**ORGANIZATION OVERVIEW:**

The **Thakur College of Science and Commerce (TCSC)** is a college in Kandivali in Mumbai of Maharashtra, India running by Thakur Educational Trust.

Thakur College was started in 1992 to serve the needs of students passing SSC examination from the schools around Kandivali area and Thakur Vidhya Mandir which has already established itself as one of the schools in the area. It offers courses at primarily the higher secondary and under-graduate levels. The courses at the undergraduate and post-graduate level are offered in affiliation with Mumbai University, Mumbai. An ISO 9001:2008 College with A grade as assessed by the National Assessment and Accreditation Council NAAC.

Name : Thakur College of Science and Commerce

Founded : 1997

Address : Thakur College of Science and Commerce, Thakur Village Kandivali (East), Mumbai 400 001.

Contacts : 022-2846 2565 / 022-2887 0627

Email : Helpdesk@tcsc.org.in

History :

* Thakur Junior College was established in 1992, by the founding members
* It was a natural augmentation by the Thakur Educational Trust
* Thakur College had a humble beginning with only 57 students in FYJC first batch of commerce stream
* Our college has accomplished a spectacular growth over the last two decades
* The College has consistently attained outstanding results in academics at both Degree & Junior levels

**Device Specifications:**

Operating System Windows 10 Pro (Version 21H2)

Processor Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz 3.20 GHz

RAM 8.00 GB

Storage 1 TB HDD (64 MB Cache Memory) (7200 RPM Read Write Speed)

System type 64-bit operating system, x64-based processor

**INTRODUCTION**

**What is Project?**

A project is a unique, transient endeavour, undertaken to achieve planned objectives, which could be defined in terms of outputs, outcomes or benefits. A project is usually deemed to be a success if it achieves the objectives according to their acceptance criteria, within an agreed timescale and budget. Time, cost and quality are the building blocks of every project.

**Time:**  scheduling is a collection of techniques used to develop and present schedules that show when work will be performed.

**Cost:** how are necessary funds acquired and finances managed?

**Quality:** how will fitness for purpose of the deliverables and management processes be assured?

**What is Project Management?**

**Project management** is the application of processes, methods, skills, knowledge and experience to achieve specific project objectives according to the project acceptance criteria within agreed parameters. Project management has final deliverables that are constrained to a finite timescale and budget.

A key factor that distinguishes project management from just 'management' is that it has this final deliverable and a finite timespan, unlike management which is an ongoing process. Because of this a project professional needs a wide range of skills; often technical skills, and certainly people management skills complete the project task and satisfy client needs to good business awareness.

**Existing way for Project management:**

Managers handle multiple projects all the time. They juggle with people, tasks, and goals to ensure that every project is successful. But managing projects, by nature, is not an easy task. Since there are a lot of moving parts, it can easily become chaotic and disorganized.

When you don’t use a project management software, chances are that you’re wasting a lot of time on the little things and doing more work than necessary. It’s difficult to keep track of who’s working on what. Collaboration among teams is inefficient and spread across email threads and chat software. As a result, deadlines are missed and clients become unsatisfied.

It’s important to use an efficient project management system to help you stay organized at work while planning and executing projects.

**DRAW BACK OF EXISTING WAY FOR PROJECT MANAGEMENT**

**1) IT IS SLOW**

It is going to be a slow development if your client isn’t clear about the requirements. Since the changes are difficult in the sequential methodology of traditional project management. The changes can break the sequence and the next development step will be trodden upon until the previous step isn’t completed.

**2) NO CUSTOMER FOCAL POINT**

There isn’t much space for the customer’s opinion in every development process. The clients are not involved or being open to the entire development process until the product is ready. If it is not according to a client’s requirements then it’s going to be a time-wasting phenomenon. The stage of finalizing the product is too late to check the marketing accountability of the product as well.

Making the customer happy is the topmost priority of today’s world. Because it’s their happiness which is going to determine your actual Position Application Development Company in the market

**3) TIME MANAGEMENT**

The time mismanagement issue can arise in existing methodology due to a lack of mutual teamwork, unity, miscommunication and mutual development.

**4) LACK OF INTUITIVENESS**

There isn’t much space for new ideas to employ in existing Methodology. The sequential method doesn’t have room for continuous evolution. You cannot do much but just to follow the sequence like waterfall.

**5) LACK OF COORDINATION**

The subcontractors are involved as individuals, not as a singular dynamic team. There is no attempt is made to have unity, mutual teamwork, and commitment to the development. The lack of coordination delays development and can cause some serious issues.

**6) LOCAL OPTIMIZATION**

All individual subcontractors only care about their own efforts and talent. If something goes wrong, there isn’t mutual teamwork to defend and tackle the problem.

No teamwork, no dream work in existing project management.

**7) NO EASY CHANGE**

Once an application is on the testing stage, it is very likely impossible, time taking and expensive to go back and fix it according to the requirements. Sometimes you need to start again to implement any changes.

**8) NOT USER FRIENDLY FOR ALL USER'S**

The existing method of management is not an easy task to control, track, balance, handling & communication. It's not even flexible for time management, budget management & modification in projects.

**9) RISK INVOLVING AND CHALLENGING DEVELOPMENT**

Sometimes the customers and stakeholders are not sure about their requirements in the start, and web application tool cannot begin without gathering all relevant details and requirements. That is why the traditional approach does not offer much space for customer’s feedback and the customized final product. It is also a not viable model for complex, big and object-oriented project development.

**Project Management Web Application (PMWA)**

PMWA is online tool used by a wide range of industries for project planning, resource allocation and scheduling. It enables project managers as well as entire teams to control their budget, quality management and all documentation exchanged throughout a project. This online tool also serves as a platform for facilitating collaboration among project stakeholders.

**What makes a good PMWA?**

From tracking down deliverables to managing resources and from budget management to collaboration with team members, there is a lot to be considered when running and managing projects. This holds true when choosing the right PMWA tool as well. Here’s a list of the top 5 key functionality aspects of PMWA to guide you in your decision.

1. **Task lists** – being able to assign and update the status of tasks so that everyone in your team is on the same page is critical
2. **Schedules** – many tools offer calendars, Gantt charts or milestone tools that help you understand where a task fits into the project as a whole and how much time there is to complete it.
3. **File sharing** – being able to share and organise key project documents eliminates time wasted searching for files
4. **Communication** – this is critical in project management as a smooth flow of communication means quick and easy problem solving
5. **Reporting** – this is important for all team members when it comes to updating themselves on the project as a whole. However this is also a huge plus for project managers who want to ensure that the project is progressing and tasks are being carried out in a timely manner

**Key benefits of having a PMWA**

PMWA come with the necessary features you need to efficiently plan your projects, manage available resources, respond to problems, and keep all the project stakeholders involved. Regardless of the services, your business offers, having the best project management tool can make a significant impact.

**1. Improved planning and scheduling**

Project planning and project scheduling is extremely important aspect of project management irrespective of the applied methodology. With these project management solutions, the previous record of the team relevant to the current project can be accessed easily.

Also, project managers can conveniently create a consistent management plan and prioritize tasks for the project’s success. With PMWA, tasks like allocating resources, identifying dependencies, setting up deadlines, and creating project deliverables can be completed conveniently within no time.

Since a good start lays the foundation for a successful project, all effective project scheduling tools come with features that help in streamlining the project planning and project scheduling process.

**2. Better collaboration**

Project teams are sometimes comprised of members from different departments. They are required to handle their day-to-day responsibilities as well, which makes it hard for all other stakeholders to regularly attend meetings and stay updated. This communication gap can create unnecessary delays in the projects and problems that can waste a lot of precious resources.

A huge advantage of the PMWA is that it makes an effective project team collaboration extremely simple. It keeps all the communication in a unified place. Insights like project timeline and status updates can be accessed easily with a single click and important alerts can be automatically sent to the relevant parties.

“75% of project professionals believe that the usage of project collaboration tools for teams will increase.”

**3. Remote working**

The physical location or department does not bind modern projects. Workforces are now scattered and managing them can be quite problematic. With remote PMWA, managing remote projects becomes easy as these programs help managers effectively lead their teams.

**4. Effective task delegation**

Managers often find it difficult to fairly assign tasks to members which may affect the efficiency of the whole team. With a project management program, managers can easily delegate project tasks to members and find who is currently available.

This is also an easier option for team members as in task management, the only thing they need to do is login in website and start working on their tasks according to their priorities. In most cases, these programs also send out automatic reminders before the deadlines to ensure smooth and efficient operations.

**5. Easier file access and sharing**

Safely accessing and sharing important documents is quite important. Professionals have been using shared drives but using the PMWA is a more practical option. PMWA offer storage where users can make changes, leave feedback, and annotate easily. These programs also keep a change log to ensure project transparency within the team.

**6. Easier integration of new members**

Projects are dynamic in nature and sometimes, inducting new members into the project team becomes necessary. Project integration is, however, a challenging task to get new members up to speed within a short time. This issue can be easily handled with the use of any effective PMWA. They have the ability to automatically maintain a project log and visualize the entire project in a simplistic manner with charts & project progress bars.

**7. Effective risk mitigation**

Identifying potential bottlenecks and employing effective solutions are the makings of a great manager. With PMWA, managers can enhance their risk mitigation strategies and take help from all the resources at hand to deploy effective countermeasures.

This improves the overall efficiency of the team and also allows managers to allocate their available resources in an effective manner to avoid project risks.

**8. Budget management**

As discussed before, a major benefit of using PMWA is that they allow users to effectively manage every resource at hand. Over the course of any project, the chances of scope creep, the tendency of projects to go off the track, get quite high.

A PMWA can help managers prevent that as it allows them to keep track of everything with the help of a real-time project reporting tool and prevent the project cost from skyrocketing.

**9. Enhanced productivity**

Another major advantage of the PMWA is that it optimizes your decision-making process and improves productivity at work. As all of the important details can be found in a single place, you can always make informed decisions within the shortest time.

**10. Effective Team Meetings**

It’s critical that project team meetings are clear, time-efficient, & project effective. Team members cannot leave a meeting with the only thought in their minds being, “that could have easily been an email.”

With project management software, it’s possible to schedule team meetings with a clear agenda & desired outcome. Plus, managers can incorporate basic meeting functions such as meeting minutes & follow-up meetings to streamline project workflow further. When working on a project, teams must have a clear understanding of the meeting’s objectives. Furthermore, it’s critical that the meetings always serve a purpose for the project’s development & add value to those working on it.

**11. Easier File Access & Sharing**

It’s important to share & have secure access to important files. In this regard, project management software solutions are extremely useful. They provide storage for users to easily make changes, leave feedback, & annotate. Such software also maintains a log to ensure project transparency among team members with any changes made.

**12. Easier On-Boarding**

The complexity & duration of projects can make it difficult to integrate new team members. With project management software, it’s slightly less cumbersome to quickly bring new members up to speed & direct them to the appropriate resources. This software facilitates access to project logs & provides a quick overview of tasks, responsibilities, & objectives.

**13. Effective Budget Tracking**

Managing costs & staying on budget can be difficult. If one project goes over budget, it can have an impact on subsequent projects, throwing everything off-kilter.

Budget & resource management are becoming easier with project management software. It allows project managers to create a detailed resource breakdown structure & track project expenses. As a result of real-time project reporting, being able to act before a project goes off track & over budget becomes more manageable.

**14. Effective Task Delegation**

Managers may find it difficult to distribute tasks to team members in a fair manner. Inefficient team progression & even internal frustration can result from inefficient task distribution.

Managers can easily delegate tasks to members using project management software because it provides clear overviews of individual ongoing team tasks & allows them to see who is available to take on any extra work.

# **Tools & Technology:**

**Tools:**

Software: React.js, Node.js, Express, CSS, Bootstrap, HTML, SCSS

Database: MongoDB

## **What is HTML?**

HTML(Hypertext Markup Language) is the language used to develop web pages. It is not an actual programming language but a markup language. Hypertext is text that has a link embedded into it that points to a different page or website. Mainly [HTML](https://en.wikipedia.org/wiki/HTML) is used for structuring a webpage and making a foundation. Basically, Html is the backbone of web pages. All the websites you see on the web are using HTML up to some extent.

Every [web developer](https://www.techtually.com/category/web-design/) or web designer has to learn HTML, to begin with. The latest version of HTML is HTML5 which is really modern and powerful. It works awesome along with CSS3. If you are thinking of learning this language then you should know these points. There are a lot of **disadvantages and disadvantages of HTML**. They are listed as follows.

**Cascading Style Sheets**

**Cascading Style Sheets** (**CSS**) is a [style sheet language](https://en.wikipedia.org/wiki/Style_sheet_language) used for describing the [presentation](https://en.wikipedia.org/wiki/Presentation_semantics) of a document written in a [markup language](https://en.wikipedia.org/wiki/Markup_language) such as [HTML](https://en.wikipedia.org/wiki/HTML). CSS is a cornerstone technology of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), alongside HTML and [JavaScript](https://en.wikipedia.org/wiki/JavaScript).

CSS is designed to enable the separation of presentation and content, including [layout](https://en.wikipedia.org/wiki/Page_layout), [colours](https://en.wikipedia.org/wiki/Color), and [fonts](https://en.wikipedia.org/wiki/Typeface).

This separation can improve content [accessibility](https://en.wikipedia.org/wiki/Accessibility); provide more flexibility and control in the specification of presentation characteristics; enable multiple [web pages](https://en.wikipedia.org/wiki/Web_page) to share formatting by specifying the relevant CSS in a separate. css file, which reduces complexity and repetition in the structural content; and enable the. css file to be [cached](https://en.wikipedia.org/wiki/Cache_(computing)) to improve the page load speed between the pages that share the file and its formatting

## **What is SCSS?**

SCSS stands for the **Sassy Cascading Style Sheets**. The more advanced variant of **CSS** is **SCSS**. It was created by **Chris Eppstein** and **Natalie Weizenbaum** and designed by **Hampton Catlin**. It is also referred to as Sassy CSS due to its advanced features. It is a pre-processor language that is compiled or interrupted into the CSS. It has a file extension of **.scss**.

We may add several extra features to CSS using SCSS, including **variables, nesting**, and many more. All of these extra features may make writing SCSS much simpler and quicker than writing the standard CSS. SCSS may use the CSS code and function. SCSS is entirely compliant with the CSS syntax, though also supporting the full power of SASS.

### **Advantages of SCSS**

Various advantages of SCSS are as follows:

1. It helps users to write clean, quick, and less CSS code in a program structure.
2. There are fewer codes in it so that we can write CSS faster.
3. SCSS offers nested so that we can use the nested syntax and useful functions, including color manipulation, math functions, and many other functions.
4. It consists of variables that help reuse the values as many times as in the CSS.
5. All versions of CSS are compatible with it. So, we can use any CSS library available.
6. SASS is versatile with feedback, but any good developer would prefer the inline documentation available in SCSS.

**ReactJS**

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library responsible only for the view layer of the application. It was created by Jordan Walke, who was a software engineer at Facebook. It was initially developed and maintained by Facebook and was later used in its products like WhatsApp & Instagram. Facebook developed ReactJS in 2011 in its newsfeed section, but it was released to the public in the month of May 2013.

Today, most of the websites are built using MVC (model view controller) architecture. In MVC architecture, React is the 'V' which stands for view, whereas the architecture is provided by the Redux or Flux.

A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The virtual DOM works fast as it only changes individual DOM elements instead of reloading complete DOM every time.

To create React app, we write React components that correspond to various elements. We organize these components inside higher level components which define the application structure. For example, we take a form that consists of many elements like input fields, labels, or buttons. We can write each element of the form as React components, and then we combine it into a higher-level component, i.e., the form component itself. The form components would specify the structure of the form along with elements inside of it.

**Why learn ReactJS?**

Today, many JavaScript frameworks are available in the market(like angular, node), but still, React came into the market and gained popularity amongst them. The previous frameworks follow the traditional data flow structure, which uses the DOM (Document Object Model). DOM is an object which is created by the browser each time a web page is loaded. It dynamically adds or removes the data at the back end and when any modifications were done, then each time a new DOM is created for the same page. This repeated creation of DOM makes unnecessary memory wastage and reduces the performance of the application.

Therefore, a new technology ReactJS framework invented which remove this drawback. ReactJS allows you to divide your entire application into various components. ReactJS still used the same traditional data flow, but it is not directly operating on the browser's Document Object Model (DOM) immediately; instead, it operates on a virtual DOM. It means rather than manipulating the document in a browser after changes to our data, it resolves changes on a DOM built and run entirely in memory.

After the virtual DOM has been updated, React determines what changes made to the actual browser's DOM. The React Virtual DOM exists entirely in memory and is a representation of the web browser's DOM. Due to this, when we write a React component, we did not write directly to the DOM; instead, we are writing virtual components that react will turn into the DOM.

**Node.js**

## **What is Node.js?**

Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009 and its latest version is v0.10.36. The definition of Node.js as supplied by its [official documentation](https://nodejs.org/) is as follows –

Node.js is a platform built on [Chrome's JavaScript runtime](https://code.google.com/p/v8/) for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

Node.js = Runtime Environment + JavaScript Library

## **Features of Node.js**

Following are some of the important features that make Node.js the first choice of software architects.

* **Asynchronous and Event Driven** − All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.
* **Very Fast** − Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.
* **Single Threaded but Highly Scalable** − Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.
* **No Buffering** − Node.js applications never buffer any data. These applications simply output the data in chunks.
* **License** − Node.js is released under the [MIT license](https://raw.githubusercontent.com/joyent/node/v0.12.0/LICENSE).

## **Who Uses Node.js?**

Following is the link on github wiki containing an exhaustive list of projects, application and companies which are using Node.js. This list includes eBay, General Electric, GoDaddy, Microsoft, PayPal, Uber, Wikipins, Yahoo!, and Yammer to name a few.

* Projects,
* Applications
* Companies Using Node

## **Concepts**

The following diagram depicts some important parts of Node.js which we will discuss in detail in the subsequent chapters.



Figure 1: Import Concept of Node.js

## **Where to Use Node.js?**

Following are the areas where Node.js is proving itself as a perfect technology partner.

* I/O bound Applications
* Data Streaming Applications
* Data Intensive Real-time Applications (DIRT)
* JSON APIs based Applications
* Single Page Applications

# **What is Bootstrap**

* Bootstrap is the most popular HTML, CSS and JavaScript framework for developing a responsive and mobile friendly website.
* It is absolutely free to download and use.
* It is a front-end framework used for easier and faster web development.
* It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many others.
* It can also use JavaScript plug-ins.
* It facilitates you to create responsive designs.

### **History of Bootstrap**

Bootstrap was developed by Mark Otto and Jacob Thornton at Twitter.It was released as an open source product in August 2011 on GitHub. In June 2014 Bootstrap was the No.1 project on GitHub.

### **Why use Bootstrap**

Following are the main advantage of Bootstrap:

* It is very easy to use. Anybody having basic knowledge of HTML and CSS can use Bootstrap.
* It facilitates users to develop a responsive website.
* It is compatible on most of browsers like Chrome, Firefox, Internet Explorer, Safari and Opera etc.

### **What is a responsive website**

A website is called responsive website which can automatically adjust itself to look good on all devices, from smart phones to desktops etc.

### **What Bootstrap package contains**

**Scaffolding:** Bootstrap provides a basic structure with Grid System, link styles, and background.

**CSS:** Bootstrap comes with the feature of global CSS settings, fundamental HTML elements style and an advanced grid system.

**Components:** Bootstrap contains a lot of reusable components built to provide iconography, dropdowns, navigation, alerts, pop-overs, and much more.

**JavaScript Plugins:** Bootstrap also contains a lot of custom jQuery plugins. You can easily include them all, or one by one.

**Customize:** Bootstrap components are customizable and you can customize Bootstrap's components, LESS variables, and jQuery plugins to get your own style.

## **What is Bootstrap 4?**

Bootstrap is the newest and latest version of Bootstrap. It is the most popular HTML, CSS, JavaScript framework for developing responsive, mobile first websites.

**MongoDB**

MongoDB tutorial provides basic and advanced concepts of SQL. Our MongoDB tutorial is designed for beginners and professionals.

MongoDB is a No SQL database. It is an open-source, cross-platform, document-oriented database written in C++.

Our MongoDB tutorial includes all topics of MongoDB database such as insert documents, update documents, delete documents, query documents, projection, sort() and limit() methods, create a collection, drop collection, etc. There are also given MongoDB interview questions to help you better understand the MongoDB database.

## **What is MongoDB**

[MongoDB](https://www.javatpoint.com/mongodb-tutorial) is an open-source document database that provides high performance, high availability, and automatic scaling.

In simple words, you can say that - Mongo DB is a document-oriented database. It is an open source product, developed and supported by a company named 10gen.

MongoDB is available under General Public license for free, and it is also available under Commercial license from the manufacturer.

The manufacturing company 10gen has defined MongoDB as:

"MongoDB is a scalable, open source, high performance, document-oriented database." - 10gen

MongoDB was designed to work with commodity servers. Now it is used by the company of all sizes, across all industry.

**MongoDB works on concept of collection and document.**

## **Database**

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

## **Collection**

Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

## **Document**

A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

The following table shows the relationship of RDBMS terminology with MongoDB.

|  |  |
| --- | --- |
| **RDBMS** | **MongoDB** |
| Database | Database |
| Table | Collection |
| Tuple/Row | Document |
| column | Field |
| Table Join | Embedded Documents |
| Primary Key | Primary Key (Default key \_id provided by MongoDB itself) |
| **Database Server and Client** | |
| mysqld/Oracle | mongod |
| mysql/sqlplus | mongo |

# **Express.js**

Express.js tutorial provides basic and advanced concepts of Express.js. Our Express.js tutorial is designed for beginners and professionals both.

Express.js is a web framework for Node.js. It is a fast, robust and asynchronous in nature.

Our Express.js tutorial includes all topics of Express.js such as Express.js installation on windows and linux, request object, response object, get method, post method, cookie management, scaffolding, file upload, template etc.

## **What is Express.js**

Express is a fast, assertive, essential and moderate web framework of Node.js. You can assume express as a layer built on the top of the Node.js that helps manage a server and routes. It provides a robust set of features to develop web and mobile applications.

From a web server development perspective Node has a number of benefits:

* Great performance! Node was designed to optimize throughput and scalability in web applications and is a good solution for many common web-development problems (e.g. real-time web applications).
* Code is written in "plain old JavaScript", which means that less time is spent dealing with "context shift" between languages when you're writing both client-side and server-side code.
* JavaScript is a relatively new programming language and benefits from improvements in language design when compared to other traditional web-server languages (e.g. Python, PHP, etc.) Many other new and popular languages compile/convert into JavaScript so you can also use TypeScript, CoffeeScript, ClojureScript, Scala, LiveScript, etc.
* The node package manager (npm) provides access to hundreds of thousands of reusable packages. It also has best-in-class dependency resolution and can also be used to automate most of the build toolchain.
* Node.js is portable. It is available on Microsoft Windows, macOS, Linux, Solaris, FreeBSD, OpenBSD, WebOS, and NonStop OS. Furthermore, it is well-supported by many web hosting providers, that often provide specific infrastructure and documentation for hosting Node sites.
* It has a very active third party ecosystem and developer community, with lots of people who are willing to help.

## **Why Express?**

Unlike its competitors like Rails and Django, which have an opinionated way of building applications, Express has no "best way" to do something. It is very flexible and pluggable.

### **Pug**

Pug (earlier known as Jade) is a terse language for writing HTML templates. It −

* Produces HTML
* Supports dynamic code
* Supports reusability (DRY)

It is one of the most popular template language used with Express.

**GANTT**

**CHART**

**Gantt Chart**

### **OBJECTIVE:**

The main objective of a Gantt chart is to assess how long a project should task and to established the order in which tasks need to be carried out by the ending the project.

### **DESCRIPTION:**

A Gantt chart is a horizontal bar chart developed as a production control tool frequently used in project management a Gantt chart provides illustration of a schedule that helps to a plan coordinate, and track specific task in a project.

Gantt charts are useful tools for planning and scheduling projects. They allow you to assess how long a project should take, determine the resources needed , and layout the order in which tasks need to be carried out . They are useful in managing the dependencies between tasks.

When a project is under way, Gantt charts are useful for monitoring its progress. You can immediately see what should have been achieved at a point in time, and can therefore take remedial action to bring the project back on course. This can be essential for the successful and profitable implementation of the project.

### **BENEFITS:**

·        Gives clear illustration of project status.

·        Helps manage the temporal dependencies between tasks.

Figure 2: Gantt Chart for Project Status

**ACTIVITY**

**CHART**

**THAKUR COLLEGE OF SCIENCE AND COMMERCE**

**Department of Computer Science**

2022-2023

**Student's Name: Jagdish Ashok Aneshwar**

**Project Name: Project Management Application**

**College Name: Thakur College of Science and Commerce**

|  |  |  |  |
| --- | --- | --- | --- |
| **PHASES** | **EXPECTED DATE OF SUBMISSION** | **ACTUAL DATE OF SUBMISSION** | **SIGNATURE** |
| **PRELIMINARY**  **INVESTIGATION** | **10th July, 2022** | **08th July, 2022** |  |
| **SYSTEM**  **ANALYSIS** | **20th July, 2022** | **20th July, 2022** |  |
| **SYSTEM**  **DESIGNING** | **27th July, 2022** | **25th July, 2022** |  |
| **SYSTEM**  **CODING** | **07th August, 2022** | **06rd August, 2022** |  |
| **SYSTEM**  **IMPLEMENTATION** | **27th September, 2022** | **27th September, 2022** |  |
| **PROJECT**  **SUBMISSION** | **19th October, 2022** | **19th October, 2022** |  |

**EVENT**

**TABLE**

**EVENT TABLE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Trigger** | **Source** | **Activity** | **Response** | **Destination** |
| Create account | Insert personal  Details like name, email, password, etc. | Manager | Creating account | Login Page | Server |
| Request | Insert operation on project, tasks, employee, address | Manager | Insert operation on project, tasks, employee, address | Your request has been sent to server for process. | Server |
| Login | Insert email id and password | Manager | Logging in | Home Page | Server |
| Overview | View status of project and tasks. | Manager | Check the status whether the project is progressing within budget and time frame. | Make necessary changes in the database. | Server |

**Activity Diagram**

This is the Activity UML diagram of Project Management System which shows the flows between the activity of Project, Task, Salary, Employee, Bug. The main activity involved in this UML Activity Diagram of Project Management System are as follows:

* Project Activity
* Task Activity
* Salary Activity
* Employee Activity
* Bug Activity

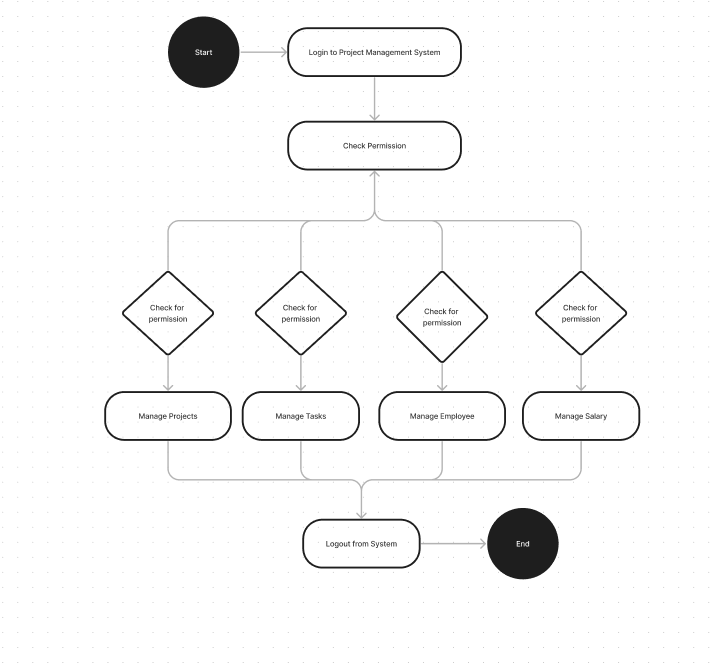


Figure 3: Activity Diagram for Project Management System

**Class Diagram:**

**Object oriented development**

1. It is a new way of thinking about software based on abstractions that exist in the real world
2. In this context, development refers to analysis, design & implementation of the software
3. Object oriented development is a conceptual process independent of a programming language.
4. It can serve as medium for specification, analysis, documentation as well as for programming.

**Object Oriented Methodology**

1. We present methodology for object oriented development and graphical notation for representing object oriented concept.
2. The methodology has the following feature:
3. Starting from the problem statement, the analysist builds the model of real world situation showing it’s properties and operation.
4. The object in the model should be application domain model and should not include computer implementation concept and data structure

**Object Modelling**

1. Object model describe real world object classes and their relationship to one another.
2. Information for the object model comes from the problem statement, expert knowledge of application domain and general knowledge of real world.
3. Identifying the classes and association first as the affect the overall structure and approach to the problem.
4. We add attributes to further describe the basis network of classes and association.

**Collaboration Diagram**

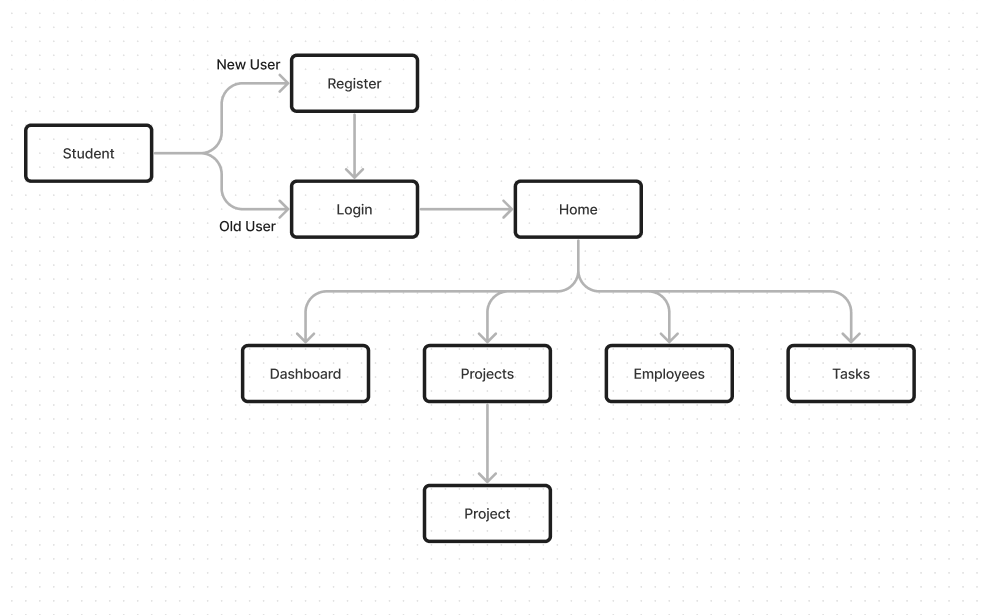


Figure 4: Collaboration Diagram for Project Management System

Collaboration diagrams (**known as Communication Diagram in UML 2.x**) are used to show how objects interact to perform the behaviour of a particular use case, or a part of a use case. Along with sequence diagrams, collaboration are used by designers to define and clarify the roles of the objects that perform a particular flow of events of a use case.  They are the primary source of information used to determining class responsibilities and interfaces. Figure 4 describes that a user first creates account then logs in Home Page and views the project status and budget.

**Data Flow Diagram**

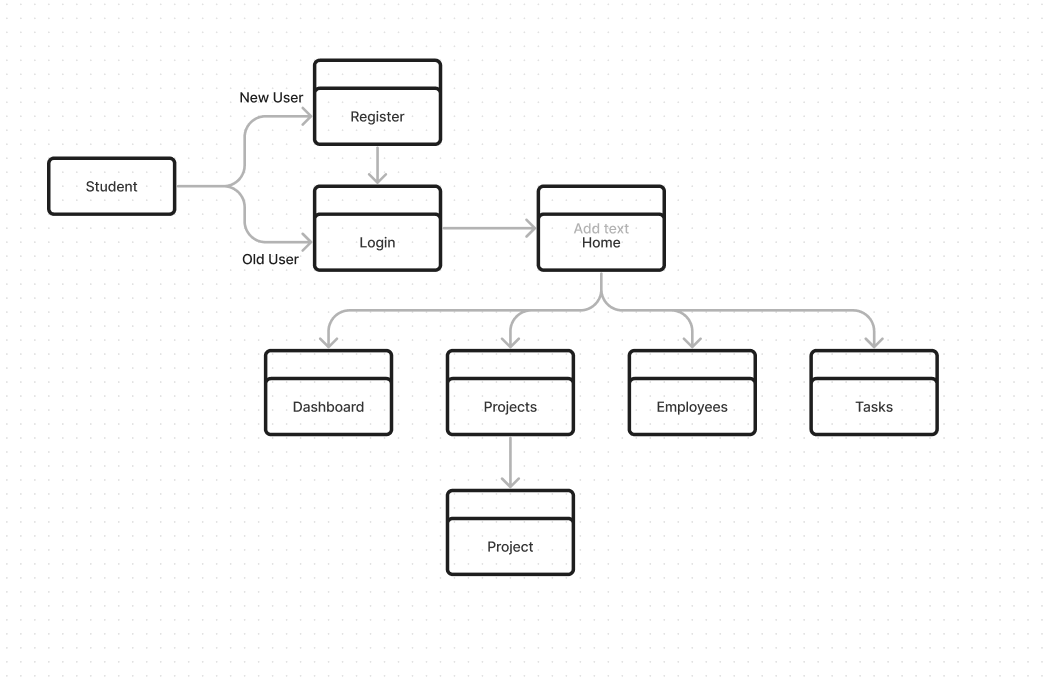
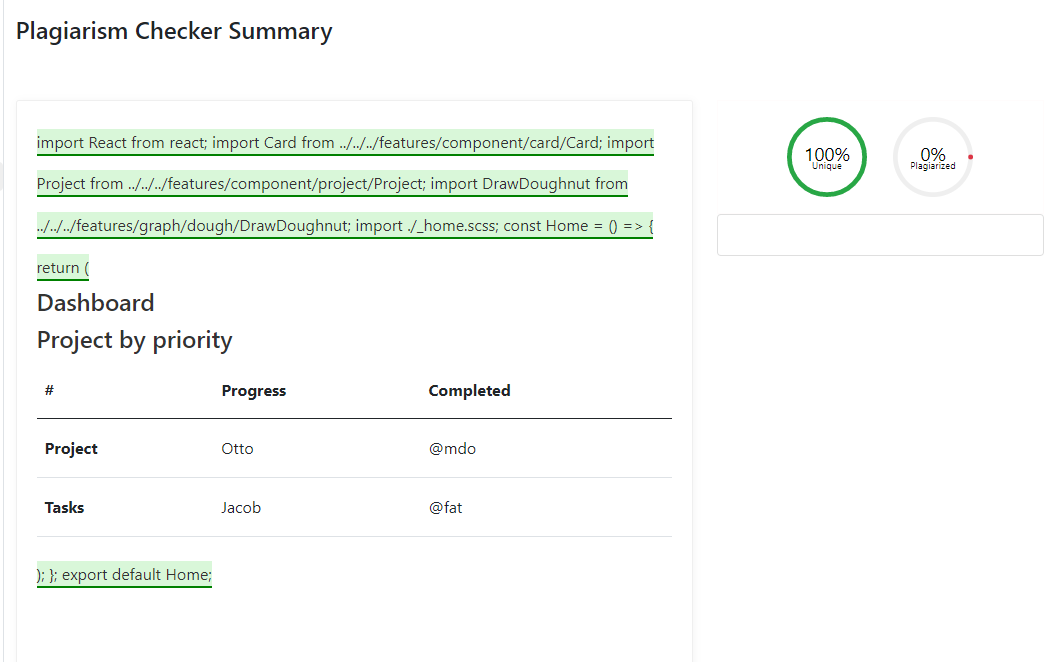


Figure 5: DFD for Project Management System

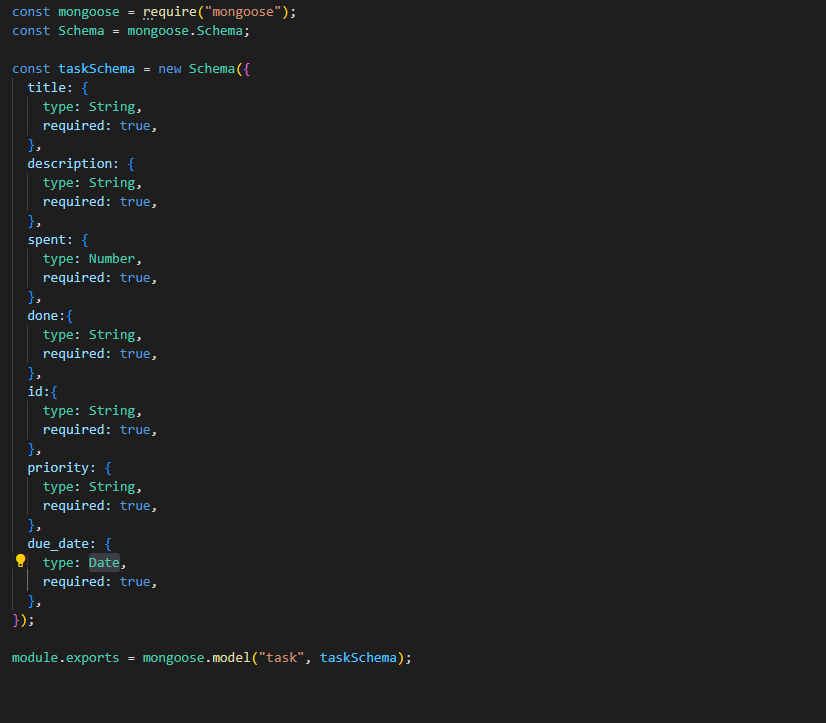
Also known as DFD, Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.Figure 5 describes that a user first creates account then logs in Home Page and views the project status and budget

**Plagiarism Report**

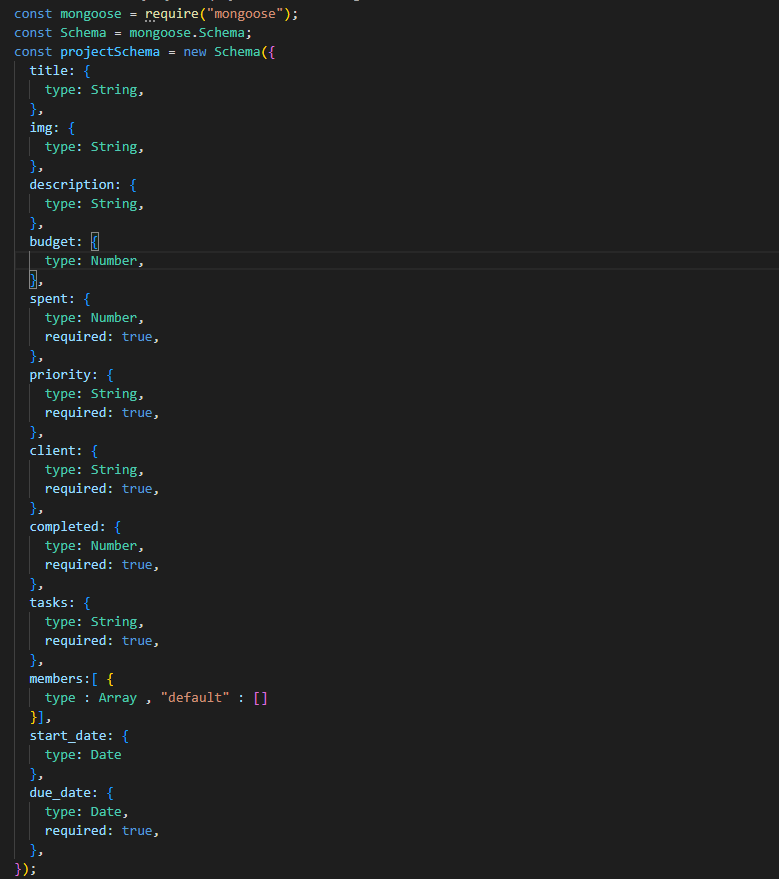


**Schema Design**

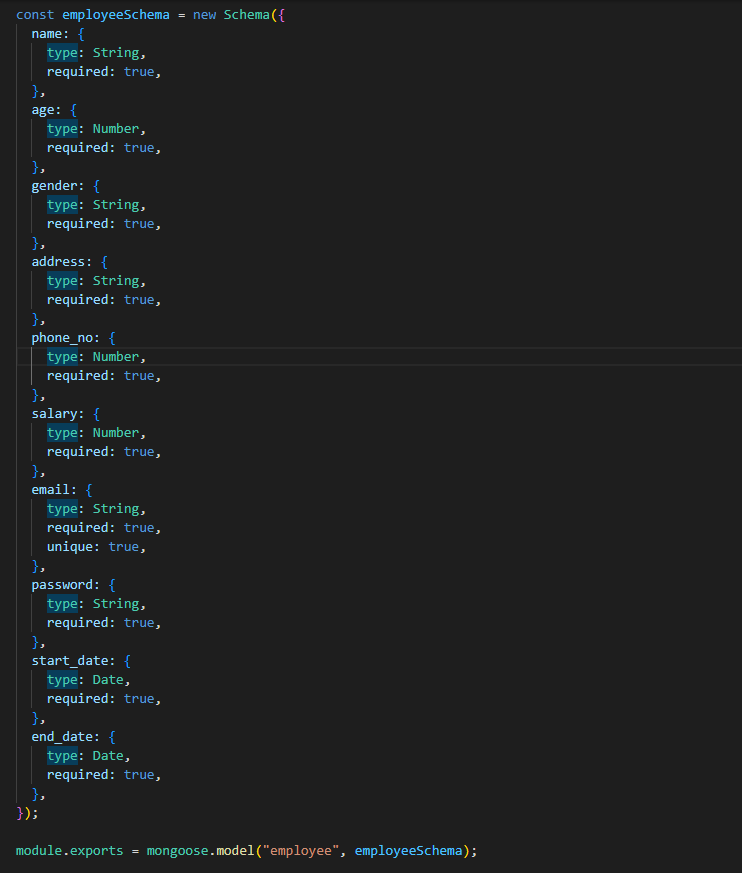
**Task**

****

**Project**

****

**Employee**

****

**Admin**

****

**Sample Code:**

**Project.js** const fetchuser = require("../middleware/fetchuser");

const Project = require("../models/Project");

const express = require("express");

const router = express.Router();

// ---------------------------------- Route 1: get all projects using GET: "api/project/allProjectDetails"

router.get("/allProjectDetails", fetchuser, async (req, res) => {

  try {

    const project = await Project.find({ user: req.user.id });

    res.json(project);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ------------------------------------------ Route 2: update an existin projectusing PUT: "api/projects/update" -login required

router.put("/updateProject/:id", fetchuser, async (req, res) => {

  const {

    title,

    description,

    budget,

    spent,

    due\_date,

    tasks,

  } = req.body;

  // create new project object

  const newProject = {};

  if (title) {

    newProject.title = title;

  }

  if (description) {

    newProject.description = description;

  }

  if (budget) {

    newProject.budget = budget;

  }

  if (spent) {

    newProject.spent = spent;

  }

  if (due\_date) {

    newProject.due\_date = due\_date;

  }

  if (client) {

    newProject.client = client;

  }

  // find the project to be updated and update it

  var project = await Project.findById(req.params.id);

  if (!project) {

    return res.status(404).send("not found");

  }

  if (project.user.toString() !== req.user.id) {

    return res.status(401).send("Unauthorized");

  }

  try {

    project = await Projects.findByIdAndUpdate(

      req.params.id,

      { $set: newProject },

      { new: true }

    );

    res.json(project);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ----------------------------------- Route 3: add new project using POSt: "api/project/addProject"

router.post("/addProject", async (req, res) => {

  const {

    title,

    description,

    budget,

    spent,

    start\_date,

    due\_date,

    priority,

    client,

    completed,

    tasks,

    members,

    img

  } = req.body;

  try {

    const project = new Project({

      title,

      description,

      budget,

      spent,

      start\_date,

      due\_date,

      priority,

      client,

      completed,

      tasks,

      members,

      img

    });

    const saveClothe = await project.save();

    res.send(saveClothe);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ------------------------------------ Route 4: delete an existing project using DELETE: "api/project/deleteProject" -login required

router.delete("/removeProject/:id", fetchuser, async (req, res) => {

    // find the project to be updated and update it

    var project = await Project.findById(req.params.id);

    if (!project) {

      return res.status(404).send("Not found");

    }

    if (project.user.toString() !== req.user.id) {

      return res.status(401).send("Unauthorized");

    }

    try {

      project = await Project.findByIdAndDelete(req.params.id);

      res.send(project);

    } catch (err) {

      console.error(err.message);

      res.json({ error: "internal Server Error", err: err.message });

    }

  });

module.exports = router;

**Employee.js** const fetchuser = require("../middleware/fetchuser");

const Employee = require("../models/Employee");

const express = require("express");

const router = express.Router();

// ----------------------------------- Route 1: add new employee using POSt: "api/employee/addEmployee"

router.post("/addEmployee", async (req, res) => {

  const {

    name,

    age,

    gender,

    address,

    start\_date,

    end\_date,

    phone\_no,

    salary,

    email,

    password: secPassword,

  } = req.body;

  try {

    const employee = new Employee({

      name,

      age,

      gender,

      address,

      start\_date,

      end\_date,

      phone\_no,

      salary,

      email,

      password: secPassword,

    });

    const saveClothe = await employee.save();

    res.send(saveClothe);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ------------------------------------------ Route 2: update an existin employeeusing PUT: "api/employees/update" -login required

router.put("/updateEmployee/:id", fetchuser, async (req, res) => {

  const {

    name,

    age,

    gender,

    address,

    start\_date,

    end\_date,

    phone\_no,

    salary,

    email } = req.body;

  // create new employee object

  const newEmployee = {};

  if (name) {

    newEmployee.name = name;

  }

  if (age) {

    newEmployee.age = age;

  }

  if (gender) {

    newEmployee.gender = gender;

  }

  if (address) {

    newEmployee.address = address;

  }

  if (start\_date) {

    newEmployee.start\_date = start\_date;

  }

  if (end\_date) {

    newEmployee.end\_date = end\_date;

  }

  if (phone\_no) {

    newEmployee.phone\_no = phone\_no;

  }

  if (salary) {

    newEmployee.salary = salary;

  }

  if (email) {

    newEmployee.email = email;

  }

  // find the employee to be updated and update it

  var employee = await Employees.findById(req.params.id);

  if (!employee) {

    return res.status(404).send("not found");

  }

  if (employee.user.toString() !== req.user.id) {

    return res.status(401).send("Unauthorized");

  }

  try {

    employee = await Employees.findByIdAndUpdate(

      req.params.id,

      { $set: newEmployee },

      { new: true }

    );

    res.json(employee);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ---------------------------------- Route 3: get all employees using GET: "api/employee/allEmployeeDetails"

router.get("/allEmployeeDetails/:id", fetchuser, async (req, res) => {

  try {

    const employee = await Employee.findById(req.params.id);

    res.json(employee);

  } catch (err) {

    console.error(err.message);

    res.json({ error: "internal Server Error", err: err.message });

  }

});

// ------------------------------------ Route 4: delete an existing employee using DELETE: "api/employee/deleteEmployee" -login required

router.delete("/removeEmployee/:id", fetchuser, async (req, res) => {

    // find the employee to be updated and update it

    var employee = await Employee.findById(req.params.id);

    if (!employee) {

      return res.status(404).send("Not found");

    }

    if (employee.user.toString() !== req.user.id) {

      return res.status(401).send("Unauthorized");

    }

    try {

      employee = await Employee.findByIdAndDelete(req.params.id);

      res.send(employee);

    } catch (err) {

      console.error(err.message);

      res.json({ error: "internal Server Error", err: err.message });

    }

  });

module.exports = router;

**Dashboard** import React from "react";

import "./\_dashboard.scss";

import Profile from "../../features/component/circular-profile/Profile";

import Home from "../tabs/home/Home";

import Projects from "../tabs/projects/Projects";

import Employee from "../tabs/employee/Employee";

import Project from "../tabs/project/Project";

const Dashboard = () => {

  return (

    <section className="dashboard ">

      <div className="vertical-nav d-flex flex-column align-items-start">

      <div class="parent-nav d-flex align-items-start">

      <div class="nav flex-column nav-pills me-3" id="v-pills-tab" role="tablist" aria-orientation="vertical">

      <div className="profile">

        <Profile/>

        </div>

      <div class="navbtn nav-link" id="v-pills-home-tab" data-bs-toggle="pill" data-bs-target="#v-pills-home" type="button" role="tab" aria-controls="v-pills-home" aria-selected="true">home</div>

      <div class="navbtn nav-link" id="v-pills-profile-tab" data-bs-toggle="pill" data-bs-target="#v-pills-profile" type="button" role="tab" aria-controls="v-pills-profile" aria-selected="false">projects</div>

      <div class="navbtn nav-link" id="v-pills-messages-tab" data-bs-toggle="pill" data-bs-target="#v-pills-messages" type="button" role="tab" aria-controls="v-pills-messages" aria-selected="false">employee</div>

      </div>

      <div class="tab-content" id="v-pills-tabContent">

        <div class="tab-pane fade show active" id="v-pills-home" role="tabpanel" aria-labelledby="v-pills-home-tab">

          <Home/>

        </div>

        <div class="tab-pane fade" id="v-pills-profile" role="tabpanel" aria-labelledby="v-pills-profile-tab">

          <Projects/>

        </div>

        <div class="tab-pane fade" id="v-pills-messages" role="tabpanel" aria-labelledby="v-pills-messages-tab">

          <Employee/>

        </div>

        </div>

    </div>

      </div>

    </section>

  );

};

export default Dashboard;

**Home.js**

import React from "react";

import Card from "../../../features/component/card/Card";

import Project from "../../../features/component/project/Project";

import DrawDoughnut from "../../../features/graph/dough/DrawDoughnut";

import "./\_home.scss";

const Home = () => {

  const data1 = ["12", "8","100"]

  const data2 = ["12", "8","10"]

  return (

    <div className="home">

      <h4 className="dashboard-title">Dashboard</h4>

      <div className="d-flex flex-row task-section  justify-content-around align-items-center pt-3">

        <Card

          img={require(`../../../res/image/ongoing.png`)}

          header="Ongoing Projects"

          value="7"

        />

        <Card

          img={require(`../../../res/image/under.png`)}

          header="Ongoing Projects"

          value="7"

        />

        <Card

          img={require(`../../../res/image/high\_risk.png`)}

          header="Ongoing Projects"

          value="7"

        />

      </div>

      <div className="project-progress d-flex flex-row justify-content-around">

      <div className="doughnut-wrap">

          <DrawDoughnut

            title="Project Progress"

            position="right"

            align="center"

            label1="Overdue"

            label2="Ongoing Projects"

            label3="Upcoming deadline"

            info1={data1[0]}

            info2={data1[1]}

            info3={data1[2]}

            titpos="top"

            titalgn="center"

          />

        </div>

        <div className="doughnut-wrap">

          <DrawDoughnut

            title="Task Progress"

            position="right"

            align="center"

            label1="Overdue"

            label2="Ongoing tasks"

            label3="Upcoming deadline"

            info1={data2[0]}

            info2={data2[1]}

            info3={data2[2]}

            titpos="top"

            titalgn="center"

          />

        </div>

        </div>

        <div className="project-lists d-flex flex-column mt-3">

        <h4 className="project-by-priority-title">Project by priority</h4>

          <Project

            title="vesa"

            progress="80"

            total\_tasks="10"

            completed\_task="8"

            priority="High"

            due="10-12-2023"

            risks="1"

            budget="30k"

            spent="12k"

          />

          <Project

            title="tera"

            progress="65"

            total\_tasks="10"

            completed\_task="3"

            priority="High"

            due="13-12-2023"

            risks="10"

            budget="50k"

            spent="8k"

          />

          <Project

            title="ai solution"

            progress="30"

            total\_tasks="10"

            completed\_task="4"

            priority="Medium"

            due="18-12-2023"

            risks="10"

            budget="30k"

            spent="12k"

          />

          <Project

            title="vesa"

            progress="36"

            total\_tasks="10"

            completed\_task="3"

            priority="High"

            due="10-12-2023"

            risks="10"

            budget="30k"

            spent="12k"

          />

      </div>

    </div>

  );

};

export default Home;

**Project.js**

import React, {useState, useEffect, useRef, useContext} from 'react'

import projectContext from "../../../context/project/projectContext"

import Select from 'react-select'

import DrawDoughnut from "../../../features/graph/dough/DrawDoughnut"

import Profile from "../../../features/component/circular-profile/Profile"

import DrawLine from '../../../features/graph/line/DrawLine';

import Task from '../../../features/component/task-card/Task';

import { useLocation } from "react-router-dom";

import "./\_project.scss";

const Project = () => {

  const location = useLocation();

    const {id,

      title,

      description,

      budget,

      spent,

      start\_date,

      due\_date} = location.state;

      const data = [ "12","30", "2"]

      const context = useContext(projectContext);

      const { projects, getProject, createTask } = context;

      useEffect(() => {

        getProject()

      });

      const ref = useRef(null);

      const refClose = useRef(null);

      const [task, settask] = useState({

        title:"",

        id:id,

        description:"",

        spent:null,

        due\_date:"",

        priority:"",

        done:"false"

      })

      const onChange = (e) => {

        settask({ ...task, [e.target.name]: e.target.value });

        };

        const onClickCreateTask = (e) => {

          ref.current.click();

      createTask(task.title, task.description,task.id, task.priority, task.due\_date, task.spent, task.done);

    };

  return (

    <div className='project-main' key={id}>

    <h4 className="dashboard-title">Dashboard</h4>

    <div className='info d-flex flex-row justify-content-around align-content-center p-3 border-bottom border-1' >

      <div className="project-profile">

      <Profile />

      </div>

      <div className='title-description'>

      <h3 className='title'>{title}</h3>

      <div className='description'>{description}</div>

      </div>

    </div>

    <h4 className='summary-title title'>Summary</h4>

    <div className='summary border-bottom border-1 mb-3'>

      <div className='summary-wrap d-flex flex-row  justify-content-between'>

        <div className='budget-info'>

        <div className='budget'>Budget: {budget}</div>

        <div className='spent'>Spent: {spent}</div>

        </div>

        <div className='date-info'>

        <div className='start-date'>Start Date: {start\_date}</div>

        <div className='due-date'>Due Date: {due\_date}</div>

        </div>

      </div>

    </div>

    <div className='visualisation d-flex flex-row'>

      <div className='project-performance'>

      <DrawDoughnut

            title="Project Progress"

            position="right"

            align="center"

            label1="Overdue"

            label2="Ongoing Projects"

            label3="Upcoming deadline"

            info1={data[0]}

            info2={data[1]}

            info3={data[2]}

            titpos="top"

            titalgn="start"

          />

    </div>

    <div className='project-line-chart'>

    <DrawLine/>

    </div>

    </div>

    <div className='tasks-overview'>

      <h4 className='task-overview-title'>Tasks Overview</h4>

      <div className='tasks-wrapper d-flex flex-row justify-content-around'>

        <div className='todo-list'>

          <h4 className='todo-list-title'>To Do</h4>

          <div className='todo-wrapper'>

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

          </div>

        </div>

        <div className='todo-list'>

          <h4 className='todo-list-title'>To Do</h4>

          <div className='todo-wrapper'>

          <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

          </div>

        </div>

        <div className='todo-list'>

          <h4 className='todo-list-title'>To Do</h4>

          <div className='todo-wrapper'>

          <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

            <Task title="Black Book" desc="Tomorrow i want to subnit bb" />

          </div>

        </div>

      </div>

    </div>

    <button type="button" className="btn btn-success btn-lg w-50" ref={ref} data-bs-toggle="modal" data-bs-target="#update">Update</button>

    <button type="button" class="btn btn-danger btn-lg w-50">Delete</button>

  <div class="task-modal modal fade" tabindex="-1" id="update" aria-labelledby="exampleModalLabel" aria-hidden="true">

  <div class="modal-dialog">

    <div class="modal-content">

      <div class="modal-header border-0">

        <h5 class="modal-title">Modal title</h5>

        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

      </div>

      <div class="modal-body">

      <form className="add-task">

          <div className="mb-3">

            <label htmlfor="exampleInputEmail1" className="form-label">

              Title

            </label>

            <input

              type="text"

              className="form-control"

              id="exampleInputEmail1"

              aria-describedby="emailHelp"

              name="title"

              onChange={onChange}

            />

          </div>

          <div className="mb-3">

            <label htmlFor="exampleInputPassword1" className="form-label">

              Description

            </label>

            <input

              type="text"

              className="form-control"

              name="description"

              onChange={onChange}

            />

          </div>

          <div className="mb-3">

            <label htmlFor="exampleInputPassword1" className="form-label">

              Cost Spent

            </label>

            <input

              type="number"

              className="form-control"

              name="spent"

              onChange={onChange}

            />

          </div>

          <div className="mb-3">

            <label htmlFor="exampleInputPassword1" className="form-label">

              Due Date

            </label>

            <input

              type="date"

              className="form-control"

              name="due\_date"

              onChange={onChange}

            />

          </div>

          <div className="mb-3">

            <label htmlFor="exampleInputPassword1" className="form-label">

              Priority

            </label>

            <input

              type="text"

              className="form-control"

              name="priority"

              onChange={onChange}

            />

          </div>

        </form>

      </div>

      <div class="modal-footer border-0">

        <button type="button" class="btn btn-secondary" data-bs-dismiss="modal"  ref={refClose}>Close</button>

        <button type="button" class="btn btn-primary"  onClick={onClickCreateTask} >Save changes</button>

      </div>

    </div>

  </div>

</div>

  </div>

  )

}

export default Project

**Projects.js** import React, {useState, useEffect, useRef, useContext} from 'react'

import ProjectOverview from '../../../features/component/project-overview-card/ProjectOverview';

import projectContext from "../../../context/project/projectContext"

import { useNavigate, Link } from "react-router-dom";

import Select from 'react-select'

import "./\_projects.scss";

const Projects = ({img}) => {

  const navigate = useNavigate();

  const context = useContext(projectContext);

  const { projects, createProject, getProject, toComponentB } = context;

  useEffect(() => {

    getProject()

  });

  const ref = useRef(null);

  const refClose = useRef(null);

  const [project, setproject] = useState({

      title: "",

      description:"",

      budget: "",

      client: "",

      members:"",

      priority:"",

      due\_date:""

  })

    const data = [

    {

      value: 1,

      label: "cerulean"

    },

    {

      value: 2,

      label: "fuchsia rose"

    },

    {

      value: 3,

      label: "true red"

    },

    {

      value: 4,

      label: "aqua sky"

    },

    {

      value: 5,

      label: "tigerlily"

    },

    {

      value: 6,

      label: "blue turquoise"

    }

  ];

  const [selectedValue, setSelectedValue] = useState([]);

  // handle onChange event of the dropdown

  const handleChange = (e) => {

    setSelectedValue(Array.isArray(e) ? e.map(x => x.value) : []);

  }

  const onChange = (e) => {

      setproject({ ...project, [e.target.name]: e.target.value });

    };

    const onClickCreateProject = (e) => {

      ref.current.click();

      createProject(project.title, project.description, project.budget, project.client, {members:selectedValue}, project.priority, project.due\_date);

    };

  return (

    <div className='projects'>

    <div class="task-modal modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

    <div class="modal-dialog">

      <div class="modal-content">

        <div class="modal-header border-0">

          <h5 class="modal-title">Add Project</h5>

          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

        </div>

        <div class="modal-body">

        <form className="add-task">

            <div className="mb-3">

              <label htmlfor="exampleInpuTitle" className="form-label">

                Title

              </label>

              <input

                type="text"

                className="form-control"

                name="title"

                onChange={onChange}

              />

            </div>

            <div className="mb-3">

              <label htmlFor="exampleInputDescription" className="form-label">

                Description

              </label>

              <input

                type="text"

                className="form-control"

                name="description"

                onChange={onChange}

              />

            </div>

            <div className="mb-3">

              <label htmlFor="exampleInputBudget" className="form-label">

                Budget

              </label>

              <input

                type="number"

                className="form-control"

                name="budget"

                onChange={onChange}

              />

            </div>

            <div className="mb-3">

              <label htmlFor="exampleInputClient" className="form-label">

                Client

              </label>

              <input

                type="text"

                className="form-control"

                name="client"

                onChange={onChange}

              />

            </div>

            <div className="mb-3">

            <label htmlFor="exampleInputAssignTo" className="form-label">

              Assign to

            </label>

            <Select

        className="dropdown"

        placeholder="Select Option"

        value={data.filter(obj => selectedValue.includes(obj.value))} // set selected values

        options={data} // set list of the data

        onChange={handleChange} // assign onChange function

        isMulti

        isClearable

      />

          </div>

            <div className="mb-3">

              <label htmlFor="exampleInputDueDate" className="form-label">

                Due Date

              </label>

              <input

                type="date"

                className="form-control"

                name="due\_date"

                onChange={onChange}

              />

            </div>

            <div className="mb-3">

              <label htmlFor="exampleInputPriority" className="form-label">

                Priority

              </label>

              <input

                type="text"

                className="form-control"

                name="priority"

                onChange={onChange}

              />

            </div>

          </form>

        </div>

        <div class="modal-footer border-0">

          <button type="button" class="btn btn-secondary" ref={refClose} data-bs-dismiss="modal">Close</button>

          <button type="button" class="btn btn-primary" onClick={onClickCreateProject}>Create Project</button>

        </div>

      </div>

    </div>

  </div>

   {/\* main-part \*/}

      <h4>Projects</h4>

     <div class="all-project-lists row row-cols-md-3">

        { projects && projects.map((project) => {

          return (

            <div

            class=" project-card-overview" key={project.\_id}

            onClick={() => {

              toComponentB(project, navigate);

            }}

            >

            <div class="project-overview">

              <img src={require("../../../res/image/img3.jpg")} class="card-img project-card-img " alt="..."/>

              <div class="card-body">

                <h5 class="card-title">{project.title}</h5>

                <p class="card-text">{project.description}</p>

              </div>

              <p className='p-3'>Priority: {project.priority}</p>

            </div>

          </div>

          );

        })

      }

      <div class=" project-card-overview">

        <div class="project-overview">

        <img src={require("../../../res/image/add.png")} alt="heelo" type="button" ref={ref} class="create-project-button" data-bs-toggle="modal" data-bs-target="#exampleModal"/>

        </div>

      </div>

      </div>

    </div>

  )

}

export default Projects

**Output**

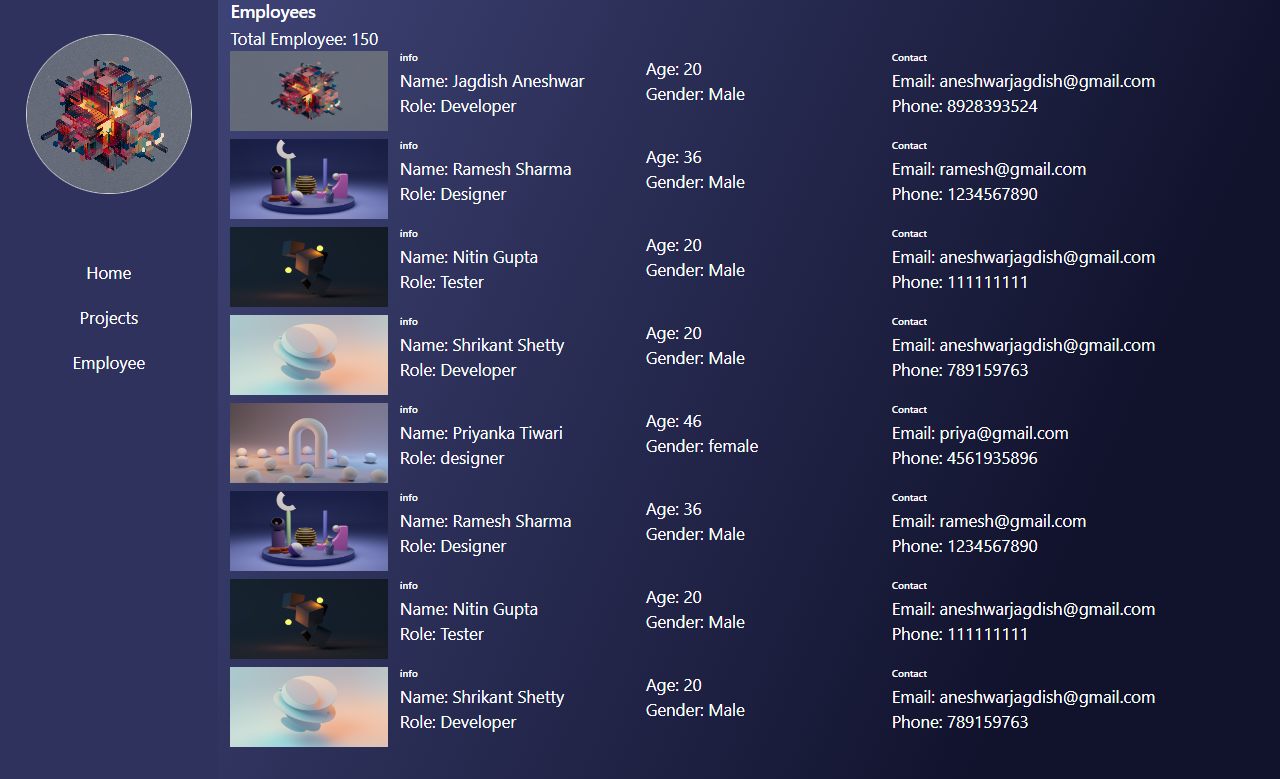
**Home Page**

****

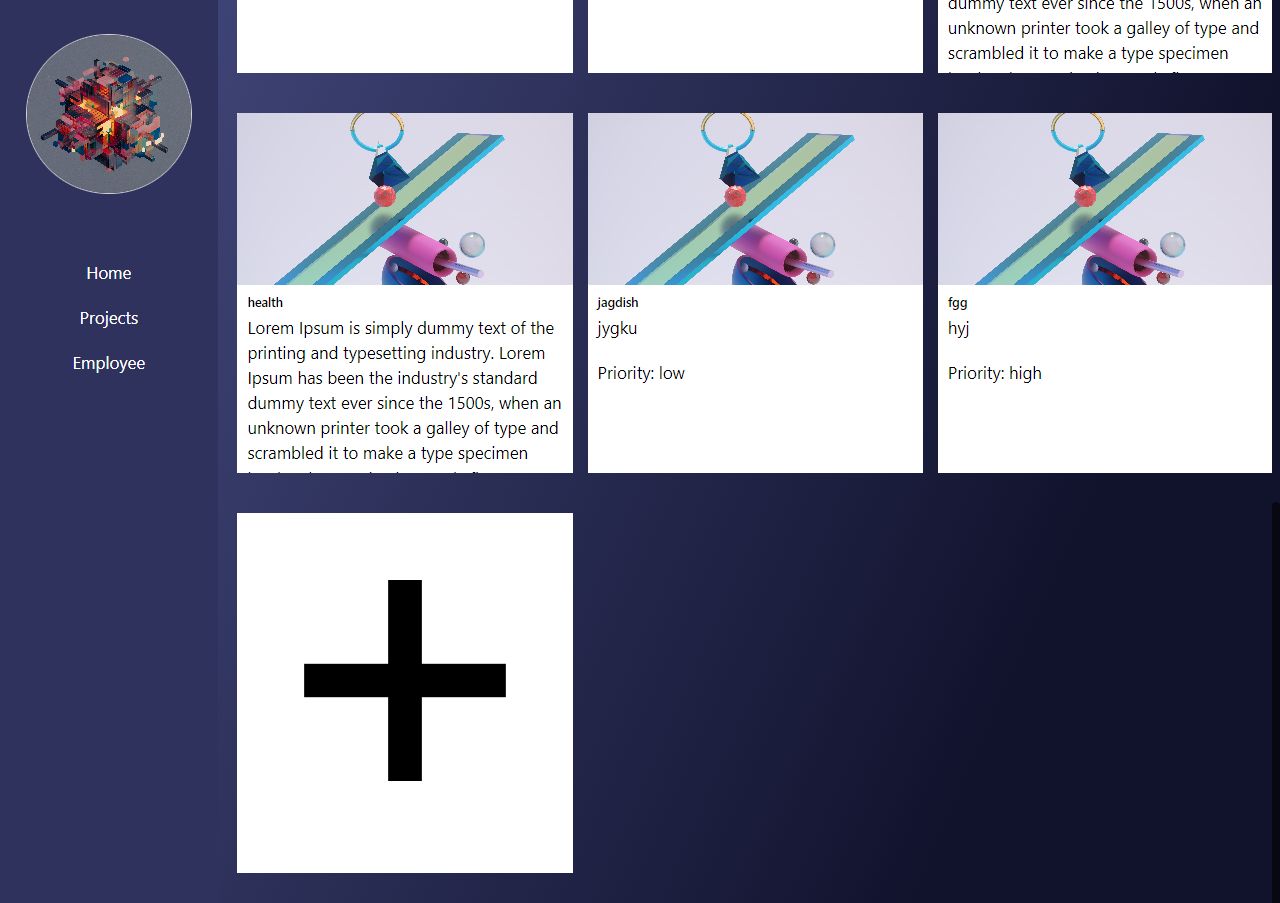
**Project Page**

****

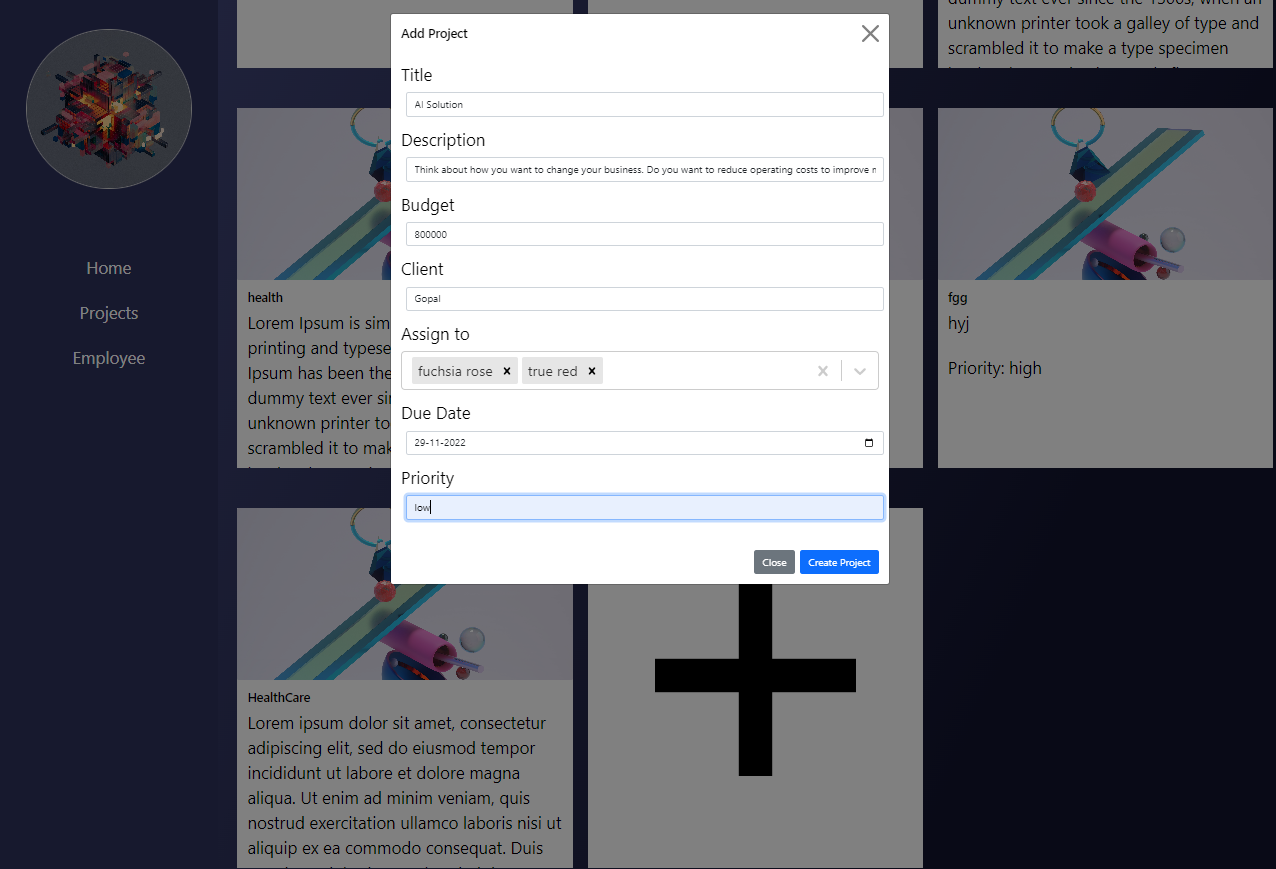
**Employee Page**

****

**Projects Page**

****

**Add Project Modal Page**

****

**TESTING**

Testing is a process of executing a program with the intent of finding error. Testing demonstrates that the software functions appear to be working according to specification, that behavior and performance requirements appear to have been met.

**System Testing**

The system has been thoroughly tested to ensure robustness and reliability.

Various testing strategies and techniques have been used as follows:

**1.Black Box Testing**

Also known as functional testing. A software testing technique whereby the internal workings of the item being tested are not known by the tester. For example, in a black box test on a software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specifications.

**The advantages of this type of testing include:**

• The test is unbiased because the designer and the tester are independent of each other.

• The tester does not need knowledge of any specific programming languages.

• The test is done from the point of view of the user, not the designer.

• Test cases can be designed as soon as the specifications are complete.

**2. White Box Testing**

White box testing strategy deals with the internal logic and structure of the code. White box testing is also called as glass, structural, open box or clear box testing. The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc.

In order to implement white box testing, the tester has to deal with the code and hence is needed to possess knowledge of coding and logic i.e. internal working of the code.

**Advantages of White box testing are:**

i) As the knowledge of internal coding structure is prerequisite, it becomes very easy to find out which type of input/data can help in testing the application effectively.

ii) The other advantage of white box testing is that it helps in optimizing the code

iii) It helps in removing the extra lines of code, which can bring in hidden defects.

**3. Unit Testing**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use.

The most common approach to unit testing requires drivers and stubs to be written. The driver simulates a calling unit and the stub simulates a called unit. The investment of developer time in this activity sometimes results in demoting unit testing to a lower level of priority and that is almost always a mistake. Even though the drivers and stubs cost time and money, unit testing provides some

undeniable advantages.

It allows for automation of the testing process, reduces difficulties of discovering errors contained in more complex pieces of the application, and test coverage is often enhanced because attention is given to each unit.

For example, if you have two units and decide it would be more cost effective to glue them together and initially test them as an integrated unit, an error could occur in a variety of places:

• Is the error due to a defect in unit 1?

• Is the error due to a defect in unit 2?

• Is the error due to defects in both units?

• Is the error due to a defect in the interface between the units?

• Is the error due to a defect in the test?

Finding the error (or errors) in the integrated module is much more complicated than first isolating the units, testing each, then integrating them and testing the whole.

Drivers and stubs can be reused so the constant changes that occur during the development cycle can be retested frequently without writing large amounts of additional test code. In effect, this reduces the cost of writing the drivers and stubs on a per-use basis and the cost of retesting is better controlled

**4. Integration testing :**

Integration Testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program.

The idea is to test combinations of pieces and eventually expand the process to test your modules with those of other groups. Eventually all the modules making up a process are tested together. Beyond that, if the program is composed of more than one process, they should be tested in pairs rather than all at once.

Integration testing identifies problems that occur when units are combined. By using a test plan that requires you to test each unit and ensure the viability of each before combining units, you know that any errors discovered when combining units are likely related to the interface between units. This method reduces the number of possibilities to a far simpler level of analysis.

**You can do integration testing in a variety of ways but the following are three common strategies:**

• **Top-down approach** to integration testing requires the highest-level modules be test and integrated first. This allows high-level logic and data flow to be tested early in the process and it tends to minimize the need for drivers. However, the need for stubs complicates test management and low-level utilities are tested relatively late in the development cycle. Another disadvantage of top-down integration testing is its poor support for early release of limited functionality.

• B**ottom-up approach** requires the lowest-level units be tested and integrated first. These units are frequently referred to as utility modules. By using this approach, utility modules are tested early in the development process and the need for stubs is minimized. The downside, however, is that the need for drivers complicates test management and high-level logic and data flow are tested late. Like the top-down approach, the bottom-up approach also provides poor support for early release of limited functionality.

• **Umbrella approach**, requires testing along functional data and control-flow paths. First, the inputs for functions are integrated in the bottom-up pattern discussed above. The outputs for each function are then integrated in the top-down manner. The primary advantage of this approach is the degree of support for early release of limited functionality. It also helps minimize the need for stubs and drivers. The potential weaknesses of this approach are significant, however, in that it can be less systematic than the other two approaches, leading to the need for more regression testing.

**CONCLUSION**

**Conclusion:**

The “Project Management Application” project developed meets certain objectives of the system.Through this project, it can be concluded that the manual system of management of project management has been removed to some extent. However some changes are still necessary. But this project has been able to reduce the human error and lots of paper work.

**FUTURE**

**ENHANCEMENTS**

**FUTURE ENHANCEMENTS**

The following future enhancements can be possible:

1. GUI can be made more attractive.
2. The client’s account page can be made more dynamic.
3. Note section can be added.
4. Gantt chart can be added to view progress.

**References**

<https://reactjs.org/>

<https://www.mongodb.com/>

<https://www.tutorialspoint.com/mongodb/index.html>

<https://www.intechopen.com/chapters/73966>

<https://www.knowledgehut.com/tutorials/project-management>

<https://www.javatpoint.com/reactjs-tutorial>