**Implementation of *InternTrack* Full Stack Web Application**

UCS2601 - Internet Programming

Mini Project

Submitted By

Prithivirajan D 3122225001099

Rithekha K 3122225001106

Samyuktha D 3122225001309

### A blue and white logo AI-generated content may be incorrect.

### **Department of Computer Science and Engineering**

### 

### **Sri Sivasubramaniya Nadar College of Engineering**

### **(An Autonomous Institution, Affiliated to Anna University, Chennai)**

### **Kalavakkam – 603 110**

### 

February 2025

**Introduction**

This project is a full stack web application developed to streamline and manage the internship records of students in an academic setting. The main goal of the system is to provide coordinators with an intuitive and efficient platform to track, view, filter, and manage internship-related data. Built using React for the frontend and integrated with a backend API for database interactions, the system allows coordinators to search for individual student records, view all submissions, apply filters based on company, location, or source of internship, and delete records if necessary. The user interface is designed for simplicity and clarity, making it easy to navigate through various functionalities. This project not only simplifies the administrative workload but also ensures that all relevant documents and statuses related to each internship are properly tracked and accessible.

**Purpose**

The primary purpose of this project is to simplify and digitize the process of managing student internship records within an educational institution. Traditionally, maintaining these records involved manual entry, scattered data, and difficulty in retrieving specific information. This system aims to overcome these challenges by offering a centralized digital platform where internship details can be securely stored, easily accessed, and efficiently managed by the coordinator. It enables quick searching by registration number, filtering based on various criteria such as location, company, or source of internship, and ensures that proof documents like offer letters and completion certificates are tracked. Ultimately, the project is designed to improve accuracy, save time, reduce paperwork, and provide a smoother experience for both coordinators and students.

**Scope**

The scope of this project encompasses the development of a full-stack web application specifically designed for college coordinators to manage and monitor internship records of students. The system allows for the collection, display, filtering, searching, and deletion of internship-related data, including company details, duration, location, and proof documents. It is designed to be user-friendly, scalable, and adaptable to various departments within the institution. The application supports role-based access, ensuring that only authorized users can modify or view sensitive information. Furthermore, the system is flexible enough to be extended with features like PDF generation of reports, analytics dashboards, or integration with external portals. Overall, this project serves as a foundation for digital transformation in internship tracking and administration.

**Technology Used**

**Frontend:**

* React.js
* React Router
* JavaScript
* HTML
* CSS (with inline styling or custom CSS files)

**Backend**

* Node.js
* Express.js

**API Communication:**

Axios (for HTTP requests)

**Database:**

Mongoose (MongoDB ODM)

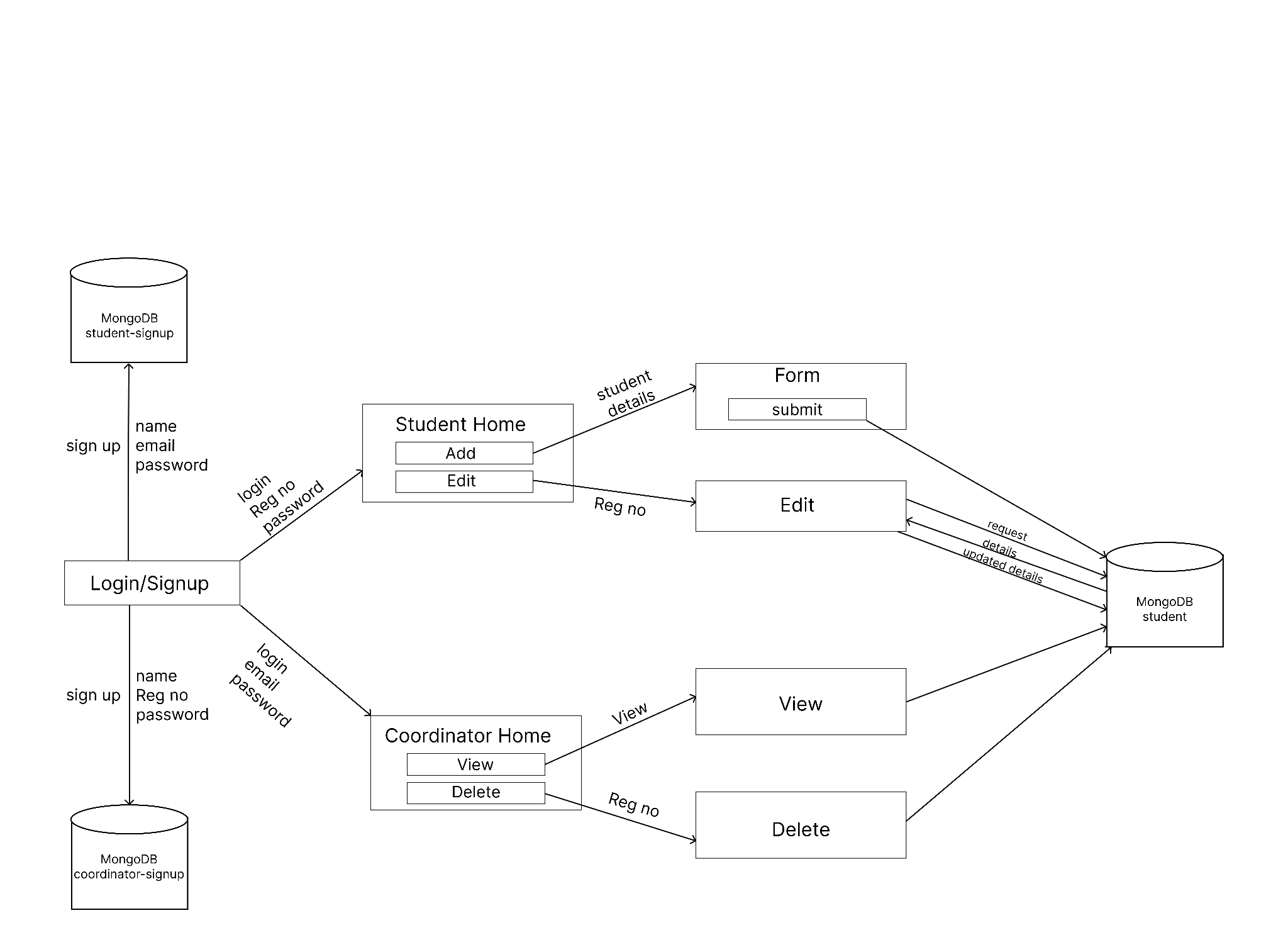
**Authentication:**

* JSON Web Token (JWT)
* CORS (Cross-Origin Resource Sharing)

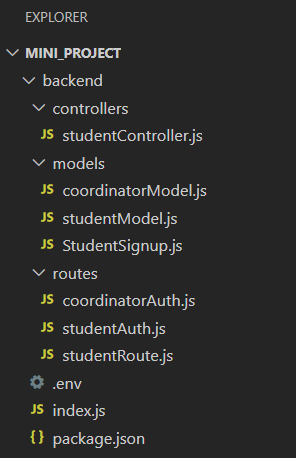
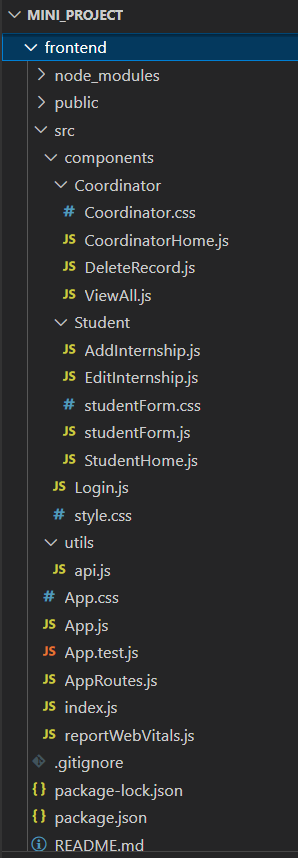
**Development Tools:**

* Visual Studio Code (VS Code)
* Postman (for API testing)

**Page Flow**

****

**Folder Structure**

****

**Implementation**

**Frontend (React)**

1. Developed using React to build a responsive and user-friendly interface for the internship coordinator.
2. Used useState for managing form inputs and storing fetched student data.
3. Implemented React Router for navigation between different pages such as:  
   1. Coordinator Home
   2. View All Students
   3. Delete a Record
4. Axios was used to send HTTP requests (GET, DELETE) to the backend API.
5. Search functionality implemented to find students based on registration number.
6. Filters were added to view records based on:  
   1. Internship Source (College / Own)
   2. Location (India / Abroad)
   3. Company name (using prompt input)
7. Table used to display student details in a structured format with proof links.
8. Basic styling handled using inline CSS and optional custom CSS file.

**Backend (Node.js + Express.js)**

1. Backend server built using Node.js with Express.js framework to handle routing and logic.
2. Created RESTful API endpoints such as:  
   1. GET / – Fetch all student records
   2. DELETE /:id – Delete a specific student record by ID
3. MongoDB used to store student internship records as documents in a collection.
4. Used Mongoose to interact with MongoDB, define schema, and perform queries.
5. JSON used as the format for data exchange between frontend and backend.
6. CORS configured to allow cross-origin requests from the React frontend.
7. JWT (JSON Web Token) used for secure authentication and access control

**1. Frontend**

|  |
| --- |
| **App.js**  **import React from "react";**  **import { BrowserRouter as Router, Routes, Route } from "react-router-dom";**  **import Login from "./components/Login";**  **import StudentHome from "./components/Student/StudentHome";**  **import AddInternship from "./components/Student/AddInternship";**  **import EditInternship from "./components/Student/EditInternship";**  **import CoordinatorHome from "./components/Coordinator/CoordinatorHome";**  **import ViewAll from "./components/Coordinator/ViewAll";**  **import DeleteRecord from "./components/Coordinator/DeleteRecord";**  **export default function App() {**  **return (**  **<Router>**  **<Routes>**  **<Route path="/" element={<Login />} />**  **<Route path="/student/home" element={<StudentHome />} />**  **<Route path="/student/add" element={<AddInternship />} />**  **<Route path="/student/edit" element={<EditInternship />} />**  **<Route path="/coordinator/home" element={<CoordinatorHome />} />**  **<Route path="/coordinator/view" element={<ViewAll />} />**  **<Route path="/coordinator/delete" element={<DeleteRecord />} />**  **</Routes>**  **</Router>**  **);**  **}** |

|  |
| --- |
| **api.js**  **import axios from "axios";**  **const api = axios.create({**  **baseURL: "http://localhost:5000/api/students",**  **});**  **export default api;** |

|  |
| --- |
| **Login.js**  **import React, { useState } from "react";**  **import { useNavigate } from "react-router-dom";**  **import axios from "axios";**  **import "./style.css";**  **export default function LoginSignup() {**  **const [role, setRole] = useState("");**  **const [mode, setMode] = useState("login");**  **const navigate = useNavigate();**  **const handleSubmit = async (e) => {**  **e.preventDefault();**  **const form = e.target;**  **const reg\_no = form.reg\_no?.value;**  **const email = form.email?.value;**  **const password = form.password.value;**  **const name = form.name?.value;**  **try {**  **let endpoint = "";**  **if (role === "student") {**  **endpoint = "http://localhost:5000/api/auth";**  **} else if (role === "coordinator") {**  **endpoint = "http://localhost:5000/api/coordinator";**  **}**  **if (mode === "signup") {**  **const data =**  **role === "student"**  **? { reg\_no, password, name }**  **: { email, password, name };**    **await axios.post(`${endpoint}/signup`, data);**  **alert(`${role.charAt(0).toUpperCase() + role.slice(1)} signup successful!`);**    **} else {**  **const data =**  **role === "student"**  **? { reg\_no, password }**  **: { email, password };**    **const res = await axios.post(`${endpoint}/login`, data);**    **if (res.data.success) {**  **if (role === "student") {**  **localStorage.setItem("reg\_no", reg\_no);**  **navigate("/student/home");**  **} else {**  **localStorage.setItem("coordinator\_email", email);**  **navigate("/coordinator/home");**  **}**  **}**  **}**  **} catch (err) {**  **alert("Error: " + (err.response?.data?.message || err.message));**  **}**  **};**    **return (**  **<div className="login-container">**  **<h2 className="login-title">InternTrack Login / Signup</h2>**  **<div className="role-buttons">**  **<button**  **className={`role-button ${role === "student" ? "active" : ""}`}**  **onClick={() => setRole("student")}**  **>**  **Student**  **</button>**  **<button**  **className={`role-button ${role === "coordinator" ? "active" : ""}`}**  **onClick={() => setRole("coordinator")}**  **>**  **Coordinator**  **</button>**  **</div>**  **{role && (**  **<>**  **<div className="mode-buttons">**  **<button**  **className={`mode-button ${mode === "login" ? "active" : ""}`}**  **onClick={() => setMode("login")}**  **>**  **Login**  **</button>**  **<button**  **className={`mode-button ${mode === "signup" ? "active" : ""}`}**  **onClick={() => setMode("signup")}**  **>**  **Signup**  **</button>**  **</div>**  **<form className="login-form" onSubmit={handleSubmit}>**  **{mode === "signup" && (**  **<div className="form-group">**  **<label>Name:</label>**  **<input type="text" name="name" required className="form-input" />**  **</div>**  **)}**  **{role === "student" && (**  **<div className="form-group">**  **<label>Register No:</label>**  **<input type="text" name="reg\_no" required className="form-input" />**  **</div>**  **)}**  **{role === "coordinator" && (**  **<div className="form-group">**  **<label>Email:</label>**  **<input type="email" name="email" required className="form-input" />**  **</div>**  **)}**  **<div className="form-group">**  **<label>Password:</label>**  **<input type="password" name="password" required className="form-input" />**  **</div>**  **<button type="submit" className="submit-button">**  **{mode === "login" ? "Login" : "Signup"}**  **</button>**  **</form>**  **</>**  **)}**  **</div>**  **);**  **}** |

|  |
| --- |
| **StudentHome.js**  **import React from "react";**  **import { useNavigate } from "react-router-dom";**  **export default function StudentHome() {**  **const navigate = useNavigate();**  **const containerStyle = {**  **padding: "20px",**  **};**  **const headingStyle = {**  **fontSize: "24px",**  **color: "#6a0dad",**  **marginBottom: "20px",**  **};**  **const buttonStyle = {**  **margin: "10px",**  **padding: "10px 20px",**  **color: "white",**  **border: "none",**  **cursor: "pointer",**  **fontSize: "16px",**  **borderRadius: "4px",**  **};**  **const handleSignOut = () => {**  **localStorage.removeItem("student\_token");**  **navigate("/");**  **};**  **return (**  **<div style={containerStyle}>**  **<h2 style={headingStyle}>Student Portal</h2>**  **<button**  **onClick={() => navigate("/student/add")}**  **style={{ ...buttonStyle, backgroundColor: "#9b59b6" }}**  **>**  **Add Internship Details**  **</button>**  **<button**  **onClick={() => navigate("/student/edit")}**  **style={{ ...buttonStyle, backgroundColor: "#9b59b6" }}**  **>**  **Edit Internship Details**  **</button>**  **<button**  **onClick={handleSignOut}**  **style={{ ...buttonStyle, backgroundColor: "#9b59b6" }}**  **>**  **Sign Out**  **</button>**  **</div>**  **);**  **}** |

|  |
| --- |
| **AddInternship.js**  **// src/components/Student/AddInternship.jsx**  **import React from 'react';**  **import StudentForm from './studentForm';**  **const AddInternship = () => {**  **return (**  **<div>**  **<h2>Add Internship Details</h2>**  **<StudentForm />**  **</div>**  **);**  **};**  **export default AddInternship;** |

|  |
| --- |
| **StudentForm.js**  **import React, { useState } from 'react';**  **import './studentForm.css';**  **import { useNavigate } from 'react-router-dom';**  **function AddInternship() {**  **const navigate = useNavigate();**  **const [formData, setFormData] = useState({**  **reg\_no: '',**  **name: '',**  **title: '',**  **mobile\_no: '',**  **section: '',**  **obtained\_internship: '',**  **period: '',**  **start\_date: '',**  **end\_date: '',**  **company\_name: '',**  **placement\_source: '',**  **stipend: '',**  **internship\_type: '',**  **location: '',**  **gdrive\_folder\_link: '',**  **proofs: {**  **offer\_letter: false,**  **joining\_letter: false,**  **completion\_certificate: false,**  **report: false**  **}**  **});**  **const [message, setMessage] = useState('');**  **const [isError, setIsError] = useState(false);**  **const handleChange = (e) => {**  **const { name, value, type, checked } = e.target;**  **if (name.startsWith("proofs")) {**  **const proofKey = name.match(/\[(.\*?)\]/)[1];**  **setFormData(prev => ({**  **...prev,**  **proofs: {**  **...prev.proofs,**  **[proofKey]: checked**  **}**  **}));**  **} else {**  **setFormData(prev => ({**  **...prev,**  **[name]: type === "checkbox" ? checked : value**  **}));**  **}**  **};**  **const handleSubmit = async (e) => {**  **e.preventDefault();**  **try {**  **const checkResponse = await fetch('http://localhost:5000/api/students');**  **const students = await checkResponse.json();**  **const alreadyExists = students.some(student => student.reg\_no === formData.reg\_no);**  **if (alreadyExists) {**  **setMessage('A student with this Registration Number already exists.');**  **setIsError(true);**  **return;**  **}**  **const response = await fetch('http://localhost:5000/api/students/create', {**  **method: 'POST',**  **headers: {**  **'Content-Type': 'application/json',**  **},**  **body: JSON.stringify(formData),**  **});**  **const data = await response.json();**  **if (response.ok) {**  **setMessage(data.message || 'Form submitted successfully!');**  **setIsError(false);**  **setFormData({**  **reg\_no: '',**  **name: '',**  **title: '',**  **mobile\_no: '',**  **section: '',**  **obtained\_internship: '',**  **period: '',**  **start\_date: '',**  **end\_date: '',**  **company\_name: '',**  **placement\_source: '',**  **stipend: '',**  **internship\_type: '',**  **location: '',**  **gdrive\_folder\_link: '',**  **proofs: {**  **offer\_letter: false,**  **joining\_letter: false,**  **completion\_certificate: false,**  **report: false**  **}**  **});**  **} else {**  **setMessage(data.message || 'Failed to submit');**  **setIsError(true);**  **}**  **} catch (err) {**  **console.error('Error submitting form:', err);**  **setMessage('Could not connect to server');**  **setIsError(true);**  **}**  **};**  **const goToHome = () => {**  **navigate('/student/home');**  **};**  **return (**  **<div className="container">**  **<h2>Student Internship Form</h2>**  **<form onSubmit={handleSubmit}>**  **<input type="hidden" name="userType" value="student" />**  **<label htmlFor="reg\_no">Registration No:</label>**  **<input type="text" id="reg\_no" name="reg\_no" value={formData.reg\_no} onChange={handleChange} required />**  **<label htmlFor="name">Name:</label>**  **<input type="text" id="name" name="name" value={formData.name} onChange={handleChange} required />**  **<label htmlFor="title">Title:</label>**  **<input type="text" id="title" name="title" value={formData.title} onChange={handleChange} required />**  **<label htmlFor="mobile\_no">Mobile No:</label>**  **<input type="text" id="mobile\_no" name="mobile\_no" value={formData.mobile\_no} onChange={handleChange} required />**  **<label htmlFor="section">Section:</label>**  **<input type="text" id="section" name="section" value={formData.section} onChange={handleChange} required />**  **<label htmlFor="obtained\_internship">Obtained Internship:</label>**  **<select id="obtained\_internship" name="obtained\_internship" value={formData.obtained\_internship} onChange={handleChange} required>**  **<option value="">Select</option>**  **<option value="Yes">Yes</option>**  **<option value="No">No</option>**  **</select>**  **<label htmlFor="period">Period:</label>**  **<input type="text" id="period" name="period" value={formData.period} onChange={handleChange} required />**  **<label htmlFor="start\_date">Start Date:</label>**  **<input type="date" id="start\_date" name="start\_date" value={formData.start\_date} onChange={handleChange} required />**  **<label htmlFor="end\_date">End Date:</label>**  **<input type="date" id="end\_date" name="end\_date" value={formData.end\_date} onChange={handleChange} required />**  **<label htmlFor="company\_name">Company Name:</label>**  **<input type="text" id="company\_name" name="company\_name" value={formData.company\_name} onChange={handleChange} required />**  **<label htmlFor="placement\_source">Placement Source:</label>**  **<select id="placement\_source" name="placement\_source" value={formData.placement\_source} onChange={handleChange} required>**  **<option value="">Select</option>**  **<option value="College">College</option>**  **<option value="Own">Own</option>**  **</select>**  **<label htmlFor="stipend">Stipend:</label>**  **<input type="number" id="stipend" name="stipend" value={formData.stipend} onChange={handleChange} required />**  **<label htmlFor="internship\_type">Internship Type:</label>**  **<select id="internship\_type" name="internship\_type" value={formData.internship\_type} onChange={handleChange} required>**  **<option value="">Select</option>**  **<option value="Academics">Academics</option>**  **<option value="Industrial">Industrial</option>**  **</select>**  **<label>Location:</label>**  **<input type="radio" id="india" name="location" value="India" checked={formData.location === "India"} onChange={handleChange} required />**  **<label htmlFor="india">India</label>**  **<input type="radio" id="abroad" name="location" value="Abroad" checked={formData.location === "Abroad"} onChange={handleChange} required />**  **<label htmlFor="abroad">Abroad</label>**  **<fieldset className="proofs-fieldset">**  **<legend>Proofs Submitted:</legend>**  **<label htmlFor="gdrive\_folder\_link">Google Drive Folder Link (Containing Proofs):</label>**  **<input**  **type="url"**  **id="gdrive\_folder\_link"**  **name="gdrive\_folder\_link"**  **value={formData.gdrive\_folder\_link}**  **onChange={handleChange}**  **required**  **/>**  **<div className="proofs-container">**  **<label>**  **<input type="checkbox" name="proofs[offer\_letter]" checked={formData.proofs.offer\_letter} onChange={handleChange} />**  **Offer Letter**  **</label>**  **<label>**  **<input type="checkbox" name="proofs[joining\_letter]" checked={formData.proofs.joining\_letter} onChange={handleChange} />**  **Joining Letter**  **</label>**  **<label>**  **<input type="checkbox" name="proofs[completion\_certificate]" checked={formData.proofs.completion\_certificate} onChange={handleChange} />**  **Completion Certificate**  **</label>**  **<label>**  **<input type="checkbox" name="proofs[report]" checked={formData.proofs.report} onChange={handleChange} />**  **Internship Report**  **</label>**  **</div>**  **</fieldset>**  **<button type="submit">Submit</button>**  **</form>**  **{message && (**  **<div id="message" className={isError ? 'error' : ''}>**  **{message}**  **</div>**  **)}**  **<button type="button" onClick={goToHome} style={{ marginTop: '20px' }}>**  **Back to Student Home**  **</button>**  **</div>**  **);**  **}**  **export default AddInternship;** |

|  |
| --- |
| **EditInternship.js**  **import React, { useState } from "react";**  **import api from "../../utils/api";**  **import { useNavigate } from "react-router-dom";**  **export default function EditInternship() {**  **const navigate = useNavigate();**  **const [regNo, setRegNo] = useState("");**  **const [formData, setFormData] = useState(null);**  **const [message, setMessage] = useState("");**  **const fetchData = async () => {**  **try {**  **const res = await api.get("/");**  **const student = res.data.find((s) => s.reg\_no === regNo);**  **if (student) {**  **setFormData(student);**  **setMessage("");**  **} else {**  **setFormData(null);**  **setMessage("Student not found");**  **}**  **} catch (err) {**  **console.error(err);**  **setMessage("Error fetching data");**  **}**  **};**  **const handleChange = (e) => {**  **const { name, value, type, checked } = e.target;**  **if (name.startsWith("proofs")) {**  **const key = name.split(".")[1];**  **setFormData((prev) => ({**  **...prev,**  **proofs: {**  **...prev.proofs,**  **[key]: checked,**  **},**  **}));**  **} else if (name === "location") {**  **setFormData((prev) => ({ ...prev, location: value }));**  **} else {**  **setFormData((prev) => ({ ...prev, [name]: value }));**  **}**  **};**  **const handleUpdate = async () => {**  **try {**  **await api.put(`/${formData.\_id}`, formData);**  **setMessage("Details updated successfully");**  **} catch (err) {**  **console.error(err);**  **setMessage("Error updating details");**  **}**  **};**  **const goToHome = () => {**  **navigate("/student/home");**  **};**  **return (**  **<div style={{ padding: "20px" }}>**  **{!formData ? (**  **<div>**  **<input**  **type="text"**  **placeholder="Enter Registration Number"**  **value={regNo}**  **onChange={(e) => setRegNo(e.target.value)}**  **style={{ border: "1px solid #ccc", padding: "8px" }}**  **/>**  **<button**  **onClick={fetchData}**  **style={{**  **marginLeft: "10px",**  **padding: "8px 16px",**  **backgroundColor: "#9b59b6",**  **color: "white",**  **border: "none",**  **}}**  **>**  **Fetch**  **</button>**  **{message && (**  **<p style={{ marginTop: "10px" }}>{message}</p>**  **)}**  **</div>**  **) : (**  **<form style={{ marginTop: "20px" }}>**  **<h2 style={{ fontSize: "20px", fontWeight: "bold" }}>**  **Edit Internship Details**  **</h2>**  **<div>**  **<label>Name</label>**  **<input**  **name="name"**  **value={formData.name}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Internship Title</label>**  **<input**  **name="title"**  **value={formData.title}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Mobile Number</label>**  **<input**  **name="mobile\_no"**  **value={formData.mobile\_no}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Section</label>**  **<input**  **name="section"**  **value={formData.section}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Internship Period</label>**  **<input**  **name="period"**  **value={formData.period}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Start Date</label>**  **<input**  **name="start\_date"**  **type="date"**  **value={formData.start\_date}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>End Date</label>**  **<input**  **name="end\_date"**  **type="date"**  **value={formData.end\_date}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Company Name</label>**  **<input**  **name="company\_name"**  **value={formData.company\_name}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>GDrive Folder Link</label>**  **<input**  **name="gdrive\_folder\_link"**  **value={formData.gdrive\_folder\_link}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Obtained Internship</label>**  **<select**  **name="obtained\_internship"**  **value={formData.obtained\_internship}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **>**  **<option value="">Select Internship Status</option>**  **<option value="Yes">Yes</option>**  **<option value="No">No</option>**  **</select>**  **</div>**  **<div>**  **<label>Placement Source</label>**  **<select**  **name="placement\_source"**  **value={formData.placement\_source}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **>**  **<option value="">Select Source</option>**  **<option value="College">College</option>**  **<option value="Own">Own</option>**  **</select>**  **</div>**  **<div>**  **<label>Stipend</label>**  **<input**  **name="stipend"**  **type="number"**  **value={formData.stipend}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **/>**  **</div>**  **<div>**  **<label>Internship Type</label>**  **<select**  **name="internship\_type"**  **value={formData.internship\_type}**  **onChange={handleChange}**  **style={{**  **display: "block",**  **border: "1px solid #ccc",**  **padding: "8px",**  **width: "100%",**  **}}**  **>**  **<option value="">Select Type</option>**  **<option value="Academics">Academics</option>**  **<option value="Industrial">Industrial</option>**  **</select>**  **</div>**  **<div>**  **<label>Internship Location</label>**  **<div>**  **<label style={{ marginRight: "20px" }}>**  **<input**  **type="radio"**  **name="location"**  **value="India"**  **checked={formData.location === "India"}**  **onChange={handleChange}**  **/>**  **India**  **</label>**  **<label>**  **<input**  **type="radio"**  **name="location"**  **value="Abroad"**  **checked={formData.location === "Abroad"}**  **onChange={handleChange}**  **/>**  **Abroad**  **</label>**  **</div>**  **</div>**  **<div>**  **<label style={{ fontWeight: "bold" }}>Proofs Submitted:</label>**  **<label>**  **<input**  **type="checkbox"**  **name="proofs.offer\_letter"**  **checked={formData.proofs?.offer\_letter || false}**  **onChange={handleChange}**  **/>**  **Offer Letter**  **</label>**  **<br />**  **<label>**  **<input**  **type="checkbox"**  **name="proofs.joining\_letter"**  **checked={formData.proofs?.joining\_letter || false}**  **onChange={handleChange}**  **/>**  **Joining Letter**  **</label>**  **<br />**  **<label>**  **<input**  **type="checkbox"**  **name="proofs.completion\_certificate"**  **checked={formData.proofs?.completion\_certificate || false}**  **onChange={handleChange}**  **/>**  **Completion Certificate**  **</label>**  **<br />**  **<label>**  **<input**  **type="checkbox"**  **name="proofs.report"**  **checked={formData.proofs?.report || false}**  **onChange={handleChange}**  **/>**  **Internship Report**  **</label>**  **</div>**  **<button**  **type="button"**  **onClick={handleUpdate}**  **style={{**  **marginTop: "20px",**  **padding: "8px 16px",**  **backgroundColor: "#9b59b6",**  **color: "white",**  **border: "none",**  **}}**  **>**  **Update**  **</button>**  **{message && (**  **<p style={{ marginTop: "10px" }}>{message}</p>**  **)}**  **</form>**  **)}**  **<button**  **type="button"**  **onClick={goToHome}**  **style={{ marginTop: "20px", padding: "8px 16px", backgroundColor: "#9b59b6", color: "white", border: "none" }}**  **>**  **Back to Student Home**  **</button>**  **</div>**  **);**  **}** |

|  |
| --- |
| **CoordinatorHome.js**  **import React from "react";**  **import { useNavigate } from "react-router-dom";**  **export default function CoordinatorHome() {**  **const navigate = useNavigate();**  **const handleSignOut = () => {**  **localStorage.removeItem("student\_token");**  **navigate("/");**  **};**  **return (**  **<div className="container">**  **<h2 className="page-title">Coordinator Portal</h2>**  **<button**  **onClick={() => navigate("/coordinator/view")}**  **>**  **Show Internship Details**  **</button>**  **<button**  **onClick={() => navigate("/coordinator/delete/")}**  **>**  **Delete a Internship Detail**  **</button>**  **<button**  **onClick={handleSignOut}**  **>**  **Sign Out**  **</button>**  **</div>**  **);**  **}** |

|  |
| --- |
| **ViewAll.js**  **import React, { useState } from "react";**  **import { useNavigate } from "react-router-dom";**  **import api from "../../utils/api";**  **import "./Coordinator.css";**  **export default function Coordinator() {**  **const [students, setStudents] = useState([]);**  **const [allStudents, setAllStudents] = useState([]);**  **const [search, setSearch] = useState("");**  **const [message, setMessage] = useState("");**  **const navigate = useNavigate(); // Hook for navigation**  **const fetchData = async () => {**  **if (!search.trim()) return;**  **try {**  **const res = await api.get("/");**  **const result = res.data.find((s) => s.reg\_no === search.trim());**  **if (result) {**  **setStudents([result]);**  **setMessage("");**  **} else {**  **setStudents([]);**  **setMessage("No student found.");**  **}**  **} catch (err) {**  **console.error(err);**  **setMessage("Error fetching data.");**  **}**  **};**  **const displayAll = async () => {**  **try {**  **const res = await api.get("/");**  **setStudents(res.data);**  **setAllStudents(res.data); // save for filtering**  **setMessage("");**  **} catch (err) {**  **console.error(err);**  **setMessage("Error fetching data.");**  **}**  **};**  **const filterBySource = (source) => {**  **const filtered = allStudents.filter(s => s.placement\_source?.toLowerCase() === source.toLowerCase());**  **setStudents(filtered);**  **};**  **const filterByLocation = (locationType) => {**  **const filtered = allStudents.filter(s =>**  **locationType === "India"**  **? s.location?.toLowerCase() === "india"**  **: s.location?.toLowerCase() !== "india"**  **);**  **setStudents(filtered);**  **};**  **const filterByCompany = () => {**  **const company = prompt("Enter company name:");**  **if (!company) return;**  **const filtered = allStudents.filter(s =>**  **s.company\_name?.toLowerCase().includes(company.toLowerCase())**  **);**  **setStudents(filtered);**  **};**  **return (**  **<div>**  **<h2>Coordinator - Internship Records</h2>**  **<div className="container">**  **<input**  **type="text"**  **value={search}**  **onChange={(e) => setSearch(e.target.value)}**  **placeholder="Enter Registration Number"**  **/>**  **<button onClick={fetchData}>Search</button>**  **<button onClick={displayAll}>View All Students</button>**  **{/\* Additional Filters \*/}**  **<button onClick={() => filterBySource("College")}>From College</button>**  **<button onClick={() => filterBySource("Own")}>From Own</button>**  **<button onClick={() => filterByLocation("India")}>Internship in India</button>**  **<button onClick={() => filterByLocation("Abroad")}>Internship in Abroad</button>**  **<button onClick={filterByCompany}>Search based on Company</button>**  **{/\* Navigation Button \*/}**  **<button onClick={() => navigate("/coordinator/home")}>Back to Home</button>**  **</div>**  **{message && <p style={{ color: "red", marginBottom: "10px" }}>{message}</p>}**  **<table>**  **<thead>**  **<tr>**  **<th>Reg No</th>**  **<th>Name</th>**  **<th>Title</th>**  **<th>Mobile No</th>**  **<th>Section</th>**  **<th>Obtained Internship?</th>**  **<th>Period</th>**  **<th>Start Date</th>**  **<th>End Date</th>**  **<th>Company Name</th>**  **<th>Placement Source</th>**  **<th>Stipend</th>**  **<th>Internship Type</th>**  **<th>Location</th>**  **<th>Proof Submissions</th>**  **<th>Offer Letter?</th>**  **<th>Joining Letter?</th>**  **<th>Completion Certificate?</th>**  **<th>Report?</th>**  **</tr>**  **</thead>**  **<tbody>**  **{students.map((s) => (**  **<tr key={s.\_id}>**  **<td>{s.reg\_no}</td>**  **<td>{s.name}</td>**  **<td>{s.title}</td>**  **<td>{s.mobile\_no}</td>**  **<td>{s.section}</td>**  **<td>{s.obtained\_internship}</td>**  **<td>{s.period}</td>**  **<td>{s.start\_date}</td>**  **<td>{s.end\_date}</td>**  **<td>{s.company\_name}</td>**  **<td>{s.placement\_source}</td>**  **<td>{s.stipend}</td>**  **<td>{s.internship\_type}</td>**  **<td>{s.location}</td>**  **<td>**  **<a href={s.gdrive\_folder\_link} target="\_blank" rel="noreferrer">**  **View**  **</a>**  **</td>**  **<td>{s.proofs?.offer\_letter ? "Yes" : "No"}</td>**  **<td>{s.proofs?.joining\_letter ? "Yes" : "No"}</td>**  **<td>{s.proofs?.completion\_certificate ? "Yes" : "No"}</td>**  **<td>{s.proofs?.report ? "Yes" : "No"}</td>**  **</tr>**  **))}**  **</tbody>**  **</table>**  **</div>**  **);**  **}** |

|  |
| --- |
| **DeleteRecord.js**  **import React, { useState } from "react";**  **import { useNavigate } from "react-router-dom";**  **import api from "../../utils/api";**  **export default function DeleteRecord() {**  **const [regNo, setRegNo] = useState("");**  **const navigate = useNavigate(); // for navigation**  **const handleDelete = async () => {**  **try {**  **const res = await api.get("/");**  **const match = res.data.find((s) => s.reg\_no === regNo);**  **if (match) {**  **await api.delete(`/${match.\_id}`);**  **alert("Deleted");**  **setRegNo("");**  **} else {**  **alert("Student not found");**  **}**  **} catch (error) {**  **console.error("Error deleting record:", error);**  **alert("Something went wrong. Please try again.");**  **}**  **};**  **return (**  **<div style={{ padding: "20px", maxWidth: "400px", margin: "auto" }}>**  **<h2 style={{ marginBottom: "10px" }}>Delete an Internship Record</h2>**    **<input**  **placeholder="Enter Register Number"**  **value={regNo}**  **onChange={(e) => setRegNo(e.target.value)}**  **style={{**  **width: "100%",**  **padding: "10px",**  **borderRadius: "4px",**  **border: "1px solid #ccc",**  **marginBottom: "10px",**  **fontSize: "16px",**  **}}**  **/>**  **<button**  **onClick={handleDelete}**  **style={{**  **padding: "10px 20px",**  **backgroundColor: "#9b59b6",**  **color: "white",**  **border: "none",**  **borderRadius: "4px",**  **fontSize: "16px",**  **cursor: "pointer",**  **marginRight: "10px",**  **}}**  **>**  **Delete**  **</button>**  **<button**  **onClick={() => navigate("/coordinator/home")}**  **style={{**  **padding: "10px 20px",**  **backgroundColor: "#9b59b6",**  **color: "white",**  **border: "none",**  **borderRadius: "4px",**  **fontSize: "16px",**  **cursor: "pointer",**  **}}**  **>**  **Back to Home**  **</button>**  **</div>**  **);**  **}** |

|  |
| --- |
| **style.css**  **/\* General Body Styling \*/**  **body {**  **font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;**  **background: linear-gradient(to right, #fdf7ff, #e8dbf5);**  **margin: 0;**  **padding: 20px;**  **color: #3c2a4d;**  **}**  **/\* Wrapper for Centered Pages \*/**  **.wrapper {**  **display: flex;**  **justify-content: center;**  **align-items: center;**  **flex-direction: column;**  **min-height: 100vh;**  **padding: 40px 20px;**  **}**  **/\* Card / Container Styling \*/**  **.container,**  **.login-container {**  **background: #ffffff;**  **padding: 40px;**  **border-radius: 12px;**  **max-width: 700px;**  **width: 100%;**  **margin: 40px auto;**  **box-shadow: 0 8px 24px rgba(106, 76, 156, 0.1);**  **}**  **/\* Titles and Headings \*/**  **h2,**  **.login-title {**  **text-align: center;**  **font-size: 30px;**  **color: #6a4c9c;**  **margin-bottom: 30px;**  **font-weight: 600;**  **}**  **/\* Form Styling \*/**  **form {**  **width: 100%;**  **max-width: 600px;**  **background: #f9f6fb;**  **padding: 35px;**  **border-radius: 12px;**  **font-size: 16px;**  **box-shadow: 0 4px 20px rgba(155, 89, 182, 0.1);**  **}**  **/\* Labels \*/**  **label {**  **font-weight: 500;**  **color: #5d3b8c;**  **display: block;**  **margin-bottom: 8px;**  **}**  **/\* Inputs and Selects \*/**  **input[type="text"],**  **input[type="url"],**  **input[type="password"],**  **input[type="email"],**  **input[type="number"],**  **input[type="date"],**  **select,**  **textarea {**  **width: 100%;**  **padding: 14px;**  **margin-bottom: 18px;**  **font-size: 15px;**  **border: 1px solid #d3cce3;**  **border-radius: 8px;**  **background-color: #fff;**  **transition: all 0.3s ease-in-out;**  **box-sizing: border-box;**  **}**  **input:focus,**  **select:focus,**  **textarea:focus {**  **border-color: #9b59b6;**  **box-shadow: 0 0 6px rgba(155, 89, 182, 0.25);**  **outline: none;**  **}**  **/\* Radio Buttons \*/**  **input[type="radio"] {**  **width: auto;**  **margin-right: 10px;**  **transform: scale(1.2);**  **}**  **/\* Buttons \*/**  **button,**  **.role-button,**  **.mode-button,**  **.submit-button {**  **width: 100%;**  **padding: 14px;**  **background: #9b59b6;**  **color: #ffffff;**  **font-size: 16px;**  **border: none;**  **border-radius: 8px;**  **cursor: pointer;**  **font-weight: 500;**  **transition: all 0.3s;**  **margin-bottom: 20px;**  **}**  **button:hover,**  **.role-button:hover,**  **.mode-button:hover,**  **.submit-button:hover {**  **background: #8e44ad;**  **opacity: 0.95;**  **}**  **button:disabled {**  **background: #d7c5e6;**  **cursor: not-allowed;**  **}**  **/\* Tables \*/**  **table {**  **width: 100%;**  **margin-top: 30px;**  **border-collapse: collapse;**  **background-color: #fff;**  **box-shadow: 0 4px 16px rgba(106, 76, 156, 0.08);**  **border-radius: 10px;**  **overflow: hidden;**  **}**  **th,**  **td {**  **padding: 14px 16px;**  **border: 1px solid #eee;**  **text-align: left;**  **font-size: 15px;**  **word-break: break-word;**  **}**  **thead {**  **background-color: #9b59b6;**  **color: white;**  **font-weight: bold;**  **}**  **tbody tr:nth-child(even) {**  **background-color: #f8f3fc;**  **}**  **tbody tr:hover {**  **background-color: #f0e5f9;**  **transition: 0.2s;**  **}**  **/\* Fieldset for Proofs \*/**  **.proofs-fieldset {**  **border: 2px solid #9b59b6;**  **padding: 20px;**  **border-radius: 10px;**  **margin-bottom: 25px;**  **background-color: #fdf7ff;**  **}**  **.proofs-fieldset legend {**  **font-weight: bold;**  **padding: 0 12px;**  **font-size: 18px;**  **color: #9b59b6;**  **}**  **/\* Checkboxes \*/**  **.proofs-container {**  **display: flex;**  **flex-wrap: wrap;**  **gap: 15px;**  **justify-content: flex-start;**  **margin-top: 10px;**  **}**  **.proofs-container input[type="checkbox"] {**  **margin-right: 8px;**  **transform: scale(1.1);**  **}**  **.proofs-container label {**  **font-size: 15px;**  **color: #5d3b8c;**  **}**  **/\* Messages \*/**  **#error,**  **#message {**  **font-size: 15px;**  **padding: 12px;**  **border-radius: 8px;**  **text-align: center;**  **margin-top: 20px;**  **}**  **#error {**  **background-color: #fdecea;**  **color: #c0392b;**  **}**  **#message {**  **background-color: #e9f7ef;**  **color: #2e7d32;**  **}**  **/\* Small screens \*/**  **@media screen and (max-width: 600px) {**  **.container,**  **.login-container,**  **form {**  **padding: 20px;**  **}**  **table {**  **font-size: 14px;**  **}**  **}** |

**Backend:**

|  |
| --- |
| **.env**  **PORT = 5000**  **URL = mongodb://localhost:27017/intern-app** |

|  |
| --- |
| **index.js**  **import express from 'express';**  **import mongoose from 'mongoose';**  **import dotenv from 'dotenv';**  **import cors from 'cors';**  **import Student from './models/studentModel.js';**  **import studentRoutes from './routes/studentRoute.js';**  **import studentAuthRoutes from './routes/studentAuth.js';**  **import coordinatorAuthRoutes from './routes/coordinatorAuth.js';**  **const app = express();**  **dotenv.config();**  **// Middleware**  **app.use(cors());**  **app.use(express.json());**  **// Routes**  **app.use('/api/students', studentRoutes);**  **app.use('/api/auth', studentAuthRoutes);**  **app.use("/api/coordinator", coordinatorAuthRoutes);**  **// Root route (optional)**  **app.get('/', (req, res) => {**  **res.send('InternTrack API is running...');**  **});**  **app.get('/api/students', async (req, res) => {**  **try {**  **const students = await Student.find({});**  **res.json(students);**  **} catch (err) {**  **res.status(500).json({ message: 'Server Error' });**  **}**  **});**  **// Connect to MongoDB and start server**  **const PORT = process.env.PORT || 5000;**  **const MONGO\_URL = process.env.URL;**  **mongoose**  **.connect(MONGO\_URL, { useNewUrlParser: true, useUnifiedTopology: true })**  **.then(() => {**  **console.log(' MongoDB connected');**  **app.listen(PORT, () => console.log(`Server running on port ${PORT}`));**  **})**  **.catch((err) => {**  **console.error(' MongoDB connection error:', err.message);**  **});** |

|  |
| --- |
| **StudentModel.js**  **import mongoose from "mongoose";**  **const proofSchema = new mongoose.Schema({**  **offer\_letter: Boolean,**  **joining\_letter: Boolean,**  **completion\_certificate: Boolean,**  **report: Boolean,**  **});**  **const studentSchema = new mongoose.Schema({**  **reg\_no: { type: String, required: true, unique: true },**  **name: String,**  **title: String,**  **mobile\_no: String,**  **section: String,**  **obtained\_internship: String,**  **period: String,**  **start\_date: Date,**  **end\_date: Date,**  **company\_name: String,**  **placement\_source: String,**  **stipend: Number,**  **internship\_type: String,**  **location: String,**  **gdrive\_folder\_link: String,**  **proofs: proofSchema,**  **});**  **export default mongoose.model("Student", studentSchema);** |

|  |
| --- |
| **StudentSignup.js**  **import mongoose from "mongoose";**  **const studentLoginSchema = new mongoose.Schema({**  **reg\_no: {**  **type: String,**  **required: true,**  **unique: true,**  **},**  **password: {**  **type: String,**  **required: true,**  **}**  **})**  **// Use export default instead of module.exports**  **export default mongoose.model('StudentLogin', studentLoginSchema, 'students\_login');** |

|  |
| --- |
| **CoordinatorModel.js**  **import mongoose from "mongoose";**  **const coordinatorSchema = new mongoose.Schema({**  **email: {**  **type: String,**  **required: true,**  **unique: true,**  **lowercase: true**  **},**  **password: {**  **type: String,**  **required: true**  **}**  **});**  **const Coordinator = mongoose.model("Coordinator", coordinatorSchema);**  **export default Coordinator;** |

|  |
| --- |
| **studentController.js**  **import Student from "../models/studentModel.js";**  **import StudentLogin from "../models/studentSignup.js"**  **export const signupStudent = async (req, res) => {**  **const { reg\_no, password } = req.body;**  **try {**  **// Check if reg\_no already exists**  **const existing = await StudentLogin.findOne({ reg\_no });**  **if (existing) {**  **return res.status(409).json({ message: 'User already exists.' });**  **}**  **// Create new login entry (hash password in production)**  **const newLogin = new StudentLogin({ reg\_no, password });**  **await newLogin.save();**  **res.status(201).json({ message: 'Signup successful.' });**  **} catch (err) {**  **console.error(err);**  **res.status(500).json({ message: 'Server Error during signup.' });**  **}**  **};**  **export const createStudent = async (req, res) => {**  **const { reg\_no } = req.body;**  **try {**  **// Authorization check**  **const loginExists = await StudentLogin.findOne({ reg\_no: reg\_no });**  **if (!loginExists) {**  **return res.status(403).json({ message: 'You are not authorized to submit this form.' });**  **}**  **// Duplication check**  **const alreadySubmitted = await Student.findOne({ reg\_no: reg\_no });**  **if (alreadySubmitted) {**  **return res.status(409).json({ message: 'You have already submitted your internship details.' });**  **}**  **const newStudent = new Student(req.body);**  **await newStudent.save();**  **res.status(201).json({ message: 'Internship record submitted successfully.' });**  **} catch (err) {**  **console.error(err);**  **res.status(500).json({ message: 'Internal Server Error' });**  **}**  **};**  **export const getStudents = async (req, res) => {**  **try {**  **const students = await Student.find();**  **res.status(200).json(students);**  **} catch (err) {**  **res.status(500).json({ message: err.message });**  **}**  **};**  **export const getStudentById = async (req, res) => {**  **try {**  **const student = await Student.findById(req.params.id);**  **if (!student) return res.status(404).json({ message: "Not found" });**  **res.status(200).json(student);**  **} catch (err) {**  **res.status(500).json({ message: err.message });**  **}**  **};**  **export const updateStudent = async (req, res) => {**  **try {**  **const updated = await Student.findByIdAndUpdate(req.params.id, req.body, { new: true });**  **res.status(200).json(updated);**  **} catch (err) {**  **res.status(500).json({ message: err.message });**  **}**  **};**  **export const deleteStudent = async (req, res) => {**  **try {**  **await Student.findByIdAndDelete(req.params.id);**  **res.status(204).end();**  **} catch (err) {**  **res.status(500).json({ message: err.message });**  **}**  **};**  **export const filterStudents = async (req, res) => {**  **try {**  **const { location, placement\_source, company\_name, title, domain, period } = req.query;**  **const query = {};**  **if (location) query.location = location;**  **if (placement\_source) query.placement\_source = placement\_source;**  **if (company\_name) query.company\_name = company\_name;**  **if (title) query.title = title;**  **if (period) query.period = period;**  **const students = await Student.find(query);**  **res.json(students);**  **} catch (err) {**  **res.status(500).json({ message: err.message });**  **}**  **};** |

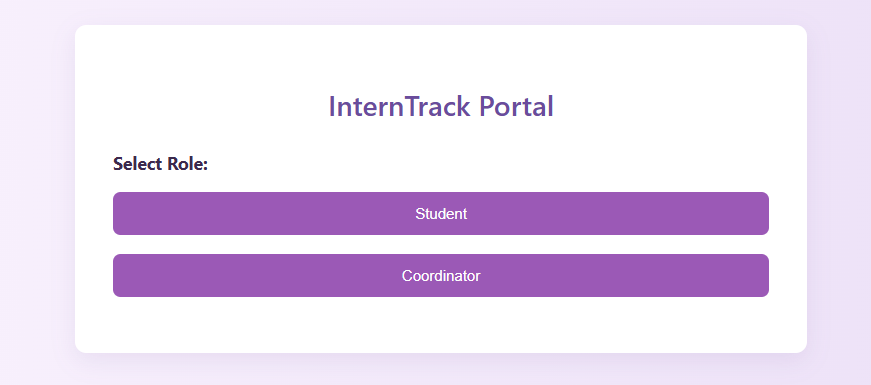
|  |
| --- |
| **StudentRoute.js**  **import express from 'express';**  **import {**  **createStudent,**  **getStudents,**  **getStudentById,**  **updateStudent,**  **deleteStudent,**  **filterStudents**  **} from '../controllers/studentController.js';**  **const router = express.Router();**  **router.post('/create', createStudent);**  **router.get('/', getStudents);**  **router.get('/:id', getStudentById);**  **router.put('/:id', updateStudent);**  **router.delete('/:id', deleteStudent);**  **router.get('/filter', filterStudents);**  **export default router;** |

|  |
| --- |
| **coordinatorAuth.js**  **import express from "express";**  **const router = express.Router();**  **import Coordinator from "../models/coordinatorModel.js";**  **// Coordinator Signup**  **router.post("/signup", async (req, res) => {**  **try {**  **const { email, password, name } = req.body;**  **const existing = await Coordinator.findOne({ email });**  **if (existing) return res.status(400).json({ success: false, message: "Coordinator already exists." });**  **const coordinator = new Coordinator({ email, password, name });**  **await coordinator.save();**  **res.status(201).json({ success: true, message: "Coordinator Signup Success" });**  **} catch (err) {**  **res.status(500).json({ success: false, message: err.message });**  **}**  **});**  **// Coordinator Login**  **router.post("/login", async (req, res) => {**  **try {**  **const { email, password } = req.body;**  **const coordinator = await Coordinator.findOne({ email });**  **if (!coordinator || coordinator.password !== password) {**  **return res.status(401).json({ success: false, message: "Invalid credentials" });**  **}**  **res.status(200).json({ success: true, message: "Login successful" });**  **} catch (err) {**  **res.status(500).json({ success: false, message: err.message });**  **}**  **});**  **export default router;** |

|  |
| --- |
| **studentAuth.js**  **import express from "express";**  **const router = express.Router();**  **import StudentLogin from "../models/studentSignup.js";**  **import Student from "../models/studentModel.js";**  **// Student Signup**  **router.post("/signup", async (req, res) => {**  **try {**  **const { reg\_no, password, name } = req.body;**  **const existing = await StudentLogin.findOne({ reg\_no });**  **if (existing) return res.status(400).json({ success: false, message: "Student already exists." });**  **const login = new StudentLogin({ reg\_no, password });**  **const student = new Student({ reg\_no, name });**  **await login.save();**  **await student.save();**  **res.status(201).json({ success: true, message: "Student Signup Success" });**  **} catch (err) {**  **res.status(500).json({ success: false, message: err.message });**  **}**  **});**  **// Student Login**  **router.post("/login", async (req, res) => {**  **try {**  **const { reg\_no, password } = req.body;**  **const student = await StudentLogin.findOne({ reg\_no });**  **if (!student || student.password !== password) {**  **return res.status(401).json({ success: false, message: "Invalid credentials" });**  **}**  **res.status(200).json({ success: true, message: "Login successful" });**  **} catch (err) {**  **res.status(500).json({ success: false, message: err.message });**  **}**  **});**  **export default router;** |

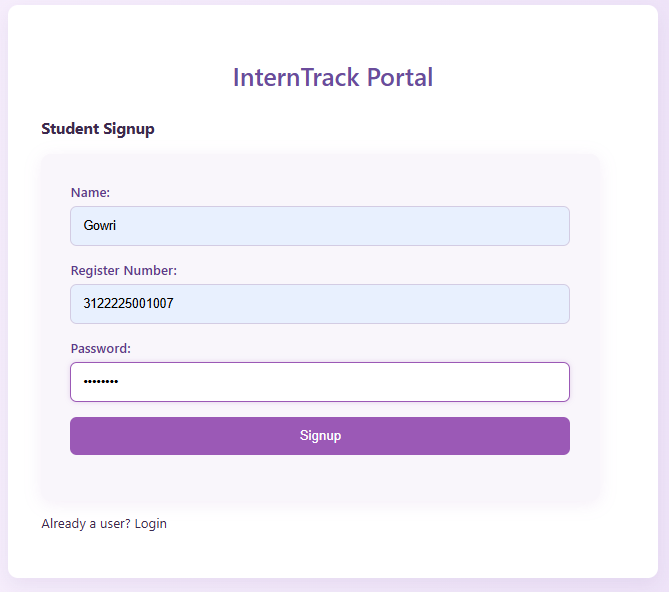
**Screenshots:**

**Login/ Signup page:**

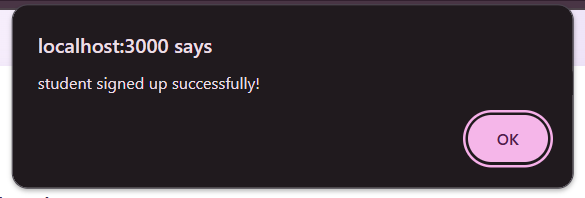
****

**A ) Lets see all the student functionality:**

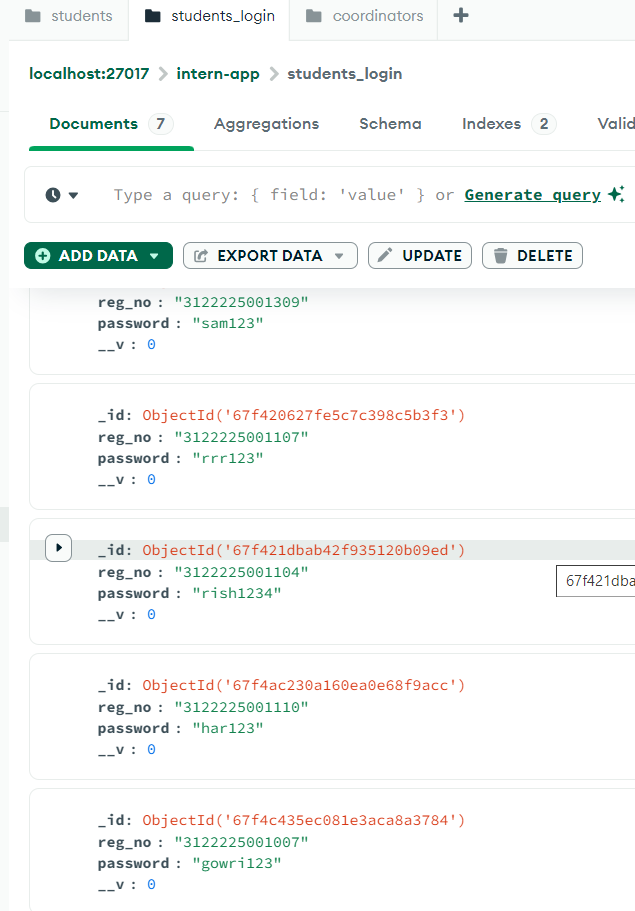
1. **Student Signup (on clicking New user? Sign up)**

****

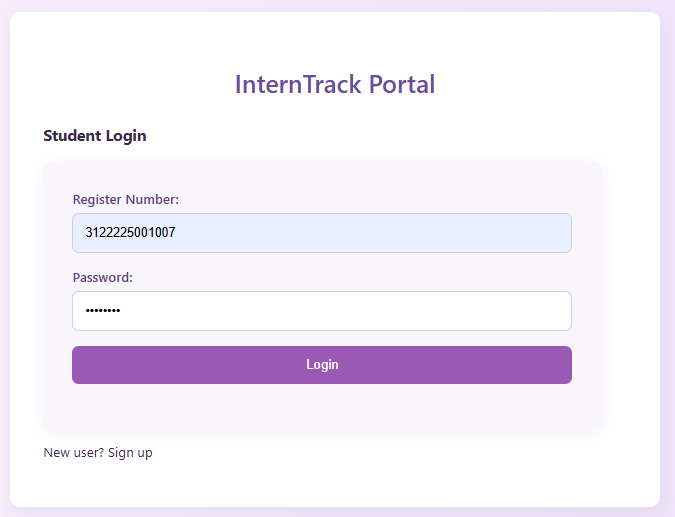
**On clicking Signup:**

****

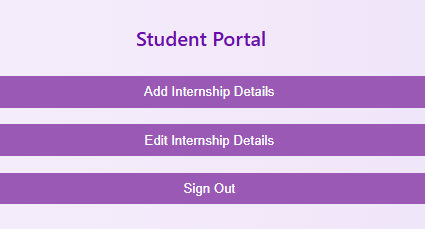
**Gowri added to students\_login**

****

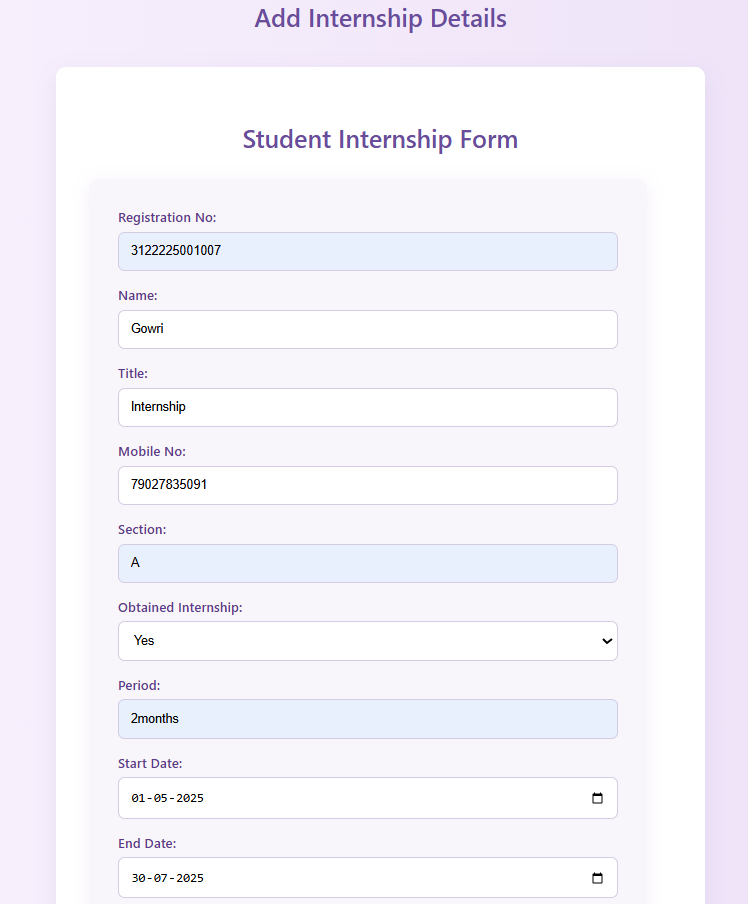
1. **Student Login**

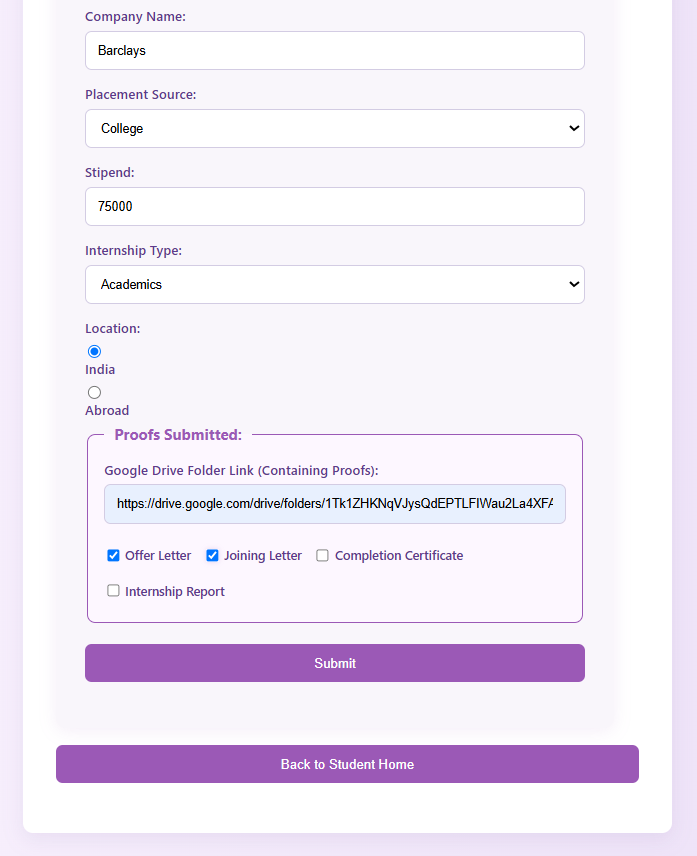
****

**On clicking Login**

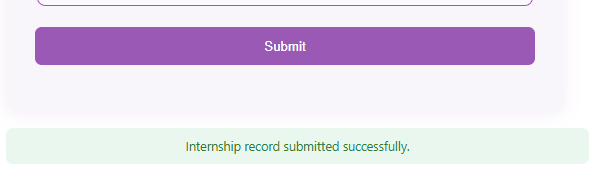
****

1. **Add Internship Details**

****

****

**On Clicking Submit:**

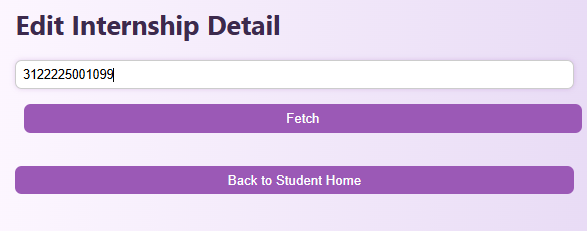
****

**Gowri added to students**

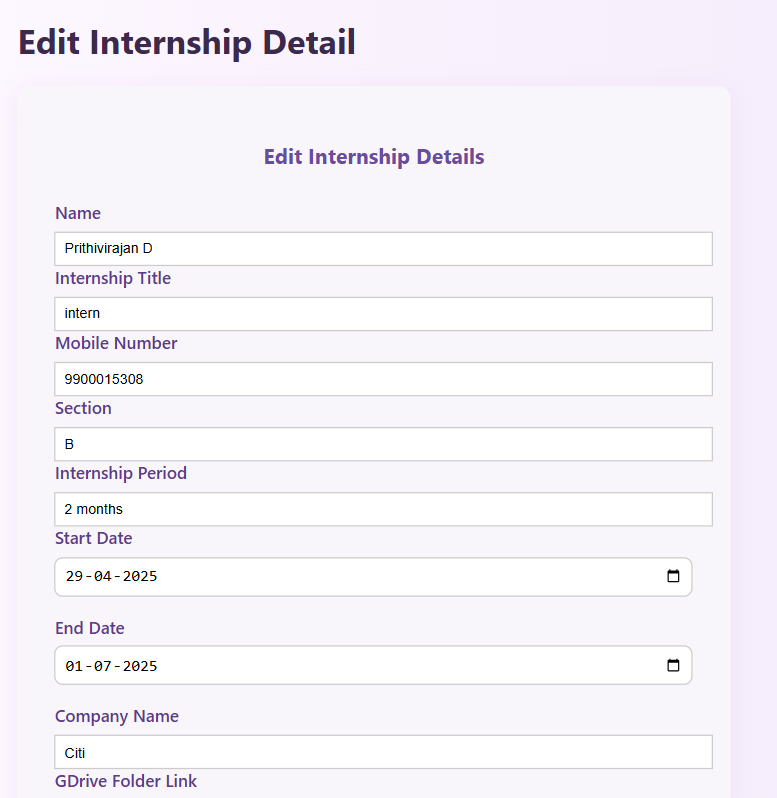
****

**Navigate back to StudentHome**

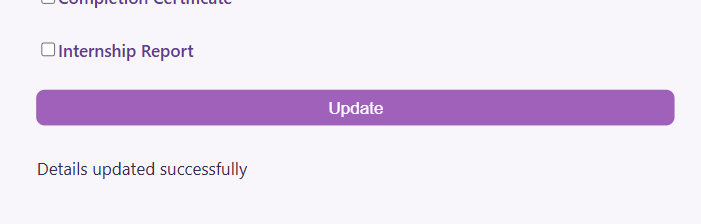
1. **Edit Internship:**

****

**On clicking Fetch: (editing period and start, end date)**

****

**On clicking update:**

****

**Before edit:**

****

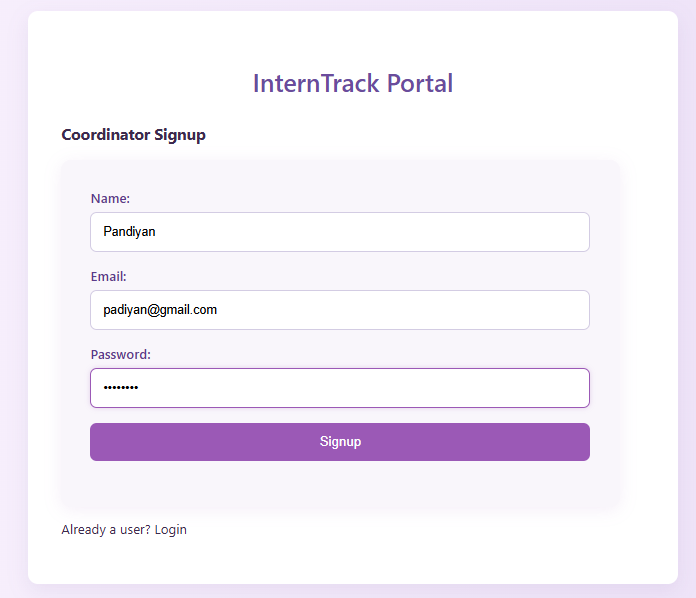
**After Edit:**

****

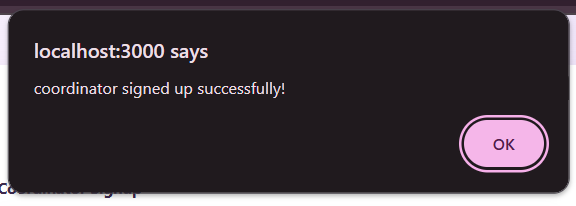
**Navigate to StudentHome and Signout.**

**B ) Lets see all the coordinator functionality:**

1. **Coordinator Signup:**

****

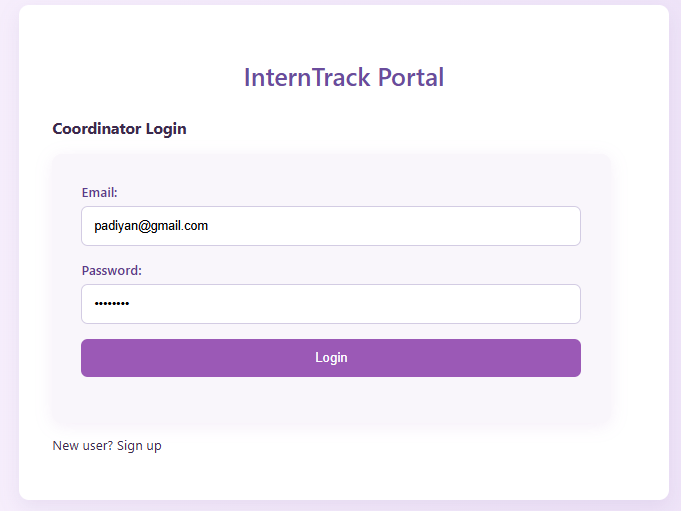
**On clicking Signup:**

****

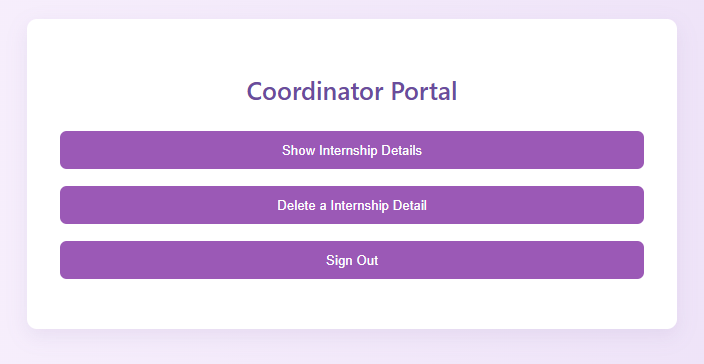
**Pandiyan added to coordinator\_signup:**

****

1. **Coordinator Login:**

****

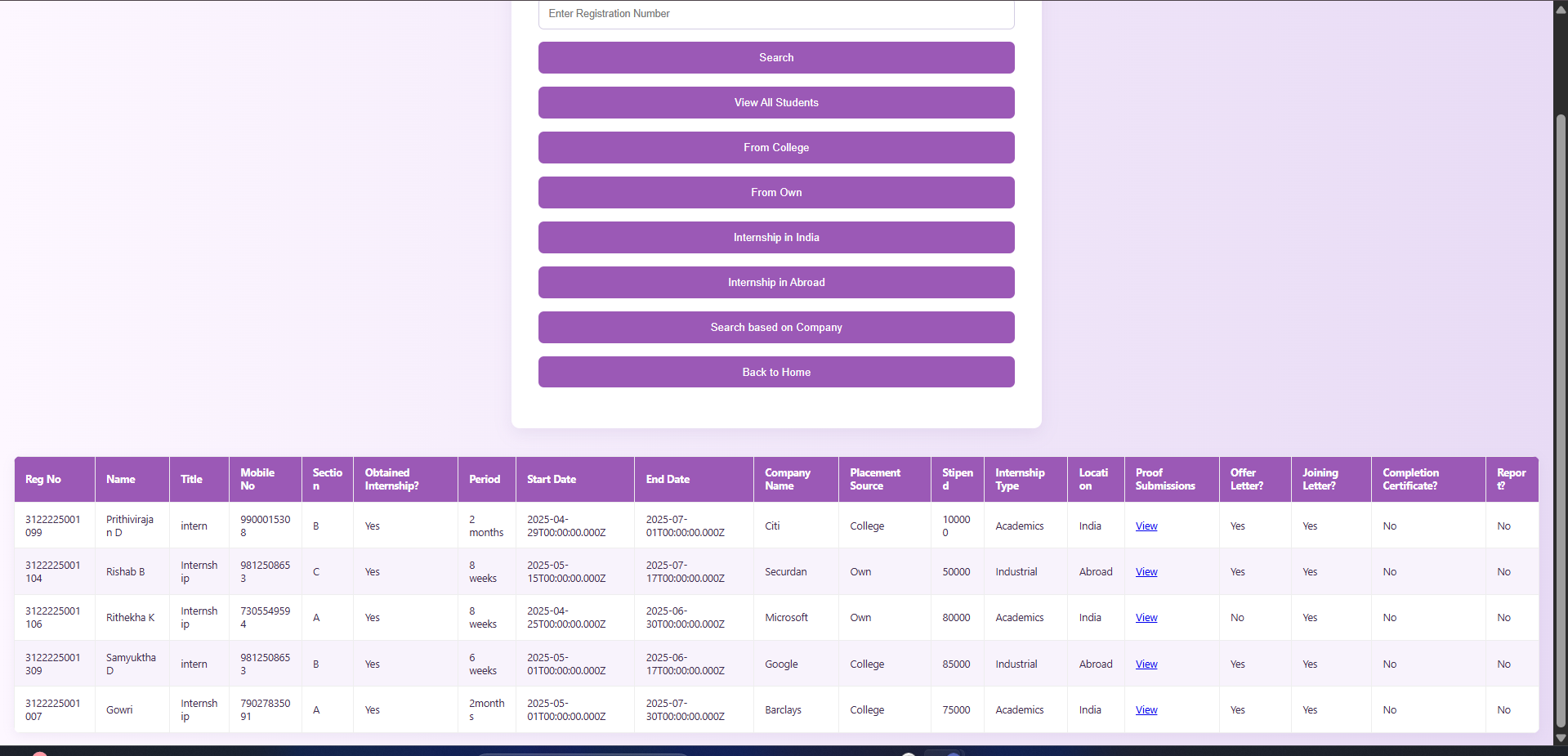
**Coordination page**

****

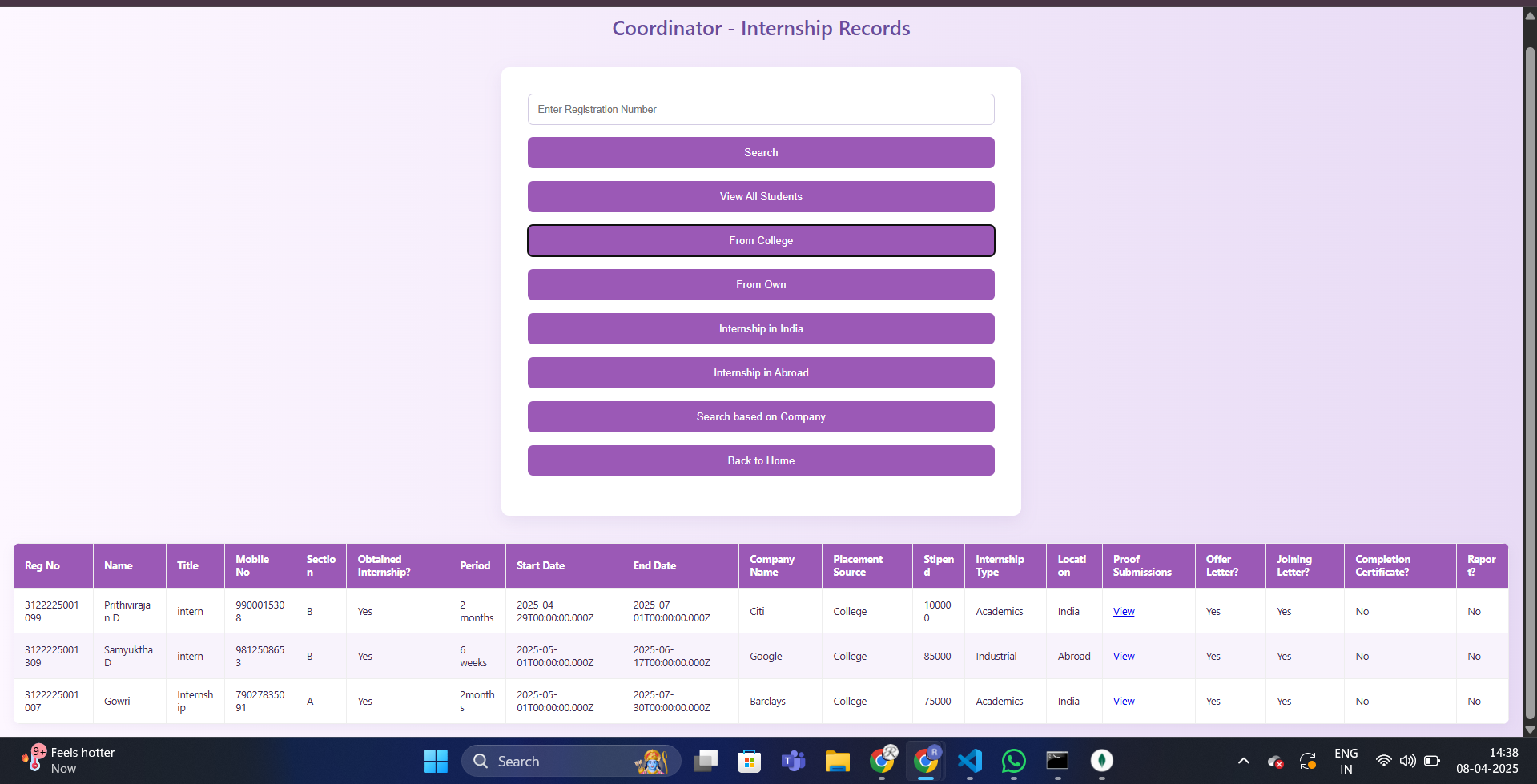
**3. Seach functionality**

1. **Search by reg no**

****

**b)View All Details**

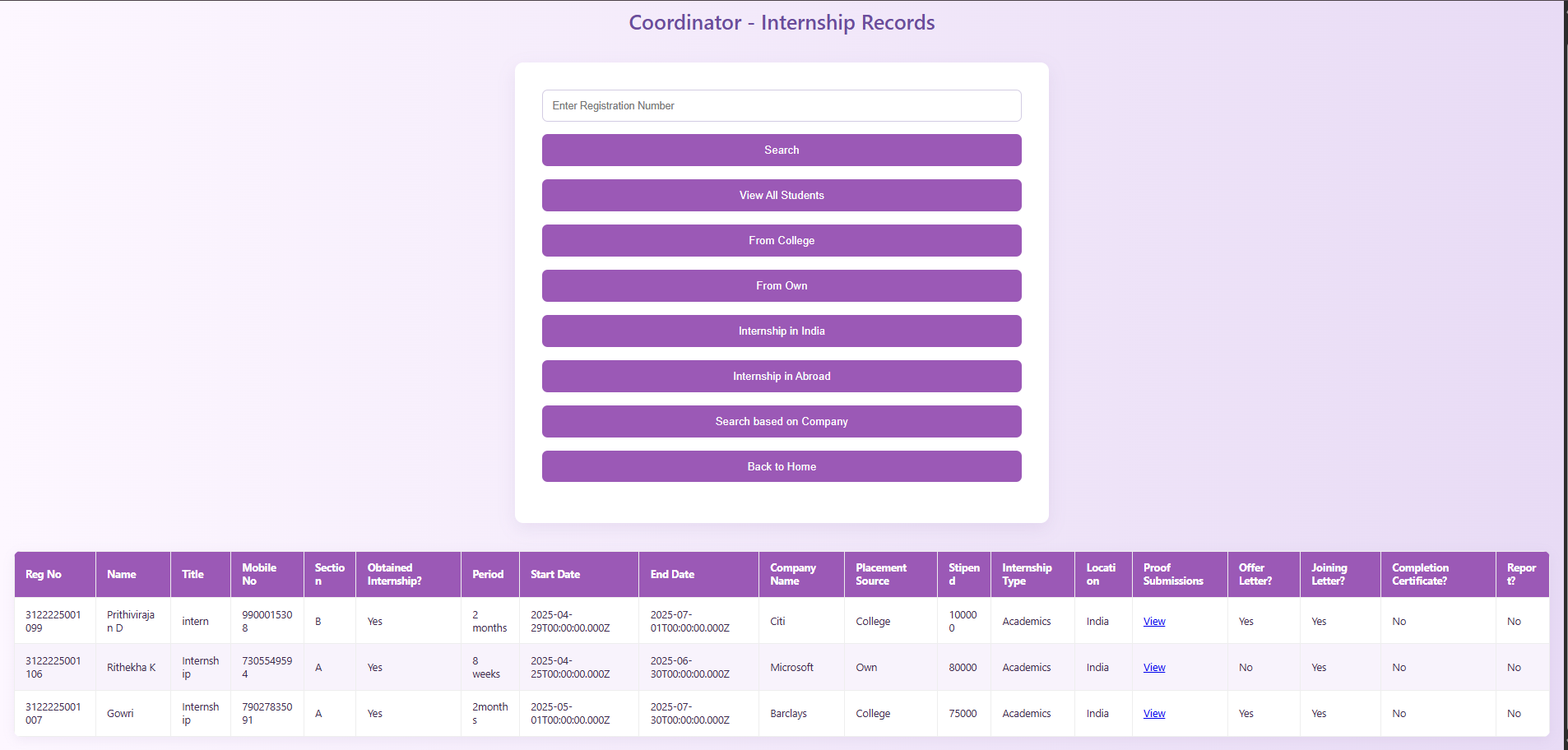
**c)View who got internship through College**

****

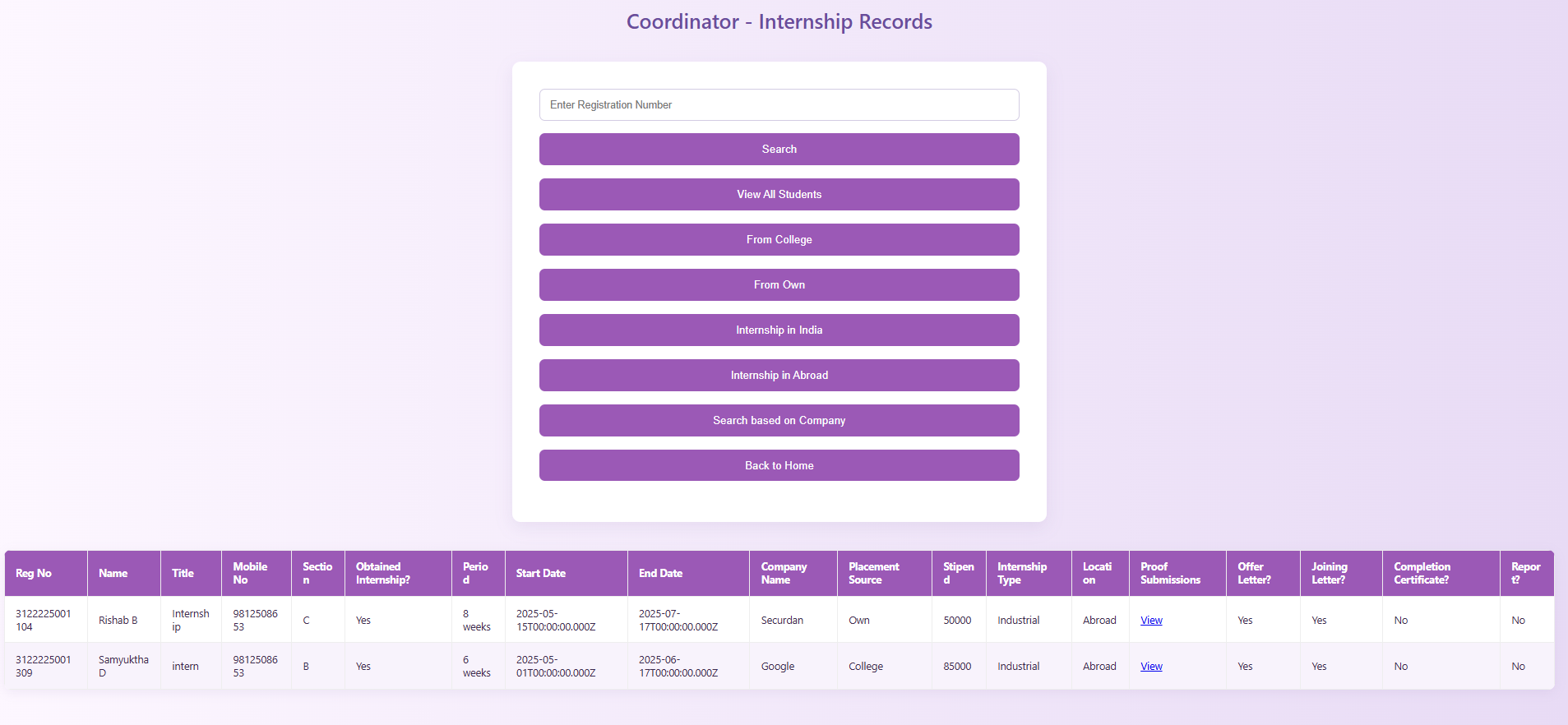
**d) View who got internship by their own**

****

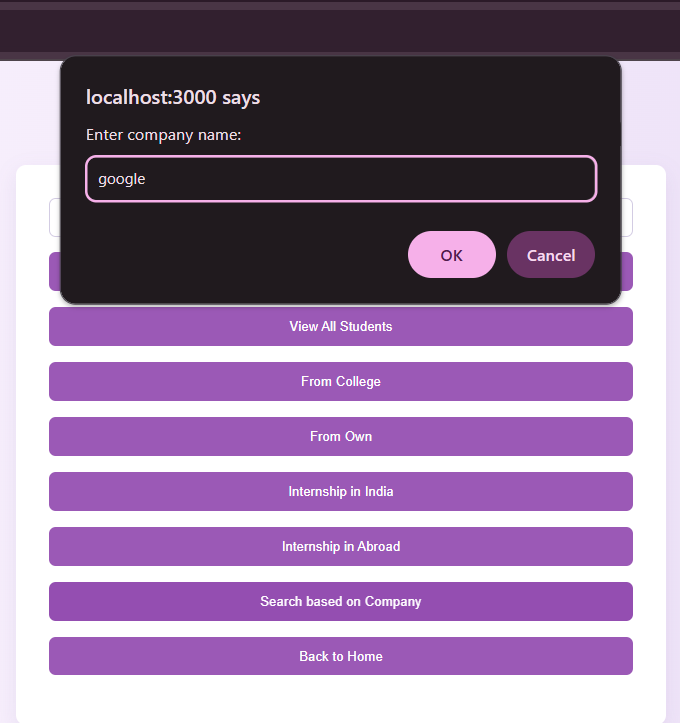
**e) View who got internship in India**

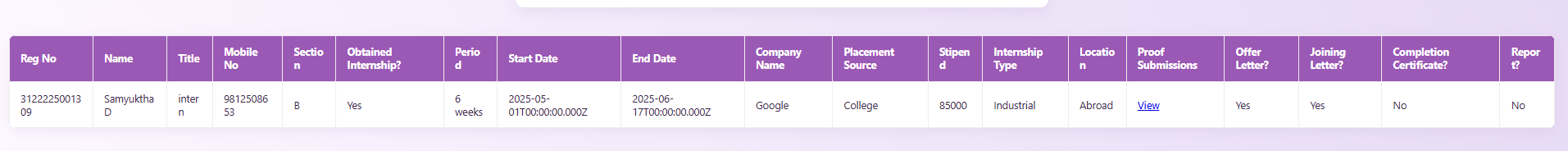
****

**f) View who got internship abroad**

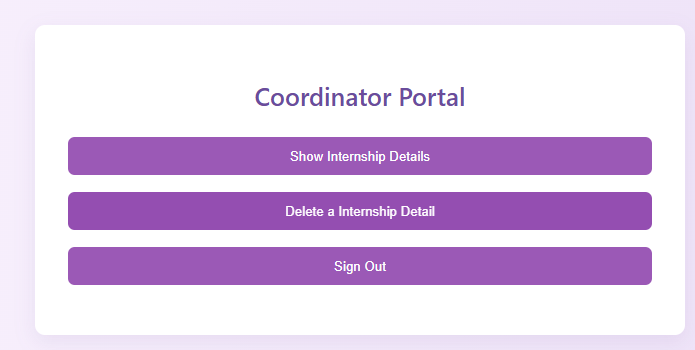
****

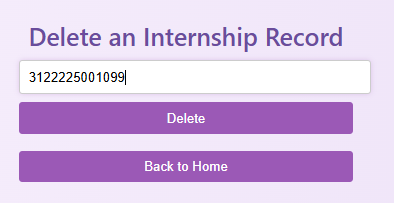
**g) Searching based on Company**

****

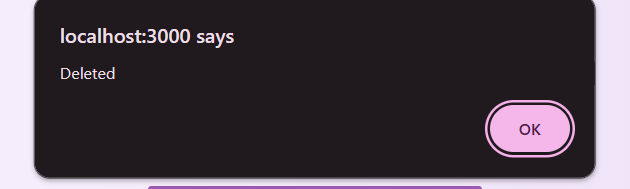
****

**4.Delete Functionality**

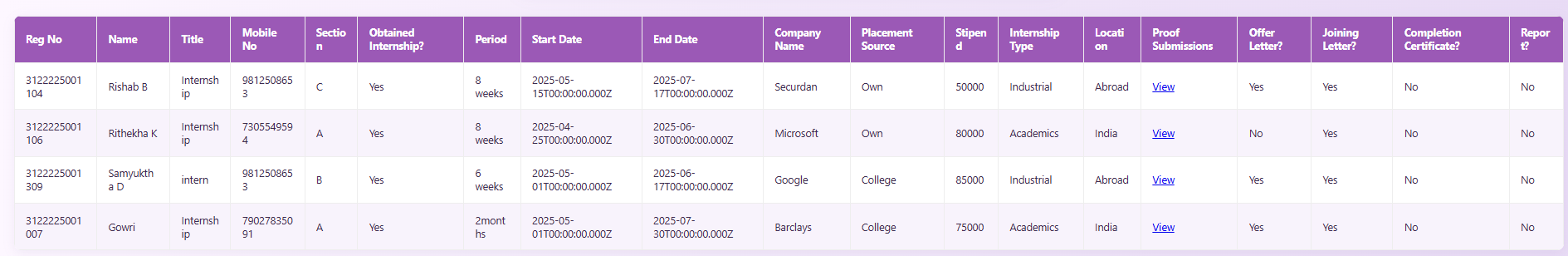
****

****

**3122225001099 records deleted**

****

**3122225001099 deleted successfully**

****

**Conclusion**

This project successfully demonstrates the development of a full-stack internship management portal using the MERN stack. It streamlines the process for coordinators to manage student internship records by providing essential functionalities such as search, filter, view, and delete through a secure and intuitive interface. The integration of technologies like JWT for authentication, MongoDB for flexible data storage, and React for a dynamic user experience ensures the system is both efficient and scalable.