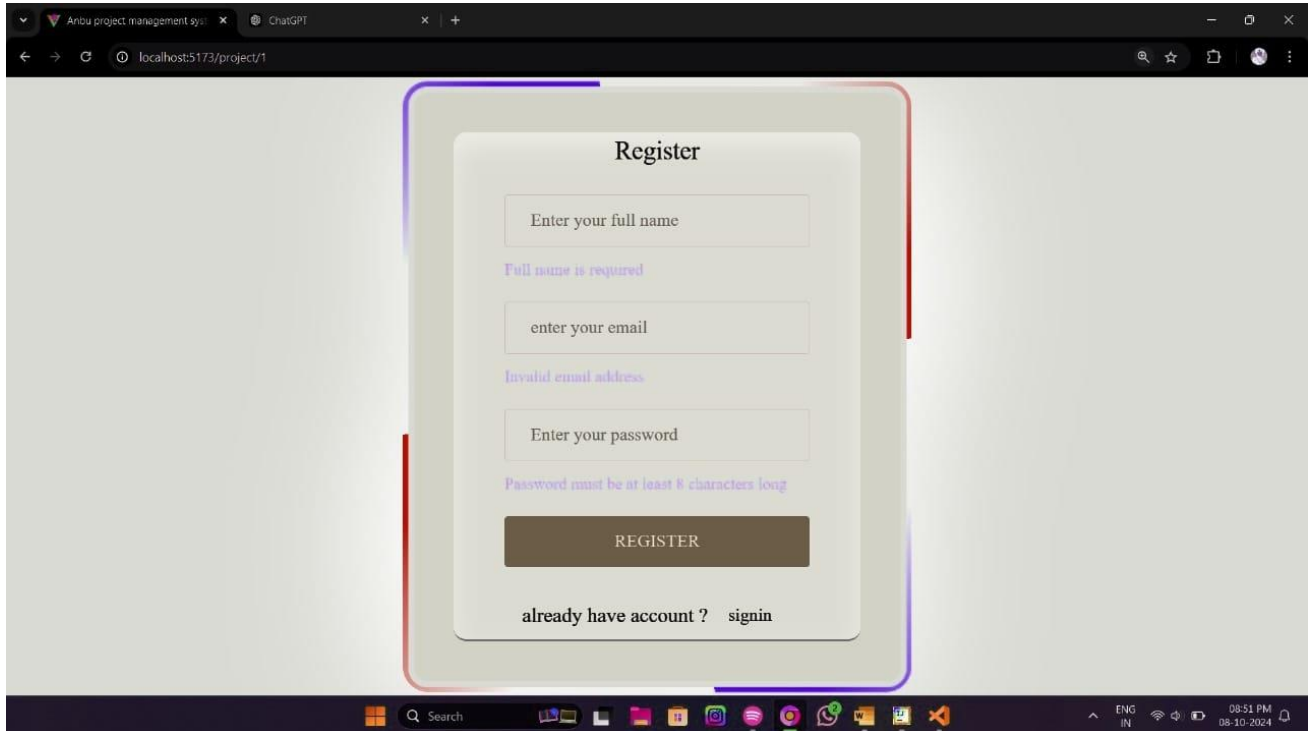
### **REGULAR EXPRESSION VALIDATION**

Regular Expression Validator is one of the most useful validators, because it can be used to check the validity of any kind of string. In this project regular expression validator is assigned for email checking entries.

### **RANGE VALIDATION**

The Range Validator does exactly what the name implies; it makes sure that the user input is within a specified range. It is used to validate numbers, strings and dates, which can make it useful in a bunch of cases. In this project, range validator is assigned for checking phone numbers.

### 5.3 ERROR MESSAGES



A screenshot of a web browser displaying the 'Register' form. The browser's address bar shows 'localhost:5173/project/1'. The form is titled 'Register' and contains three input fields: 'Enter your full name', 'enter your email', and 'Enter your password'. Below the first field is the error message 'Full name is required'. Below the second field is 'Invalid email address'. Below the third field is 'Password must be at least 8 characters long'. A 'REGISTER' button is at the bottom of the form, and below it is the text 'already have account ?' followed by a 'signin' link. The browser's taskbar at the bottom shows the Windows logo, a search bar, and various application icons. The system clock in the bottom right corner indicates 08:51 PM on 08-10-2024.

Register

Enter your full name

Full name is required

enter your email

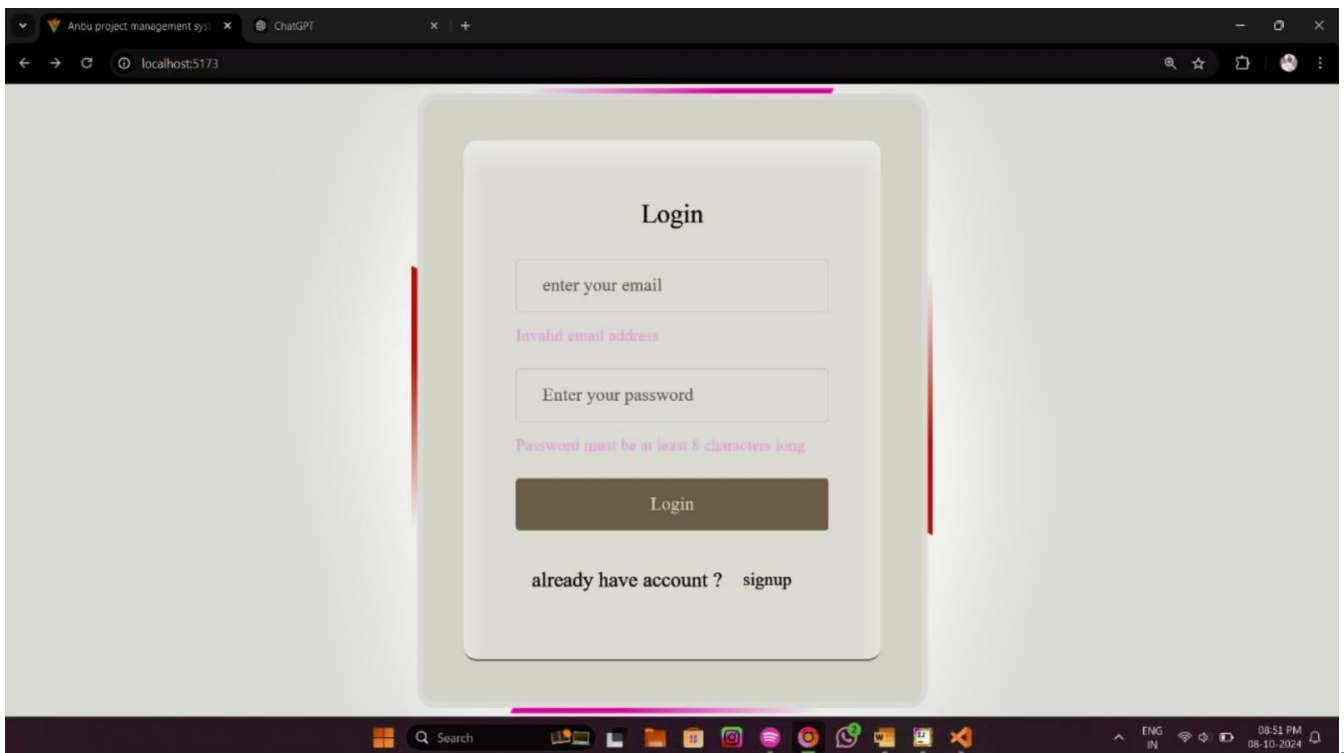
Invalid email address

Enter your password

Password must be at least 8 characters long

REGISTER

already have account ? [signin](#)



A screenshot of a web browser displaying the 'Login' form. The browser's address bar shows 'localhost:5173'. The form is titled 'Login' and contains two input fields: 'enter your email' and 'Enter your password'. Below the first field is the error message 'Invalid email address'. Below the second field is 'Password must be at least 8 characters long'. A 'Login' button is at the bottom of the form, and below it is the text 'already have account ?' followed by a 'signup' link. The browser's taskbar at the bottom shows the Windows logo, a search bar, and various application icons. The system clock in the bottom right corner indicates 08:51 PM on 08-10-2024.

Login

enter your email

Invalid email address

Enter your password

Password must be at least 8 characters long

Login

already have account ? [signup](#)

## 6.1 INSTALLATION MANUAL

- Download the required software and Unzip the files.
- Install all the software one by one.
- Install MySQL and Java
- Open the Workbench
- Open vs code for react app and open the port for live actions
- Open your web browser through terminal in vs code

If the browser shows the MYSQL version and other things, it means the MYSQL is successfully installed.

## 6.2 OPERATIONAL MANUAL

- Open the project and open the home page
- First click on login and enter Username and Password
- After logged in, click projects to view the details of the projects.
- Next click add project to add new projects.
- Next click add category to the project category.
- In View project, anyone can view the users and work allotted to workers. The employee can be added by the admin.
- Then the Admin can logout from the page
- The users had to first sign in to update their works
- The user can view the status of the project, posted by admin.
- Then the user can logout.

## **7.1 SPECIAL FEATURES OF LANGUAGES :**

### **JAVA:**

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages.

Java gained popularity shortly after its release, and has been a very popular programming language since then. Java was the third most popular programming language in 2022 according to GitHub. Although still widely popular, there has been a gradual decline in use of Java in recent years with other languages using JVM gaining popularity.



## UNIQUE FEATURES OF SPRINGBOOT:

**Spring boot**, which is developed by the renowned Spring team, has made Java Programming much easier. Think of it as a toolbox full of handy tools that save a lot of time and effort for Java developers and make the user experience much better. With the Spring Boot, developers don't have to write the same code over and over, saving them time. Basically, Spring Boot has tools that make hard tasks easy. **Spring Boot** helps Java web application development through its various features to build stable and reliable applications with minimal setup. These features not only make the process more simple but also increase the efficiency of developers. Now, Let's dive into 10 key features that make Spring Boot a go-to choice for Java developers.

**Flexibility:** Spring Boot transforms how you approach Java programming tasks, radically streamlining your experience. Spring Boot combines necessities such as an application context and an auto-configured, embedded web server to make microservice development a cinch. To go even faster, you can combine Spring Boot with Spring Cloud's rich set of supporting libraries, servers, patterns, and templates, to safely deploy entire microservices-based architectures into the cloud, in record time.

**Cross-platform compatibility:** Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run".

We take an opinionated view of the Spring platform and third-party libraries so you can get started with minimum fuss. Most Spring Boot applications need minimal Spring configuration.

**Open Source:** The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. The framework does not impose any specific programming model [citation needed]. The framework has become popular in the Java community as an addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is free and open source software.

**Error reporting and exceptions:** SPRINGBOOT supports more errors reporting constants to generate errors and relevant warning at runtime. For example, ERROR, WARNING.

**Active community support:** Red Hat OpenShift and Red Hat Enterprise Linux (RHEL) customers deploying Spring Boot applications can now comfortably do so with Red Hat community support for Spring Boot. This is a step to ensure the user experience and adoption of runtimes on the OpenShift platform for the developer community. Red Hat also supports other runtimes such as Quarkus and Node.js, and has for many years provided container images for OpenJDK on RHEL and UBI.

**Maintenance:** spring-boot-starter-maintenance is an extensible auto-configuration library for spring boot web and security projects supporting Java and Kotlin applications to block access to your application during maintenances and still provide a secure open door for your maintainers.

**Memory and CPU usage information:** SPRINGBOOT can provide memory usage information from functions like `memory_get_usage()` or `memory_get_peak_usage()`, which can help the developers optimize their code. In the similar way, the CPU power consumed by any script can be retrieved for further optimization.

## MySQL

MySQL is an open source relational database management system (RDMBS). It's name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for structured Query Language.

MySQL is free and open-source software under the terms of the GNU general public license, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by sun Microsystems. In 2010 when oracle acquired sun, Widenius forked the open-source MySQL project to create MariaDB.

MySQL is a component of the lamp web application software stack, which is an acronym for Linux, Apache, MySQL, and Perl. MySQL is used by many database-driven web applications, including drupal, joomla, phpBB, and Word Press. MySQL is also used by many popular websites, including Google, face book, Twitter and You Tube. The main features of MySQL includes

**Easy to use:** MySQL is easy to use. We have to get only the basic knowledge of SQL. We can build and interact with MySQL by using only a few simple SQL statements.

**It is secure:** MySQL consists of a solid data security layer that protects sensitive data from intruders. Also, passwords are encrypted in MySQL.

**Client/ Server Architecture:** MySQL follows the working of a client/server architecture. There is a database server (MySQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they can query data, save changes, etc.

**Free to download:** MySQL is free to use so that we can download it from MySQL official website without any cost.

**Compatible on many operating systems:** MySQL is compatible to run on many operating systems, like Novell NetWare, Windows\* Linux\*, many varieties of UNIX\* (such as Sun\* Solaris\*, AIX, and DEC\* UNIX), OS/2, FreeBSD\*, and others. MySQL also provides a facility that the clients can run on the same computer as the server or on another computer (communication via a local network or the Internet).

## **8 FUTURE ENHANCEMENT**

The project “Project management system” can have the further updates. The application can be extended further by connecting all the areas in the city.

Implementation of wide frameworks and networks such as node and ai tools to provide users the tools they want.

This project can be enhanced into primary application for any company that need its use with its own customization. It can even provide any additional tools that a software developer need in his project in which he’s working on.

## 9 CONCLUSION

The arrival of technology and the escalating use of internet have reduced our burden of work. The orders can be posted and the status of those orders can be viewed anywhere and at any time by the user. All the processes are transparent to the admin. In comparison with the manual system, the benefit under a computer system is considerable in saving man power, working hours and efforts.

Various validation techniques have been used to implement accuracy of data in all formats of input. Updating of information becomes so easier. The system has produced all the reports required for the management. It is concluded that the application works well and satisfies the users.

The application is tested very well and errors are properly debugged. The project works according to the restrictions provided to the users respectively. Further enhancements can be made to the application, so that the “Project Management Systems” can be functioned more efficiently than the present one.

## **10 BIBLIOGRAPHY**

### **BOOK REFERENCES**

- Spring Boot: Up and Running - Building Cloud Native Java and Kotlin Applications (Grayscale Indian Edition).
- Spring in Action(5th Edition); Spring Micro services in Action; Reactive Spring; Spring Boot in Action; Expert Spring MVC and Web Flow.
- Spring Boot 3 and Spring Framework 6 (Grayscale Indian Edition) by Christian Ullenboom.

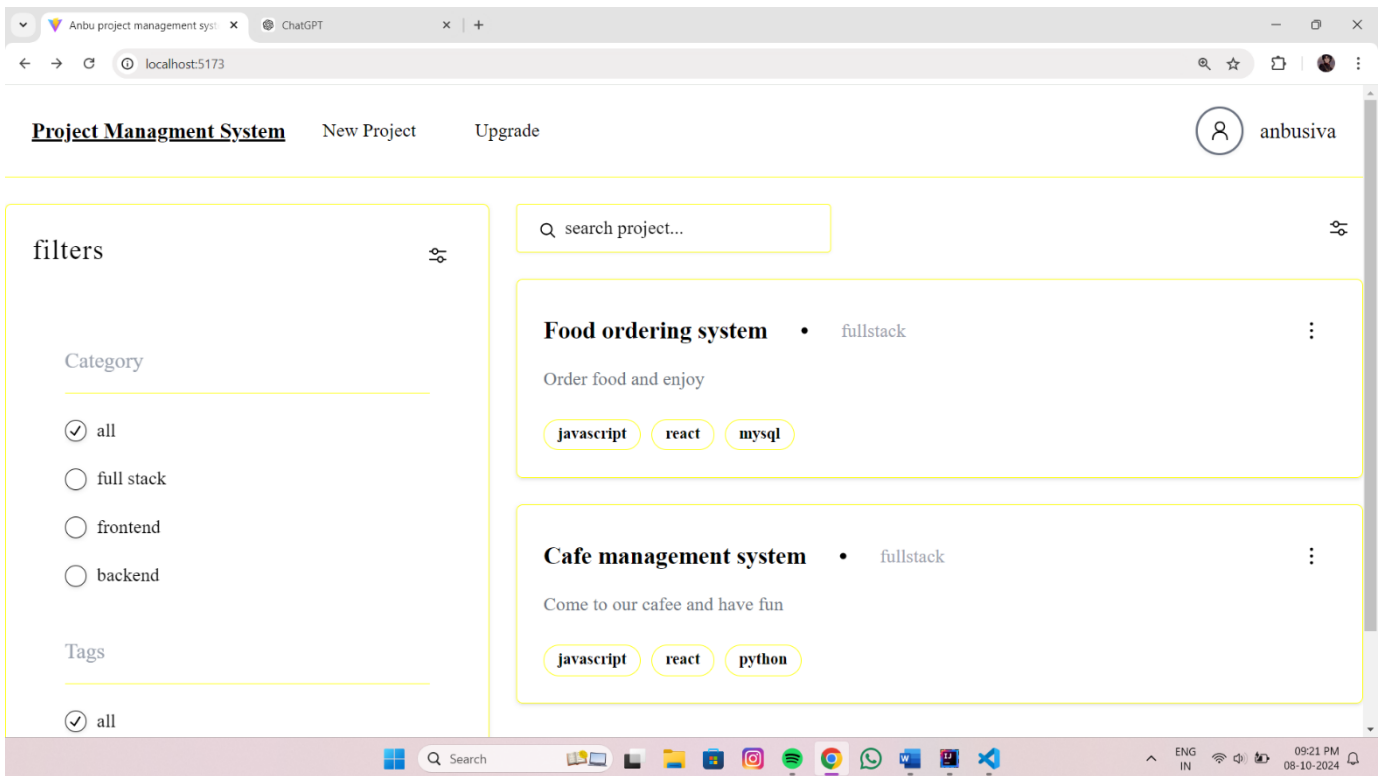
### **WEBSITE REFERENCES**

- [www.tutorialspoint.com](http://www.tutorialspoint.com)
- [www.spring.io](http://www.spring.io)
- [www.codecademy.com](http://www.codecademy.com)
- [www.stackoverflow.com](http://www.stackoverflow.com)

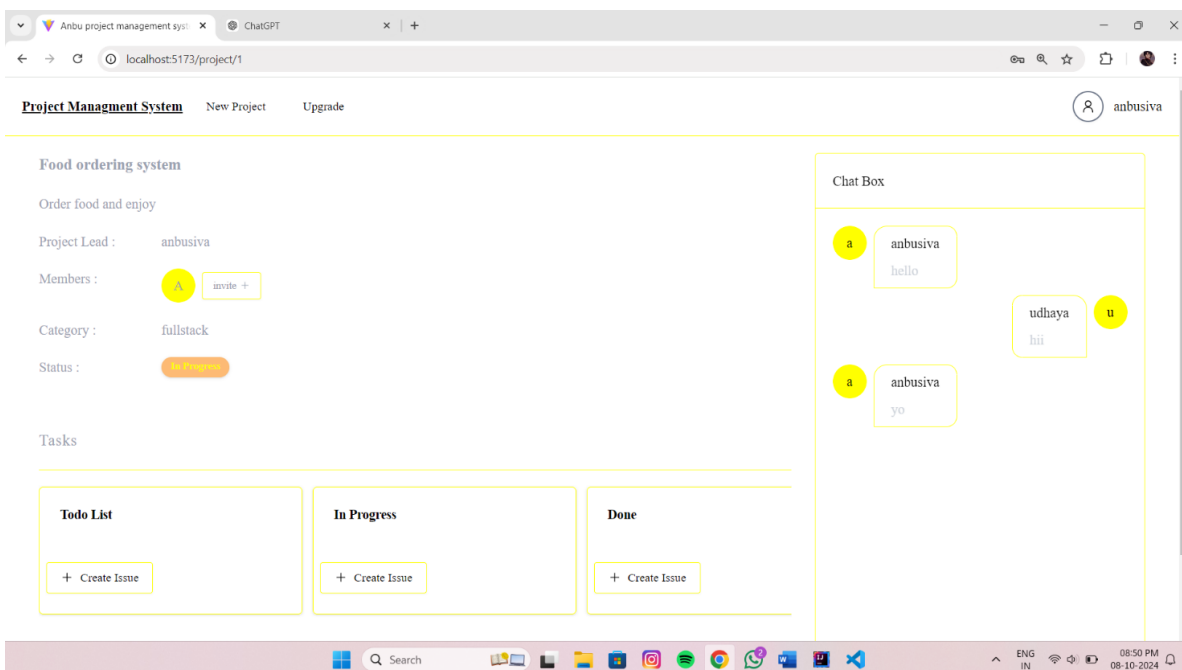
## 11 APPENDIX

### 11.1 SAMPLE SCREEN LAYOUTS

#### App:



#### project details:



## Register:

The screenshot shows a web browser window with the URL `localhost:5173/project/1`. The page displays a 'Register' form with the following elements:

- Register** (Section Header)
- (Full name input field)
- Full name is required (Error message)
- (Email input field)
- Invalid email address (Error message)
- (Password input field)
- Password must be at least 8 characters long (Error message)
- REGISTER** (Submit button)
- already have account ? [signin](#) (Link to login page)

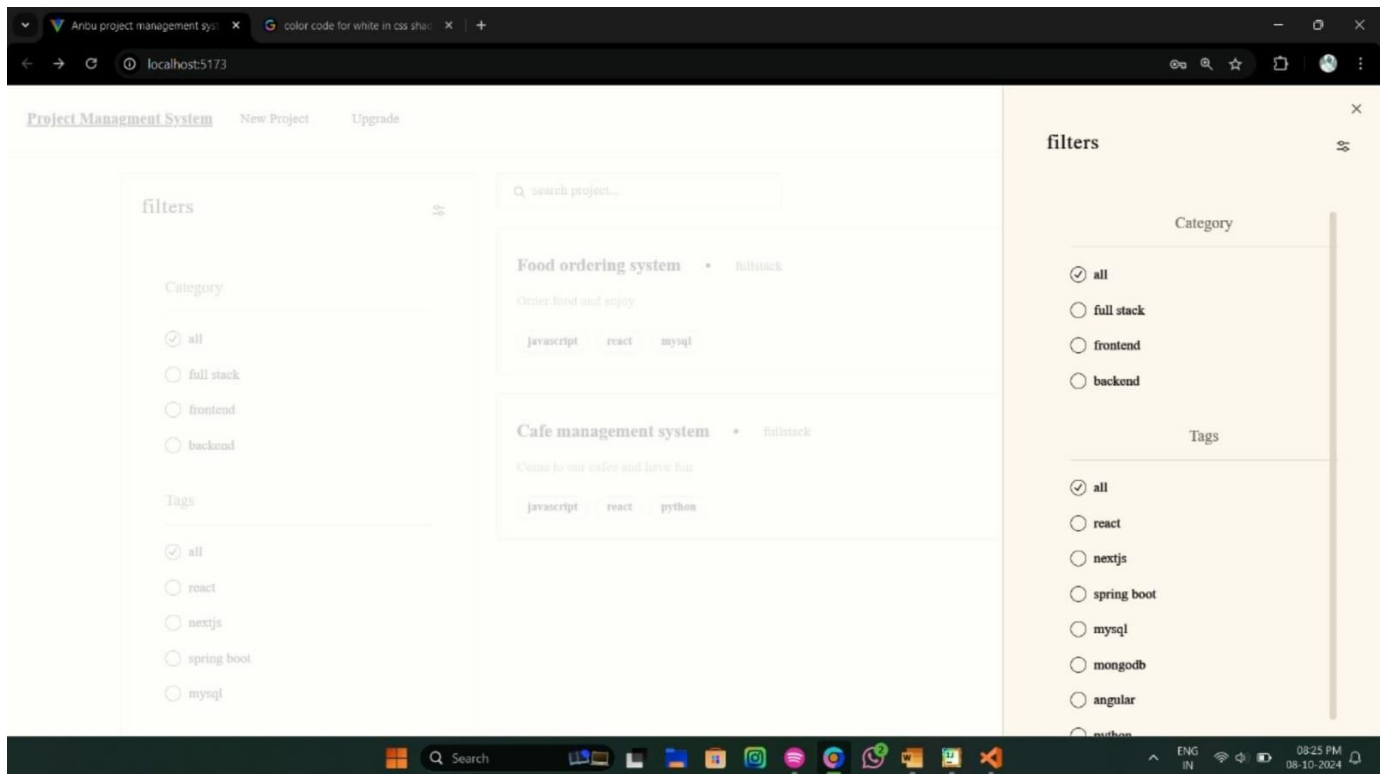
## USER LOGIN PAGE

The screenshot shows a web browser window with the URL `localhost:5173`. The page displays a 'Login' form with the following elements:

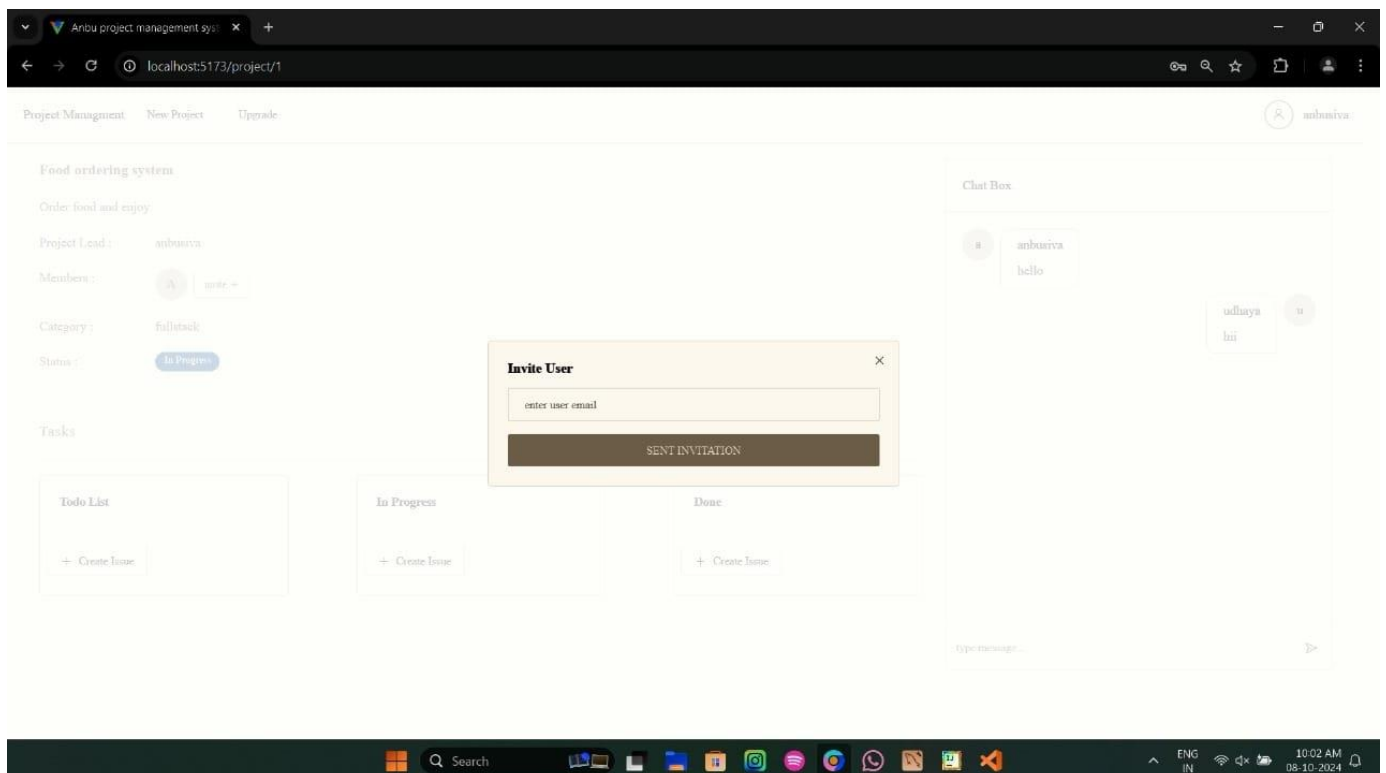
- Login** (Section Header)
- (Email input field)
- Invalid email address (Error message)
- (Password input field)
- Password must be at least 8 characters long (Error message)
- Login** (Submit button)
- already have account ? [signup](#) (Link to registration page)



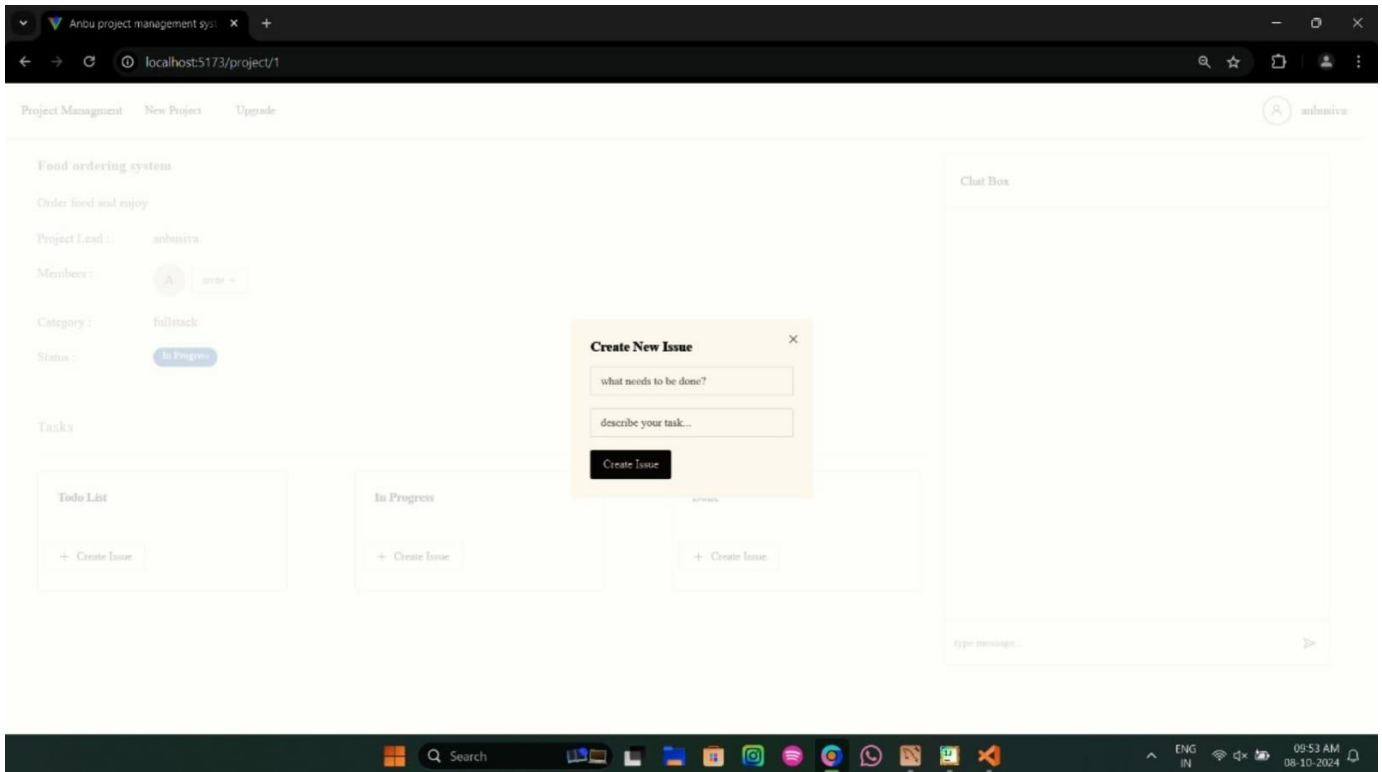
## Filter page:



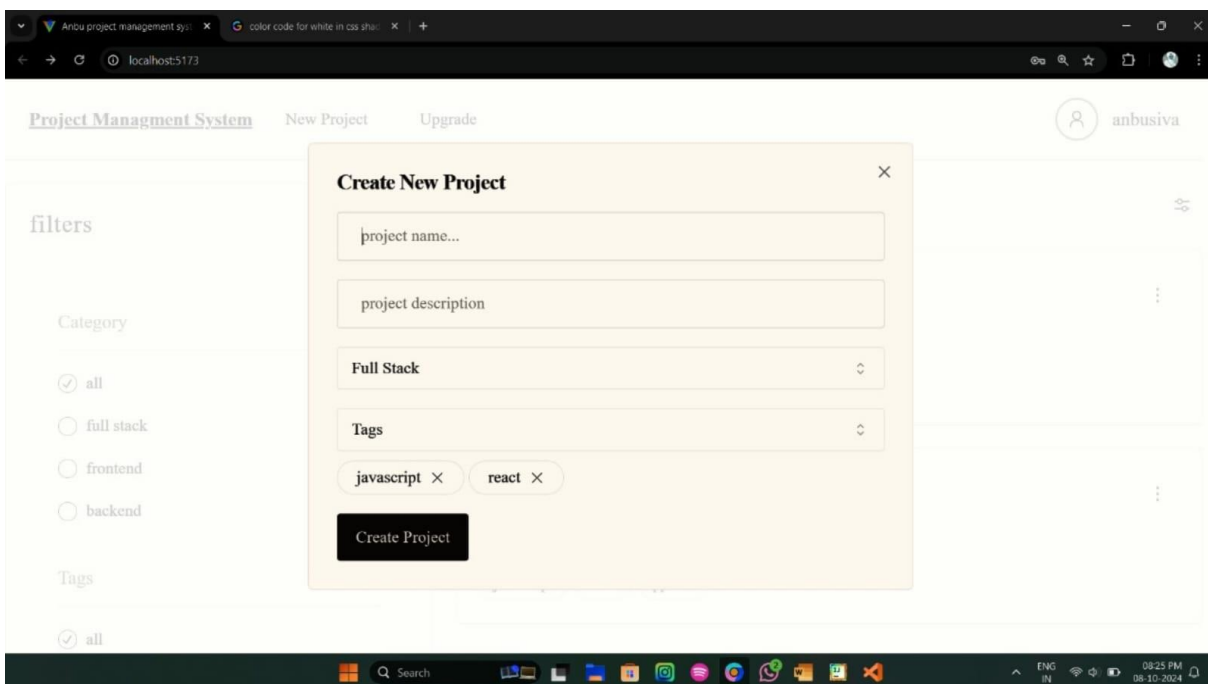
## Sent invitation:



## To do list:



## Project Creation:



## CHAT BOX:

The screenshot shows a web browser window with the URL `localhost:5173/project/1`. The page title is "Project Management System". The interface includes a header with "New Project" and "Upgrade" buttons, and a user profile "anbusiva". The main content area is titled "Food ordering system" and includes a section "Order food and enjoy". Below this, there are fields for "Project Lead : anbusiva", "Members : [Avatar] invite +", "Category : fullstack", and "Status : In Progress". A "Tasks" section contains three columns: "Todo List", "In Progress", and "Done", each with a "+ Create Issue" button. On the right, a "Chat Box" is open, showing a conversation with "anbusiva" (a) and "udhaya" (u). The chat messages are: "hello" (a), "hii" (u), and "yo" (a).

## SUBSCRIPTION:

The screenshot shows a web browser window with the URL `localhost:5173/upgrade_plan`. The page title is "Pricing". It features three pricing plans:

- Free**  
₹0/ FREE  
Current Plan  
Features:
  - ✓ Add only 3 projects
  - ✓ Basic Task Management
  - ✓ Project Collaboration
  - ✓ Basic Reporting
  - ✓ Email Notifications
  - ✓ Basic Access Control
- Monthly Paid Plan**  
₹799/ MONTHLY  
Get Started  
Features:
  - ✓ Add unlimited project
  - ✓ Access to live chat
  - ✓ Add unlimited team member
  - ✓ Advanced Reporting
  - ✓ Priority Support
  - ✓ Customization Options
  - ✓ Integration Support
  - ✓ Advanced Security
  - ✓ Training and Resources
  - ✓ Access Control
  - ✓ Custom Workflows
- Annual Paid Plan**  
₹6711/ ANNUALLY  
30% off  
Get Started  
Features:
  - ✓ Add unlimited project
  - ✓ Access to live chat
  - ✓ Add unlimited team member
  - ✓ Advanced Reporting
  - ✓ Priority Support
  - ✓ Everything which montly plan has

## Update project:

The screenshot shows a web browser window with the URL `localhost:5173/project/update/3`. The page title is "Project Management System". In the top right corner, there is a user profile icon labeled "anbusiva". The main content area contains a form titled "Update Project". The form has two text input fields: the first contains "asfaf" and the second contains "asf". Below these are two dropdown menus: "Category" and "Tags". The "Tags" dropdown is open, showing "javascript" and "react" as selected tags. At the bottom of the form is a blue button labeled "Update Project". The browser's taskbar at the bottom shows various application icons and the system clock indicating 09:16 PM on 08-10-2024.

## Delete:

The screenshot shows a web browser window with the URL `localhost:5173`. The page features a sidebar on the left with a "filters" section. Under "Category", there are radio buttons for "all" (selected), "full stack", "frontend", and "backend". Under "Tags", there are radio buttons for "all" (selected) and an empty one. The main content area has a search bar labeled "search project...". Below the search bar, there are two project cards. The first card is titled "Food ordering system" with a "fullstack" tag. It has a description "Order food and enjoy" and tags "javascript", "react", and "mysql". The second card is titled "Cafe management system" with a "fullstack" tag. It has a description "Come to our cafee and have fun" and tags "javascript", "react", and "python". To the right of the second card, there is a small menu with "Update" and "Delete" options. The browser's taskbar at the bottom shows various application icons and the system clock indicating 09:21 PM on 08-10-2024.

## 11.2 SAMPLE CODINGS

### **Projectmanagementapplication.java:**

```
package pmts;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class ProjectmanagementApplication {

    public static void main(String[] args) {
        SpringApplication.run(ProjectmanagementApplication.class, args);
    }
}
```

### **App.js:**

```
import { useEffect } from "react";
import Auth from "../pages/Auth/Auth";
import { useDispatch, useSelector } from "react-redux";
import { getUser } from "../redux/Auth/Action";
import { Route, Routes } from "react-router-dom";
import Home from "../pages/Home/Home";
import Navbar from "../pages/Navbar/Navbar";
import ProjectDetails from "../pages/Project/ProjectDetails";
import IssueDetails from "../pages/Issue/IssueDetails";
import UpdateProjectForm from "../pages/Project/UpdateProjectForm";
import Loader from "../pages/Loader/Loader";
import AcceptInvitation from "../pages/Project/AcceptInvitation";
import Subscription from "../pages/subscription/Subscription";
import UpgradeSuccess from "../pages/subscription/UpgradeSuccess";
import { getUserSubscription } from "../redux/Subscription/Action";
function App() {
    const dispatch = useDispatch();
    const { auth } = useSelector((store) => store);
    useEffect(() => {
        dispatch(getUser(auth.jwt || localStorage.getItem("jwt")));
        dispatch(getUserSubscription(auth.jwt || localStorage.getItem("jwt")));
    }, [auth.jwt]);
    return (
```

```
<>
{ auth.loading?<Loader/> : auth.user ? (
  <>
  <Navbar />
  <Routes>
    <Route path="/" element={<Home />}></Route>
    <Route path="/project/:id" element={<ProjectDetails />}></Route>
    <Route path="/project/update/:id" element={<UpdateProjectForm />}></Route>
    <Route
      path="/project/:projectId/issue/:issueId"
      element={<IssueDetails />}
    ></Route>
    <Route
      path="/accept_invitation"
      element={<AcceptInvitation />}
    ></Route>
    <Route path="/upgrade_plan" element={<Subscription />}></Route>
    <Route path="/upgrade_plan/success" element={<UpgradeSuccess
/>}></Route>
  </Routes>
</>
) : (
  <Auth />
)}
</>
);
}
export default App;
```

**AuthController.java:**

```
Package anbuproject;
Import anbuproject.*;
@RestController
@RequestMapping("/auth")
public class AuthController {
  @Autowired
  private UserRepository userRepository;
  @Autowired
  private PasswordEncoder passwordEncoder;
  @Autowired
  private CustomUserServiceImplementation customUserDetails;
```

```
@Autowired
private UserService userService;
@Autowired
private SubscriptionService subscriptionService;
@Autowired
private SubscriptionRepository subscriptionRepository;
@PostMapping("/signup")
public ResponseEntity<AuthResponse> createUserHandler(
    @RequestBody User user) throws UserException {
    String email = user.getEmail();
    String password = user.getPassword();
    String fullName = user.getFullName();
    String role=user.getRole();
    User isEmailExist = userRepository.findByEmail(email);
    if (isEmailExist!=null) {
        throw new UserException("Email Is Already Used With Another
Account");
    }
    // Create new user
    User createdUser = new User();
    createdUser.setEmail(email);
    createdUser.setFullName(fullName);
    createdUser.setPassword(passwordEncoder.encode(password));
    createdUser.setRole(role);
    User savedUser = userRepository.save(createdUser);
    Subscription subscription =
subscriptionService.createSubscription(savedUser);
//    subscriptionRepository.save(subscription);
    Authentication authentication = new
UsernamePasswordAuthenticationToken(email, password);
    SecurityContextHolder.getContext().setAuthentication(authentication);
    String token = JwtProvider.generateToken(authentication);
    AuthResponse authResponse = new AuthResponse();
    authResponse.setJwt(token);
    authResponse.setMessage("Register Success");
    return new ResponseEntity<AuthResponse>(authResponse, HttpStatus.OK);
}
@PostMapping("/signin")
public ResponseEntity<AuthResponse> signin(@RequestBody LoginRequest
loginRequest) {
```

```
String username = loginRequest.getEmail();
String password = loginRequest.getPassword();
System.out.println(username + " ----- " + password);
Authentication authentication = authenticate(username, password);
SecurityContextHolder.getContext().setAuthentication(authentication);
String token = JwtProvider.generateToken(authentication);
AuthResponse authResponse = new AuthResponse();
authResponse.setMessage("Login Success");
authResponse.setJwt(token);
return new ResponseEntity<AuthResponse>(authResponse, HttpStatus.OK);
}

private Authentication authenticate(String username, String password) {
    UserDetails userDetails =
customUserDetails.loadUserByUsername(username);
    System.out.println("sign in userDetails - " + userDetails);
    if (userDetails == null) {
        System.out.println("sign in userDetails - null " + userDetails);
        throw new BadCredentialsException("Invalid username or password");
    }
    if (!passwordEncoder.matches(password, userDetails.getPassword())) {
        System.out.println("sign in userDetails - password not match " +
userDetails);
        throw new BadCredentialsException("Invalid username or password");
    }
    return new UsernamePasswordAuthenticationToken(userDetails, null,
userDetails.getAuthorities());
}
}
```

**CommentController.java:**

```
package com.zosh.controller;

import com.zosh.exception.IssueException;
import com.zosh.exception.ProjectException;
import com.zosh.exception.UserException;
import com.zosh.model.Comment;
import com.zosh.model.User;
import com.zosh.request.CreateCommentRequest;
import com.zosh.response.MessageResponse;
import com.zosh.service.CommentService;
import com.zosh.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
```



```
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/comments")
public class CommentController {

    private CommentService commentService;
    private UserService userService;

    @Autowired
    public CommentController(CommentService commentService, UserService userService) {
        this.commentService = commentService;
        this.userService = userService;
    }

    @PostMapping()
    public ResponseEntity<Comment> createComment(

        @RequestBody CreateCommentRequest req,
        @RequestHeader("Authorization") String jwt) throws UserException,
        IssueException, ProjectException {
        User user = userService.findUserProfileByJwt(jwt);
        Comment createdComment = commentService.createComment(req.getIssueId(),
        user.getId(), req.getContent());
        return new ResponseEntity<>(createdComment, HttpStatus.CREATED);
    }

    @DeleteMapping("/{commentId}")
    public ResponseEntity<MessageResponse> deleteComment(@PathVariable Long
    commentId,

        @RequestHeader("Authorization") String jwt) throws
    UserException, IssueException, ProjectException {
        User user = userService.findUserProfileByJwt(jwt);
        commentService.deleteComment(commentId, user.getId());
        MessageResponse res=new MessageResponse();
        res.setMessage("comment deleted successfully");
        return new ResponseEntity<>(res, HttpStatus.OK);
    }

    @GetMapping("/{issueId}")
```

```
    public ResponseEntity<List<Comment>> getCommentsByIssueId(@PathVariable Long
issueId) {
        List<Comment> comments = commentService.findCommentByIssueId(issueId);
        return new ResponseEntity<>(comments,HttpStatus.OK);
    }
}
```

**MessageServiceImpl.java:**

```
package com.zosh.service;
```

```
import java.time.LocalDateTime;
```

```
import java.util.List;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Service;
```

```
import com.zosh.exception.ChatException;
```

```
import com.zosh.exception.ProjectException;
```

```
import com.zosh.exception.UserException;
```

```
import com.zosh.model.Chat;
```

```
import com.zosh.model.Message;
```

```
import com.zosh.model.User;
```

```
import com.zosh.repository.ChatRepository;
```

```
import com.zosh.repository.MessageRepository;
```

```
import com.zosh.repository.UserRepository;
```

```
@Service
```

```
public class MessageServiceImpl implements MessageService {
```

```
    @Autowired
```

```
    private MessageRepository messageRepository;
```

@Autowired

```
private UserRepository userRepository;
```

@Autowired

```
private ProjectService projectService;
```

@Override

```
public Message sendMessage(Long senderId, Long projectId, String content) throws  
UserException, ChatException, ProjectException {
```

```
    User sender = userRepository.findById(senderId)
```

```
        .orElseThrow(() -> new UserException("User not found with id: " + senderId));
```

```
    Chat chat = projectService.getProjectById(projectId).getChat();
```

```
    Message message = new Message();
```

```
    message.setContent(content);
```

```
    message.setSender(sender);
```

```
    message.setCreatedAt(LocalDateTime.now());
```

```
    message.setChat(chat);
```

```
    Message savedMessage=messageRepository.save(message);
```

```
    chat.getMessages().add(savedMessage);
```

```
    return savedMessage;
```

```
}
```

@Override

```
public List<Message> getMessagesByProjectId(Long projectId) throws ProjectException,  
ChatException {
```

```
    Chat chat = projectService.getChatByProjectId(projectId);
```

```
        List<Message> findByChatIdOrderByCreatedAtAsc =  
messageRepository.findByChatIdOrderByCreatedAtAsc(chat.getId());  
        return findByChatIdOrderByCreatedAtAsc;  
    }  
}
```

**ProjectServiceImpl.java:**

```
package com.zosh.service;  
  
import java.util.List;  
import java.util.Optional;  
import java.util.stream.Collectors;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import com.zosh.exception.ChatException;  
import com.zosh.exception.ProjectException;  
import com.zosh.exception.UserException;  
import com.zosh.model.Chat;  
import com.zosh.model.Project;  
import com.zosh.model.User;  
import com.zosh.repository.ProjectRepository;  
  
import jakarta.transaction.Transactional;  
  
@Service  
public class ProjectServiceImpl implements ProjectService {  
  
    @Autowired
```

```
private ProjectRepository projectRepository;

@Autowired
private ChatService chatService;

@Autowired
private InvitationService inviteTokenService;

@Autowired
private UserService userService;

@Override
public Project createProject(Project project, Long id) throws UserException {
    User user = userService.findUserById(id);
    Project createdProject = new Project();

    createdProject.setOwner(user);
    createdProject.setTags(project.getTags());
    createdProject.setName(project.getName());
    createdProject.setCategory(project.getCategory());
    createdProject.setDescription(project.getDescription());
    createdProject.getTeam().add(user);

    System.out.println(createdProject);
    Project savedProject = projectRepository.save(project);

    savedProject.getTeam().add(user);

    Chat chat = new Chat();
    chat.setProject(savedProject);
    Chat projectChat = chatService.createChat(chat);
}
```

```
        savedProject.setChat(projectChat);

    }

    return savedProject;
}

@Override
public List<Project> getProjectsByTeam(User user,String category,String tag) throws
ProjectException {
    List<Project> projects=
projectRepository.findByTeamContainingOrOwner(user,user);

    if (category != null) {
        projects = projects.stream()
            .filter(project -> project.getCategory().equals(category))
            .collect(Collectors.toList());
    }

    if (tag != null) {
        projects = projects.stream()
            .filter(project -> project.getTags().contains(tag))
            .collect(Collectors.toList());
    }

    return projects;
}
```

@Override

```
public Project getProjectById(Long projectId) throws ProjectException {  
    Optional<Project> project = projectRepository.findById(projectId);  
    if(project.isPresent()) {  
        return project.get();  
    }  
    throw new ProjectException("No project exists with the id "+projectId);  
}
```

@Override

```
public String deleteProject(Long projectId, Long id) throws UserException {  
    User user = userService.findUserById(id);  
    System.out.println("user ____>" + user);  
    if(user != null) {  
        projectRepository.deleteById(projectId);  
        return "project deleted";  
    }  
    throw new UserException("User doesnot exists");  
}
```

@Override

```
public Project updateProject(Project updatedProject, Long id) throws  
ProjectException {  
    Project project = getProjectById(id);  
  
    if (project != null) {  
        // Update the existing project with the fields from updatedProject  
        if (updatedProject.getName() != null) {  
            project.setName(updatedProject.getName());  
        }  
    }  
}
```

```
        if (updatedProject.getDescription() != null) {
            project.setDescription(updatedProject.getDescription());
        }

        if (updatedProject.getTags() != null) {
            project.setTags(updatedProject.getTags());
        }

        // Save the updated project once
        return projectRepository.save(project);
    }

    throw new ProjectException("Project does not exist");
}

@Override
public List<Project> searchProjects(String keyword, User user) throws
ProjectException {
    String partialName = "%" + keyword + "%";
    // projectRepository.findByPartialNameAndTeamIn(partialName, user);

    List<Project> list =
projectRepository.findByNameContainingAndTeamContains(keyword,user);
    if(list!=null) {
        return list;
    }

    throw new ProjectException("No Projects available");
}

@Override
```



```
@Transactional

public void addUserToProject(Long projectId, Long userId) throws UserException,
ProjectException {

    Project project = projectRepository.findById(projectId).orElseThrow(() -> new
ProjectException("project not found"));

    User user = userService.findUserById(userId);

    if (!project.getTeam().contains(user)) {

        project.getChat().getUsers().add(user);
        project.getTeam().add(user);
        projectRepository.save(project);

    }

}
```

```
@Override

public void removeUserFromProject(Long projectId, Long userId) throws
UserException, ProjectException {

    Project project = projectRepository.findById(projectId)

        .orElseThrow(() -> new ProjectException("project not
found"));

    User user = userService.findUserById(userId);

    if (project.getTeam().contains(user)) {

        project.getTeam().remove(user);
        project.getChat().getUsers().remove(user);

    }

}
```

```
@Override

    public Chat getChatByProjectId(Long projectId) throws ProjectException,
ChatException {

        Project project = projectRepository.findById(projectId).orElseThrow(()-> new
ProjectException("Project not found"));

        if( project != null ) return project.getChat() ;


        throw new ChatException("no chats found");

    }


    public List<User> getUsersByProjectId(Long projectId) throws ProjectException {

        Project project = projectRepository.findById(projectId).orElse(null);

        if( project != null) return project.getChat().getUsers();

        throw new ProjectException("no project found with id "+projectId);

    }

}
```