React:

import logo from './logo.svg';

import './App.css';

function **App**() {

  return (

    <div *className*="App">

      <header *className*="App-header">

        <img *src*={logo} *className*="App-logo" *alt*="logo" />

        <p>

          Edit <code>src/App.js</code> and save to reload.

        </p>

        <a

*className*="App-link"

*href*="https://reactjs.org"

*target*="\_blank"

*rel*="noopener noreferrer"

        >

          Learn React

        </a>

      </header>

    </div>

  );

}

export default **App**;

App.css

.App {

  text-align: center;

}

.App-logo {

  height: 40vmin;

  pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

  .App-logo {

    animation: App-logo-spin infinite 20s linear;

  }

}

.App-header {

  background-color: #282c34;

  min-height: 100vh;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  font-size: **calc**(10px + 2vmin);

  color: white;

}

.App-link {

  color: #61dafb;

}

@keyframes App-logo-spin {

  from {

    transform: **rotate**(0deg);

  }

  to {

    transform: **rotate**(360deg);

  }

}

-------------------------------------

<!DOCTYPE html>

<html> <head> <link rel="preconnect" href="https://fonts.googleapis.com"> <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin> <link href="https://fonts.googleapis.com/css2?family=Lora:ital,wght@0,400..700;1,400..700&display=swap" rel="stylesheet"> <link href="Project6.css" rel="stylesheet"> <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous"> <title>Website</title> </head> <body> <div class="container-fluid A"> <div> <nav class="navbar navbar-expand-lg "> <div class="container-fluid"> <a class="navbar-brand" href="#"><h1 class="display-5">Spybrick</h1></a> <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="[[#navbarSupportedContent](https://mattermost.spybrick.in/tech-tribe/messages/@nagadivya.y)](https://mattermost.spybrick.in/tech-tribe/messages/@sivaram#)" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation"> <span class="navbar-toggler-icon"></span>

2.

!DOCTYPE html>

<html> <head> <link rel="preconnect" href="https://fonts.googleapis.com"> <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin> <link href="https://fonts.googleapis.com/css2?family=Lora:ital,wght@0,400..700;1,400..700&display=swap" rel="stylesheet"> <link href="Project6.css" rel="stylesheet"> <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous"> <title>Website</title> </head> <body> <div class="container-fluid A"> <div> <nav class="navbar navbar-expand-lg "> <div class="container-fluid"> <a class="navbar-brand" href="#"><h1 class="display-5">Spybrick</h1></a> <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="[#navbarSupportedContent](https://mattermost.spybrick.in/tech-tribe/messages/@sivaram#)" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation"> <span class="navbar-toggler-icon"></span> </button> <div class="collapse navbar-collapse" id="navbarSupportedContent"> <ul class="navbar-nav me-auto mb-2 mb-lg-0"> <li class="nav-item"> <a class="nav-link" aria-current="page" href="#">Home</a> </li> <li class="nav-item"> <a class="nav-link" href="ProjectNav.html">Solution</a> </li> <li class="nav-item"> <a class="nav-link" href="#">Capabilities</a> </li> <li class="nav-item"> <a class="nav-link" href="#">Insights</a> </li> <li class="nav-item dropdown"> <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false"> Dropdown </a> <ul class="dropdown-menu" aria-labelledby="navbarDropdown"> <li><a class="dropdown-item" href="#">Action</a></li> <li><a class="dropdown-item" href="#">Another action</a></li> <li><hr class="dropdown-divider"></li> <li><a class="dropdown-item" href="#">Something else here</a></li> </ul> </li> <li class="nav-item"> <a class="nav-link " href="#" >Product</a> </li> </ul> <button type="button" class="btn btn-lg">Connect</button> </div> </div> </nav> </div> <div class="container B"> <div class="row"> <div class="col"> <h1 class="display-2">Revamp your digital presence with user-centric design</h1> </div> <div class="col B1"> <form> <div class="mb-3"> <label for="exampleInputEmail1" class="form-label">Email address</label> <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" style="width: 400px;"> <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div> </div> <div class="mb-3"> <label for="exampleInputPassword1" class="form-label">Password</label> <input type="password" class="form-control" id="exampleInputPassword1" style="width: 400px;"> </div> <div class="mb-3 form-check"> <input type="checkbox" class="form-check-input" id="exampleCheck1"> <label class="form-check-label" for="exampleCheck1">Check me out</label> </div> <button type="submit" class="btn btn-lg">Submit</button> </form> </div> </div> </div> <div class="container C"> <div class="row"> <div class="col"> <h1 class="display-4">Unlocking the Potential of AI: Innovations and Applications</h1> <button type="submit" class="btn btn-lg">Connect</button> </div> <div class="col"> <p style="font-size: 1.25rem;">In recent years, the field of artificial intelligence (AI) has witnessed unprecedented growth, unlocking vast potential across various industries and applications. From healthcare and finance to manufacturing and transportation, AI technologies are revolutionizing how we work, live, and interact with the world around us. Innovations in machine learning, natural language processing, and computer vision have enabled AI systems to perform tasks once thought impossible, driving efficiency, accuracy, and scalability to new heights.</p> <p style="font-size: 1.25rem;">As AI technologies continue to advance, the possibilities are endless. From autonomous vehicles and smart cities to personalized virtual assistants and predictive analytics, the potential of AI innovations and applications is vast and far-reaching. By harnessing the power of AI, we can address complex challenges, drive sustainable development, and create a brighter future for generations to come.</p> </div> </div> </div> <div class="container-fluid D"> <div class="row"> <div class="col D1"> <h1 class="h1">The Future is Now: Exploring AI Technology Advancements</h1> <button type="submit" class="btn btn-lg">Read More</button> </div> <div class="col D2"> <h1 class="h1">AI in Action: Real-world Examples of Cutting-edge Technology</h1> <button type="submit" class="btn btn-lg">Read More</button> </div> </div> </div> <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js" integrity="sha384-..." crossorigin="anonymous"></script> <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js" integrity="sha384-..." crossorigin="anonymous"></script> </div>

</body> </html>

Show less

---------

function Carousel() {

return (

<div className="container1">

{/\* Carousel \*/}

<div className="container text-center">

<div className="row">

<div className="col">

<h2>Inspiration for home interior designs</h2>

{/\* <h2><a href="/interior-design-inspiration">Inspiration for home interior designs</a></h2>\*/}

<p>

Give your home a new look with these interior design ideas curated for you

{/\* <a href="/interior-design-ideas">interior design ideas</a>\*/}

</p>

</div>

</div>

</div>

<div id="carouselExampleControls" className="carousel slide" data-ride="carousel">

<div className="carousel-inner">

<div className="carousel-item active">

<div className="row">

<div className="col-md-4 custom-column">

<img src={livingroomimage1} alt="Living Room" />

<h2>Living Room</h2>

</div>

<div className="col-md-4 custom-column">

<img src={masterbedroom2} alt="Master Bedroom" />

<h2>Master Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={falseceiling3} alt="False Ceiling" />

<h2>False Ceiling</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homesbylivspace4} alt="Homes by Livspace" />

<h2>Homes by Livspace</h2>

</div>

<div className="col-md-4 custom-column">

<img src={kitchen5} alt="Kitchen" />

<h2>Kitchen</h2>

</div>

<div className="col-md-4 custom-column">

<img src={wardrobe6} alt="Wardrobe" />

<h2>Wardrobe</h2>

</div>

</div>

</div>

{/\* Second slide \*/}

<div className="carousel-item">

<div className="row">

<div className="col-md-4 custom-column">

<img src={kids7} alt="Kids Bedroom" />

<h2>Kids Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homeoffice8} alt="Home Office" />

<h2>Home Office</h2>

</div>

<div className="col-md-4 custom-column">

<img src={guestbedroom9} alt="Guest Bedroom" />

<h2>Guest Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={foyer10} alt="Foyer" />

<h2>Foyer</h2>

</div>

<div className="col-md-4 custom-column">

<img src={dining11} alt="Dining" />

<h2>Dining</h2>

</div>

<div className="col-md-4 custom-column">

<img src={bathroom12} alt="Bathroom" />

<h2>Bathroom</h2>

</div>

</div>

</div>

</div>

<a className="carousel-control-prev" href="#carouselExampleControls" role="button" data-slide="prev">

<span className="carousel-control-prev-icon" aria-hidden="true"></span>

<span className="sr-only">Previous</span>

</a>

<a className="carousel-control-next" href="#carouselExampleControls" role="button" data-slide="next">

<span className="carousel-control-next-icon" aria-hidden="true"></span>

<span className="sr-only">Next</span>

</a>

</div>

</div>

);

}

2.

function Carousel() {

return (

<div className="container1">

{/\* Carousel \*/}

<div className="container text-center">

<div className="row">

<div className="col">

<h2>Inspiration for home interior designs</h2>

<p>

Give your home a new look with these interior design ideas curated for you

</p>

</div>

</div>

</div>

<div id="carouselExampleControls" className="carousel slide" data-ride="1000">

<div className="carousel-inner">

<div className="carousel-item active">

<div className="row">

<div className="col-md-4 custom-column">

<img src={livingroomimage1} alt="Living Room" />

<h2>Living Room</h2>

</div>

<div className="col-md-4 custom-column">

<img src={masterbedroom2} alt="Master Bedroom" />

<h2>Master Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={falseceiling3} alt="False Ceiling" />

<h2>False Ceiling</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homesbylivspace4} alt="Homes by Livspace" />

<h2>Homes by Livspace</h2>

</div>

<div className="col-md-4 custom-column">

<img src={kitchen5} alt="Kitchen" />

<h2>Kitchen</h2>

</div>

<div className="col-md-4 custom-column">

<img src={wardrobe6} alt="Wardrobe" />

<h2>Wardrobe</h2>

</div>

</div>

</div>

{/\* Second slide \*/}

<div className="container1">

{/\* Carousel \*/}

<div className="container text-center">

<div className="carousel-inner">

<div className="carousel-item ">

<div className="row">

<div className="col-md-4 custom-column">

<img src={kids7} alt="Kids Bedroom" />

<h2>Kids Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homeoffice8} alt="Home Office" />

<h2>Home Office</h2>

</div>

<div className="col-md-4 custom-column">

<img src={guestbedroom9} alt="Guest Bedroom" />

<h2>Guest Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={foyer10} alt="Foyer" />

<h2>Foyer</h2>

</div>

<div className="col-md-4 custom-column">

<img src={dining11} alt="Dining" />

<h2>Dining</h2>

</div>

<div className="col-md-4 custom-column">

<img src={bathroom12} alt="Bathroom" />

<h2>Bathroom</h2>

</div>

</div>

</div>

</div>

</div>

</div>

<a className="carousel-control-prev" href="#carouselExampleControls" role="button" data-slide="prev">

<span className="carousel-control-prev-icon" aria-hidden="true"></span>

<span className="sr-only">Previous</span>

</a>

<a className="carousel-control-next" href="#carouselExampleControls" role="button" data-slide="next">

<span className="carousel-control-next-icon" aria-hidden="true"></span>

<span className="sr-only">Next</span>

</a>

</div>

</div>

</div>

);

}

export default Carousel;

3

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Horizontal Scrolling Example</title>

<style>

.container {

width: 100%;

overflow-x: auto;

white-space: nowrap;

}

.item {

display: inline-block;

width: 200px;

height: 200px;

margin-right: 10px;

background-color: #ccc;

}

/\* Optional: Style the images \*/

.item img {

width: 100%;

height: 100%;

object-fit: cover;

}

</style>

</head>

<body>

<div class="container">

<div class="item">

<img src="https://via.placeholder.com/200x200" alt="Image 1">

</div>

<div class="item">

<img src="https://via.placeholder.com/200x200" alt="Image 2">

</div>

<div class="item">

<img src="https://via.placeholder.com/200x200" alt="Image 3">

</div>

<div class="item">

<img src="https://via.placeholder.com/200x200" alt="Image 4">

</div>

<div class="item">

<img src="https://via.placeholder.com/200x200" alt="Image 5">

</div>

<!-- Add more items here if needed -->

</div>

</body>

</html>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

<script>

// Activate carousel and set interval for automatic sliding

$('.carousel').carousel({

interval: 1000 // Adjust as needed (in milliseconds)

});

</script>

<!-- Bootstrap JS -->

</body>

</html>

4.

/\*\*\*\*\*/

.image-container img {

display: block;

width: 100%; /\* Ensure the image fills its container \*/

height: auto; /\* Maintain aspect ratio \*/

}

.centered-content {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

text-align: center; /\* Center align the content \*/

}

.centered-content h2 {

margin-bottom: 20px; /\* Adjust spacing between the image and the heading \*/

}

.centered-button {

background-color: #007bff; /\* Button background color \*/

color: rgb(150, 41, 41); /\* Button text color \*/

border: none; /\* Remove button border \*/

border-radius: 5px; /\* Add border radius for button \*/

padding: 10px 20px; /\* Adjust button padding as needed \*/

font-size: 16px; /\* Set font size as needed \*/

cursor: pointer; /\* Set cursor style \*/

}

.centered-button {

background-color: hsl(348, 85%, 52%); /\* Your custom color \*/

color: white; /\* Text color \*/

border-radius: 200px;

transition: background-color 0.3s; /\* Add transition for smooth color change \*/

}

.social-links {

list-style: none;

padding: 0;

}

.social-links li {

display: inline-block; /\* Display list items horizontally \*/

margin-right: 10px; /\* Add some spacing between the social icons \*/

}

.social-links li:last-child {

margin-right: 0; /\* Remove margin from the last list item \*/

}

footer {

background-color: #dad8de; /\* Set the background color for the footer \*/

padding: 20px 0; /\* Add padding for better spacing \*/

text

}

.container1 {

width: 80%; /\* Adjust as needed \*/

max-width: 1200px; /\* Adjust as needed \*/

margin: auto; /\* Center the container horizontally \*/

}

.container1 .row {

display: flex;

justify-content: space-between;

}

.col {

flex: 1;

text-align: center;

}

.col img {

display: block;

margin: 0 auto 20px; /\* Add margin to center the image \*/

}

.social-links {

list-style: none;

padding: 0;

margin-top: 10px;

}

.social-links li {

display: inline-block;

margin-right: 10px;

}

.social-links li:last-child {

margin-right: 0;

}

/\* Change color on hover \*/

.social-links li a {

color: #fefafa; /\* Set default color \*/

transition: color 0.3s ease; /\* Add transition effect \*/

}

.social-links li a:hover {

color: #007bff; /\* Change color on hover \*/

}

.image-container {

position: relative;

width: 100%; /\* Adjust this according to your layout \*/

height: 300px; /\* Adjust this according to your layout \*/

}

.image-container img {

width: 100%;

height: 100%;

object-fit: cover;

}

.centered-content {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

text-align: center;

color: white; /\* Adjust text color according to your needs \*/

}

.centered-content h2 {

font-size: 24px; /\* Adjust font size according to your needs \*/

margin-bottom: 10px;

}

.centered-button {

padding: 10px 20px;

background-color: #007bff; /\* Adjust button background color according to your needs \*/

color: white; /\* Adjust button text color according to your needs \*/

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px; /\* Adjust button font size according to your needs \*/

}

Cards2.css

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: Arial, sans-serif;

}

h2{

text-align: left;

font-weight: bold;

text-align: center;

}

/\*\*/

.container1 .carousel-item img {

width: 100%;

height: 300px; /\* Adjust image height \*/

object-fit: cover;

border-radius: 10px;

background-origin: padding-box;

transform: translateX(5px); /\* Adjust as needed \*/

transition: transform 0.2s ease;

}

.carousel-item1 h2 {

position: absolute;

bottom: 10px;

right: 50%;

/\* text background colour\*/

background-color: rgba(0, 0, 0, 0.5);

color: white;

padding: 5px 10px;

border-radius: 5px;

}

p {

text-align: left;

font-weight: bold;

}

.container .carousel-item img {

width: 100%;

height: 300px; /\* Adjust image height \*/

object-fit: cover;

border-radius: 10px;

background-origin: padding-box;

}

.carousel-item h2 {

position: absolute;

bottom: 10px;

left: 50%;

transform: translateX(-50%);

background-color: rgba(0, 0, 0, 0.5);

color: white;

padding: 5px 10px;

border-radius: 5px;

}

.image-container {

position: relative;

display: inline-block;

display: flex;

}

.image-container img {

display: block;

width: 100%; /\* Ensure the image fills its container \*/

height: auto; /\* Maintain aspect ratio \*/

}

.centered-content {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

text-align: center; /\* Center align the content \*/

}

.centered-button {

background-color: #007bff; /\* Button background color \*/

color: white; /\* Button text color \*/

border: none; /\* Remove button border \*/

border-radius: 5px; /\* Add border radius for button \*/

padding: 10px 20px; /\* Adjust button padding as needed \*/

font-size: 16px; /\* Set font size as needed \*/

cursor: pointer; /\* Set cursor style \*/

}

.centered-button {

background-color: hsl(348, 85%, 52%); /\* Your custom color \*/

color: white; /\* Text color \*/

border-radius: 200px;

transition: background-color 0.3s; /\* Add transition for smooth color change \*/

}

<div className="container1">

<div className="container text-center">

<div className="row">

<div className="col">

<h2>Inspiration for home interior designs</h2>

<p>

Give your home a new look with these interior design ideas curated for you

</p>

</div>

</div>

</div>

<div id="carouselExampleControls" className="carousel slide" data-ride="1000">

<div className="carousel-inner">

<div className="carousel-item active">

<div className="row">

<div className="col-md-4 custom-column">

<img src={livingroomimage1} alt="Living Room" />

<h2>Living Room</h2>

</div>

<div className="col-md-4 custom-column">

<img src={masterbedroom2} alt="Master Bedroom" />

<h2>Master Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={falseceiling3} alt="False Ceiling" />

<h2>False Ceiling</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homesbylivspace4} alt="Homes by Livspace" />

<h2>Homes by Livspace</h2>

</div>

<div className="col-md-4 custom-column">

<img src={kitchen5} alt="Kitchen" />

<h2>Kitchen</h2>

</div>

<div className="col-md-4 custom-column">

<img src={wardrobe6} alt="Wardrobe" />

<h2>Wardrobe</h2>

</div>

</div>

</div>

{/\* Second slide \*/}

<div className="container1">

<div className="container text-center">

<div className="carousel-inner">

<div className="carousel-item ">

<div className="row">

<div className="col-md-4 custom-column">

<img src={kids7} alt="Kids Bedroom" />

<h2>Kids Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homeoffice8} alt="Home Office" />

<h2>Home Office</h2>

</div>

<div className="col-md-4 custom-column">

<img src={guestbedroom9} alt="Guest Bedroom" />

<h2>Guest Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={foyer10} alt="Foyer" />

<h2>Foyer</h2>

</div>

<div className="col-md-4 custom-column">

<img src={dining11} alt="Dining" />

<h2>Dining</h2>

</div>

<div className="col-md-4 custom-column">

<img src={bathroom12} alt="Bathroom" />

<h2>Bathroom</h2>

</div>

</div>

</div>

</div>

</div>

</div>

<a className="carousel-control-prev" href="#carouselExampleControls" role="button" data-slide="prev">

<span className="carousel-control-prev-icon" aria-hidden="true"></span>

<span className="sr-only">Previous</span>

</a>

<a className="carousel-control-next" href="#carouselExampleControls" role="button" data-slide="next">

<span className="carousel-control-next-icon" aria-hidden="true"></span>

<span className="sr-only">Next</span>

</a>

</div>

</div>

</div>

Cards2.jsx

<div className="container1">

<div className="container text-center">

<div className="row">

<div className="col">

<h2>Inspiration for home interior designs</h2>

<p>

Give your home a new look with these interior design ideas curated for you

</p>

</div>

</div>

</div>

<div id="carouselExampleControls" className="carousel slide" data-ride="carousel">

<div className="carousel-inner">

<div className="carousel-item active">

<div className="row">

<div className="col-md-4 custom-column">

<img src={livingroomimage1} alt="Living Room" />

<h2>Living Room</h2>

</div>

<div className="col-md-4 custom-column">

<img src={masterbedroom2} alt="Master Bedroom" />

<h2>Master Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={falseceiling3} alt="False Ceiling" />

<h2>False Ceiling</h2>

</div>

</div>

</div>

<div className="carousel-item">

<div className="row">

<div className="col-md-4 custom-column">

<img src={homesbylivspace4} alt="Homes by Livspace" />

<h2>Homes by Livspace</h2>

</div>

<div className="col-md-4 custom-column">

<img src={kitchen5} alt="Kitchen" />

<h2>Kitchen</h2>

</div>

<div className="col-md-4 custom-column">

<img src={wardrobe6} alt="Wardrobe" />

<h2>Wardrobe</h2>

</div>

</div>

</div>

<div className="carousel-item">

<div className="row">

<div className="col-md-4 custom-column">

<img src={kids7} alt="Kids Bedroom" />

<h2>Kids Bedroom</h2>

</div>

<div className="col-md-4 custom-column">

<img src={homeoffice8} alt="Home Office" />

<h2>Home Office</h2>

</div>

<div className="col-md-4 custom-column">

<img src={guestbedroom9} alt="Guest Bedroom" />

<h2>Guest Bedroom</h2>

</div>

</div>

</div>

<div className="carousel-item">

<div className="row">

<div className="col-md-4 custom-column">

<img src={foyer10} alt="Foyer" />

<h2>Foyer</h2>

</div>

<div className="col-md-4 custom-column">

<img src={dining11} alt="Dining" />

<h2>Dining</h2>

</div>

<div className="col-md-4 custom-column">

<img src={bathroom12} alt="Bathroom" />

<h2>Bathroom</h2>

</div>

</div>

</div>

</div>

<a className="carousel-control-prev" href="#carouselExampleControls" role="button" data-slide="prev">

<span className="carousel-control-prev-icon" aria-hidden="true"></span>

<span className="sr-only">Previous</span>

</a>

<a className="carousel-control-next" href="#carouselExampleControls" role="button" data-slide="next">

<span className="carousel-control-next-icon" aria-hidden="true"></span>

<span className="sr-only">Next</span>

</a>

</div>

</div>

<Router>

<Routes>

<Route path='/' element={<Navbar/>}/>

<Route path='/Home' element={<Home/>}/>

<Route path='/Link' element={<Link/>}/>

<Route path='/Action' element={<Action/>}/>

</Routes>

</Router>

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

npm install react-router-dom

Mysql root password for local host :

Bha@1999

<div className="container">

<div className="row justify-content-center mt-5">

<div className="col-md-6">

<div className="card">

<div className="card-body">

<h3 className="card-title text-center mb-4">Login</h3>

<form onSubmit={handleSubmit}>

<div className="form-group">

<label htmlFor="email">Email address</label>

<input type="email" className="form-control" id="email" placeholder="Enter email" value={email} onChange={handleEmailChange} />

</div>

<div className="form-group">

<label htmlFor="password">Password</label>

<input type="password" className="form-control" id="password" placeholder="Password" value={password} onChange={handlePasswordChange} />

</div>

<button type="submit" className="btn btn-primary btn-block">Login</button>

</form>

</div>

</div>

</div>

</div>

</div>

App.js

import React from 'react';

import Navbar from './Components/Navbar1.jsx';

import SecondNavbar from './Components/SecondNavbar.jsx';

import Slides from './Components/Slides.jsx'

import CardComponent from './Components/Cards.jsx';

import { Carousel, ImageContainer } from './Components/cards2.jsx';

import Footer from './Components/Footer.jsx';

function App() {

return (

<div>

<Navbar />

<SecondNavbar />

<Slides />

<CardComponent />

<Carousel />

<ImageContainer />

<Footer />

</div>

);

}

………………………………………………………………….

102053024545

PS C:\Users\surya\OneDrive\Desktop\ReactJS> cd server

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm init

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields

and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (server)

version: (1.0.0)

description:

entry point: (index.js)

test command:

git repository:

keywords:

author:

license: (ISC)

About to write to C:\Users\surya\OneDrive\Desktop\ReactJS\server\package.json:

{

"name": "server",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC"

}

npm uninstall express

npm uninstall express --save

Is this OK? (yes) yes

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install express

added 64 packages, and audited 65 packages in 7s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install mysql

added 12 packages, and audited 77 packages in 4s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install cors

added 2 packages, and audited 79 packages in 2s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server>

PS C:\Users\surya\OneDrive\Desktop\ReactJS> cd server

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm init

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields

and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (server)

version: (1.0.0)

description:

entry point: (index.js)

test command:

git repository:

keywords:

author:

license: (ISC)

About to write to C:\Users\surya\OneDrive\Desktop\ReactJS\server\package.json:

{

"name": "server",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC"

}

npm uninstall express

npm uninstall express --save

Is this OK? (yes) yes

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install express

added 64 packages, and audited 65 packages in 7s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install mysql

added 12 packages, and audited 77 packages in 4s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server> npm install cors

added 2 packages, and audited 79 packages in 2s

12 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\surya\OneDrive\Desktop\ReactJS\server>

Client:

1.signup.js

import React, { useState } from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from '../components/Navbar';

import './Signupcss.css';

export default function Signup() {

const navigate = useNavigate();

const [name, setName] = useState('');

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [mobile, setMobile] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

const [userType, setUserType] = useState('recruiter');

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[@$!%\*?&])[A-Za-z\d@$!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

// Sanitizing user inputs

const handleInputChange = (setter) => (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setter(sanitizedValue);

};

const handleErrorMessage = () => {

setErrorMessage('');

};

// Email verification

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

} catch (error) {

console.error(error);

}

};

const handleSubmit = async (event) => {

event.preventDefault();

const sanitizedName = DOMPurify.sanitize(name);

const sanitizedEmail = DOMPurify.sanitize(email);

const sanitizedMobile = DOMPurify.sanitize(mobile);

const sanitizedPassword = DOMPurify.sanitize(password);

const sanitizedUserType = DOMPurify.sanitize(userType);

if (!validateEmail(email)) {

setEmailError('Please enter a valid email address');

} else {

setEmailError('');

}

if (!checkPasswordStrength(password)) {

setPasswordError('Password must be strong');

} else {

setPasswordError('');

}

if (!validateMobile(mobile)) {

setMobileError('Please enter a valid mobile number');

} else {

setMobileError('');

}

if (validateEmail(email) && checkPasswordStrength(password) && validateMobile(mobile)) {

const salt = CryptoJS.lib.WordArray.random(16); // Generate a random 16-byte salt

const saltedPassword = salt + sanitizedPassword; // Add the salt to the password

const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString();

try {

const response = await axios.post('http://localhost:5000/create', {

name: sanitizedName,

email: sanitizedEmail,

mobile: sanitizedMobile,

password: encryptedPassword,

userType: sanitizedUserType,

salt: salt.toString(),

});

console.log(response.data);

sendVerificationEmail(sanitizedEmail);

window.alert('Please check your email for verification.');

navigate('/AddDetails');

} catch (error) {

console.error(error);

if (error.response && error.response.status === 400) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

} else if (error.response && error.response.status === 500) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

}

}

}

};

return (

<div>

<div className="container-fluid n">

<Navbar />

</div>

<div className="container-fluid h1">

<h2>Sign Up Form</h2>

</div>

<div className="container-fluid s1">

<form onSubmit={handleSubmit}>

<div className="input-container">

<i className="fa fa-user icon"></i>

<input

className="input-field"

type="text"

value={name}

placeholder="Username"

name="name"

onChange={handleInputChange(setName)}

/>

</div>

<div className="input-container">

<i className="fa fa-envelope icon"></i>

<input

className="input-field"

type="text"

value={email}

placeholder="Email"

name="email"

onChange={handleInputChange(setEmail)}

/>

</div>

<div className="e">{emailError && <span style={{ color: 'red' }}>{emailError}</span>}</div>

<div className="input-container">

<i className="fa fa-mobile icon"></i>

<input

className="input-field"

type="text"

value={mobile}

placeholder="Mobile"

name="mobile"

onChange={handleInputChange(setMobile)}

/>

</div>

<div className="e">{mobileError && <span style={{ color: 'red' }}>{mobileError}</span>}</div>

<div className="input-container">

<i className="fa fa-key icon"></i>

<input

className="input-field"

type="password"

value={password}

placeholder="Password"

name="password"

onChange={handleInputChange(setPassword)}

/>

</div>

<div className="e">{passwordError && <span style={{ color: 'red' }}>{passwordError}</span>}</div>

<br />

<div className="radioButtons">

<div className="form-check">

<input

className="form-check-input"

type="radio"

name="userType"

id="jobSeeker"

value="job seeker"

checked={userType === 'job seeker'}

onChange={handleInputChange(setUserType)}

/>

<label className="form-check-label" htmlFor="jobSeeker">

Job seeker

</label>

</div>

<div className="form-check">

<input

className="form-check-input"

type="radio"

name="userType"

id="recruiter"

value="recruiter"

checked={userType === 'recruiter'}

onChange={handleInputChange(setUserType)}

/>

<label className="form-check-label" htmlFor="recruiter">

Recruiter

</label>

</div>

</div>

<div className="container-fluid b1">

<button type="submit" className="h" onClick={handleErrorMessage}>

Sign Up

</button>

{errorMessage && <div style={{ color: 'red' }}>{errorMessage}</div>}

</div>

</form>

</div>

</div>

);

}

Server:

1.create.js

const express = require('express');

const router = express.Router();

const connection = require('./dbConnection');

let userId;

router.post('/', (req, res) => {

const { name, email, password, mobile, userType, salt } = req.body;

if (!name || !email || !password || !mobile || !userType || !salt) {

return res.status(400).json({ error: 'All fields are required' });

}

// Check if the email already exists in the database

const query = 'SELECT email FROM person WHERE LOWER(email) = LOWER(?)';

connection.query(query, [email], (error, results) => {

if (error) {

console.error('Error checking email existence:', error);

return res.status(500).json({ error: 'Internal server error' });

}

if (results.length > 0) {

return res.status(400).json({ error: 'Email already has an account' });

}

// Save the user's data to the database

const sql = 'INSERT INTO person (name, email, password, mobile, userType, salt) VALUES (?, ?, ?, ?, ?, ?)';

connection.query(sql, [name, email, password, mobile, userType, salt], (err, result) => {

if (err) {

console.error('Error saving user data:', err);

return res.status(500).json({ error: 'Error saving user data' });

}

userId = result.insertId;

console.log('User data saved successfully:', result);

console.log('userId:', userId);

module.exports.variable = userId;

res.status(201).json({ message: 'User data saved successfully', userId });

});

});

});

module.exports = router;

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

import React,{useState} from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from "../components/Navbar"

import './Signupcss.css';

export default function Signup(){

const navigate = useNavigate();

const [name, setName] = useState('');

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [mobile, setMobile] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

const [userType, setUserType] = useState("recruiter");

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.[a-z])(?=.[A-Z])(?=.\d)(?=.[@

!

!!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

//sanitizing userInputs

const handleNameChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setName(sanitizedValue);

};

const handleEmailChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setEmail(sanitizedValue);

};

const handlePasswordChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setPassword(sanitizedValue);

};

const handleMobileChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setMobile(sanitizedValue);

};

const handleUserTypeChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setUserType(sanitizedValue);

};

const handleErrorMessage = () => {

setErrorMessage(" ");

}

//email verification

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

// show a success message to the user

} catch (error) {

console.error(error);

// show an error message to the user

}

}

const handleSubmit = (event) => {

event.preventDefault();

// Sanitize input before sending to server

const sanitizedName = DOMPurify.sanitize(name);

const sanitizedEmail = DOMPurify.sanitize(email);

const sanitizedMobile = DOMPurify.sanitize(mobile);

const sanitizedPassword = DOMPurify.sanitize(password);

const sanitizeduserType = DOMPurify.sanitize(userType);

1.client (signup)

import React,{useState} from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from "../components/Navbar"

import './Signupcss.css';

export default function Signup(){

const navigate = useNavigate();

const [name, setName] = useState('');

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [mobile, setMobile] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

const [userType, setUserType] = useState("recruiter");

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.[a-z])(?=.[A-Z])(?=.\d)(?=.[@

!

!!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

//sanitizing userInputs

const handleNameChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setName(sanitizedValue);

};

const handleEmailChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setEmail(sanitizedValue);

};

const handlePasswordChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setPassword(sanitizedValue);

};

const handleMobileChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setMobile(sanitizedValue);

};

const handleUserTypeChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setUserType(sanitizedValue);

};

const handleErrorMessage = () => {

setErrorMessage(" ");

}

//email verification

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

// show a success message to the user

} catch (error) {

console.error(error);

// show an error message to the user

}

}

const handleSubmit = (event) => {

event.preventDefault();

// Sanitize input before sending to server

const sanitizedName = DOMPurify.sanitize(name);

const sanitizedEmail = DOMPurify.sanitize(email);

const sanitizedMobile = DOMPurify.sanitize(mobile);

const sanitizedPassword = DOMPurify.sanitize(password);

const sanitizeduserType = DOMPurify.sanitize(userType);

-------------------------------------------

1.Index.js

const express = require('express');

const mysql = require('mysql');

const app= express();

const path = require('path')

const connection = mysql.createConnection({

host: 'localhost',

user: 'root',

password: 'Bj@1997',

database: 'employees'

});

connection.connect((err) => {

if (err) {

console.error('Error connecting to database:', err);

return;

}

console.log('Connected to database!');

});

app.use(express.static(path.join(\_\_dirname,'/client/build')));

app.post('/create',(req, res) => {

const FirstName= req.body.FirstName;

const sql = 'INSERT INTO user\_reg\_signin (FirstName) VALUES(?);'

connection.query(sql,[FirstName],(err, result) => {

if(err) {

console.error('Error saving user data:', err);

return res.status(500).send({error: 'Errro saving user data'});

}

res.send({message:'User data saved successfully'});

})

})

app.listen(5000, () => console.log('server runningon port 5000'));

2.signup.jsx

import React, { useState } from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from "./Navbar1";

import './signup.css';

export default function Signup() {

const navigate = useNavigate();

const [name, setName] = useState('');

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [mobile, setMobile] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[@!%\*?&])[A-Za-z\d@!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

// Sanitize user inputs

const handleNameChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setName(sanitizedValue);

};

const handleEmailChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setEmail(sanitizedValue);

};

const handlePasswordChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setPassword(sanitizedValue);

};

const handleMobileChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setMobile(sanitizedValue);

};

const handleErrorMessage = () => {

setErrorMessage(" ");

};

// Email verification

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

// Show a success message to the user

} catch (error) {

console.error(error);

// Show an error message to the user

}

};

const handleSubmit = (event) => {

event.preventDefault();

// Sanitize input before sending to server

const sanitizedName = DOMPurify.sanitize(name);

const sanitizedEmail = DOMPurify.sanitize(email);

const sanitizedMobile = DOMPurify.sanitize(mobile);

const sanitizedPassword = DOMPurify.sanitize(password);

if (!validateEmail(email)) {

setEmailError('Please enter a valid email address');

} else {

setEmailError('');

}

if (!checkPasswordStrength(password)) {

setPasswordError('Password must be strong');

} else {

setPasswordError('');

}

if (!validateMobile(mobile)) {

setMobileError('Please enter a valid mobile number');

} else {

setMobileError('');

}

if (validateEmail(email) && validateMobile(mobile) && checkPasswordStrength(password)) {

// Encrypt password

const salt = CryptoJS.lib.WordArray.random(16); // Generate a random 16-byte salt

const saltedPassword = salt + sanitizedPassword; // Add the salt to the password

const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString();

console.log('Salt:', salt.toString());

console.log('Hashed password:', encryptedPassword);

// Submit the form

axios.post('http://localhost:5000/create', {

name: sanitizedName,

email: sanitizedEmail,

mobile: sanitizedMobile,

password: encryptedPassword, // Send the encrypted password to the server

salt: salt.toString()

})

.then(response => {

console.log(response.data);

})

.catch(error => {

console.error(error);

if (error.response.status === 400) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

} else if (error.response.status === 500) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

}

});

sendVerificationEmail(sanitizedEmail); // Send the verification email

window.alert("Please check your email for verification.");

navigate('/AddDetails');

}

};

return (

<div>

<div className='container-fluid n'><Navbar /></div>

<div className='container-fluid h1'><h2>Sign Up Form</h2></div>

<div className='container-fluid s1'>

<form onSubmit={handleSubmit}>

<div className="input-container">

<i className="fa fa-user icon"></i>

<input className="input-field" type="text" value={name} placeholder="Username" name="name" onChange={handleNameChange} />

</div>

<div className="input-container">

<i className="fa fa-envelope icon"></i>

<input className="input-field" type="text" value={email} placeholder="Email" name="email" onChange={handleEmailChange} />

</div>

<div className='e'>{emailError && <span style={{ color: 'red' }}>{emailError}</span>}</div>

<div className="input-container">

<i className="fa fa-mobile icon"></i>

<input className="input-field" type="text" value={mobile} placeholder="Mobile" name="mobile" onChange={handleMobileChange} />

</div>

<div className='e'>{mobileError && <span style={{ color: 'red' }}>{mobileError}</span>}</div>

<div className="input-container">

<i className="fa fa-key icon"></i>

<input className="input-field" type="password" value={password} placeholder="Password" name="password" onChange={handlePasswordChange} />

</div>

<div className='e'> {passwordError && <span style={{ color: 'red' }}>{passwordError}</span>}</div>

<div className='container-fluid b1'><button type="submit" className="h" onClick={handleErrorMessage}>Sign Up</button>

{errorMessage && <div style={{ color: 'red' }}>{errorMessage}</div>}</div>

</form>

</div>

</div>

);

}

--

Signupcss.css

/\* signup.css \*/

.container-fluid {

width: 100%;

padding-right: 15px;

padding-left: 15px;

margin-right: auto;

margin-left: auto;

}

.n {

background-color: #f8f9fa;

}

.h1 {

text-align: center;

margin-top: 20px;

}

.s1 {

max-width: 600px;

margin: 0 auto;

padding: 20px;

background: #ffffff;

box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);

border-radius: 8px;

}

.input-container {

display: flex;

width: 100%;

margin-bottom: 15px;

position: relative;

}

.input-container .icon {

padding: 10px;

background: #007bff;

color: white;

min-width: 50px;

text-align: center;

border-top-left-radius: 4px;

border-bottom-left-radius: 4px;

}

.input-field {

width: 100%;

padding: 10px;

outline: none;

border: 1px solid #ccc;

border-left: none;

border-top-right-radius: 4px;

border-bottom-right-radius: 4px;

}

.input-field:focus {

border-color: #007bff;

}

.e {

margin-bottom: 15px;

}

.radioButtons {

margin-top: 20px;

display: flex;

justify-content: space-around;

}

.form-check {

display: flex;

align-items: center;

}

.form-check-input {

margin-right: 10px;

}

.b1 {

text-align: center;

margin-top: 20px;

}

button.h {

background-color: #007bff;

color: white;

padding: 10px 20px;

border: none;

border-radius: 4px;

cursor: pointer;

font-size: 16px;

}

button.h:hover {

background-color: #0056b3;

}

button.h:focus {

outline: none;

}

span {

display: block;

font-size: 14px;

margin-top: 5px;

}

-> Newly created (practice)

Signup.js

import React, { useState } from 'react';

import './signup.css'

function RegistrationForm() {

  const [username, setUsername] = useState('');

  const [email, setEmail] = useState('');

  const [phoneNumber, setPhoneNumber] = useState('');

  const [password, setPassword] = useState('');

  const [confirmPassword, setConfirmPassword] = useState('');

  const [isSubmitted, setIsSubmitted] = useState(false);

  const [errorMessage, setErrorMessage] = useState('');

  const handleChange = (event) => {

    const { name, value } = event.target;

    switch (name) {

      case 'username':

        setUsername(value);

        break;

      case 'email':

        setEmail(value);

        break;

      case 'phoneNumber':

        setPhoneNumber(value);

        break;

      case 'password':

        setPassword(value);

        break;

      case 'confirmPassword':

        setConfirmPassword(value);

        break;

      default:

        break;

    }

  };

  const handleSubmit = (event) => {

    event.preventDefault(); // Prevent default form submission

    if (password !== confirmPassword) {

      setErrorMessage('Passwords do not match!');

      return;

    }

    // Implement form submission logic here (e.g., API call, validation)

    setIsSubmitted(true); // For displaying success message or handling redirection

  };

  return (

    <div className="registration-form">

      <header>Signup</header>

      <form onSubmit={handleSubmit}>

        <input

          type="text"

          placeholder="Enter your username"

          name="username"

          value={username}

          onChange={handleChange}

          required

        />

        <input

          type="email"

          placeholder="Enter your email"

          name="email"

          value={email}

          onChange={handleChange}

          required

        />

        <input

          type="tel" // Use "tel" for phone number input

          placeholder="Enter your phone number"

          name="phoneNumber"

          value={phoneNumber}

          onChange={handleChange}

        />

        <input

          type="password"

          placeholder="Create a password"

          name="password"

          value={password}

          onChange={handleChange}

          required

        />

        <input

          type="password"

          placeholder="Confirm your password"

          name="confirmPassword"

          value={confirmPassword}

          onChange={handleChange}

          required

        />

        <input type="submit" value="Signup" className="button" />

      </form>

      {isSubmitted && <p className="success-message">Registration successful!</p>}

      {errorMessage && <p className="error-message">{errorMessage}</p>}

      <div className="signup">

        <span className="signup">

          Already have an account? <label htmlFor="check">Login</label>

        </span>

      </div>

    </div>

  );

}

export default RegistrationForm;

signupcss.css

.registration-form {

  max-width: 400px;

  margin: 50px auto;

  padding: 20px;

  border: 1px solid #ccc;

  border-radius: 10px;

  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

  font-family: Arial, sans-serif;

}

.registration-form header {

  font-size: 24px;

  font-weight: bold;

  text-align: center;

  margin-bottom: 20px;

}

.registration-form form {

  display: flex;

  flex-direction: column;

}

.registration-form input[type="text"],

.registration-form input[type="email"],

.registration-form input[type="tel"],

.registration-form input[type="password"] {

  padding: 10px;

  margin: 10px 0;

  border: 1px solid #ccc;

  border-radius: 5px;

  font-size: 16px;

}

.registration-form input[type="submit"] {

  padding: 10px;

  margin-top: 20px;

  background-color: #28a745;

  border: none;

  border-radius: 5px;

  color: white;

  font-size: 18px;

  cursor: pointer;

}

.registration-form input[type="submit"]:hover {

  background-color: #218838;

}

.success-message {

  color: #28a745;

  font-size: 16px;

  text-align: center;

  margin-top: 20px;

}

.error-message {

  color: #dc3545;

  font-size: 16px;

  text-align: center;

  margin-top: 20px;

}

.signup {

  text-align: center;

  margin-top: 20px;

}

.signup label {

  color:  #28a745;

  cursor: pointer;

  text-decoration: underline;

}

.signup label:hover {

  text-decoration: none;

}

node -v

npm install create-react-app

<Router>

<Routes>

<Route path='/' element={<Navbar/>}/>

<Route path='/Home' element={<Home/>}/>

<Route path='/Link' element={<Link/>}/>

<Route path='/Action' element={<Action/>}/>

</Routes>

</Router>

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

npm install react-router-dom

1.signup.jsx

import React,{useState} from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from "../components/Navbar"

import './Signupcss.css';

export default function Signup(){

const navigate = useNavigate();

const [name, setName] = useState('');

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [mobile, setMobile] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

const [userType, setUserType] = useState("recruiter");

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.[a-z])(?=.[A-Z])(?=.\d)(?=.[@

!

!!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

//sanitizing userInputs

const handleNameChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setName(sanitizedValue);

};

const handleEmailChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setEmail(sanitizedValue);

};

const handlePasswordChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setPassword(sanitizedValue);

};

const handleMobileChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setMobile(sanitizedValue);

};

const handleUserTypeChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setUserType(sanitizedValue);

};

const handleErrorMessage = () => {

setErrorMessage(" ");

}

//email verification

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

// show a success message to the user

} catch (error) {

console.error(error);

// show an error message to the user

}

}

const handleSubmit = (event) => {

event.preventDefault();

// Sanitize input before sending to server

const sanitizedName = DOMPurify.sanitize(name);

const sanitizedEmail = DOMPurify.sanitize(email);

const sanitizedMobile = DOMPurify.sanitize(mobile);

const sanitizedPassword = DOMPurify.sanitize(password);

const sanitizeduserType = DOMPurify.sanitize(userType);

if (!validateEmail(email)) {

setEmailError('Please enter a valid email address');

} else {

setEmailError('');

}

if (!checkPasswordStrength(password)) {

setPasswordError('Password must be Strong ');

} else {

setPasswordError('');

}

if (!validateMobile(mobile)) {

setMobileError('Please enter a valid mobile number');

} else {

setMobileError('');

}

if (!userType) {

alert('Please select a user type.');

return;

}

if (validateEmail(email) && validateMobile(mobile) && checkPasswordStrength(password)) {

// Encrypt password

const salt = CryptoJS.lib.WordArray.random(16); // Generate a random 16-byte salt

const saltedPassword = salt + sanitizedPassword; // Add the salt to the password

const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString();

console.log('Salt:', salt.toString());

console.log('Hashed password:', encryptedPassword);

// submit the form

axios.post('http://localhost:5000/create', {

name: sanitizedName,

email:sanitizedEmail,

mobile: sanitizedMobile,

password: encryptedPassword, // send the encrypted password to the server

userType:sanitizeduserType,

salt:salt.toString()

})

.then(response => {

console.log(response.data);

})

.catch(error => {

console.error(error);

if (error.response.status === 400) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

} else if (error.response.status === 500) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

}

}

);}

// perform form validation and submit

if (validateEmail(email) && validateMobile(mobile) && checkPasswordStrength(password)) {

sendVerificationEmail(sanitizedEmail); // send the verification email

window.alert("Please check your email for verification.");

navigate('/AddDetails');

}

};

return(

<div>

<div className='container-fluid n'><Navbar/></div>

<div className='container-fluid h1'><h2>Sign Up Form</h2></div>

<div className='container-fluid s1'>

<form onSubmit={handleSubmit}>

<div className="input-container">

<i className="fa fa-user icon"></i>

<input className="input-field" type="text" value={name} placeholder="Username" name="name" onChange={handleNameChange}/>

</div>

<div className="input-container">

<i className="fa fa-envelope icon"></i>

<input className="input-field" type="text" value={email} placeholder="Email" name="email" onChange={handleEmailChange}/>

</div>

<div className='e'>{emailError && <span style={{ color: 'red'}}>{emailError}</span>}</div>

<div className="input-container">

<i className="fa fa-mobile icon"></i>

<input className="input-field" type="text" value={mobile} placeholder="Mobile" name="mobile" onChange={handleMobileChange}/>

</div>

<div className='e'>{mobileError && <span style={{ color: 'red'}}>{mobileError}</span>}</div>

<div className="input-container">

<i className="fa fa-key icon"></i>

<input className="input-field" type="password" value={password} placeholder="Password" name="password" onChange={handlePasswordChange}/>

</div>

<div className='e'> {passwordError && <span style={{ color: 'red' }}>{passwordError}</span>}</div>

<br />

<div className="radioButtons">

<div className="form-check">

<input className="form-check-input" type="radio" name="userType" id="jobSeeker" value="job seeker" checked={userType === "job seeker"}

onChange={handleUserTypeChange}/>

<label className="form-check-label" htmlFor="flexRadioDefault1">

Job seeker

</label>

</div>

<div className="form-check">

<input className="form-check-input" type="radio" name="userType" id="recruiter" value="recruiter" checked={userType === "recruiter"}

onChange={handleUserTypeChange}/>

<label className="form-check-label" htmlFor="flexRadioDefault2">

Recruiter

</label>

</div>

</div>

<div className='container-fluid b1'><button type="submit" className="h" onClick={handleErrorMessage}>Sign Up</button>

{errorMessage && <div style={{ color : 'red' }}>{errorMessage}</div>}</div>

</form>

</div>

</div>

);

}

2.index.js

const express = require('express');

const router = express.Router();

const connection = require('./dbConnection');

let userId;

router.post('/', (req, res) => {

const name = req.body.name;

const email = req.body.email;

const password = req.body.password;

const mobile = req.body.mobile;

const userType = req.body.userType;

const salt = req.body.salt;

// Check if the email already exists in the database

const query = SELECT email FROM person WHERE Lower(email) = ?;

connection.query(query,[email], (error, results) => {

if (error) {

console.error(error);

return res.status(500).json({ error: 'Internal server error' });

}

console.log('Query results:', results);

if (results.length > 0) {

return res.status(400).json({ error: 'Email already has an account' });

}

console.log('no error');

// Save the user's data to the database here

const sql = INSERT INTO person (name, email, password,mobile,userType,salt) VALUES (?,?,?,?,?,?);

connection.query(sql,[name,email,password,mobile,userType,salt], (err, result) => {

if (err) {

console.error('Error saving user data:', err);

return res.status(500).send({ error: 'Error saving user data' });

}

console.log('User data saved successfully:', result);

userId = result.insertId;

console.log('userId',userId)

module.exports.variable = userId;

res.send({ message: 'User data saved successfully', userId });

});

});

});

module.exports = router;

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

1.

npm install axios

npm install dompurify

npm install crypto-js

import CryptoJS from 'crypto-js';

import axios from 'axios';

import DOMPurify from 'dompurify';

2:31 PM

const handleNameChange = (event) => {

const sanitizedValue = DOMPurify.sanitize(event.target.value);

setName(sanitizedValue);

};

const handleSubmit = (event) => {}

axios.post('http://localhost:5000/create', { }

)

.then(response => {

console.log(response.data);

})

.catch(error => {

console.error(error);

if (error.response.status === 400) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

} else if (error.response.status === 500) {

setErrorMessage(DOMPurify.sanitize(error.response.data.error));

}

}

);

const sanitizedName = DOMPurify.sanitize(name);

name: sanitizedName

const express = require('express');

const mysql = require('mysql');

const connection = mysql.createConnection({ host: 'localhost', user: 'root', password: 'password', database: 'employeesystem' }); console.log('...........') connection.connect((err) => { if (err) { console.error('Error connecting to database:', err); return; } console.log('Connected to database!'); });

**Yesterday**

**sivaram**

[4:27 PM](https://mattermost.spybrick.in/tech-tribe/pl/bgy43hmaxpnd7fw3ezzfkkei9c)

npm install cors

const cors = require('cors');

app.use(express.static(path.join(\_\_dirname, 'client/build')));

app.use(express.json());

[4:32 PM](https://mattermost.spybrick.in/tech-tribe/pl/91uhfyghp7fg5n4dniokfhqfuc)

app.use(cors({ origin: '[http://localhost:3000](http://localhost:3000/)', // Allow requests from this origin methods: 'GET,HEAD,PUT,PATCH,POST,DELETE', credentials: true, // Allow cookies to be sent with requests allowedHeaders: 'Content-Type, Authorization' }));

DESCRIBE employee;

const salt = CryptoJS.lib.WordArray.random(16); // Generate a random 16-byte salt

const saltedPassword = salt + sanitizedPassword; // Add the salt to the password

const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString();

// Check if the email and password exists in the database

const query = SELECT \* FROM person WHERE Lower(email) = ?;

connection.query(query,[email], (error, results) => {

if (error) {

console.error(error);

return res.status(500).json({ error: 'Internal server error' });

}

7:20 PM

if (results.length > 0) {

//return res.status(400).json({ error: 'Email already has an account' });

const userType = results[0].userType;

const hashedPassword = results[0].password;

const salt = results[0].salt;

const saltedPassword = salt + password; // Add the salt to the provided password

const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString(); // Hash the salted password

const isPasswordMatch = encryptedPassword === hashedPassword;

if (isPasswordMatch) {

console.log('Login successful!');

console.log(userType);

return res.status(200).json({ data: userType, message: 'Successfully Logging In' });

} else {

console.log('Invalid email or password');

return res.status(600).json({ error: 'Invalid email or password' });

}

}

else{

console.log('Invalid email or password');

return res.status(400).json({ error: 'Invalid email or password' });

}

-----

.login

import React, { useState } from 'react';

import {Link } from 'react-router-dom';

import axios from 'axios';

import './logincss.css'; // Assuming you have a CSS file for styling

const Login1 = () => {

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const [error, setError] = useState('');

const navigate = useNavigate();

const handleEmailChange = (e) => {

setEmail(e.target.value);

};

const handlePasswordChange = (e) => {

setPassword(e.target.value);

};

const handleSubmit = async (e) => {

e.preventDefault();

setError('');

try {

const response = await axios.post('http://localhost:5000/Login', {

username: email,

password: password,

});

if (response.data.message === 'Login successful!') {

// Navigate to the desired page after successful login

navigate('/dashboard'); // Change '/dashboard' to your desired route

} else {

setError('Invalid email or password');

}

} catch (error) {

setError('An error occurred. Please try again.');

console.error('There was an error logging in:', error);

}

};

return (

<div className="login1-container">

<div className="text-container">

<h1>Bring home beautiful interiors <span className="highlight">that fit your budget</span></h1>

<p>Experience unmatched quality & timely delivery with Livspace</p>

</div>

<div className="container">

<input type="checkbox" id="check" />

<div className="login form">

<header>Login</header>

<form onSubmit={handleSubmit}>

<input

type="text"

placeholder="Enter your email"

value={email}

onChange={handleEmailChange}

/>

<input

type="password"

placeholder="Enter your password"

value={password}

onChange={handlePasswordChange}

/>

<a href="password.html">Forgot password?</a>

<input type="submit" className="button" value="Login" />

{error && <p className="error">{error}</p>}

</form>

<div className="signup">

<span className="signup">Don't have an account?

<Link to="/Signup">Signup</Link>

</span>

</div>

</div>

</div>

</div>

);

};

export default Login1;

.Signup encryption decryption:

import React, { useState } from 'react';

import CryptoJS from 'crypto-js';

import { useNavigate } from 'react-router-dom';

import axios from 'axios';

import DOMPurify from 'dompurify';

import Navbar from "./Navbar1";

import './signup.css';

export default function Signup() {

  const navigate = useNavigate();

  const [FirstName, setName] = useState('');

  const [email, setEmail] = useState('');

  const [password, setPassword] = useState('');

  const [mobile, setMobile] = useState('');

  const [errorMessage, setErrorMessage] = useState('');

  const [emailError, setEmailError] = useState('');

  const [passwordError, setPasswordError] = useState('');

  const [mobileError, setMobileError] = useState('');

  const validateEmail = (email) => {

    const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

    return regex.test(email);

  };

  const checkPasswordStrength = (password) => {

    const regex = /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[@!%\*?&])[A-Za-z\d@!%\*?&]{8,}$/;

    return regex.test(password);

  };

  const validateMobile = (mobile) => {

    const regex = /^\d{10}$/;

    return regex.test(mobile);

  };

  // Sanitize user inputs

  const handleNameChange = (event) => {

    const sanitizedValue = DOMPurify.sanitize(event.target.value);

    setName(sanitizedValue);

  };

  const handleEmailChange = (event) => {

    const sanitizedValue = DOMPurify.sanitize(event.target.value);

    setEmail(sanitizedValue);

  };

  const handlePasswordChange = (event) => {

    const sanitizedValue = DOMPurify.sanitize(event.target.value);

    setPassword(sanitizedValue);

  };

  const handleMobileChange = (event) => {

    const sanitizedValue = DOMPurify.sanitize(event.target.value);

    setMobile(sanitizedValue);

  };

  const handleErrorMessage = () => {

    setErrorMessage(" ");

  };

  // Email verification

  const sendVerificationEmail = async (email) => {

    try {

      const response = await axios.post('http://localhost:5000/send-verification-url', { email });

      console.log(response.data.message);

      // Show a success message to the user

    } catch (error) {

      console.error(error);

      // Show an error message to the user

    }

  };

  const handleSubmit = (event) => {

    event.preventDefault();

    // Sanitize input before sending to server

    const sanitizedName = DOMPurify.sanitize(FirstName);

    const sanitizedEmail = DOMPurify.sanitize(email);

    const sanitizedMobile = DOMPurify.sanitize(mobile);

    const sanitizedPassword = DOMPurify.sanitize(password);

    if (!validateEmail(email)) {

      setEmailError('Please enter a valid email address');

    } else {

      setEmailError('');

    }

    if (!checkPasswordStrength(password)) {

      setPasswordError('Password must be strong');

    } else {

      setPasswordError('');

    }

    if (!validateMobile(mobile)) {

      setMobileError('Please enter a valid mobile number');

    } else {

      setMobileError('');

    }

    if (validateEmail(email) && validateMobile(mobile) && checkPasswordStrength(password)) {

      // Encrypt password

      const salt = CryptoJS.lib.WordArray.random(16); // Generate a random 16-byte salt

      const saltedPassword = salt + sanitizedPassword; // Add the salt to the password

      const encryptedPassword = CryptoJS.SHA256(saltedPassword).toString();

      console.log('salt:', salt.toString());

      console.log('Hashed password:', encryptedPassword);

      // Submit the form

      axios.post('http://localhost:5000/create', {

        FirstName: sanitizedName,

        email: sanitizedEmail,

        mobile: sanitizedMobile,

        password: encryptedPassword, // Send the encrypted password to the server

        salt: salt.toString()

      })

        .then(response => {

          console.log(response.data);

        })

        .catch(error => {

          console.error(error);

          if (error.response.status === 400) {

            setErrorMessage(DOMPurify.sanitize(error.response.data.error));

          } else if (error.response.status === 500) {

            setErrorMessage(DOMPurify.sanitize(error.response.data.error));

          }

        });

      sendVerificationEmail(sanitizedEmail); // Send the verification email

      window.alert("Please check your email for verification.");

      navigate('/AddDetails');

    }

  };

  return (

    <div>

      <div className='container-fluid n'><Navbar /></div>

      <div className='container-fluid h1'><h2>Sign Up Form</h2></div>

      <div className='container-fluid s1'>

        <form onSubmit={handleSubmit}>

          <div className="input-container">

            <i className="fa fa-user icon"></i>

            <input className="input-field" type="text" value={FirstName} placeholder="Username" name="name" onChange={handleNameChange} />

          </div>

          <div className="input-container">

            <i className="fa fa-envelope icon"></i>

            <input className="input-field" type="text" value={email} placeholder="Email" name="email" onChange={handleEmailChange} />

          </div>

          <div className='e'>{emailError && <span style={{ color: 'red' }}>{emailError}</span>}</div>

          <div className="input-container">

            <i className="fa fa-mobile icon"></i>

            <input className="input-field" type="text" value={mobile} placeholder="Mobile" name="mobile" onChange={handleMobileChange} />

          </div>

          <div className='e'>{mobileError && <span style={{ color: 'red' }}>{mobileError}</span>}</div>

          <div className="input-container">

            <i className="fa fa-key icon"></i>

            <input className="input-field" type="password" value={password} placeholder="Password" name="password" onChange={handlePasswordChange} />

          </div>

          <div className='e'> {passwordError && <span style={{ color: 'red' }}>{passwordError}</span>}</div>

          <div className='container-fluid b1'><button type="submit" className="h" onClick={handleErrorMessage}>Sign Up</button>

            {errorMessage && <div style={{ color: 'red' }}>{errorMessage}</div>}</div>

        </form>

        </div>

      </div>

  );

}

Const FirstName = req.body. FirstName

const { FirstName, email, mobile, password } = req.body; // Destructure properties from request body

23-05-2024

return res.status(200).json({ data: userType, message: 'Successfully Logging In' });

return res.status(600).json({ error: 'Invalid email or password' });

return res.status(400).json({ error: 'Invalid email or password' });

response.status === 200

throw new Error('Error: ' + response.status);

import { useNavigate } from 'react-router-dom';

const navigate = useNavigate();

setTimeout(()=>navigate('/userVerified'),1000);

10:48 AM

.catch(error => {

console.error(error);

if (error.response?.status === 400) {

setErrorMessage(DOMPurify.sanitize(error.response?.data?.error));

setTimeout(()=>setErrorMessage(''),3000);

} else if (error.response?.status === 500) {

setErrorMessage(DOMPurify.sanitize(error.response?.data?.error));

setTimeout(()=>setErrorMessage(''),3000);

}

else if (error.response?.status === 600) {

setErrorMessage(DOMPurify.sanitize(error.response?.data?.error));

setTimeout(()=>setErrorMessage(''),3000);

}

});

24-05-2024

const sendVerificationEmail = async (email) => {

try {

const response = await axios.post('http://localhost:5000/send-verification-url', { email });

console.log(response.data.message);

// show a success message to the user

} catch (error) {

console.error(error);

// show an error message to the user

}

}

const [errorMessage, setErrorMessage] = useState('');

const [emailError, setEmailError] = useState('');

const [passwordError, setPasswordError] = useState('');

const [mobileError, setMobileError] = useState('');

10:59 AM

const validateEmail = (email) => {

const regex = /^[^\s@]+@[^\s@]+.[^\s@]+$/;

return regex.test(email);

};

const checkPasswordStrength = (password) => {

const regex = /^(?=.[a-z])(?=.[A-Z])(?=.\d)(?=.[@

!

!!%\*?&]{8,}$/;

return regex.test(password);

};

const validateMobile = (mobile) => {

const regex = /^\d{10}$/;

return regex.test(mobile);

};

if (validateEmail(email) && validateMobile(mobile) && checkPasswordStrength(password)) {

sendVerificationEmail(sanitizedEmail); // send the verification email

window.alert("Please check your email for verification.");

navigate('/AddDetails');

}

npm install nodemailer

11:16 AM

npm install jsonwebtoken

const router = express.Router();

module.exports = router;

11:27 AM

const connection = require('./dbConnection');

app.use('/create', create);

11:56 AM

const create = require('./routes/create');

app.use(express.static(path.join(\_\_dirname, '/client/build')));

12:19 PM

const path = require('path')

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

1.index.js

const express = require('express');

const bodyParser = require('body-parser');

const app = express();

const cors = require('cors')

app.use(cors());

app.use(express.json());

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

const Signin = require('./Routes/Signin');

const SendVerification = require('./Routes/Sendverification');

const Verify = require('./Routes/Verify');

const Login = require('./Routes/Login');

const path = require('path')

app.use(express.static(path.join(\_\_dirname, '/client/build')));

console.log('...........')

//sign up

app.use('/Signin', Signin);

//update details

//send email

app.use('/SendVerification', SendVerification);

//email verification

app.use('/verify', Verify);

//user login

app.use('/Login',Login);

//user Info

app.listen(5000, () => console.log('Server running on port 5000'));

2.email (done)

const express = require('express');

const nodemailer = require('nodemailer');

const jwt = require('jsonwebtoken');

const connection = require('./Dbconnection')

//Email verification

const transporter = nodemailer.createTransport({

host: 'smtp.gmail.com',

port: 465,

secure: true,

auth: {

user: 'sravanasandhyaravipati@gmail.com',

pass: 'mknu infg mcue ovfx',

},

});

const sendVerificationEmail = async (Email) => {

const secretKey = 'mySecretKey'; // Replace with your actual secret key

const token = jwt.sign({ Email }, secretKey, { expiresIn: '1h' });

const verificationUrl = http://localhost:5000/verify?token=${token};

// save the token to the database

const sql = UPDATE employee SET verification\_token = ? WHERE Email = ?;

connection.query(sql, [token, Email], (err) => {

if (err) {

console.error('Error saving user data:', err);

} else {

console.log('User data saved successfully:');

}

});

const mailOptions = {

from: 'sravanasandhyaravipati@gmail.com',

to: Email,

subject: 'Verify your Email address',

html: `<p>Please click on the following link to verify your Email address:</p>

<a href="${verificationUrl}">${verificationUrl}</a>`,

};

await transporter.sendMail(mailOptions);

};

module.exports = sendVerificationEmail;

3.sendverification

const express = require('express');

const Router = express.Router();

const sendVerificationEmail = require('./Email');

Router.post('/', async (req, res) => {

try {

const { Email } = req.body; // Destructure Email from req.body

console.log(Email);

console.log('Email verification');

await sendVerificationEmail(Email); // Pass Email to sendVerificationEmail function

res.json({ message: 'Verification Email sent' });

} catch (error) {

console.error('Error sending verification Email:', error);

res.status(500).json({ message: 'Failed to send verification Email' });

}

});

module.exports = Router;

4.DB

const express = require("express");

const mysql = require("mysql");

const connection = mysql.createConnection({

host: "localhost",

user: "root",

password: "Sandhya",

database: "firstdb"

});

connection.connect((err) => {

if (err) {

console.error('Error connecting to the database:', err);

return;

}

console.log('Connected to the database');

});

Login and signup

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

Twilio

FAYCSU9HV2FKHDCUWP6MU1QG

**Congratulations! You’ve got a Twilio phone number!**

**+12233380371**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1.generate the otp and sent it to your email id

Login using otp:

Frentend:

1. Loginemail.jsx
2. import React, { useState } from 'react';
3. import axios from 'axios';
4. const LoginPage = () => {
5. const [email, setEmail] = useState('');
6. const [otp, setOtp] = useState('');
7. const [otpSent, setOtpSent] = useState(false);
8. const sendOtp = async () => {
9. try {
10. const response = await axios.post('http://localhost:5000/sendotpp', { email });
11. if (response.status === 200) {
12. alert('OTP sent successfully!');
13. setOtpSent(true);
14. }
15. } catch (error) {
16. console.error('Error sending OTP:', error);
17. alert('Failed to send OTP.');
18. }
19. };
20. const verifyOtp = async () => {
21. try {
22. const response = await axios.post('http://localhost:5000/verifyotpp', { email, otp });
23. if (response.status === 200) {
24. alert('OTP verified successfully!');
25. } else {
26. alert('Invalid OTP.');
27. }
28. } catch (error) {
29. console.error('Error verifying OTP:', error);
30. alert('Failed to verify OTP.');
31. }
32. };
33. return (
34. <div className="login-container">
35. <h2>Login</h2>
36. <div>
37. <label>Email:</label>
38. <input type="email" value={email} onChange={(e) => setEmail(e.target.value)} required />
39. </div>
40. {otpSent ? (
41. <div>
42. <label>OTP:</label>
43. <input type="text" value={otp} onChange={(e) => setOtp(e.target.value)} required />
44. <button onClick={verifyOtp}>Verify OTP</button>
45. </div>
46. ) : (
47. <button onClick={sendOtp}>Send OTP</button>
48. )}
49. </div>
50. );
51. };
52. export default LoginPage;

app.js

<Route path='/LoginPage' element={<LoginPage/>}/>

<http://localhost:3000/LoginPage> - Login using otp

<http://localhost:3000/Emailverification> -> email through otp

<http://localhost:3000/PhoneNumberForm> ->send otp in phone

<http://localhost:3000/otpVerification> -> enter the code

<http://localhost:3000/verifyEmail> -> this signup time the email to link

RUF

frentend ( login time giving otp)

1.

import React, { useState } from 'react';

import axios from 'axios';

const LoginPage = () => {

const [email, setEmail] = useState('');

const [otp, setOtp] = useState('');

const [otpSent, setOtpSent] = useState(false);

const sendOtp = async () => {

try {

const response = await axios.post('http://localhost:5000/send-otp', { email });

if (response.status === 200) {

alert('OTP sent successfully!');

setOtpSent(true);

}

} catch (error) {

console.error('Error sending OTP:', error);

alert('Failed to send OTP.');

}

};

const verifyOtp = async () => {

try {

const response = await axios.post('http://localhost:5000/verify-otp', { email, otp });

if (response.status === 200) {

alert('OTP verified successfully!');

} else {

alert('Invalid OTP.');

}

} catch (error) {

console.error('Error verifying OTP:', error);

alert('Failed to verify OTP.');

}

};

return (

<div className="login-container">

<h2>Login</h2>

<div>

<label>Email:</label>

<input type="email" value={email} onChange={(e) => setEmail(e.target.value)} required />

</div>

{otpSent ? (

<div>

<label>OTP:</label>

<input type="text" value={otp} onChange={(e) => setOtp(e.target.value)} required />

<button onClick={verifyOtp}>Verify OTP</button>

</div>

) : (

<button onClick={sendOtp}>Send OTP</button>

)}

</div>

);

};

export default LoginPage;

backend

2.

const express = require('express');

const nodemailer = require('nodemailer');

const bodyParser = require('body-parser');

const cors = require('cors');

require('dotenv').config(); // Load environment variables

const app = express();

const PORT = process.env.PORT || 5000;

// Middleware

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

app.use(cors());

// Function to generate a 6-digit OTP

const generateOTP = () => Math.floor(100000 + Math.random() \* 900000).toString();

// Nodemailer transporter

const transporter = nodemailer.createTransport({

service: 'gmail',

auth: {

user: process.env.EMAIL,

pass: process.env.EMAIL\_PASSWORD,

},

});

// Store OTPs in memory (in a real application, use a database)

let storedOtps = {};

// Route to send OTP

app.post('/send-otp', async (req, res) => {

const { email } = req.body;

const otp = generateOTP();

storedOtps[email] = otp;

const mailOptions = {

from: process.env.EMAIL,

to: email,

subject: 'Your OTP Code',

text: `Your OTP code is ${otp}`,

};

try {

await transporter.sendMail(mailOptions);

console.log('Email sent successfully');

res.status(200).json({ message: 'OTP sent successfully' });

} catch (error) {

console.error('Error sending email:', error);

res.status(500).send('Internal Server Error: Unable to send OTP');

}

});

// Route to verify OTP

app.post('/verify-otp', (req, res) => {

const { email, otp } = req.body;

if (storedOtps[email] === otp) {

res.status(200).json({ message: 'OTP verified successfully' });

} else {

res.status(400).json({ message: 'Invalid OTP' });

}

});

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

3.index.js

app.use('/sendotpp',Logonotp);

const express = require('express');

const cors = require('cors');

const app = express();

const bodyParser = require('body-parser');

app.use(express.json());

app.use(cors());

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

-----------------------------------------------------------

navya:Email otp related

1.

const express = require('express');

const nodemailer = require('nodemailer');

const jwt = require('jsonwebtoken');

const connection = require('./DBconnection');

// Email verification

const transporter = nodemailer.createTransport({

service: 'gmail',

auth: {

user: 'navyasuryadevara99@gmail.com',

pass: 'pdio arfb pbag xemg', // Ensure this is an App Password, not your regular email password

},

});

const generateOtp = () => Math.floor(100000 + Math.random() \* 900000);

const SendEmailOtp = async (Email) => {

const secretKey = 'mySecretKey'; // Replace with your actual secret key

const otp = generateOtp();

console.log(otp);

const mailOptions = {

from: 'navyasuryadevara99@gmail.com',

to: Email,

subject: 'Verify your email address',

html: `<p>verify your email with Otp:${otp}</p>`

};

await transporter.sendMail(mailOptions);

};

module.exports = SendEmailOtp;

2.

const express = require('express');

const router = express.Router();

const SendEmailOtp = require('./SendEmailOtp');

const connection = require('./DBconnection');

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

const Email = req.body.Email;

await SendEmailOtp(Email);

res.json({ message: 'Verification email sent' });

});

module.exports =router;

9:54 PM

3.index.js

const express = require('express');

const app = express();

const path = require('path');

const cors = require('cors');

const Login = require('./Login');

const Signup = require('./Signup');

const Sendverification = require('./Sendverification');

const verify = require('./Verify');

const sendotp = require('./SendOtp');

const verifyotp = require('./VerifyOtp');

const SendEmailVerify = require('./SendEmailVerify');

app.use(express.static(path.join(\_\_dirname, '/client/build')));

app.use(express.json());

app.use(cors({

origin: 'http://localhost:3000', // Allow requests from this origin

methods: 'GET,HEAD,PUT,PATCH,POST,DELETE',

credentials: true, // Allow cookies to be sent with requests

allowedHeaders: 'Content-Type, Authorization'

}));

app.use('/Signin', Signup);

app.use('/Sendverification', Sendverification);

app.use('/verify', verify);

app.use('/Login', Login);

app.use('/send-otp', sendotp);

app.use('/verify-otp', verifyotp);

app.use('/Send-EmailVerification', SendEmailVerify);

// Catch-all handler for any requests that don't match any of the above routes

app.get('\*', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'client', 'build', 'index.html'));

});

// Error handling middleware (optional, but recommended)

app.use((err, req, res, next) => {

console.error(err.stack);

//res.status(500).send('Something went wrong!');

});

app.listen(5000, () => console.log('Server running on port 5000'));

email

import React, { useState } from 'react';

import axios from 'axios';

const EmailOtpForm = () => {

const [otp, setOtp] = useState('');

const handleSubmit = async (e) => {

e.preventDefault();

try {

const response = await axios.post('http://localhost:5000/verify-emailotp', {

otp

});

if (response.status === 200) {

alert('OTP verified successfully!');

} else {

alert('Failed to verify OTP.');

}

} catch (error) {

console.error('Error verifying OTP:', error);

alert('Failed to verify OTP.');

}

};

return (

<form onSubmit={handleSubmit}>

<div>

<label>

OTP:

<input

type="text"

value={otp}

onChange={(e) => setOtp(e.target.value)}

required

placeholder="Enter the OTP"

/>

</label>

</div>

<button type="submit">Verify OTP</button>

</form>

);

};

export default EmailOtpForm;

RUF

Navbar :-

import React from 'react';

import './Navbar1css.css';

import image from './Images/Image.png';

function Navbar() {

return (

<nav className="navbar navbar-expand-lg bg-body-tertiary">

<div className="container-fluid">

<button className="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

<span className="navbar-toggler-icon"></span>

</button>

<a className="navbar-brand" href="#">

<img src={image} alt="Logo" />

</a>

<div className="collapse navbar-collapse justify-content-between" id="navbarSupportedContent">

<ul className="navbar-nav me-auto mb-2 mb-lg-0">

<li className="nav-item">

<a className="nav-link" href="#">Interiors</a>

</li>

<li className="nav-item dropdown">

<a className="active nav-link " href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Design ideas

</a>

<ul className="dropdown-menu">

<li><a className="dropdown-item" href="#">Modular Kitchen Designs</a></li>

<li><a className="dropdown-item" href="#">Wardrobe Designs</a></li>

<li><a className="dropdown-item" href="#">Bathroom Designs</a></li>

<li><hr className="dropdown-divider" /></li>

<li><a className="dropdown-item" href="#">Something else here</a></li>

</ul>

</li>

<li className="nav-item dropdown">

<a className="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Magazine

</a>

<ul className="dropdown-menu">

<li><a className="dropdown-item" href="#">Latest Trends</a></li>

<li><a className="dropdown-item" href="#">Home Decor</a></li>

<li><a className="dropdown-item" href="#">Expert Tips</a></li>

<li><hr className="dropdown-divider" /></li>

<li><a className="dropdown-item" href="#">Something else here</a></li>

</ul>

</li>

<li className="nav-item dropdown">

<a className="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Cities

</a>

<ul className="dropdown-menu">

<li><a className="dropdown-item" href="#">New York</a></li>

<li><a className="dropdown-item" href="#">London</a></li>

<li><a className="dropdown-item" href="#">Tokyo</a></li>

</ul>

</li>

</ul>

<div class="text-end">

<a href='/Login' class="btn btn-outline-primary">Login</a>

</div>

</div>

</div>

</nav>

);

}

export default Navbar;

----------------------------------------------------------------

emaiotp.js

const [email, setEmail] = useState('');

const [otp, setOtp] = useState('');

const [errorMessage, setErrorMessage] = useState('');

const handleSubmit = async (event) => {

event.preventDefault();

try {

const response = await fetch('/api/request-otp', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ email })

});

const data = await response.json();

if (data.success) {

// Handle successful OTP request (e.g., show success message)

} else {

setErrorMessage(data.error); // Set error message from response

}

} catch (error) {

console.error(error);

setErrorMessage('An error occurred. Please try again later.');

}

};

3. OTP Verification:

const handleVerifyOtp = async (event) => {

event.preventDefault();

try {

const response = await fetch('/api/verify-otp', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ email, otp })

});

const data = await response.json();

if (data.success) {

// Handle successful OTP verification (e.g., redirect to authenticated area)

} else {

setErrorMessage(data.error); // Set error message from response

}

} catch (error) {

console.error(error);

setErrorMessage('An error occurred. Please try again later.');

}

};

<img src={livingroomimage1} alt="Living Room" />

RUF

router.post('/', (req, res) => {

const { FirstName, email, password, mobile, verification\_token, salt } = req.body;

// Check if email is provided

if (!email) {

return res.status(400).json({ error: 'Email is required' });

}

// Check if the email already exists in the database

const query = 'SELECT email FROM employee WHERE LOWER(email) = ?';

connection.query(query, [email.toLowerCase()], (error, results) => {

if (error) {

console.error('Error during email check:', error);

return res.status(500).json({ error: 'Internal server error' });

}

if (results.length > 0) {

return res.status(400).json({ error: 'Email already has an account' });

}

// Save the user's data to the database here

const sql = 'INSERT INTO employee (FirstName, email, mobile, password, salt, verification\_token) VALUES (?, ?, ?, ?, ?, ?)';

connection.query(sql, [FirstName, email, mobile, password, salt, verification\_token], (err, result) => {

if (err) {

console.error('Error saving user data:', err);

return res.status(500).send({ error: 'Error saving user data' });

}

console.log('User data saved successfully:', result);

const userId = result.insertId; // Get the inserted user ID

res.send({ message: 'User data saved successfully', userId });

console.log('Data Saved');

});

});

});

module.exports = router;

,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

const query = 'SELECT email FROM employee WHERE LOWER(email) = ?';

connection.query(query, [email.toLowerCase()], (error, results) => {

if (error) {

console.error('Error during email check:', error);

return res.status(500).json({ error: 'Internal server error' });

}

if (results.length > 0) {

return res.status(400).json({ error: 'Email already has an account' });

}

const express = require('express');

const router = express.Router();

const sendVerificationEmail = require('./Routes/Email');

const connection = require('./Routes/Dbconnection');

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

try {

console.log(email);

console.log('Email verification');

const email = req.body.email;

await sendVerificationEmail(email);

res.json({ message: 'Verification email sent' });

} catch (error) {

console.error('Error sending verification email:', error);

res.status(500).json({ error: 'Internal server error' });

}

});

module.exports =router;

email

const express = require('express');

const nodemailer = require('nodemailer');

const jwt = require('jsonwebtoken');

const connection = require('./Routes/Dbconnection');

//database connection

//email verification

const transporter = nodemailer.createTransport({

host: 'smtp.gmail.com',

port: 465,

secure: true,

auth: {

user: 'suryadevarabhavana@gmail.com',

pass: 'houx pkfx glxu ucuv',

},

});

const sendVerificationEmail = async (email) => {

const secretKey = 'mySecretKey'; // Replace with your actual secret key

const token = jwt.sign({ email }, secretKey, { expiresIn: '1h' });

const verificationUrl = `http://localhost:5000/verify?token=${token}`;

// save the token to the database

const sql = `INSERT INTO user (email, Token) VALUES(?,?)`;

connection.query(sql,[ email,token], (err) => {

if (err) {

console.error('Error saving user data:', err);

}

console.log('User data saved successfully:');

});

const mailOptions = {

from: 'suryadevarabhavana@gmail.com',

to: email,

subject: 'Verify your email address',

html: `<p>Please click on the following link to verify your email address:</p>

<a href="${verificationUrl}">${verificationUrl}</a>`,

};

await transporter.sendMail(mailOptions);

};

module.exports = sendVerificationEmail;

Github:

### …or create a new repository on the command line

echo "# livspace.com" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/spybrick-technologies/livspace.com.git

git push -u origin main

**…or push an existing repository from the command line**

git remote add origin https://github.com/spybrick-technologies/livspace.com.git

git branch -M main

git push -u origin main

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Here are some common Git commands:

**Basic Git Commands**

1. **git init**: Initializes a new Git repository in the current directory.
2. **git add <file>**: Stages a file for the next commit.
3. **git add.**: Stages all changes in the current directory and subdirectories.
4. **git commit -m "<message>"**: Commits changes with a meaningful commit message.
5. **git log**: Displays a log of all commits made to the repository.
6. **git status**: Displays the status of the repository, including any changes that need to be committed.

**Branching and Merging**

1. **git branch <branch-name>**: Creates a new branch with the given name.
2. **git checkout <branch-name>**: Switches to the specified branch.
3. **git merge <branch-name>**: Merges changes from another branch into the current branch.
4. **git branch -d <branch-name>**: Deletes a branch.

**Remote Repositories**

1. **git remote add <name> <url>**: Adds a remote repository with the given name and URL.
2. **git fetch <remote>**: Fetches changes from a remote repository.
3. **git push <remote> <branch>**: Pushes changes to a remote repository.
4. **git pull <remote> <branch>**: Pulls changes from a remote repository and merges them into the current branch.

**Undoing Changes**

1. **git reset <file>**: Unstages a file.
2. **git reset --hard**: Resets the repository to the last commit, discarding all changes.
3. **git checkout -- <file>**: Reverts changes to a file.
4. **git revert <commit>**: Reverts a specific commit.

**Other Commands**

1. **git tag <tag-name>**: Creates a new tag with the given name.
2. **git show <commit>**: Displays information about a specific commit.
3. **git diff <commit>**: Displays the differences between two commits.
4. **git clean**: Removes untracked files and directories.
5. **git config**: Configures Git settings, such as the user name and email.

**Git Flow**

1. **git flow init**: Initializes a Git Flow repository.
2. **git flow feature start <feature-name>**: Starts a new feature branch.
3. **git flow feature finish <feature-name>**: Finishes a feature branch and merges it into the develop branch.
4. **git flow release start <release-name>**: Starts a new release branch.
5. **git flow release finish <release-name>**: Finishes a release branch and merges it into the master branch.

**Git Submodules**

1. **git submodule add <url>**: Adds a submodule to the repository.
2. **git submodule init**: Initializes a submodule.
3. **git submodule update**: Updates a submodule to the latest commit.
4. **git submodule foreach <command>**: Runs a command on each submodule.

These are just a few of the many Git commands available. For more information, you can use **git help** or **git <command> --help** to get detailed documentation on a specific command.

<https://github.com/sravanasandhyaravipati/BitGet.git>