Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

AI-Powered Next-Gen Job Recommendation System Using NLP and Machine Learning

Course Title: Integrated Design Project II
Course Code: CSE - 406
Section: 213 - D6

Students Details

Name	ID	Role
Nazmul Hasan	213902003	Leader
Irteja Mahmud	213902016	Member-1
Pankaj Mahanta	213902002	Member-2

Submission Date: 17 - 05 - 25
Course Teacher's Name: Babe Sultana

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

1 Introduction **3**

1.1 Introduction . . . 3

1.2 Problem Statement . . . 3

1.3 Motivation . . . 4

1.4 Aims and Objectives . . . 4

1.5 Project Description . . . 4

2 Requirement Analysis and Design **5**

2.1 Requirement Analysis . . . 5

2.1.1 Functional Requirements . . . 5

2.1.2 Non-Functional Requirements . . . 5

2.2 Tools and Techniques . . . 6

2.3 Gantt Chart . . . 6

2.4 SDLC Model Selection . . . 7

2.5 Use Case Diagram . . . 7

2.6 Data Flow Diagram . . . 8

2.6.1 Level-0 DFD . . . 8

2.6.2 Level-1 DFD . . . 8

2.7 System Architecture . . . 8

2.7.1 High-Level Design . . . 8

2.7.2 Low-Level Design . . . 9

2.8 E-R Diagram and Database Schema . . . 9

2.9 Conclusion . . . 10

3 Interface Design and Implementation **11**

3.1 3.1 User Interface Design . . . 11

3.1.1 Homepage . . . 11

3.1.2 User Registration and Login 12
3.1.3 Role-Based Dashboards 12
3.2 Job Seeker Features 13
3.3 Recruiter Features 15
3.4 Backend Development 16
3.4.1 Database Structure 16
3.5 Resume Upload and Profile Completion 17
3.6 Job Recommendation System 17
3.7 Admin Panel 18
3.8 Conclusion 18

4 Test Cases **19**

5 Conclusion and Future Work **26**
5.1 Summary of the Project 26
5.2 Social Impact 26
5.3 Future Work 27

Chapter 1

Introduction

1.1 Introduction

Looking for a job that matches your skills and goals can be time-consuming and frustrating especially for fresh graduates who are just entering the job market. Many job seekers end up applying to roles that don't really fit them, while recruiters often receive applications from unqualified candidates. There's a clear disconnect between the two sides.

This project, titled **NextWorkX**, is a web-based job recommendation system that tries to solve this problem. It uses machine learning and natural language processing to recommend jobs to users based on their profiles, resumes, and interests. The platform is built to serve both job seekers and recruiters, offering personalized suggestions, easy application tracking, and job management tools. It's designed to make the job hunt smarter and more targeted.

1.2 Problem Statement

Traditional job portals operate like bulletin boards: they list jobs, and it's up to the user to search, filter, and apply. This often leads to mismatched applications, wasted time, and frustration on both ends. Many platforms don't offer intelligent suggestions or take into account the user's skills, experience, and preferences in a meaningful way.

The main question this project addresses is: **How can machine learning and natural language processing be used to recommend jobs that are genuinely relevant to an individual user's profile?**

The system also aims to:

- Help job seekers find suitable positions without manually filtering through hundreds of listings.
- Allow recruiters to post and manage jobs while viewing matched applicants.
- Match users with jobs based on resume content, skill keywords, and profile data.

1.3 Motivation

The idea for this project came from observing how difficult it is for many students and recent graduates to find jobs that truly fit their background. We've seen classmates apply to jobs outside their field, not because they want to—but because they can't find better options. Job boards can be overwhelming, and most don't help users figure out what suits them best.

We were also inspired by how machine learning is being used in real-world applications, especially in personalization systems like Netflix or Spotify. We wanted to bring that same intelligence to the job market, creating a system that doesn't just list jobs—but understands people and recommends accordingly [1].

1.4 Aims and Objectives

This project has a few clear goals:

- Build a fully working job recommendation system for both freshers and professionals.
- Use machine learning and NLP techniques (TF-IDF, keyword matching) to match users with jobs.
- Create role-based dashboards for job seekers and recruiters.
- Allow recruiters to post jobs, manage applications, and set up company profiles.
- Enable job seekers to create multi-step profiles and upload resumes.
- Make the interface user-friendly, dynamic, and mobile-responsive.

1.5 Project Description

NextWorkX is a full-stack web application with separate interfaces for job seekers, recruiters, and admins. It includes an authentication system, profile management, resume uploads, job listings, and an intelligent job recommendation engine powered by Python-based ML models.

The platform combines frontend web technologies (HTML, CSS, JavaScript) with backend development in PHP and MySQL. Machine learning is implemented using Python and integrated with the backend to provide personalized job suggestions based on the user's data. The final output is a system that feels smart, relevant, and practical for real-world use.

Chapter 2

Requirement Analysis and Design

2.1 Requirement Analysis

2.1.1 Functional Requirements

The system is built to support three types of users: job seekers, recruiters, and admins. Each user has their own tasks and views.

- Job seekers can register, log in, create multi-step profiles, upload resumes, search for jobs, and apply.
- Recruiters can register, post jobs, set up company profiles, view applications, and manage job listings.
- Admins can manage users, jobs, and monitor platform activity.
- The system recommends jobs based on resume content, profile data, and keywords using machine learning.
- Real-time filtering and pagination are available on the job search page.

2.1.2 Non-Functional Requirements

These focus on how the system behaves rather than what it does.

- **Performance:** Pages should load within 2 seconds under normal traffic.
- **Usability:** Interface is simple, responsive, and mobile-friendly.
- **Scalability:** Backend is designed to handle growing numbers of users and job posts.
- **Security:** User passwords are hashed; SQL injection and file upload vulnerabilities are handled.
- **Maintainability:** PHP files are modular; code is structured for updates.

2.2 Tools and Techniques

The **NextWorkX** platform was developed using a combination of web technologies, backend scripting, and machine learning tools:

- **Frontend:** HTML5, CSS3, JavaScript, and Tailwind CSS were used to build a responsive and interactive user interface.
- **Backend:** PHP handled session management, server-side logic, and dynamic content.
- **Database:** MySQL was used to manage users, jobs, applications, and profile data. phpMyAdmin supported database management.
- **Machine Learning:** Python libraries like `Scikit-learn`, `spaCy`, and `Pandas` enabled NLP-based job matching using TF-IDF and keyword extraction.
- **Development Environment:** XAMPP provided a local server stack for testing with Apache and MySQL.
- **Version Control:** Git was used for tracking changes and collaborative development.
- **Debugging:** Chrome DevTools assisted in frontend inspection and debugging.

2.3 Gantt Chart

The Gantt chart shows the planned schedule of tasks including research, design, development, testing, and reporting.

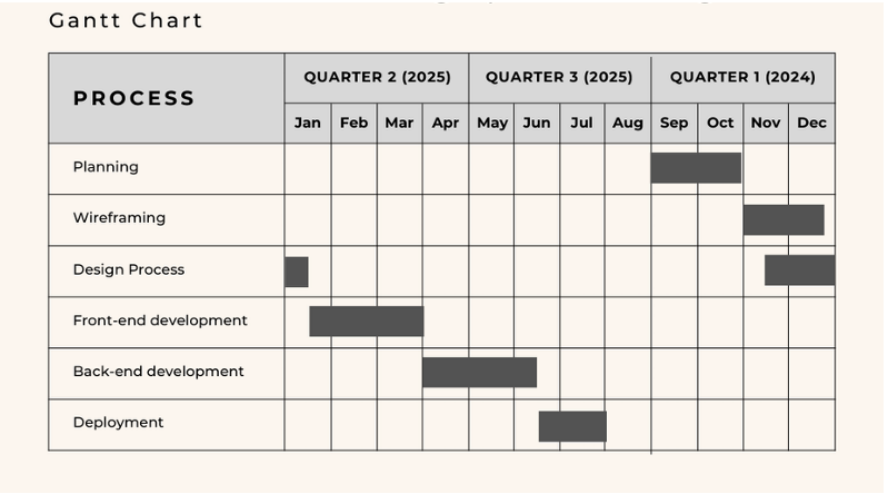

Figure 2.1: Gantt Chart of the Project

2.4 SDLC Model Selection

For this project, the **Agile Model** was used. It was a good fit because the system was developed in stages. Features were added and tested in cycles, which helped improve the design over time. This also allowed flexibility in handling changes or new ideas during development.

Table 2.1: Comparison Matrix of SDLC Models for NextWorkX.

Priority	Criteria	Waterfall	V-Shape	Iterative	Spiral	Agile	Prototype
10	User-Centric Design	No	No	Yes	Yes	**Yes**	Yes
10	User Feedback Integration	No	No	Yes	Yes	**Yes**	Yes
10	Flexibility for Changes	No	Yes	Yes	Yes	**Yes**	No
9	Risk Management	No	No	No	Yes	No	No
9	Adaptability to New Technologies	No	No	Yes	Yes	**Yes**	Yes
8	Integration with Machine Learning	No	No	Yes	Yes	**Yes**	No
8	Time to Market	No	No	Yes	No	**Yes**	Yes
8	Performance	No	Yes	Yes	Yes	**Yes**	No
8	Scalability	No	No	Yes	Yes	**Yes**	Yes
7	Security and Dependability	No	Yes	No	No	**Yes**	No
7	Complexity of Development	No	Yes	Yes	Yes	**Yes**	Yes
6	Time-Consuming	Yes	Yes	No	Yes	No	No
Total	**Overall Score**	20	57	95	101	**105**	82

2.5 Use Case Diagram

The use case diagram outlines key interactions between users and the system, such as login, job posting, applying, and profile setup.

Figure 2.2: Use Case Diagram

2.6 Data Flow Diagram

2.6.1 Level-0 DFD

This diagram shows the core data flow between users (job seeker, recruiter, admin) and the system as a whole.

Figure 2.3: Level-0 Data Flow Diagram

2.6.2 Level-1 DFD

The Level-1 diagram breaks down key processes like job application, recommendation, and profile handling.

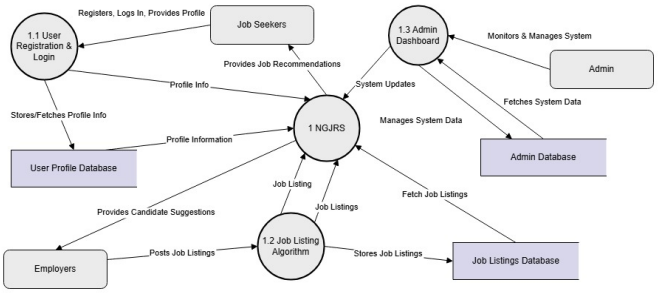

Figure 2.4: Level-1 Data Flow Diagram

2.7 System Architecture

2.7.1 High-Level Design

This design shows how the frontend, backend, database, and ML model work together.

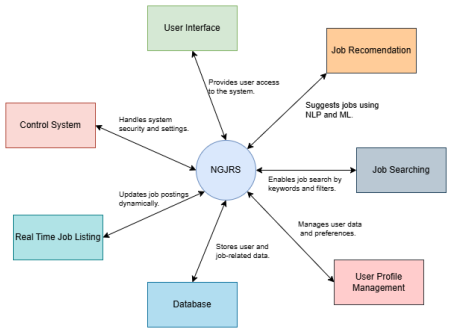

Figure 2.5: High-Level System Architecture

2.7.2 Low-Level Design

This includes how modules like authentication, job post, resume matching, and profile tracking are structured and connected.

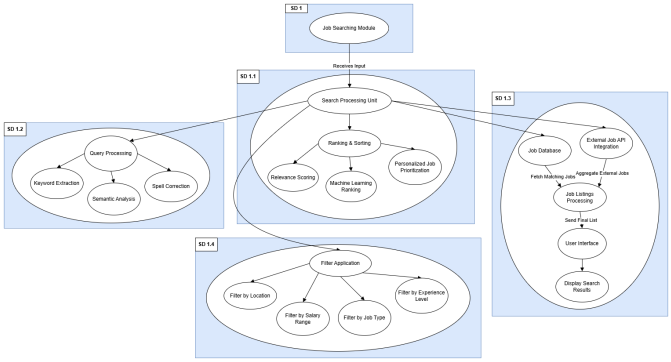

Figure 2.6: Low-Level Component Design

2.8 E-R Diagram and Database Schema

Entity-Relationship (E-R) Diagram

The Entity-Relationship (E-R) diagram illustrates the logical structure of the database. It shows the relationships between key entities such as users, jobs, applications, resumes, and company profiles. This model ensures data consistency and supports efficient querying across the platform.

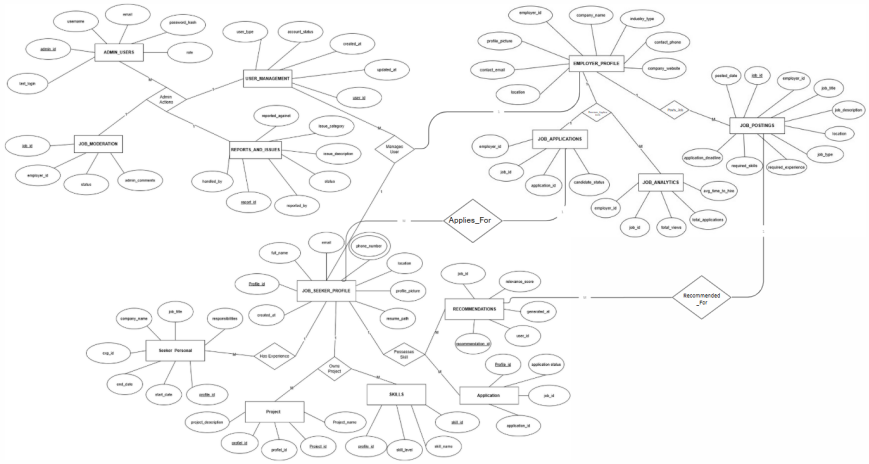

Figure 2.7: E-R Diagram

Database Schema

This includes table structures like users, job_seekers, recruiters, jobs, applications, resumes, and company_profiles.

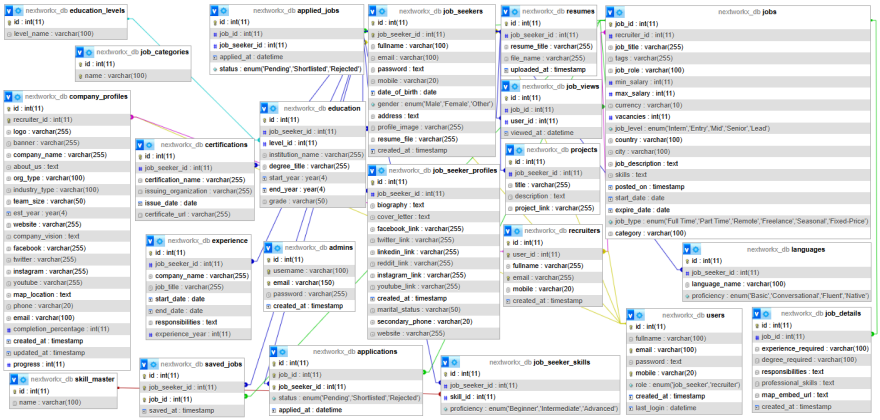

Figure 2.8: Database Schema Design

2.9 Conclusion

This chapter defined the foundational blueprint of the NextWorkX platform. It detailed the system requirements, selected tools, SDLC model, and architectural design. Diagrams such as the Use Case, Data Flow, and E-R models were presented to illustrate system flow and database relationships. These design elements collectively provide a structured framework that guided the development and ensured the system met its intended functionality.

Chapter 3

Interface Design and Implementation

This chapter explains how the user interfaces were designed and how each feature was implemented in the system. It includes screenshots and step-by-step descriptions for the main pages and functions.

3.1 3.1 User Interface Design

The system interface is simple and user-friendly. HTML, CSS, and Tailwind CSS were used for styling.

3.1.1 Homepage

The homepage introduces the platform, highlights top categories, featured jobs, and action buttons for users.

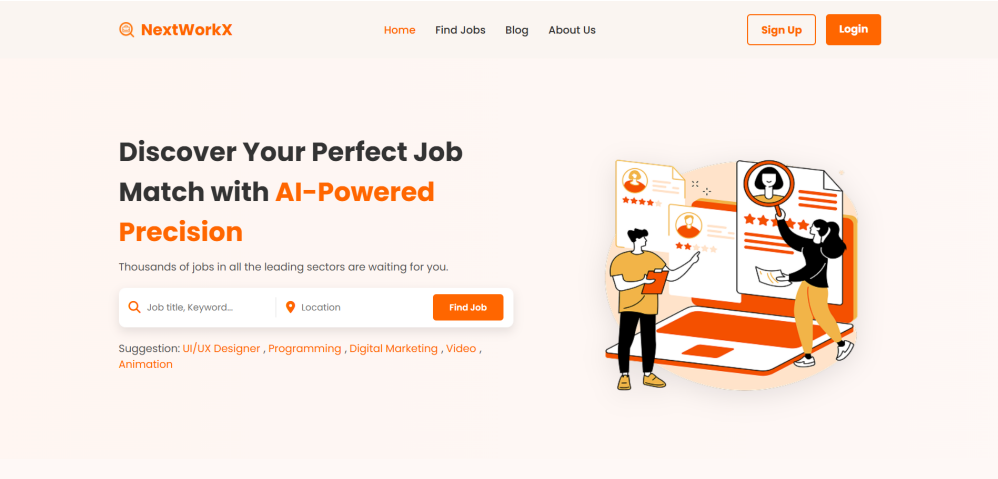

Figure 3.1: Homepage Interface

3.1.2 User Registration and Login

Users can register as either a job seeker or recruiter. Form validations are applied during both registration and login to ensure accurate data entry and secure access.

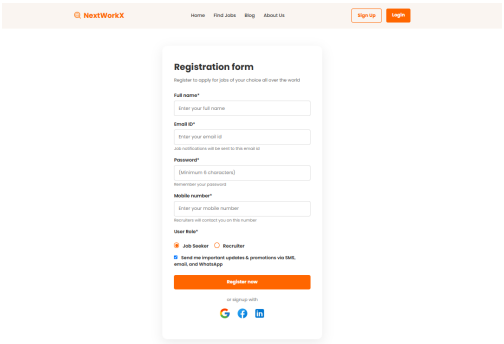

Figure 3.2: Registration Page

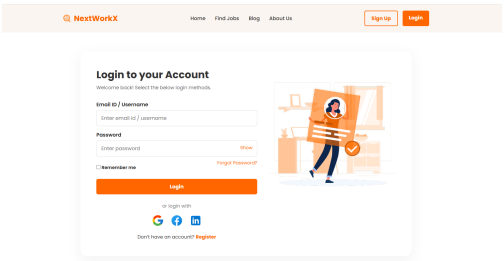

Figure 3.3: Login Page

3.1.3 Role-Based Dashboards

The NextWorkX platform provides dedicated dashboards for each user role, ensuring an intuitive and goal-oriented experience.

Job Seeker Dashboard: Job seekers can explore recommended jobs, apply, view application history, save listings, and complete a multi-step profile with education, experience, skills, and resume upload. A profile completion indicator encourages users to build a strong profile.

Recruiter Dashboard: Recruiters can post and manage jobs, view matched applicants, and update their company profile in steps. They also get insights into job status and applicant engagement.

This separation of interfaces improves usability, security, and workflow efficiency for both user types.

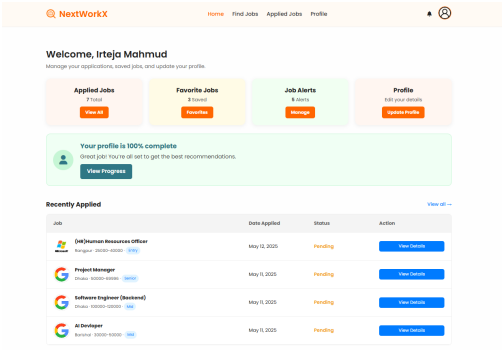

Figure 3.4: Job Seeker Dashboard

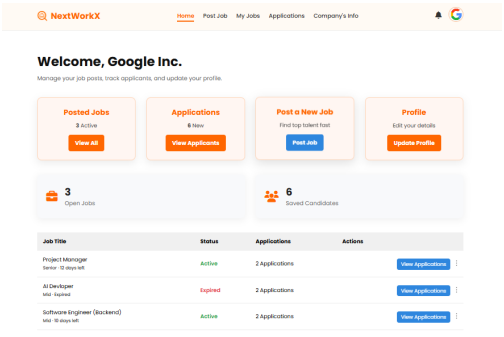

Figure 3.5: Recruiter Dashboard

3.2 Job Seeker Features

The job seeker interface provides users with essential tools to manage their career search effectively. The dashboard is designed to be intuitive and provides access to several core features:

Find Jobs

Job seekers can search for jobs using filters such as location, category, salary range, and keywords. The system also displays personalized recommendations.

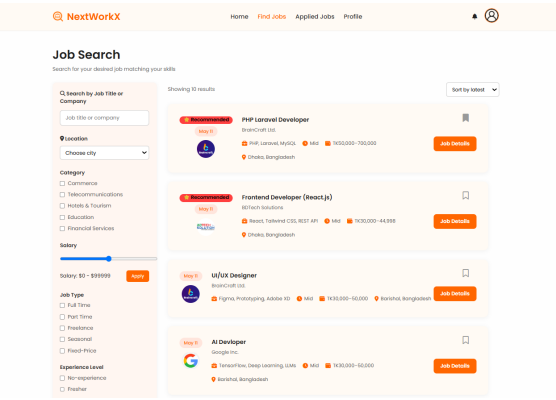

Figure 3.6: Job Seeker – Job Search Interface

Applied Jobs

Users can view a list of jobs they have applied for, along with the status of each application.

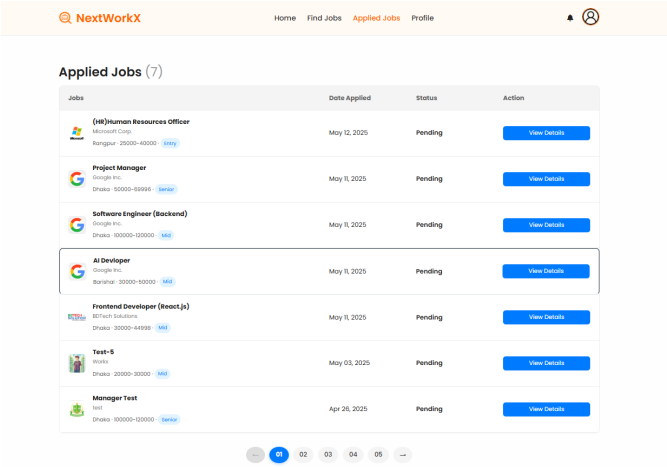

Figure 3.7: Job Seeker – Applied Jobs View

Profile

A multi-step form allows users to complete their profile, including personal information, education, experience, skills, and resume upload.

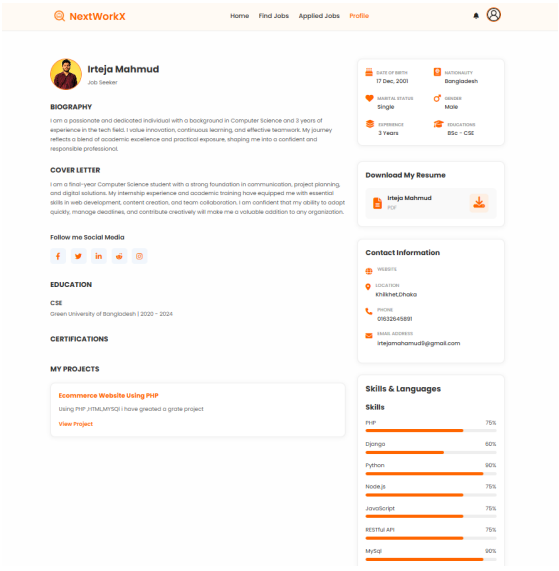

Figure 3.8: Job Seeker – Profile Completion Interface

Profile Settings

Job seekers can update their account details, change passwords, and manage notification settings.

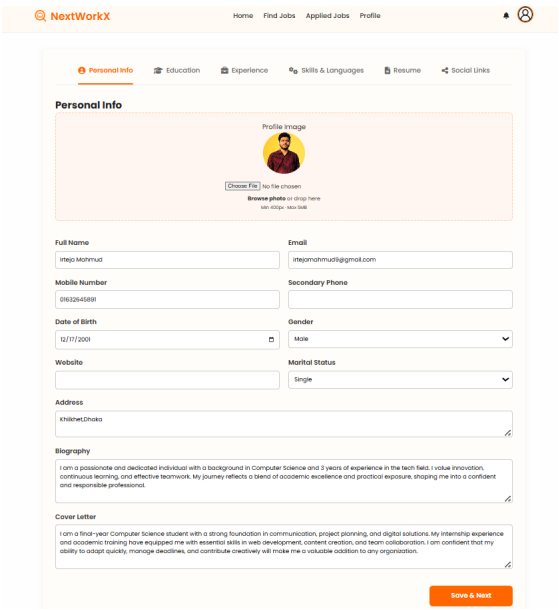

Figure 3.9: Job Seeker – Profile Settings Page

3.3 Recruiter Features

The recruiter interface is focused on simplifying the hiring process and managing job posts. It provides the following core features:

My Jobs

Recruiters can view, edit, and manage all their posted jobs, including their current status (active or expired).

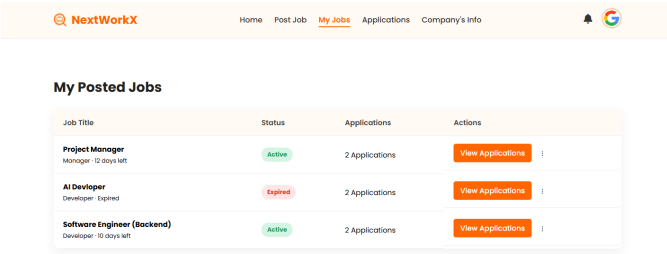

Figure 3.10: Recruiter – My Jobs Management

Applications

A detailed applications view allows recruiters to see which job seekers have applied to each position, along with their resumes and matching skills.

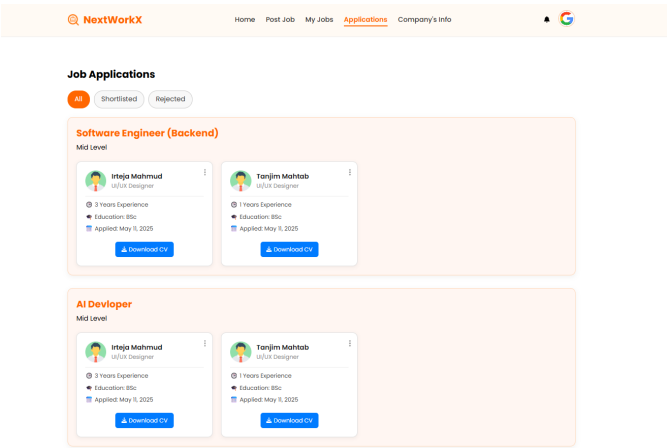

Figure 3.11: Recruiter – Applications Management Interface

Profile Completion

Recruiters can complete a multi-step company profile including company info, founding year, contact, and social media. A progress bar tracks completion level.

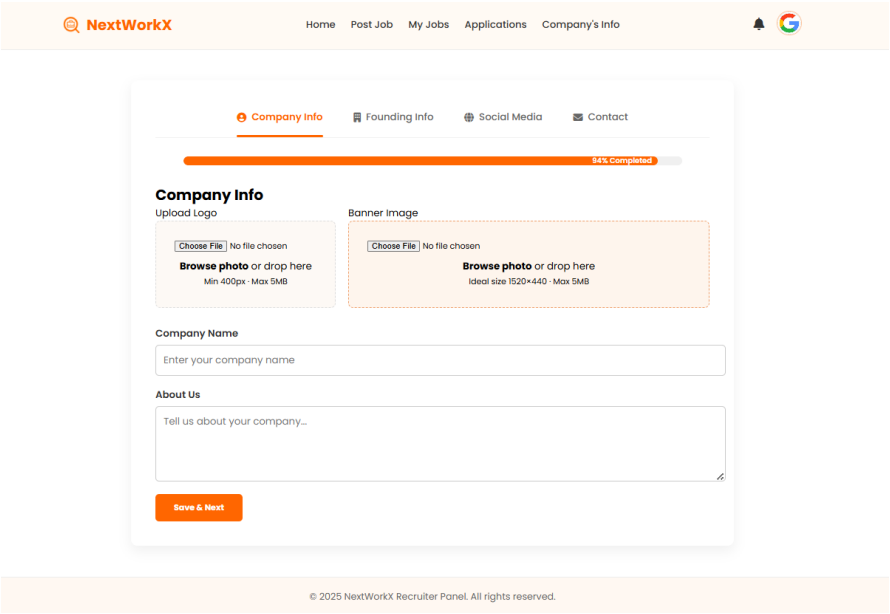

Figure 3.12: Recruiter – Profile Completion Interface

3.4 Backend Development

The backend was built with PHP and connected to MySQL. Each feature runs through specific PHP scripts with form validation, session handling, and database queries.

3.4.1 Database Structure

Several tables were created such as ‘users‘, ‘jobs‘, ‘applications‘, ‘resumes‘, and ‘bookmarks‘. These are linked with foreign keys and used to track job activity and user profiles.

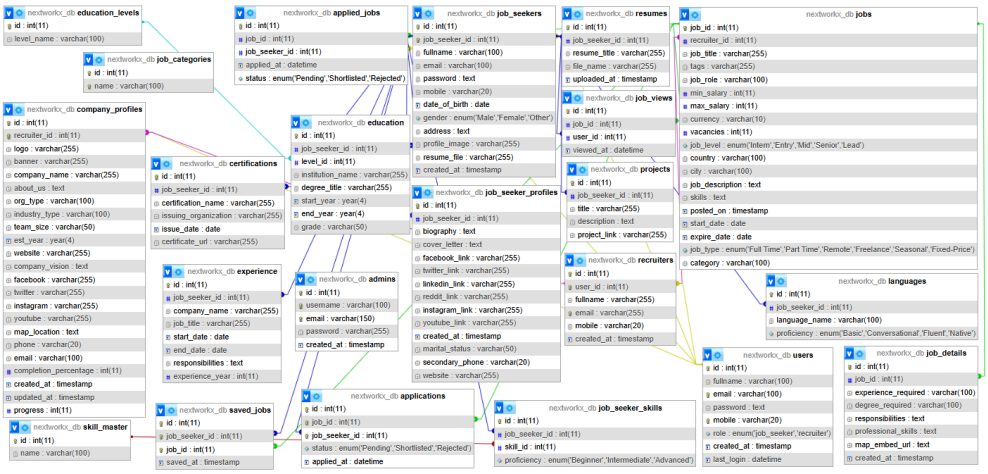

Figure 3.13: Database Schema

3.5 Resume Upload and Profile Completion

Job seekers can upload a resume and complete their profile in steps: Personal Info, Education, Experience, Skills, and Languages. A progress bar updates after each step.

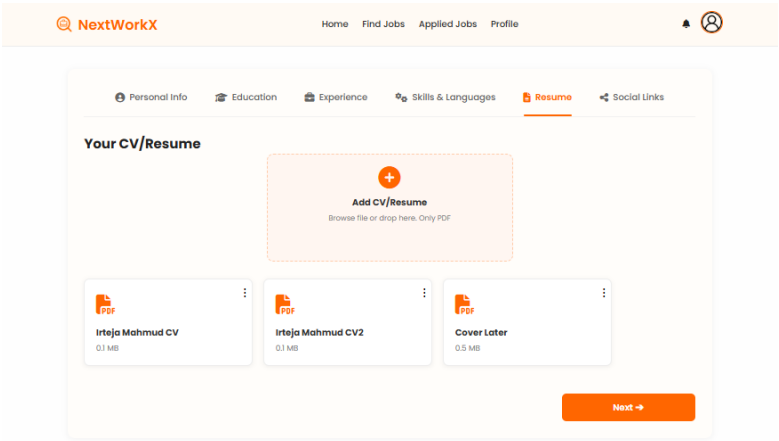

Figure 3.14: Profile Completion and Resume Upload

3.6 Job Recommendation System

A Python script runs the recommendation engine. It uses TF-IDF to compare resume keywords with job descriptions and returns a ranked list of jobs.

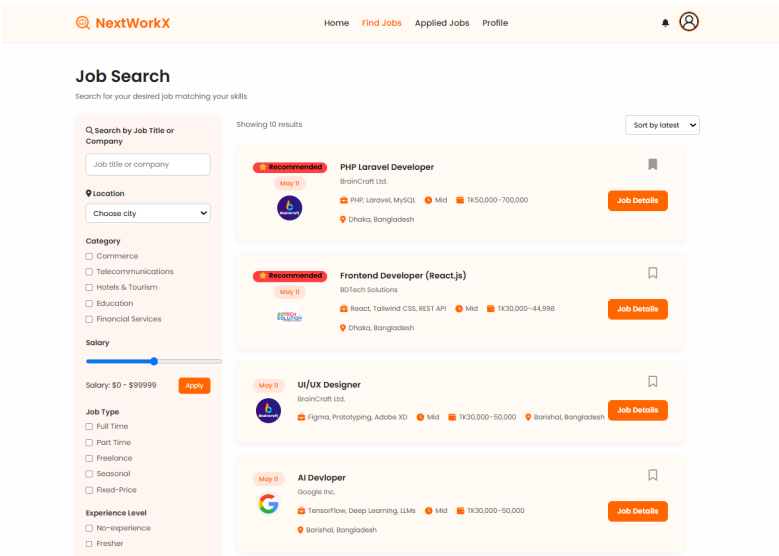

Figure 3.15: Job Recommendation Result

3.7 Admin Panel

Admins can log in and view all jobs, users, and applications. They can also delete records or manage platform content.

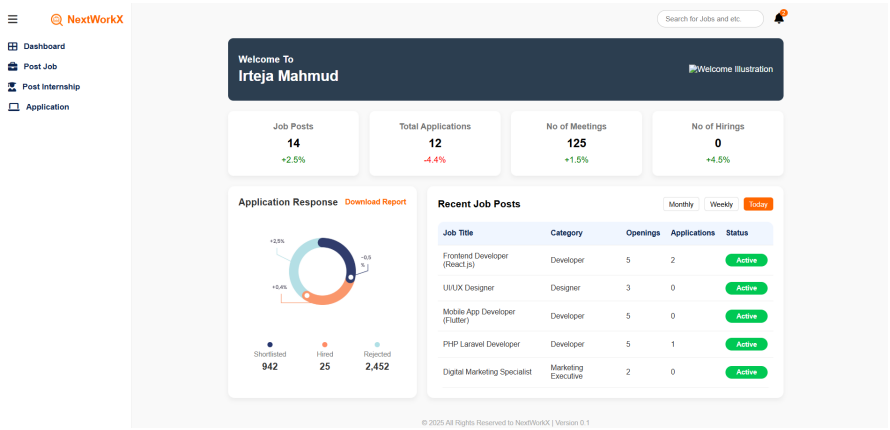

Figure 3.16: Admin Panel Interface

3.8 Conclusion

This chapter walked through the development process and interface structure of the system. With the front-end design, backend logic, and machine learning model working together, the project is now fully functional and ready for testing or future improvements.

Each feature was carefully implemented to align with user needs, ensuring role-based access, ease of use, and seamless navigation. The integration between the job recommendation engine and user profiles enhances job matching accuracy. This implementation lays the groundwork for future scalability, additional role-specific features, and performance optimization.

Chapter 4

Test Cases

This chapter includes the test cases created to verify each module of the system. Each test case was executed and validated based on expected output, and a figure is provided after every test case to show the execution result or snapshot.

Test Case: TC-001 – Valid Registration

Criteria: User registers with valid details

Steps:

1. Go to Sign Up page
2. Enter all required fields
3. Click 'Sign Up'

Expected Output: Account is created, data is saved, and user is redirected to login

Actual Output: Account created and redirected.

Result: Pass

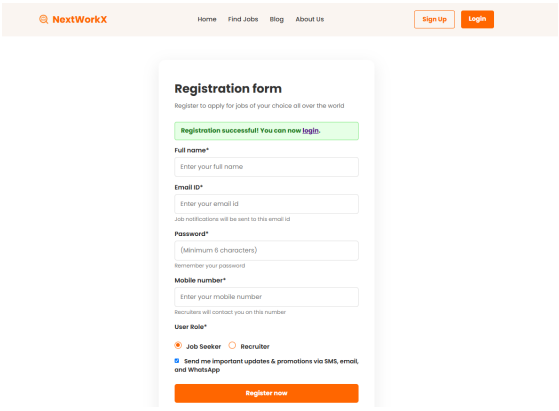

Figure 4.1: Test Case TC-001 – Valid Registration

Test Case: TC-002 – Valid Login

Criteria: User logs in with valid credentials

Steps:

1. Go to Login page
2. Enter credentials
3. Click 'Login'

Expected Output: User is authenticated and redirected to dashboard

Actual Output: User authenticated and dashboard loaded

Result: Pass

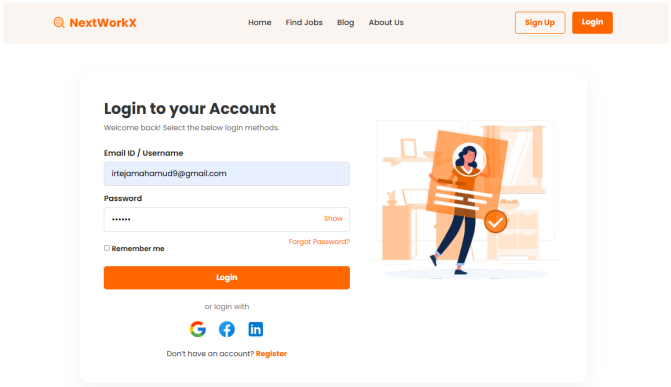

Figure 4.2: Test Case TC-002 – Valid Login

Test Case: TC-003 – Job Recommendation

Criteria: System recommends jobs based on user skills

Steps:

1. Login as job seeker
2. Click 'Show My Matches'

Expected Output: Job recommendations based on TF-IDF scores are shown

Actual Output: Top 5 job matches displayed

Result: Pass

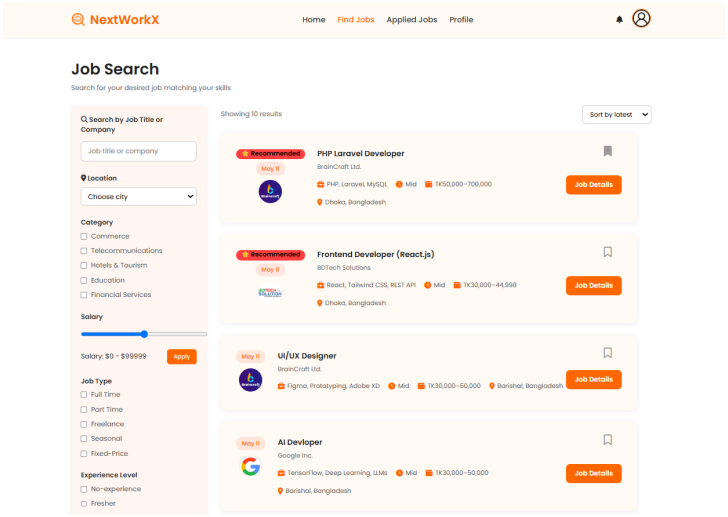

Figure 4.3: Test Case TC-003 – Job Recommendation

Test Case: TC-004 – Job Application

Criteria: User applies for a selected job

Steps:

1. View job details
2. Click 'Apply Now'

Expected Output: Application submitted and recorded in the database

Actual Output: Application saved successfully

Result: Pass

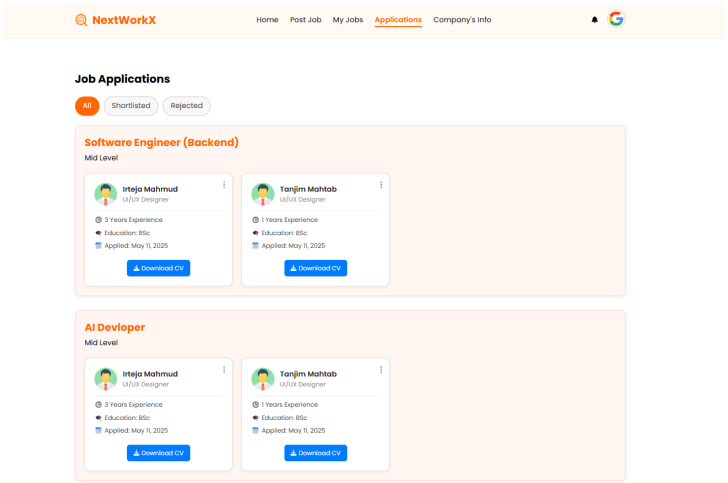

Figure 4.4: Test Case TC-004 – Job Application

Test Case: TC-005 – Resume Upload

Criteria: User uploads a valid resume

Steps:

1. Go to resume upload section
2. Upload a valid PDF resume

Expected Output: Resume is validated and uploaded successfully

Actual Output: Resume saved and displayed

Result: Pass

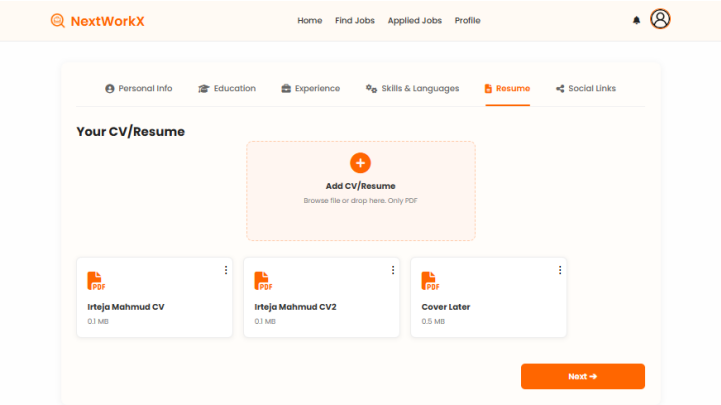

Figure 4.5: Test Case TC-005 – Resume Upload

Test Case: TC-006 – Admin Login

Criteria: Admin logs in with valid/invalid credentials

Steps:

1. Go to Admin login page
2. Enter credentials
3. Click 'Login'

Expected Output: Admin is redirected to dashboard or shown an error

Actual Output: Admin login tested successfully for both cases

Result: Pass

Test Case: TC-007 – Job Posting

Criteria: Recruiter posts a new job

Steps:

1. Login as recruiter
2. Fill job post form
3. Submit

Expected Output: Job saved in database; error shown for empty required fields

Actual Output: Job posted and validation worked correctly

Result: Pass

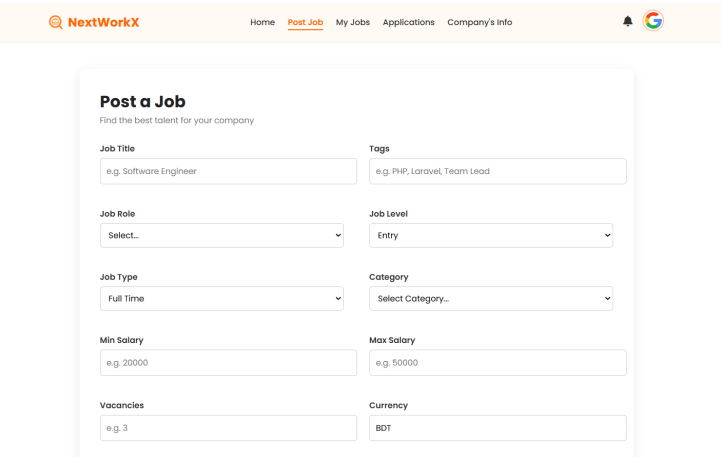

Figure 4.6: Test Case TC-007 – Job Posting

Test Case: TC-008 – Company Profile Setup

Criteria: Recruiter completes company profile

Steps:

1. Go to Company Profile in Settings
2. Fill and submit each step

Expected Output: Info saved step-by-step; errors if skipped

Actual Output: Profile completed and validations worked

Result: Pass

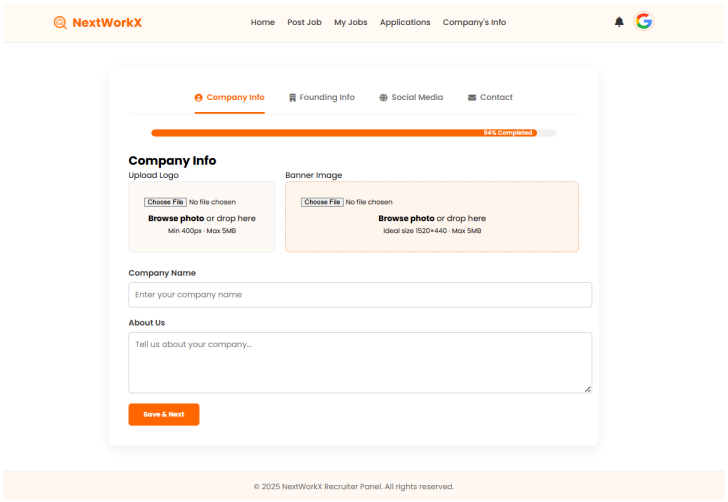

Figure 4.7: Test Case TC-008 – Company Profile Setup

Test Case: TC-009 – Bookmark Job

Criteria: Job seeker bookmarks a job

Steps:

1. Login and click on job bookmark
2. View and remove from saved list

Expected Output: Job added and removed from bookmark list as expected

Actual Output: Functionality worked correctly

Result: Pass

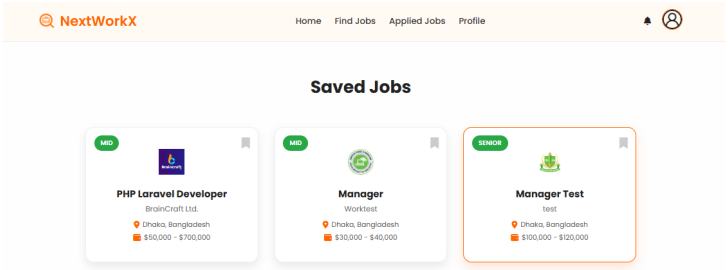

Figure 4.8: Test Case TC-009 – Bookmark Job

Test Case: TC-010 – Search Filter

Criteria: User searches and filters jobs

Steps:

1. Enter search keyword and filter options

2. Click search
3. Clear filters

Expected Output: Filtered jobs shown correctly; reset works as expected

Actual Output: Filters applied and cleared successfully

Result: Pass

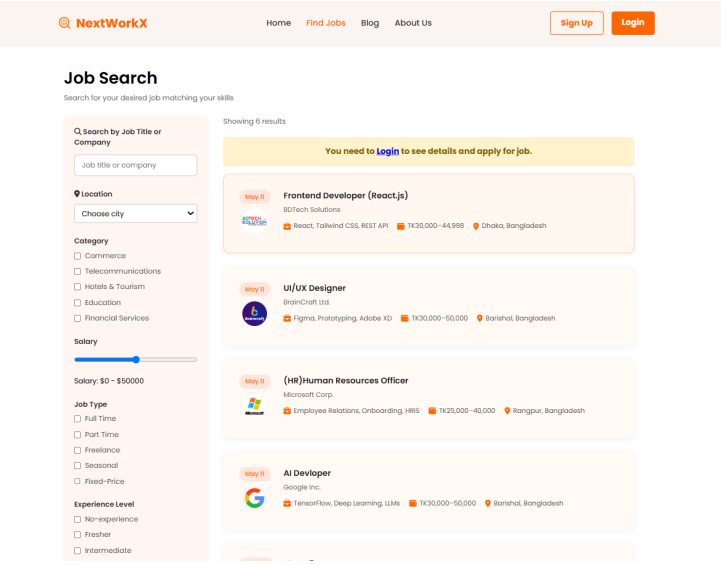

Figure 4.9: Test Case TC-010 – Search Filter

Conclusion

All the critical features of the NextWorkX platform were thoroughly tested through a series of structured test cases. Each module, from user registration to job recommendation and application, performed as expected and passed validation checks. The results confirm that the system is stable, functional, and ready for deployment in a real-world environment.

Chapter 5

Conclusion and Future Work

5.1 Summary of the Project

The aim of this project was to develop an AI-powered job recommendation platform named **NextWorkX** that bridges the gap between job seekers and recruiters through intelligent matchmaking. The system leverages Natural Language Processing (NLP) techniques and Machine Learning (ML) models to suggest jobs based on a user's skills, profile information, and uploaded resume content.

The platform is built as a full-stack web application with a structured interface for different user roles: job seekers, recruiters, and administrators. Job seekers can search, apply, and track job applications, while completing a step-by-step profile. Recruiters can post jobs, manage applications, and build company profiles. An admin panel is included for overall system oversight.

The core recommendation logic uses keyword extraction and TF-IDF-based similarity scoring to match candidate profiles with job descriptions. All backend logic is handled using PHP and Python scripts, with MySQL managing relational data and a modular frontend developed using HTML, CSS, JavaScript, and Tailwind CSS.

This project successfully integrated UI/UX design, role-based access, and intelligent job matching into a cohesive and functional platform.

5.2 Social Impact

The NextWorkX platform addresses a critical issue in today's job market: the disconnect between applicants and relevant job opportunities. Traditional job portals often rely on manual search and filtering, which can be overwhelming and inefficient. Our system simplifies this process by intelligently suggesting jobs based on the actual content of user profiles and resumes.

This project has the potential to:

- Help fresh graduates and experienced professionals find jobs faster, with higher relevance.

- Reduce the time and effort spent browsing irrelevant listings.
- Assist recruiters in identifying suitable candidates without sifting through unqualified applications.
- Promote fairer access to employment by recommending opportunities based on merit and content, not keyword stuffing or manual searches.

In the long run, platforms like this can contribute to reducing unemployment mismatches, supporting career development, and streamlining digital hiring in both urban and rural regions.

5.3 Future Work

While the current system is functional and has passed all major test cases, there is room for significant enhancement. Future development can focus on the following areas:

- **Advanced Resume Parsing:** Integrating deep learning-powered NLP models to extract structured information from unformatted resumes (PDF/DOCX).
- **OAuth Integration:** Adding third-party authentication (Google, LinkedIn) for faster and more secure sign-in processes.
- **BERT-Based Matching:** Moving beyond TF-IDF to use context-aware models like BERT for more accurate and semantic job matching.
- **Mobile Application:** Designing and deploying a mobile app version of NextWorkX to increase accessibility on smartphones and tablets.
- **Notification and Engagement Tools:** Including features like job alerts, recruiter messages, interview scheduling, and application tracking dashboards.
- **Analytics Dashboard for Admin:** Visual analytics for jobs posted, user engagement, and skill/job market trends.
- **AI Interview Preparation Tools:** Adding modules that help job seekers practice interview questions based on job roles and skill levels.
- **Gamified Learning and Profile Improvement Tips:** Encouraging users to complete their profile through interactive progress badges and tips for improving their visibility to recruiters.

With these enhancements, the platform can evolve into a more intelligent, inclusive, and impactful recruitment tool.

Bibliography

[1] S. Panchasara, R. K. Gupta and A. Sharma, "AI Based Job Recommendation System using BERT," in *2023 7th International Conference On Computing*.

[2] V. S. Pendyala, N. Atrey, T. Aggarwal and S. Goyal, "Enhanced Algorithmic Job Matching based on a Comprehensive Candidate Profile using NLP and Machine Learning," 2022.

[3] E. Abdollahnejad, M. Kalman and B. H. Far, "A Deep Learning BERT-Based Approach to Person-Job Fit in Talent Recruitment," 2021.

[4] S. Jogi, S. Warang, H. N. Bhor, D. Solanki and H. Patanwadia, "NLP Unleashed: Transforming Employment Landscapes with Dynamic Recruitment Platforms," 2024.

[5] A. Pradhan, "Employee–Job profile matching challenges: AI/ML-based recommender systems solutions for knowledge workers," 2024.