

INTEGRATED PROJECT REPORT

On

PROPREP

Submitted in partial fulfilment of the requirement for the
Course IP (22CS203) of

COMPUTER SCIENCE AND ENGINEERING
B.E. Batch-2022

in

Jan -2025



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CHITKARA UNIVERSITY
PUNJAB

Under the Guidance of

Mr. Kamal Saluja

Submitted By

Nishtha

Roll No. 2210991992

Pragti Gupta

Roll No. 2210992056

Priya Gupta

Roll No. 2210992096

CERTIFICATE

This is to be certified that the project entitled “ProPrep” has been submitted for the Bachelor of Computer Science Engineering at Chitkara University, Punjab during the academic semester January 2025 - June 2025 is a bona fide piece of project work carried out by Nishtha 2210991992, Pragti Gupta 2210992056 and Priya Gupta 2210992096 towards the partial fulfilment for the award of the course Integrated Project (CS203) under the guidance of Mr. Kamal Saluja and supervision.

Sign. of Project Guide:

Mr. Kamal Saluja

CANDIDATE'S DECLARATION

We, Nishtha 2210991992, Pragti Gupta 2210992056, Priya Gupta 2210992096, B.E.-2022 of the Chitkara University, Punjab hereby declare that the Integrated Project Report entitled **“ProPrep”** is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other course.

Sign. of Student 1

Nishtha

ID No 2210991992

Sign. of Student 2

Pragti Gupta

ID No 2210992056

Sign. of Student 3

Priya Gupta

ID No 2210992096

Place: Punjab

Date:04-03-2025

ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed to the development of this work and who influenced my thinking, behavior and acts during the course of study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project as the part of the curriculum.

We are thankful to Mr. Kamal Saluja for his support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

We also extend our sincere appreciation to *Mr. Kamal Saluja* who provided his valuable suggestions and precious time in accomplishing our integrated project report.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to day experience and received lots of suggestions that improve our quality of work.

Nishtha
ID No 2210991992

Pragti Gupta
ID No 2210992056

Priya Gupta
ID No 2210992096

Table of Contents

| S. No | Topic | Page No. |
|--------------|--|-----------------|
| 1. | Abstract | 6 |
| 2. | Introduction | 6 - 7 |
| 2.1 | Background | 7 - 8 |
| 2.2 | Problem Statement | 8 - 9 |
| 3. | Software and Hardware Requirement Specification | 9 - 14 |
| 3.1 | Methods | 9 - 10 |
| 3.2 | Programming Working Environment | 10 - 12 |
| 3.3 | Requirements to Run the Application | 12 - 14 |
| 4. | Database Analyzing Design and Implementation | 15 - 16 |
| 5. | Program's Structure Analyzing and GUI Constructing | 16 - 23 |
| 6. | Code Implementation and Database Connections | 24 - 29 |
| 7. | Limitations | 29 - 31 |
| 8. | Conclusion | 32 - 33 |
| 9. | Future Scope | 33 - 35 |
| 10. | Bibliography/References | 35 |

1. Abstract/ Keywords

Abstract: ProPrep is an AI-powered career preparation platform designed to assist job seekers in optimizing their resumes and enhancing their interview skills. It addresses a common challenge faced by students, recent graduates, and professionals alike—navigating the complex and often overwhelming job application process. The platform integrates two core functionalities: an AI Resume Builder, which generates tailored, ATS-friendly resumes with intelligent suggestions based on industry standards and user input, and an AI Mock Interview App, which offers real-time, interactive interview simulations complete with adaptive feedback, voice analysis, and role-specific questions.

ProPrep bridges the gap between preparation and confidence by offering a personalized experience for each user. Whether you're applying for your first internship, transitioning careers, or preparing for high-stakes job interviews, ProPrep equips you with the tools to stand out. Built using modern technologies like React, Tailwind CSS, Next.js, Drizzle ORM, Clerk, and Gemini AI, the platform ensures a smooth, fast, and responsive user experience across devices.

By leveraging AI-driven insights, real-time analytics, and a clean, intuitive interface, ProPrep simplifies and humanizes the job hunt—turning stress into strategy. It empowers users to build professional-grade resumes and practice confidently for interviews anytime, anywhere. Ultimately, ProPrep streamlines the job application journey, making career growth more accessible, personalized, and effective in today's competitive market.

2. Introduction to the project

In today's highly competitive job market, candidates—especially fresh graduates, career switchers, and even experienced professionals—often find themselves overwhelmed by the dual challenge of creating impactful resumes and preparing effectively for interviews. With job descriptions becoming increasingly complex and applicant tracking systems (ATS) filtering out resumes before they reach human eyes, simply having the right skills is no longer enough. This is where **ProPrep** steps in—an innovative AI-powered platform designed to bridge this crucial gap by offering a comprehensive and accessible career preparation solution.

ProPrep combines two essential tools in a single, intuitive interface: an AI Resume Builder, which generates personalized, ATS-optimized resumes with intelligent suggestions based on user inputs and industry best practices, and an AI Mock Interview App, which simulates real-time interview experiences using voice interaction, adaptive questioning, and feedback to help users sharpen their responses and reduce anxiety.

Built using modern technologies such as React, Tailwind CSS, Clerk, Next.js, Drizzle ORM, and Gemini AI, ProPrep offers a responsive, user-friendly experience across devices. It caters to users from diverse backgrounds—whether you're a student applying for your first internship, a job seeker struggling to clear interviews, or a working professional aiming for a better role. By leveraging the power of artificial intelligence, the platform delivers personalized resume enhancements, insightful analytics, and realistic interview practice—all designed to boost your confidence and readiness for the real world.

Ultimately, ProPrep is more than just a tool—it's a career companion. It empowers individuals to present their best selves to employers and navigate the job market with clarity and confidence, turning preparation into progress and ambition into achievement.

2.1 Background

In today's rapidly evolving and highly competitive job market, standing out has become more challenging than ever. Job seekers—whether they are fresh graduates entering the workforce, professionals seeking career advancement, or individuals switching industries—often struggle with two major hurdles: creating a polished, ATS-compatible resume and preparing confidently for high-pressure interviews. Traditional methods such as static resume templates or generic interview prep guides frequently fall short. They lack personalization, fail to reflect real-time industry trends, and offer no dynamic feedback, leaving users uncertain and underprepared.

Recognizing this gap, ProPrep was developed as a comprehensive AI-powered career preparation platform aimed at revolutionizing how individuals approach their job search. The project brings together two essential tools under one roof: an AI Resume Builder, which intelligently analyses user input to provide tailored content suggestions, structure optimization, and design formatting that align with ATS standards; and an AI Mock Interview App, which mimics real interview scenarios using voice input, role-specific questions, and adaptive, real-time AI feedback to help users improve their communication and confidence.

By integrating modern technologies like React, Tailwind CSS, Clerk, Next.js, Drizzle ORM, and Gemini AI, ProPrep ensures a seamless, responsive, and interactive user experience. The platform isn't just a set of tools—it's a personalized career coach that supports users throughout their job preparation journey. It was designed with empathy and inclusivity in mind, making it suitable for users with different levels of experience, technical knowledge, or career goals.

Ultimately, ProPrep empowers job seekers to take control of their career path with confidence. It shortens the learning curve, saves time, and reduces anxiety—streamlining the transition from job search to employment and significantly improving the chances of landing the right opportunity in an increasingly demanding job market.

2.2 Problem Statement

Job seekers today face a multitude of challenges that go beyond just finding job openings. One of the most common struggles is creating a well-structured, professional resume that not only reflects their skills and achievements but also passes through Applicant Tracking Systems (ATS) used by most companies. At the same time, preparing for interviews—especially in high-stakes or unfamiliar environments—can be equally daunting. Many candidates are left guessing what questions to expect, how to structure their answers, or how to communicate their value confidently and clearly.

These hurdles are often worsened by the fragmented nature of existing solutions. Most tools available in the market focus on either resume building or interview practice, rarely both, and even then, they lack personalization, contextual advice, or real-time feedback. Users end up toggling between multiple websites, static templates, outdated guides, or generic videos—all of which fail to address their unique needs or help them improve in a meaningful way. For students, first-time job seekers, or professionals returning to the workforce, this disjointed experience leads to confusion, anxiety, and missed opportunities.

This project aims to solve these pain points through ProPrep, an integrated AI-powered platform that brings together two essential tools in one cohesive experience. The AI Resume Builder App helps users craft professional, ATS-friendly resumes tailored to their background and the roles they are applying for, offering intelligent content suggestions and formatting

assistance. Meanwhile, the AI Mock Interview App simulates realistic interview environments, complete with real-time questioning, voice input, and personalized AI feedback that adapts to each user's performance and target role.

By leveraging artificial intelligence, ProPrep simplifies and personalizes the job application process—transforming it from a stressful, trial-and-error journey into a guided, confidence-building experience. The platform empowers users to present their best selves to potential employers, improves their readiness for real-world hiring scenarios, and ultimately increases their chances of landing the right job in a competitive landscape.

3. Software and Hardware Requirement Specification

3.1 Methods

To ensure that ProPrep functions smoothly, securely, and intuitively across devices and use cases, a combination of modern software tools and frameworks was used for both backend and frontend development, as well as database management. The goal was to build a scalable, maintainable, and user-friendly platform that meets the real needs of job seekers.

Backend Development:

- **User Authentication:** For managing sign-ups, logins, and secure sessions, Clerk was integrated. This allows users to access their personal dashboard, saved resumes, and interview history from any device, without worrying about losing their data or privacy. Clerk also supports multi-factor authentication and password recovery, making the platform trustworthy for long-term use.
- **API Development:** Backend operations like user profile handling, resume creation, interview scheduling, and communication with the AI services were implemented using Next.js API routes. This helped keep the platform fast and efficient, while maintaining a unified codebase between the frontend and backend.
- **AI Integration:** Gemini AI was embedded to power smart features such as resume content suggestions and real-time interview feedback. This brings a level of personalization and intelligence that manual tools simply can't match—turning a traditionally stressful process into a guided and supportive experience.

- **Data Management:** To ensure consistent and safe data handling, Drizzle ORM was used for managing structured interactions with the PostgreSQL database. Whether it's saving a user's resume draft or recording interview results, Drizzle keeps the data organized, efficient, and easy to maintain.

Frontend Development:

- **UI/UX Design:** The user interface was built using React (with Next.js) and styled with Tailwind CSS, offering a clean, modern, and mobile-responsive design. Every screen—from the resume editor to the interview dashboard—was crafted to be intuitive for users of all backgrounds, even those with limited technical experience.
- **Routing:** Navigation between major sections like resume building, mock interviews, and analytics was handled through Next.js routing, ensuring fast page loads and a seamless browsing experience, much like a single-page application.
- **State Management:** Using React Hooks and built-in state logic, the platform smoothly handles real-time user interactions—such as switching between resume sections or updating feedback—without lag or interruptions, enhancing the overall experience.

Database Design:

- **PostgreSQL:** A relational database was chosen to ensure structured, reliable storage of key data such as user profiles, resume versions, interview histories, and AI-generated feedback. PostgreSQL's reliability and performance make it ideal for managing large-scale data in a secure way.
- **ORM Usage:** By leveraging Drizzle ORM, database queries and schema definitions were kept clean and modular. This simplifies development and makes it easy to scale the system as more users join the platform or as new features are added.

3.2 Programming/Working Environment

To develop a robust, scalable, and user-friendly platform like ProPrep, a carefully selected modern tech stack was used—designed to balance developer efficiency, performance, and seamless user experience. Each component of the working environment was chosen not only for its technical strengths but also for how well it supports the kind of rapid, AI-driven features that job seekers expect in today's digital world.

- **Backend Environment:** The backend of ProPrep was built using **Node.js**, known for its speed and scalability in handling server-side operations. To streamline both frontend and backend development under a unified framework, **Next.js API routes** were implemented for building efficient server-side logic. These routes manage everything from user account creation and authentication to resume generation and real-time communication with AI models.
 - **PostgreSQL** serves as the relational database, chosen for its reliability, structure, and ability to manage complex relationships among user data, resume content, interview sessions, and AI feedback logs. It provides the necessary foundation to handle the growing dataset as more users join the platform.
 - **Clerk** was integrated to handle **user authentication** in a secure, scalable way. It manages sign-ups, logins, session tracking, and password recovery without requiring the development team to build these features from scratch. For users, this means a seamless login experience with strong security standards.
 - For handling data between the application and the database, **Drizzle ORM** was adopted. This modern ORM offers type-safe queries and schema consistency, making it easier to write and manage database logic. It reduces errors and helps developers maintain a clean, modular backend.
- **Frontend Environment:** The frontend was designed to be sleek, intuitive, and responsive, making it accessible to users of all skill levels, whether they're tech-savvy developers or non-technical job seekers.
 - Built using **Next.js** with **React components**, the frontend benefits from both the performance advantages of server-side rendering and the interactivity of client-side navigation. This ensures pages load quickly and transitions feel smooth and natural.
 - **Tailwind CSS** was used for styling to ensure a modern, clean UI that adapts well to different screen sizes—from desktops to smartphones. It helps developers maintain visual consistency and quickly implement design changes without having to write verbose CSS from scratch.
- **Development Tools:** To enhance productivity and streamline collaboration, a suite of professional development tools was used:

- **Visual Studio Code (VS Code)** served as the primary code editor. Its powerful extensions, built-in terminal, and Git integration make it ideal for full-stack development.
- **Postman** was used extensively to test and debug API endpoints. This allowed the team to simulate real API requests and verify server responses—ensuring that resume generation, AI feedback, and user authentication worked flawlessly before going live.
- **Git and GitHub** were used for version control, code collaboration, and project management. With Git branches, pull requests, and commit history, developers could work in parallel without conflict, track progress, and quickly roll back if needed.

3.3 Requirements to Run the Application

In order to develop, run, and maintain a seamless experience on ProPrep—an AI-powered career preparation platform—several tools, technologies, and environments need to be properly set up. Each component in the tech stack plays a vital role in ensuring smooth operations, responsive performance, secure user handling, and intelligent decision-making behind the scenes. The following are the software and system requirements needed to run and scale the application effectively.

- **Node.js and npm(Node Package Manager)**

Purpose: Node.js is the runtime environment that allows JavaScript to run on the server-side. It forms the backbone of the backend API routes, enabling asynchronous, event-driven programming. npm is used to install and manage the libraries and packages required for both frontend and backend development.

Why it matters: Node.js ensures that user actions—like saving a resume, starting a mock interview, or requesting feedback—are processed swiftly on the server without delays. It helps power real-time functionalities and improves the responsiveness of the application.

Real-Life Examples: A job seeker uses ProPrep to edit their resume. When they click "Generate Suggestions," Node.js instantly processes the request, sends it to Gemini AI, and returns smart, tailored inputs—within seconds.

- **PostgreSQL**

Purpose: PostgreSQL is a highly reliable, open-source relational database used to store all structured data—ranging from user accounts and resume blocks to interview history and AI-generated feedback.

Why it matters: The application handles sensitive and critical user information, which must be stored with integrity and retrieved quickly. PostgreSQL supports advanced querying, indexing, and relational data modeling, making it the ideal choice for managing resume versions, interview logs, and analytics.

Real-Life Examples: Each time a user saves a resume version or completes an interview session, all associated content—AI suggestions, timestamps, user inputs—are stored securely in PostgreSQL, making it retrievable anytime the user logs in again.

- **Next.js Framework:**

Purpose: Next.js is a powerful React framework used for both frontend development and backend API routes. It allows server-side rendering, static site generation, and easy routing, enabling faster loading and better SEO optimization.

Why it matters: Next.js reduces loading times, enhances navigation, and supports the split between backend API logic and dynamic frontend rendering—all from a single codebase.

Real-Life Examples: A candidate switches from the resume builder to their analytics dashboard. With Next.js, this transition feels instant and smooth, without needing to reload the page or re-fetch static data unnecessarily.

- **Tailwind CSS:**

Purpose: Tailwind CSS is a utility-first CSS framework used to rapidly build clean, responsive, and mobile-friendly user interfaces.

Why it matters: Tailwind ensures that ProPrep’s interface looks polished and consistent across devices. It simplifies styling by removing the need to write custom CSS, speeding up frontend development significantly.

Real-Life Examples: Whether a user accesses ProPrep on their laptop or mobile phone, Tailwind ensures the UI elements like buttons, modals, and form inputs adjust perfectly—improving overall accessibility and usability.

- **Clerk Authentication:**

Purpose: Clerk is a robust authentication and user management platform that handles secure logins, sign-ups, password resets, and session tracking.

Why it matters: Security and privacy are critical for an application that deals with personal career data. Clerk makes it easy to implement secure auth flows with modern UX, including magic links, email verification, and OAuth support.

Real-Life Examples: A user signs in to ProPrep using their Google account. Clerk verifies the user's identity, creates a session, and directs them to their personalized dashboard—all in a few clicks.

- **Gemini AI Integration:**

Purpose: Gemini AI is used to power the intelligent systems behind ProPrep. It provides smart resume suggestions, formulates realistic interview questions, and gives adaptive, constructive feedback based on user responses.

Why it matters: Rather than relying on static templates or scripted answers, users get dynamic, real-time guidance that mimics real-world recruiter interactions—enhancing their preparation and boosting their confidence.

Real-Life Examples: During a mock interview, Gemini AI analyzes the user's answers and gives targeted feedback like: “Try to include measurable results when describing your achievements.” This replicates the guidance of a career coach or hiring manager.

- **Drizzle ORM:**

Purpose: Drizzle ORM is a type-safe Object-Relational Mapper that facilitates communication between the application and the PostgreSQL database using structured queries and schema definitions.

Why it matters: ORMs reduce the complexity of writing raw SQL, minimize errors, and ensure that database logic remains consistent across different modules of the app.

Real-Life Examples: When a user edits their resume and clicks “Save,” Drizzle ORM automatically updates the right row in the database with minimal code and maximum reliability—keeping the application stable and maintainable.

4. Database Analyzing, Design and Implementation

A well-structured and reliable database is the backbone of any interactive platform—especially one like ProPrep, which deals with dynamic user data, real-time feedback, and personalized content generation. For this project, a PostgreSQL database schema was carefully designed to support all core functionalities of both the AI Resume Builder and AI Mock Interview App, with a focus on performance, scalability, and data integrity.

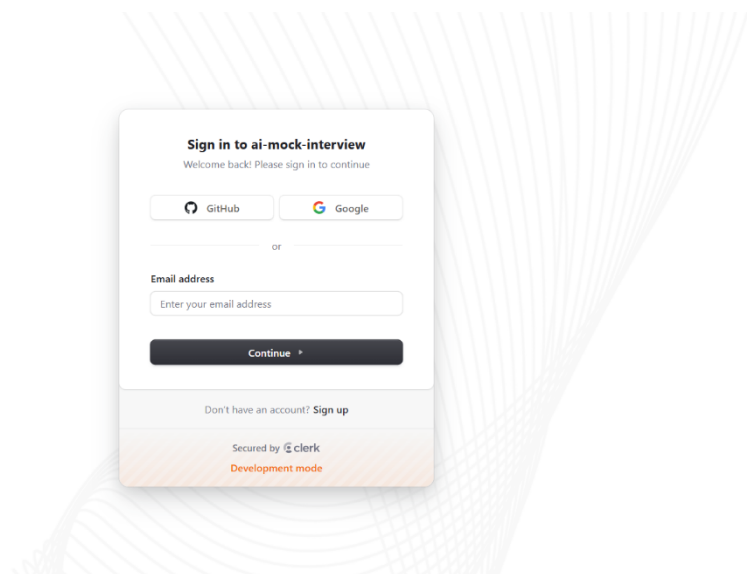
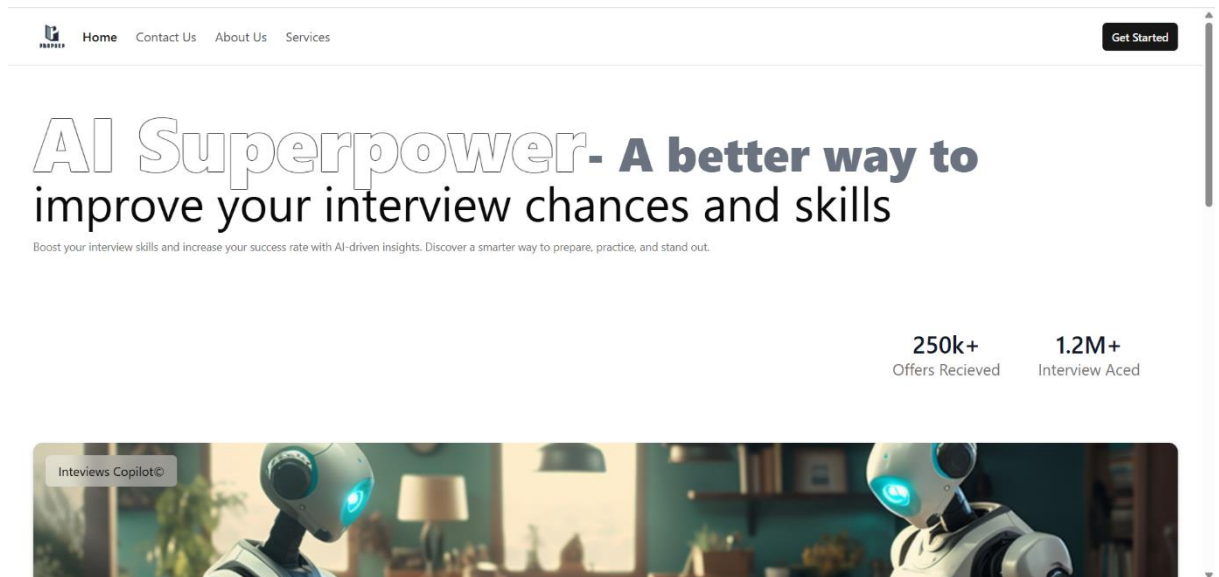
The database schema incorporates several essential entities, including users, resumes, interview sessions, responses, AI feedback, and session logs. Each table was optimized for efficient CRUD (Create, Read, Update, Delete) operations, ensuring that the system performs smoothly even as the number of users grows. To maintain a clean and maintainable codebase, Drizzle ORM was employed to handle structured interactions between the database and application logic, enabling type-safe queries and modular schema definitions across components.

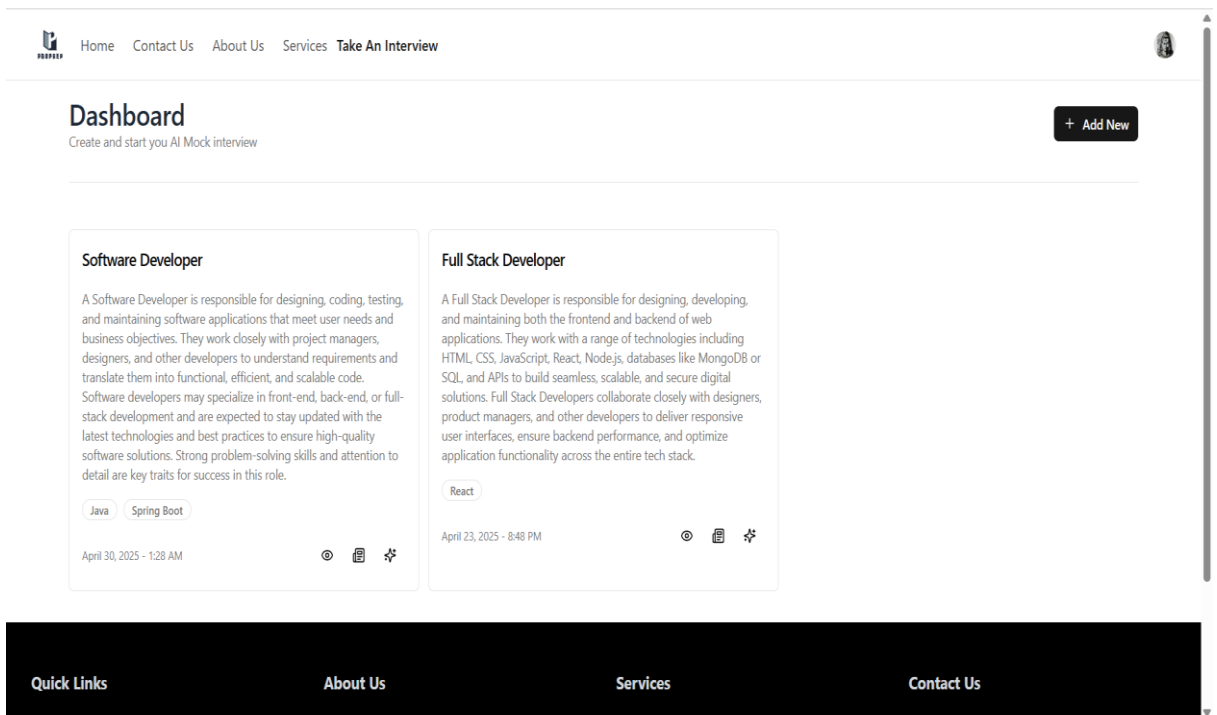
The database is designed not just to store information, but to support a rich, personalized experience for users. Specifically, it enables:

- **User profiles:** Each user has a dedicated profile that stores their personal details, login credentials (secured via authentication systems like Clerk), multiple resume versions, and a history of mock interview sessions. This allows users to revisit, revise, and track their career preparation journey over time.
- **Resume Data:** The system stores content blocks generated by AI, such as tailored professional summaries, skill highlights, and experience sections. It also remembers formatting preferences and export logs—ensuring users can build, tweak, and download their resumes with minimal friction.
- **Interview sessions:** Each mock interview session records AI-generated questions, the user's voice/text responses, timestamps, and session context. This structured data makes it possible to offer meaningful performance reviews and let users reflect on their growth.
- **AI feedback:** Personalized, real-time feedback—whether it's resume improvement tips or constructive interview critiques—is stored securely for future reference. This empowers users to learn from each session and continuously improve.
- **Session logs:** By tracking user activity, ProPrep can provide useful progress analytics, recommend next steps, and even identify usage patterns to improve the overall user

experience. These insights also help the platform evolve intelligently based on real-world engagement.

5. Program's Structure Analyzing and GUI Constructing





[Home](#) > [Mock Interviews](#) > [Create](#)

Create a new mock interview

Job Role / Job Position

eg:- Full Stack Developer

Job Description

eg:- describe your job role


Years of Experience


eg:- 5 Years


Tech Stacks

eg:- React, Typescript...

ResetCreate

 [Home](#) [Contact Us](#) [About Us](#) [Services](#) [Take An Interview](#)




[Home](#) > [Mock Interviews](#) > [Software Developer](#) Start 

Software Developer

A Software Developer is responsible for designing, coding, testing, and maintaining software applications that meet user needs and business objectives. They work closely with project managers, designers, and other developers to understand requirements and translate them into functional, efficient, and scalable code. Software developers may specialize in front-end, back-end, or full-stack development and are expected to stay updated with the latest technologies and best practices to ensure high-quality software solutions. Strong problem-solving skills and attention to detail are key traits for success in this role.

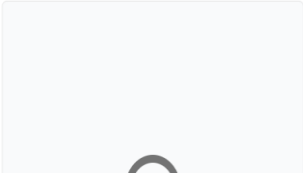
[Java](#) [Spring Boot](#)

April 30, 2025 - 1:28 AM


 **Important Information**

Please enable your webcam and microphone to start the AI-generated mock interview. The interview consists of five questions. You'll receive a personalized report based on your responses at the end.





Note: Your video is **never recorded**. You can disable your webcam at any time.



Explain the difference between `==` and `.equals()` in Java. When should you use one over the other?



Stop Recording

Your Answer:

Start recording to see your answer here

Interview Feedback

Describe a situation where you had to optimize the performance of a Java/Spring Boot application. What steps did you take to identify the bottleneck and improve the performance?

★ Rating : 3

Expected Answer

In a previous project, we experienced slow response times for a particular API endpoint. To identify the bottleneck, I first used profiling tools like VisualVM and JProfiler to monitor the application's performance under load. This revealed that the main issue was inefficient database queries and excessive garbage collection. Specifically, the application was fetching a large amount of data from the database that wasn't actually being used. To improve performance, I took the following steps: 1) Optimized the database query by using indexes, rewriting the query to be more selective, and implementing pagination. 2) Implemented caching using Spring's @Cacheable annotation to reduce the number of database calls for frequently accessed data. 3) Tuned the JVM garbage collection settings to reduce the frequency of full GC cycles. I also reviewed the code for potential memory leaks and optimized data structures and algorithms where necessary. Finally, I conducted thorough load testing to verify the performance improvements.

Your Answer

used tools like attitude and visual VM to identify slow database square root optimise with induction reduce UN necessity join and used catching to improve responsible

Feedback

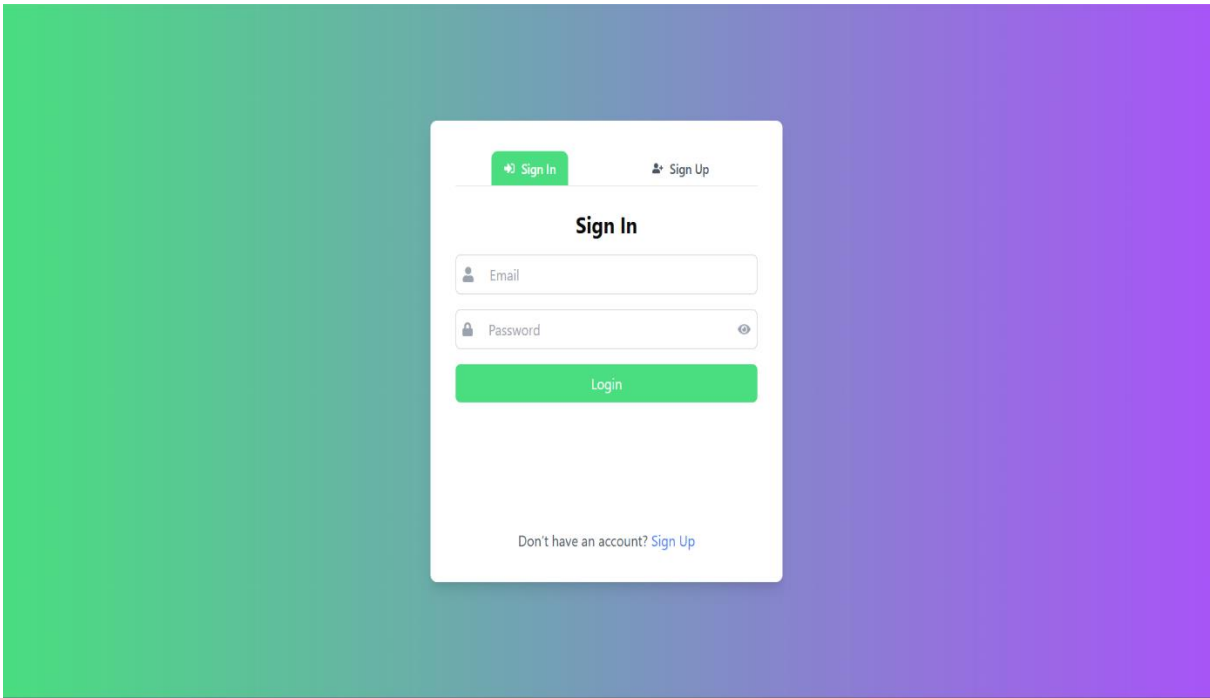
The user's answer touches on some relevant concepts (profiling tools, database optimization, caching, GC), but it's very vague and lacks detail. The answer mentions tools like 'attitude' (likely a typo for 'Attitude' or intended to be 'jconsole' or 'jvisualvm') and VisualVM, but doesn't explain 'how' they were used to identify the bottleneck. It mentions 'slow database square root' (likely meaning slow database queries), 'optimise with induction' (unclear meaning), 'reduce UN necessity join' (possibly referring to reducing unnecessary joins), and 'used catching to improve responsible' (likely meant caching). While some keywords are present, the response lacks a coherent narrative, specific examples, and a clear description of the steps taken to improve performance. To improve, the user should focus on providing a structured response that includes: 1) The initial problem/symptom. 2) The tools used for identification. 3) Specific examples of what was optimized (e.g., a particular query, a specific caching strategy). 4) The results of the optimization (e.g., reduced response time, improved throughput).

[Sign In](#) Sign Up

Sign Up

Register User

Already have an account? [Sign In](#)

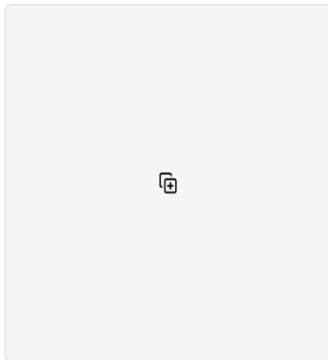


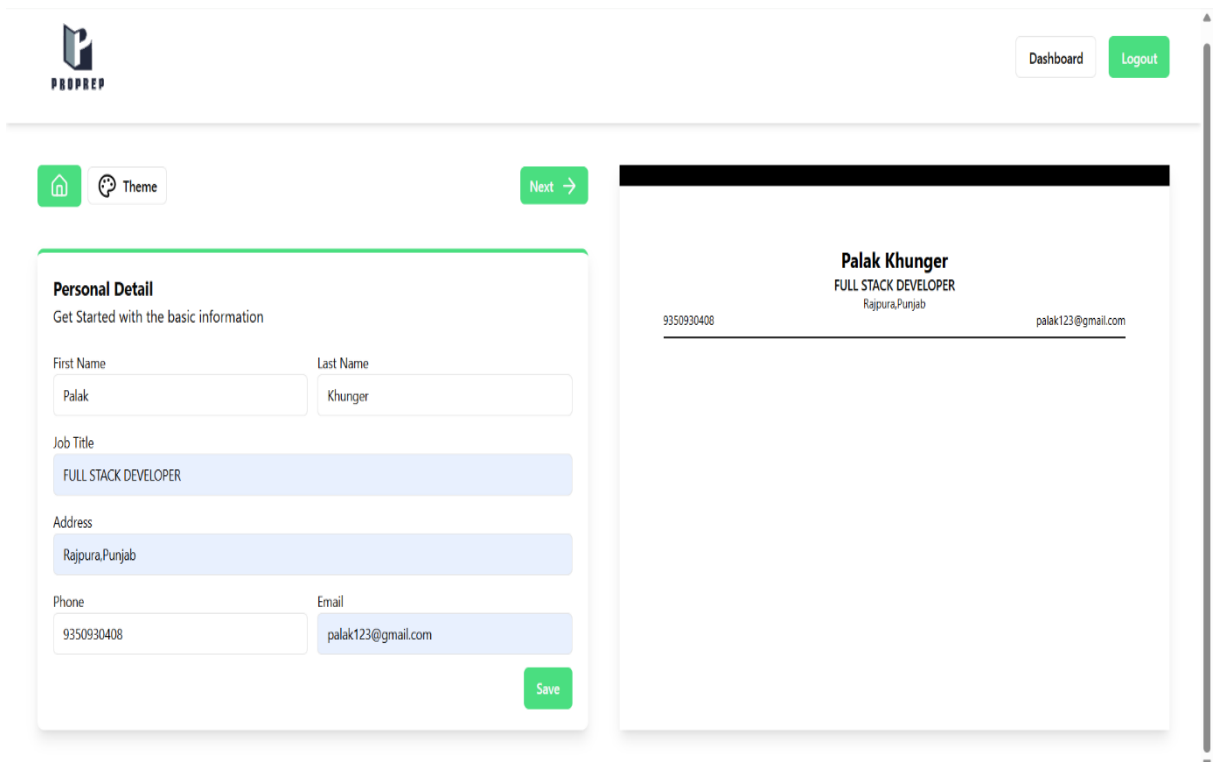
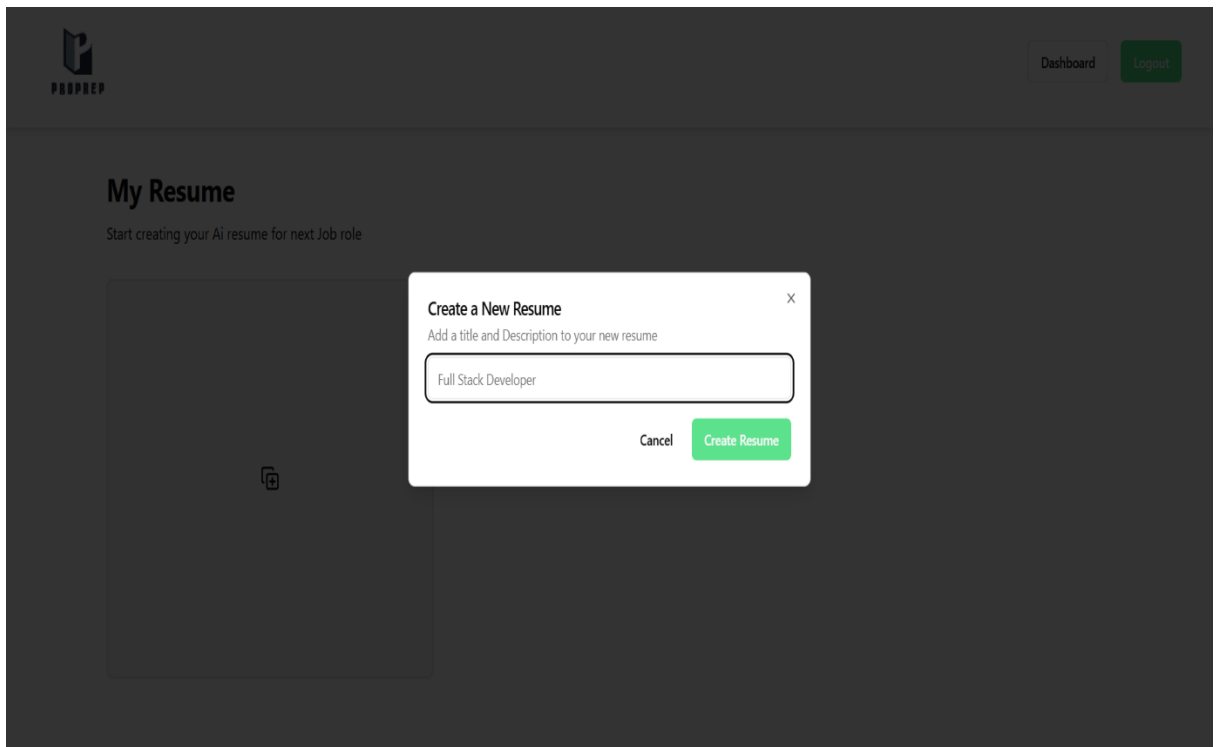
Dashboard

Logout

My Resume

Start creating your Ai resume for next Job role





Experience

Add Your Previous Job Experience

Experience 1



| | |
|----------------|--------------|
| Position Title | Company Name |
| Tech Intern | High Radius |
| City | State |
| Hyderabad | Telangana |
| StartDate | End Date |
| 26-01-2025 | 20-05-2025 |

Summary

[Generate from AI](#)

B *I* U **ab** **≡** **≡** [Link](#)

Palak Khunger

FULL STACK DEVELOPER

Rajpura,Punjab

9350930408

palak123@gmail.com

A highly skilled Full Stack Developer with 5+ years of experience in designing, developing, and deploying robust and scalable web applications. Proven ability to lead and mentor junior developers, and a strong understanding of agile development methodologies. Expertise in various programming languages and frameworks, including Java, Spring Boot, React, and Node.js. Passionate about creating innovative and user-friendly applications.

Professional Experience

Tech Intern

High Radius, Hyderabad, Telangana

2025-01-26 To 2025-05-20

- Assisted senior developers in designing, developing, and testing software applications.
- Gained practical experience in various programming languages, including Java, Python, and C++.
- Contributed to the development of company projects by performing tasks such as bug fixing and code maintenance.
- Developed and implemented innovative solutions to enhance the performance and usability of software applications.
- Collaborated effectively with team members in a fast-paced, agile development environment.
- Learned and applied industry best practices in software documentation.
- Presented project findings and technical solutions to

Resume Updated

[Show hidden icons](#)

Congrats! Your Ultimate AI generates Resume is ready !

Now you are ready to download your resume and you can share unique resume url with your friends and family

[Download](#)

[Share](#)

Palak Khunger

FULL STACK DEVELOPER

Rajpura,Punjab

9350930408

palak123@gmail.com

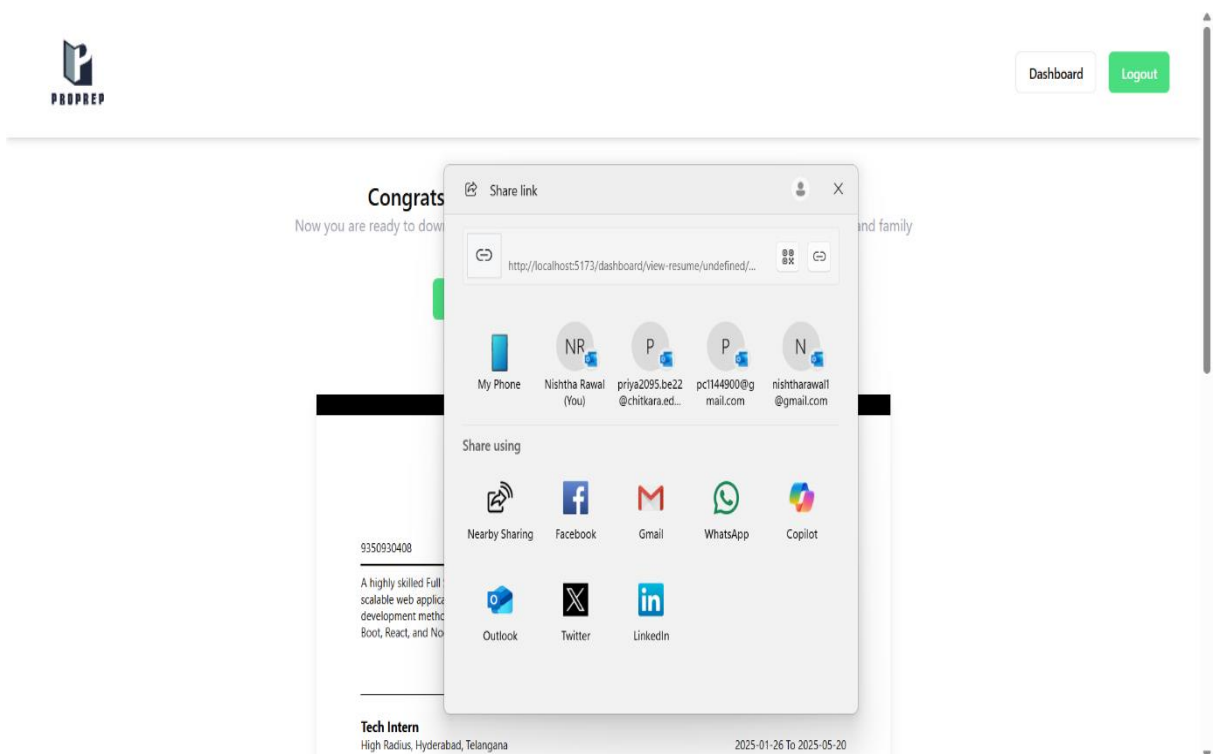
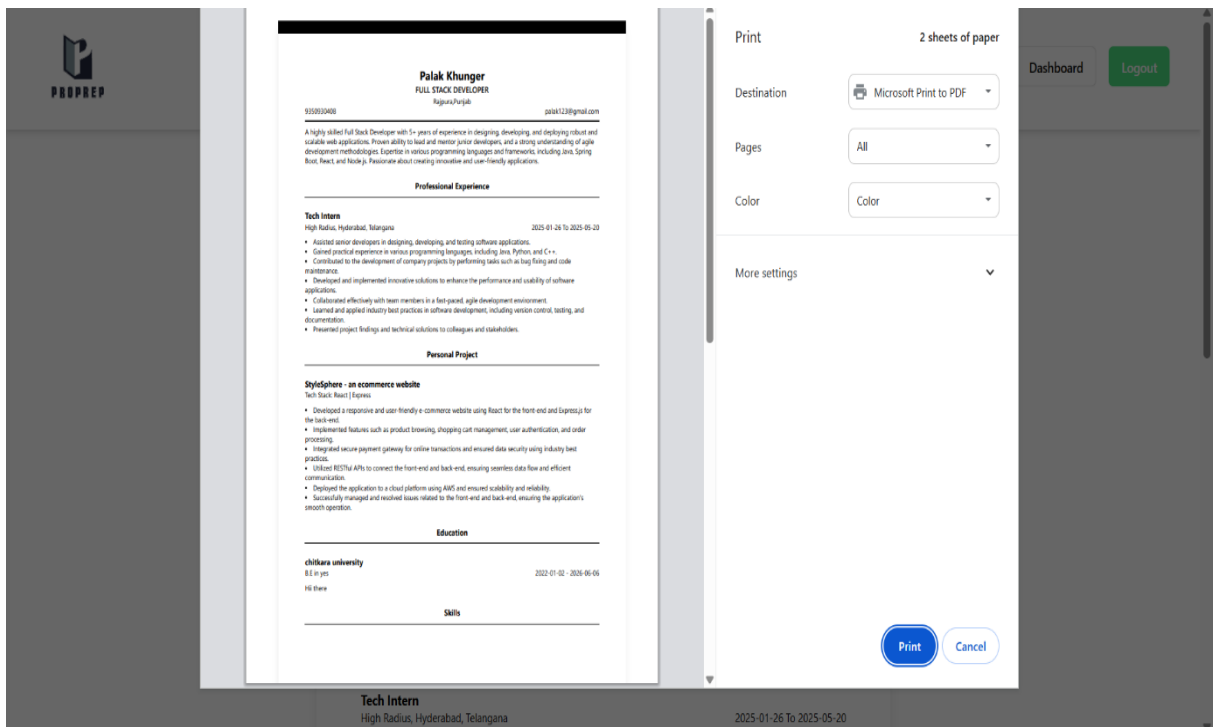
A highly skilled Full Stack Developer with 5+ years of experience in designing, developing, and deploying robust and scalable web applications. Proven ability to lead and mentor junior developers, and a strong understanding of agile development methodologies. Expertise in various programming languages and frameworks, including Java, Spring Boot, React, and Node.js. Passionate about creating innovative and user-friendly applications.

Professional Experience

Tech Intern

High Radius, Hyderabad, Telangana

2025-01-26 To 2025-05-20



6. Code-Implementation and Database Connections

- The ProPrep platform adopts a modular, full-stack architecture to ensure scalability, maintainability, and seamless user experience across both its AI Mock Interview and Resume Builder functionalities. The backend is developed using Next.js API routes, integrating Gemini AI for generating intelligent interview responses and resume enhancement suggestions. Clerk is used for secure user authentication and session management, ensuring a safe and personalized user environment. The frontend, built with Next.js and Tailwind CSS, offers an intuitive and responsive interface that supports dynamic resume creation, real-time interview simulations, and feedback visualization. API routes facilitate smooth communication between the frontend and backend for handling user data, interview sessions, resume generation, and feedback mechanisms.
- Database Connections: Database connectivity is handled through Drizzle ORM integrated with PostgreSQL, providing efficient and structured data management across both modules. The database schema is designed to store user profiles, resume data, interview sessions, AI-generated feedback, and session logs. API endpoints securely interact with PostgreSQL using Drizzle queries to perform CRUD operations and maintain data consistency. This setup ensures optimized data retrieval and manipulation based on user interactions, while supporting high availability and security. Overall, the system enables comprehensive tracking of both interview preparation progress and resume-building activities, forming a robust foundation for the platform's AI-driven career development tools.


```
ion View Go Run Terminal Help ← → IP_Project

@ App.jsx X
ai-mock-interview-react-vite-typescript-january-2025-main > src > @ App.jsx ...
 8 import AuthenticationLayout from "@layouts/auth-layout";
 9 import ProtectedRoutes from "@layouts/protected-routes";
10 import { MainLayout } from "@layouts/main-layout";
11
12 import HomePage from "@routes/home";
13 import { SignInPage } from "../routes/sign-in";
14 import { SignUpPage } from "../routes/sign-up";
15 import { Generate } from "../components/generate";
16 import { Dashboard } from "../routes/dashboard";
17 import { CreateEditPage } from "../routes/create-edit-page";
18 import { MockLoadPage } from "../routes/mock-load-page";
19 import { MockInterviewPage } from "../routes/mock-interview-page";
20 import { Feedback } from "../routes/feedback";
21
22 const App = () => {
23   return (
24     <Router>
25       <Routes>
26         <Route path="/" element={<PublicLayout />} />
27         <Route path="/home" element={<HomePage />} />
28         <Route path="/sign-in" element={<SignInPage />} />
29         <Route path="/sign-up" element={<SignUpPage />} />
30         <Route path="/generate" element={<Generate />} />
31         <Route path="/dashboard" element={<Dashboard />} />
32         <Route path="/create-edit" element={<CreateEditPage />} />
33         <Route path="/mock-load" element={<MockLoadPage />} />
34         <Route path="/mock-interview" element={<MockInterviewPage />} />
35         <Route path="/feedback" element={<Feedback />} />
36       </Routes>
37     </Router>
38   );
39 };
40
41 export default App;
```

```
lection View Go Run Terminal Help ← → IP_Project

@ dashboard.jsx X
ai-mock-interview-react-vite-typescript-january-2025-main > src > routes > @ dashboard.jsx > @ Dashboard
13 export const Dashboard = () => {
14   const [interview, setInterview] = useState<Interview>({});
15   const [loading, setLoading] = useState<boolean>(false);
16   const [userId, setUserId] = useState<string>("");
17
18   useEffect(() => {
19     setInterview({});
20     const interviewQuery = query(
21       collection(db, "interviews"),
22       where("userId", "==", userId)
23     );
24
25     const unsubscribe = onSnapshot(
26       interviewQuery,
27       (snapshot) => {
28         const interviewList: Interview[] = snapshot.docs.map((doc) => {
29           const id = doc.id;
30           return {
31             id,
32             ...doc.data(),
33           };
34         });
35         setInterviewList(interviewList);
36         setLoading(false);
37       },
38       (error) => {
39         console.log("Error on fetching :", error);
40         toast.error("Error", {
41           description: "Something went wrong.. Try again later...",
42         });
43         setLoading(false);
44       }
45     );
46
47     return () => unsubscribe();
48   }, [userId]);
49
50   return (
51     <div className="flex w-full items-center justify-between">
52       <div>
53         <h2>Dashboard</h2>
54         <p>Create and start your AI Mock Interview</p>
55       </div>
56       <div>
57         <button type="button" className="btn btn-primary">Generate</button>
58         <button type="button" className="btn btn-secondary">Add New</button>
59       </div>
60     </div>
61     <div>
62       <div>
63         <h3>Build your Resume</h3>
64         <p>Build your Resume</p>
65         <button type="button" className="btn btn-primary">Build</button>
66       </div>
67       <div>
68         <h3>Mock Interview</h3>
69         <p>Mock Interview</p>
70         <button type="button" className="btn btn-primary">Mock</button>
71       </div>
72     </div>
73   );
74 };
```

```
on View Go Run Terminal Help ← → IP_Project 08 11 11

dashboards X
#mock-interview-real-vite-typescript-january-2025-main > src > routes > dashboard > Dashboard

15 export const Dashboard = () => {
16   const [interviews, setInterviews] = useState<Interview[]>([]);
17   const [loading, setLoading] = useState(false);
18   const [userId] = useAuth();
19
20   useEffect(() => {
21     setloading(true);
22     const interviewQuery = query(
23       collection(db, "interviews"),
24       where("userId", "==", userId)
25     );
26
27     const unsubscribe = onSnapshot(
28       interviewQuery,
29       (snapshot) => {
30         const interviewList: Interview[] = snapshot.docs.map((doc) => {
31           const id = doc.id;
32           return {
33             id,
34             ...doc.data(),
35           };
36         }) as Interview[];
37         setInterviews(interviewList);
38         setloading(false);
39       },
40       (error) => {
41         console.log("Error on fetching : ", error);
42         toast.error("Error..", {
43           description: "Something went wrong.. Try again later..",
44         });
45         setloading(false);
46       }
47     );
48
49     return () => unsubscribe();
50   }, [userId]);
51
52   return (
53     <div className="flex w-full items-center justify-between">
54       <h2>Headings</h2>
55       <h3>title="Dashboard"
56         description="create and start you AI Mock interview"
57       </h3>
58       <Link to="/generate/create">
59         <Button size="sm">
60           <Plus /> Add New
61         </Button>
62       </Link>
63     </div>
64     <Separator className="my-8" />
65   );
66 }
```

```
on View Go Run Terminal Help ← → AI-Resume-Build-main 08 11 11

# usercontroller X
AI-Resume-Build-main > Backend > src > controller > # usercontroller.js > ...

1 import mongoose from "mongoose";
2 import User from "../models/user.model.js";
3 import jwt from "jsonwebtoken";
4 import { ApiResponse } from "../../utils/ApiResponse.js";
5 import { ApiError } from "../../utils/ApiError.js";
6
7 const start = async (req, res) => {
8   if (req.user) {
9     return res.status(200).json(new ApiResponse(200, req.user, "User Found"));
10   } else {
11     return res.status(404).json(new ApiResponse(404, null, "User Not Found"));
12   }
13 };
14
15 const registerUser = async (req, res) => {
16   console.log("Registration Started");
17   const { fullName, email, password } = req.body;
18
19   if (!fullName || !email || !password) {
20     console.log("Registration failed data insufficient");
21     return res
22       .status(400)
23       .json(
24         new ApiError(
25           400,
26           "Please provide all required fields: fullName, email, and password."
27         )
28       );
29   }
30
31   try {
32     const existingUser = await User.findOne({ email });
33
34     if (existingUser) {
35       console.log("Registration failed already registered user");
36       return res
37         .status(409)
38         .json(new ApiError(409, "User already registered."));
39     }
40
41     const newUser = await User.create({
42       fullName,
43       email,
44       password,
45     });
46
47     console.log("Registration Successful");
48     res.status(201).json(
49       new ApiResponse(
50         201,
51         {
52           user: {
53             id: newUser._id,
54             fullName: newUser.fullName,
55             email: newUser.email,
```

```

View Go Run Terminal Help ← → Ai-Resume-Buildier-main
App.jsx x
Ai-Resume-Buildier-main > Frontend > src > App.jsx > ...
1 import { Outlet, useNavigate } from "react-router-dom";
2 import Header from "../components/custom/Header";
3 import { Toaster } from "../components/ui/sonner";
4 import { useDispatch } from "react-redux";
5 import { useSelector } from "react-redux";
6 import { useEffect } from "react";
7 import { addUserData } from "../features/user/userFeatures";
8 import { startUser } from "../Services/login";
9 import { resumeStore } from "../store/store";
10 import { Provider } from "react-redux";
11
12 function App() {
13   const navigate = useNavigate();
14   const user = useSelector((state) => state.editUser.userData);
15   const dispatch = useDispatch();
16
17   useEffect(() => {
18     const fetchResponse = async () => {
19       try {
20         const response = await startUser();
21         if (response.statusCode == 200) {
22           dispatch(addUserData(response.data));
23         } else {
24           dispatch(addUserData(""));
25         }
26       } catch (error) {
27         console.log("Got Error while fetching user from app", error.message);
28         dispatch(addUserData(""));
29       }
30     };
31     fetchResponse();
32   }, []);
33
34   if (!user) {
35     navigate("/");
36   }
37
38   return (
39     <
40       <Provider store={resumeStore}>
41         <Header user={user} />
42         <Outlet />
43         <Toaster />
44       </Provider>
45     </

```

```

View Go Run Terminal Help ← → Ai-Resume-Buildier-main
main.jsx x
Ai-Resume-Buildier-main > Frontend > src > main.jsx > ...
1 import { createRoot } from "react-dom/client";
2 import App from "../App.jsx";
3 import "../index.css";
4 import { RouterProvider, createBrowserRouter } from "react-router-dom";
5 import HomePage from "../pages/home/HomePage.jsx";
6 import Dashboard from "../pages/dashboard/Dashboard.jsx";
7 import { EditResume } from "../pages/dashboard/edit-resume/[resume_id]/EditResume.jsx";
8 import ViewResume from "../pages/dashboard/view-resume/[resume_id]/ViewResume.jsx";
9 import AuthPage from "../pages/auth/customAuth/AuthPage.jsx";
10 import { resumeStore } from "../store/store";
11 import { Provider } from "react-redux";
12
13
14 const router = createBrowserRouter([
15   {
16     element: <App />,
17     children: [
18       {
19         path: "/dashboard",
20         element: <Dashboard />,
21       },
22       {
23         path: "/dashboard/edit-resume/:resume_id",
24         element: <EditResume />,
25       },
26       {
27         path: "/dashboard/view-resume/:resume_id",
28         element: <ViewResume />,
29       },
30     ],
31   },
32   {
33     path: "/",
34     element: <HomePage />,
35   },
36   {
37     path: "/auth/sign-in",
38     element: <AuthPage />,
39   },
40 ]);
41 ReactDOM.createRoot(document.getElementById("root")).render(
42   <
43     <Provider store={resumeStore}>
44       <React.StrictMode>
45         <RouterProvider router={router} />
46       </React.StrictMode>
47     </Provider>
48   </

```

```
on View Go Run Terminal Help ← → AI-Resume-Buildier-main 08 1 1 1 1
@ HomePage.jsx X
AI-Resume-Buildier-main > Frontend > src > pages > home > @ HomePage.jsx > ...
1 import Header from "@components/custom/Header";
2 import React, { useEffect } from "react";
3 import heroSnapshot from "@assets/heroSnapshot.png";
4 import { useNavigate } from "react-router-dom";
5 import { FaGithub, FaCircle, FaInfoCircle } from "react-icons/fa";
6 import { Button } from "@components/ui/Button";
7 import { startUser } from "../Services/login.js";
8 import { useDispatch, useSelector } from "react-redux";
9 import { addUserData } from "@Features/user/userFeatures.js";
10
11 function HomePage() {
12   const user = useSelector((state) => state.editUser.userData);
13   const navigate = useNavigate();
14   const dispatch = useDispatch();
15   const handleClick = () => {
16     window.open(
17       "https://github.com/sahidrajaansari/AI-Resume-Buildier",
18       "_blank"
19     );
20   };
21
22   useEffect(() => {
23     const fetchResponse = async () => {
24       try {
25         const response = await startUser();
26         if (response.statusCode === 200) {
27           dispatch(addUserData(response.data));
28         } else {
29           dispatch(addUserData(""));
30         }
31       } catch (error) {
32         console.log(
33           "Printing from Home Page there was a error ->",
34           error.message
35         );
36         dispatch(addUserData(""));
37       }
38     };
39     fetchResponse();
40   }, []);
41
42   const handleGetStartedClick = () => {
43     if (user) {
44       console.log("Printing from Homepage User is There ");
45       navigate("/dashboard");
46     } else {
47       console.log("Printing from Homepage User Not Found");
48       navigate("/auth/sign-in");
49     }
50   };
51   return (
52     <>
53     <Header user={user} />
54     <section className="pt-24 pb-20 bg-white">
55       <div className="px-12 mx-auto max-w-7xl">
```

```
@ AuthPage.jsx X
AI-Resume-Buildier-main > Frontend > src > pages > auth > customAuth > @ AuthPage.jsx > ...
1 import React, { useState } from "react";
2 import {
3   FaUser,
4   FaLock,
5   FaSignInAlt,
6   FaUserPlus,
7   FaEye,
8   FaEyeSlash,
9 } from "react-icons/fa";
10 import { motion } from "framer-motion";
11 import { loginUser, registerUser } from "@Services/login";
12 import { Loader2 } from "lucide-react";
13 import { useNavigate } from "react-router-dom";
14
15 function AuthPage() {
16   const [isLoading, setIsLoading] = useState(false);
17   const [showPassword, setShowPassword] = useState(false);
18   const [signInError, setSignInError] = useState("");
19   const [email, setEmail] = useState("");
20   const [password, setPassword] = useState("");
21   const [signInError, setSignInError] = useState("");
22   const [loading, setLoading] = useState(false);
23   const navigate = useNavigate();
24
25   const handleSignInSubmit = async (event) => {
26     setSignInError("");
27     event.preventDefault();
28     const { email, password } = event.target.elements;
29
30     const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
31     if (!emailPattern.test(email.value)) {
32       setError("Please enter a valid email address.");
33       return;
34     }
35
36     setLoading(true);
37     const data = {
38       email: email.value,
39       password: password.value,
40     };
41
42     try {
43       console.log("Login Started in Frontend");
44       const user = await loginUser(data);
45       console.log("Login Completed");
46
47       if (user?.statusCode === 200) {
48         navigate("/");
49       }
50       console.log(user);
51     } catch (error) {
52       setSignInError(error.message);
53       console.log("Login Failed");
54     } finally {
55       setLoading(false);
56     }
57   };
58 }
```

```

AuthPage.jsx X
AI Resume Builder main > Frontend > src > pages > auth > customAuth > AuthPage.jsx > ...
15 function AuthPage() {
16   const handleSignInSubmit = async (event) => {
17     const data = {
18       email: email.value,
19       password: password.value,
20     };
21     try {
22       const response = await registerUser(data);
23       if (response?.statusCode === 201) {
24         console.log("User Registration Started");
25         handleSignInSubmit(event);
26       }
27     } catch (error) {
28       console.log("User Registration Failed");
29       setSignInError(error.message);
30     } finally {
31       setloading(false);
32     }
33   };
34   return (
35     <div className="flex flex-col items-center justify-center min-h-screen p-6 bg-gradient-to-r from-green-400 to-purple-500">
36       <motion.div
37         className="relative w-full max-w-md p-8 bg-white rounded-lg shadow-lg"
38         initial={{ opacity: 0, y: -50 }}
39         animate={{ opacity: 1, y: 0 }}
40         transition={{ duration: 0.5 }}
41       >
42         <div className="flex justify-around mb-6 border-b border-gray-200">
43           <button
44             onClick={() => setSignIn(false)}
45             className="flex items-center gap-2 px-4 py-2 text-sm font-semibold transition-colors duration-300 rounded-t-lg ${{
46               isSignIn ? "bg-green-400 text-white" : "text-gray-600"
47             }}
48           >
49             <FaSignInAlt />
50             Sign In
51           </button>
52           <button
53             onClick={() => setSignIn(true)}
54             className="flex items-center gap-2 px-4 py-2 text-sm font-semibold transition-colors duration-300 rounded-t-lg ${{
55               isSignIn ? "bg-green-400 text-white" : "text-gray-600"
56             }}
57           >
58             <FaUserPlus />
59             Sign Up
60           </button>
61         </div>
62         <div className="relative overflow-hidden h-80">
63           <"/>
64           </* Added height to ensure content is visible */>
65         </div>
66       </motion.div>
67     </div>
68   );
69 }

```

7. Limitations

While ProPrep provides an innovative and practical solution for job seekers through AI-driven resume building and mock interviews, the platform is still evolving. Like any product in its early stages, ProPrep has certain limitations that may affect user experience or restrict its current scope of functionality. Below are some of the most relevant limitations along with the context of why they exist and how they impact users.

- Absence of Live Video Interview Functionality:** Currently, ProPrep supports only text-based AI mock interview simulations. While this mode offers a solid foundation for interview preparation—especially for users who prefer typing out answers and reading feedback—it lacks the real-time pressure and spontaneity that are critical aspects of actual face-to-face interviews.

Many users have expressed the desire to simulate real-world interview conditions, such as responding to questions on camera, managing body language, and maintaining eye contact—all of which are vital soft skills during the hiring process. The absence of a live video-based mock interview limits the platform's ability to prepare candidates for

roles that require strong communication and presentation skills, such as sales, customer success, consulting, and executive roles.

Moreover, in text-based simulations, users can take their time to craft responses, whereas real interviews demand quick thinking and verbal articulation. This is a major gap that we plan to bridge by integrating video conferencing tools like WebRTC, ZegoUIKit, or Twilio Video SDK. These technologies will allow users to practice in realistic, timed scenarios—either solo with AI avatars or with real-time human feedback.

Until then, users seeking to practice live behavioral interviews or presentation-based interviews may need to rely on external video platforms, limiting the convenience and cohesiveness that ProPrep strives to offer.

- **Limited Customization in AI Feedback and Resume Suggestions:** Another important limitation lies in the scope of customization available to users when interacting with the AI modules. While Gemini AI currently generates intelligent, structured responses for mock interviews and gives helpful resume content suggestions, the responses are generic in nature.

For example, a candidate applying for a marketing role at a startup will receive feedback that is not significantly different from that provided to someone applying for a software engineering job at a large corporation. In the real world, however, resumes and interview responses must be tailored to job roles, industry standards, experience levels, and company culture.

- Currently, users cannot:
- Adjust interview difficulty (e.g., beginner, intermediate, expert)
- Select industry-specific mock interview question sets (e.g., tech, healthcare, law, finance)
- Customize resume sections according to industry-specific trends or requirements

This lack of personalization may make the platform less impactful for advanced users or professionals with niche roles. As a result, users may feel the need to edit AI-generated content manually, which reduces the automation advantage that ProPrep aims to provide.

In future iterations, we plan to introduce advanced customization controls, enabling users to:

- Choose their experience level and job type before starting an interview

- Select from curated resume templates tailored to specific industries
- Adjust tone and structure based on job geography (e.g., US vs. EU standards)
- Provide context about a specific job posting to refine AI responses
- **Lack of Multi-Language Support:** At present, ProPrep operates exclusively in English, which makes it inaccessible to a large number of users who are either non-native English speakers or more comfortable preparing in their local languages.

This is particularly restrictive for users in countries like India, Germany, Japan, Spain, Brazil, and many others, where local language resumes and interviews are common in the domestic job market. It also alienates users seeking jobs in international regions but preferring preparation in their native language for better comprehension and confidence.

- Some of the problems this creates include:
- Users having to translate content manually post-generation, risking inaccuracies
- Difficulty in fully understanding AI feedback for non-fluent English speakers
- Limited access for rural or underserved populations that aren't English-proficient
- Additionally, the job search ecosystem is becoming increasingly global, and platforms like ProPrep need to support multi-lingual, multi-cultural experiences to stay competitive and inclusive.
- To address this, future updates will aim to integrate multi-language capabilities, allowing users to:
- Choose their preferred language for the entire platform UI
- Generate resumes and get feedback in local languages (e.g., Hindi, Spanish, French, Mandarin)
- Practice interviews in multiple languages for both domestic and international roles

This will be achieved by leveraging language models capable of translation and localization, such as Google Translate API, Amazon Translate, or multilingual capabilities built into Gemini AI and other LLMs.

8. Conclusion

The development of ProPrep, an innovative AI-powered career preparation platform, represents a significant leap in bridging the gap between job seekers and their dream careers. With its dual-core functionalities—AI Mock Interviews and the AI Resume Builder—ProPrep has evolved into more than just a tool; it is a digital mentor guiding users through one of the most critical phases of their professional lives. In an increasingly competitive and fast-paced job market, standing out requires more than just qualifications—it demands confidence, personalization, and preparation. ProPrep addresses this need with precision, scalability, and a deep focus on user empowerment.

The backend system, engineered using Next.js API routes, Clerk for authentication, Gemini AI for intelligence, and Drizzle ORM with PostgreSQL for data handling, ensures that the platform is not only secure and reliable but also smart and responsive. It seamlessly manages everything from user onboarding and resume content generation to real-time analysis of interview responses and activity logging. The use of AI-driven insights allows the system to adapt to each user's career background, tailoring suggestions that align with their goals and industry expectations.

On the frontend, React (Next.js) and Tailwind CSS come together to create a visually appealing and highly intuitive interface that promotes ease of use. Whether users are building a professional resume from scratch or practicing interview questions in a simulated environment, the experience is smooth, engaging, and rewarding. Features like dynamic component rendering, responsive layouts, and interactive feedback mechanisms ensure that users remain fully immersed in their journey toward job readiness.

Additionally, ProPrep plans to integrate career analytics dashboards, where users can track their progress, receive suggestions for improvement, and analyze performance trends across resume edits and mock interview sessions. This will help users better understand their strengths and weaknesses, enabling continuous improvement over time.

Beyond the technical advancements, what truly sets ProPrep apart is its commitment to user-centered design and accessibility. The platform acknowledges that every user has a unique journey, different career aspirations, and varying levels of confidence. By offering a blend of automation, personalization, and empathy, ProPrep positions itself as not just a product but a career companion. Whether someone is a college graduate entering the job market for the first

time, a mid-career professional seeking new opportunities, or someone returning to the workforce after a break, ProPrep offers practical tools and moral support through technology. Despite several features still in development, the current foundation is both functional and future-ready. Users already benefit from AI-curated resume suggestions, interview simulations tailored to specific roles, and a polished UI that enhances focus and usability. The platform has proven its capability to help users build confidence, improve communication skills, and stand out to potential employers in a crowded job pool.

In summary, ProPrep is not just solving a problem—it is reshaping the way people prepare for their careers. By combining modern web technologies with artificial intelligence and user-first design, it offers a one-stop solution to a widespread challenge. With an eye on continuous improvement and inclusivity, ProPrep is poised to become a long-term ally in every job seeker's career journey, helping them not only to get hired but to grow, thrive, and succeed in their chosen path.

9. Future Scope

The vision for ProPrep extends far beyond its current capabilities, aiming to evolve into a complete digital ecosystem for career development that adapts to individual needs and industry trends. Several promising advancements are on the roadmap to make the platform more intelligent, user-centric, and globally accessible.

- **Personalized AI Feedback & Reports:** Future iterations of ProPrep will introduce more in-depth, data-driven feedback mechanisms. Instead of generic responses, users will receive performance reports enriched with visual analytics—including heatmaps, score breakdowns, and behavioral patterns during mock interviews. These reports will be dynamically generated after each session and stored in the user dashboard for continuous tracking. Additionally, custom learning paths will be designed based on individual strengths, weaknesses, and career goals. For instance, if a user consistently struggles with behavioral interview questions, the system will curate exercises and tips tailored to improving those areas. This personalized approach ensures that every user receives not just feedback—but growth guidance.
- **AI Resume Analysis & Smart Job Recommendations:** In the near future, the platform will include a smart resume scanner and analysers, capable of highlighting weak points in structure, wording, formatting, and keyword density. It will help users align their resumes with the latest ATS (Applicant Tracking System) standards

used by top employers. Moreover, a smart job matching engine will be introduced. This feature will automatically match users with relevant job openings based on their skills, past experiences, resume content, and preferences. These job matches will be tailored in real-time using AI models trained on current labor market trends, job descriptions, and hiring patterns—providing users with a faster, smarter path to employment.

- **Interactive Interview Preparation Tools:** ProPrep will introduce voice-based and video-based interview simulations, where users can interact with AI avatars or record their responses and receive tone analysis, facial expression feedback, and speaking pace evaluations. This will add a more realistic and immersive layer to interview practice. Additionally, users will be able to choose from a range of difficulty levels, industries, and job roles, making the mock interview experience more aligned with their actual targets.
- **Gamification & Skill Progression System:** To increase user engagement and encourage consistent use, gamified features like achievement badges, progress levels, leaderboards, and weekly challenges will be introduced. Users will be able to unlock advanced content and resume templates based on their activity level, performance improvements, and consistency—turning the preparation journey into an enjoyable and rewarding experience.
- **Collaborative Learning & Peer Review:** A future goal includes launching a community feature, where users can connect, collaborate, and give feedback on each other's resumes and mock interviews. Peer review will be moderated and supplemented by AI insights, helping users gain multiple perspectives and learn from real-time examples. There will also be mentor-integration features where professionals can volunteer to guide beginners, bringing a human touch to the platform's AI-centric approach.
- **Accessibility & Localization:** Recognizing the diverse backgrounds of its users, ProPrep will introduce multi-language support, including local dialects and regional formatting preferences for resumes. The platform will also implement text-to-speech, screen reader compatibility, and dark mode UI options to support users with visual or cognitive challenges—ensuring inclusivity and compliance with global accessibility standards like WCAG.
- **Integration with Professional Networks & Job Boards:** To help users take actionable steps post-preparation, ProPrep aims to integrate directly with platforms

like LinkedIn, Indeed, and Naukri.com, allowing users to apply for jobs directly from within the platform, using their optimized resumes and interview performance metrics as part of the application profile.

- **Scalable and Flexible Deployment:** Although the deployment of ProPrep is yet to be executed, detailed planning has been done to ensure that the platform will be scalable, flexible, and capable of handling increasing user demands across different regions. The current development environment is optimized for seamless deployment, and various options are being evaluated to guarantee high availability, fast response times, and robust performance once the platform goes live.

10. Bibliography/References

- <https://nodejs.org/en>
- <https://nextjs.org/>
- <https://orm.drizzle.team/docs/overview>
- <https://ai.google.dev/>
- <https://clerk.com/docs>
- <https://tailwindcss.com/docs/installation/using-vite>
- <https://www.postgresql.org/docs/>
- <https://nextjs.org/docs/pages/building-your-application/routing/api-routes>
- <https://clerk.dev/docs>
- <https://ai.google.dev/gemini-api/docs>
- <https://code.visualstudio.com/docs>
- <https://www.postman.com/product/api-client/>
- <https://docs.github.com/en>
- <https://git-scm.com/doc>

