Resume Builder

A PROJECT REPORT

For

"FULL STACK DEVELOPMENT"

Submitted by
G. Sri Vibhu - AP23110010932
M. Bhanu Teja - AP23110010800
T. Siddharth - AP23110011028

Submitted to:

Syed Arshad

SEM: V

SAP NTUT

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

SCHOOL OF ENGINEERING AND SCIENCES

SRM University-AP, Neerukonda, Andhra Pradesh 522240

December 2025

TABLE OF CONTENTS

1. Introduction 3
2. Scenario-Based Intro 4
3. Target Audience 5
4. Project Goals and Objectives 5
5. Key Features.. 6
6. Pre -Requisites 8
7. Project Structure 13
8. Project Flow 16
9. Project Execution... 19
10. Output... 20

1.Introduction

Step into the future of professional resume creation – a revolutionary resume-building experience awaits you with our state-of-the-art Resume Builder, intricately designed using the brilliance of React.js. Merging innovation with a user-centric approach, our application is poised to redefine how you create, customize, and manage professional resumes.

Crafted for modern job seekers and professionals, our React-based Resume Builder seamlessly integrates robust functionality with an intuitive user interface. From exploring multiple professional templates to creating ATS-friendly resumes, our platform ensures a comprehensive resume-building journey tailored to your unique career profile.

The core of our Resume Builder beats with React, a dynamic and feature-rich JavaScript library. Immerse yourself in a visually stunning and interactive interface, where every section, template selection, and real-time preview feels like a professional revelation. Whether you're on a desktop, tablet, or smartphone, our responsive design guarantees a consistent and enjoyable resume-building experience across all devices.

Bid farewell to the constraints of traditional resume creation tools and embrace a realm of possibilities with our React-based Resume Builder. Join us on this professional expedition as we transform the way you connect with and present your career achievements. Get ready to elevate your job application experience – it's time to build a resume that stands out.

2.Scenario-Based Intro

Imagine a typical day in the life of a recent graduate or a professional seeking new opportunities. You've just found a perfect job posting, and you need to submit your resume immediately. You open your favorite resume builder application, "Resume Builder."

As you begin your resume creation session, Resume Builder greets you with a clean, intuitive interface, customized to your preferences. The dark theme reduces eye strain during late-night editing sessions, while the real-time preview shows exactly how your resume will look to recruiters. You start by filling in your personal information, and as you type, the preview updates instantly, giving you immediate feedback.

Later, you need to add multiple work experiences and projects. With Resume Builder's dynamic form sections, you can add as many entries as needed with a single click. The editor's intelligent template system allows you to switch between Modern, Elegant, Creative, Professional, and Minimal designs without losing any of your content.

When you're satisfied with your resume, you simply click the download button, and your professionally formatted resume is ready to submit. The application automatically saves your work, so you can return anytime to update your resume for different job applications.

3.Target Audience

Resume Builder is tailored for a diverse community of individuals, including:

Job Seekers: Recent graduates, career changers, and professionals actively looking for new opportunities who need a quick and efficient way to create professional resumes.

Students: College and university students preparing for internships, part-time jobs, or their first full-time positions who want to present their academic achievements professionally.

Professionals: Experienced individuals updating their resumes for promotions, career transitions, or new opportunities who need a flexible and customizable resume solution.

Career Counselors: Educational institutions and career centers that provide resume-building assistance to students and alumni.

4.Project Goals and Objectives

The primary goal of Resume Builder is to offer a seamless platform for creating professional resumes, enabling an efficient and user-friendly resume-building experience. The objectives of the project include:

User-Friendly Interface: Creating an intuitive interface that allows users to navigate, edit, and manage their resumes effortlessly, ensuring a smooth and productive workflow.

Multiple Professional Templates: Offering five distinct resume templates (Modern, Elegant, Creative, Professional, and Minimal) that cater to different industries and personal preferences, allowing users to choose the design that fits their needs.

Real-Time Preview: Integrating a live preview system that updates instantly as users enter information, enabling them to see exactly how their resume will appear to recruiters.

Data Persistence: Allowing users to save multiple resumes, edit them later, and manage their saved versions through a dashboard with search and filter options.

Modern Tech Stack: Using technologies such as React.js, Tailwind CSS, and JSON Server to deliver an efficient, responsive, and engaging user experience throughout the application.

Export Functionality: Providing the ability to download resumes as PDF files, enabling users to export a clean, print-ready version suitable for job applications.

5.Key Features

Multiple Resume Templates: The application provides five professionally crafted templates (Modern, Elegant, Creative, Professional, and Minimal). Users can switch between templates instantly without losing any content, ensuring complete flexibility in design selection.

Real-Time Preview: The system offers a live, side-by-side preview panel that updates immediately as users type. This ensures users always know how their resume will look to recruiters, improving accuracy and design consistency.

Comprehensive Resume Sections: The resume builder includes dedicated sections for Personal Information and Contact Details, Professional Summary, Skills (covering Programming, Frameworks, Databases, Tools, and Soft Skills), Work Experience with multiple entries, Projects with links and descriptions, Education with CGPA support, Certifications, Achievements, and Languages. Each section is fully editable and designed to follow professional resume standards.

Save and Manage Resumes: Users can create and store multiple resume versions. A dedicated dashboard allows users to edit saved resumes at any time, organize them efficiently, and use search and template-based filters for quick access.

User Authentication: A secure login and signup system ensures personalized and protected access to user data. Each user receives a dedicated space for storing and managing their resumes.

Dark Mode Support: The interface includes a dark mode toggle that enhances comfort during long editing sessions and improves visibility in low-light environments.

Responsive Design: The application is fully responsive and optimized for desktop, laptop, tablet, and mobile devices. Users can build and edit resumes seamlessly across different screen sizes.

PDF Export: Users can export their resume in high-quality PDF format with proper A4 layout. The exported file is clean, professional, and ready to be submitted for job applications or printed without formatting issues.

Search and Filter: A robust search and filtering system allows users to quickly find saved resumes using name, email, title, or template type. This improves productivity and helps users manage multiple resume versions efficiently.

Additional Features: The resume builder also offers auto-save functionality to prevent data loss, customizable color highlights for template elements, and error prompts that guide users when required fields are incomplete. Built-in validation ensures every resume maintains a professional structure.

6.Pre -Requisites

Before running the Resume Builder Application, the following tools, software, and technical knowledge are required. Since the project uses React.js (Vite) for the frontend, Tailwind CSS for styling, JSON Server for mock backend storage, and React Router for navigation, the setup remains lightweight, efficient, and beginner-friendly.

Code Editor (VS Code Recommended)
A good code editor is necessary to write and manage the Resume Builder project files.

- Visual Studio Code is ideal due to JSX support, modern JavaScript features, and built-in terminal.
- Recommended VS Code extensions:
- Prettier – Code formatting
- ESLint – Detect JS/React code issues
- React Snippets – Quick React templates
- Tailwind CSS IntelliSense – Helps with Tailwind classes
- Live Server – Useful for testing static HTML/CSS files

Web Browser (Chrome / Firefox)
A modern browser is needed to run and debug your React-based Resume Builder app.

- Google Chrome is preferred for its DevTools and the React Developer Tools extension.
- Firefox/Edge can be used for cross-browser UI testing.

Node.js & npm (Mandatory for React + Vite)
Your Resume Builder relies on Node.js to run the development environment smoothly.

- Node.js provides the JavaScript runtime for React scripts.
- npm is required to install dependencies such as
- React
- React Router
- Tailwind CSS
- Axios
- React-Toastify
- React-to-Print
- JSON Server

Without Node.js and npm, the project cannot be created or executed.

React.js Framework Knowledge

Since the Resume Builder is built using React.js, developers should understand:

- Functional components with JSX
- Props and State management
- Hooks like useState, useEffect
- Component-based architecture for resume sections (Skills, Education, Experience, etc.)
- Routing between pages using React Router
- Form handling for capturing user input
- State lifting for sharing resume data between components
- Passing data to the preview section dynamically

This knowledge ensures smooth creation of reusable components and a responsive UI.

Vite (For Efficient React Project Setup)
The project uses Vite instead of Create React App for faster builds and instant HMR.

Developers should know:

Creating a project using:

```
npm create vite@latest
```

- Selecting React as the framework

Running the dev server:

```
npm run dev
```

Vite ensures a fast, modern development workflow.

Tailwind CSS Knowledge (For Resume Styling)

Tailwind CSS is used to design the resume templates.

- Developers should understand:
- Using utility classes (flex, grid, text-xl, p-4, etc.)
- Creating responsive layouts (sm, md, lg breakpoints)
- Customizing themes in tailwind.config.js
- Designing clean and modern resume UI sections quickly

Tailwind drastically speeds up UI development for resume templates.

JSON Server (Local Mini Backend for Storing User Data)

The project uses JSON Server to store user profile, skills, education, and experience locally.

Developers should know:

Running JSON Server:

```
npx json-server --watch db.json --port 5000
```

Performing CRUD operations using Axios:

- GET → Load resume data

- POST → Save user sections

- PUT/PATCH → Update resume content

- DELETE → Remove sections if needed

- Understanding JSON format for resume fields

JSON Server acts as a lightweight database—no SQL required.

Axios & API Handling

Axios is used to communicate with JSON Server.

Developers should know:

- Sending GET/POST/PUT/DELETE requests
- Handling promises and responses
- Error handling and alerts

(Toast messages using React-Toastify)

React-Toastify Knowledge

React-Toastify is used to show real-time notifications such as:

- “Resume Saved”
- “Section Added”
- “Error: Could Not Load Template”

Basic understanding of toast configuration is required.

React-to-Print (For Exporting Resume as PDF/Print)

This library allows users to print or export their resume.

Developers should know:

- Creating a reference to the resume preview
- Triggering printing via a button
- Ensuring styles are applied to the print view

Basic Skills in HTML, CSS, and JavaScript

Even though Tailwind and React handle most tasks, developers should know:

- JSX structure (HTML-like syntax inside JS)
- CSS utility classes
- JavaScript ES6+ features:

- Arrow functions
- async/await
- Array methods (map, filter)
- Template literals
- Building forms and handling input events

These skills are essential for creating dynamic resume templates and form components.

Version Control (Git + GitHub Optional)

Using Git helps track project changes and collaborate.

- Git installation recommended
- GitHub for hosting and sharing the Resume Builder project

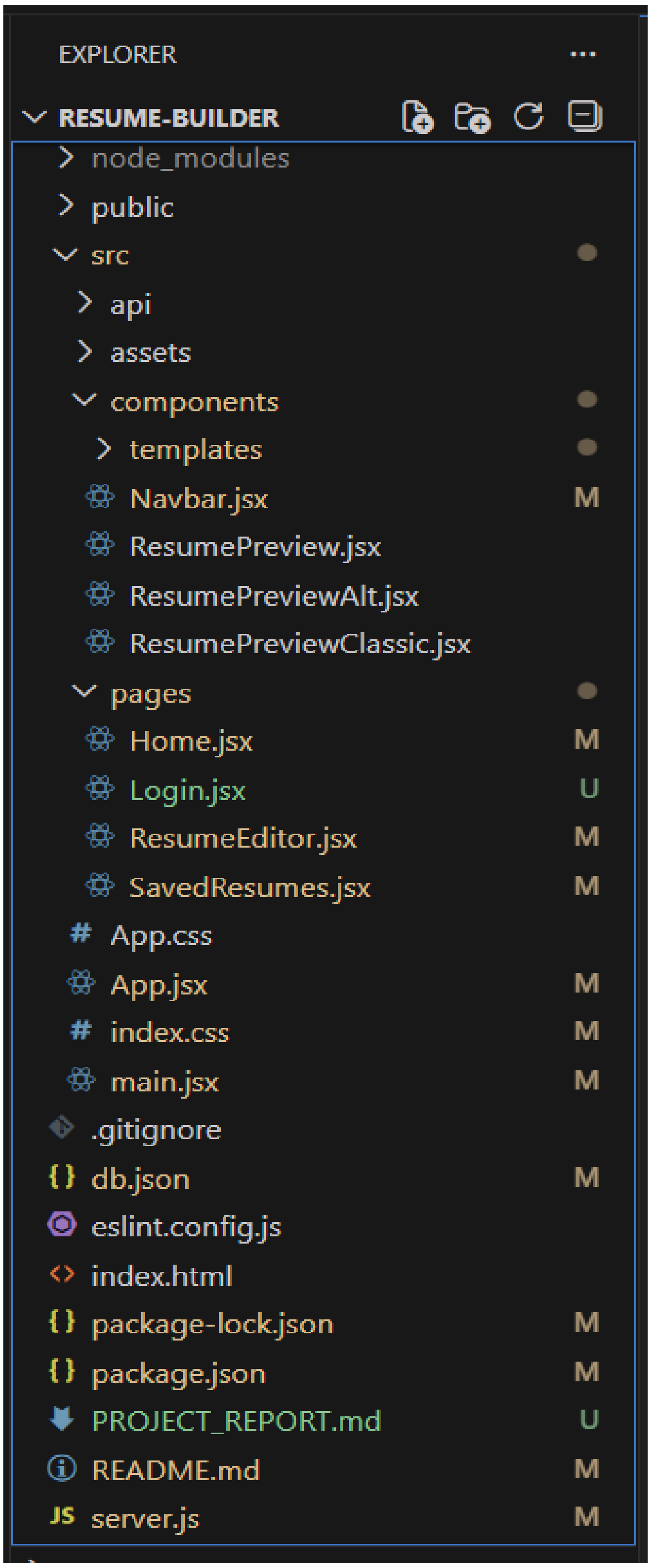
System Requirements

To run the Resume Builder smoothly:

- Minimum 4GB RAM (8GB recommended)
- Dual-core processor
- 5GB free disk space
- Stable internet connection for installing dependencies

7.Project Structure

The Resume Builder application is organized into a clear and modular structure to ensure easy development, scalability, and clean separation of UI, logic, and backend simulation.



```
EXPLORER                                   ...
∨ RESUME-BUILDER
  > node_modules
  > public
  ∨ src                                    ●
    > api
    > assets
    ∨ components                           ●
      > templates                          ●
        Navbar.jsx                         M
        ResumePreview.jsx
        ResumePreviewAlt.jsx
        ResumePreviewClassic.jsx
    ∨ pages                                ●
        Home.jsx                           M
        Login.jsx                          U
        ResumeEditor.jsx                   M
        SavedResumes.jsx                   M
      App.css
      App.jsx                              M
      index.css                            M
      main.jsx                             M
    .gitignore
    db.json                                M
    eslint.config.js
    index.html
    package-lock.json                      M
    package.json                           M
    PROJECT_REPORT.md                      U
    README.md                              M
    server.js                              M
```

1. Frontend (React.js + Vite)
The core of the project contains all UI, pages, and resume-editing logic.
Components (src/components/):

- Navbar.jsx – Navigation bar
- ResumePreview*.jsx – Different resume preview layouts
- Templates (TemplateModern, TemplateElegant, etc.)
- ResumeA4Wrapper.jsx – Handles A4 print layout

Purpose: Live preview, template switching, clean UI, print formatting.
Pages (src/pages/):

- Home.jsx – Landing page
- Login.jsx – Authentication
- ResumeEditor.jsx – Main resume builder
- SavedResumes.jsx – Shows saved resumes

Each page combines UI + logic + API calls.

2. API Layer (Axios)
src/api/api.js
Handles Axios setup, base URL, and all CRUD operations with JSON Server.
Keeps API logic separate from UI for cleaner code.

3. Assets (src/assets/)
Stores icons, illustrations, React logo, and images for login/landing/preview pages.
Used for consistent visuals and branding.

4. Styles

- index.css – Global styles
- App.css – App-level styling
- Tailwind CSS – Utility styling inside components
- Template CSS files – Print and template layout styles

Ensures professional, responsive, print-ready design.

5. Utilities (Optional)
Reusable helpers like:

- Text formatting
- LocalStorage functions

- Form validation
- Common logic

Used to keep components clean and scalable.

6. Backend (JSON Server)

Mock backend used instead of MySQL/Node/PHP.

Files:

- db.json – Resume data storage
- server.js – JSON Server setup

Provides full CRUD for saving, editing, deleting resumes

.

7. Configuration Files

- package.json – Dependencies + scripts
- vite.config.js – Vite configuration
- index.html – Root HTML entry
- .gitignore – Ignore unnecessary files
- README.md – Setup documentation

8. Entry Files

- main.jsx – Mounts React + Router
- App.jsx – Defines routes, global layout, navbar

These boot up the entire application.

8.Project Flow

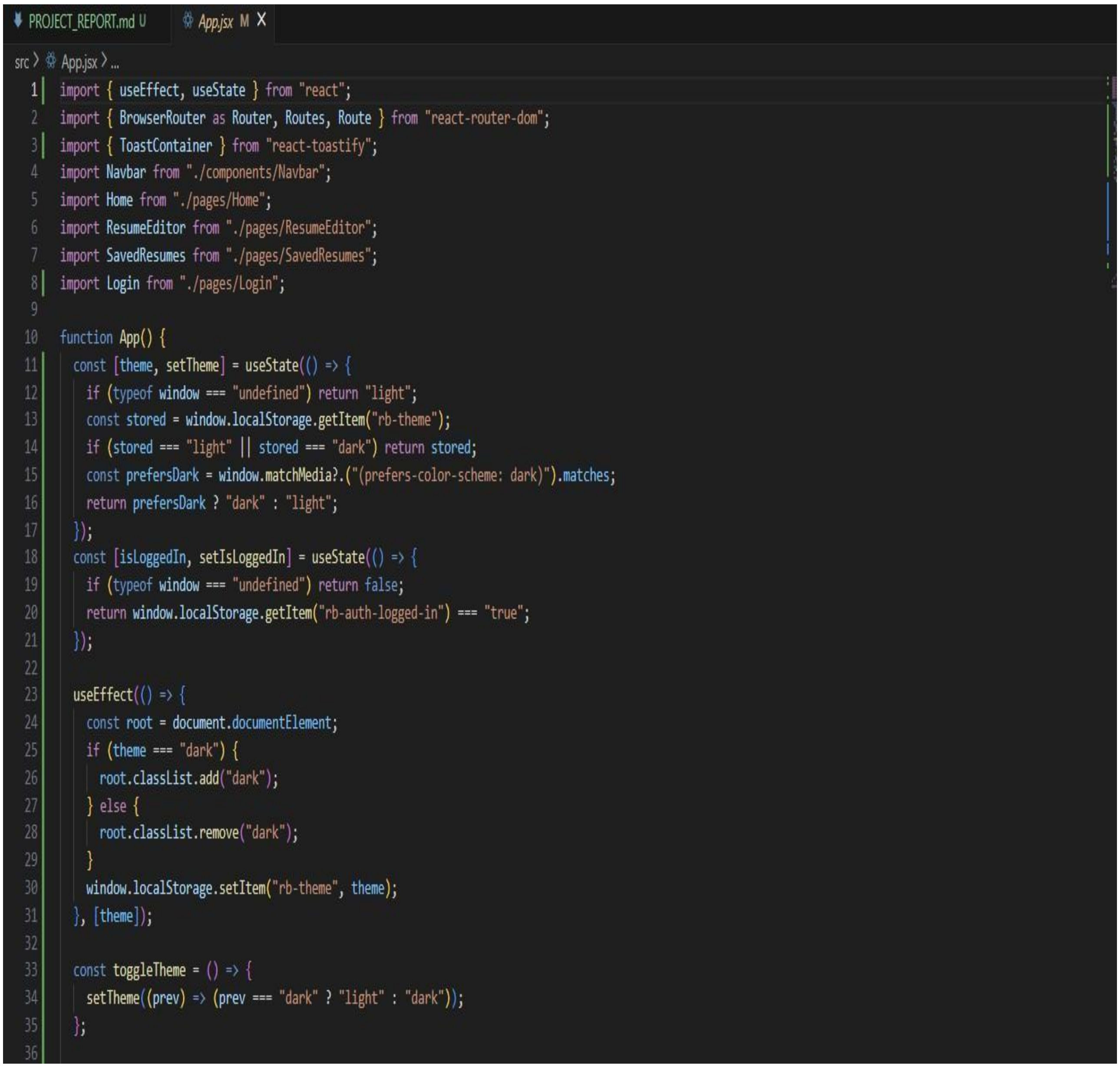
Project demo:

Before starting to work on this project, let’s see the demo.

Demolink: [Demo Link]

Git hub link (code): [Github link]

App.js **component**



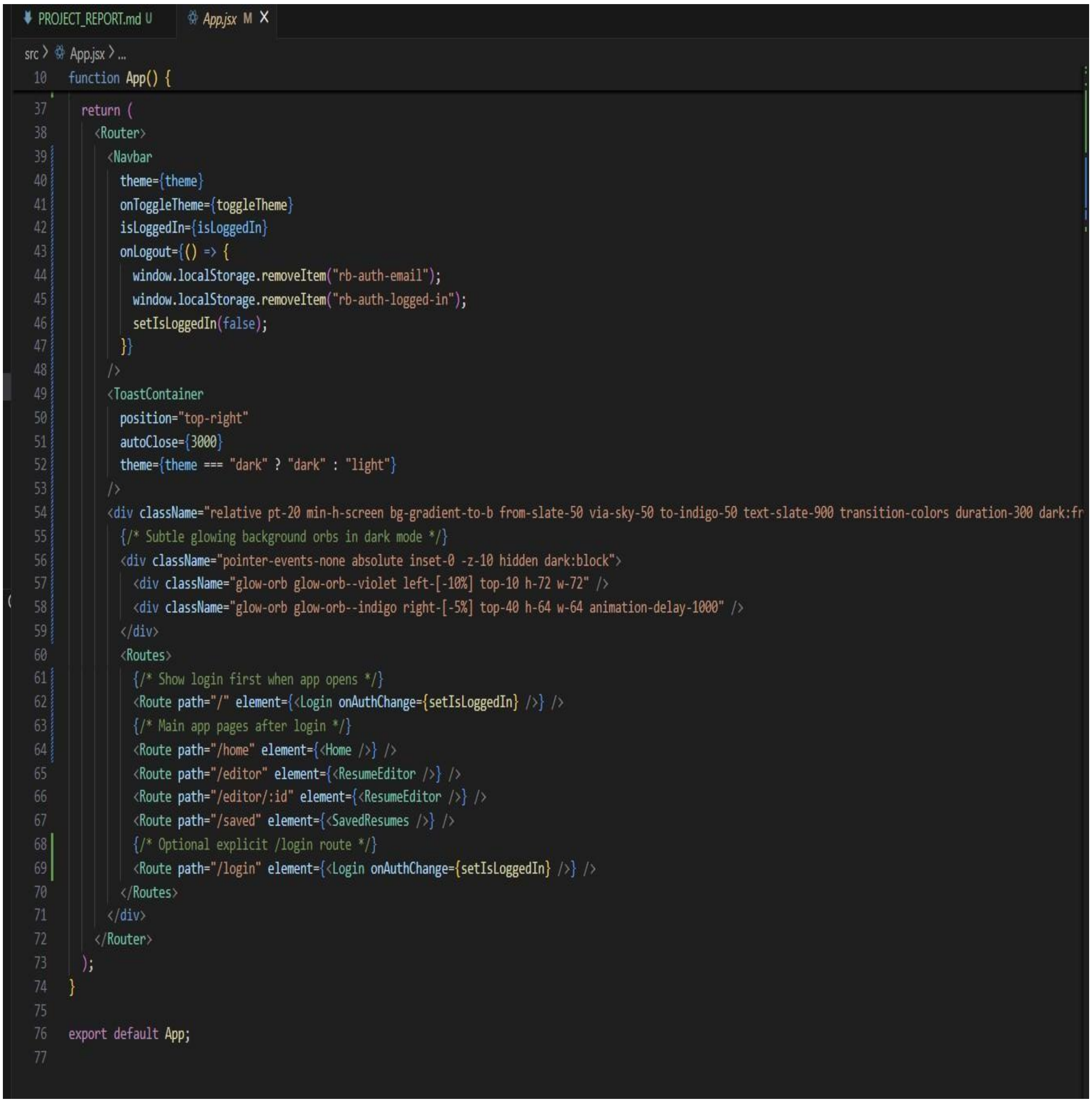
```
import { useEffect, useState } from "react";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import { ToastContainer } from "react-toastify";
import Navbar from "./components/Navbar";
import Home from "./pages/Home";
import ResumeEditor from "./pages/ResumeEditor";
import SavedResumes from "./pages/SavedResumes";
import Login from "./pages/Login";

function App() {
  const [theme, setTheme] = useState(() => {
    if (typeof window === "undefined") return "light";
    const stored = window.localStorage.getItem("rb-theme");
    if (stored === "light" || stored === "dark") return stored;
    const prefersDark = window.matchMedia?.("(prefers-color-scheme: dark)").matches;
    return prefersDark ? "dark" : "light";
  });
  const [isLoggedIn, setIsLoggedIn] = useState(() => {
    if (typeof window === "undefined") return false;
    return window.localStorage.getItem("rb-auth-logged-in") === "true";
  });

  useEffect(() => {
    const root = document.documentElement;
    if (theme === "dark") {
      root.classList.add("dark");
    } else {
      root.classList.remove("dark");
    }
    window.localStorage.setItem("rb-theme", theme);
  }, [theme]);

  const toggleTheme = () => {
    setTheme((prev) => (prev === "dark" ? "light" : "dark"));
  };

```



```jsx
function App() {
  return (
    <Router>
      <Navbar
        theme={theme}
        onToggleTheme={toggleTheme}
        isLoggedIn={isLoggedIn}
        onLogout={() => {
          window.localStorage.removeItem("rb-auth-email");
          window.localStorage.removeItem("rb-auth-logged-in");
          setIsLoggedIn(false);
        }}
      />
      <ToastContainer
        position="top-right"
        autoClose={3000}
        theme={theme === "dark" ? "dark" : "light"}
      />
      <div className="relative pt-20 min-h-screen bg-gradient-to-b from-slate-50 via-sky-50 to-indigo-50 text-slate-900 transition-colors duration-300 dark:fr
        {/* Subtle glowing background orbs in dark mode */}
        <div className="pointer-events-none absolute inset-0 -z-10 hidden dark:block">
          <div className="glow-orb glow-orb--violet left-[-10%] top-10 h-72 w-72" />
          <div className="glow-orb glow-orb--indigo right-[-5%] top-40 h-64 w-64 animation-delay-1000" />
        </div>
        <Routes>
          {/* Show login first when app opens */}
          <Route path="/" element={<Login onAuthChange={setIsLoggedIn} />} />
          {/* Main app pages after login */}
          <Route path="/home" element={<Home />} />
          <Route path="/editor" element={<ResumeEditor />} />
          <Route path="/editor/:id" element={<ResumeEditor />} />
          <Route path="/saved" element={<SavedResumes />} />
          {/* Optional explicit /login route */}
          <Route path="/login" element={<Login onAuthChange={setIsLoggedIn} />} />
        </Routes>
      </div>
    </Router>
  );
}

export default App;
```

Code Description
Imports React

Needed to build components and use hooks like useState and useEffect.

Imports All Components & Pages

These are the main UI screens of your Resume Builder app.

Components:

- Navbar
- ResumePreview
- ResumeEditor
- SavedResumes
- Template components (Modern, Elegant, Creative, Professional, Minimal)

Pages:

- Home
- Editor
- Saved
- Login

Each one is a separate part of the website interface.

Defines App Component

The App component handles:

- Page routing
- Theme (dark/light mode)
- Authentication status
- Navigation bar
- Toast notifications
- This acts as the core layout + logic of your entire application.

useEffect

Runs only once when the app loads:

It loads the saved theme (dark/light mode) from localStorage:

```
useEffect(() => {
  loadTheme();
}, []);
```

Main Content Using Routes

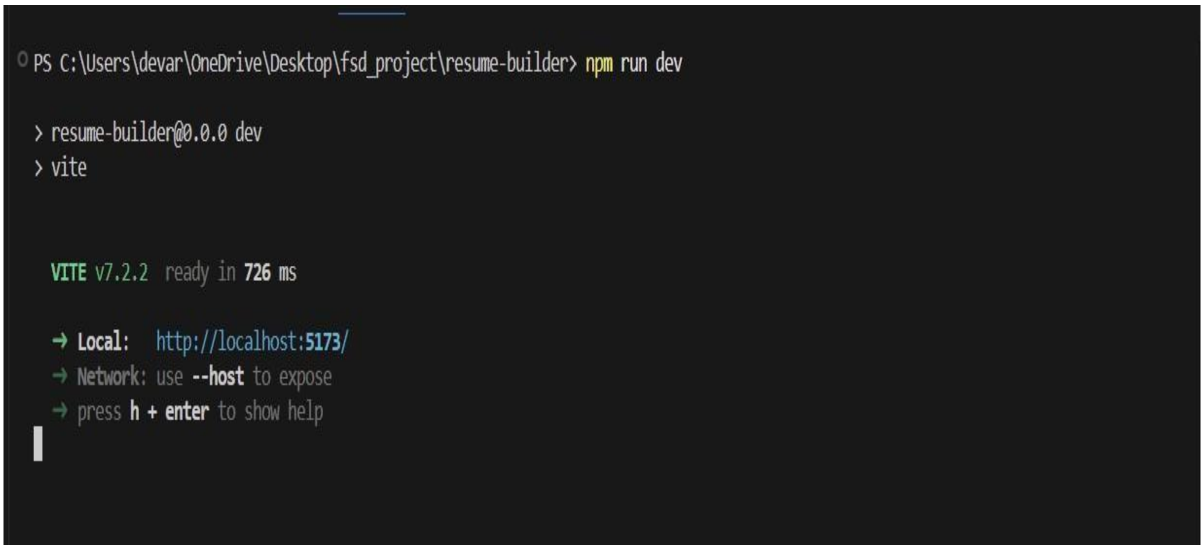
Each <Route /> opens a different page:

- "/" → Home page
- "/editor" → Create a new resume
- "/editor/:id" → Edit an existing resume
- "/saved" → View all saved resumes
- "/login" → Login / Signup page

This is how your resume builder switches between pages inside the browser.

9.Project Execution:

After completing the code, run your CookBook JSON server using “npm run server” and start the frontend using “npm run dev”.
These commands will launch your JSON Server API and your Vite React application.

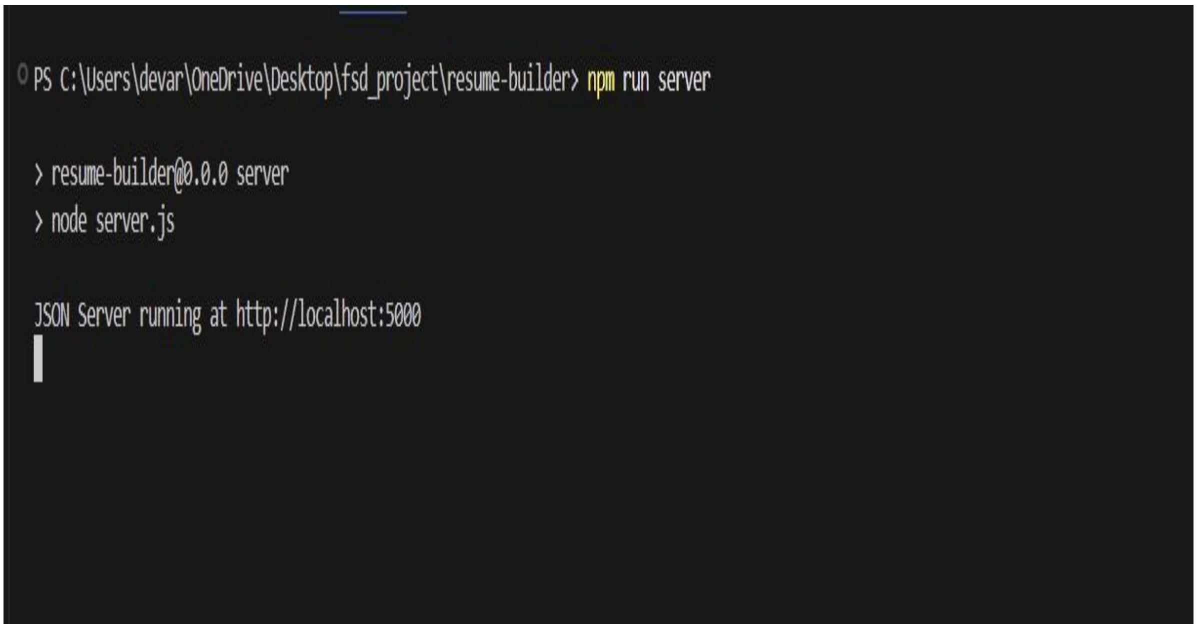


```
PS C:\Users\devar\OneDrive\Desktop\fsd_project\resume-builder> npm run dev

> resume-builder@0.0.0 dev
> vite


  VITE v7.2.2  ready in 726 ms

  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h + enter to show help
```



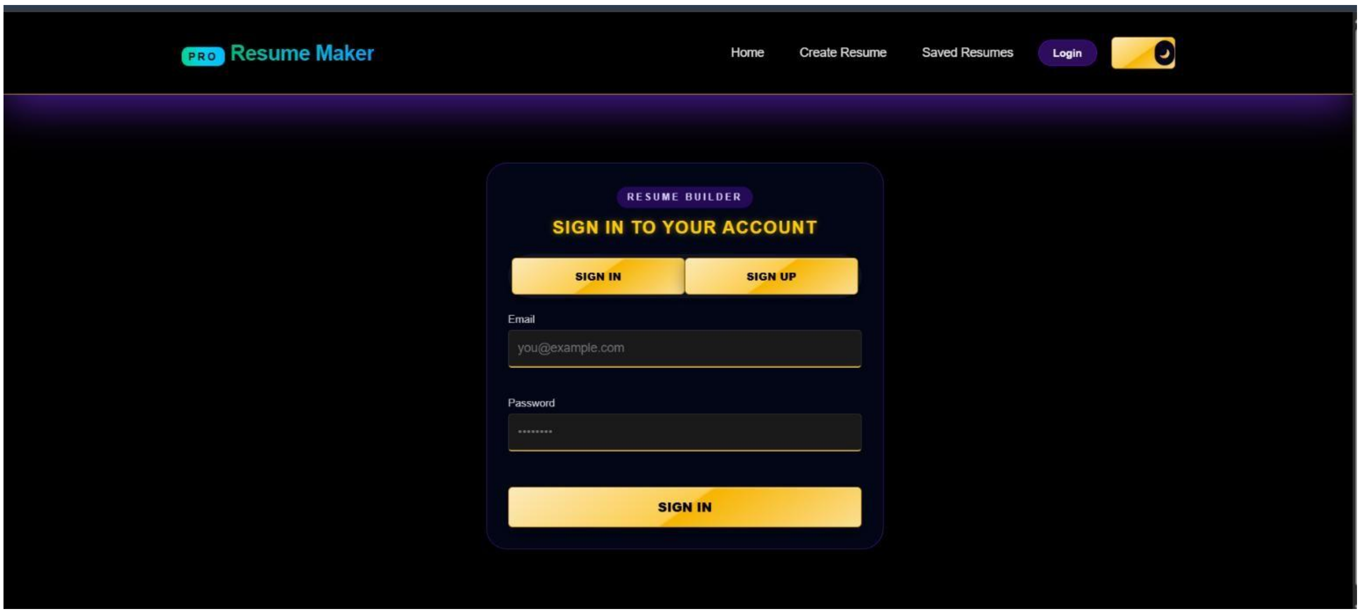
```
PS C:\Users\devar\OneDrive\Desktop\fsd_project\resume-builder> npm run server

> resume-builder@0.0.0 server
> node server.js

JSON Server running at http://localhost:5000
```

10. Output

Login page



SIGN IN TO YOUR ACCOUNT

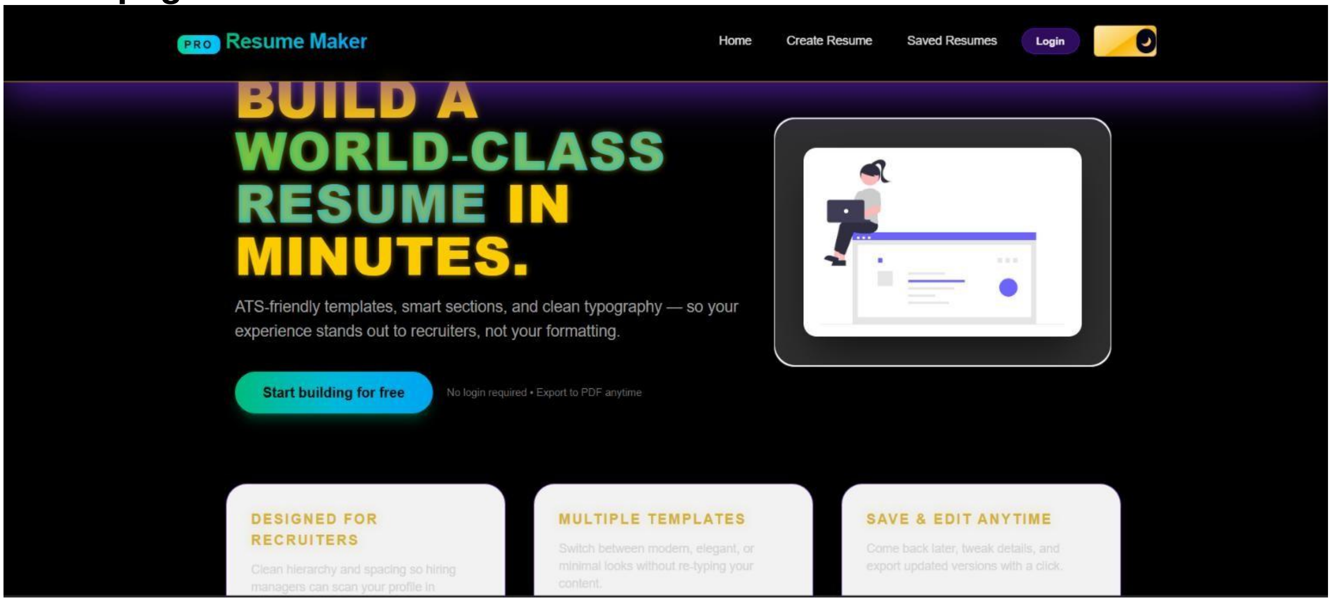
Email

you@example.com

Password

••••••••

Home page



BUILD A WORLD-CLASS RESUME IN MINUTES.

ATS-friendly templates, smart sections, and clean typography — so your experience stands out to recruiters, not your formatting.

No login required • Export to PDF anytime

DESIGNED FOR RECRUITERS

Clean hierarchy and spacing so hiring managers can scan your profile in

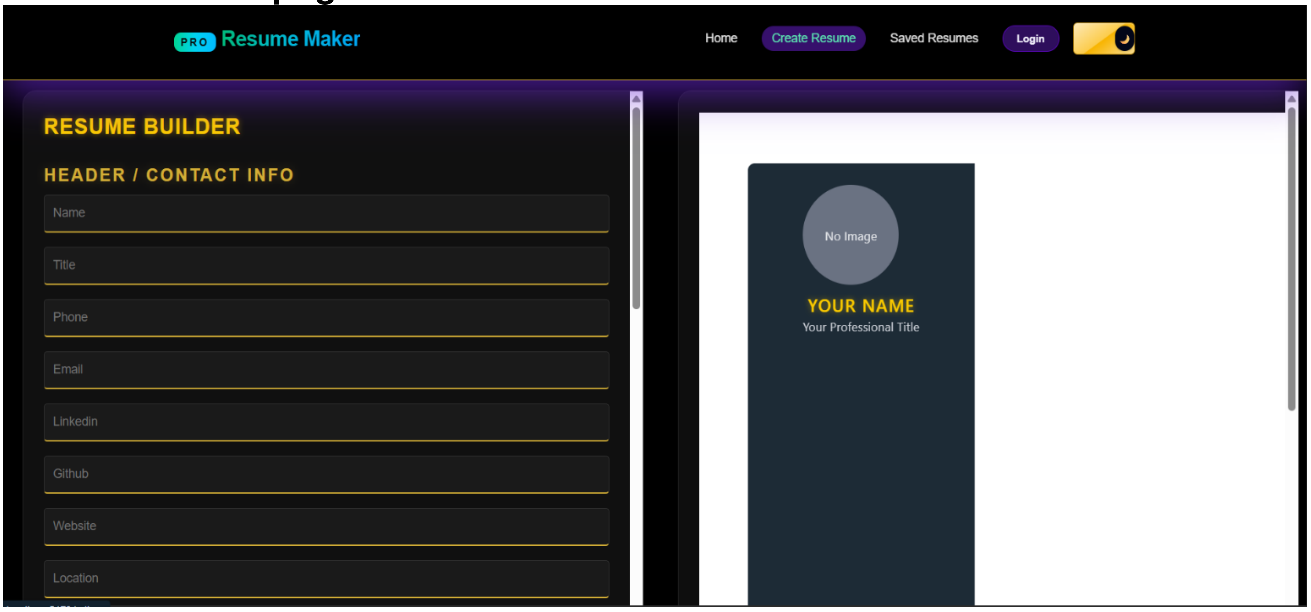
MULTIPLE TEMPLATES

Switch between modern, elegant, or minimal looks without re-typing your content.

SAVE & EDIT ANYTIME

Come back later, tweak details, and export updated versions with a click.

Create resume page



PRO Resume Maker

Home | Create Resume | Saved Resumes | Login

RESUME BUILDER

HEADER / CONTACT INFO

Name

Title

Phone

Email

Linkedin

Github

Website

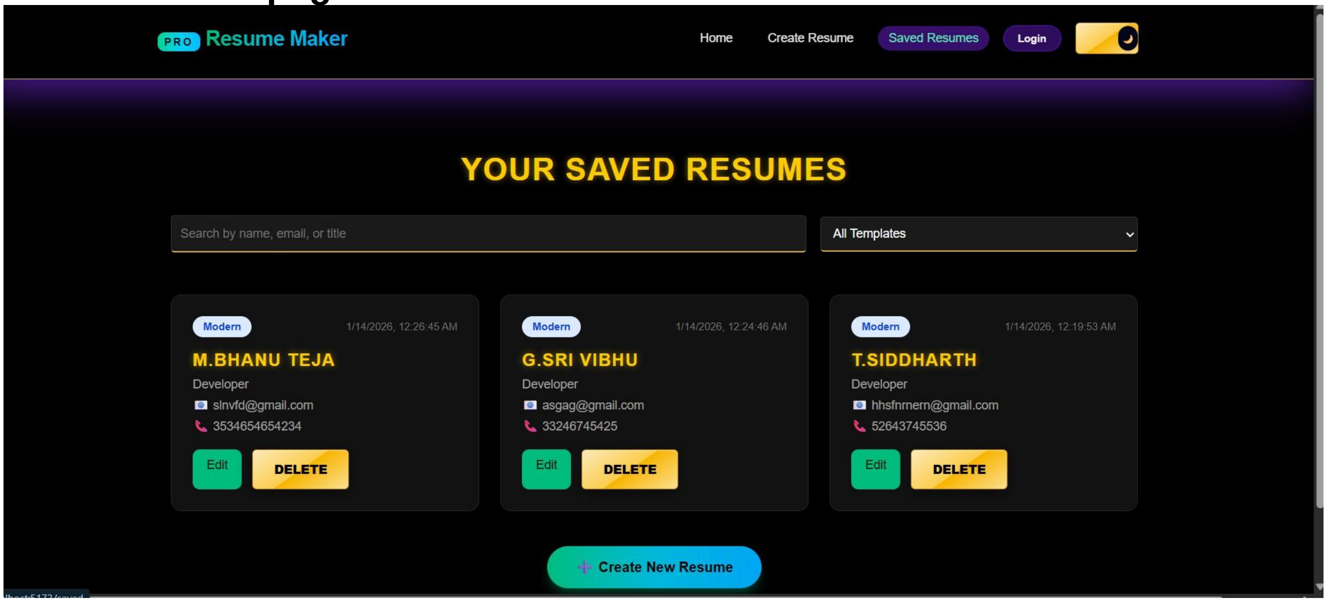
Location

No Image

YOUR NAME

Your Professional Title

Saved resume page



PRO Resume Maker

Home | Create Resume | Saved Resumes | Login

YOUR SAVED RESUMES

Search by name, email, or title

All Templates

Modern 1/14/2026, 12:26:45 AM
M.BHANU TEJA
Developer
slnvfd@gmail.com
3534654654234
Edit DELETE

Modern 1/14/2026, 12:24:46 AM
G.SRI VIBHU
Developer
asgag@gmail.com
33246745425
Edit DELETE

Modern 1/14/2026, 12:19:53 AM
T.SIDDHARTH
Developer
hhsfnrnern@gmail.com
52643745536
Edit DELETE

+ Create New Resume