PrepScore
Career Profile Optimizer

Minna Joby
Roll No: 35
Reg.No.: KTE24MCA-2038

Guided By : Prof. Shilpa M. Thomas /Dr. Reena Murali
Department of Computer Applications
Rajiv Gandhi Institute Of Technology, Kottayam

Introduction

- Students lack a data-driven way to measure their job readiness; traditional resumes are static and offer no feedback.
- A smart web application to build a comprehensive digital profile (skills, education, experience).
- An intelligent engine that analyzes the profile to generate a real-time **Career Readiness Score.**

Existing Systems

- A review was conducted on the primary tools students use for professional profile management.
- The analysis focused on two main categories:
  - **Professional Networking Platforms** (e.g., LinkedIn)
  - **Online Resume Builders** (e.g., Zety, Canva)
- The objective was to evaluate their core features to identify functional gaps in providing career readiness guidance.

Literature Review

Tool Category	Strengths	Limitations / Gap
Networking Platforms (e.g., LinkedIn)	Builds online presence; has a basic profile strength meter.	Lacks a deep, quantitative analysis of skills. Feedback is generic and not personalized.
Resume Builders	Creates visually appealing, well-formatted PDF resumes.	Acts as a static, one-time tool. Offers no content analysis or guidance for improvement.

Gap Identified

- **No Quantitative Score:** Existing tools fail to provide a simple, data-driven score to benchmark job readiness.
- **Static, Not Strategic:** Platforms focus on listing past achievements, not providing a strategic path for future improvement.
- **The Need:** A dedicated **Profile Optimizer** with a dynamic feedback loop.

Proposed System

- **A Centralized Profile Hub:** A secure web app with full **CRUD** functionality for users to manage their skills, education, and experience.
- **Intelligent Scoring Engine:** A core algorithm that analyzes the profile to generate a real-time, quantitative **"Career Readiness Score."**
- **Dynamic Feedback Loop:** The score updates instantly with profile changes, providing immediate feedback to guide strategic improvement.

System Requirements

Software Requirements	Development Tools
▪ **Operating System:** Windows, Linux	▪ **Version Control:** Git & GitHub
▪ **Backend:** Python 3.+, Django 5.+	▪ **Code Editor:** Visual Studio Code
▪ **Frontend:** HTML5, CSS3	▪ **Database GUI:** pgAdmin
Hardware Requirements	
▪ **Development:** Standard PC/Laptop	
Minimum 4GB of RAM	
▪ **End-User:** Any device with a modern browser	

Problem Statement

To develop a web application that solves the uncertainty students face by replacing static resumes with a dynamic analysis of their professional profile.The core challenge is to generate a quantitative "**Career Readiness Score**" to serve as a clear benchmark for self-assessment and guided improvement.

Objectives

- **Develop a Full-Featured Web Application:** Build a secure platform with user authentication and complete **CRUD (Create, Read, Update, Delete)** functionality for all profile sections.
- **Implement an Intelligent Scoring Engine:** Create a **rule-based algorithm** to score a user's profile in real-time, with the future scope of evolving it into a predictive **Machine Learning model**.
- **Provide Actionable Feedback & Guidance:** Generate personalized **suggestions** based on the profile analysis to help users strategically improve their career readiness.

Scope & Relevance

Scope

A full-stack web application with user authentication, complete profile CRUD, and a real-time, rule-based scoring engine . Students and recent graduates preparing for the job market . Implementation of an advanced ML model and personalized suggestions.

Relevance

Addresses student uncertainty by providing a data-driven tool for career self-assessment . Shifts the focus from static resumes to dynamic profile optimization . Integrates full-stack development, database management, and data analysis principles.

Development Methodology - Model

The project was developed using an **Agile methodology**, specifically following an **Iterative and Incremental** process.

Why Agile?

- **Flexibility:** Allowed for continuous improvement and adaptation to new requirements.
- **Rapid Prototyping:** Delivered functional parts of the application in short, manageable cycles (iterations).
- **Early Feedback:** Enabled testing and validation of features at each stage, reducing overall risk.

Development Methodology - Phases

Phase 1: Requirement Analysis & Design	• Conducted literature review, identified gaps, and defined the project scope, objectives, and system architecture. Designed the database schema (ER Diagram).
Phase 2: Foundation & Authentication	• Set up the Django project, configured the PostgreSQL database, and implemented the complete, secure user authentication system (Register, Login, Logout).
Phase 3: Core Feature Implementation (CRUD)	• Developed the full Create, Read, Update, and Delete functionality for all core profile components (Profile, Skills, Education, Experience, Certifications).
Phase 4: Intelligence Layer & UI Polish	• Currently implementing the rule-based scoring engine and integrating it with the user dashboard. This phase also includes final UI enhancements and testing.

Development Methodology - Workflow

A systematic and repeatable workflow was followed for the implementation of each feature:

1. **Plan:** A clear, specific task was identified from the project plan (e.g., "Implement User Login").
2. **Develop (MVT Pattern):** The feature was built following Django's Model-View-Template architecture, ensuring a clean separation of concerns.
3. **Test:** The functionality was manually and thoroughly tested in the browser to identify and fix any bugs immediately.
4. **Commit & Document:** Once working, the code was saved to version control with a clear **Git** commit message, and the progress was logged in the **Scrum Book**.

Data Flow Diagram

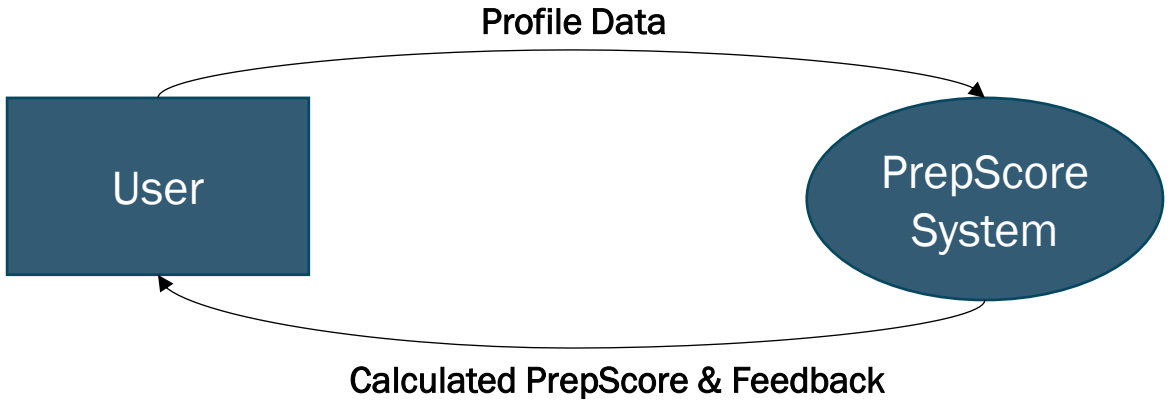

FIG 1: DFD LEVEL 0

Data Flow Diagram

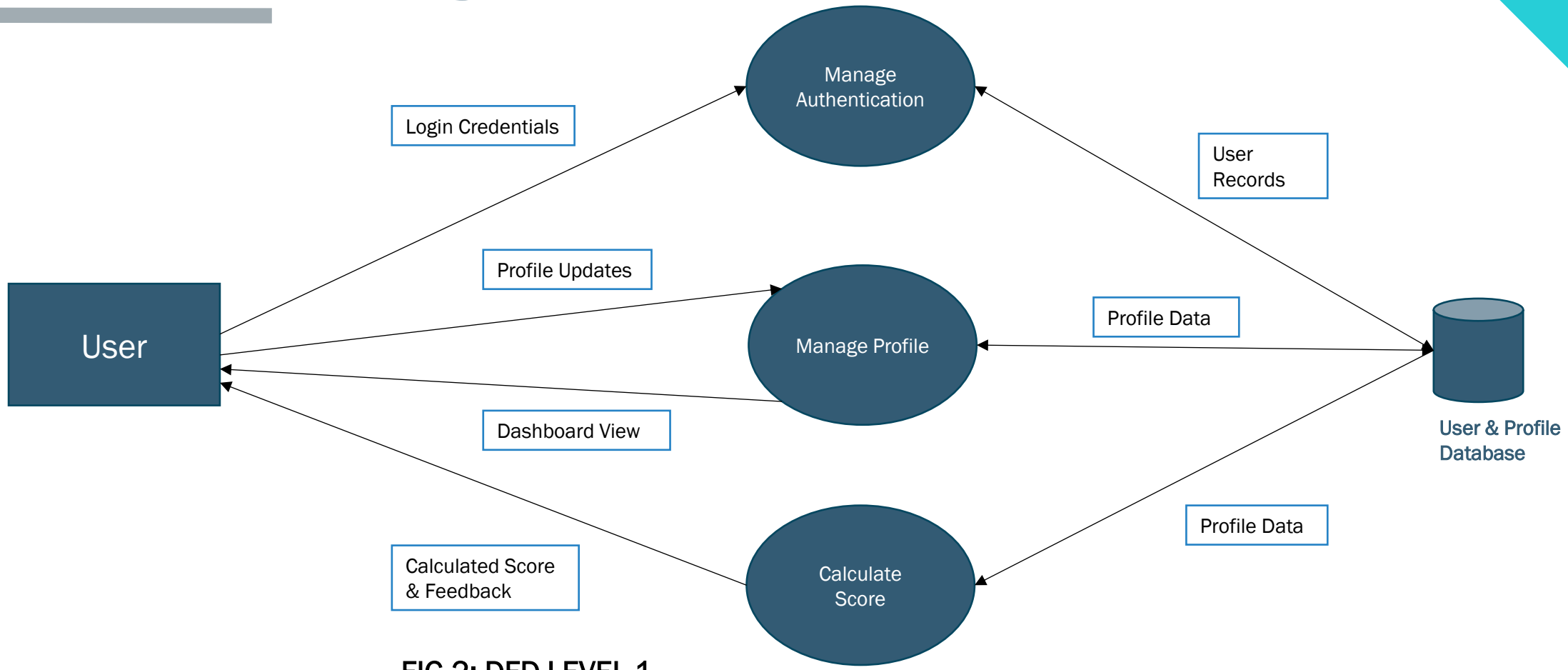

FIG 2: DFD LEVEL 1

Implementation Details

Backend Logic

Framework: Built on the **Django 5.2.4 MVT (Model-View-Template)** architecture.

Core Logic:

- **Views:** Handle all business logic, with sensitive pages protected by the @login_required decorator.
- **Custom Authentication:** A custom backend allows users to log in with either a username or an email.
- **Forms:** Django ModelForms are used for rapid development and validation of all CRUD forms.

Implementation Details

Frontend

Template Inheritance:

- Utilizes a base.html for public pages and a separate dashboard_base.html app shell for a consistent, logged-in user experience.
- This maximizes code reusability (DRY principle).

- Dynamic Rendering:
  - **Conditional Logic ({% if %}):** Used to display different navigation for guests vs. logged-in users.
  - **Loops ({% for %}):** Used to dynamically display lists of user data (skills, education) on the dashboard.

Implementation Details

Database & Security

- **Database:** A robust **PostgreSQL** relational database ensures data integrity.
- **Data Security:**
  - **Ownership Checks:** All Update/Delete views verify that the user (request.user) owns the data they are trying to modify.
  - **CSRF Protection:** Django's built-in CSRF tokens are used in all forms to prevent cross-site attacks.
- **Configuration Security:**
  - Sensitive credentials (e.g., SECRET_KEY, DB password) are securely stored in a .env file and are excluded from version control.

Results

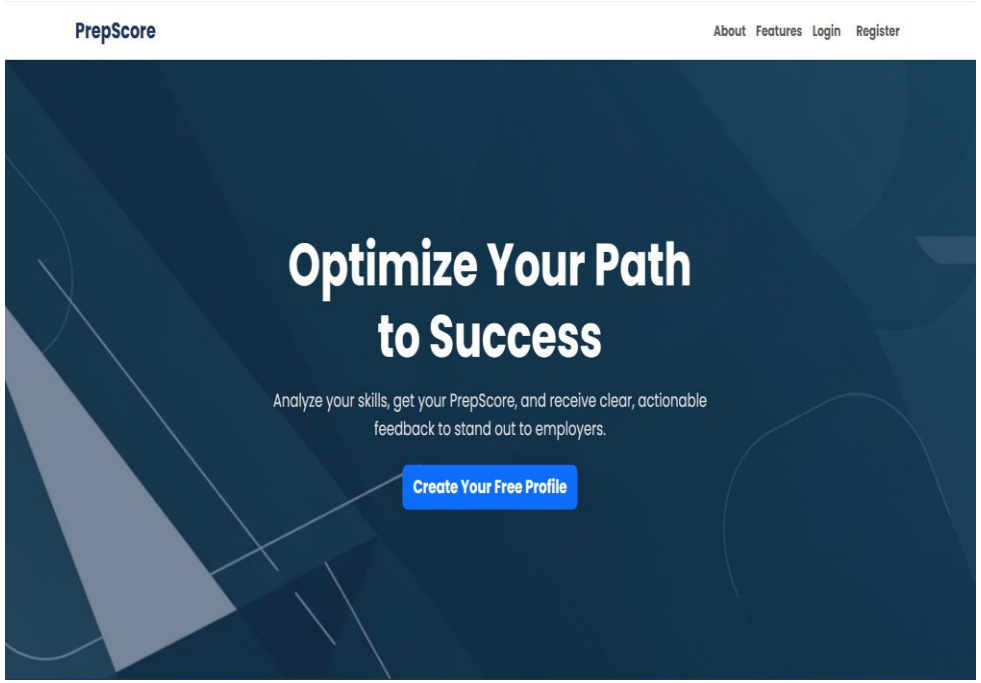

FIG 3: HOMEPAGE

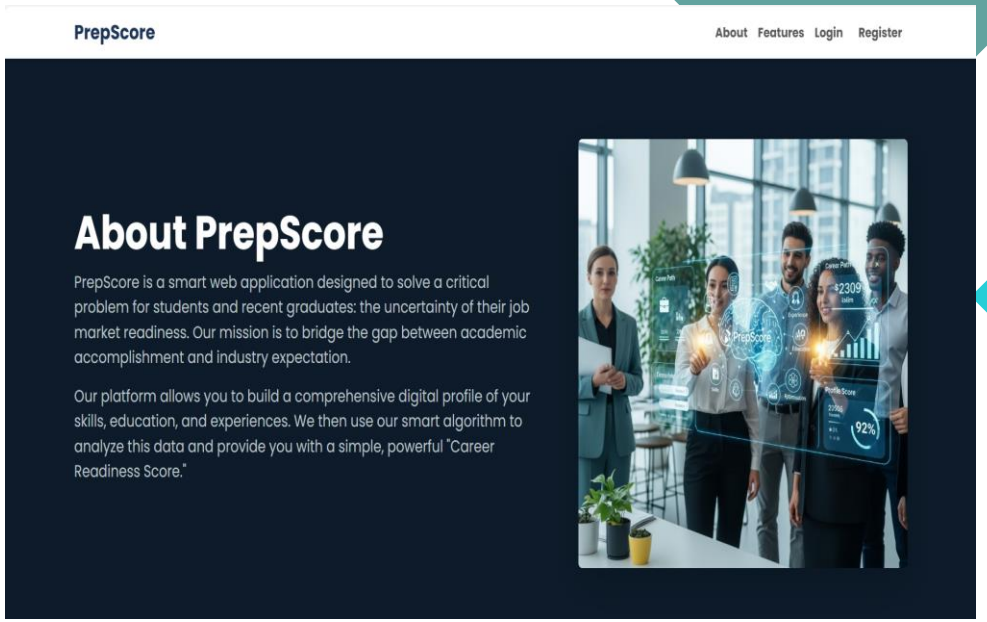

FIG 4: ABOUT SECTION

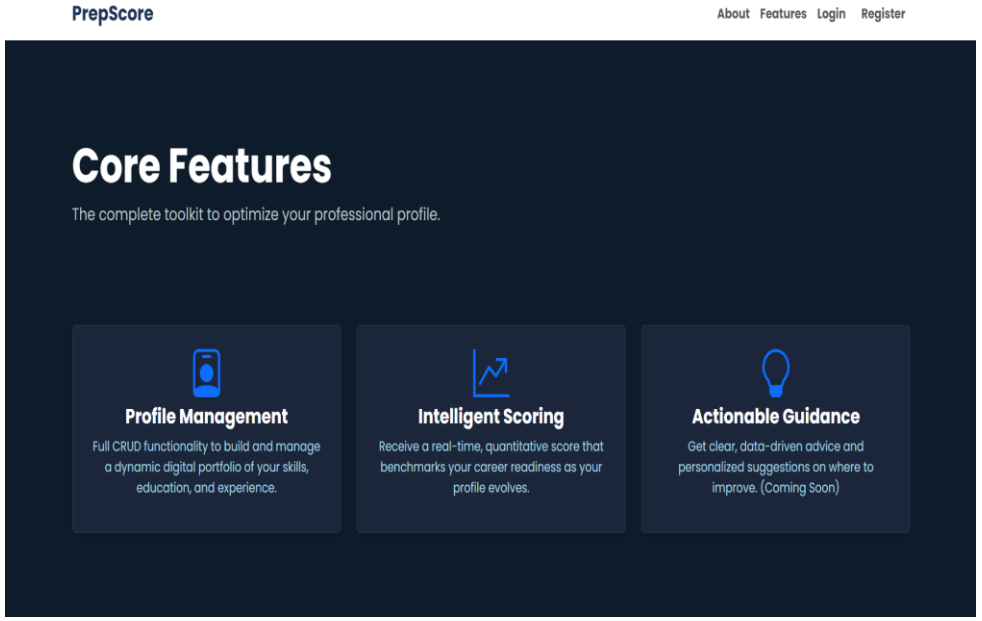

FIG 5: FEATURES

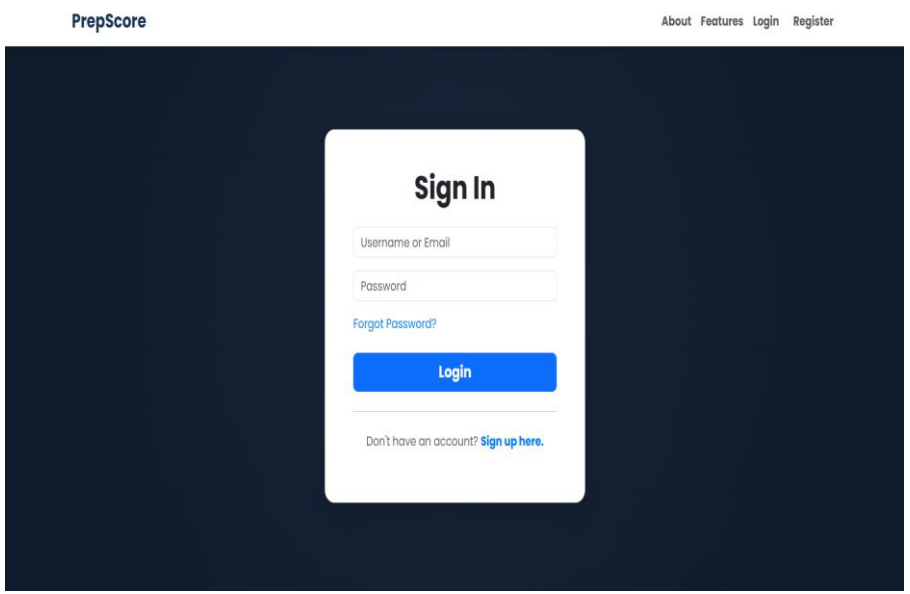

FIG 6: LOGIN

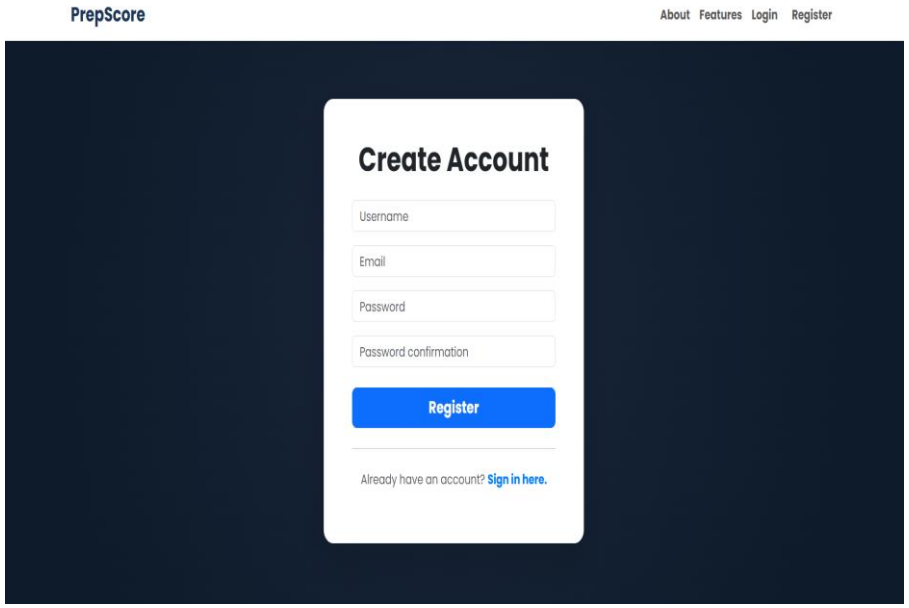

FIG 7: REGISTER

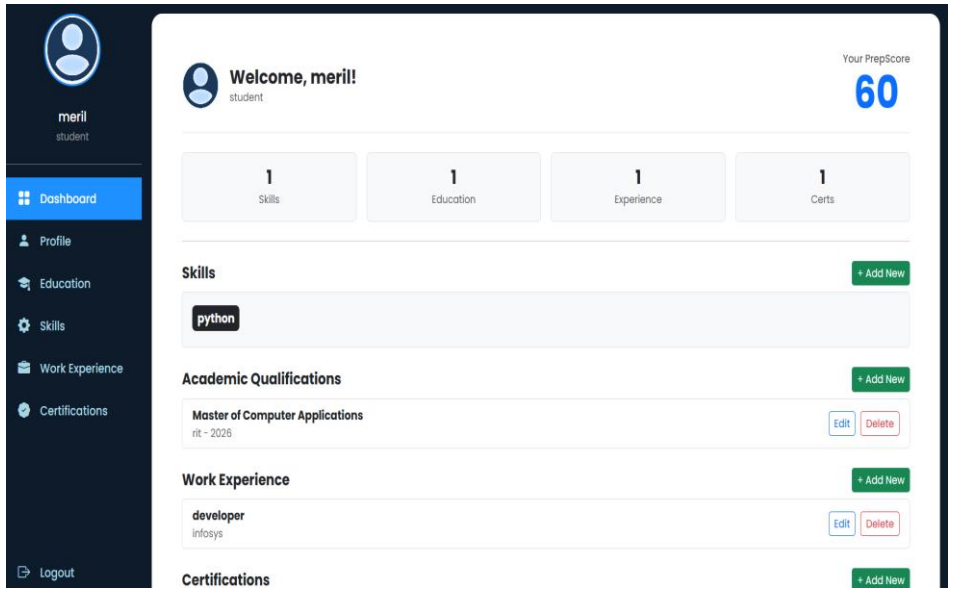

FIG 8: DASHBOARD

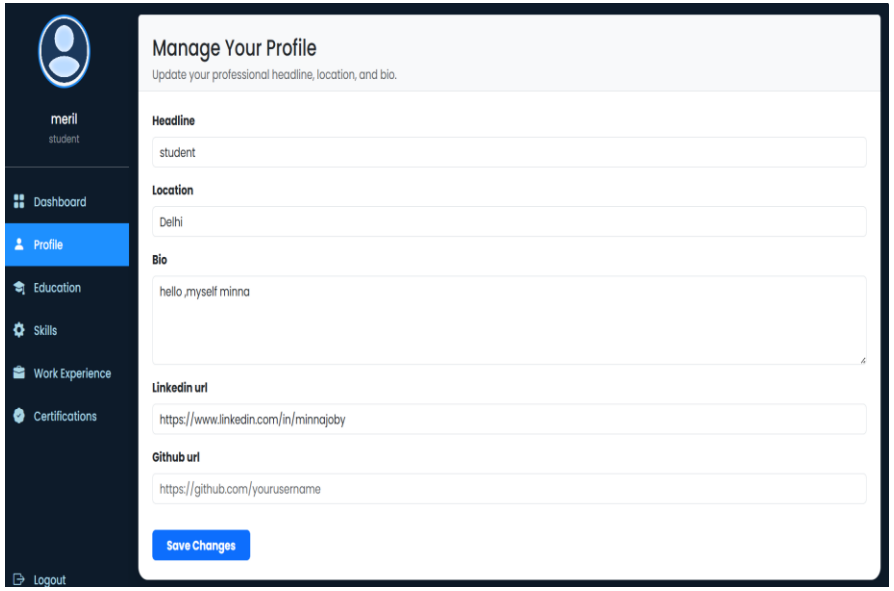

FIG 9: PROFILE MANAGEMENT

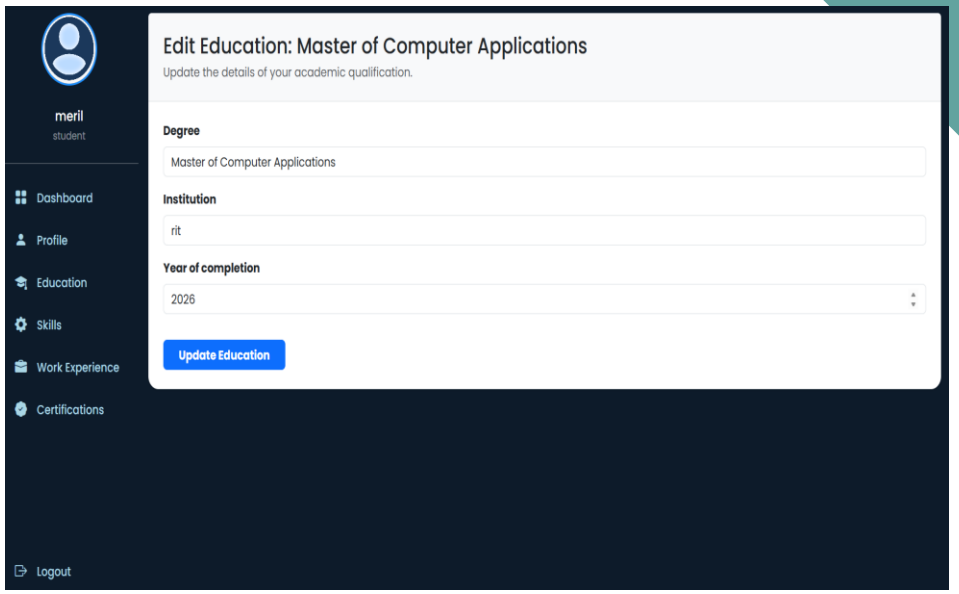

FIG 10: EDITING FEATURE

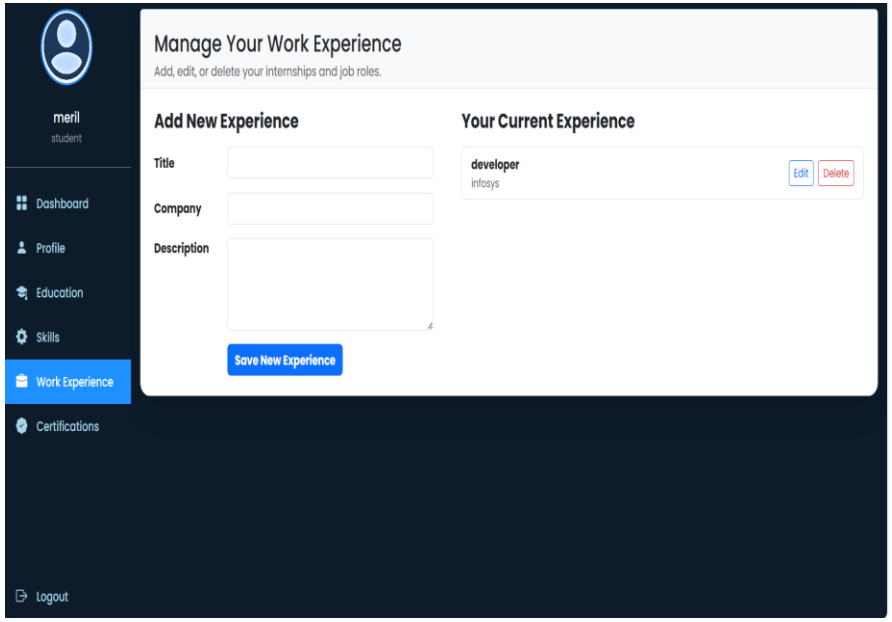

FIG 11: EXPERIENCE FORM

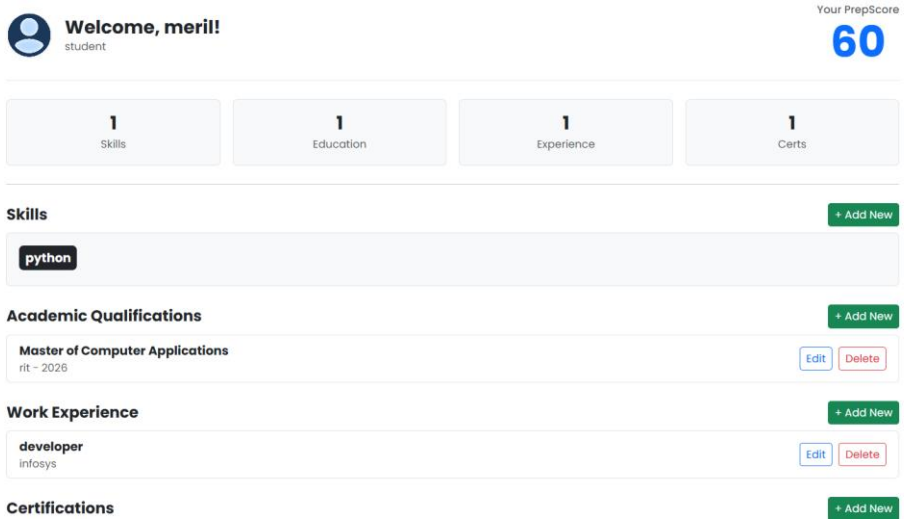

FIG 12 : PREPSCORE

Current Status of Work

- The project is currently **70% complete**.
- All foundational architecture and core user-facing features required for a functional application are implemented and working.
- **Completed Components:**
  - Full User Authentication System (Login, Register, Logout)
  - Complete Profile Management (Full CRUD for all models)
  - A functional, real-time, rule-based Scoring Engine
  - A professional and responsive user interface

Work Progress

- **Foundation & Security (Complete):**

Project setup, PostgreSQL database, and user authentication are fully implemented.

- **Core Functionality (Complete):**

Full CRUD (Create, Read, Update, Delete) for the entire user profile is functional.

- **Intelligence Engine (In Progress):**

A **rule-based scoring algorithm** has been successfully implemented.

Next Step: Evolve the engine into a predictive **Machine Learning model**.

Pending Works

- **Evolve to a Machine Learning Model:**
  - Advance the current rule-based scoring engine by training a predictive **Machine Learning model** for a more accurate and nuanced profile analysis.
- **Implement Personalized Guidance Engine:**
  - Develop the algorithm to generate targeted, data-driven **suggestions** for user improvement based on their profile data.
- **Enhance UI & Finalize:**
  - Improve the dashboard with **data visualizations** (e.g., charts for the score) and conduct final end-to-end testing and documentation.

Project Plan

Phase	Phases	Key Deliverables	Timeline (Weeks)	Status
1	**Analysis & Design**	Requirement Analysis, System Design (DFD, ERD), Database Schema	1 - 2	**Completed**
2	**Foundation & Auth**	Project Setup, Security, Full User Authentication System	2 - 4	**Completed**
3	**Core Functionality**	Full CRUD for Profile, Skills, Education, Experience, Certifications	4 - 6	**Completed**
4	**Intelligence & Polish**	Rule-Based Scoring, UI Enhancements, Final Testing	6 - 8	**In Progress**

Conclusion & Future Scope

Conclusion:

- Successfully developed a **secure, functional web application** with a complete profile CRUD system and a working rule-based scoring engine. The project is **70% complete** and meets all evaluation milestones.

Future Scope:

- Evolve the scoring engine into a predictive **Machine Learning model**.
- Implement a **personalized suggestions** engine.
- Add a **PDF resume generation** feature.

Git History

Commits on Sep 1, 2025

Feature: Build complete CRUD functionality and dashboard UI
minnajoby committed 3 days ago — f44222e

Commits on Aug 30, 2025

Build professional About and Features pages with Bootstrap
minnajoby committed 5 days ago — 8571e63

Commits on Aug 19, 2025

Feature: Add Skill editing and refactor Dashboard UI
minnajoby committed 2 weeks ago — 8847c5a

Commits on Aug 5, 2025

Design: Implement homepage layout with header and footer
minnajoby committed on Aug 5 — 6f82cea

Commits on Jul 29, 2025

Feature: Implement user authentication with login, logout, and dashboard
minnajoby committed on Jul 29 — 5842eed

Commits on Jul 22, 2025

Secure secrets using .env file
minnajoby committed on Jul 22 — 9a570f7

Feature: Create and display basic home page
minnajoby committed on Jul 22 — 8b37335

Commits on Jul 20, 2025

Initial Commit - prepscore project
minnajoby committed on Jul 20 — b4e1392

Bibliography

Core Technologies & Documentation:

- Python:
  - *Official Documentation:* https://docs.python.org/3/
- Django Framework:
  - *Official Documentation:* https://docs.djangoproject.com/en/5.2/
- PostgreSQL:
  - *Official Documentation:* https://www.postgresql.org/docs/

Thank you