

**Janardhan Bhagat Shikshan Prasarak Sanstha’s**

**CHANGU KANA THAKUR ARTS, COMMERCE AND SCIENCE COLLEGE NEW PANVEL(Autonomous)**

**PROJECT ON**

**AI Career Coach**

**DEVELOPED BY**

**Mr. Anuj Rajkumar Prajapati**

**UNDER THE GUIDANCE OF**

**Mr. Sagar Vyavahare**

**ACADEMIC YEAR**

### **2024-2025**

**Department of Computer Science**

## CERTIFICATE

**This is to certified that the project entitled**

**“AI Career Coach”**

Is successfully completed by **Mr. Anuj Rajkumar Prajapati,** Roll No: 80, Examination No: **CS24677** under the guidance of **Mr. Sagar Vyavahare**, during the academic period of 3rd Dec 2024 to 5th Feb 2025 as per the syllabus, fulfilment for the completion of the BSc degree in the Computer Science. It is also to certify that this is original work of the candidate done during academic year 2024-2025.

**Place:** Panvel

**Date: 10/02/2025**

**Internal Examiner Principal**

**External Examiner Head of Department**

# ACKNOWLEDGEMENT

It is indeed a matter of great pleasure and proud privilege to be able to present this project on “**AI Career Coach**”.

I would also like to express my deep regards and the gratitude towards the principal **Prof. (Dr.) S.K Patil.**

I respect and thank Head of the department **Prof. Mrs. Pratibha Jadhav**, for providing me an opportunity to do the project work and giving me all the support and guidance. Also, I would like to tender our sincere thanks to all the teachers for their co-operation.

The completion of the project work is a milestone in student life and it execution is inevitable in the hands of guide. I am highly indebted the projects guide **Mr. Sagar Vyavahare** for her invaluable guidance and appreciation for giving form and substance to this report. It is due to her enduring efforts; patience and enthusiasm, which has given a sense of direction and purposefulness to this project and ultimately made it a Success.

I would Wish to thank the non - teaching staff and my friends who have helped me.

Really it is highly impossible to repay the dept of all the people who have me all the time in one way or the other directly or indirectly helped me for performing the project.

# INDEX

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page No.** |
| **01** | **Introduction** | **05** |
| **02** | **Requirement Specification** | **10** |
| **03** | **System Design** | **12** |
| **04** | **System Implementation** | **16** |
| **05** | **Result** | **41** |
| **06** | **Conclusion and Future Scope** | **44** |
| **07** | **Plagiarism Report** | **47** |
| **08** | **References** | **49** |



**INTRODUCTION**

1. **Introduction:**

In today’s competitive job market, finding the right career opportunities and preparing for them can be challenging. **AI Career Coach** is an intelligent web- based application designed to assist job seekers in streamlining their job search process. This project leverages advanced AI technology to provide **personalized career insights, automated resume and cover letter generation, and interview preparation** tailored to the user’s industry and skills.

Built using modern technologies such as **React 19, Next.js 15, Tailwind CSS, Prisma, NeonDB, Clerk Authentication, Inngest, Gemini API, and Shadcn UI**, the application ensures a seamless and efficient user experience.

This web application is provide to advance your career with personalized guidance, interview preparation, and AI-powered tools for job success.By leveraging artificial intelligence, this platform helps users create professional job application materials and gain industry-specific knowledge, making them more confident and prepared for their career journey.

The **Industry Insights** feature provides users with AI-generated insights based on their profile and chosen industry. By analyzing market trends, required skills, and job opportunities, the system helps users stay informed about the latest developments in their field. This enables job seekers to make well-informed career decisions and tailor their applications accordingly.

The **Resume & Cover Letter Generator** assists users in creating professional job application materials effortlessly. By gathering user input, such as experience, skills, and job preferences, the AI crafts tailored resumes and cover letters that align with industry standards. This feature ensures that job seekers present themselves in the best possible way, increasing their chances of securing interviews.

The **Interview Preparation** module helps users get ready for job interviews by generating industry-specific questions and suggested responses. This AI-powered tool provides realistic interview scenarios, allowing users to practice effectively and improve their confidence. By simulating real interview conditions, this feature ensures that job seekers are well-prepared to tackle challenging questions and make a strong impression on potential employers.

## Existing System:

In the current job market, job seekers rely on multiple platforms and manual efforts to prepare for their careers. They search for industry trends through various websites, create resumes and cover letters using generic templates, and practice interviews with limited resources. Many people struggle to format their resumes professionally, highlight key skills effectively, and tailor their applications to specific job roles. Additionally, interview preparation often involves researching common questions without personalized feedback or industry-specific insights. This fragmented approach makes the job search process time-consuming and inefficient, leading to missed opportunities and increased stress for job seekers.

## Proposed System:

**AI Career Coach** aims to address these challenges by integrating all essential job preparation tools into a single, AI-powered platform. The system provides **real-time industry insights** based on user profiles, helping job seekers stay informed about the latest trends and required skills in their field. The **AI-driven resume and cover letter generator** creates professionally formatted documents tailored to each user’s background, ensuring a higher chance of catching recruiters' attention. Additionally, the **interview preparation module** generates industry-specific questions and suggested responses, allowing users to practice effectively.

Built with modern technologies like **React 19, Next.js 15, Tailwind CSS, Prisma, NeonDB, Clerk Authentication, Inngest, Gemini API, and Shadcn UI**, the proposed system offers a seamless, efficient, and intelligent solution for job seekers, making the career preparation process more structured and effective.

## Working of the Online Learning System

The **AI Career Coach** follows a structured workflow that guides users through career planning, resume creation, and interview preparation using AI-powered tools. Below is a step-by-step breakdown of how the system functions:

### **1. User Authentication & Onboarding**

* Users sign up or log in through **Clerk authentication**, ensuring secure access.
* After authentication, they complete an **onboarding form** that collects essential details like career preferences, skills, and industry selection.
* The form uses **React Hook Form and Zod** for validation, ensuring accurate data input.

### **2. Industry Insights Generation**

* Once onboarding is complete, the system uses the **Gemini API** to generate **real-time industry insights** based on the user’s selected field.
* Users receive AI-driven career suggestions, job market trends, and required skills to improve their employability.

### **3. Resume & Cover Letter Builder**

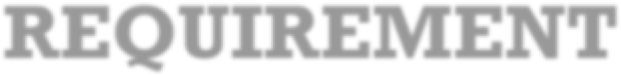
* The user can generate a **resume** by providing input such as experience, education, and skills.
* The AI-powered **resume builder** formats the details into a professional layout.
* The system also generates a **customized cover letter**, aligning with the job role and industry requirements.

### **4. Interview Preparation & Performance Tracking**

* Users can access an **AI-powered interview preparation module**, which generates **industry-specific questions and suggested responses**.
* The system tracks user progress by providing a **quiz assessment**, allowing them to evaluate their skills and readiness for interviews.
* Performance tracking enables users to monitor their improvements over time and refine their interview skills.

### **5. Data Storage & Management**

* All user information, including industry insights, resumes, and quiz results, is stored securely using **Prisma ORM with PostgreSQL**.
* Prisma ensures efficient database management and smooth data retrieval.



# REQUIREMENT SPECIFICATION

1. **Requirement Specification:**

## Software Requirements:

#### ****1.Frontend****

* **Framework:** Next.js 15 (React 19)
* **UI Library:** Tailwind CSS & Shadcn UI
* **Form Validation:** React Hook Form & Zod
* **Authentication:** Clerk
* **State Management:** React Context API or Zustand

#### ****2.Backend****

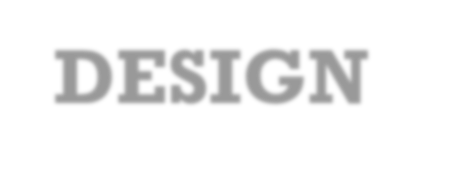
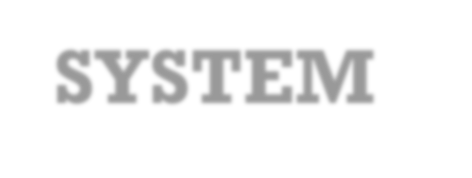
* **Server-side Framework:** Next.js API Routes
* **Database Management:** Prisma ORM
* **Database:** NeonDB (PostgreSQL)
* **AI Integration:** Gemini API
* **Task Scheduling & Execution:** Inngest

#### ****3.Other Software Dependencies****

* **Node.js (v18 or higher)** – Required for running Next.js and backend services.
* **NPM or Yarn** – Package management for dependencies.
* **IDE/Text Editor:** Visual Studio Code (VS Code)

## Hardware Requirement:

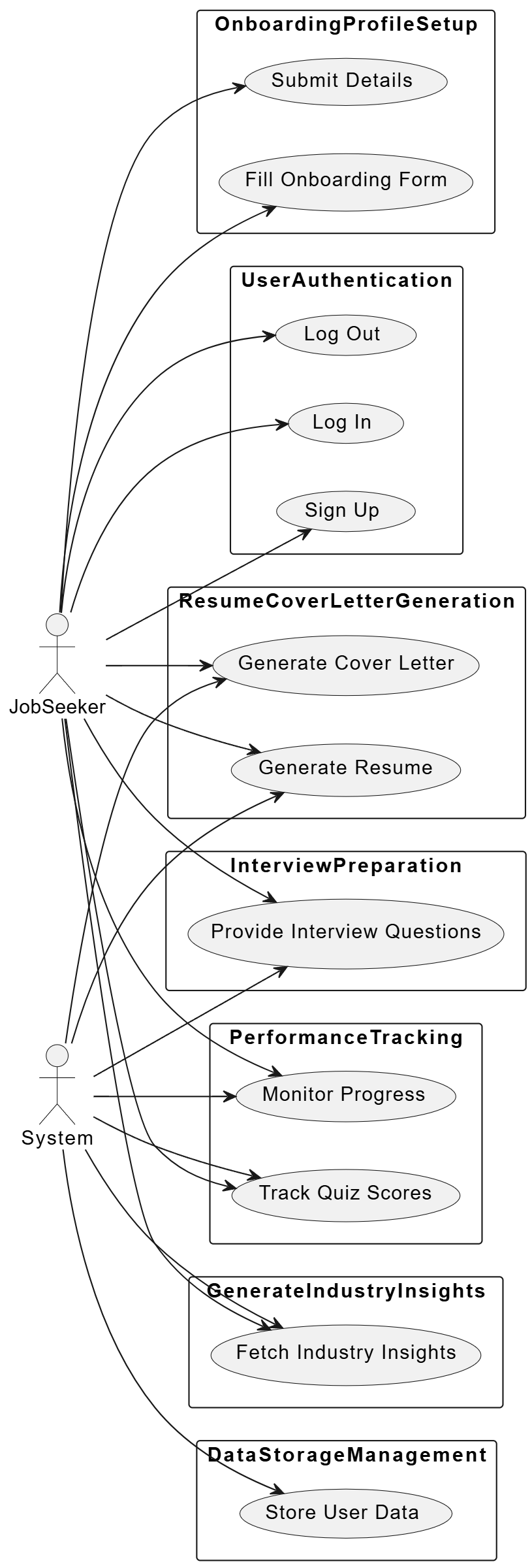
1. **Processor:** Intel Core i5 (8th Gen or later) / AMD Ryzen 5 or equivalent.
2. **Storage: 2** GB
3. **Ram: 8** GB
4. **Internet:** Stable connection for API calls and cloud services.
5. **Display:** 1080p resolution monitor.



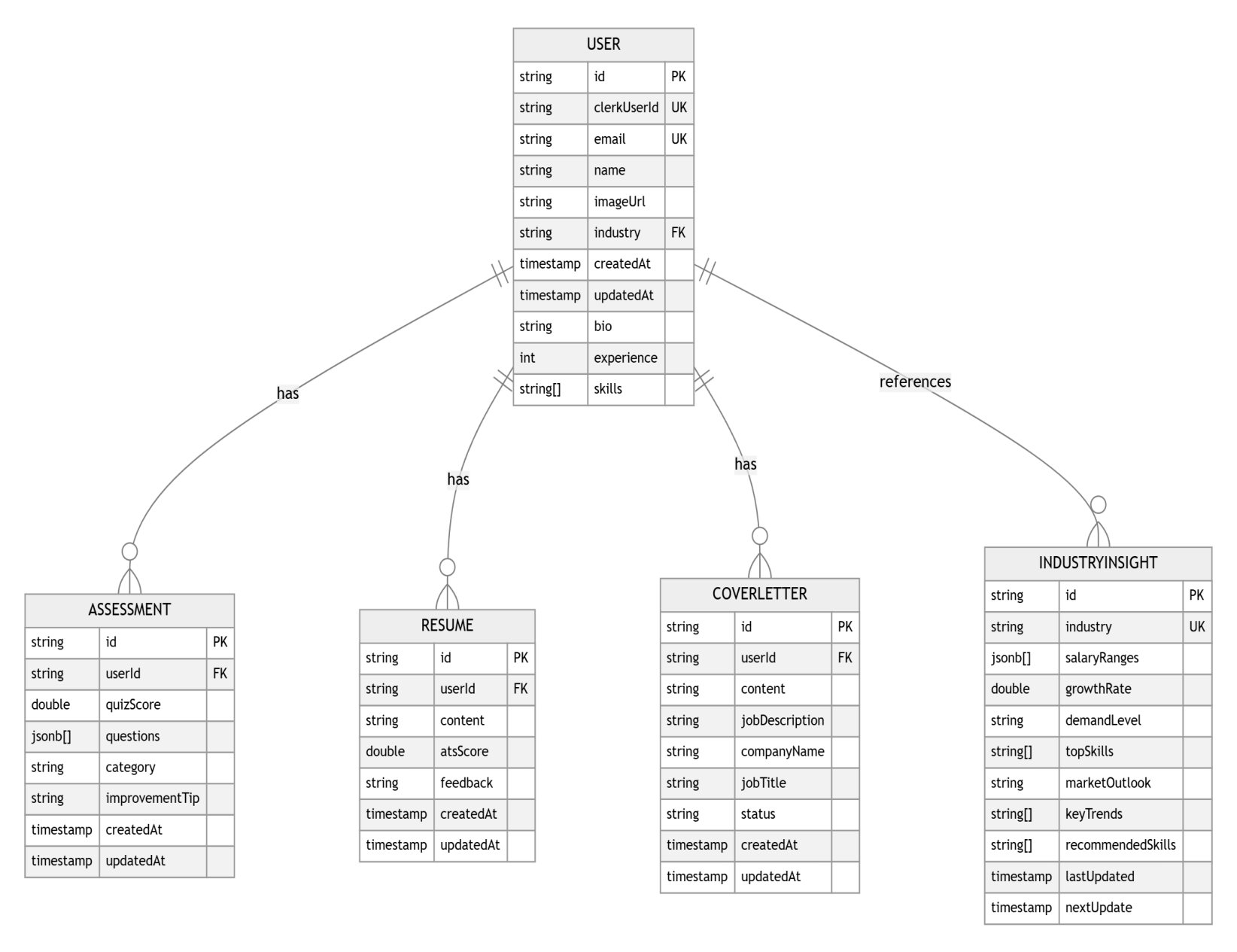
# SYSTEM DESIGN

**3. System Design Details:**

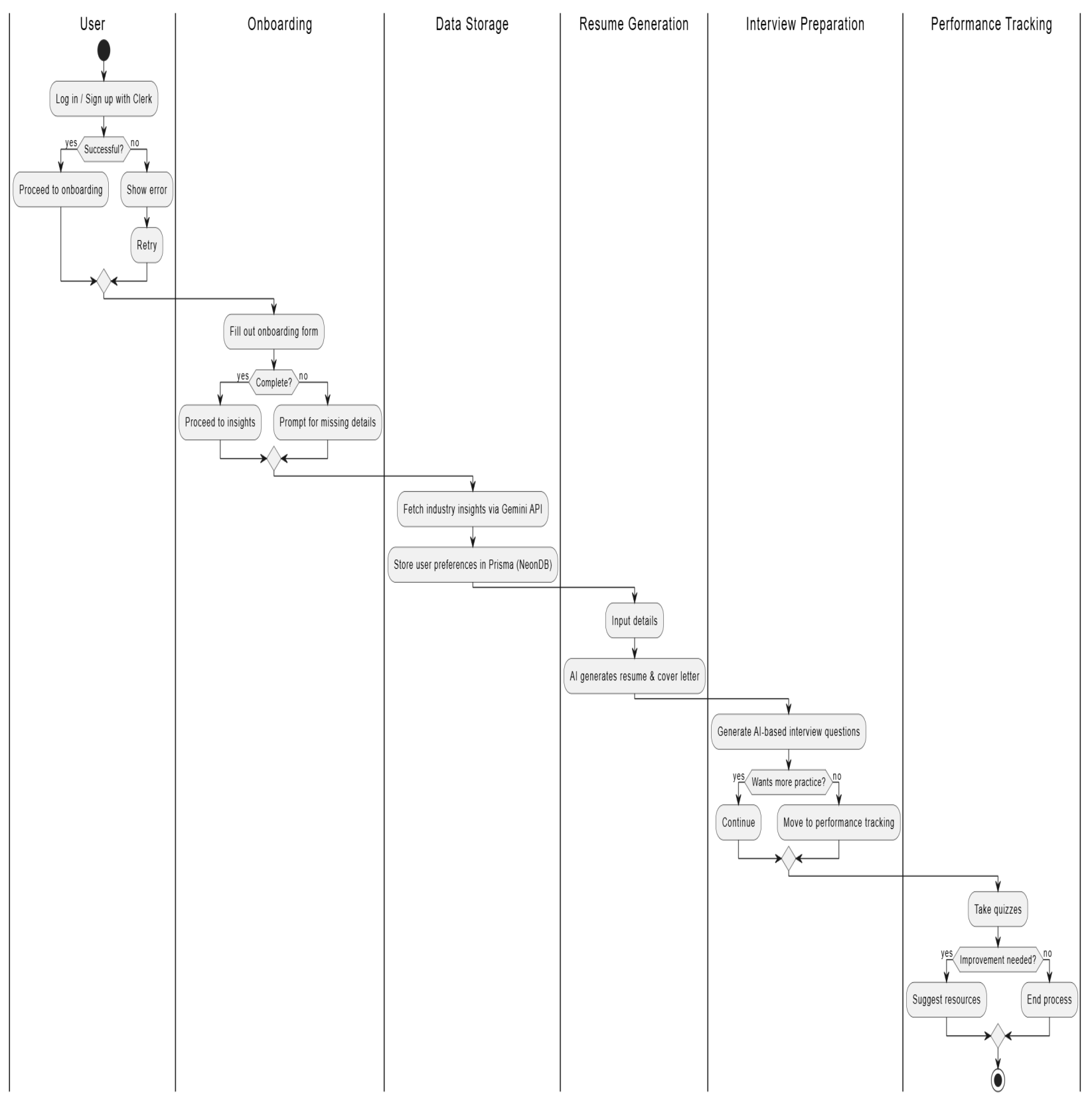
* 1. **Use-Case Diagram**

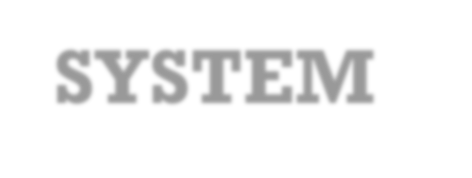


* 1. **Entity Relationship Diagram**



* 1. **Activity Diagram**





# SYSTEM IMPLEMENTATION

1. **System Implementation:**

**App/page.jsx :**

import React from "react";

import Link from "next/link";

import { Button } from "@/components/ui/button";

import { Card, CardContent } from "@/components/ui/card";

import {

  ArrowRight,

  Trophy,

  Target,

  Sparkles,

  CheckCircle2,

} from "lucide-react";

import HeroSection from "@/components/hero";

import {

  Accordion,

  AccordionContent,

  AccordionItem,

  AccordionTrigger,

} from "@/components/ui/accordion";

import Image from "next/image";

import { features } from "./data/features";

import { testimonial } from "./data/testimonial";

import { faqs } from "./data/faqs";

import { howItWorks } from "./data/howItWorks";

export default function LandingPage() {

  return (

    <>

      <div className="grid-background"></div>

      {/\* Hero Section \*/}

      <HeroSection />

      {/\* Features Section \*/}

      <section className="w-full py-12 md:py-24 lg:py-32 bg-background">

        <div className="container mx-auto px-4 md:px-6">

          <h2 className="text-3xl font-bold tracking-tighter text-center mb-12">

            Powerful Features for Your Career Growth

          </h2>

          <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6 max-w-6xl mx-auto">

            {features.map((feature, index) => (

              <Card

                key={index}

                className="border-2 hover:border-primary transition-colors duration-300"

              >

                <CardContent className="pt-6 text-center flex flex-col items-center">

                  <div className="flex flex-col items-center justify-center">

                    {feature.icon}

                    <h3 className="text-xl font-bold mb-2">{feature.title}</h3>

                    <p className="text-muted-foreground">

                      {feature.description}

                    </p>

                  </div>

                </CardContent>

              </Card>

            ))}

          </div>

        </div>

      </section>

      {/\* Stats Section \*/}

      <section className="w-full py-12 md:py-24 bg-muted/50">

        <div className="container mx-auto px-4 md:px-6">

          <div className="grid grid-cols-2 md:grid-cols-4 gap-8 max-w-4xl mx-auto text-center">

            <div className="flex flex-col items-center justify-center space-y-2">

              <h3 className="text-4xl font-bold">50+</h3>

              <p className="text-muted-foreground">Industries Covered</p>

            </div>

            <div className="flex flex-col items-center justify-center space-y-2">

              <h3 className="text-4xl font-bold">1000+</h3>

              <p className="text-muted-foreground">Interview Questions</p>

            </div>

            <div className="flex flex-col items-center justify-center space-y-2">

              <h3 className="text-4xl font-bold">95%</h3>

              <p className="text-muted-foreground">Success Rate</p>

            </div>

            <div className="flex flex-col items-center justify-center space-y-2">

              <h3 className="text-4xl font-bold">24/7</h3>

              <p className="text-muted-foreground">AI Support</p>

            </div>

          </div>

        </div>

      </section>

      {/\* How It Works Section \*/}

      <section className="w-full py-12 md:py-24 bg-background">

        <div className="container mx-auto px-4 md:px-6">

          <div className="text-center max-w-3xl mx-auto mb-12">

            <h2 className="text-3xl font-bold mb-4">How It Works</h2>

            <p className="text-muted-foreground">

              Four simple steps to accelerate your career growth

            </p>

          </div>

          <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-8 max-w-6xl mx-auto">

            {howItWorks.map((item, index) => (

              <div

                key={index}

                className="flex flex-col items-center text-center space-y-4"

              >

                <div className="w-16 h-16 rounded-full bg-primary/10 flex items-center justify-center">

                  {item.icon}

                </div>

                <h3 className="font-semibold text-xl">{item.title}</h3>

                <p className="text-muted-foreground">{item.description}</p>

              </div>

            ))}

          </div>

        </div>

      </section>

      <section className="w-full py-12 md:py-24 bg-muted/50">

        <div className="container mx-auto px-4 md:px-6">

          <h2 className="text-3xl font-bold text-center mb-12">

            What Our Users Say

          </h2>

          <div className="grid grid-cols-1 md:grid-cols-3 gap-8 max-w-6xl mx-auto">

            {testimonial.map((testimonial, index) => (

              <Card key={index} className="bg-background">

                <CardContent className="pt-6">

                  <div className="flex flex-col space-y-4">

                    <div className="flex items-center space-x-4 mb-4">

                      <div className="relative h-12 w-12 flex-shrink-0">

                        <Image

                          width={40}

                          height={40}

                          src={testimonial.image}

                          alt={testimonial.author}

                          className="rounded-full object-cover border-2 border-primary/20"

                        />

                      </div>

                      <div>

                        <p className="font-semibold">{testimonial.author}</p>

                        <p className="text-sm text-muted-foreground">

                          {testimonial.role}

                        </p>

                        <p className="text-sm text-primary">

                          {testimonial.company}

                        </p>

                      </div>

                    </div>

                    <blockquote>

                      <p className="text-muted-foreground italic relative">

                        <span className="text-3xl text-primary absolute -top-4 -left-2">

                          &quot;

                        </span>

                        {testimonial.quote}

                        <span className="text-3xl text-primary absolute -bottom-4">

                          &quot;

                        </span>

                      </p>

                    </blockquote>

                  </div>

                </CardContent>

              </Card>

            ))}

          </div>

        </div>

      </section>

      {/\* FAQ Section \*/}

      <section className="w-full py-12 md:py-24">

        <div className="container mx-auto px-4 md:px-6">

          <div className="text-center max-w-3xl mx-auto mb-12">

            <h2 className="text-3xl font-bold mb-4">

              Frequently Asked Questions

            </h2>

            <p className="text-muted-foreground">

              Find answers to common questions about our platform

            </p>

          </div>

          <div className="max-w-3xl mx-auto">

            <Accordion type="single" collapsible className="w-full">

              {faqs.map((faq, index) => (

                <AccordionItem key={index} value={`item-${index}`}>

                  <AccordionTrigger className="text-left">

                    {faq.question}

                  </AccordionTrigger>

                  <AccordionContent>{faq.answer}</AccordionContent>

                </AccordionItem>

              ))}

            </Accordion>

          </div>

        </div>

      </section>

      {/\* CTA Section \*/}

      <section className="w-full">

        <div className="mx-auto py-24 gradient rounded-lg">

          <div className="flex flex-col items-center justify-center space-y-4 text-center max-w-3xl mx- auto">

            <h2 className="text-3xl font-bold tracking-tighter text-primary-foreground sm:text-4xl md:text- 5xl">

              Ready to Accelerate Your Career?

            </h2>

            <p className="mx-auto max-w-[600px] text-primary-foreground/80 md:text-xl">

              Join thousands of professionals who are advancing their careers

              with AI-powered guidance.

            </p>

            <Link href="/dashboard" passHref>

              <Button

                size="lg"

                variant="secondary"

                className="h-11 mt-5 animate-bounce"

              >

                Start Your Journey Today <ArrowRight className="ml-2 h-4 w-4" />

              </Button>

            </Link>

          </div>

        </div>

      </section>

    </>

  );

}

**App/layout.js:**

import { Geist, Geist\_Mono } from "next/font/google";

import "./globals.css";

import { ThemeProvider } from "@/components/theme-provider";

import Header from "@/components/header";

import { ClerkProvider } from "@clerk/nextjs";

import { Toaster } from "@/components/ui/sonner";

const geistSans = Geist({

  variable: "--font-geist-sans",

  subsets: ["latin"],

});

const geistMono = Geist\_Mono({

  variable: "--font-geist-mono",

  subsets: ["latin"],

});

export const metadata = {

  title: "AI Career Coach",

  description: "Generated by create next app",

};

export default function RootLayout({ children }) {

  return (

    <ClerkProvider>

    <html lang="en" suppressHydrationWarning>

      <body

        className={`${geistSans.variable} ${geistMono.variable} antialiased`}

      >

        <ThemeProvider

            attribute="class"

            defaultTheme="dark"

            enableSystem

            disableTransitionOnChange

          >

            {/\*\* header \*/}

            <Header/>

            <main className="min-h-screen">

            {children}

            </main>

            <Toaster richColors />

            <footer className="bg-muted/50 py-12" >

              <div className="container mx-auto px-4 text-center text-gray-200">

                <p>

                Made By Anuj Prajapati

                </p>

              </div>

            </footer>

            {/\*\* footer \*/}

          </ThemeProvider>

      </body>

    </html>

    </ClerkProvider>

  );

}

**dashboard-view.jsx :**

"use client";

import React from "react";

import {

  BarChart,

  Bar,

  XAxis,

  YAxis,

  CartesianGrid,

  Tooltip,

  ResponsiveContainer,

} from "recharts";

import {

  BriefcaseIcon,

  LineChart,

  TrendingUp,

  TrendingDown,

  Brain,

} from "lucide-react";

import { format, formatDistanceToNow } from "date-fns";

import {

  Card,

  CardContent,

  CardDescription,

  CardHeader,

  CardTitle,

} from "@/components/ui/card";

import { Badge } from "@/components/ui/badge";

import { Progress } from "@/components/ui/progress";

const DashboardView = ({ insights }) => {

  // Transform salary data for the chart

  const salaryData = insights.salaryRanges.map((range) => ({

    name: range.role,

    min: range.min / 1000,

    max: range.max / 1000,

    median: range.median / 1000,

  }));

  const getDemandLevelColor = (level) => {

    switch (level.toLowerCase()) {

      case "high":

        return "bg-green-500";

      case "medium":

        return "bg-yellow-500";

      case "low":

        return "bg-red-500";

      default:

        return "bg-gray-500";

    }

  };

  const getMarketOutlookInfo = (outlook) => {

    switch (outlook.toLowerCase()) {

      case "positive":

        return { icon: TrendingUp, color: "text-green-500" };

      case "neutral":

        return { icon: LineChart, color: "text-yellow-500" };

      case "negative":

        return { icon: TrendingDown, color: "text-red-500" };

      default:

        return { icon: LineChart, color: "text-gray-500" };

    }

  };

  const OutlookIcon = getMarketOutlookInfo(insights.marketOutlook).icon;

  const outlookColor = getMarketOutlookInfo(insights.marketOutlook).color;

  // Format dates using date-fns

  const lastUpdatedDate = format(new Date(insights.lastUpdated), "dd/MM/yyyy");

  const nextUpdateDistance = formatDistanceToNow(

    new Date(insights.nextUpdate),

    { addSuffix: true }

  );

  return (

    <div className="space-y-6">

      <div className="flex justify-between items-center">

        <Badge variant="outline">Last updated: {lastUpdatedDate}</Badge>

      </div>

      {/\* Market Overview Cards \*/}

      <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4">

        <Card>

          <CardHeader className="flex flex-row items-center justify-between space-y-0 pb-2">

            <CardTitle className="text-sm font-medium">

              Market Outlook

            </CardTitle>

            <OutlookIcon className={`h-4 w-4 ${outlookColor}`} />

          </CardHeader>

          <CardContent>

            <div className="text-2xl font-bold">{insights.marketOutlook}</div>

            <p className="text-xs text-muted-foreground">

              Next update {nextUpdateDistance}

            </p>

          </CardContent>

        </Card>

        <Card>

          <CardHeader className="flex flex-row items-center justify-between space-y-0 pb-2">

            <CardTitle className="text-sm font-medium">

              Industry Growth

            </CardTitle>

            <TrendingUp className="h-4 w-4 text-muted-foreground" />

          </CardHeader>

          <CardContent>

            <div className="text-2xl font-bold">

              {insights.growthRate.toFixed(1)}%

            </div>

            <Progress value={insights.growthRate} className="mt-2" />

          </CardContent>

        </Card>

        <Card>

          <CardHeader className="flex flex-row items-center justify-between space-y-0 pb-2">

            <CardTitle className="text-sm font-medium">Demand Level</CardTitle>

            <BriefcaseIcon className="h-4 w-4 text-muted-foreground" />

          </CardHeader>

          <CardContent>

            <div className="text-2xl font-bold">{insights.demandLevel}</div>

            <div

              className={`h-2 w-full rounded-full mt-2 ${getDemandLevelColor(

                insights.demandLevel

              )}`}

            />

          </CardContent>

        </Card>

        <Card>

          <CardHeader className="flex flex-row items-center justify-between space-y-0 pb-2">

            <CardTitle className="text-sm font-medium">Top Skills</CardTitle>

            <Brain className="h-4 w-4 text-muted-foreground" />

          </CardHeader>

          <CardContent>

            <div className="flex flex-wrap gap-1">

              {insights.topSkills.map((skill) => (

                <Badge key={skill} variant="secondary">

                  {skill}

                </Badge>

              ))}

            </div>

          </CardContent>

        </Card>

      </div>

      {/\* Salary Ranges Chart \*/}

      <Card className="col-span-4">

        <CardHeader>

          <CardTitle>Salary Ranges by Role</CardTitle>

          <CardDescription>

            Displaying minimum, median, and maximum salaries (in thousands)

          </CardDescription>

        </CardHeader>

        <CardContent>

          <div className="h-[400px]">

            <ResponsiveContainer width="100%" height="100%">

              <BarChart data={salaryData}>

                <CartesianGrid strokeDasharray="3 3" />

                <XAxis dataKey="name" />

                <YAxis />

                <Tooltip

                  content={({ active, payload, label }) => {

                    if (active && payload && payload.length) {

                      return (

                        <div className="bg-background border rounded-lg p-2 shadow-md">

                          <p className="font-medium">{label}</p>

                          {payload.map((item) => (

                            <p key={item.name} className="text-sm">

                              {item.name}: ₹{item.value}K

                            </p>

                          ))}

                        </div>

                      );

                    }

                    return null;

                  }}

                />

                <Bar dataKey="min" fill="#94a3b8" name="Min Salary (K)" />

                <Bar dataKey="median" fill="#64748b" name="Median Salary (K)" />

                <Bar dataKey="max" fill="#475569" name="Max Salary (K)" />

              </BarChart>

            </ResponsiveContainer>

          </div>

        </CardContent>

      </Card>

      {/\* Industry Trends \*/}

      <div className="grid grid-cols-1 md:grid-cols-2 gap-4">

        <Card>

          <CardHeader>

            <CardTitle>Key Industry Trends</CardTitle>

            <CardDescription>

              Current trends shaping the industry

            </CardDescription>

          </CardHeader>

          <CardContent>

            <ul className="space-y-4">

              {insights.keyTrends.map((trend, index) => (

                <li key={index} className="flex items-start space-x-2">

                  <div className="h-2 w-2 mt-2 rounded-full bg-primary" />

                  <span>{trend}</span>

                </li>

              ))}

            </ul>

          </CardContent>

        </Card>

        <Card>

          <CardHeader>

            <CardTitle>Recommended Skills</CardTitle>

            <CardDescription>Skills to consider developing</CardDescription>

          </CardHeader>

          <CardContent>

            <div className="flex flex-wrap gap-2">

              {insights.recommendedSkills.map((skill) => (

                <Badge key={skill} variant="outline">

                  {skill}

                </Badge>

              ))}

            </div>

          </CardContent>

        </Card>

      </div>

    </div>

  );

};

export default DashboardView;

**Function.js :**

import { db } from "@/lib/prisma";

import { inngest } from "./client";

import { GoogleGenerativeAI } from "@google/generative-ai";

const genAI = new GoogleGenerativeAI(process.env.GEMINI\_API\_KEY);

const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

export const generateIndustryInsights = inngest.createFunction(

  { name: "Generate Industry Insights" },

  { cron: "0 0 \* \* 0" }, // Run every Sunday at midnight

  async ({ event, step }) => {

    const industries = await step.run("Fetch industries", async () => {

      return await db.industryInsight.findMany({

        select: { industry: true },

      });

    });

    for (const { industry } of industries) {

      const prompt = `

         Analyze the current state of the ${industry} industry in India and provide insights in ONLY the following JSON format.  All salary ranges MUST be in Indian Rupees (INR). Do not include any USD values.

          {

            "salaryRanges": [

              { "role": "string", "min": number, "max": number, "median": number, "location": "string" }

            ],

            "growthRate": number,

            "demandLevel": "HIGH" | "MEDIUM" | "LOW",

            "topSkills": ["skill1", "skill2"],

            "marketOutlook": "POSITIVE" | "NEUTRAL" | "NEGATIVE",

            "keyTrends": ["trend1", "trend2"],

            "recommendedSkills": ["skill1", "skill2"]

          }

          IMPORTANT: Return ONLY the JSON. No additional text, notes, or markdown formatting.

          Include at least 5 common roles for salary ranges.

          Growth rate should be a percentage.

          Include at least 5 skills and trends.

        `;

      const res = await step.ai.wrap(

        "gemini",

        async (p) => {

          return await model.generateContent(p);

        },

        prompt

      );

      const text = res.response.candidates[0].content.parts[0].text || "";

      const cleanedText = text.replace(/```(?:json)?\n?/g, "").trim();

      const insights = JSON.parse(cleanedText);

      await step.run(`Update ${industry} insights`, async () => {

        await db.industryInsight.update({

          where: { industry },

          data: {

            ...insights,

            lastUpdated: new Date(),

            nextUpdate: new Date(Date.now() + 7 \* 24 \* 60 \* 60 \* 1000),

          },

        });

      });

    }

  }

);

**Inggest/client.js :**

import { Inngest } from "inngest";

export const inngest = new Inngest({

  id: "career-coach", // Unique app ID

  name: "Career Coach",

  credentials: {

    gemini: {

      apiKey: process.env.GEMINI\_API\_KEY,

    },

  },

});

**user.js :**

"use server";

import { db } from "@/lib/prisma";

import { auth } from "@clerk/nextjs/server";

import { revalidatePath } from "next/cache";

import { generateAIInsights } from "./dashboard";

export async function updateUser(data) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  try {

    // Start a transaction to handle both operations

    const result = await db.$transaction(

      async (tx) => {

        // First check if industry exists

        let industryInsight = await tx.industryInsight.findUnique({

          where: {

            industry: data.industry,

          },

        });

        // If industry doesn't exist, create it with default values

        if (!industryInsight) {

          const insights = await generateAIInsights(data.industry);

          industryInsight = await db.industryInsight.create({

            data: {

              industry: data.industry,

              ...insights,

              nextUpdate: new Date(Date.now() + 7 \* 24 \* 60 \* 60 \* 1000),

            },

          });

        }

        // Now update the user

        const updatedUser = await tx.user.update({

          where: {

            id: user.id,

          },

          data: {

            industry: data.industry,

            experience: data.experience,

            bio: data.bio,

            skills: data.skills,

          },

        });

        return { updatedUser, industryInsight };

      },

      {

        timeout: 10000, // default: 5000

      }

    );

    revalidatePath("/");

    return { success:true , ...result  };

  } catch (error) {

    console.error("Error updating user and industry:", error.message);

    throw new Error("Failed to update profile" + error.message);

  }

}

export async function getUserOnboardingStatus() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  try {

    const user = await db.user.findUnique({

      where: {

        clerkUserId: userId,

      },

      select: {

        industry: true,

      },

    });

    return {

      isOnboarded: !!user?.industry,

    };

  } catch (error) {

    console.error("Error checking onboarding status:", error);

    throw new Error("Failed to check onboarding status");

  }

}

**Dashboard.js**

"use server";

import { db } from "@/lib/prisma";

import { auth } from "@clerk/nextjs/server";

import { GoogleGenerativeAI } from "@google/generative-ai";

const genAI = new GoogleGenerativeAI(process.env.GEMINI\_API\_KEY);

const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

export const generateAIInsights = async (industry) => {

  const prompt = `

         Analyze the current state of the ${industry} industry in India and provide insights in ONLY the following JSON format.  All salary ranges MUST be in Indian Rupees (INR). Do not include any USD values.

         {

           "salaryRanges": [

             { "role": "string", "min": number, "max": number, "median": number, "location": "string" }

           ],

           "growthRate": number,

           "demandLevel": "HIGH",

           "topSkills": ["skill1", "skill2"],

           "marketOutlook": "POSITIVE",

           "keyTrends": ["trend1", "trend2"],

           "recommendedSkills": ["skill1", "skill2"]

         }

         IMPORTANT: Return ONLY the JSON. No additional text, notes, or markdown formatting.

         Include at least 5 common roles for salary ranges.

         Growth rate should be a percentage.

         Include at least 5 skills and trends.

        `;

  const result = await model.generateContent(prompt);

  const response = result.response;

  const text = response.text();

  const cleanedText = text.replace(/```(?:json)?\n?/g, "").trim();

  return JSON.parse(cleanedText);

};

export async function getIndustryInsights() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

    include: {

      industryInsight: true,

    },

  });

  if (!user) throw new Error("User not found");

  // If no insights exist, generate them

  if (!user.industryInsight) {

    const insights = await generateAIInsights(user.industry);

    const industryInsight = await db.industryInsight.create({

      data: {

        industry: user.industry,

        ...insights,

        nextUpdate: new Date(Date.now() + 7 \* 24 \* 60 \* 60 \* 1000),

      },

    });

    return industryInsight;

  }

  return user.industryInsight;

}

**Interview.js :**

"use server";

import { db } from "@/lib/prisma";

import { auth } from "@clerk/nextjs/server";

import { GoogleGenerativeAI } from "@google/generative-ai";

const genAI = new GoogleGenerativeAI(process.env.GEMINI\_API\_KEY);

const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

export async function generateQuiz() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

    select: {

      industry: true,

      skills: true,

    },

  });

  if (!user) throw new Error("User not found");

  const prompt = `

    Generate 10 technical interview questions for a ${

      user.industry

    } professional${

    user.skills?.length ? ` with expertise in ${user.skills.join(", ")}` : ""

  }.

    Each question should be multiple choice with 4 options.

    Return the response in this JSON format only, no additional text:

    {

      "questions": [

        {

          "question": "string",

          "options": ["string", "string", "string", "string"],

          "correctAnswer": "string",

          "explanation": "string"

        }

      ]

    }

  `;

  try {

    const result = await model.generateContent(prompt);

    const response = result.response;

    const text = response.text();

    const cleanedText = text.replace(/```(?:json)?\n?/g, "").trim();

    const quiz = JSON.parse(cleanedText);

    return quiz.questions;

  } catch (error) {

    console.error("Error generating quiz:", error);

    throw new Error("Failed to generate quiz questions");

  }

}

export async function saveQuizResult(questions, answers, score) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  const questionResults = questions.map((q, index) => ({

    question: q.question,

    answer: q.correctAnswer,

    userAnswer: answers[index],

    isCorrect: q.correctAnswer === answers[index],

    explanation: q.explanation,

  }));

  // Get wrong answers

  const wrongAnswers = questionResults.filter((q) => !q.isCorrect);

  // Only generate improvement tips if there are wrong answers

  let improvementTip = null;

  if (wrongAnswers.length > 0) {

    const wrongQuestionsText = wrongAnswers

      .map(

        (q) =>

          `Question: "${q.question}"\nCorrect Answer: "${q.answer}"\nUser Answer: "${q.userAnswer}"`

      )

      .join("\n\n");

    const improvementPrompt = `

      The user got the following ${user.industry} technical interview questions wrong:

      ${wrongQuestionsText}

      Based on these mistakes, provide a concise, specific improvement tip.

      Focus on the knowledge gaps revealed by these wrong answers.

      Keep the response under 2 sentences and make it encouraging.

      Don't explicitly mention the mistakes, instead focus on what to learn/practice.

    `;

    try {

      const tipResult = await model.generateContent(improvementPrompt);

      improvementTip = tipResult.response.text().trim();

      console.log(improvementTip);

    } catch (error) {

      console.error("Error generating improvement tip:", error);

      // Continue without improvement tip if generation fails

    }

  }

  try {

    const assessment = await db.assessment.create({

      data: {

        userId: user.id,

        quizScore: score,

        questions: questionResults,

        category: "Technical",

        improvementTip,

      },

    });

    return assessment;

  } catch (error) {

    console.error("Error saving quiz result:", error);

    throw new Error("Failed to save quiz result");

  }

}

export async function getAssessments() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  try {

    const assessments = await db.assessment.findMany({

      where: {

        userId: user.id,

      },

      orderBy: {

        createdAt: "asc",

      },

    });

    return assessments;

  } catch (error) {

    console.error("Error fetching assessments:", error);

    throw new Error("Failed to fetch assessments");

  }

}

**Resume.js :**

"use server";

import { db } from "@/lib/prisma";

import { auth } from "@clerk/nextjs/server";

import { GoogleGenerativeAI } from "@google/generative-ai";

import { revalidatePath } from "next/cache";

const genAI = new GoogleGenerativeAI(process.env.GEMINI\_API\_KEY);

const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

export async function saveResume(content) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  try {

    const resume = await db.resume.upsert({

      where: {

        userId: user.id,

      },

      update: {

        content,

      },

      create: {

        userId: user.id,

        content,

      },

    });

    revalidatePath("/resume");

    return resume;

  } catch (error) {

    console.error("Error saving resume:", error);

    throw new Error("Failed to save resume");

  }

}

export async function getResume() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  return await db.resume.findUnique({

    where: {

      userId: user.id,

    },

  });

}

export async function improveWithAI({ current, type }) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

    include: {

      industryInsight: true,

    },

  });

  if (!user) throw new Error("User not found");

  const prompt = `

    As an expert resume writer, improve the following ${type} description for a ${user.industry} professional.

    Make it more impactful, quantifiable, and aligned with industry standards.

    Current content: "${current}"

    Requirements:

    1. Use action verbs

    2. Include metrics and results where possible

    3. Highlight relevant technical skills

    4. Keep it concise but detailed

    5. Focus on achievements over responsibilities

    6. Use industry-specific keywords

    Format the response as a single paragraph without any additional text or explanations.

  `;

  try {

    const result = await model.generateContent(prompt);

    const response = result.response;

    const improvedContent = response.text().trim();

    return improvedContent;

  } catch (error) {

    console.error("Error improving content:", error);

    throw new Error("Failed to improve content");

  }

}

**Cover-letter.js :**

"use server";

import { db } from "@/lib/prisma";

import { auth } from "@clerk/nextjs/server";

import { GoogleGenerativeAI } from "@google/generative-ai";

const genAI = new GoogleGenerativeAI(process.env.GEMINI\_API\_KEY);

const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

export async function generateCoverLetter(data) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  const prompt = `

    Write a professional cover letter for a ${data.jobTitle} position at ${

    data.companyName

  }.

    About the candidate:

    - Industry: ${user.industry}

    - Years of Experience: ${user.experience}

    - Skills: ${user.skills?.join(", ")}

    - Professional Background: ${user.bio}

    Job Description:

    ${data.jobDescription}

    Requirements:

    1. Use a professional, enthusiastic tone

    2. Highlight relevant skills and experience

    3. Show understanding of the company's needs

    4. Keep it concise (max 400 words)

    5. Use proper business letter formatting in markdown

    6. Include specific examples of achievements

    7. Relate candidate's background to job requirements

    Format the letter in markdown.

  `;

  try {

    const result = await model.generateContent(prompt);

    const content = result.response.text().trim();

    const coverLetter = await db.coverLetter.create({

      data: {

        content,

        jobDescription: data.jobDescription,

        companyName: data.companyName,

        jobTitle: data.jobTitle,

        status: "completed",

        userId: user.id,

      },

    });

    return coverLetter;

  } catch (error) {

    console.error("Error generating cover letter:", error.message);

    throw new Error("Failed to generate cover letter");

  }

}

export async function getCoverLetters() {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  return await db.coverLetter.findMany({

    where: {

      userId: user.id,

    },

    orderBy: {

      createdAt: "desc",

    },

  });

}

export async function getCoverLetter(id) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  return await db.coverLetter.findUnique({

    where: {

      id,

      userId: user.id,

    },

  });

}

export async function deleteCoverLetter(id) {

  const { userId } = await auth();

  if (!userId) throw new Error("Unauthorized");

  const user = await db.user.findUnique({

    where: { clerkUserId: userId },

  });

  if (!user) throw new Error("User not found");

  return await db.coverLetter.delete({

    where: {

      id,

      userId: user.id,

    },

  }); }

**industries.js :**

export const industries = [

  {

    id: "tech",

    name: "Technology",

    subIndustries: [

      "Software Development",

      "IT Services",

      "Cybersecurity",

      "Cloud Computing",

      "Artificial Intelligence/Machine Learning",

      "Data Science & Analytics",

      "Internet & Web Services",

      "Robotics",

      "Quantum Computing",

      "Blockchain & Cryptocurrency",

      "IoT (Internet of Things)",

      "Virtual/Augmented Reality",

      "Semiconductor & Electronics",

    ],

  },

  {

    id: "finance",

    name: "Financial Services",

    subIndustries: [

      "Banking",

      "Investment Banking",

      "Insurance",

      "FinTech",

      "Wealth Management",

      "Asset Management",

      "Real Estate Investment",

      "Private Equity",

      "Venture Capital",

      "Cryptocurrency & Digital Assets",

      "Risk Management",

      "Payment Processing",

      "Credit Services",

    ],

  },

  {

    id: "healthcare",

    name: "Healthcare & Life Sciences",

    subIndustries: [

      "Healthcare Services",

      "Biotechnology",

      "Pharmaceuticals",

      "Medical Devices",

      "Healthcare IT",

      "Telemedicine",

      "Mental Health Services",

      "Genomics",

      "Clinical Research",

      "Healthcare Analytics",

      "Elder Care Services",

      "Veterinary Services",

      "Alternative Medicine",

    ],

  },

  {

    id: "manufacturing",

    name: "Manufacturing & Industrial",

    subIndustries: [

      "Automotive",

      "Aerospace & Defense",

      "Electronics Manufacturing",

      "Industrial Manufacturing",

      "Chemical Manufacturing",

      "Consumer Goods",

      "Food & Beverage Processing",

      "Textile Manufacturing",

      "Metal Fabrication",

      "3D Printing/Additive Manufacturing",

      "Machinery & Equipment",

      "Packaging",

      "Plastics & Rubber",

    ],

  },

  {

    id: "retail",

    name: "Retail & E-commerce",

    subIndustries: [

      "E-commerce Platforms",

      "Retail Technology",

      "Fashion & Apparel",

      "Consumer Electronics",

      "Grocery & Food Retail",

      "Luxury Goods",

      "Sports & Recreation",

      "Home & Garden",

      "Beauty & Personal Care",

      "Pet Products",

      "Specialty Retail",

      "Direct-to-Consumer (D2C)",

      "Department Stores",

    ],

  },

  {

    id: "media",

    name: "Media & Entertainment",

    subIndustries: [

      "Digital Media",

      "Gaming & Esports",

      "Streaming Services",

      "Social Media",

      "Digital Marketing",

      "Film & Television",

      "Music & Audio",

      "Publishing",

      "Advertising",

      "Sports Entertainment",

      "News & Journalism",

      "Animation",

      "Event Management",

    ],

  },

  {

    id: "education",

    name: "Education & Training",

    subIndustries: [

      "EdTech",

      "Higher Education",

      "Professional Training",

      "Online Learning",

      "K-12 Education",

      "Corporate Training",

      "Language Learning",

      "Special Education",

      "Early Childhood Education",

      "Career Development",

      "Educational Publishing",

      "Educational Consulting",

      "Vocational Training",

    ],

  },

  {

    id: "energy",

    name: "Energy & Utilities",

    subIndustries: [

      "Renewable Energy",

      "Clean Technology",

      "Oil & Gas",

      "Nuclear Energy",

      "Energy Management",

      "Utilities",

      "Smart Grid Technology",

      "Energy Storage",

      "Carbon Management",

      "Waste Management",

      "Water & Wastewater",

      "Mining",

      "Environmental Services",

    ],

  },

  {

    id: "consulting",

    name: "Professional Services",

    subIndustries: [

      "Management Consulting",

      "IT Consulting",

      "Strategy Consulting",

      "Digital Transformation",

      "Business Advisory",

      "Legal Services",

      "Accounting & Tax",

      "Human Resources",

      "Marketing Services",

      "Architecture",

      "Engineering Services",

      "Research & Development",

      "Business Process Outsourcing (BPO)",

    ],

  },

  {

    id: "telecom",

    name: "Telecommunications",

    subIndustries: [

      "Wireless Communications",

      "Network Infrastructure",

      "Telecom Services",

      "5G Technology",

      "Internet Service Providers",

      "Satellite Communications",

      "Data Centers",

      "Fiber Optics",

      "Mobile Technology",

      "VoIP Services",

      "Network Security",

      "Telecom Equipment",

      "Cloud Communications",

    ],

  },

  {

    id: "transportation",

    name: "Transportation & Logistics",

    subIndustries: [

      "Electric Vehicles",

      "Autonomous Vehicles",

      "Logistics & Supply Chain",

      "Aviation",

      "Railways",

      "Maritime Transport",

      "Urban Mobility",

      "Fleet Management",

      "Last-Mile Delivery",

      "Warehousing",

      "Freight & Cargo",

      "Public Transportation",

      "Space Transportation",

    ],

  },

  {

    id: "agriculture",

    name: "Agriculture & Food",

    subIndustries: [

      "AgTech",

      "Farming",

      "Food Production",

      "Sustainable Agriculture",

      "Precision Agriculture",

      "Aquaculture",

      "Vertical Farming",

      "Agricultural Biotechnology",

      "Food Processing",

      "Organic Farming",

      "Plant-Based Foods",

      "Agricultural Equipment",

      "Indoor Farming",

    ],

  },

  {

    id: "construction",

    name: "Construction & Real Estate",

    subIndustries: [

      "Commercial Construction",

      "Residential Construction",

      "Real Estate Development",

      "Property Management",

      "Construction Technology",

      "Building Materials",

      "Infrastructure Development",

      "Smart Buildings",

      "Interior Design",

      "Facilities Management",

      "Real Estate Technology",

      "Sustainable Building",

      "Urban Planning",

    ],

  },

  {

    id: "hospitality",

    name: "Hospitality & Tourism",

    subIndustries: [

      "Hotels & Resorts",

      "Restaurants & Food Service",

      "Travel Technology",

      "Tourism",

      "Event Planning",

      "Vacation Rentals",

      "Cruise Lines",

      "Catering",

      "Theme Parks",

      "Travel Agencies",

      "Hospitality Management",

      "Online Travel Booking",

      "Cultural Tourism",

    ],

  },

  {

    id: "nonprofit",

    name: "Non-Profit & Social Services",

    subIndustries: [

      "Charitable Organizations",

      "Social Services",

      "Environmental Conservation",

      "Humanitarian Aid",

      "Education Non-Profits",

      "Healthcare Non-Profits",

      "Arts & Culture",

      "Community Development",

      "International Development",

      "Animal Welfare",

      "Youth Organizations",

      "Social Enterprise",

      "Advocacy Organizations",

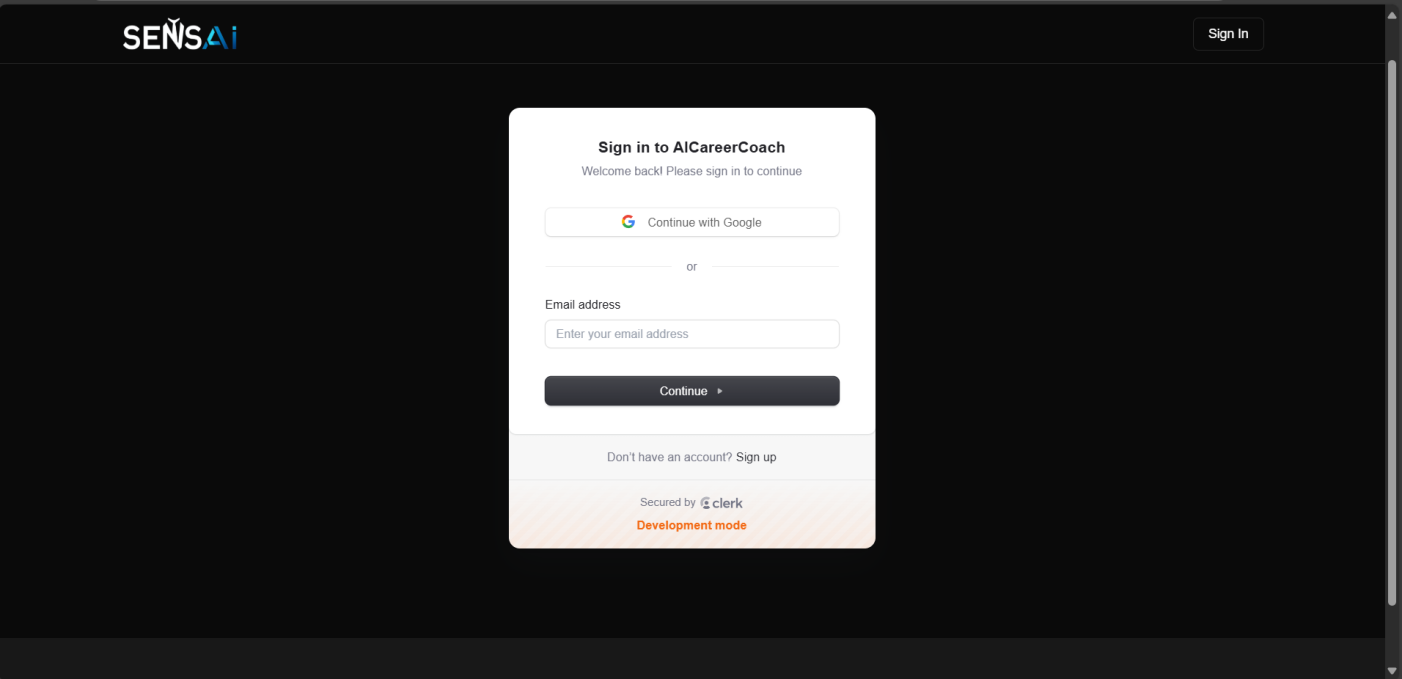
    ],

  },

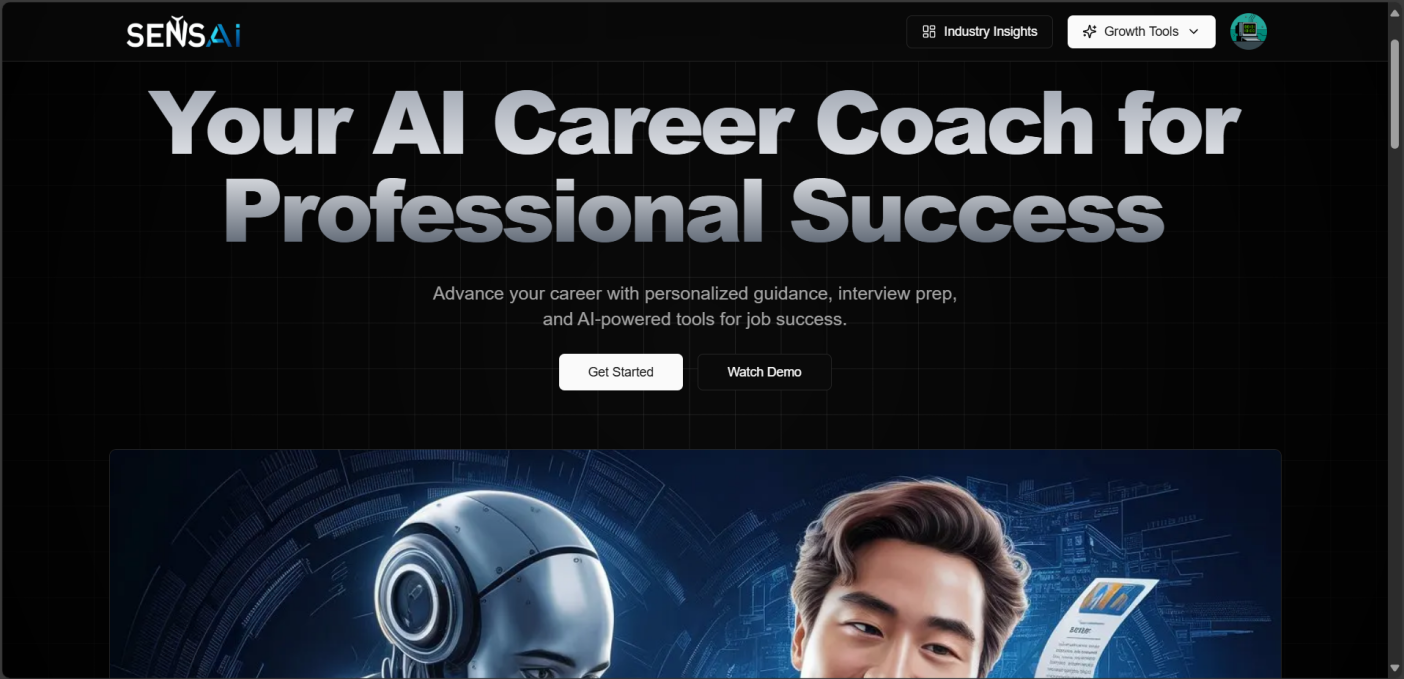
];

**Output:**

**Login Page Using Clerk**



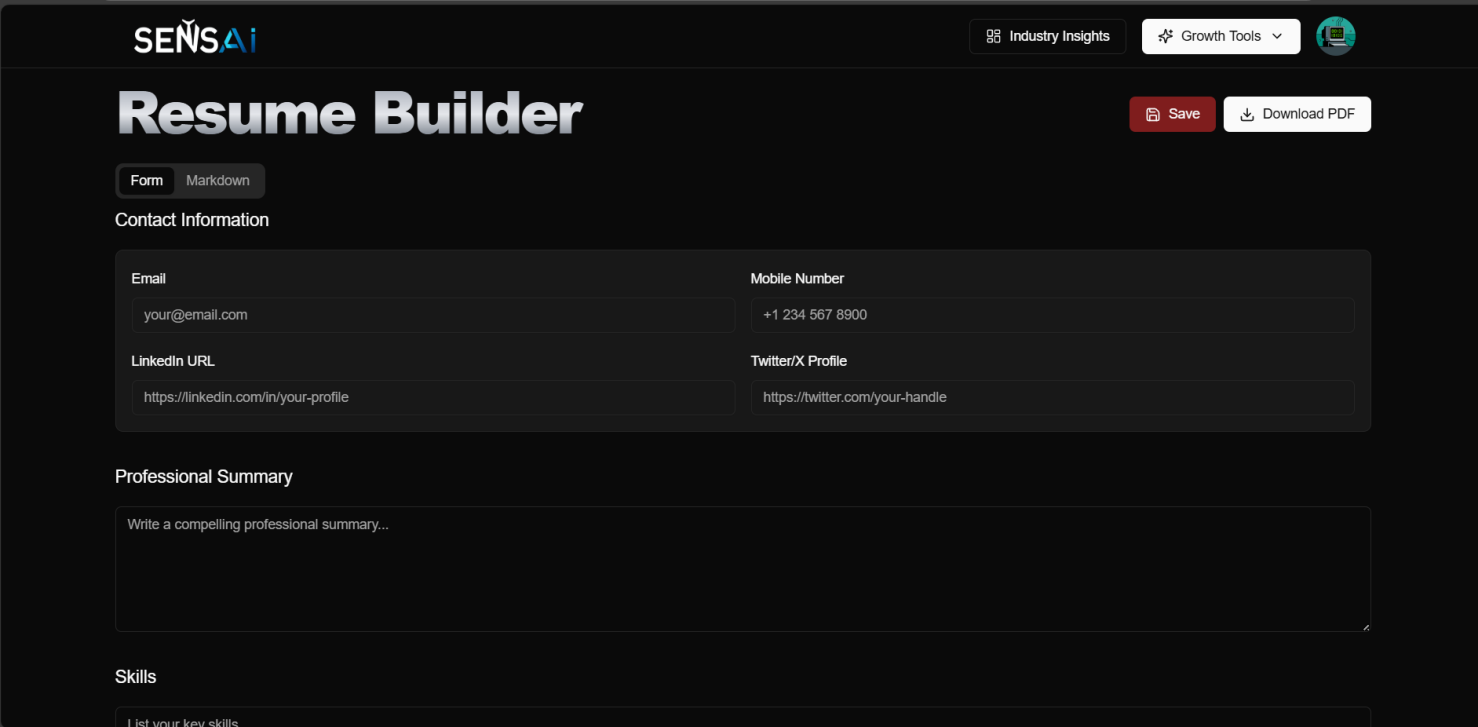
**Home Page**



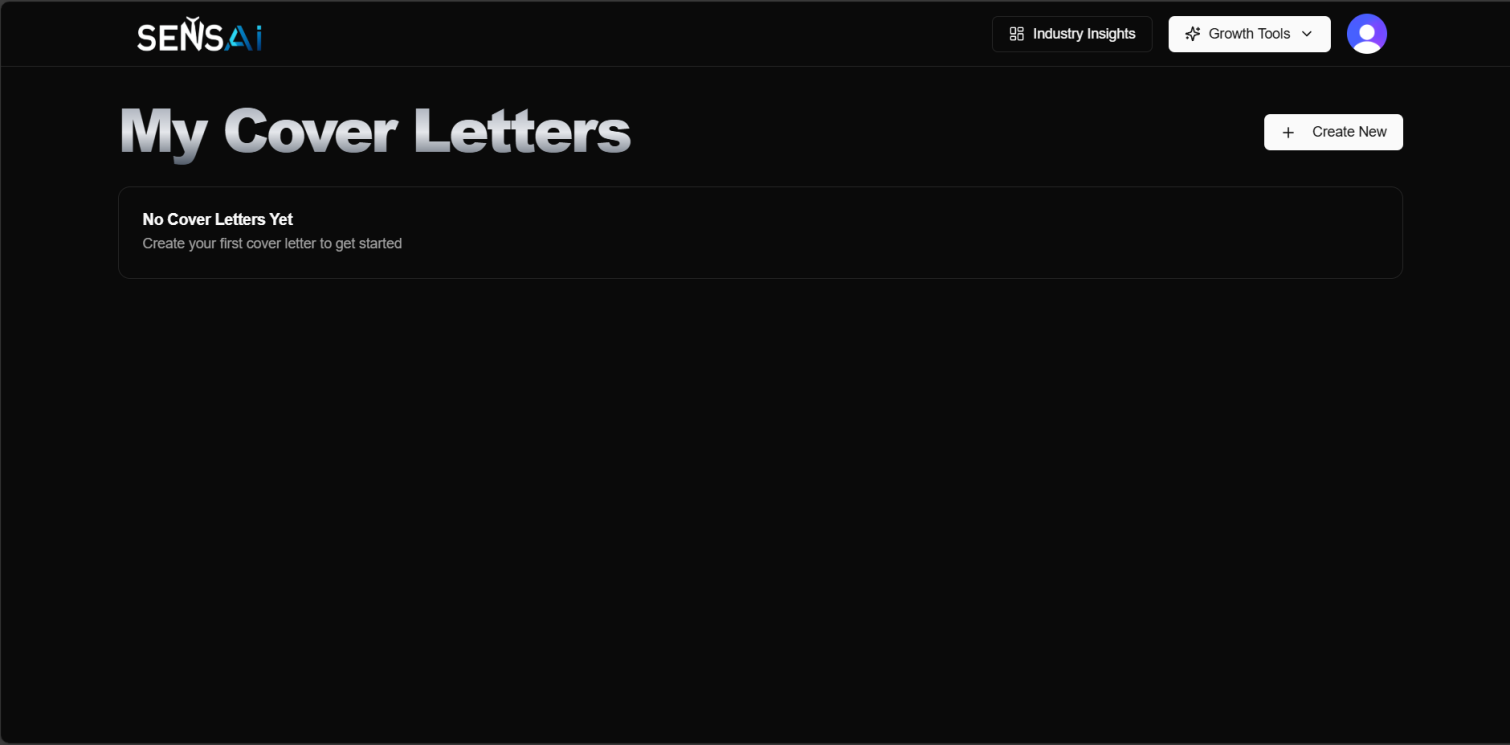
**Industry Insights Page**



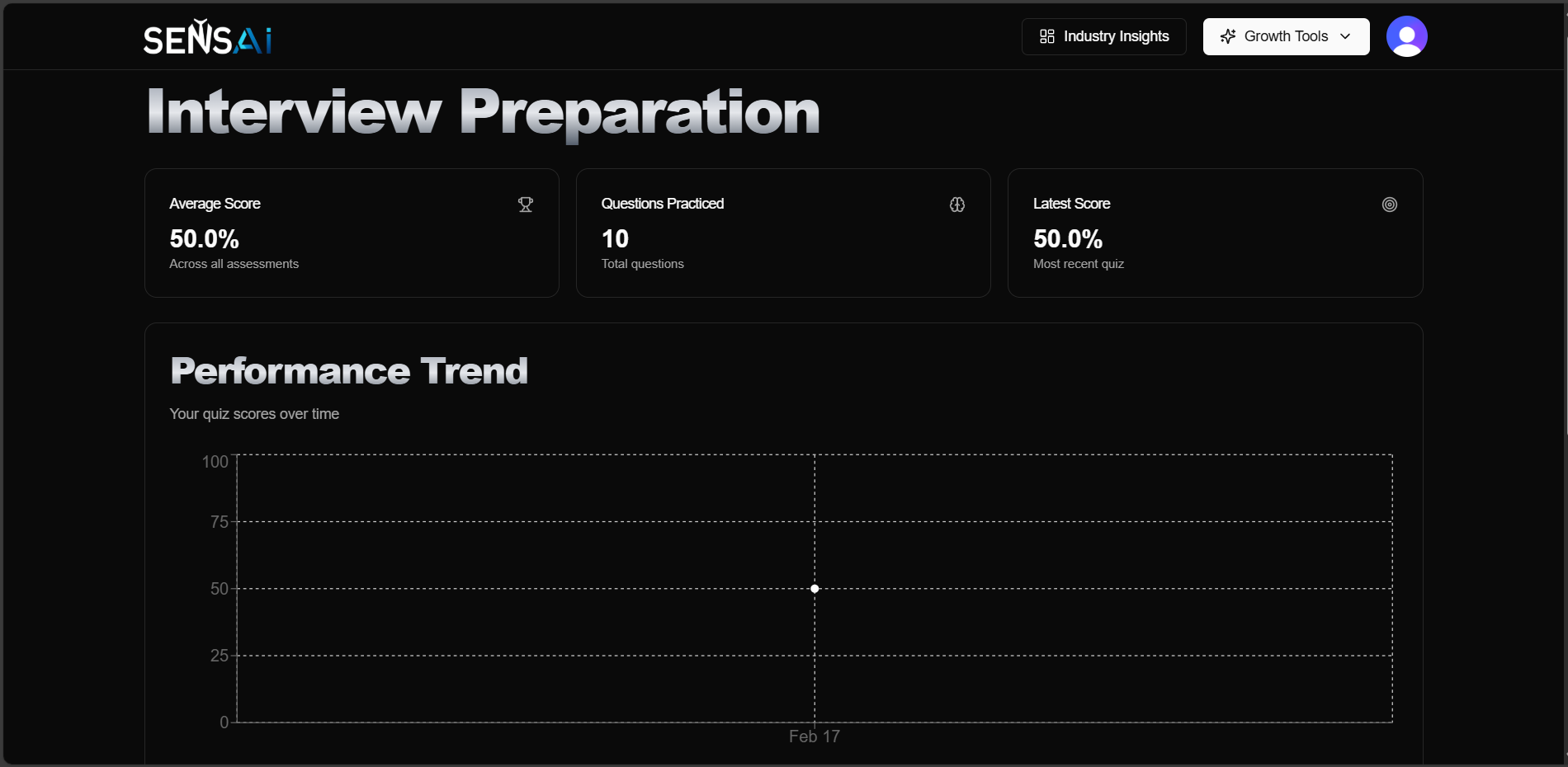
**Resume Builder Page**

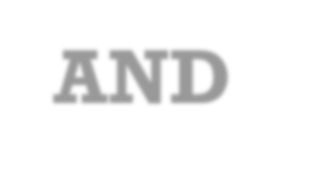
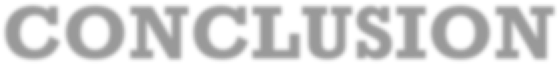


**Cover Letter Page**



**Interview Preparation Page**





1. **Conclusion and Future Scope:**

## Conclusion:

The AI Career Coach is an advanced full-stack application designed to help users navigate their career journey with AI-powered tools. This system provides **personalized industry insights, AI- generated resumes and cover letters, and tailored interview preparation** to enhance users' job search experience. By integrating technologies like **Next.js 15, React 19, Tailwind CSS, Clerk Authentication, Prisma ORM, NeonDB, and the Gemini API**, the platform ensures a seamless and efficient user experience.

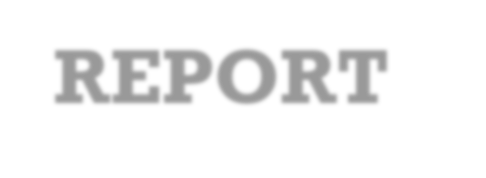
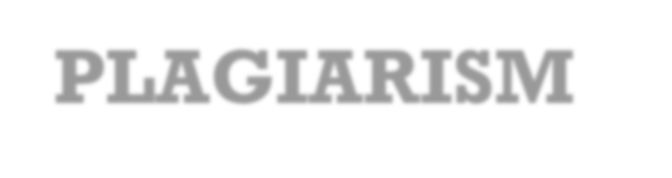
One of the key features of this project is its **intelligent career guidance system**, which analyzes user inputs and provides data-driven insights based on their selected industry. The **resume and cover letter generation** feature enables users to create professional job application documents effortlessly. Additionally, the **interview preparation module** enhances user confidence by generating industry-specific questions and AI-suggested responses, making the practice process more effective.

To further support career growth, the platform includes **performance tracking through quizzes and assessments**, allowing users to evaluate their progress and identify areas for improvement. The **secure data management using Prisma and NeonDB** ensures that user information is safely stored and easily accessible when needed.

This project demonstrates expertise in modern **full-stack web development, AI integration, authentication, and data management**, making it a valuable addition to a developer's portfolio. It highlights the ability to build **scalable, interactive, and intelligent applications** that provide real-world benefits. The AI Career Coach is not just a tool; it is a **comprehensive career assistant** that empowers users to take control of their professional journey with confidence.

## Future Scope :

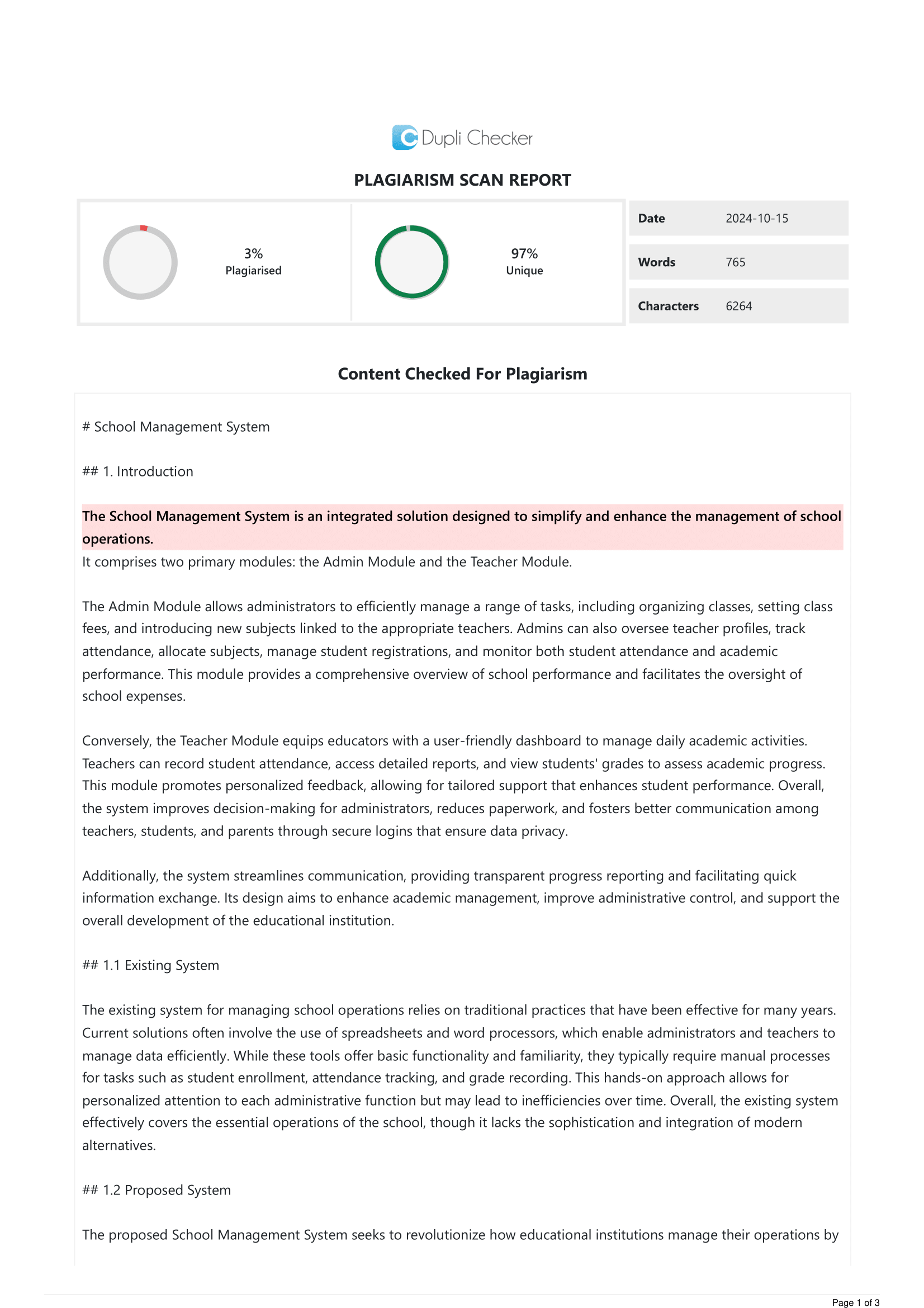
1. **Smarter AI Assistance** – Use better AI to give **more accurate career advice** and real-time job recommendations.
2. **Better Resume & Cover Letter Features** – Add **multiple resume templates**, **resume scoring**, and **LinkedIn integration** for better job applications.
3. **Mock Interviews** – Provide **AI-powered interview feedback**, including voice analysis to check confidence and clarity.
4. **Job Application Automation** – Link with **job portals like LinkedIn and Indeed** for **easy job applications** and real-time job alerts.
5. **Skill Improvement & Gamification** – Add **learning paths**, **courses**, and **reward systems** to keep users engaged.
6. **Multi-Language & Accessibility Support** – Make the platform **available in different languages** and improve accessibility for all users.
7. **Mentorship & Networking** – Connect users with **mentors, career coaches, and discussion forums** for better career guidance.
8. **Mobile & Cloud Expansion** – Create a **mobile app** and add **cloud storage** for resumes, cover letters, and interview notes.

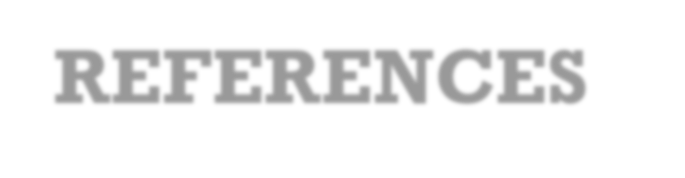


# PLAGIARISM

# REPORT

1. **Plagiarism Report:**

**<https://www.duplichecker.com/>**



# REFERENCES

1. **References:**

* <https://www.youtube.com/watch?v=UbXpRv5ApKA&list=PPSV&t=5054>
* Book :- **Fullstack React: The Complete Guide to ReactJS and Friends**
* <https://diagrammingai.com/>
* **<https://stackoverflow.com/>**

**Software Development and Technologies**

* Next.js: <https://nextjs.org/docs>
* PostgreSQL: <https://www.postgresql.org/docs/>
* Shadcn UI: <https://ui.shadcn.com/>
* Inngest : <https://app.inngest.com/env/production/functions>
* Tailwind css: <https://v2.tailwindcss.com/docs>
* Neon Database: <https://neon.tech/docs/connect/connect-intro>
* Google AI Studio : <https://aistudio.google.com/app/prompts/new_chat>
* Prisma : <https://www.prisma.io/docs>