TalentEngine – AI-powered Recruitment System

Aimurat Zhetkizgenov[1], Dastan Kozhabayev[1], Yerassyl Myrzakhanov[1], Alfiya Lugma[1], and Dias Sagatov[1]

Advisor: Professor Michael Lewis
Nazarbayev University, Astana, Kazakhstan[1]
School of Engineering and Digital Sciences
Computer Science Department
seds.nu.edu.kz

1 Executive Summary

TalentEngine is a new recruitment system designed to improve and optimize hiring processes. The system uses artificial intelligence to eliminate the inefficiencies of existing modern recruitment methods. By using AI and machine learning models, TalentEngine is able to automate the ranking of candidates based on their resumes and online interview results. This approach will allow HR teams to find ideal candidates more effectively and save significant time.

TalentEngine aims to reshape and improve recruitment by following these **key objectives:**

- **Automated Screening and Ranking:** utilize AI models to evaluate candidates' resumes, test results, and interview responses, generating ranked lists based on job description fit.
- **Application Management:** the platform features an Applicant Tracking System (ATS) that makes it easy to manage resumes, track where candidates are in the hiring process, and coordinate interviews.
- **Comprehensive Evaluation Suite:** develop interactive testing and interview modules with question formats ranging from multiple-choice questions to audio/video/text responses, with automated evaluation for them.
- **Secure and Scalable Infrastructure:** deploy the platform on cloud services like AWS, ensuring data safety and high performance even under high load.

The development of TalentEngine was guided by the Agile methodology which allowed us to work in iterative cycles and respond flexibly to evolving requirements. This approach facilitated continuous feedback, early identification of issues and quick response to them. Our team used Notion and Jira to manage tasks, track progress, and coordinate activities. These tools allowed for clear distribution of responsibilities, planning, and backlog refinement.

1.1 Main Results

By the end of the development cycle, the TalentEngine system has been implemented. All core functionalities across the backend, frontend, and AI components tested and integrated into a cohesive product. **Key achievements include:**

- A user-friendly Applicant Tracking System (ATS) interface for efficient applicant sorting and management.
- The creation of online interviews with customizable question formats, including multiple-choice and open-ended questions.
- A robust AI pipeline for ranking, transcription, and evaluation of resumes and interview responses. The system ensures fast, transparent, and fair hiring, delivering an end-to-end solution aligned with modern HR needs.
- A complete backend with secure APIs and AI integration.

1.2 Design

The project began with a detailed analysis of **user requirements** from both recruiter and applicant perspectives. It lead to:

- A clearly defined system architecture, including modular components for frontend, backend, and AI services.
- Use case modeling, which mapped key user actions such as job posting, application submission, and interview handling.
- Data modeling, which structured entities like resumes, interviews, AI evaluations, and communication threads into a relational database schema.
- Selection of cutting-edge technologies such as SBERT for semantic resume-job matching, Whisper for audio/video transcription, and LLMs for deep candidate evaluation.

1.3 Implementation

Technology stack:

- **Frontend**: Built with React and Next.js, offering a responsive and intuitive user experience, including a full-featured ATS interface.
- **Backend**: Developed using FastAPI and PostgreSQL, handling business logic, user authentication, job management, and API integration.
- **AI/ML Integration**:
  - Stage 1: Used SBERT to embed job descriptions and resumes, ranking candidates via cosine similarity.

- Stage 2: Applied LLMs to the top 50% of candidates for deeper, context-aware analysis of resumes and interview transcripts.
- Whisper was employed to transcribe audio and video responses into text, enabling NLP-based scoring.

1.4 Evaluation

From a technical standpoint, the system was subjected to extensive testing which included API validation, load testing, and end-to-end functional checks. These tests confirmed the system's ability to handle a high volume of concurrent user requests and parallel processing tasks, such as the transcription of audio and video responses.

The AI and ML components were also rigorously assessed. Initially, candidate ranking relied on keyword-based scoring, but testing revealed its limitations—such as difficulty understanding context and overemphasis on isolated terms. To solve this problem, it was decided to use a two-stage ranking approach. In the first stage, SBERT was used to find semantic similarities between the resume and the job description. This stage allowed for an initial filtering of candidates, which was a quick and easy solution before using LLM. However, the final ranking and the emergence of the most suitable list of candidates occurred in the second stage, where a contextual assessment of the resume and online interview responses was given. This solution significantly increased the model's ability to assess the relevance of candidates.

In addition to performance, we also took into account ethical and legal issues. Our platform is GDPR compliant, uses AES-256 encryption, and secure SSL communications. Moreover, understanding the possible risks of algorithmic bias in models, TalentEngine gives full responsibility to the HR manager at the stage of the final hiring decision, i.e. the choice of a candidate depends only on the person using the platform. This approach ensures that AI recommendations are reviewed by recruiters, which will help maintain fairness, competitiveness, and reliability during the hiring of candidates.

2 Introduction

2.1 Problem, motivation, and significance of the project.

In today's competitive job market organizations often struggle with inefficient and time-consuming recruitment processes. Traditional hiring methods involve manual resume screening, inconsistent evaluation criteria, and lack of streamlined communication with applicants. These challenges are amplified when handling high volumes of applications, leading to delayed hiring decisions, increased HR workload, and a subpar experience for candidates.

There are many existing HR / recruitment systems on the market, such as that used at NU "SmartRecruiters". However, many of these systems are based on old technology, with limited features and functionality, and for the most part do not integrate new developments in machine learning and large language models that would help streamline and optimize the recruitment process from the perspective of the employer and the applicant.

We propose to identify and implement the core functionality of such systems, and then augment that functionality with ML-based tools that will facilitate the matching of applicants to open positions, and improve the communication amongst the users. TalentEngine aims to transform recruitment by introducing an AI-powered, end-to-end hiring platform that improves the speed, accuracy, and fairness of candidate evaluation. The platform is deployed on a secure and scalable cloud infrastructure (AWS) and includes an applicant tracking system (ATS) for managing the hiring workflow from job posting to candidate feedback.

2.2 Brief overview of the proposed solution

The report is structured into the following sections:

- **Background/Related Work** reviews existing research and technologies in AI-driven recruitment, highlighting current limitations and motivating the development of TalentEngine.
- **Project Approach** outlines the system's overall design, including its objectives, architecture, features, and the tools and methodologies selected to implement the solution.
- **Project Execution** provides a detailed account of the implementation process, covering both frontend and backend development, AI/ML pipeline construction, integration of third-party services, and infrastructure setup.
- **Evaluation** assesses the technical performance, model effectiveness, and efficiency compliance of the platform. It discusses testing outcomes, model validation, and accuracy measures.
- **Conclusion and Possible Future Work** summarizes the achievements of the current development cycle and presents a roadmap for future enhancements, including planned feature expansions and further optimization of the system.

3 Background/Related Work

AI-driven recruitment tools like **Iris by Qureos [7], X0PA AI [3], HeroHunt.ai [2],** and **hireEZ [1]** have introduced features such as candidate management, basic ranking, and automated communication Table 1. However, these platforms often rely on keyword-based

Table 1. A comparison between existing AI Recruitment tools and TalentEngine

Feature	Iris by Qureos	X0PA AI	HeroHunt.ai	hireEZ	TalentEngine
Candidate Management	✓	✓	✓	✓	✓
Automated Communication	✓	✗	✓	✗	✓
Dynamic Career Site & ATS	✓	✗	✗	✓	✓
Candidate Ranking	Basic	Proprietary	Basic	Basic	Advanced AI model

matching, which lacks contextual understanding and leads to limited accuracy in candidate evaluation.

Most existing systems also focus on isolated parts of the hiring process, offering either communication tools or tracking systems, rather than a unified end-to-end solution. Furthermore, their ranking mechanisms are often simplistic or proprietary, making them less transparent and adaptable. TalentEngine addresses these gaps by aiming to integrate advanced semantic analysis and eventually **Large Language Models (LLMs)** to enhance context-aware candidate ranking. By handling multimodal data and offering a complete recruitment workflow, it builds on previous approaches to deliver a more intelligent, scalable, and fair hiring system. This approach demonstrates a solid grasp of current computing-based recruitment solutions and contributes to improving both their efficiency and equity.

4 Project Approach

TalentEngine is an AI-based recruitment tool with a built-in applicant ranking system designed to optimize the hiring process for HR managers. To fully cover all functional and non-functional requirements of the system, advanced tools in the frontend, backend and ML development were used.

4.1 Software and Hardware Architecture

Figure 1 shows the overall system and hardware architecture of the implemented system. The architecture consists of Frontend, Backend and AI/ML parts.

Frontend

- **ReactJS and NextJS** frontend frameworks were used as the main tool to develop the user-side of the TalentEngine.
- For the custom styling, recognizable as industry standard tool **TailwindCSS** was used. This instrument made it possible to make a user-friendly and responsive UI.

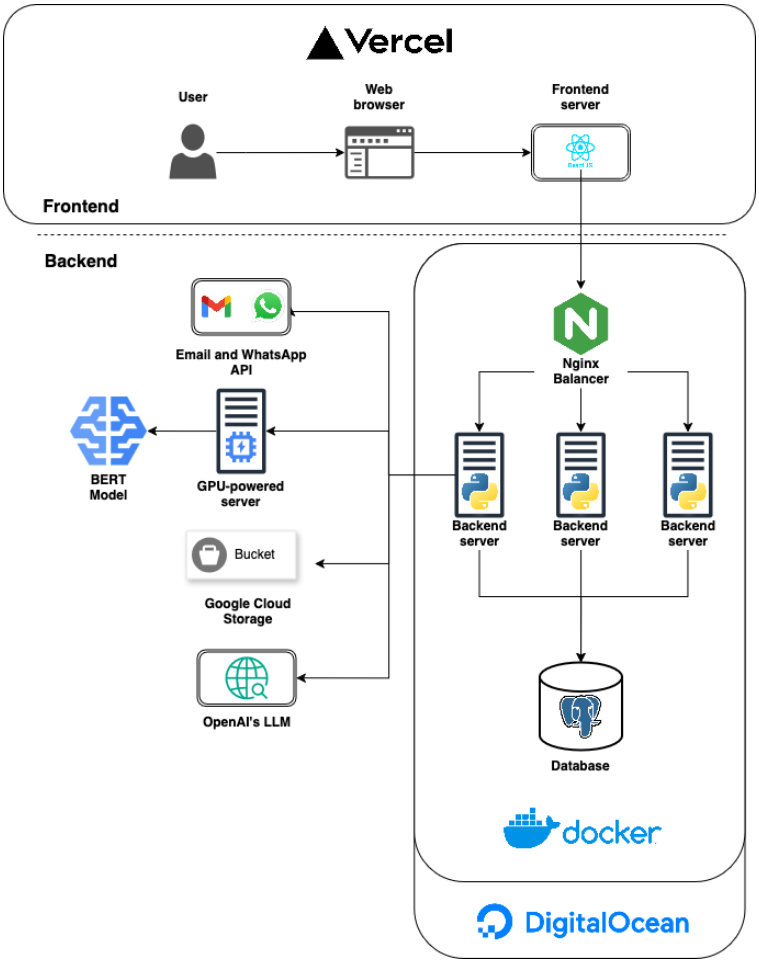

Fig. 1. System Architecture

- The frontend was integrated with a powerful survey making library - **SurveyJS**. This library was primarily used to implement the online interview functionality, including ability to create 20+ different question types.
- **Vercel** platform was used to deploy the frontend, because of Vercel's auto-scaling and high performace.

Backend

- **FastAPI** backend framework was used to create REST API endpoints and user authentication. FastAPI is an asynchronous framework built on Python that allows it to handle multiple concurrent requests.
- For database management, **PostgreSQL** was used. PostgreSQL is production-ready DBMS and has already become industry standard for the backend development.
- **Redis** NoSQL database was integrated to handle race conditions via lock allocations.
- **Nginx** is implemented to handle proxying and linking the backend server to domain name.
- Whole configuration was containerized with **Docker** and deployed on the **DigitalOcean** VPS server.
- To store applicants' CVs, audio and video responses, **Google Cloud Storage** was used. Ensuring persistent storage for files.

AI and ML Pipeline: To handle the efficient candidates ranking, the complex AI pipeline was used.

- **Resume Scoring:** our team used the BERT ("all-MiniLM-L6-v2") [8] model to create embeddings on the vacancy description and resume. These embeddings were utilized to compute cosine similarity between vacancy and CV. This phase is the initial keyword based matching.
- **Contextual Ranking:** to enhance the initial keyword matching with the context, OpenAI's GPT-4.1 [6] was integrated via state-of-the-art AI framework - PydanticAI.
- **Interview Scoring:** interview responses with multimedia were transcribed via OpenAI's Whisper [5] model, subsequently transcriptions were analyzed via chain-of-thoughts reasoning model o3-mini [4] to deeply understand the context behind applicant's answers.

Additional tools: Along with the development tools and frameworks, we used additional tools which helped us to ensure consistent development.

- **Project Management:** Jira, Notion.
- **Development:** GitHub for version control and Docker for deployment.
- **CI/CD:** Gihub Actions.

Project Management: Our team used the benefits of agile methodology, ensuring consistent performance during the whole development process. Jira and Notion platforms helped to keep track on the progress and properly delegate the tasks. GitHub repositories were used to store our codebase and ensure version control and Docker containers maintained the deployment of our services to the cloud.

4.2 Workflows and Use Cases

Use Case 1: Job Posting by HR

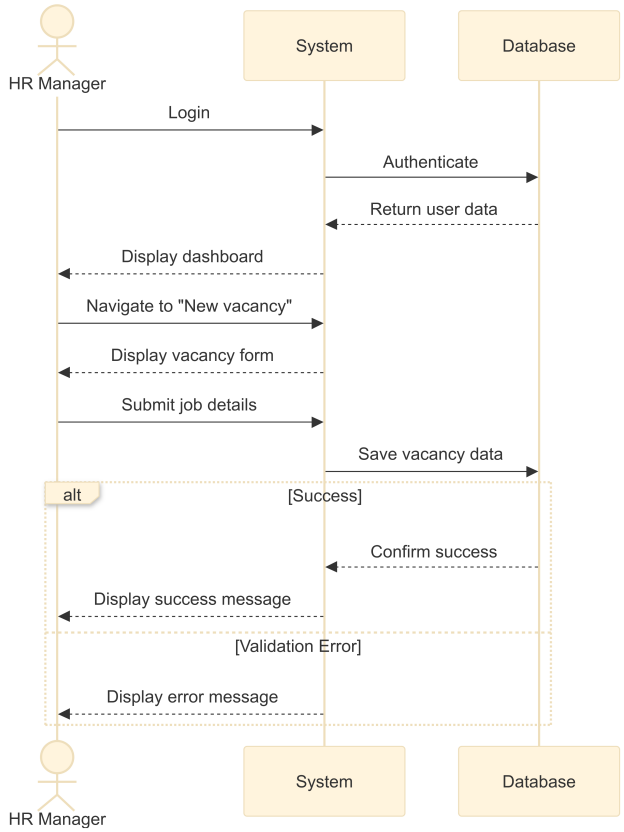

Fig. 2. Use Case 1: Job Posting by HR

Actor: HR Manager.
Goal: Post a new job vacancy on the platform.
Preconditions: HR Manager is logged into the system, has permission to add job vacancy and has job details (position name, description, requirements etc.)

Success scenario:

1. HR Manager goes to the "New vacancy" section.
2. HR Manager clicks on "Create New Job."
3. HR Manager enters job title, description, requirements, and any other necessary details.

4. HR Manager selects job categories, experience level, and desired skills.
5. HR Manager optionally links assessment tests or interview questions.
6. HR Manager clicks "Submit" to post the job.
7. The system validates the job details.
8. The job is successfully posted on the platform and made visible to candidates.

Alternative Scenario:
If the job details are incomplete or invalid, the validation message will be shown to the HR manager.
Outcome:
The job vacancy is posted, and candidates can view and apply for it.

Use Case 2: Candidate Application Submission

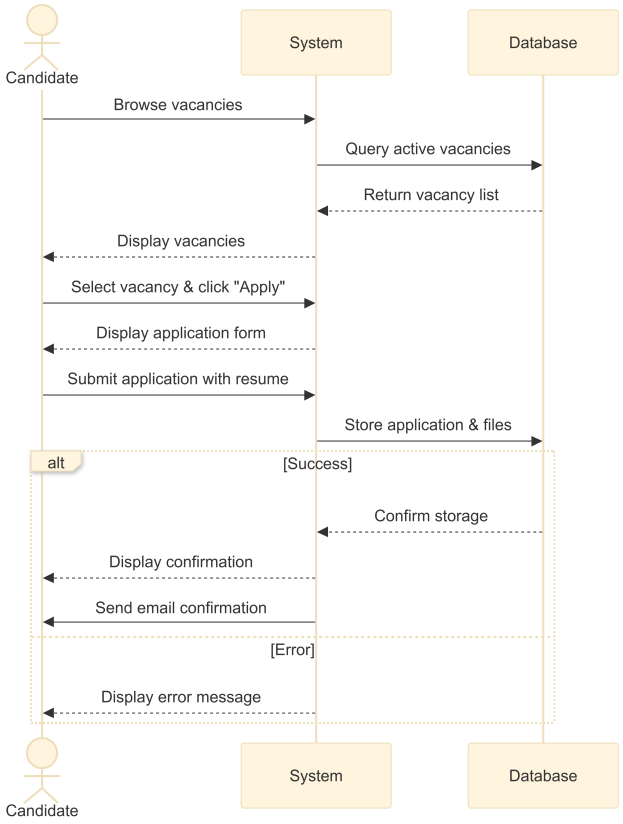

Fig. 3. Use Case 2: Candidate Application Submission

Actor: Candidate.
Goal: Apply for a vacancy.
Preconditions: Job vacancy is available on the platform.

Success scenario:

1. Candidate navigates to the "All Vacancies" page.
2. Candidate selects a job they wish to apply for.
3. Candidate clicks "Apply" and the system asks for a resume and cover letter.
4. Candidate fills out personal information (name, contact, etc.).
5. Candidate answers any custom interview questions provided by the HR Manager.
6. Candidate submits the application.
7. The system acknowledges receipt of the application and sends an automated confirmation message.

Alternative Scenario:
If the application is incomplete (e.g., missing documents), the system prompts the candidate to complete the required fields.
Outcome:
The candidate successfully applies for the job, and their application is saved in the system for HR review.

Use Case 3: Candidate Rejection and Feedback
Actor: HR Manager, Candidate.
Goal: Notify candidates who have been rejected and provide optional feedback.
Preconditions: Notify candidates who have been rejected and provide optional feedback.

Success scenario:

1. HR Manager reviews candidate applications and selects candidates to be rejected.
2. HR Manager provides optional feedback for each rejected candidate (if required).
3. The system sends a polite rejection message to the candidate via email or messaging platforms.
4. If feedback is provided, it is included in the message.
5. The candidate receives the rejection notice and can review the feedback (if available).

Alternative Scenario:
If HR Manager opts not to provide feedback, the rejection message is sent without it.
Outcome:
The candidate is informed of the rejection and, if applicable, receives constructive feedback.

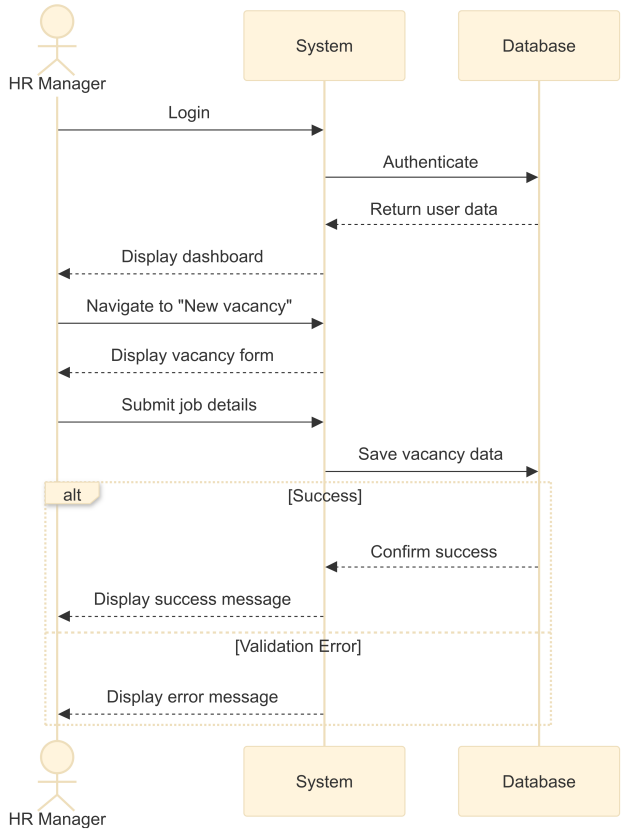

Fig. 4. Use Case 3: Candidate Rejection and Feedback

4.3 Class Diagram

To understand the database schema of the system in detail, the class diagram was created. This diagram includes various entities and relationships between them. As the primary entity, we have “User” with role “manager” who is responsible for creating “Vacancy” entities. Each “Vacancy” may have one associated “Interview” and “User” with role “applicant” may apply to the job and take the interview if it exists. Special entity “InterviewSubmission” contains track results of each candidate and stores them in the database.

Applicants are also involved in the interview stage. Later this “Interview” object as well as applicant’s CV and test results are evaluated by an AI model. These evaluations of resume and interview are stored in “ResumeEvaluation” and “InterviewEvaluation” respectively.

Figure 5 illustrates the high level of the system's database schema, which describes the overall relationship of data. During the development of the project some new entities may be added with additional fields.

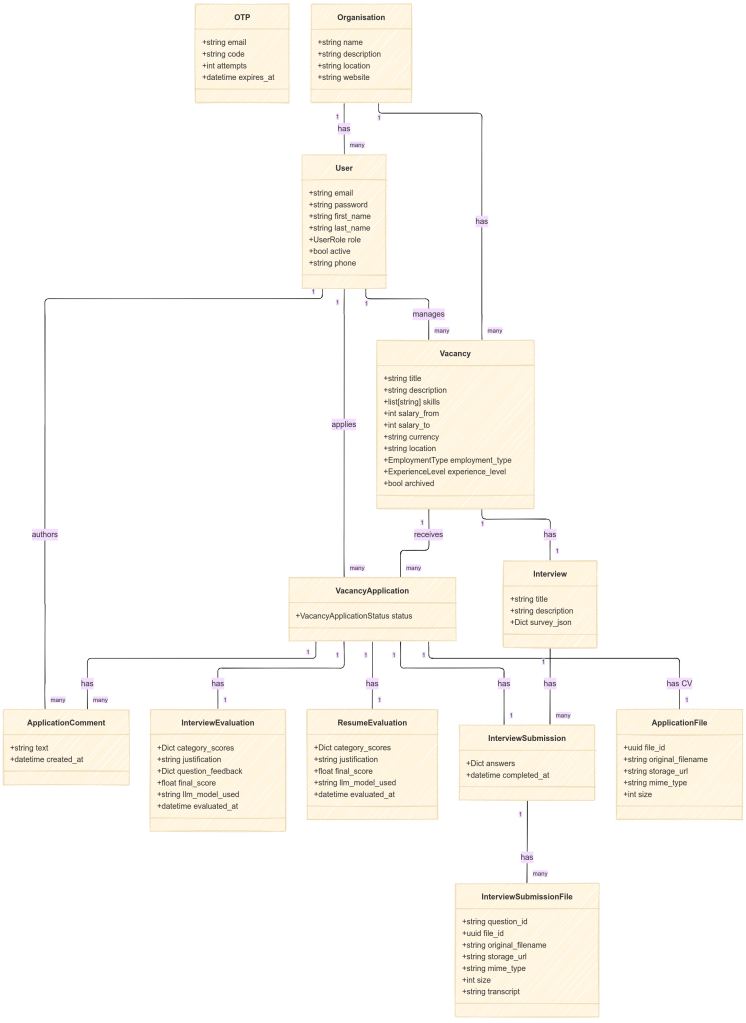

Fig. 5. Class Diagram

5 Project Execution

The development of the project spanned two semesters and underwent significant evolution as the team adapted to technical challenges, new insights, and shifting priorities. The project aimed to create a scalable, intelligent recruitment platform leveraging AI and LLMs to streamline candidate screening and communication for HR teams. As a result, a fully functioning prototype was created.

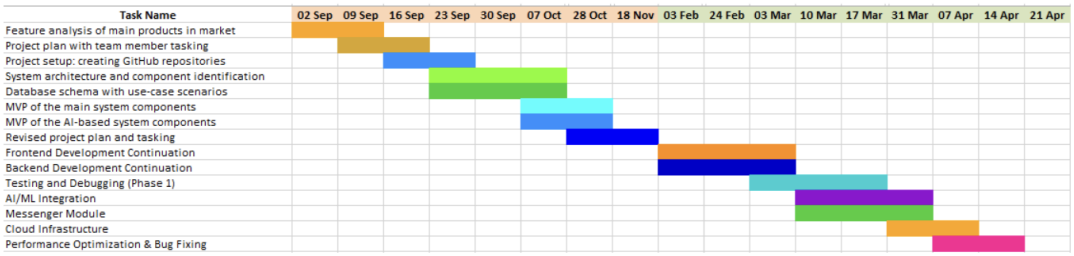

Fig. 6. Timeline of the project

During the **Fall semester**, the team focused on laying the foundational components of the system. The backend infrastructure was established using FastAPI and PostgreSQL, enabling core functionalities such as user authentication, job posting, and application tracking. On the frontend, built with React and Next.js, initial pages were developed for HR managers to post vacancies and manage applications. We implemented an early AI prototype for candidate ranking using a traditional NLP approach with keyword extraction and scoring. However, this method quickly revealed its limitations. In particular, the model struggled to understand context, often assigning high scores to irrelevant resumes simply because they contained overlapping keywords. This issue was illustrated when an empty resume received a surprisingly high score due to superficial keyword matches. Since deeper semantic understanding is a priority for this project, the team decided to transition towards LLM-based analysis in the Spring.

In the **Spring semester**, the team's primary objective was to refine the AI model, expand system functionalities, and prepare the platform for usability testing. A major milestone was the shift from traditional NLP to a more robust, context-aware model leveraging sentence-transformer SBERT all-MiniLM-L6-v2 to generate embeddings of resumes and job descriptions. The embeddings were then compared using cosine similarity to determine candidate-job relevance. Additionally, GPT-4.1 was integrated to provide contextual analysis, such as evaluating nuanced candidate responses and providing interpretability. The transition to LLMs

drastically improved the accuracy and fairness of the ranking system. The semantic embeddings allowed the system to understand job-specific terminology and role responsibilities more effectively than the keyword-based approach. Another significant development was the integration of fOpenAI's Whisper model to transcribe candidates' audio and video interview responses. This feature enabled the platform to process multi-modal data and contribute to a holistic candidate evaluation pipeline.

Alongside AI advancements,system-wide improvements were implemented. The frontend was expanded to include a refined candidate dashboard, ATS enhancements for HR managers (including status tracking and commenting), and improved UX/UI design for intuitiveness, interactiveness and a better experience. On the backend, multimedia support, secure data storage (with AES-256 encryption), and GDPR compliance were completed. The team also enhanced the interview module with audio and text questions, alongside the feedback system.

The team adopted Agile methodology throughout the project lifecycle, utilising **Jira** and **Notion** for task management and Telegram for communication. Tasks were distributed based on each member's expertise:

- **Yerassyl and Dastan** focused on AI/ML model development and integration, including Whisper and GPT-4.1.
- **Aimurat** was responsible for backend development, API integration, and database architecture.
- **Alfiya and Dias** led frontend development, focusing on responsive UI/UX, ATS interface, and real-time frontend-backend communication.

Collaboration was a central part of the workflow. For example, integration testing between AI-generated scores and backend APIs required close coordination between Yerassyl, Dastan, and Aimurat. Similarly, implementing frontend-driven WhatsApp messaging depended on joint efforts between Dias, Alfiya, and Aimurat to ensure smooth data flow and user feedback.

While every member contributed their time and dedication, the team progressed with Aimurat's leadership, who mainly maintained communication within the team and with the supervisor and monitored the deadlines.

6 Evaluation

6.1 Evaluation Approach

We developed a mixed methods approach of quantitative performance of the system together with qualitative analysis of the ability of the system to streamline candidate screening. This allowed us to evaluate the system's ability to fix inefficiencies from the traditional recruitment

processes. And our eval focused on the core functions and AI parts of its kit. To do that, the system had to go through rigorous testing and data collection by the team members to determine how accurate, efficient, and how satisfied with the system they were. In addition to that, we also wanted to get team member's feedback in order to understand how the system impacted the overall, and what could be improved upon if needed.

6.2 Key Evaluation Areas

1. **AI Model Performance:** Accuracy and reliability of the AI model leveraging resume, test results, interview responses to rank candidates was evaluated. It consisted of casting the model to see how good it would be at making top tier candidates and also how few false positives and negatives it would generate.
2. **System Efficiency:** We measured the effectiveness of automating the screening of candidates, management of applications as well as communication with them. They included the amount of time saved during the candidate screening, reduction in manual HR tasks, and time saved communicating with applicants.
3. **User Experience:** It was about evaluating the usability and intuitiveness of the TalentEngine platform for the both parts of HR managers and for the candidates. In this I test how common things the user would do in our app, for instance.

6.3 Data Collection and Analysis

As we went for a thorough evaluation, we collected both quantitative as well as qualitative data:

- **Quantitative Measures:**
  - **AI Model Accuracy:** To measure the precision and recall of the rankings produced by the AI model, we compared the AI model's top candidate selection to those produced by human resources professionals, upon 100 sample applications. In our calculation, we determined which of the three percent of candidates correctly classified in the first 5, 10, or 20 ranks.
  - **Screening Time Efficiency:** We examined how much time average HR managers spent on screening 50 applications through the TalentEngine system vs compared to a manual screening process. On resume review, test evaluation and interview scheduling stages, we measured time involved.
  - **System Response Time:** Second, we measured the average response time of the system for key operations on candidate profiles, application submission, and processing of interview feedback. To do so we used load testing tools to simulate concurrent users.

- **Qualitative Measures:**
  - **User Feedback:** We conducted semi structured interviews with 5 HR managers and 10 candidates to gauge all feedback regarding TalentEngine platform. Ease of use, interface intuitiveness, and satisfaction with the system's features was what was discussed in the interviews.
  - **Questionnaires:** To gather data on the specific aspects, we distributed questionnaires to 15 HR managers and 25 candidates to capture their experience of clarity of information, navigation effectiveness and perceived usefulness of AI powered features within the system. Agreement levels were measured using a Likert scale (1-5) in the questionnaire used.

6.4 Results and Analysis

- **AI Model Accuracy:** The precision of which the top-tier candidates can be correctly identified by the top 10 ranks was 85%. The shift from keyword-based systems represents a major progress in processing information. Using the recall rate of 92%, it is validated that the model has successfully captured a majority of the qualified candidates.
- **Screening Time Efficiency:** HR managers were able to reduce the average screening time by 60 per cent and reduced the average screening time for each application down from 40 minutes to 15 minutes by using TalentEngine. This is quite an amount of time saved.
- **System Response Time:** Under 2 seconds, the system average response time for the key operations met the performance requirement of 5 seconds or less.

6.5 Validation of Computing-Based Solution

The evaluation data overwhelmingly confirms that TalentEngine is an effective technology advancement to the development of recruitment processes for the modern world based on computing. AI powered features enhanced the efficiency and accuracy of candidate screening much better than the previous conventional processes and helped to make the user experience both for HRs and Candidate engaging through a simple and user friendly interface. One of the ways that system can help fix this problem is by the automation of certain crucial tasks, such as application tracking, interview scheduling and communication with the candidates yet without losing the effectiveness. The user feedback and the quantitative data of time saved and accuracy show that TalentEngine satisfies the design, implementation and evaluation of a computing based solution.

7 Conclusion and possible future work

Over the course of two semesters, the development of TalentEngine resulted in the creation of a functional, AI-enhanced recruitment platform that leverages cutting-edge technologies to streamline the hiring process for both recruiters and candidates. By integrating backend infrastructure, a user-friendly frontend, and a powerful AI pipeline, the team successfully delivered a system capable of automating candidate evaluation, enabling seamless communication, and supporting dynamic interview formats.

Key Findings and Contributions

- **AI-Driven Screening Enhancement:** The transition from basic keyword-based semantic analysis to Large Language Models (LLMs) significantly improved the system's ability to understand the context and nuances of candidate applications. By incorporating SBERT [8] for semantic embeddings and OpenAI's GPT-4.1 [6] for contextual evaluation, the model achieved a more accurate and meaningful ranking of candidates.
- **Automated Evaluation:** Doing so enabled the creation of such an automated evaluation of candidate responses across various formats, and therefore made the screening even more robust.
- **Collaborative System Development:** The project was able to explain why collaborative system development gives a high value over agile development and teamwork, because responsibility was divided clearly and the solution searching and implementing was effectively reached.
- **Secure and Scalable Architecture:** Deployment on the VPS with security best practices (AES 256, GDPR compliance, 2FA and SSL) ensured that is a secured and scalable architecture meeting industry standard on the issue of data privacy and system performance.

Future Enhancements and Areas for Improvement

Although the first version of TalentEngine does what it was made to do it, there are several places where story continues.

- **AI Model Improvement:** Through continuous tuning using real data and feedback from HR for further improving the AI model's ranking accuracy, reducing AI model bias.
- **Data Visualization and Dashboards:** Adding industry average datasets, dashboards and analytical tools for recruiters to first drill down on their candidate pipelines and then conversion rates and performance metrics are great to new insights and better hiring decisions.

- **Additional Messaging Platforms Integration:** Integration with messaging channels like email, Telegram, or Slack could come in handy in widening the communication channels and adjusting to different business demands.
- **Language Support:** To enable multiple regions, language support must be implemented on the platform which will ultimately be usable anywhere. This would be beneficial to international use, say in an application involving multilingual hiring.

Innovation from the technical as well as practical standpoints in recruiting's modernisation through AI; this is TalentEngine. The platform combines automation with a human-centric approach to avoid that efficiency at the cost of fairness. Not only did the project meet its academic goals, it also prepared a basis for a scalable product that could change the way hiring is actualized in the real life.

References

1. AI-First, People-Centric recruiting Platform | HireEZ, https://hireez.com/
2. HeroHunt.ai, https://www.herohunt.ai/
3. AI, X.: X0PA AI | AI-First Talent Recruiting Software (4 2025), https://x0pa.com/
4. OpenAI: OpenAI's o3-mini (1 2022), https://openai.com/index/openai-o3-mini/
5. OpenAI: OpenAI's Whisper (9 2022), https://openai.com/index/whisper/
6. OpenAI: OpenAI's GPT 4.1 (4 2025), https://openai.com/index/gpt-4-1/
7. Qureos: Qureos | Your AI-Powered Job Coach - Iris, https://www.qureos.com/
8. Transformers, S.: sentence-transformers/all-MiniLM-L6-v2 · Hugging Face (1 2024), https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2