Skill Sprint WebApplication

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering/Information Technology

By

Ansh Jaiswal (191541)

Under the supervision of Dr. Ruchi Verma

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology
Waknaghat, Solan-173234, Himachal Pradesh

Declaration

I hereby declare that the work presented in this report entitled **“ Skill Sprint Web Application”** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out under the supervision of **Dr. Ruchi Verma,** Assistant Professor (Grade - I) in the Department of Computer Science and Engineering and Information Technology.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Ansh Jaiswal (191541)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Dr. Ruchi Verma Assistant Professor
Department of Computer Science and Engineering

Acknowledgement

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes it possible for us to complete the project work successfully. I am grateful and wish my profound indebtedness to my supervisor Dr. Ruchi Verma, Department of Computer Science Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of our supervisor in the field of "Web Development" to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartfelt gratitude to Prof. Dr. Vivek Kumar Sehgal, Head of Department, Computer Science Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat.

I am grateful to each and every individual who directly or indirectly helped me in making this project a success.

Finally, I must acknowledge with due respect the constant support and patience of my parents and grandparents.

Table of Contents

List of Figures

Abstract

The website's objective is to enable users to post new job openings and apply for new jobs. My organization gave me this task to give me a better understanding of the abilities required in the field of website design and development. The knowledge we have learned while studying web design and development over the past few months was expected to be put into practise with this website. Three tiers make up the architecture of this website. The presentation tier is kept straightforward and approachable. This tier's primary functions are to display information and gather user data. The core of web applications is the application tier. This is where all of the user data that was gathered from the presentation tier is saved and processed. Node.js is extensively used in its implementation. The information processed by the application tier is saved in the data tier, which is the last tier. This layer is used to store user data such as emails, passwords(encrypted), phone number, etc. This website uses Mongodb to implement the data tier (NoSQL). The chosen library is Mongoose and the platform is MongoCompass.

The three-tier architecture ensured faster development as each tier can be developed simultaneously. It also ensures improved scalability, reliability and security. Enhanced optimization techniques are used for smooth and hassle-free UI experience.

1 INTRODUCTION

1.1 Introduction

I'm working on a MERN stack project called Skill-Sprint as a part of my internship. A user of the online dashboard Skill-Sprint can search through the many job offers that the administrator has uploaded. The API will show the records in the table beneath the filter form after retrieving them from the database based on the filter. Each column will include a "Apply now" button that, when clicked, will display the further information needed to register for the position.

1.2 Problem Statement

Creating a quick, scalable, and secure online application by using industry best practises. to gain understanding of front-end development, apply it to real-world projects, and comprehend the intricate operation of MERN Stack applications.

1.3 Objectives

Employing industry best practices, to produce tested, organized, tidy, and maintainable web apps, to put the technological information we have learned so far into practice and acquire hands-on experience.

1.4 Methodology

This website's front end was created entirely in Reactjs. Styling is done using standard CSS. Scalability and speedier development are made possible by this combination. As long as no errors are made in any one component, all components can be produced simultaneously. ExpressJS was utilized to create the APIs on the backend, while MongoDB served as the database. Mongoose is a framework for ExpressJS and MongoDB.

The task of processing HTTP requests falls to React. This allows them to configure dynamic data downloads without having to reload the website. The website now loads much more quickly than normal.

1.5 Organization

Paxcom: Over 400 e-commerce aficionados make up the Paxcom team, which is committed to using technology to make digital payments and commerce easier for international firms. The e-commerce solutions offered by Paxcom include analytics software, advanced data analytics, marketing and advertising, content, sentiment analytics, software for constructing web stores, ordering, and inventory management. Their cutting-edge systems gather and analyze data from all e-commerce channels to provide insightful information on inventory levels, availability, competitive pricing, product discoverability, and marketing expenses. They also provide enhanced ROI, placement, and other services. In order to promote e-commerce and win customer support, Paxcom has 148 platforms, 450+ PIN codes, 1.4+ rack products, multi-crawl advertising spending each day, and seller and location-specific advertising. It is a member of a large network of businesses that conducts research and operates in a number of different fields, including e-commerce, IT services, billing, and payment solutions. Paxcom is a distinctive SAAS-based program that offers a total e-commerce automation solution, including the establishment and maintenance of e-stores and marketplaces, as well as powerful data analytics, quick sales growth, managed warehouses, and supply chains.

Figure 1: Paxcom logo

Paymentus: A software provider with its headquarters in North Carolina, Paymentus offers comprehensive billing solutions. Paymentus was essentially developed with the intention of streamlining, altering, and improving the way invoices are paid using innovation and technology. With a desire to use technology to innovate and simplify complicated challenges, Paymentus has amassed 1,700 clients, including the biggest businesses in North America, to become the market leader in paperless electronic billing and other associated payment solutions. Paymentus, one of the North American businesses with the fastest four-year growth rates according to Deloitte, is constantly working to create billing and payment platforms that are more efficient, quicker, safer, and less expensive. Paymentus is a highly customer-focused business that strives to continuously increase customer satisfaction and offer the best services to its clientele.

The project or team assigned for Paymentus India Pvt. Ltd. at the Mohali branch was part of this internship with Paxcom India. The internship was with a team working on the development of payment dashboarding and related products as well as other payment data tasks primarily based on Java Core and Java Frameworks like Struts2, JDBC (Java Database Connectivity API), Spring MVC (Model, View, Control), and SpringBoot, along with extensive tasks on the MERN (MongoDB, Express, React, and NodeJS) tech stack. The main focus was on successfully understanding their working structure and pattern as well as gaining experience. Learning soft skills like teamwork and business communication was the main focus. The objectives of the internship were to gain a thorough grasp of some of the existing products and projects of the firm as well as the various coding paradigms utilized by the company to construct its applications.

The organization has a fantastic workplace atmosphere. Work and play are typically mixed together in Paymentus. Virtual technologies like Meet and Slack were used for a significant amount of communication during the internship. The project completed during this time period offered excellent learning opportunities and practice working under pressure and haste to complete real-time, live customer tasks. Whatever the case, this opportunity gave me the most in terms of the duty that required me to put on a specific object because it actually opened the door to my vocation. The most amazing aspect of this temporary job experience was how it successfully surmounted any obstacles between learning new things and really using some of that knowledge in a real-world project and getting paid for it. It was quite helpful that this entry-level role immediately connected with what was needed to begin working on corporate initiatives, from contributing to truly controlling them. The ability to study, practice, and work on Full Stack Engineering abilities as well as develop soft skills like communication was provided through the internship at Paxcom, Paymentus.

Fintech Technology: The term FinTech is made up two words Finance and Technology, it is basically the aggregation of the two fields where the technology is used to innovate and improve newer domains into the world of the finance.

From the ability for consumers or users to view and make financial transactions online, to applications that allows them to make financial transaction within seconds to anyone across the globe, to means that enable financial institutions to make rapid financial transactions in huge volume and velocity, everything is the sub domain of innovation in finance aided with technology. FinTech's closest examples in our day to day life would be implementation of apps like Paytm, GPay etc. or the investment apps like Grow, Zerodha.

One of the biggest opportunity provided by the advent of the fintech in the current marketspace is in terms of aiding the common man with the financial literacy as well as financial accessibility, it has brought more and more control to people’s life in terms of their financial decisions and investments, it has opened the investment opportunities which were earlier beyond the reach of common man.

FinTech's growth is primarily due to the opportunity for smaller players to compete in the same space as traditional banks and financial institutions. Thanks to FinTech, it's not about who is the biggest, but who is the fastest and most responsive to effectively respond to the ever-changing consumer demands. Moreover, the solutions offered by fintech companies are no longer "universal." Instead, it provides targeted (often niche) services that fill gaps in specific financial needs. It may also be offered at a much lower cost than the services offered by traditional financial providers.

Electronic Billing: Electronic billing, or eBilling is the process of sending and paying invoices electronically. This process allows customers to receive invoices in email, a web portal, or machine-readable data format for more efficient delivery and payment. The eBilling process is managed by accounts receivable by receiving results from accounting software or an ERP system and providing them to customers. Aspects of this paperless process are becoming more and more automated to improve speed and accuracy, such as follow-up when invoice payment times are delayed. With the localization of the technology and more and more people accessing the internet and other means of technology, it has brought more and more people paying their bills online generating thousands of transactions per second and henceforth resulting in millions of invoices that needed to be transferred, repealed and stored. This industry revolutionizes the traditional billing system in a lot of intuitive ways. It eliminates process inefficiencies, saves time and money, ensures more accurate and faster delivery of invoices to your customers, and enables instant payments. A well-deployed electronic billing system is a strategic advantage, not just an operational upgrade.

Payment Dashboard Products: Paymentus offers multiple products and services in domains of the payment be it in terms of eBilling, invoice generation or simple payment gateway kind of services.

The current team under which this particular internship is working on the dashboarding product of the Paymentus. The following products are a more of a visualization, analytics, as well as data querying code allowing the opportunity to the end user be it big businesses with large volume of clientele and volume and velocity of transactions,or be it a single end common user, to basically access, analyze the transaction done from their account and address financial questions, or in case of some special test cases also offer to answer certain business logics.

Following are the major dashboards which are currently operating :-

- Agent Dashboard Systems : This feature provides our clients with dedicated customer support to query their customer problems by a dedicated dashboard which includes payment reports , history visualization etc. It also provides customer support with features for IVR and assisted IVR payments.
- Customer Portal : This feature acts as a payment solution to customers by providing them with features like wallets or direct payment portal for registered users. It more or less acts as a direct common user all payment kind of solution and to provide them with multiple accessibilities.
- ROTP (Remote One Time Payment) : This feature is basically a payment gateway kind of solution for the guest users to make payments using multiple sources and multiple types.

2 LITERATURE SURVEY

2.1 React documentation

ReactJS is one of the most widely used JavaScript libraries at the moment, with a strong foundation and a sizable community.

A JavaScript library called ReactJS efficiently advertises and customizes reusable UI components. It is a primary-based, open-source library that only contains the application layer layout. Facebook created and managed it at first, and then used it in products like WhatsApp and Instagram.

ReactJS is one of the most widely used JavaScript libraries at the moment, with a strong foundation and a sizable community.

A JavaScript library called ReactJS efficiently advertises and customizes reusable UI components. It is a primary-based, open-source library that only contains the application layer layout. Facebook created and managed it at first, and then used it in products like WhatsApp and Instagram.

These parts are arranged inside of higher-level parts that establish the organization of the application. It is possible to write each form element as a React component, which we then combine to create the form component itself. The form's components would specify both the form's layout and the contents of the form.

2.2 Javascript documentation - Mozilla

Developing asynchronous Javascript at any level, from beginner to expert, is necessary for creating a robust and scalable online application.

2.3 Git and Github

An official documentary that introduces the ideas behind Git, a version control system, and how it integrates with GitHub.

2.4 MERN: A Full-Stack development

The phrase "MERN Stack" primarily refers to the terminology used to develop a complete stack platform utilizing JavaScript as the programming language.

The most widely used programming language today in terms of technology is Javascript.

A group of technologies known as the MERN stack enable quicker application development. Worldwide, developers utilise it. The fundamental goal of the MERN stack is to create apps that solely use JavaScript. This is due to the fact that all four of the technologies that comprise the technological stack are JS-based. So, the backend, frontend, and database may be operated simply if one is familiar with JavaScript (and JSON).

2.5 MongoDB documentation

A document-oriented database is MongoDB, a NOSQL system. Collections of documents that resemble JSON are used. These documents allow for embedded fields, allowing for the storage of pertinent data. Although MongoDB enables both vertical and horizontal scalability, it is easy to refer to it as a bigger database.

Using an interface similar to Mongoose, Mongo is called and used with the Node and Express backends. The object data modeling (ODM) library for MongoDB, Mongoose, may be considered the implementation interface linking the real data with the application.

In contrast to SQL, the MongoDB schema is adaptable and changeable on the fly.

2.6 Introducing JSX - React

A render method is a feature of every React component. A React component's HTML output is defined by the render function. A React plugin called JSX (JavaScript Extension) enables the creation of JavaScript code that resembles HTML. To put it another way, JSX is an extension of ECMAScript utilized by React to allow HTML-like syntax to coexist alongside JavaScript/React code. Preprocessors (also known as transpilers, such as Babel) employ the syntax to convert HTML-like syntax into typical JavaScript objects that a JavaScript engine can interpret. JSX coding tutorial for beginners on the official React website. Describes how the Babel compiler operates as well.

JSX allows you to write HTML/XML-like structures (e.g., DOM-like tree structures) in the same file where you write JavaScript code, and then the preprocessor will transform these expressions into actual JavaScript code. Just like XML/HTML, JSX tags have a tag name, attributes, and children.

2.7 Object Oriented Programming

Object-oriented programming (OOP) is a programming paradigm which does not have a major focus on functions and logic, but is more concerned about the functionality that revolves around data and objects.

It merely focuses on the logic required for the management, but strongly on the objects. For the projects and ideology that is large and is prone to update, the developers prefer this method of programming. If the work is concerned with the production systems, designing, or mobile applications, the developers aim at working on the principles and features of OOPs.

2.8 Structure of OOPS

- Classes: It is just a blueprint of the methods, attributes and objects. It is considered to be a logical entity and used to define the states of variables, methods and attributes.

- Objects: Object is an actual instance of a class that implements the traits of the methods and attributes of the class. It is an entity that does not exist logically but has a structure and an existence.

- Methods: These are the functions inside a class to describe the behaviors of objects. They are meant to perform certain tasks.

- Attributes: Attributes are needed to store the information and data. Inside a class, they are used to represent the state and quality of the objects.

2.9 Features of OOPS

- Encapsulation: As the name suggests, encapsulation means to put the content of the class into a capsule. The important information is bound inside it and all the unnecessary details are hidden from the user.

- Data Binding: Binding the information, attributes, methods and data inside a class.

- Data Hiding: Hiding the unnecessary details from the user. Its major focus is to provide security.

- Abstraction: This is a feature which is intended to not disclose the backend functionality and details to the user, as it is totally not needed.

- Inheritance: Inheritance is a feature by virtue of which classes can inherit the properties of some other classes. The classes which inherit the properties are called the derived classes and whose properties are inherited are called the base classes.

- Polymorphism: Polymorphism is the process by which a piece of code, data, technique, or object acts differently in different situations.

- Compile-time Polymorphism: Compile time polymorphism, commonly referred to as static polymorphism, is a sort of polymorphism that occurs during the compilation process. The compiler determines the shape or value that the entity in the image must take.

- Run-time Polymorphism: Runtime polymorphism, also known as Dynamic Polymorphism, is a sort of polymorphism that occurs while the programme is running. It signifies that the compiler cannot decide. As a result, the execution determines what shape or value must be taken.

2.10 Data Structures

Data Structures is a medium of storing data and organizing it. It is a way of arranging data on the system in order to have an efficient access and update. During the internship, before we could actually move to the major work ahead, our mentor wanted us to brush up our skills on Data Structures.

During the course of our proceedings, we were not asked to go through all the data structures, just a few of them that were needed for our further meetings.

Arrays: Arrays are the Data Structures that enable contiguous memory allocation. It follows an index based allocation, which makes it easier to access an element. All the elements have the same data type.

Stack: Stack are the special Data Structures that follow the Last In First Out principle.

It is a linear data structure.

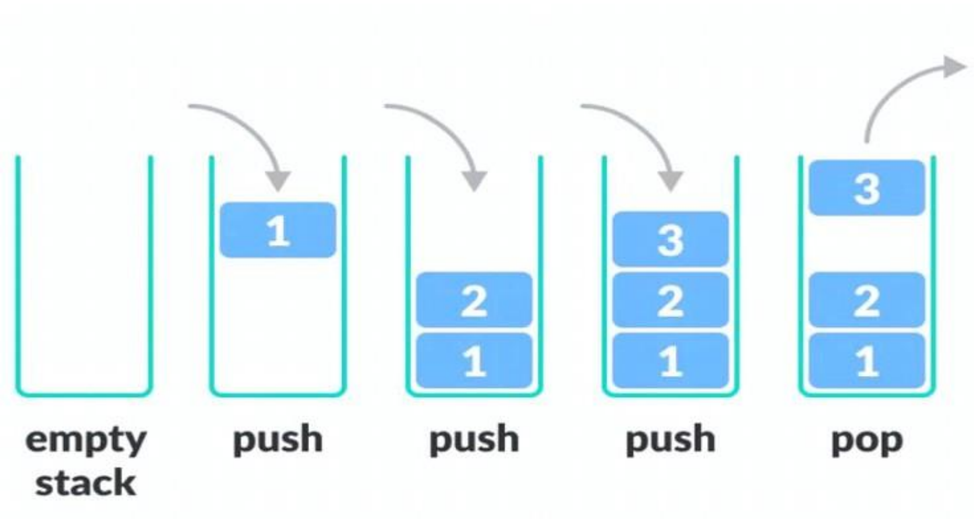


Figure 2: Stack Representation

Queue: Queue Data Structure follows the First In First Out principle. It is a linear data structure.

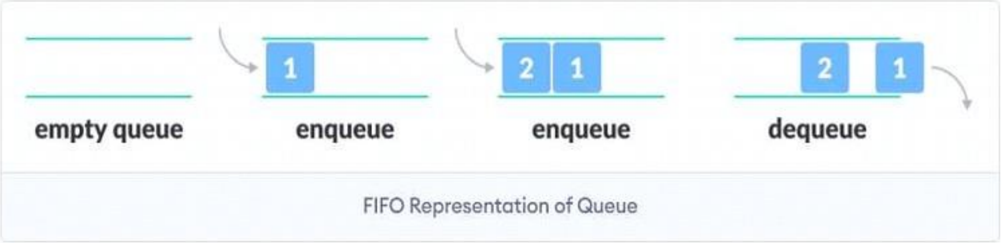


Figure 3: Queue Representation

The two operations are Enqueue and Dequeue.

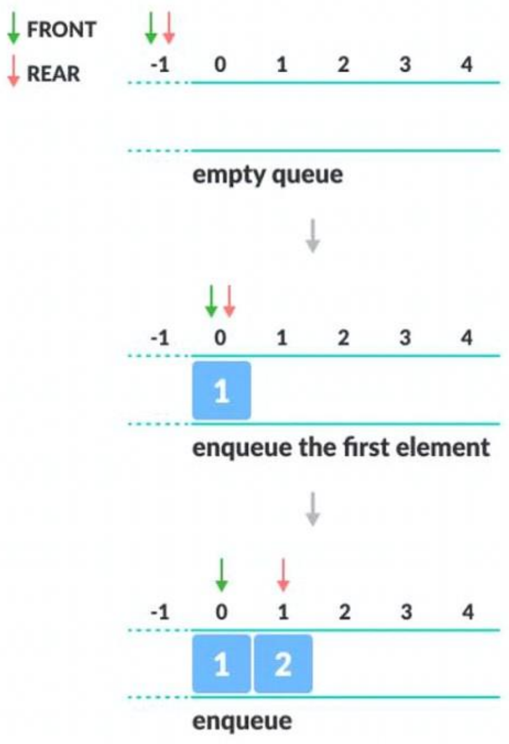


Figure 4: Enqueue

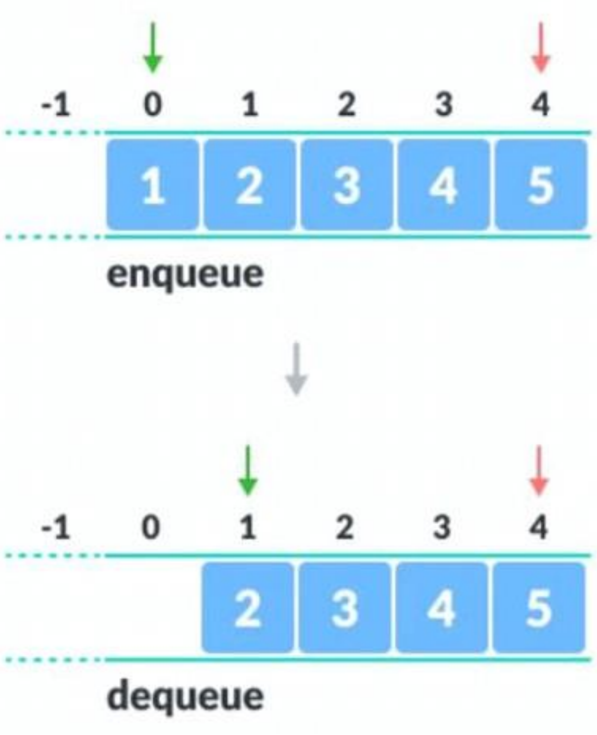


Figure 5: Dequeue

Linked List: This is a linear Data structure in which elements are connected to each other in a series of nodes. A node contains two important pieces of information, the value and the address of the next node.

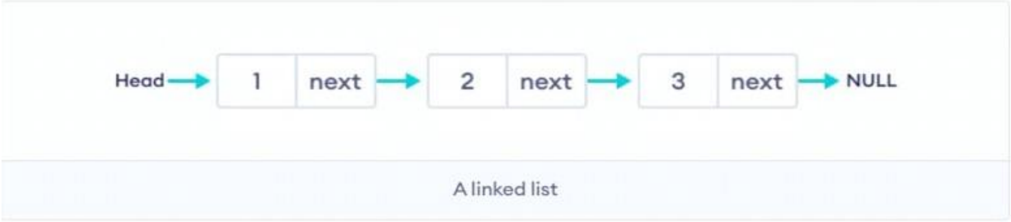


Figure 6: Linked List

3. SYSTEM DEVELOPMENT

3.1 System design diagram

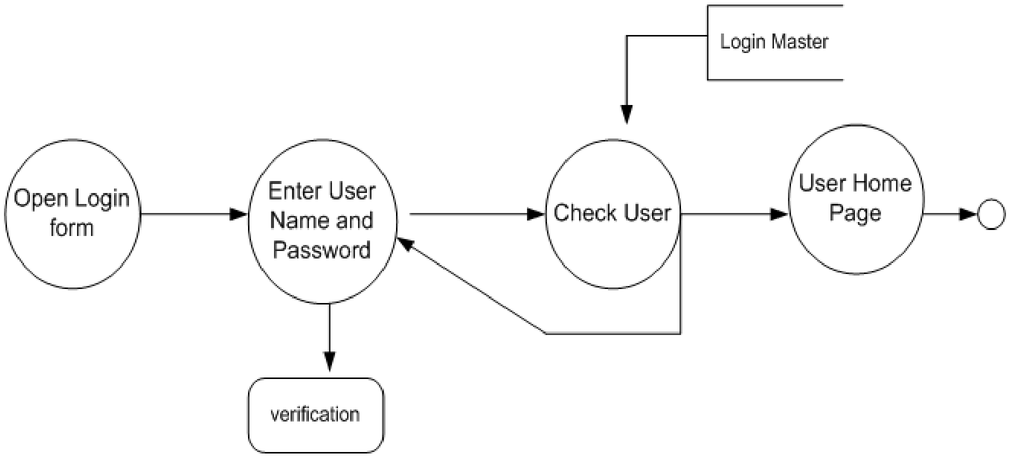


Figure 7: System design diagram

3.2 System design implementation

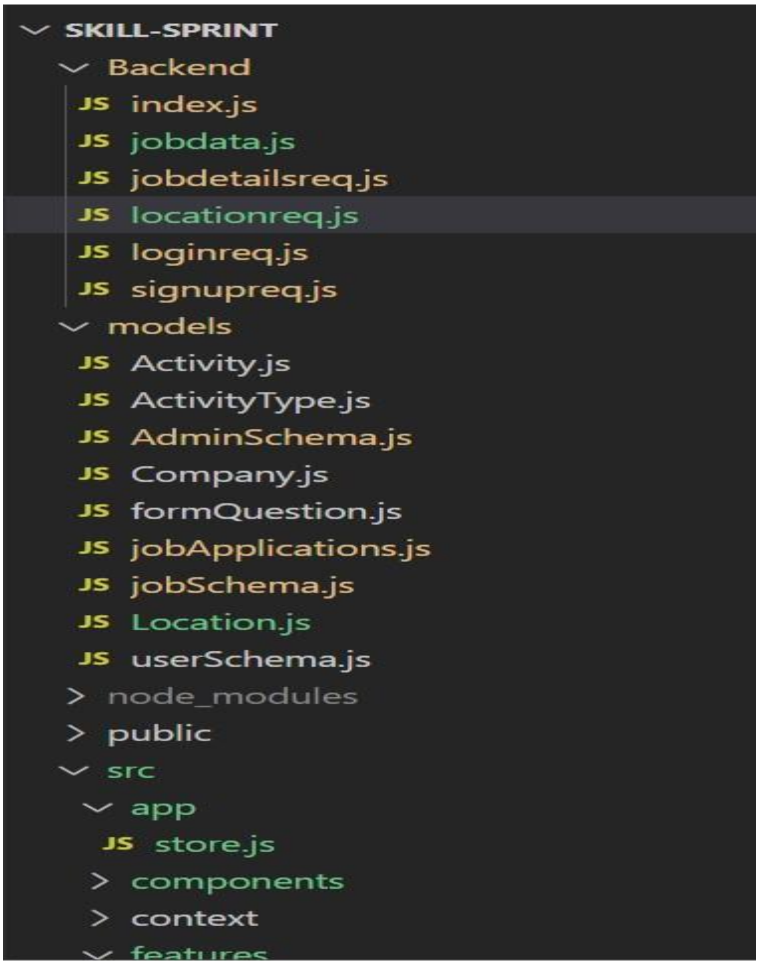
Separation of concerns : Client-side and server-side apps often have their own sub-folders in a react web application. The hierarchy is then used to partition each sub-folder into its own folders and files. The creation, deletion, updating, and retrieval of the data are controlled in the server side folder, while the client side is mostly in charge of the user interface and experience.

3.2.1 Identification of features

The features of a web application include:

- Creation of a post with a test body and image in jpg format.
- Updation of a post only by a user who is logged-in.
- Deletion of an existing entry only by the user who created it.
- Fetching a post based on title or hashtag.

- Creating a user (signup).
- Letting the user login.



```
⌄ SKILL-SPRINT
  ⌄ Backend
    JS index.js
    JS jobdata.js
    JS jobdetailsreq.js
    JS locationreq.js
    JS loginreq.js
    JS signupreq.js
  ⌄ models
    JS Activity.js
    JS ActivityType.js
    JS AdminSchema.js
    JS Company.js
    JS formQuestion.js
    JS jobApplications.js
    JS jobSchema.js
    JS Location.js
    JS userSchema.js
  > node_modules
  > public
  ⌄ src
    ⌄ app
      JS store.js
    > components
    > context
    ⌄ features
```

Figure 8: MERN work environment

3.2.2 Libraries and frameworks used

JS is a powerful, yet lightweight scripting language that may be either compiled or interpreted. It is one of the greatest server-side scripting languages and is used by several non-browser contexts, including Node.js, Apache CouchDB, and Adobe Acrobat.

Reactjs : React is a Javascript-based front-end technology that is open-source and free and is used to build component-based user interfaces. Meta and a number of individual programmers and companies keep it updated. React may be used to build single-page, mobile, or server-rendered apps with frameworks like Next.js. Because React is only interested in the user interface and rendering components to the DOM, libraries are typically used to provide routing and other client-side functionality in React applications.

Nodejs : Environment for open-source servers Node.js is cross-platform and compatible with many different operating systems, such as Windows, Linux, Unix, macOS, and others. The V8 JavaScript Engine is used by the back-end runtime environment Node.js, which enables JavaScript code to be run outside of a web browser. Developers may utilize JavaScript, a scripting language, to build command-line tools and server-side scripts for Node.js applications. The server frequently has to be able to run JavaScript code in order to build dynamic web page content before the page is sent to the user's web browser.

Mongoose : A MongoDB library called Mongoose is used to create understandable, succinct code. It manages data relationships, provides validation, and converts between the representations of the same objects in MongoDB and those written in code. This shows that Mongoose makes it possible to define objects with strongly-typed schemas that correspond to MongoDB documents. Mongoose provides a mind-boggling array of tools for creating and dealing with schemas. Using Mongoose, CRUD operations that are difficult to do with raw MongoDB may be completed fast and efficiently.

Nodemon : Nodemon launches the server when we make modifications to the file and save them. Without Nodemon, every update requires an automatic server restart. It reduces time and effort significantly. The "nodemon app" command may be used to test the website on localhost.

3.2.3 Technical Requirements

- **VS Code**(preferred IDE)
- **Postman** api platform for building and testing APIs.
- **MongoDBCompass** (Nosql) database.

3.3 Website Design

Header includes a navigation bar and the website's name. Links to the "Profile,"Jobs "," and "Help" sections can be found in the navigation bar. **Body,** This has unique stuff on each page. For instance, the content of the "Jobs" page comprises all the jobs and details in a tabular form along with the apply button, but the "security" page has a login form.

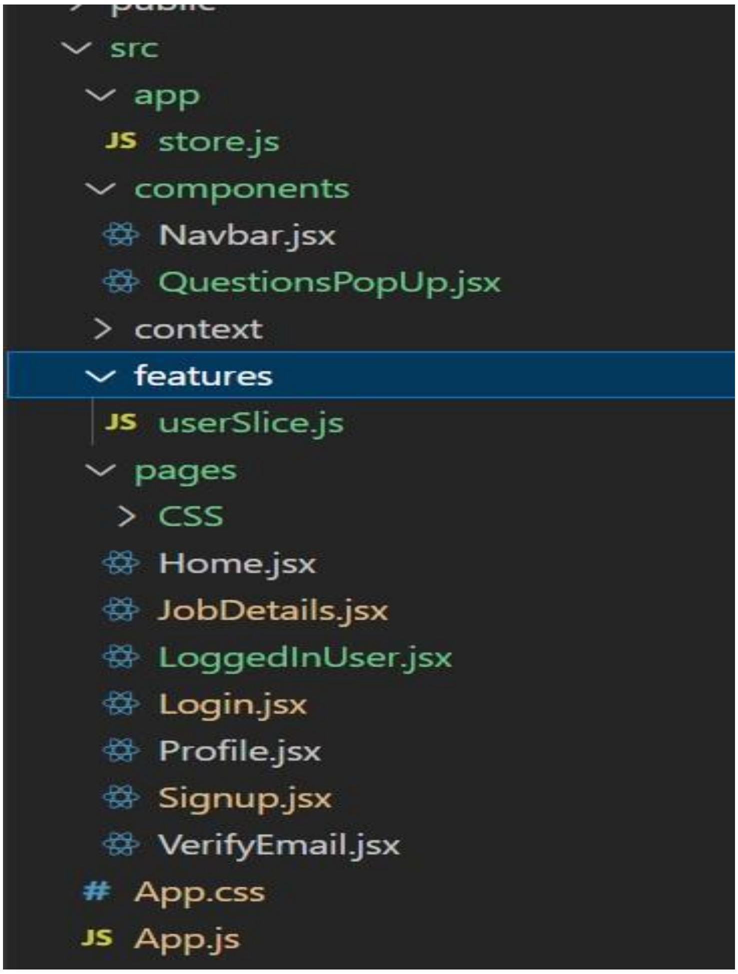
Front-end implementation

MOVING FORWARD WITH REACT

3.3.1 Bootstrapping a basic react application

Run the ‘npx create-react-app **Skill Sprint** ’ command, and it will automatically generate a folder containing all of the necessary files for building a react application.

3.3.2 React Workspace and folder hierarchy



```
> public
v src
  v app
    JS store.js
  v components
    Navbar.jsx
    QuestionsPopUp.jsx
  > context
  v features
    JS userSlice.js
  v pages
    > CSS
    Home.jsx
    JobDetails.jsx
    LoggedInUser.jsx
    Login.jsx
    Profile.jsx
    Signup.jsx
    VerifyEmail.jsx
  # App.css
  JS App.js
```

Figure 9:Bootstrapped react app made using npm create-react-app command

3.3.3 Installing packages and dependencies

With the following instructions, numerous packages and dependencies may be installed in React:

npm install packagename - for installing dependencies normally.
npm i - for installing all the dependencies in one go.

3.4.4 Importing and exporting components

To reuse a component repeatedly in React, we may export it and then import it into a parent component.

Importing :

Importing a component -

import Gallery from

'./Gallery.js'; Importing

dependencies - import

axios from ‘axios’;

Importing hooks - import

{useState} from ‘react’;

Exporting :

We can export a component as follows : export default function App() {

```
  return (
    <Gallery />
  );
}
```

3.3.4 How react works (JSX) :

JSX stands for JavaScript XML. JSX allows you to write html inside javascript. It is this feature that makes react so powerful and clean.

An expression in JSX : const myElement = <h1>Current version of React is {9+9}.</h1>;

A block of html in react can be written as :

```
const myElement = (
  <div>
    <p>First paragraph.</p>
    <p>Second paragraph</p>
  </div>)
```

States in React

Each modification performed by the user is regarded as a state change in React. A state includes details about the element it is present in. A component renders itself in a new state whenever we alter its state. A component's state may be changed using the setState() function Object() { [native code] }. For instance, if we type something into the search bar, the component must re-render because the state changes with each letter.

Example :

```
Class MyClass extends React.Component { constructor(props) {
super(props);
this.state = { attribute : "value" };
}
}
```

3.3.5 Props in react

Properties are denoted by the abbreviation "props." similar to HTML attributes. The main distinction between a state and a prop in react is that a prop may be handed from a parent component to a child component, but a state cannot. This procedure is referred to as "prop drilling."

Eg:-

Adding an attribute called ‘brand’ to ‘Vehicle’ component :

```
const Ele = <Vehicle brand="Tata" />;
```

Passing the prop to the component :

```
function Truck(data) {

    return <h1>The price is : { data.price }</h1>;
}
```

3.4.8 Language Integrated Query (LINQ)

A group of technologies collectively known as Linguistic Integrated Query (LINQ) include query functionality in the C# language. Before, the developer was expected to get familiar with many query languages for various data sources, including SQL databases, XML, web services, etc. With the least amount of code possible, LINQ may offer customers several benefits, including filtering, grouping actions, and ordering. LINQ prefers a straightforward query expression pattern for querying and transforming data in a SQL database, XML document or stream, or.NET collection.

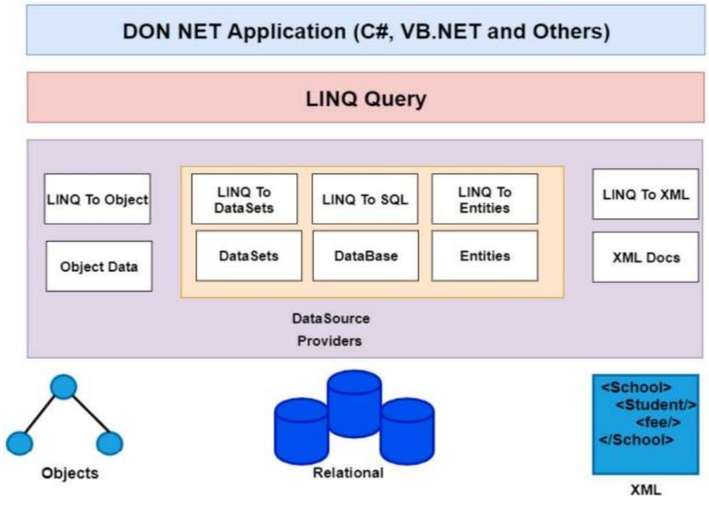


Figure 10: LINQ Architecture

3.4 State Management using React-redux:

One of the most crucial React technologies is Redux. It is an open-source Javascript library that is used to keep track of all the states in one place. It is mostly employed for managing the state. When there are several states, Redux saves them all in a single area known as the Redux Store. The 'actions' are then sent out to update the data. Redux offers a straightforward and practical method for managing states. It operates under the tenet of "unidirectional data flow." It aids in scaling the application and improving its management.

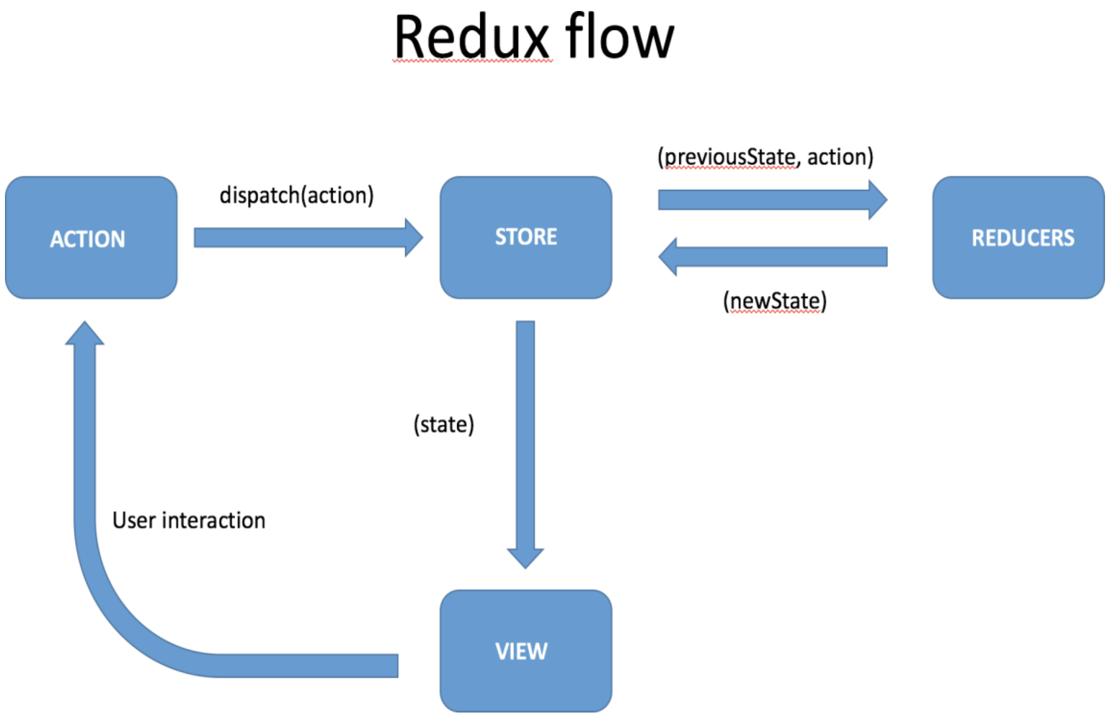


Figure 11 **:** The flow control of redux

The three main components of Redux are :

Redux Store : Stores the current states of all the components. Whenever the data is needed, we need to access the Store. The reducers then update the data and an action is dispatched. The updated data is then stored in the Redux Store.

Creating a Redux Store :

```
const store = createStore(reducers) ReactDOM.render(
   <Provider store={store}>

      <App />

</Provider>, document.getElementById('root'));
```

Action : Actions are straightforward objects with the typical two properties of type and payload. The payload property is an optional property that holds some data that is necessary to complete any given task, whereas the type property is often a text that specifies the action.

Sending information from the application to the Redux store is the primary purpose of an action.

Reducers : Reducers are pure functions that adjust the application's state in response to user input. Reducers accept an action and a previous state as input and output a modified state. Due to the immutability of the state, a reducer always produces a new state that represents an updated version of the prior state. Basically it is used to perform some action and update data/state present in the Redux Store.

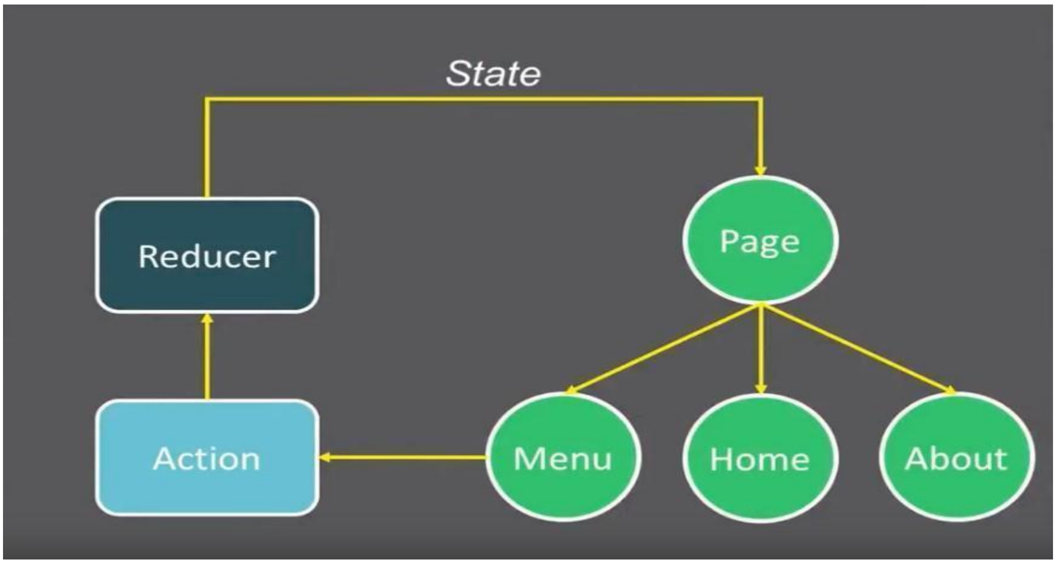


Figure 12 : Reducers

```
case AUTH:
```

```
localStorage.setItem('profile',

JSON.stringify({ ...action?.data

})); return { ...state,

authData:action?.data };

return { ...state, authData: null, loading: false, errors: null };
case LOGOUT: localStorage.clear();
```

Here the reducers are updating the state when a user tries to login and logout.

Data flow in a Redux application :

- The flow of data is triggered when a user interacts with a component. The action creators dispatch an ‘action’ due to this interaction.
- Once an action is dispatched, it is received by the application's root reducer and distributed to all other reducers. Therefore, based on the dispatched action, it is the reducer's responsibility to decide whether it needs to update the state.
- This is verified by filtering out the necessary actions using a straightforward switch statement. Each (smaller) reducer in the application accepts the dispatched action and returns a newly updated state if the type of the dispatched action matches.
- It is important to remember that with redux, the state never actually changes. Instead, the reducer always creates a new state that is an exact duplicate of the old state but has undergone some changes.
- The component is then notified by the store of the altered state, which causes it to retrieve it and render the component again.
- The fact that data flow in a React-Redux application is unidirectional, or only going in one direction, is another crucial finding in this context.

Routing

React Router is a commonly used utility in React for creating custom routes. It facilitates modifying the browser's URL, allowing browsing between numerous pages produced by different React components, and maintaining UI synchronization with the URL.

Installation : npm install react-router-dom

Usage : This application has five main routes. Home, Auth, Posts, Search, :id. The exact paths are as follows :

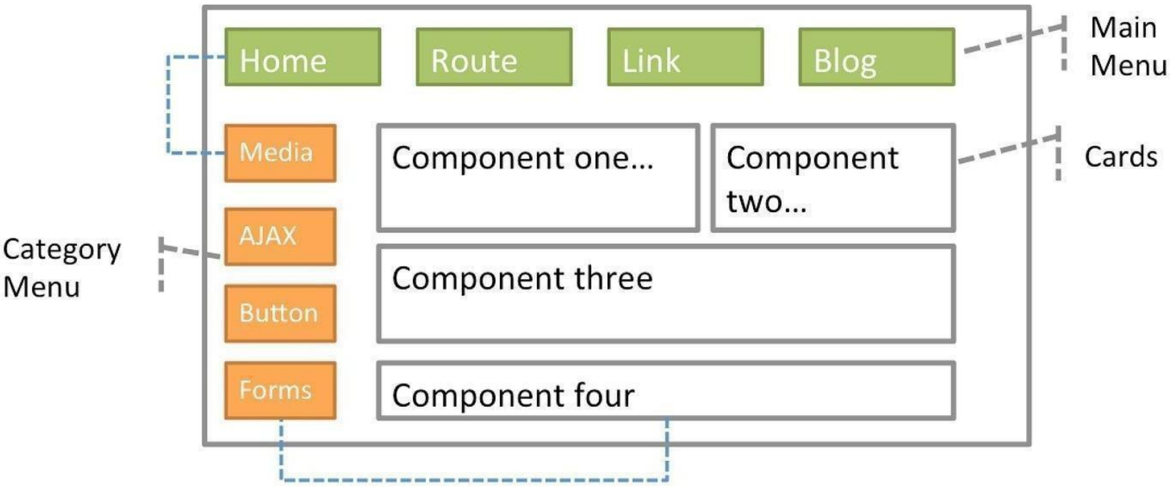


Figure 13: Working of react routMongoDB and Mongoose

Mongoose is an ODM library for MongoDB and Node.js. It offers many different kinds of validation and also manages relationships between data. Some features of MongoDB are :

- Schemaless NoSQL
- Data stored in the form of json objects.
- No fixed structure.
- Fast as it is written in C++.
- Reduces complexity of deployment.
-

3.7 MongoDB Models

Total 5 models are created in this application :

1. Admin Model: It is used to store the login and personal details of the administrator. It includes the name, email, password and phone number of the admin along with the companies for which he is eligible to post jobs. Name, email and phone number are required so that the applicants can contact the admin if necessary.

A password is required for validation.
Company is required for the integration of the project.

2. User Model: It is used to store the login and personal details of the normal user. It includes the name, email, password and phone number of the user along with previous experience, skillset and current company for the admin to judge the applicant.

3. Job Model: It is used to store the jobs posted by the admin with details including required skillsets, due date, description, title, CTC and location. When the admin posts a new job, its details are stored automatically in this model.

4. Questions Model: It is a sub-model inside the Job model and it stores all the questions related to that job posted by the admin in this model. It also stores a reference ID of the job the questions are related.

5. Job Application Model: It is the model which stores the application of the user whenever he applies for a job and the answers to the questions given by the user for that job. It also contains a submodel for answers of the questions asked by the administrator.

Example for a MongoDB model:

```
const mongoose=require('mongoose');

const adminSchema=new mongoose.Schema({
   name:{type:String, required:true},
   email:{type:String, required:true},
   password:{type:String, required:true},
   phoneNo:{type:String,},
   companies:{type:Array,},
})

const Admin= mongoose.model('ADMIN',adminSchema);
module.exports= Admin;
```

3.8 WEB API

API stands for Application Programming Interface. It is a software intermediary that allows two or more apps to communicate with each other.

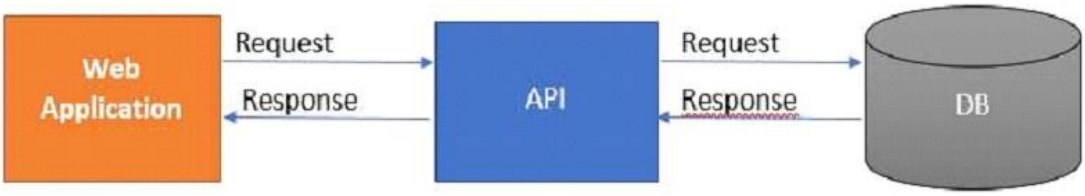


Figure 14: Web API

A web API is an application programming interface for a web application or web server, to put it simply. It communicates with clients and websites using the HTTP protocol for data access.
The Asp.net Core web API is a platform-independent web API.

3.8.1 Why is Web API required?

The customer wishes to utilize the app on a variety of platforms, including mobile, web, and Google devices. A web API can be handy in this situation.

The web API receives requests from many devices and responds in JSON format. The majority of gadgets can interpret JSON output.
Take a look at the web API architecture diagram below.

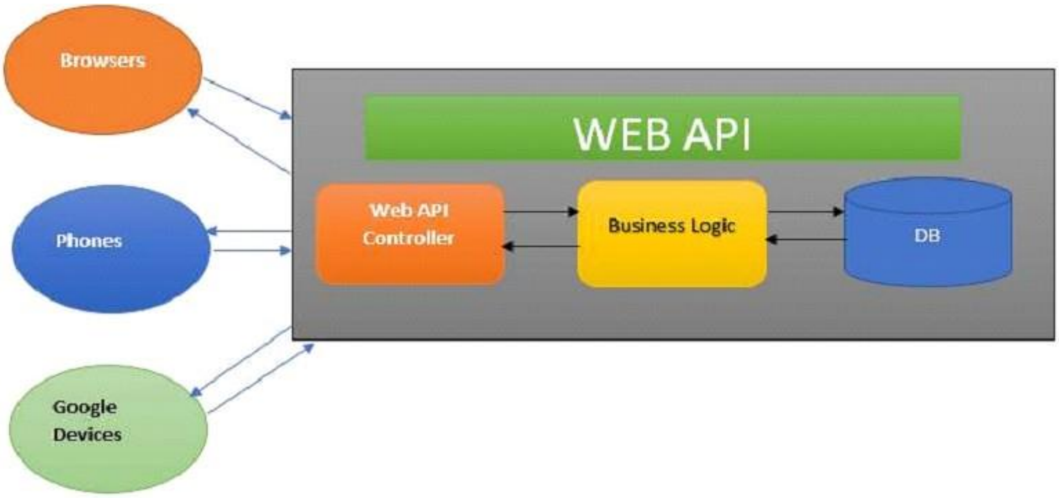


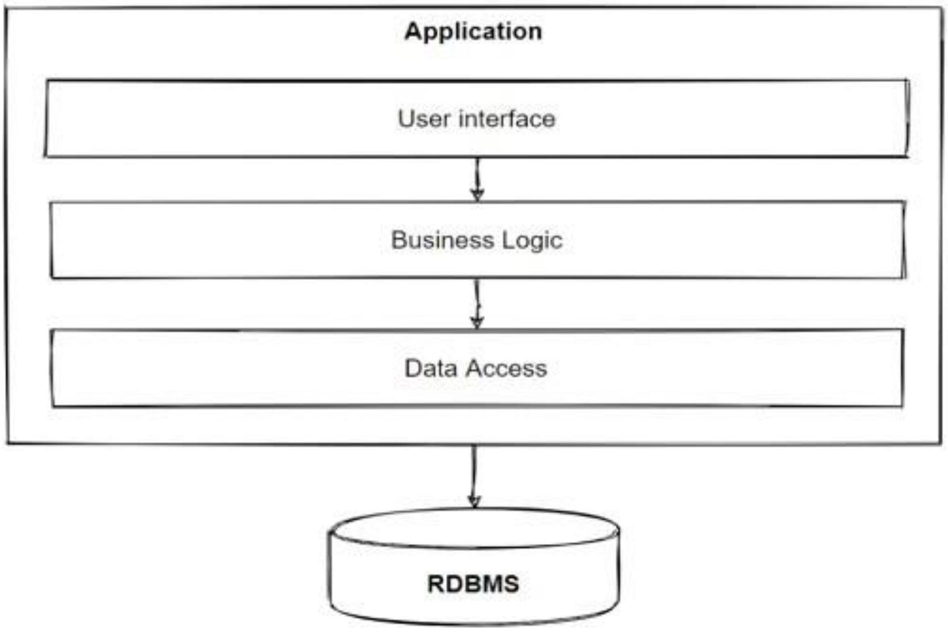
Figure 15: API Architecture

Web API project Objective:

The objective of this subproject was to convert our previously created C# console application into a .Net based web API. This API can handle customer booking services and operations and store data in an SQL Server database. This Web API followed N- Layered Architecture
N-Layered Architecture: Using N-tier architecture, we can divide our project

into different layers, enabling separation of concerns.

N-Layered Architecture: using N-tier architecture we can divide our project into different layers enabling separation of concerns.



`

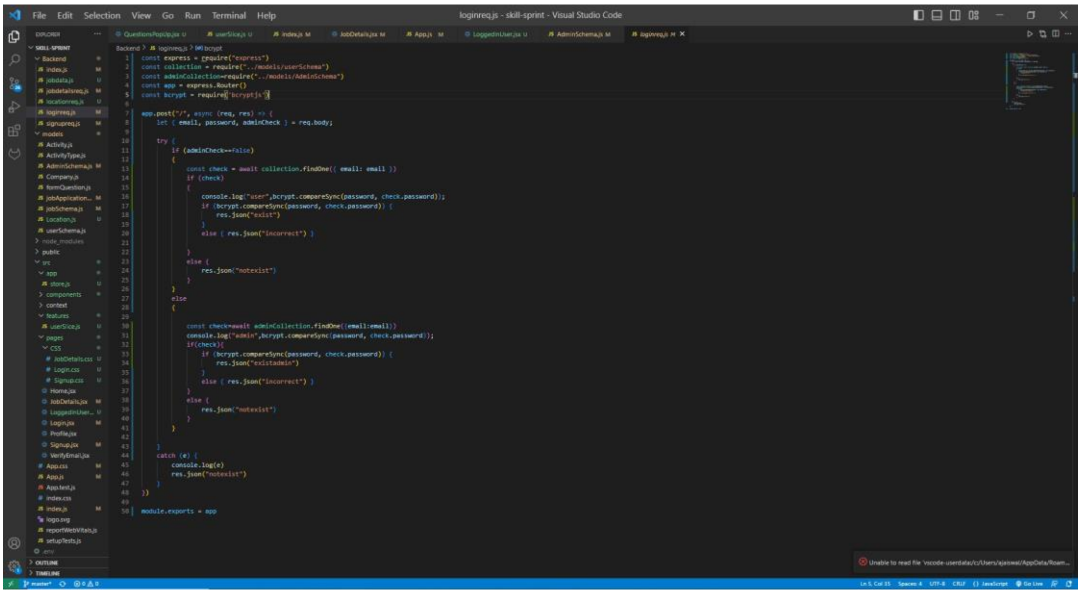
Figure 16: N Layer Architecture

3.8.3 Controller

A Web API controller is a class that may be built in the Controllers folder or any other location in the root folder of your project. A controller class must have a name that ends in "Controller" and be inherited from the system. Web.Http.ApiController class. Action methods refer to all of the controller's public methods.

Business Logic Layer: Business Logic Layer is responsible for all the logical operations of WEB API. It consists of different functions which are invoked in the controller. Business Logic Layer further connects to Data Access Layer for storing and retrieving data from our database.

Data Access Layer: It helps in performing all the operations on the database.



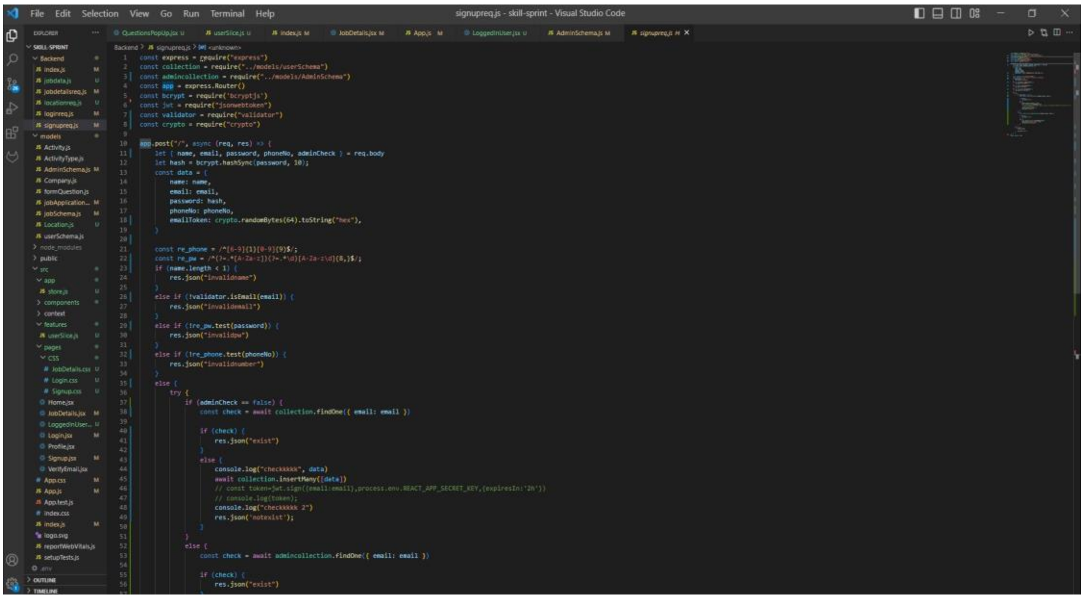
```javascript
const express = require("express")
const collection = require("../models/userSchema")
const adminCollection=require("../models/AdminSchema")
const app = express.Router()
const bcrypt = require('bcryptjs')

app.post("/", async (req, res) => {
    let { email, password, adminCheck } = req.body;

    try {
        if (adminCheck==false)
        {
            const check = await collection.findOne({ email: email })
            if (check)
            {
                console.log("user",bcrypt.compareSync(password, check.password));
                if (bcrypt.compareSync(password, check.password)) {
                    res.json("exist")
                }
                else { res.json("incorrect") }

            }
            else {
                res.json("notexist")
            }
        }
        else
        {

            const check=await adminCollection.findOne({email:email})
            console.log("admin",bcrypt.compareSync(password, check.password));
            if(check){
                if (bcrypt.compareSync(password, check.password)) {
                    res.json("existadmin")
                }
                else { res.json("incorrect") }
            }
            else {
                res.json("notexist")
            }
        }

    }
    catch (e) {
        console.log(e)
        res.json("notexist")
    }
})

module.exports = app
```

Figure 17 : Login Controller

Login Controller: It is used to control the login requests from the frontend to the backend. It connects the frontend react part to the database and stores the session of the user. Authentication and authorization are also performed with the help of the Login Controller.

Job Details Controller: It is used to store the job details in the database and send the success status as a response or an error 404 as a response if unsuccessful.

Locations Controller: It is used to verify the location of the user and check if he is fit for the job he is applying.



```javascript
const express = require("express")
const collection = require("../models/userSchema")
const admincollection = require("../models/AdminSchema")
const app = express.Router()
const bcrypt = require('bcryptjs')
const jwt = require("jsonwebtoken")
const validator = require("validator")
const crypto = require("crypto")

app.post("/", async (req, res) => {
    let { name, email, password, phoneNo, adminCheck } = req.body
    let hash = bcrypt.hashSync(password, 10);
    const data = {
        name: name,
        email: email,
        password: hash,
        phoneNo: phoneNo,
        emailToken: crypto.randomBytes(64).toString("hex"),
    }

    const re_phone = /^[6-9]{1}[0-9]{9}$/;
    const re_pw = /^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/;
    if (name.length < 1) {
        res.json("invalidname")
    }
    else if (!validator.isEmail(email)) {
        res.json("invalidemail")
    }
    else if (!re_pw.test(password)) {
        res.json("invalidpw")
    }
    else if (!re_phone.test(phoneNo)) {
        res.json("invalidnumber")
    }
    else {
        try {
            if (adminCheck == false) {
                const check = await collection.findOne({ email: email })

                if (check) {
                    res.json("exist")
                }
                else {
                    console.log("checkkkkk", data)
                    await collection.insertMany([data])
                    // const token=jwt.sign({email:email},process.env.REACT_APP_SECRET_KEY,{expiresIn:'2h'})
                    // console.log(token);
                    console.log("checkkkkk 2")
                    res.json('notexist');
                }
            }
            else {
                const check = await admincollection.findOne({ email: email })

                if (check) {
                    res.json("exist")
```

Figure 18 : Signup Controller

Signup Controller: It is used to control the registration of a new user. It handles the registration for both administrators and normal users. It encrypts the password for all users using 16-bit encryption and then stores the details in the database.

3.8.4 Postman

Postman is an API creation, testing, and modification tool. This program contains practically every functionality a developer might want. Over 5 million developers use it every month to make API development simple. It can perform a wide range of HTTP requests (GET, POST, PUT and PATCH), save environments for later use, and convert APIs to code in a number of languages (like JavaScript and Python).

3.8.5 HTTP Request method

A client (browser) sends an HTTP request to the server, which is subsequently answered by the server. The answer may include the requested content as well as status information about the request.

HTTP methods: GET, POST, PUT, HEAD, DELETE, PATCH, OPTIONS, CONNECT, TRACE

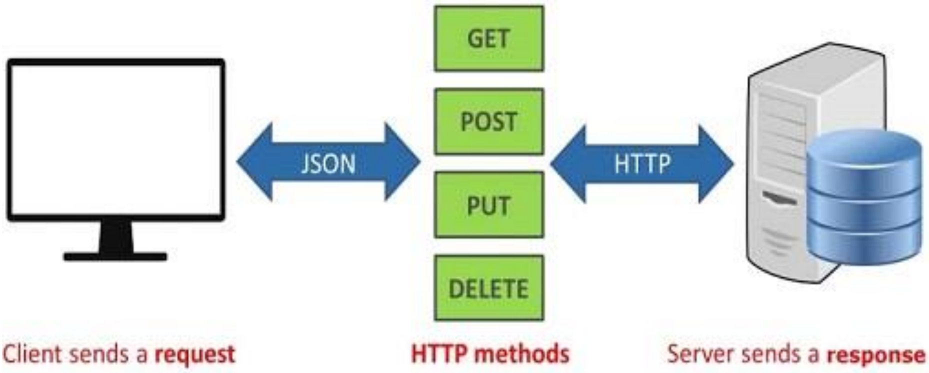


Figure 19 : HTTP Request

3.8.6 .NET Framework

.Net is an open source and a cross platform used for different operating systems in order to develop console and web applications. And, .NET

Framework is an actual implementation of .NET to develop applications on Windows.

- **Architecture of .NET Framework :**

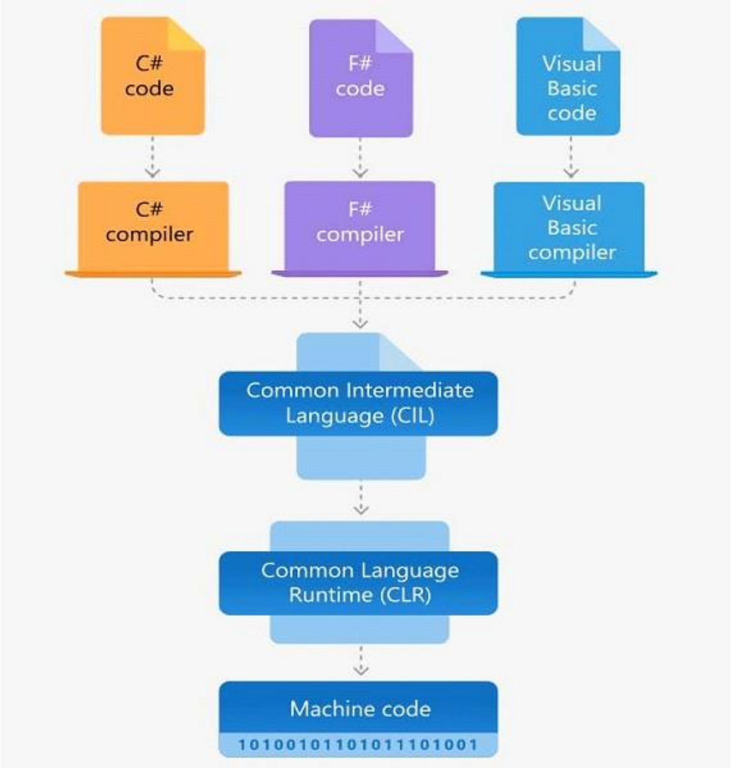


Figure 20: Architecture of .NET Framework

The major components of .NET Framework are:

- Common Language Runtime (CLR) : The running applications are handled by an engine which is executive in nature. The major beneficiary properties offered by it, includes threading, management, type-safety, and handling the exceptions and errors.

- Class Library: When the need comes to writing and reading files, connection with the database, or providing types for numbers, dates, strings, the role of Class Library gets into play. Providing a set of Application Interfaces, common functionality types, Class Library performs the job.

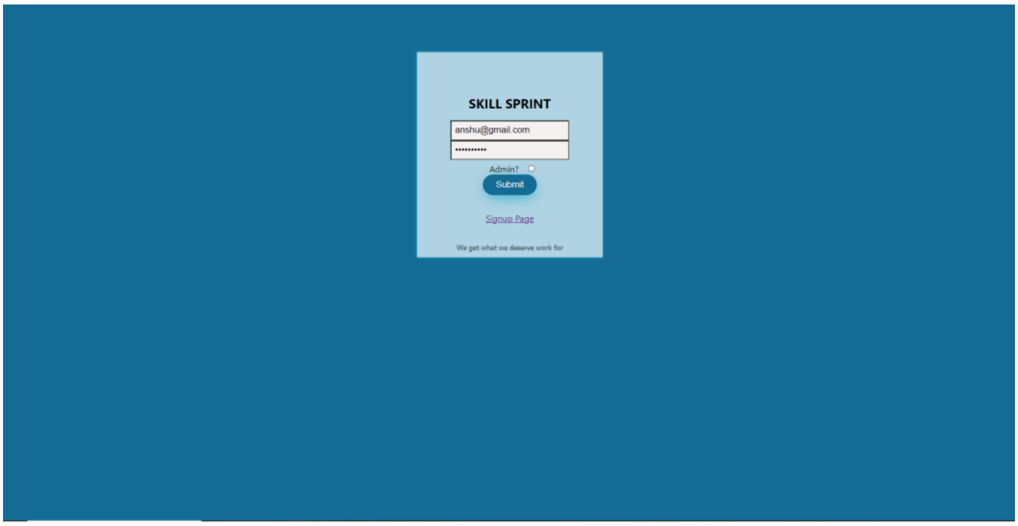
The programming languages used for writing the applications in .NET are C#, F# or Visual BASIC. The code compilation is into a language independent and agnostic intermediate language i.e, Common Intermediate Language (CIL) and which is further stored as .dll or
.exe files in assemblies.

At the time when the application starts to run, CLR turns the machine code for executing specific architecture with the help of JIT (Just-in-Time) compiler and in the meantime CLR takes the assemblies.

4. RESULT

4.1 Login Page

Login functionality is being achieved using an API from a Node.js server, upon entering the credentials, they will be validated; if they are valid, a token and a refresh token will be returned; otherwise, an error message will be returned, which will be shown to the users. If credentials are valid, they will be authenticated and redirected to the job application adding form if admin has logged in or else to all job applications page if a normal user has logged in.



SKILL SPRINT

anshu@gmail.com

••••••••••

Admin? □

Figure 21 : Login Page

The submit button sends a request to the database to verify the user credentials and allow the user to login if the credentials match, or else display an alert pop-up showing the user is not registered or has entered the wrong password.

The user can click on the Admin checkbox for admin login or leave it unchecked for normal user login.

The credentials are matched with the admin database if the Admin checkbox is checked, or they are matched with the user database if the credentials don’t match.

4.2 Signup page

Signup functionality is being achieved using an API from a Node.js server, upon entering the credentials, they will be validated; if they are valid, a token and a refresh token will be returned; otherwise, an error message will be returned, which will be shown to the users. If the credentials are valid, they will be encrypted and stored in the database, and then the user will be redirected to the login page.

The user can click on the Admin checkbox for admin sign up or leave it unchecked for normal user sign up.

The data is stored in the admin database if the Admin checkbox is checked or in the user database if not checked.

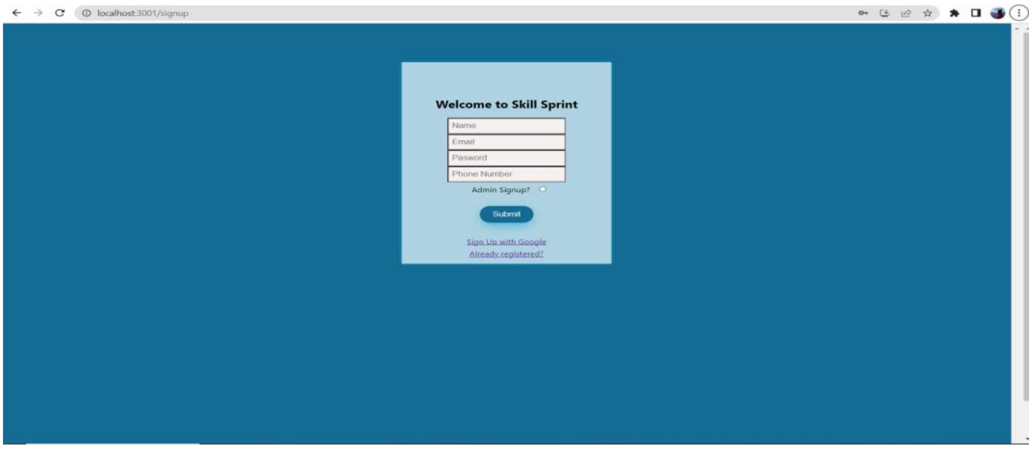
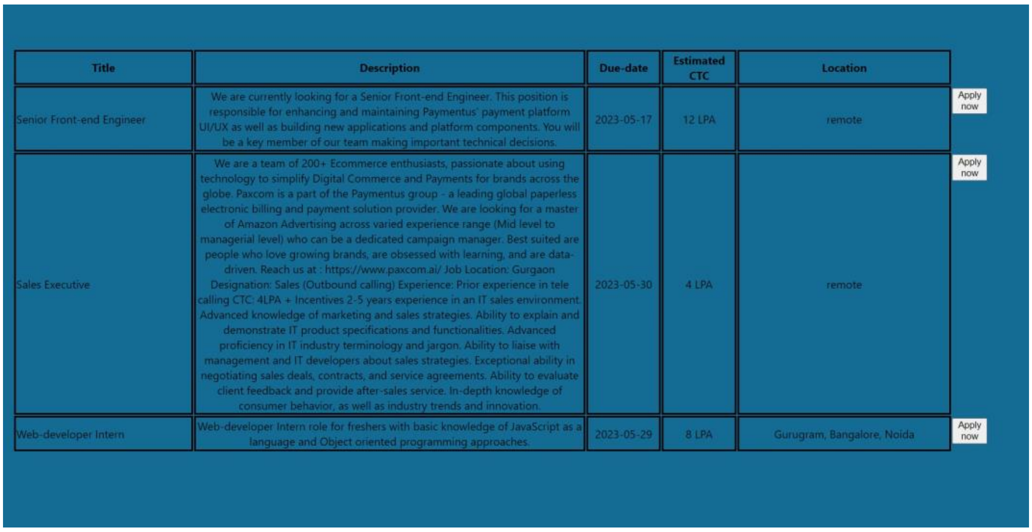


Figure 22: Signup Page

4.3 Job Applications Page

All the job openings added by the admin are displayed on this page.

When a normal user logs in he is redirected to this page where he can view various job openings and apply for it.

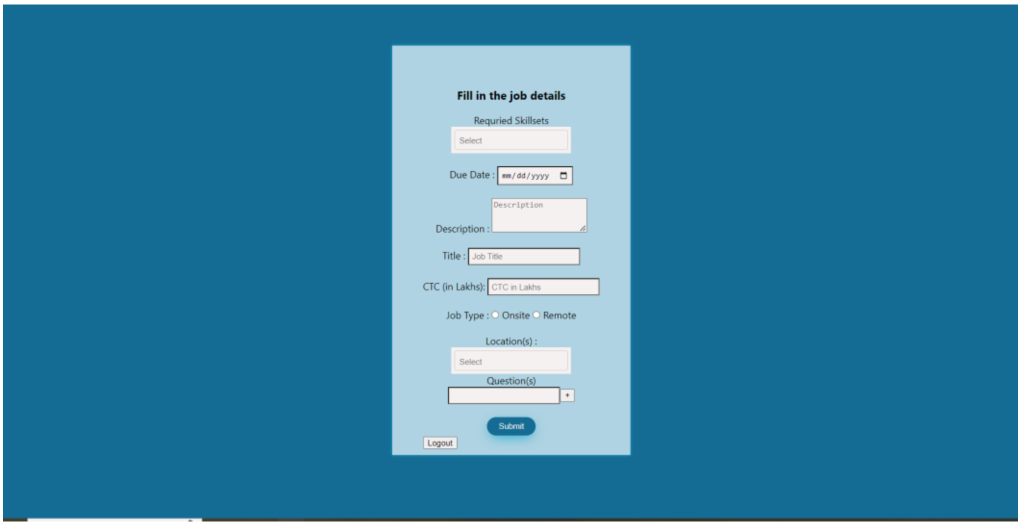


Title	Description	Due-date	Estimated CTC	Location	
Senior Front-end Engineer	We are currently looking for a Senior Front-end Engineer. This position is responsible for enhancing and maintaining Paymentus' payment platform UI/UX as well as building new applications and platform components. You will be a key member of our team making important technical decisions.	2023-05-17	12 LPA	remote	Apply now
Sales Executive	We are a team of 200+ Ecommerce enthusiasts, passionate about using technology to simplify Digital Commerce and Payments for brands across the globe. Paxcom is a part of the Paymentus group - a leading global paperless electronic billing and payment solution provider. We are looking for a master of Amazon Advertising across varied experience range (Mid level to managerial level) who can be a dedicated campaign manager. Best suited are people who love growing brands, are obsessed with learning, and are data-driven. Reach us at : https://www.paxcom.ai/ Job Location: Gurgaon Designation: Sales (Outbound calling) Experience: Prior experience in tele calling CTC: 4LPA + Incentives 2-5 years experience in an IT sales environment. Advanced knowledge of marketing and sales strategies. Ability to explain and demonstrate IT product specifications and functionalities. Advanced proficiency in IT industry terminology and jargon. Ability to liaise with management and IT developers about sales strategies. Exceptional ability in negotiating sales deals, contracts, and service agreements. Ability to evaluate client feedback and provide after-sales service. In-depth knowledge of consumer behavior, as well as industry trends and innovation.	2023-05-30	4 LPA	remote	Apply now
Web-developer Intern	Web-developer Intern role for freshers with basic knowledge of JavaScript as a language and Object oriented programming approaches.	2023-05-29	8 LPA	Gurugram, Bangalore, Noida	Apply now

Figure 23: Job Applications Page

4.4 Job Registration Form

The administrator can add a new job opening with the help of this form. All the required skill sets and conditions for any job application can be filled by the administrator. The admin can also add any number of unique questions related to that job.



Fill in the job details

Requried Skillsets

Select

Due Date : mm/dd/yyyy

Description : Description

Title : Job Title

CTC (in Lakhs): CTC in Lakhs

Job Type : ○ Onsite ○ Remote

Location(s) :

Select

Question(s)

Figure 24: Job Registration Form

4.5 Questions Form

The admin, while posting a new job, can add any number of custom questions with the help of this feature. This is done to improve individuality of all the job openings posted by the admin and to give more customization options for the admin.

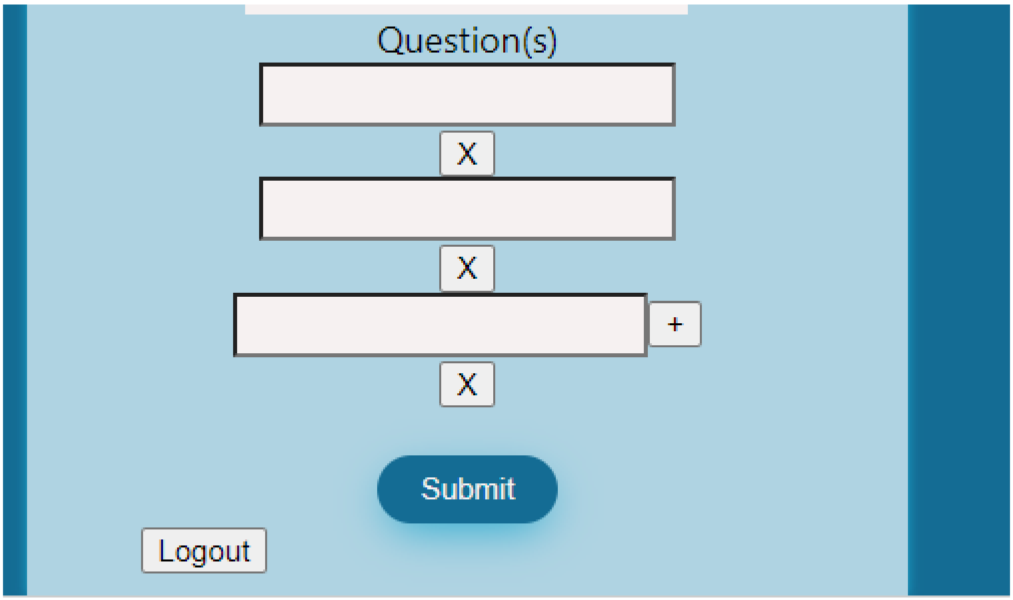


Figure 25: Questions Form

4 Conclusion

5.1 Conclusions

The training's primary goal was to enable participants to comprehend and put ReactJS, MongoDB, and ExpressJS principles into practice, as well as to build a web application that could carry out CRUD activities and be tested using Postman utilizing the three-layered design. These were the objectives I was able to accomplish with this project. My internship experience here showed me that developers should think about how to improve the user experience as well as the code itself. To make it easier for visitors to traverse the website, the interface should be consistent and fluid. In addition, producing clear and understandable code makes it simpler for other developers to comprehend it and locate any issues that may exist. In the future, we should aim to build reusable code. There is always room for improvement, and we should work to take note of the comments made.

5.2 Future Work

1. Adding ‘Activity Log’ page for admin CRUD operations.
2. Adding ‘Questions’ option for separate job application forms.
3. Adding a pop-up window on clicking the ‘Apply now’ button in front of all the job applications which will lead to the related questions and answers form.
4. Adding an ‘Edit Profile’ page for all users.
5. Providing third-party authentication for better security.

References

[1] Sourabh Mahadev Malewade, Archana Ekbote “Performance Optimization using MERN Stack on Web Applications,”, publisher: IJRASET, vol. 10, doi : 6.06.2021, pp. 2278-0181.

[2] Yogesh Baiskar, Priyas Paulzagade, Krutik Koradia, Pramod Ingole, Dhiraj Shirbhate “MERN : A Full stack development”,publisher: IJRASET, vol. 1, doi : https://doi.org/10.22214/ijraset.2022.39982

[3] Aarti Singh, Ananya Anikesh “Web Development and Computer Science and Engineering”, publisher : IJRASET, Vol 1, doi : https://doi.org/10.53555/cse.v2i4.612

[4] Prakarsh Kaushik, Shashikant Suman, Basu Dev Shivahare, Vimal Bibhu “Web development and performance comparison of web development technologies in Nodejs and python”, publisher : ICTAI, doi : 10.1109/ictai53825.2021.9673464

[5] Pratiksha D. Dutonde “Website development technologies : A review”, publisher: IJRASET, vol. 10(1), doi: 10.22214/ijraset.2022.39839, pp. 359-366.