TITLE OF THE PROJECT

E-recruitment System

Course Code: CIS499

Fall: 2023

Submitted By

Ishrat Jahan Esha

ID: 201-16-515

Department of Computing and Information System (CIS)

DAFFODIL INTERNATIONAL UNIVERSITY

Supervised By

Mr. Md. Mehedi Hassan

Lecturer

Department of Computing and Information System (CIS)

DAFFODIL INTERNATIONAL UNIVERSITY

Date of Submission: 23 January 2024

PROJECT BASED LEARNING

Declaration

I hereby declare that; this project has been done by me under supervision of **Md Mehedi Hassan Lecturer** department of Computing and Information System (CIS) of Daffodil International University. I am also declaring that this project or any part of there has never been submitted anywhere else for the award of any educational degree like, B.Sc., M.Sc., Diploma or other qualifications.

Supervised By

23.07.2024

Mr. Md. Mehedi Hassan
Lecturer
Department of CIS
Daffodil International University

Submitted By

Ishrat

Name: Ishrat Jahan Esha
ID: 201-16-515
Department of CIS
Daffodil International University

APPROVAL

This Project titled " E-recruitment System", Submitted by Ishrat Jahan Esha, ID No: 201-16-515 to the Department of Computing & Information Systems, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computing & Information Systems and approved as to its style and contents. The presentation has been held on 23-01-2024.

BOARD OF EXAMINERS

23/01/24

Md Sarwar Hossain Mollah — **Chairman**
Associate Professor and Head
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Dr. Shekh Abdullah-Al-Musa Ahmed — **Internal Examiner**
Assistant Professor
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Md. Nasimul Kader — **Internal Examiner**
Assistant Professor
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Prof. Mohammad Shorif Uddin, — **External Examiner**
Professor
Department of Computer Science and Engineering
Jahangirnagar University

Acknowledgment

I want to start by saying how grateful I am to Almighty Allah for giving me this chance. Without the Almighty's help, I would be unable to complete this project. Finally, I want to thank DCL for choosing me as an intern trainee since this internship program has taught me a lot. It is believed that a teacher is a student's second parent since they provide them with the proper guidance, allowing them to progress in life. In my internship program, I had two professors who always advised me in making the best decisions and provided me with advice on how to successfully perform challenging tasks. They never allowed me to struggle alone and were constantly reducing my stress by providing me with mental fortitude. One of them is Md. Mazharul Islam Masum is my intern trainer and Mr. Md. Mehedi Hassan is supervising my academic work.

Additionally, I am appreciative of them for their kindness, readiness to provide a help, and faith in me. I want to thank them for helping me, for their instruction, for teaching me how to work through any challenges, and for teaching me how to handle a crucial project in the future.

Abstract

The E-recruitment Project aims to revolutionize the talent acquisition and management processes within an organization through the development and implementation of an advanced Human Resource Management System (HRMS) focused on recruitment. This comprehensive system integrates cutting-edge technologies and robust functionalities to streamline the end-to-end recruitment lifecycle, from job requisition to candidate onboarding.

Table of Contents

List of Figure

List of Table

Chapter 1– Introduction

Recruitment-Centric HRMS, also known as HRMS centered on Recruiting, is a Human Resource Management System that prioritizes the recruitment process in both its functionality and design. A recruiting-focused HRMS places a higher priority on attracting, assessing, and employing top people than typical HRMS systems, which handle different HR responsibilities including payroll, benefits, and performance management.

Chapter 2 – Initial Study

2.1 Project Proposal

Project Proposal: E-recruitment Software

2.2. Background

The talent acquisition landscape in modern business is competitive. A data-driven, engaging, and effective recruitment process is necessary to draw in and hire top talent. Regrettably, a lot of businesses still use antiquated techniques, manual processes, and disorganized tools, which results in inefficiencies, annoyance, and poor hiring choices. For businesses of all sizes, making the strategic decision to invest in a strong E-recruitment software will pay off well. Through the automation of chores, optimization of workflows, and provision of insightful data, this software will help businesses recruit and retain top personnel, enhance their ability to make decisions, and obtain a competitive advantage in the market. Before the creation of the HRMS, HR staff had to manage HR procedures across five separate systems, requiring users to get familiar with and log into numerous pieces of software in order to complete jobs. This was intended to be avoided by HRMS, which integrated the required applications into software modules. Every software module is a representation of the earlier HR systems. The fact that the different systems were designed and coded consistently to provide the same user experience across all modules is an advantage of integrating them into HRMS. Through the integration of several modules, HRMS functions as a comprehensive one-stop solution for all HR procedures. Furthermore, because HRMS is scalable, ITS can continue to add and modify modules as needed in the future.

2.3 Problem Area:

- Manual procedures: Spreadsheets, unconnected systems, and paper-based applications all lead to errors, lost data, and laborious activities
- Bad interviewing experience Candidates become frustrated and disengaged from sluggish communication, opaque application processes, and opaque hiring practices.
- Limited data insights: It is challenging to monitor hiring metrics, evaluate talent pools, and determine the best hiring practices in the absence of consolidated data.
- Risk of noncompliance: The risk of breaking labor rules and regulations is increased by inconsistent procedures and fragmented data.

2.4 Proposed Solution:

E-recruitment Software

We propose the development of a comprehensive HRMS recruitment software specifically designed to address these challenges. This software will offer a centralized platform for managing all aspects of the recruitment process, from job posting to onboarding.

Benefits and Impact of proposed solution:

The proposed E-recruitment software will:

- Boost productivity and efficiency by streamlining processes, automating operations, and lowering the administrative load.
- Improve the applicant experience to draw in and keep top talent, offer an easy-to-use and transparent hiring procedure.
- Boost the selection process for hiring: Get insights from data to find the top applicants and make well-informed recruiting decisions.
- Cut expenses by optimizing hiring practices, streamlining procedures, and lowering compliance risks.
- Increase worker satisfaction by: Provide a satisfying onboarding process and provide staff members the authority to handle their own data.

Chapter 3 – Literature Review

3.1 Technology and Literature Review

E-recruitment systems play a pivotal role in modern talent acquisition by leveraging digital technologies to streamline the hiring process. This review explores the key technologies and existing literature in the realm of E-recruitment systems.

3.1.1 Technologies in E-recruitment:

- **Artificial Intelligence (AI):**

- Literature: Research suggests that AI, particularly machine learning algorithms, enhances candidate matching, automates resume screening, and predicts candidate success.
- Technology: Natural Language Processing (NLP) for resume parsing, AI-driven catboats for candidate engagement, and machine learning models for predictive analytics.

- **Social Media Integration:**

- Literature: Studies indicate that integrating social media into recruitment processes increases the reach of job postings and positively impacts employer branding.
- Technology: APIs for seamless integration with platforms like LinkedIn, Twitter, and Facebook, facilitating job posting and candidate sourcing.

- **Mobile Recruitment:**

- Literature: Mobile recruitment improves accessibility, allowing candidates to apply on-the-go, leading to a broader and more diverse applicant pool.
- Technology: Responsive design for mobile-friendly application interfaces, mobile application development, and SMS notifications.

- **Big Data Analytics:**

- Literature:* Big Data analytics in recruitment aids in predicting talent trends, improving decision-making, and optimizing recruitment strategies.
- Technology: Data warehousing, data mining, and analytics tools for processing and analyzing large datasets from various sources.

- **Video Interviewing:**

- Literature: Research indicates that video interviews offer flexibility, reduce time-to-hire, and provide a more comprehensive view of candidates.
- Technology: Video conferencing tools, asynchronous video interview platforms, and facial recognition for candidate sentiment analysis.

3.1.2 Literature Review:

- **Candidate Experience in E-recruitment:**

- Studies emphasize the importance of a positive candidate experience in E-recruitment, impacting employer reputation and attracting top talent.

- **Security and Privacy Concerns:**

- Literature highlights the need for robust security measures to protect sensitive candidate information and maintain compliance with data protection regulations.

- **Adoption Challenges in E-recruitment:**

- Research identifies challenges in the adoption of E-recruitment systems, including resistance from traditional hiring practices and the need for effective change management.

- **Impact on Diversity and Inclusion:**

- Scholars discuss how E-recruitment systems can either promote or hinder diversity and inclusion, emphasizing the importance of fair algorithms and inclusive practices.

3.2 Best Features:

- Applicant Tracking System (ATS): Manage job postings, receive and screen applications, and track candidates through the pipeline.
- Talent Pool Management: Build and maintain a database of qualified candidates for future opportunities.
- Career Portal: Create a user-friendly interface for candidates to submit applications, track their progress, and receive updates.
- Interview Scheduling and Management: Streamline interview scheduling, assign interviewers, and collect feedback seamlessly.
- Onboarding and Offer Management: Automate offer letter generation, onboarding paperwork, and new hire welcome processes.
- Reporting and Analytics: Gain valuable insights into recruitment performance with data on time-to-hire, source of hire, cost-per-hire, and other metrics.
- Compliance Management: Ensure adherence to relevant labor laws and regulations with built-in compliance features.

3.3 Limitations of E-Recruitment Software:

While HRMS recruitment software offers numerous benefits, it's important to acknowledge its limitations and consider approaches to mitigate them:

- Dependence on technology: Errors or defects in software may cause the hiring process to be interrupted. Make sure you have regular updates, dependable hosting, and data backups.
- Automation and bias: Certain candidates may be unfairly disadvantaged by algorithmic bias in applicant tracking systems. Prioritize human judgment when making final selections, evaluate algorithms for bias, and assemble diverse recruiting teams.
- Lack of human touch: An excessive dependence on automation may result in an impersonal and sterile application process. Keep in touch with each other personally via phone conversations, emails, and video interviews.
- Data security and privacy: Robust security measures are necessary for sensitive candidate data. Put in place access controls, data encryption, and frequent security audits.
- Lack of customization: The degree of customization offered by software solutions varies. Select a system that can adjust to your unique requirements and work processes.
- Continued expenses: After the first purchase, software 5. Lack of customization: The degree of customization offered by software solutions varies. Select a system that can adjust to your unique requirements and work processes.

3.4 Recommended Approach:

- A balanced approach combines the effectiveness of software with the personal touch of face-to-face meetings and tailored correspondence.
- Data-driven decision making: Take advantage of data insights to make well-informed hiring decisions while keeping personal preferences and biases in mind.
- Continuous improvement entails tracking the program's effects, routinely getting user input, and modifying your strategy in response to information and input.
- Invest in training: Give HR personnel and workers thorough instruction on how to use the program in an ethical and successful manner.
- Security focus: Make data security a top priority by putting strong safeguards in place and keeping up of compliance requirements.
- Transparency in the hiring process should be upheld, and any worries candidates may have regarding the use of technology should be addressed.

- Frequent assessment: Assess the software's influence on hiring results, employee satisfaction, and recruitment efficiency on a regular basis.

May optimize the advantages of HRMS recruitment software and reduce any potential disadvantages by being aware of its limitations and adopting a reasonable, data-driven strategy.
Recall that software is a tool, not a substitute for human discretion and compassion. It can be strategically used to improve candidate experiences and your organization's recruitment efforts.
I hope this gives you a thorough understanding of the drawbacks and suggested strategies for HRMS hiring software.

Chapter 4 – Methodology for E-recruitment Software

Methodology for HRMS Recruitment Software
Now that we know this chapter focuses on an HRMS recruitment software.

4.1 What to Use:

- **Technology Stack:**

  The choice of technology stack is critical for the successful implementation of HRMS recruitment software. Considerations should include database management systems, programming languages, frameworks, and any third-party integrations.

i. Database: MySQL
ii. Programming Language: Java, JavaScript
iii. Frameworks: Spring Boot (Java).
iv. Library: React Js

- **Development Tools:**

  Select development tools that facilitate collaboration, version control, and project management. Common tools include Get for version control, Jiri for project management, and collaborative platforms like Slack or Microsoft Teams.

4.2 Why to Use:

- **Scalability:**

  The HRMS recruitment software's scalability requirements should be supported by the technological stack that is selected. The system should be able to accommodate more data, users, and transactions as the company expands.

- **Security:**

  When managing sensitive HR data, security is crucial. Data encryption, secure authentication, and access control are examples of industry-standard security measures that should be followed by the selected technology and development processes.

- **Integration Capabilities:**

  The technologies' ability to be integrated should be taken into account. To guarantee a smooth workflow, the HRMS software should interface with databases, third-party apps, and current HR systems with ease.

4.3 Sections of Methodology:

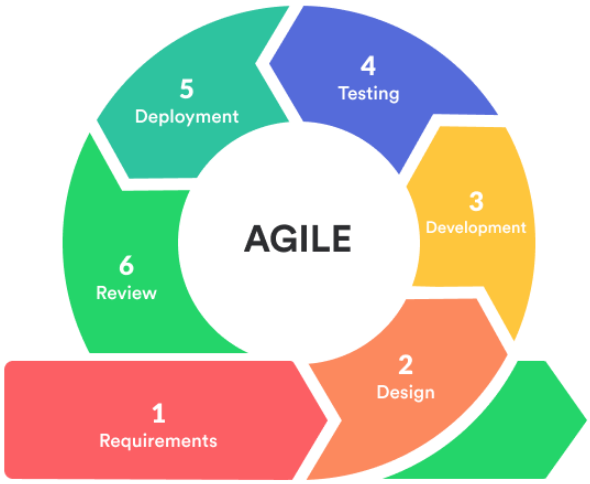
- **System Requirements Analysis:**
  Conduct a thorough analysis of HRMS requirements, involving HR professionals, recruiters, and IT specialists. Define functional and non-functional requirements to guide the development process.

- **Technology Selection:**
  Justify the selection of the chosen technology stack, providing a rationale for each component. Consider factors such as cost, ease of maintenance, community support, and alignment with organizational goals.

- **Prototyping and User Feedback:**
  Develop prototypes or wireframes to visualize key features of the HRMS recruitment software. Gather feedback from end-users to ensure that the system meets their needs and expectations.

- **Agile Development Methodology:**

Adopt an Agile development methodology to ensure flexibility and responsiveness to changing requirements. Break down the development process into iterative sprints with regular feedback loops.



- **Testing and Quality Assurance:**
  Put into practice a thorough testing plan that includes user acceptability, integration, and unit testing. Before deploying the program, make sure it satisfies quality criteria and is free of serious problems.

4.4. Implementation Plans:

- **Phased Rollout:**
  Plan for a phased rollout of the HRMS recruitment software. Identify key milestones and prioritize the release of features based on their criticality and impact on daily operations.

- **Training Programs:**
  Develop training programs for HR professionals, recruiters, and end-users. Conduct workshops and provide documentation to ensure a smooth transition to the new system.

- **User Support and Feedback:**
  Establish a user support system to address queries and issues promptly. Encourage continuous feedback from users to identify areas for improvement and future enhancements.
- **Data Migration Strategy:**
  Implement a data migration strategy to seamlessly transfer existing HR data to the new system. Ensure data integrity and accuracy during the migration process.

- **Monitoring and Optimization:**
  Implement monitoring tools to track system performance, user engagement, and potential issues. Continuously optimize the system based on user feedback and evolving business needs.

i. Research Design:

- Exploratory/Descriptive: If you're evaluating existing software or user needs, consider surveys, interviews, and focus groups.
- Experimental/Quasi-Experimental: If you're testing specific features or comparing software, consider user testing experiments with A/B testing or similar methods.

ii. Data Collection Methods:

- Quantitative: Surveys with rating scales, observation of usage metrics, and analysis of existing recruitment data.
- Qualitative: Interviews with HR professionals, candidates, and stakeholders, focus groups for in-depth feedback, user experience testing.

iii. Data Analysis Techniques:

- Quantitative: Statistical analysis of survey data, usage metrics analysis, cost-benefit analysis.

Why to Use It:

- Align with Research Questions: Are you focusing on user experience, functionality, efficiency, or cost-effectiveness? Choose methods that answer those questions.
- Suitability for Data: Consider the type of data you need to collect. Surveys for quantitative data, interviews for rich qualitative insights.
- Validity and Reliability: Ensure your methods measure what they are supposed to and produce consistent results. Pilot testing can be helpful.

Chapter 5 – Planning

5.1 Project Plan

Work Breakdown Structure (WBS)

Level 1

- Planning and Requirements Gathering
- System Design and Development
- Testing and Deployment
- Training and Support
- Maintenance and Updates

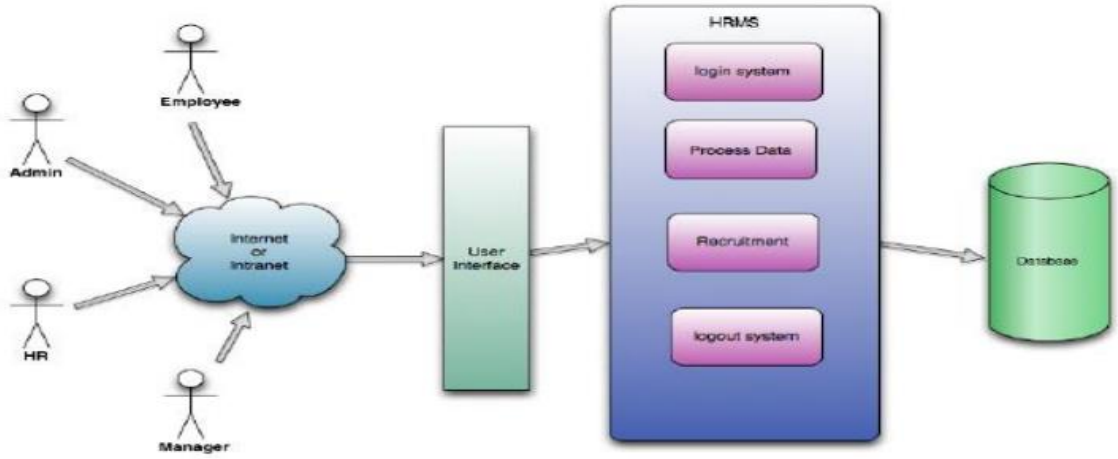
Level 2 (Planning and Requirements Gathering)

- Conduct needs assessment
- Gather user requirements
- Define system scope and objectives
- Develop project plan and budget
- Select HRMS vendor (if applicable)

Overview: Refers to the systems development life cycle a set of processes or stages and all stages of the system where a number of steps and the main stage falls below .all the steps and methodologies are:

- **Plan**
- **Analysis**
- **Design**
- **Implementation**

HRMS architecture: HRMS which is an online intranet system will be used by four types of employees. These types who have different roles can be stated as; admin, manager, hr, employee. Every user enters the main authentication page and after that, system will grant them authorization. After being authorized according to their permissions (role type) users will basically query and edit the database via HRMS.



Description of procedures and functions: This section will explain the major functions of HRMS along with the data flow. So the major functionality of the project such as authentication mechanism, personal data processing, recruitment, and will explained step by step.

Authentication

- **Login user:** can login to the HRMS system with his/her username and password.
- **Logout user:** can logout from the HRMS system.
- **Login failure:** if the user does not exist in the database or the user did not get authorized by the HRMS admin yet.

Authorization

- User role check: after logging in, the user role will be checked from the database and the user interface will be created according to that role/roles.

Process data

- **Display:** Users with certain roles are able to see the database's contents. To be more precise, the employee's only viewable information is personal data. The manager has access to both their personal data and the data of the workers who are covered by them. Both admin and HR have the ability to view their own data as well as that of every employee.

- **Edit:** An employment role holder has the ability to amend their personal data. With the exception of user job type, the manager may only update employee personal information that is under his or her purview. All employee data, with the exception of user role type, is editable by HR. The administrator can change every detail pertaining to every employee, including their user role type.

.

- **Search:** A user with the manager role type can search the database's contents for the workers that fall within their purview. Roles in HR and administration have access to the database's complete employee database. The search function focuses on particular keywords to provide details about an employee's traits, abilities, capabilities, etc.

- **Report:** Essentially, this feature is used to filter the search mechanism's contents. As we indicated in the search feature above, for example. The HR department is looking for a list of specific employees who are proficient in PHP. After obtaining the list of employees through the search tool, the user can pick the appropriate checkbox for each individual to obtain a specific report. Alternatively, by checking the box, a management role type can receive a report listing any or all of the workers who fall under his or her purview. All other role kinds, such as admin, hr, and manager, can use this capability, with the exception of the employee role type.

- **Update authentication:** Only the admin role type is able to use this feature. A specific user's role type can be updated by the admin. For instance, if a worker receives a promotion, his position type may shift from worker to manager. The administrator has the ability to modify this authentication system.

Recruitment

- **Add a new vacation:** employee is able to add a new vacation to the database. The employee will have all the required personal information related to his/her and his/her vacations data. The new created vacation will have an id.

- **Add a new training:** after being created employees by hr. role, HR role is responsible for creating a new training by the specified id assigned in employee feature.

5.2 Resource Allocation

Study of Current System

Before Crest HRMS, Company was using third party's HRMS.Which was not feasible and was costly too. It was difficult to change existing HRMS.

User Characteristics

- The users are the normal employees who want to check their leave balance, profile, and other employee's general information. Admin is a special user who has extra rights like accept or reject leave, bulk change functionality, can show all details of any user.

Hardware and Software Requirements

Hardware Requirements

- Dual Core 5 GHZ or latter CPU
- 8 GB RAM
- 228 GB storage minimum

Software Requirements

- Chrome Latest (version: 47.0.2526 or above)
- Intellij IDE
- Visual Code Studio
- MySQL
- Time Duration / Time Boxing

Table Milestones and Deliverables:

PHASE	DELIVERABLES	PURPOSE
System Requirement and Analysis	• Requirement Gathering and analysis. • Functional Specification • Non-functional Specification	It gives an exact understanding of the user's requirements.
System Design	• Use case Diagram • Activity Diagram • Data Flow Diagram	It gives the logical structure that describes the system.
Implementation and Testing	• The output obtained for the required functionality after implementing and doing various types of testing.	It makes the system robust and reliable.

Table 1 Requirement

5.3 Testing Plan

System testing aims to verify that each program operates as planned, that the programs integrate to fulfill the requirements, and that the computer system functions in tandem with related clerical and other activities. Systems are tested as individual systems; they are not designed as complete systems. Testing of the system and units must be done by the analyst.

There are various kinds of testing techniques accessible. We have tested our system in a number of ways, such as if the application achieves the objectives for which it was created. Given that the program was intended to be used across a wide network, this was a crucial question that we had to answer.

To achieve its objective of being able to run. System testing aims to verify that each program operates as planned, that the programs integrate to fulfill the requirements, and that the computer system functions in tandem with related clerical and other activities.

5.4 Risk Management:

Risk Identification:

- **Technology Risks:**
  Identifying potential issues with technology integration, system compatibility, or unforeseen technical challenges during development and deployment.

- **Data Security Risks:**
  Recognizing the possibility of data breaches, unauthorized access, or data loss, especially considering the sensitive nature of HR information.

- **Resource Constraints:**
  Identifying risks related to insufficient manpower, expertise, or time, which could impact project timelines and deliverables.

Impact vs. Likelihood Matrix:

- Assessing the potential impact of identified risks on the project objectives against the likelihood of their occurrence.

Risk Severity Levels:

- Categorizing risks into low, medium, and high severity levels based on their potential impact on project success.

Risk Precaution / Action Plan:

- Technology Risks:
  i. Precaution: Regularly update technology stacks and conduct thorough compatibility testing.
  ii. Action Plan: Maintain a contingency fund for potential technology upgrades or unforeseen technical challenges.

Data Security Risks:

- Precaution: Implement robust encryption, access controls, and conduct regular security audits.
- Action Plan: Develop and rehearse an incident response plan to address any security breaches promptly.

Resource Constraints:

- Precaution: Continuously monitor resource allocation and adjust project timelines accordingly.

- Action Plan: Cross-train team members to mitigate the impact of key resource dependencies.

Steps Taken for Possible Risks:

- Regular Risk Reviews:
- Conducting regular risk reviews throughout the project lifecycle to identify new risks and reassess existing ones.

Prototyping and Testing:

- Utilizing prototyping and testing phases to identify and rectify potential issues before full-scale implementation.

Change Management:

Factors That Might Cause Change:

i. **Regulatory Changes:**
   - Changes in labor laws, data protection regulations, or other legal requirements impacting HR processes.

ii. **Organizational Restructuring:**
   - Changes in the organizational structure, mergers, acquisitions, or reorganizations affecting HR workflows.

DSDM Atern Welcomes Change:

i. **Agile Mindset:**
   - Embracing change as an inherent part of the development process, accommodating alterations to requirements.

ii. **Incremental Development:**
   - Implementing changes incrementally to allow for continuous improvement and flexibility.

Considering Business Value / Priority:

i. **Impact Assessment:**
   - Evaluating the business value and priority of proposed changes based on their potential impact on organizational goals.

Change Workshop:

i. Stakeholder Collaboration:
   - Conducting change workshops involving key stakeholders to gather input, assess implications, and ensure alignment with organizational objectives.

Changes That Are Allowed:

i. Scope Adjustments:
- Allowing changes within predefined project scope boundaries, ensuring flexibility without compromising project objectives.

Key Decision Takers of Change:

i. Change Control Board (CCB):
- Establishing a CCB with representatives from key stakeholders, project management, and subject matter experts to evaluate and approve changes.

5.5 Quality Management:

Rules Applied to Maintain Quality:

i. Adherence to Standards:
- Strict adherence to industry standards, coding practices, and project management methodologies.

DSDM Atern Standard Quality Measures:

i. Iterative Testing:
- Implementing iterative testing cycles, including unit testing, integration testing, and user acceptance testing throughout development.

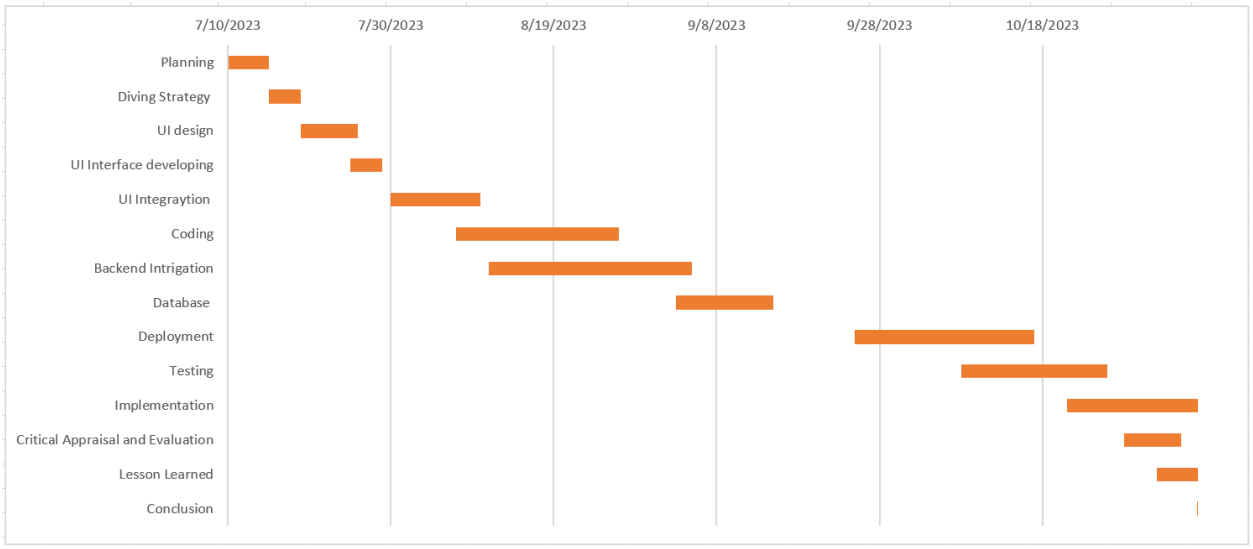
Quality Plan and Measuring Meter:

i. Quality Metrics:
- To guarantee continual progress, define and monitor quality measures like as defect density, test coverage, and customer satisfaction.

Embracing change, retaining high standards, and managing risks effectively are essential elements of developing an HRMS recruitment system. The project guarantees smooth progress by proactive risk identification, assessment, and mitigation. Accepting change within the agile framework promotes adaptation and flexibility, which guarantees that the system is in line with changing corporate requirements. The implementation of continuous testing and standardization in quality

management techniques serves as the foundation for a solid and dependable HRMS. When combined, these tactics aid in the successful creation and implementation of an effective and efficient HRMS hiring system.

Gantt chart



Chapter 6 – Feasibility

All possible type of feasibility

6.1 All type of Feasibility

The feasibility of software can be tested in four dimensions:

Technical Feasibility

Since the project uses reliable tools like open source technology like react, node.js, python and Mongo DB, the system can be implemented efficiently without any issues. The trio of this technology can efficiently handle data, requests and also create user friendly applications. Hence this project has a good technical feasibility.

Timetable Viability

The project that needs to be built consists of four distinct modules, making it laborious to handle. However, since there are enough human resources available, it should be possible to develop the application in the allocated amount of time. Additionally, the time will adjust in accordance with any changes in requirements.

Practicality of Operation

This phase raises questions about who will utilize the initiative and how it will operate. We must investigate the issues with the current system to determine whether they warrant fixing. The user's time and effort in analyzing employee data will be greatly reduced by this project, which will also offer more capabilities. It is therefore operationally possible.

Implementation Feasibility

The requirements mentioned above can be fulfilled using various technologies available. React-.js, Spring Boot and SQL, the implementation of the project is feasible.

6.2 Cost-Benefit Analysis (CBA):

Cost Analysis:

6.2.1 Development Costs:

- Expenses related to software development, including salaries, tools, and technologies.

Recruitment Module

- Job creation
- Profile creation
- Job posting on various platforms and websites
- Interview scheduling
- Candidate assessment
- Sharing CVs

With the integration of these features, the development cost of the recruitment module would be around 26,000 USD to 29,000 USD.

6.2.2 Training Costs:

- Costs associated with training HR professionals and other users on the new HRMS software.

6.2.3 Integration Costs:

- Expenses related to integrating the HRMS software with existing systems and databases.

6.2.4 Maintenance Costs:

- Ongoing expenses for system maintenance, updates, and support.

Benefit Analysis:

- **Efficiency Gains:**

- Faster recruitment processes leading to reduced time-to-hire and increased productivity.

- **Improved Candidate Quality:**

- Enhanced tools for candidate assessment and matching leading to better hires.

- **Resource Optimization:**

- Efficient allocation of HR resources, reducing manual and repetitive tasks.

- **Data-Driven Decision-Making:**

- **Improved analytics and reporting for better strategic decision-making.**

- **Enhanced User Experience:**

- Positive impact on user satisfaction and engagement.

- **Compliance and Risk Mitigation:**

- Reduced risks related to compliance issues through automated tracking and reporting.

The cost-benefit analysis aims to quantify the financial impact of implementing the HRMS recruitment software. The benefits, such as efficiency gains and improved decision-making, should ideally outweigh the costs, making the project financially viable.

6.3 Dynamic Systems Development Method (DSDM) - Good or Not:

Positive Aspects:

- **Agile Approach:**
  DSDM follows an agile methodology, allowing for iterative development and adaptability to changing requirements, which is beneficial for a project like HRMS recruitment software.
- **User Involvement:**
  DSDM encourages active involvement of end-users throughout the development process, ensuring that the final product aligns closely with user needs.
- **Incremental Delivery:**
  DSDM supports incremental delivery, allowing for the gradual implementation of features. This aligns well with the modular nature of HRMS software.
- **Business Value Focus:**
  DSDM emphasizes delivering business value early and continuously. This aligns with the goal of achieving tangible benefits quickly in HRMS software.

Considerations:

- **Resource Intensive:**
  DSDM requires active and continuous involvement from users and stakeholders, which can be resource-intensive for busy HR professionals.
- **Adaptation Challenges:**
  If the organization is not accustomed to agile practices, there might be challenges in adapting to the DSDM approach.
- **Change Management:**
  DSDM welcomes changes, and managing change effectively is crucial. This may require a robust change management strategy.

6.4 Project Approach Questionnaire (PAQ):

6.4.1 Project Scope:

- Does the HRMS recruitment software align with the overall organizational goals and objectives?

6.4.2 Stakeholder Engagement:

- Have key stakeholders been identified, and is there a plan for continuous engagement throughout the project?

6.4.3 Resource Availability:

- Are the necessary resources, including manpower and technology, available for successful project implementation?

6.4.4 Risk Assessment:

- Have potential risks, such as data security and system integration challenges, been thoroughly assessed?

6.4.5 User Training:

- Is there a well-defined plan for training HR professionals and other users on the new HRMS software?

6.4.6 Compliance Considerations:

- Have legal and compliance aspects been considered, and is the software designed to meet relevant regulations?

6.4.7 Feedback Mechanism:

- Is there a mechanism in place for gathering user feedback during and after the software development process?

6.4.8 Implementation Timeline:

- Is there a realistic timeline for the implementation of the HRMS recruitment software, considering project scope and resources?

The PAQ helps assess the readiness and viability of the project, considering various aspects such as scope, stakeholder engagement, resource availability, and risk management. The responses to these questions provide insights into potential challenges and areas that may require additional attention during the project lifecycle.

Chapter 7 – Foundation

7.1 Problem Area Identification:

- **Manual procedures:** Spreadsheets, unconnected systems, and paper-based applications all lead to errors, lost data, and laborious activities. Bad interviewing experience Candidates become frustrated and disengaged from sluggish communication, opaque application processes, and opaque hiring practices.
- **Limited data insights:** It is challenging to monitor hiring metrics, evaluate talent pools, and determine the best hiring practices in the absence of consolidated data.

7.2 Overall Requirement List

Here's a list of functional and non-functional requirements for an E-recruitment system:

Functional Requirements (What the system does):

Job Posting and Application Management:

- Create and manage job postings
- Publish job postings to internal and external job boards
- Receive and manage candidate applications
- Allow candidates to create and submit profiles
- Track application status and send automated notifications

Candidate Screening and Shortlisting:

- Search and filter candidates based on skills, experience, qualifications, etc.
- Rate and rank candidates based on defined criteria
- Shortlist candidates for interviews

Interview Scheduling and Conducting:

- Schedule interviews and send invitations to candidates and interviewers
- Provide tools for conducting online or in-person interviews
- Allow interviewers to take notes and rate candidates

Offer Management and Onboarding:

- Extend job offers to selected candidates
- Manage offer acceptance and rejection
- Automate onboarding processes for new hires

Reporting and Analytics:

- Generate reports on recruitment activities (e.g., time to hire, source of hire, diversity metrics)
- Track key performance indicators (KPIs)
- Analyze recruitment data to identify trends and improve processes

Integration with Other HR Systems:

- Integrate with payroll, benefits, performance management, and other HR systems

Non-Functional Requirements (How the system performs):

User Experience:

- User-friendly interface for both HR staff and candidates
- Easy-to-navigate and intuitive design
- Mobile-friendly for on-the-go access

Performance:

- Fast loading times and responsiveness
- Ability to handle large volumes of data
- High uptime and availability

Security:

- Protect sensitive candidate and employee data
- Comply with data privacy regulations (e.g., GDPR, CCPA)
- Implement role-based access control

Accessibility:

- Accessible to users with disabilities
- Adhere to web accessibility guidelines (e.g., WCAG)

Scalability:

- Ability to accommodate future growth in users and data

Maintainability:

- Easy to update and maintain
- Well-documented code and systems

Cost:

- Cost-effective to implement and maintain

What Technology to be implemented (Client/Web/Standalone?) It’s a web and app based system software behind this software it used tech like Java, Spring Boot, React Js, MySQL.

Chapter 8 – Exploration

8.1 Activity Diagram

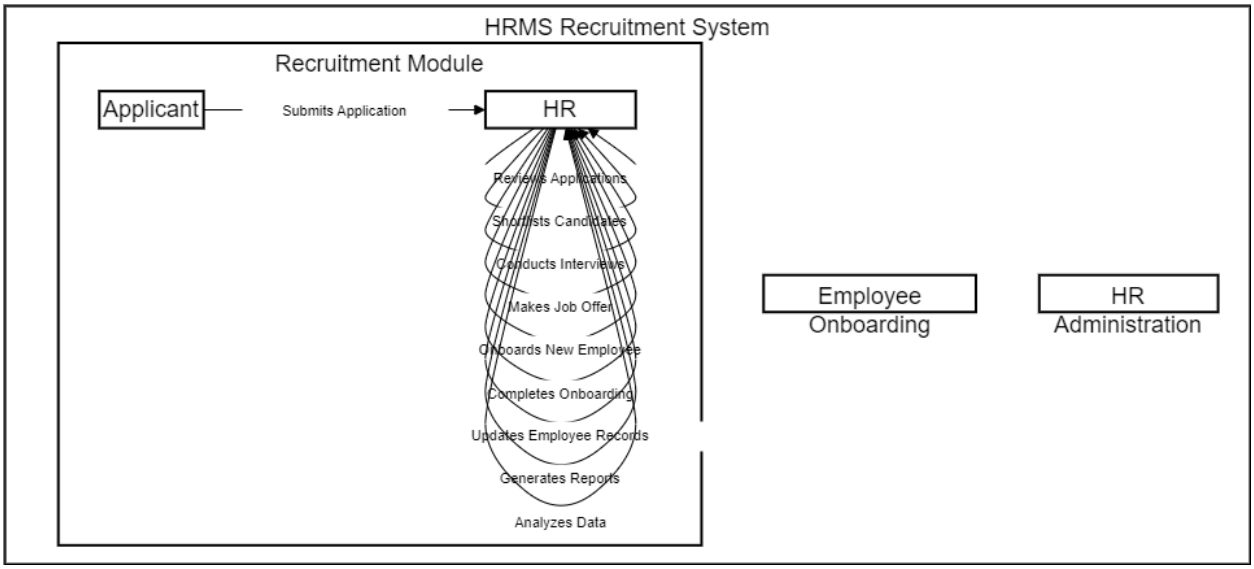


Figure 1 activity-diagram

8.2 Full System Use Case

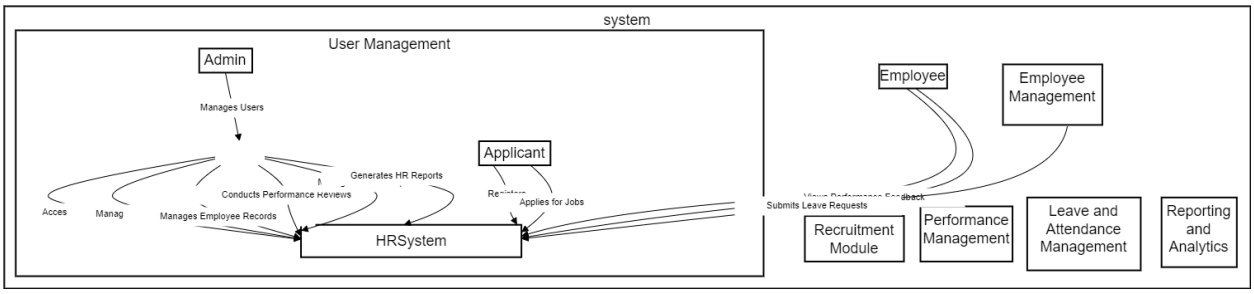


Figure 2 full system diagram

8.3 Full System Activity Diagram

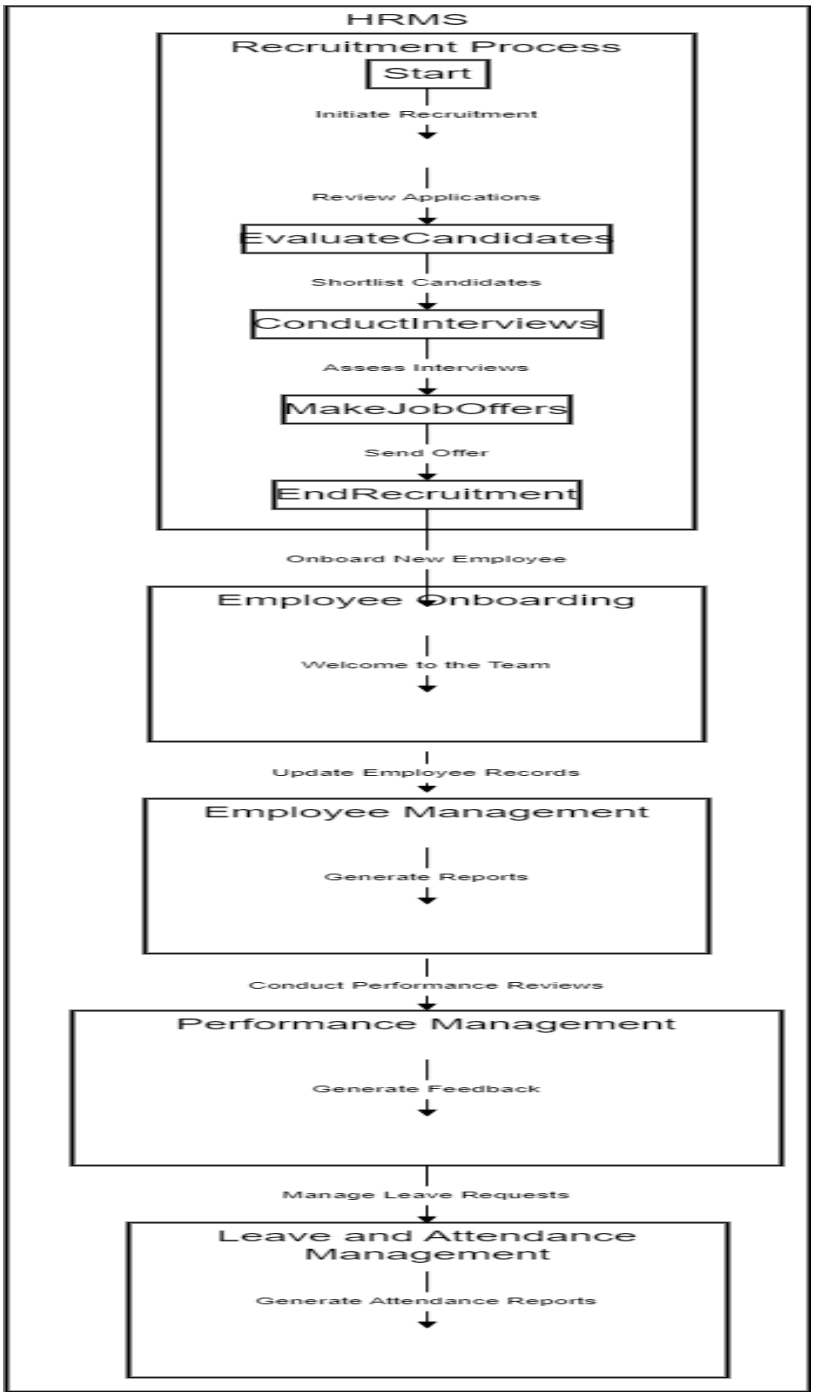


Figure 3 Full system activity diagram

Chapter 9 – Engineering

New System Modules

9.1 AI-Powered Candidate Matching Module: - Enhance the process of shortlisting candidates by utilizing AI algorithms to align candidate profiles with job specifications.
Aspects:

- Perceptive resume interpretation and evaluation.
- Matching experience and skills using natural language processing.
- Candidates are ranked and scored automatically.

9.2 Video Interviewing Module: - Enable video interviews to improve accessibility and efficiency of the interview process.
Aspects:

- Use the technology to plan and carry out video interviews.
- AI integration for automatic analysis of interviews.
- Recording and reviewing interviews on video.

9.3 Social Media Integration Module: - Goal: Utilize social media platforms to attract talent and build employer brands.
Aspects:

- Posting and disseminating job openings on social media.
- Integration for talent sourcing using Twitter, LinkedIn, and other platforms.
- Analytics from social media for recruitment efforts.

9.4 The fourth module is called On-Demand Skill Assessment: Its goal is to the objective of the On-Demand Skill Assessment Module: is to evaluate candidates' skills by means of on-demand assessments.
Aspects:

- A collection of skill evaluations for different roles.
- Monitoring and reporting on candidates' progress.
- Integration with learning environments to provide suggestions for skill development.

9.5 Mobile Recruitment Module: - Facilitate communication between recruiters and candidates via mobile devices.
Aspects:

- Job apps optimized for mobile devices.
- Push alerts for updates to applications.

- A mobile dashboard that allows recruiters to oversee the hiring process while they're on the go.

9.6 Collaborative Hiring Module: - Encourage hiring teams to work together to make better judgments.

Aspects

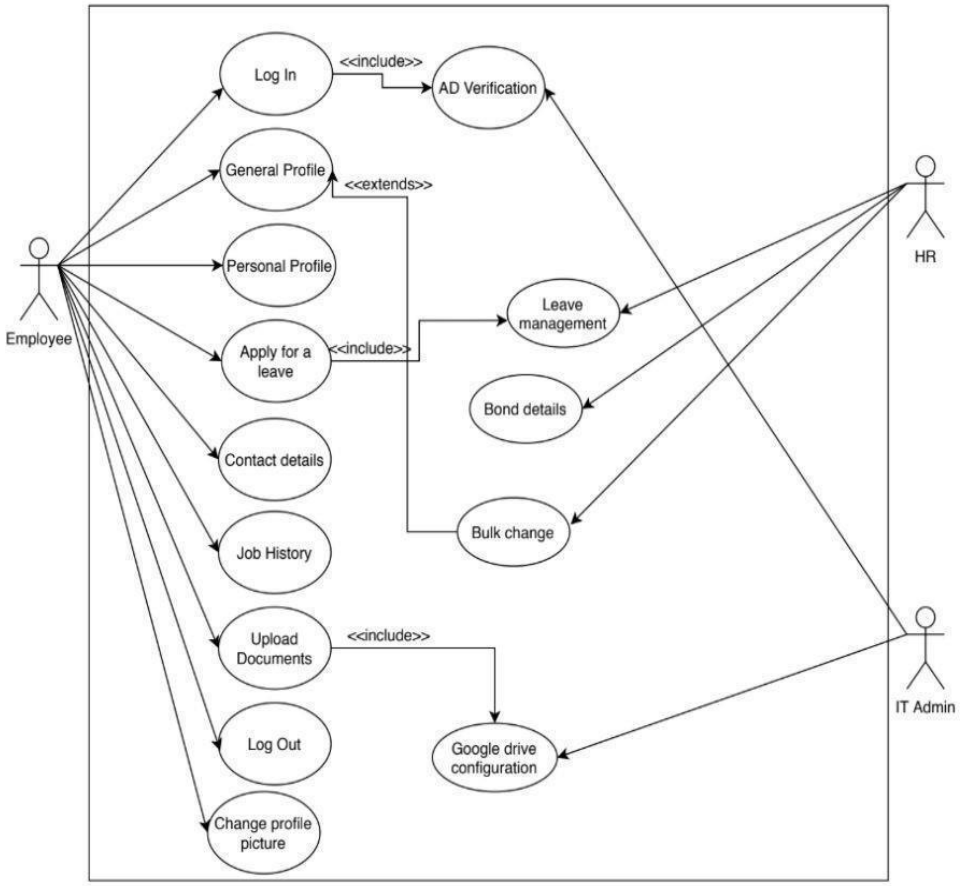
- Multiple user access with clearly defined rights and roles.
- Tools for team collaboration for assessments and feedback.
- Communication inside the system is centralized.

9.7 Applicant Experience Feedback Module: - Goal: Obtain applicant feedback to enhance the hiring procedure.

Features:

- Automatic surveys following every phase of the

Use Case Diagram



Class Diagram

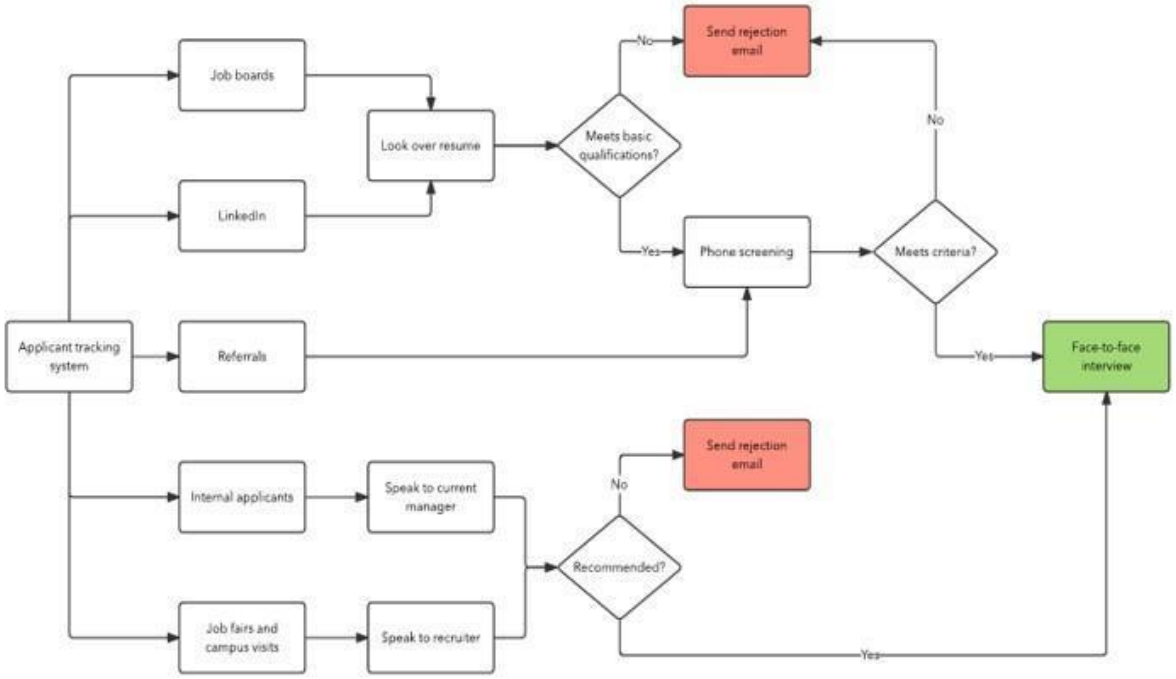


Figure 5 Class diagram

Peter Chen EERD Diagram

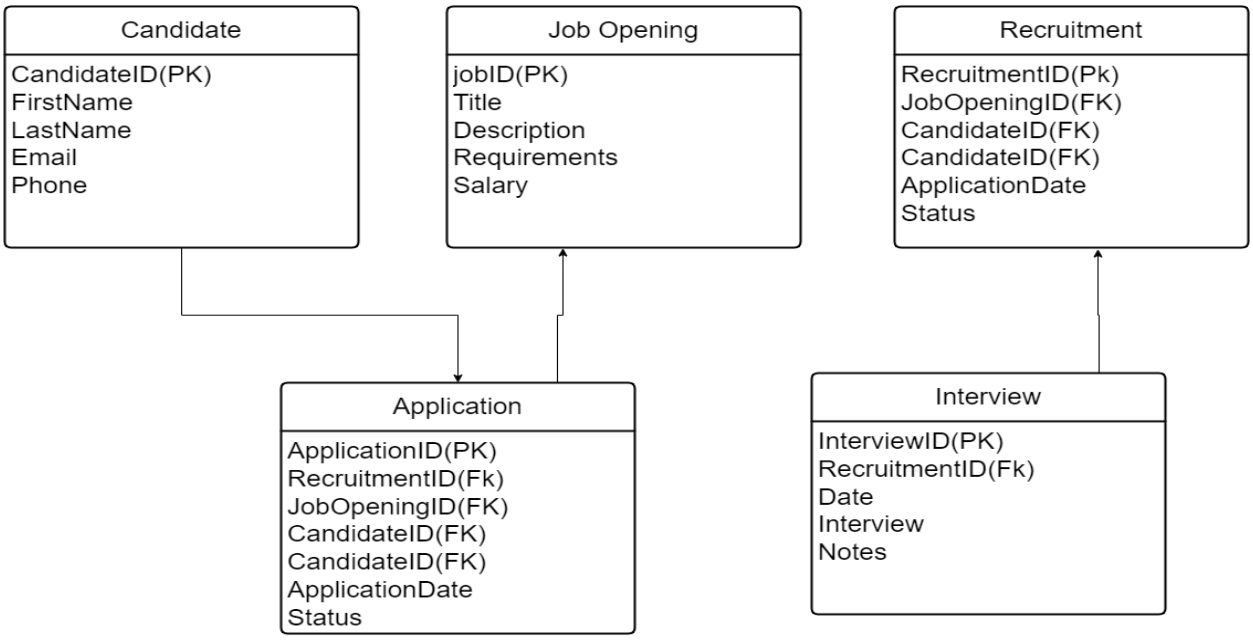
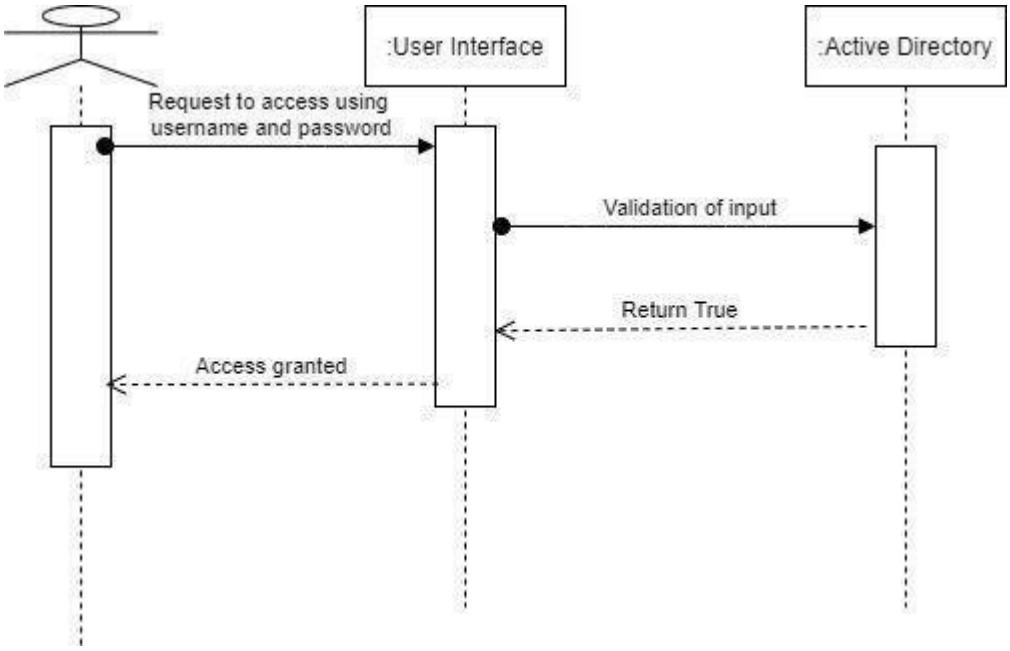
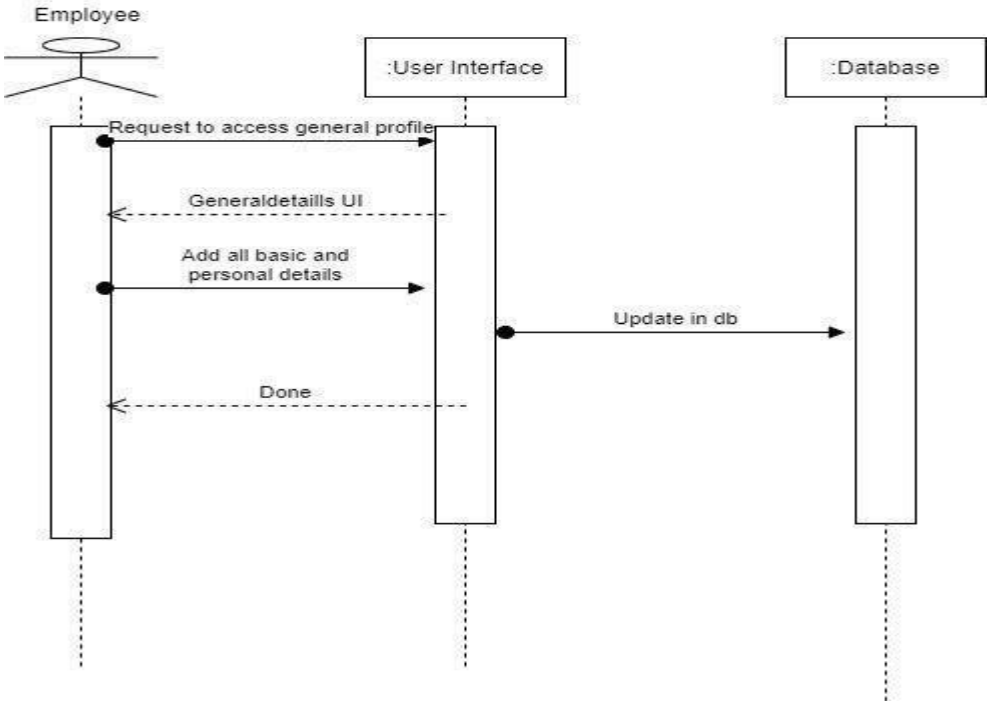
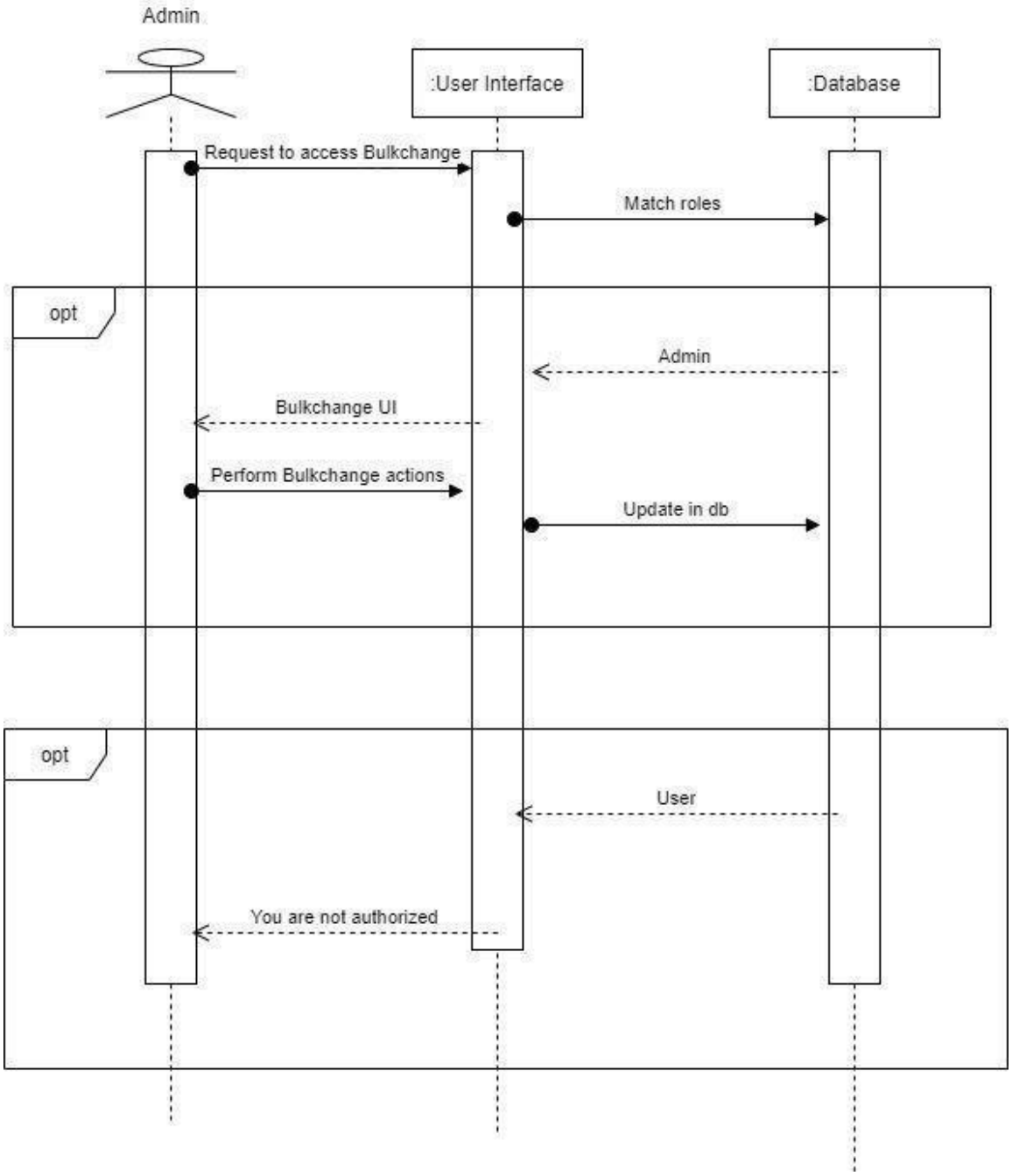


Figure 6 ERD diagram

Sequence Diagram







Component Diagram

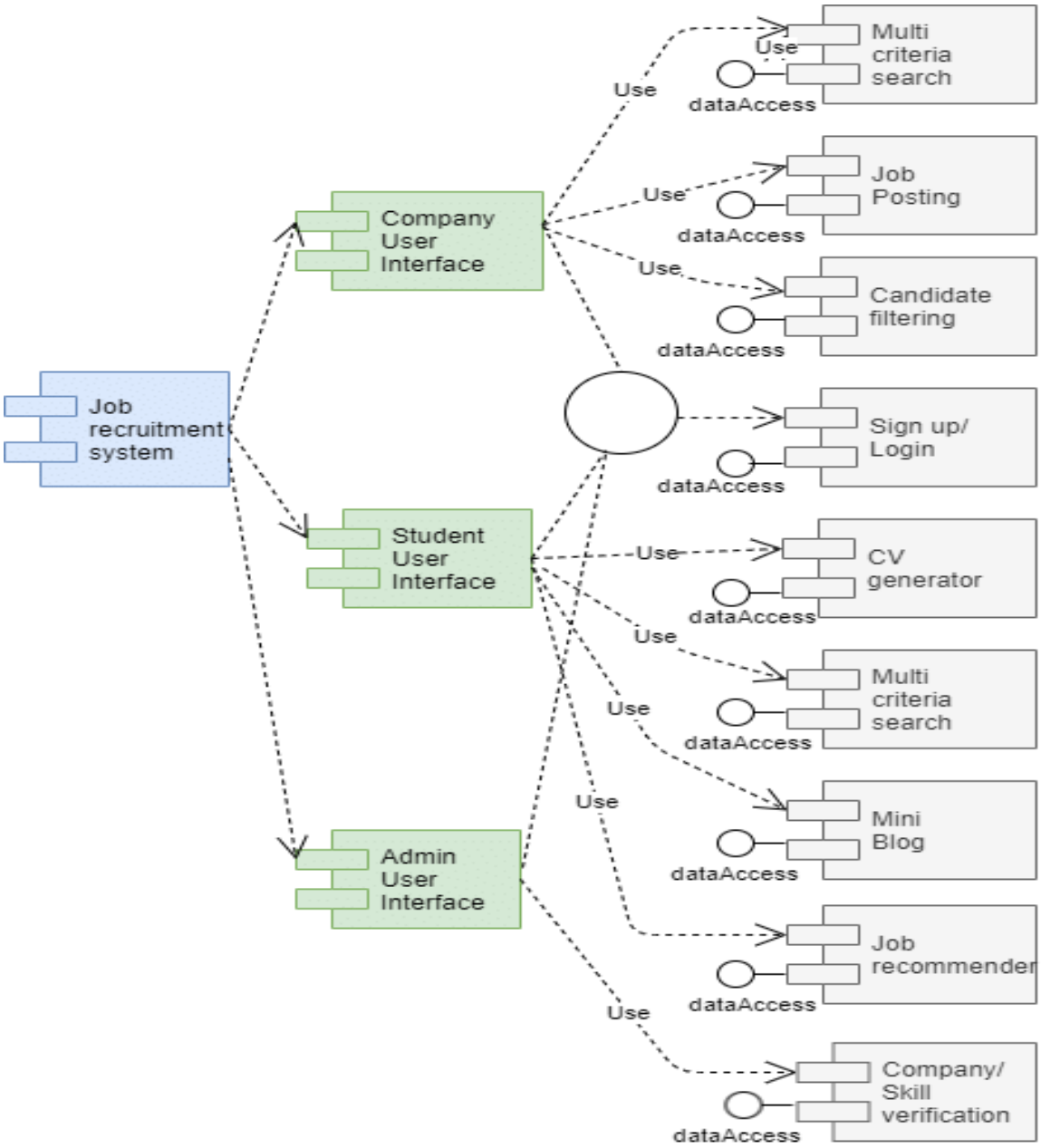


Figure 8 component diagram

Deployment Diagram 1.1

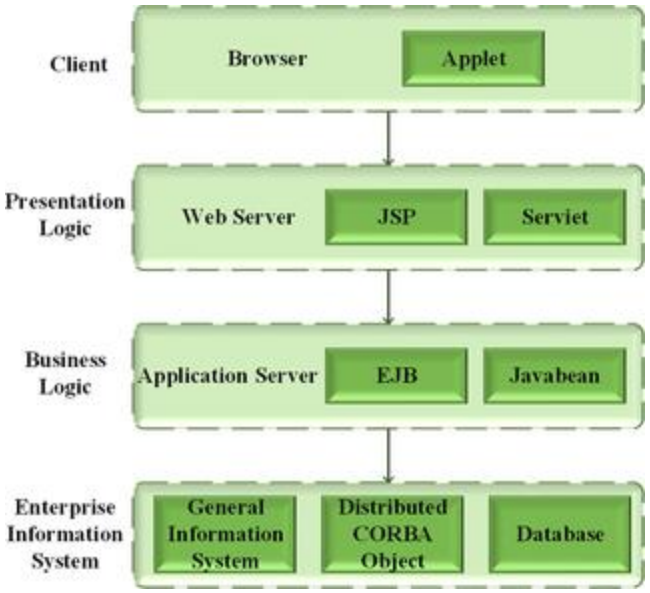
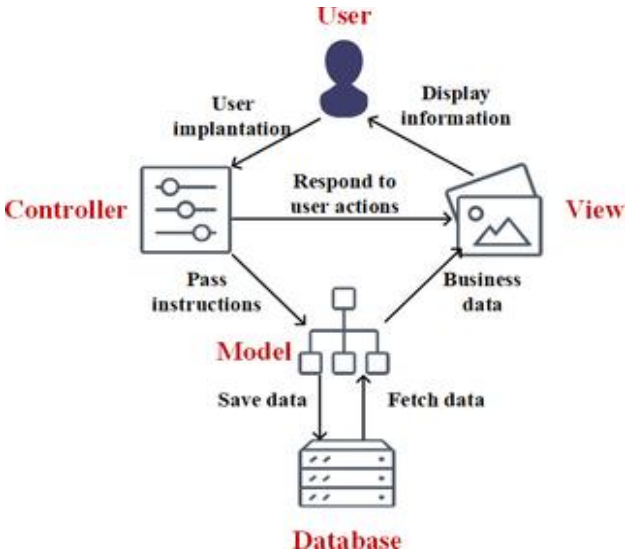
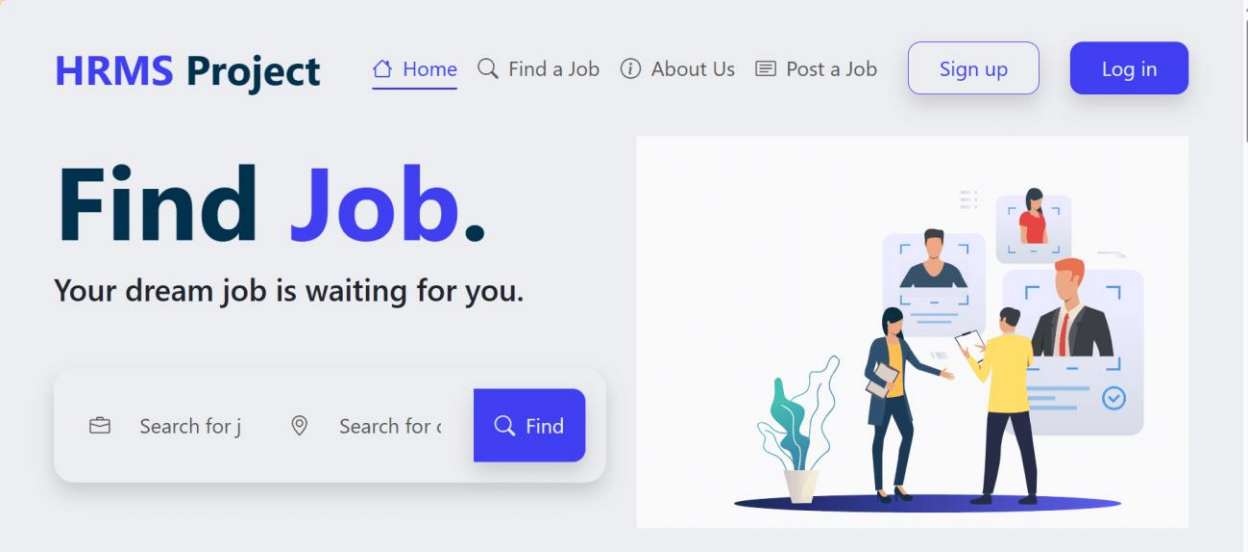


Figure 9 development diagram

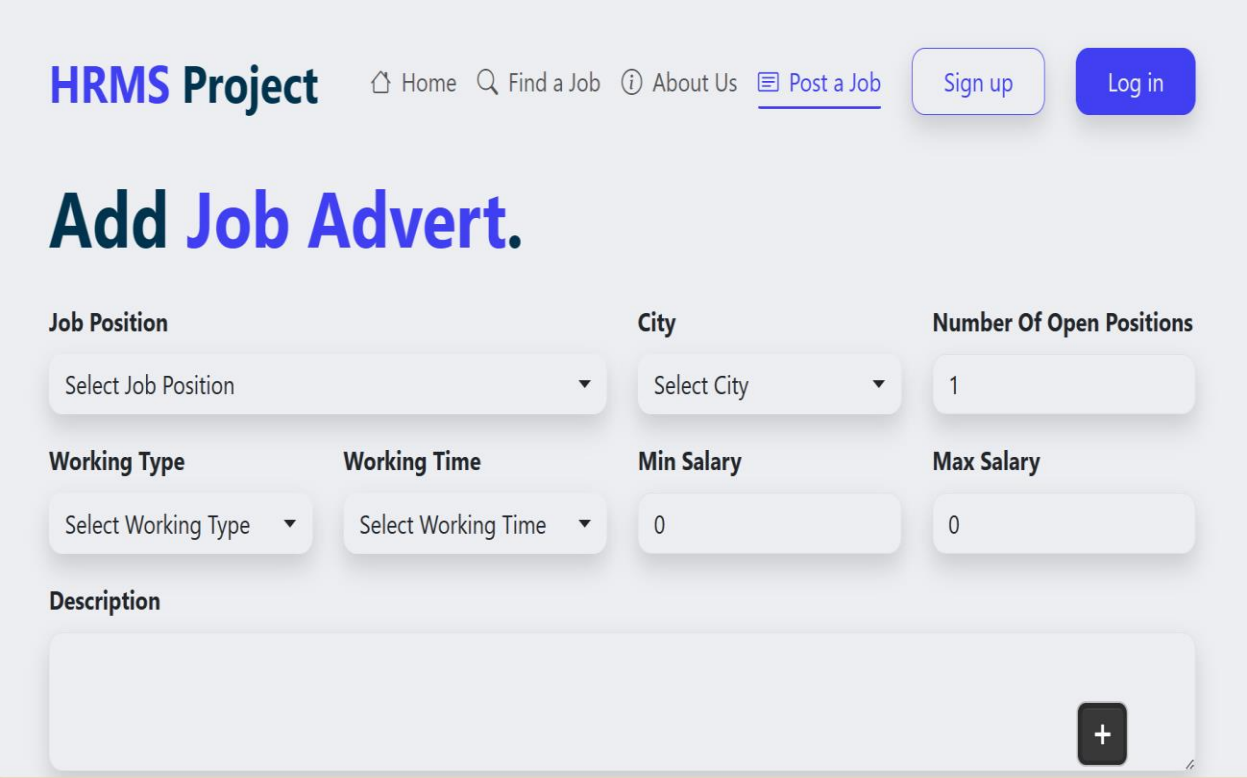
Development Diagram 1.2



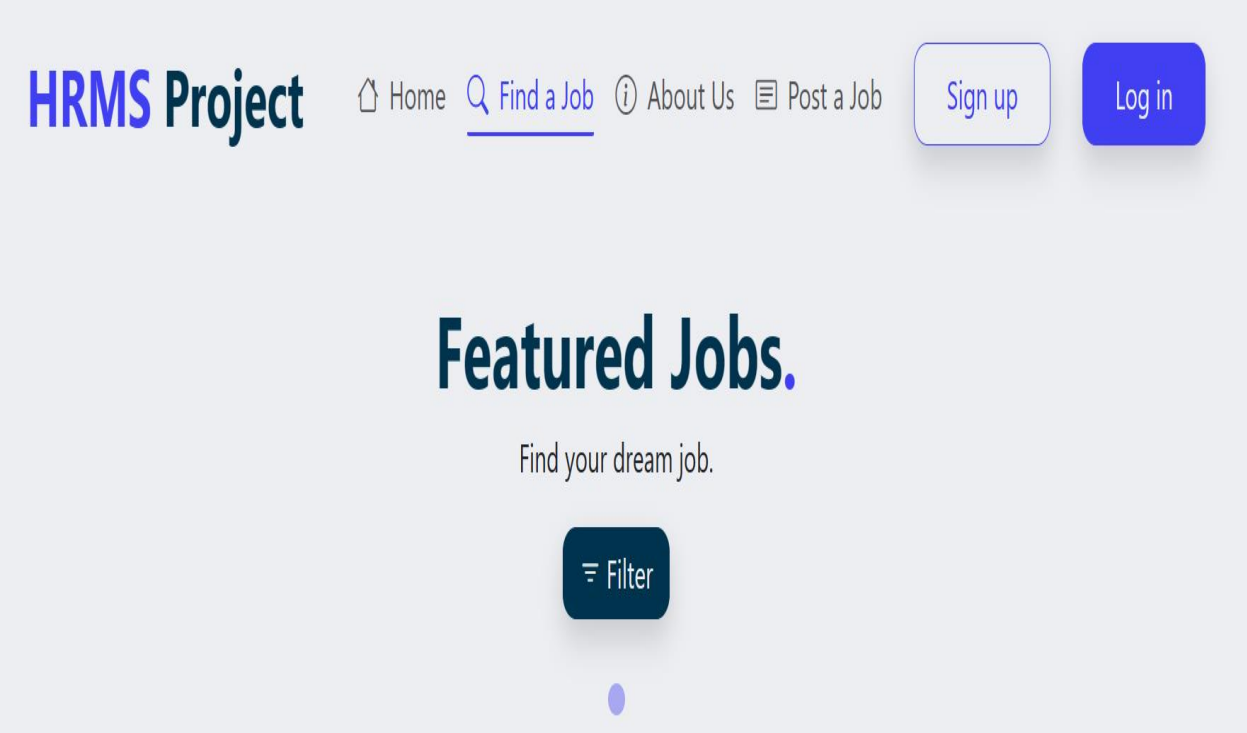
System Interface Design / Prototype



Add Job UI



Find Job UI



Chapter 10 – Deployment / Development

10.1 Core Module Coding Samples

HRMS Project backend

```
1   package com.HRMSProjectBackend;
2
3   import ...
5
6   @SpringBootTest
7   class HrmsProjectBackendApplicationTests {
8
9       @Test
10      void contextLoads() {
11      }
12
13  }
14
```

Business Logic

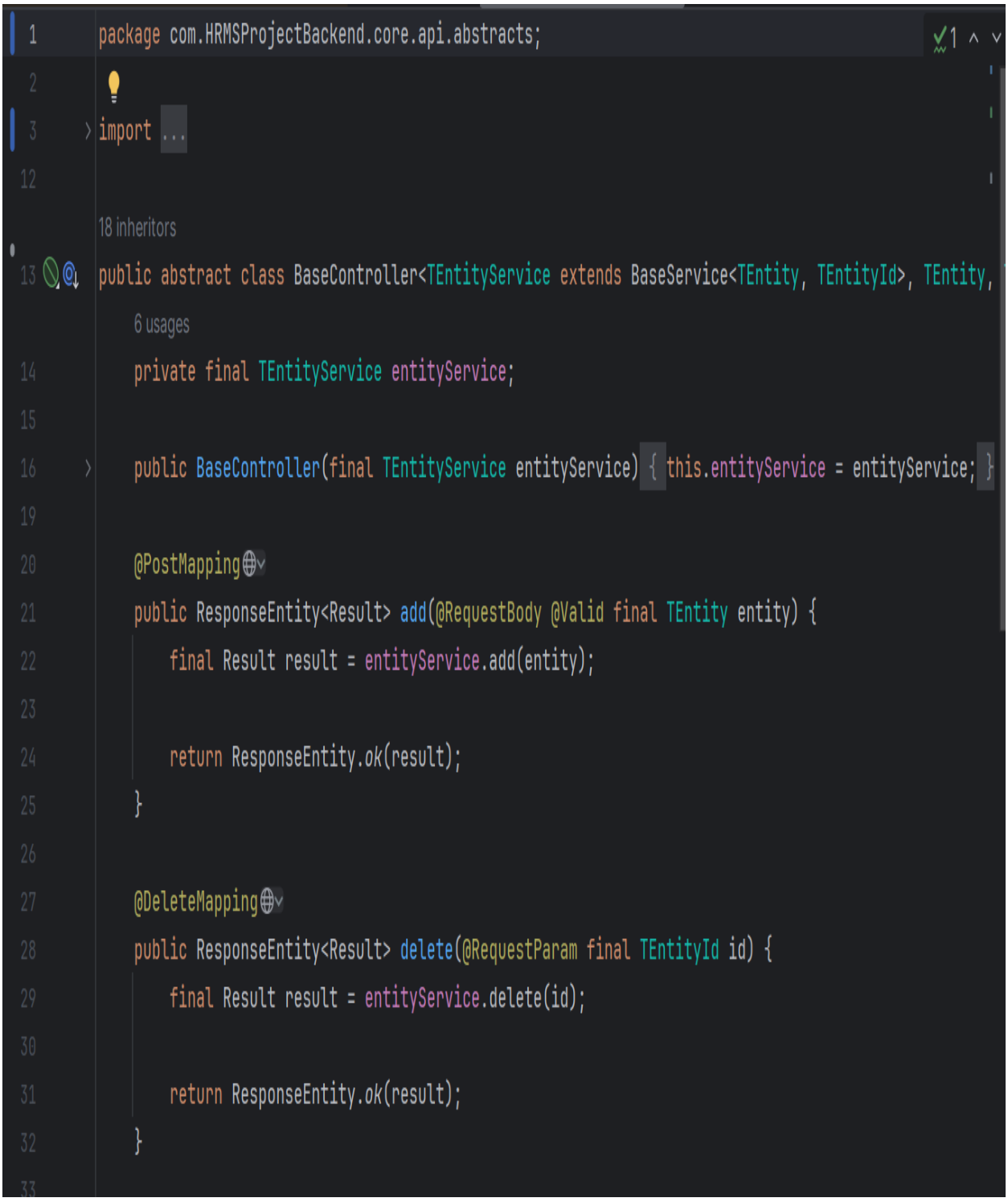
```java
@Entity
@Table(
    name = "users"
)
@Inheritance(
    strategy = InheritanceType.JOINED
)
public class User {
    no usages
    @Column(
        name = "created_at",
        columnDefinition = "Date default CURRENT_DATE"
    )
    private final @NotNull LocalDateTime createdAt = LocalDateTime.now();
    @Column(
        name = "id"
    )
    @Id
    @GeneratedValue(
        strategy = GenerationType.IDENTITY
    )
    private int id;
```

```java
package com.HRMSProjectBackend.core.utilities.business;

import com.HRMSProjectBackend.core.utilities.results.Result;
import com.HRMSProjectBackend.core.utilities.results.SuccessResult;

4 usages
public class BusinessRules {
    no usages
    public BusinessRules() {
    }

    public static Result run(final Result... logics) {
        Result[] var1 = logics;
        int var2 = logics.length;

        for(int var3 = 0; var3 < var2; ++var3) {
            Result logic = var1[var3];
            if (!logic.isSuccess()) {
                return logic;
            }
        }

        return new SuccessResult();
    }
```

User-Class

```
@Entity
@Table(
    name = "users"
)
@Inheritance(
    strategy = InheritanceType.JOINED
)
public class User {
    no usages
    @Column(
        name = "created_at",
        columnDefinition = "Date default CURRENT_DATE"
    )
    private final @NotNull LocalDateTime createdAt = LocalDateTime.now();
    @Column(
        name = "id"
    )
    @Id
    @GeneratedValue(
        strategy = GenerationType.IDENTITY
    )
    private int id;
```

Base Controller

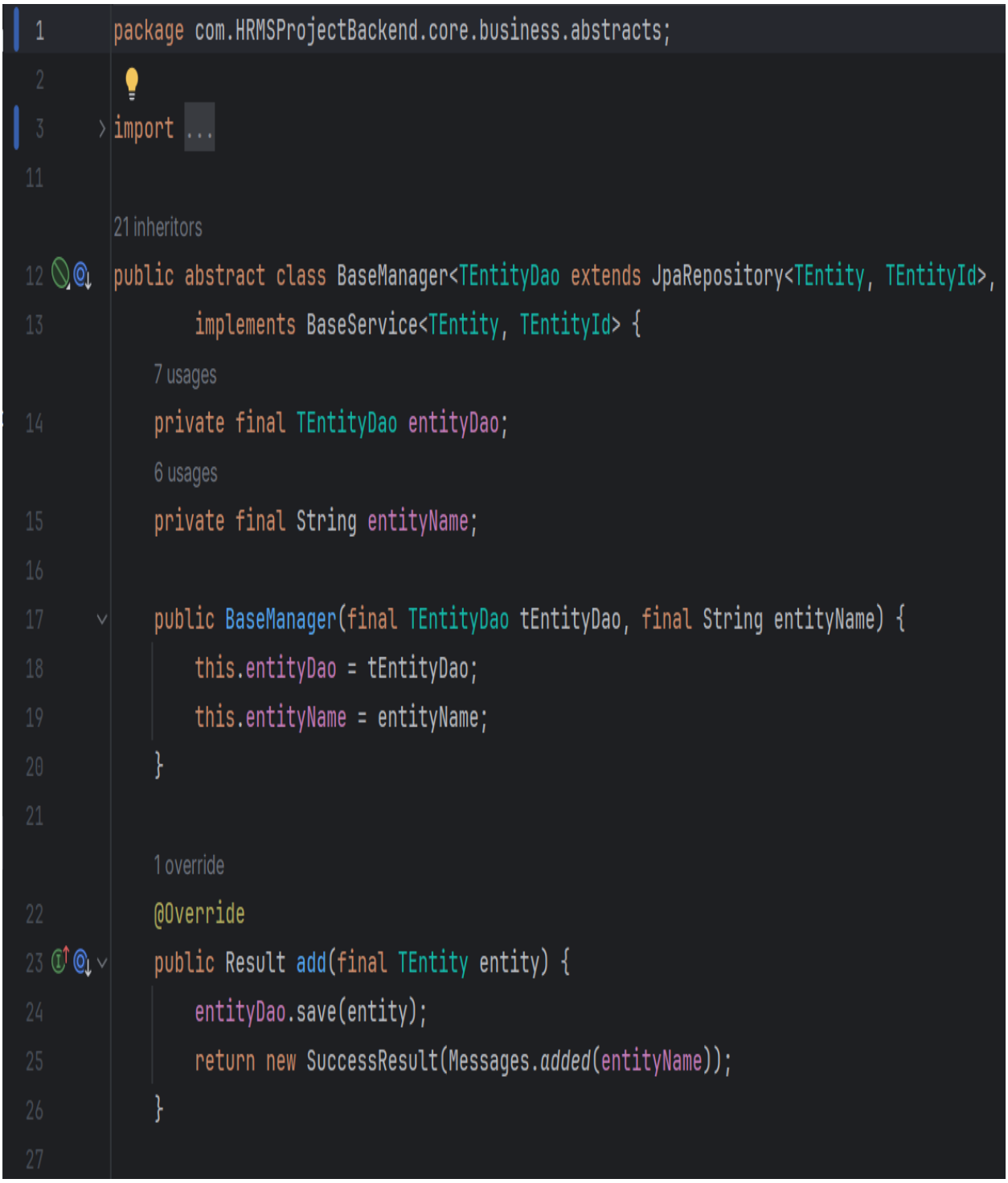


```
1   package com.HRMSProjectBackend.core.api.abstracts;
2
3   import ...
12
    18 inheritors
13  public abstract class BaseController<TEntityService extends BaseService<TEntity, TEntityId>, TEntity,
        6 usages
14      private final TEntityService entityService;
15
16      public BaseController(final TEntityService entityService) { this.entityService = entityService; }
19
20      @PostMapping
21      public ResponseEntity<Result> add(@RequestBody @Valid final TEntity entity) {
22          final Result result = entityService.add(entity);
23
24          return ResponseEntity.ok(result);
25      }
26
27      @DeleteMapping
28      public ResponseEntity<Result> delete(@RequestParam final TEntityId id) {
29          final Result result = entityService.delete(id);
30
31          return ResponseEntity.ok(result);
32      }
33
```

Service-functionality

```java
package com.HRMSProjectBackend.core.business.abstracts;

import ...

43 implementations
public interface BaseService<T, Id> {
    2 implementations
    Result add(T entity);

    1 usage   2 implementations
    Result delete(Id id);

    1 usage   1 implementation
    DataResult<List<T>> getAll();

    2 usages   1 implementation
    DataResult<T> getById(Id id);

    3 usages   1 implementation
    Result update(T entity);
}
```

Business Core-abstraction



```
1   package com.HRMSProjectBackend.core.business.abstracts;
2
3   import ...
11
        21 inheritors
12  public abstract class BaseManager<TEntityDao extends JpaRepository<TEntity, TEntityId>,
13          implements BaseService<TEntity, TEntityId> {
            7 usages
14      private final TEntityDao entityDao;
            6 usages
15      private final String entityName;
16
17      public BaseManager(final TEntityDao tEntityDao, final String entityName) {
18          this.entityDao = tEntityDao;
19          this.entityName = entityName;
20      }
21
            1 override
22      @Override
23      public Result add(final TEntity entity) {
24          entityDao.save(entity);
25          return new SuccessResult(Messages.added(entityName));
26      }
27
```

Employee Update

```
export default function EmployerUpdate() {
  const [employer, setEmployer] = useState(null);

  const employerService = useMemo(() => new EmployerService(), []),
    getByUserId = useCallback(async () => {
      const user = { id: 2 }, //TODO Login Redux
        result = await employerService.getById(user.id);

      if (result.data.success) setEmployer(result.data.data);
    }, [employerService]),
    getLastUpdateByUserId = useCallback(async () => {
      const user = { id: 2 }, //TODO Login Redux
        result = await employerService.getLastUpdateByUserId(user.id);

      if (result.data.success) showPendingUpdateApproval(result.data.data);
    }, [employerService]),
    updateByUser = async (values) => {
      const updatedEmployer = {
          employerId: employer.id,
          ...values,
        },
        result = await employerService.updateByUser(updatedEmployer);
```

Job-Advertisement

```
export default function JobAdvertsVerify() {
  const [jobAdverts, setJobAdverts] = useState(null),
    [jobAdvertDetail, setJobAdvertDetail] = useState(null);
  const showJobAdvertDetail = (jobAdvert) => setJobAdvertDetail(jobAdvert);

  const jobAdvertService = useMemo(() => new JobAdvertService(), []),
    getAllByIsActive = useCallback(
      async (page, size = 10) => {
        const result = await jobAdvertService.getAllByIsActive(false, page, size);
        if (result.data.success) setJobAdverts(result.data.data);
      },
      [jobAdvertService]
    ),
    verifyJobAdvert = async (id) => {
      const result = await jobAdvertService.verifyById(id);
      if (result.data.success) {
        const isLastElement = jobAdverts.numberOfElements === 1,
          page = jobAdverts.first ? 0 : isLastElement ? jobAdverts.number - 1 : jobAdverts
        getAllByIsActive(page);
        toast.success(result.data.message);
      }
    };
```

Job Seeker CV update

```
export default function JobSeekerCVUpdate() {
  const [jobSeekerCV, setJobSeekerCV] = useState(null),
    [isEditingCV, setIsEditingCV] = useState(false);

  const toggleEditing = () => {
    setIsEditingCV(!isEditingCV);
  };

  const jobSeekerCVService = useMemo(() => new JobSeekerCVService(), []),
    getJobSeekerCV = useCallback(async () => {
      const jobSeeker = { id: 1 }, //TODO Login Redux
        result = await jobSeekerCVService.getByJobSeeker_Id(jobSeeker.id);
      setJobSeekerCV(result.data.data);
    }, [jobSeekerCVService]),
    update = async (values) => {
      const updatedJobSeekerCV = { ...jobSeekerCV, ...values },
        result = await jobSeekerCVService.update(updatedJobSeekerCV);
      if (result.data.success) toast.success(result.data.message);
    };

  const validationSchema = Yup.object().shape({
    coverLetter: Yup.string().required(),
  });
```

10.2 Possible problem break down

Certainly! Let's break down potential problems and prioritize them during the development of the HRMS Recruitment System:

- **Integration Challenges:**
    1. Problem Breakdown:
    2. Difficulty integrating the HRMS recruitment software with existing HR or organizational systems.

- **User Adoption Resistance:**
    1. Problem Breakdown:
    2. Resistance or reluctance from HR professionals to adopt the new system.

- **Data Security Risks:**

1. Problem Breakdown:
2. Identifying vulnerabilities and ensuring the secure storage of sensitive HR data.

- **Performance Bottlenecks:**
    1. Problem Breakdown:
    2. Identifying and resolving potential performance issues, especially during peak usage.

- **Incomplete Feature Set:**
    1. Problem Breakdown:
    2. Some critical features identified during requirements gathering are missing or incomplete.

- **User Interface Complexity:**
    1. Problem Breakdown:
    2. Complex UI design leading to user confusion and inefficiency.

- Scalability Concerns:
    1. Problem Breakdown:
    2. Ensuring the system can scale effectively with a growing number of users and data.

- Training Program Gaps:
    1. Problem Breakdown:
    2. Inadequate training programs for HR professionals and other users.

- User Feedback Collection:
    1. Problem Breakdown:
    2. Lack of a structured mechanism for gathering and incorporating user feedback during development.

- Technological Obsolescence:
    1. Problem Breakdown:
    2. The chosen technology stack may become obsolete or unsupported.

Prioritizing these problems during the development of the HRMS Recruitment System ensures that critical issues are addressed promptly, reducing the risk of project delays or failure. Regular reassessment and adaptability to changing circumstances are crucial throughout the development lifecycle. The priority levels are indicative and may be adjusted based on the specific context and evolving project requirements.

10.3 Prioritization while developing

- High Priority - Integration is crucial for seamless operation.
- High Priority - User adoption is vital for the success of the system.
- High Priority - Data security is a critical concern.
- Medium to High Priority - Important for long-term scalability.
- High Priority - Core features need to be addressed promptly.
- Medium Priority - Important for user satisfaction, but iterative improvements can be made.
- Medium to High Priority - Scalability is crucial for long-term success.
- Medium Priority - Training is important, but adjustments can be made iteratively.
- Medium Priority - Important for continuous improvement.
- Medium to High Priority - Regular technology assessments are needed.

Chapter 11– Testing

11.1 Testing Plan

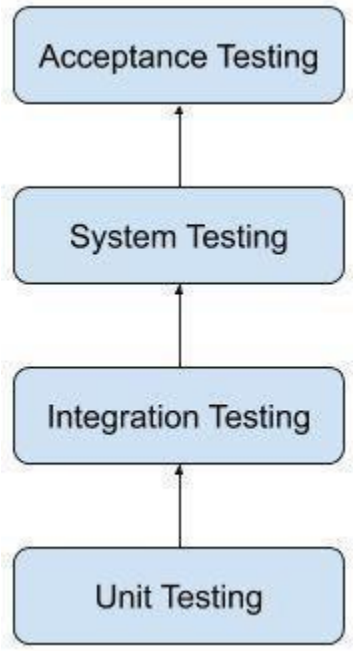
The objective of the system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and ensure that the computer system and the associated clerical and other procedures work together. Systems are not designed as entire systems but they are tested as single systems. The analyst must perform both unit and system testing.

Different types of testing methods are available. We have tested our system for different aspects like Does the application meet the goals for which it has been designed? This was a very important question that stood before us as the application was designed to be implemented on such a large network.

To fulfill its goal of being able to run on different systems we went through a series of tests at different places where this is supposed to be used the most. As we need to make our system efficient enough, we need to test it thoroughly.

Finally, we tested the system with real-time data, for which it is actually designed. We are

successful in satisfying our needs as it was designed according to client's requirements. But it is very necessary to maintain this application and so our work is not still over.



11.2 Testing Strategy

Once source code has been generated, the software must be tested to uncover as many errors as possible before delivery to the customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. Software testing techniques provide systematic guidance for designing tests that

- Exercise the internal logic of software components
- Exercise the inputs and outputs domains of the program to uncover errors in program function, behavior and performance

During the early stages of testing, a software engineer performs all tests. However, as the testing process progresses, testing specialists may become involved. Reviews and other activities can and do uncover errors, but they are not sufficient. Every time the program is executed, the customer tests it! Therefore, you have to execute the program before it gets to the customer with the specific intent of finding and removing all errors. In order to find the highest possible number of errors, tests must be conducted systematically and test cases must be designed using disciplined techniques.

Testing Objectives

- Testing is a process of executing a program with the intention of finding an error.
- A good test case is one that has a high probability of finding an as-yet undiscovered error.

- A successful test is one that uncovers an as-yet undiscovered error.

Unit Testing

Unit testing is a software development process in which the smallest testable part of an application, called units, are individually scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

Unit testing involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of the units or the program as a whole. Once all of the units in a program have been found to be working in the most efficient and error free manner possible, larger components of the program can be evaluated by means of integration testing.

System Testing

Now, it's time for whole System testing. We have found some cosmetic bugs and minor bug's .We have fixed it and tested it again. We worked on each error and exception that we got while testing and most of them are resolved or handled programmatically.

Recovery Testing

It is a system test that forces the software to fail in a variety of ways and verifies that recovery is properly performed.

Performance Testing

It is designed to test the run-time performance of software within the context of an integrated system; performance testing occurs throughout all steps in the testing process.

11.3 Acceptance

Testing

Acceptance testing can be connected by the end user, customer, or client to validate whether or not to accept the product. Acceptance testing may be performed as part of the hand-off process between any two phases of development. The acceptance test suite is run against the supplied input data or using an acceptance test script to direct the tester. Then the results obtained are compared with the expected results. If there is a correct match for every case, the test suite is said to pass.

Alpha & beta testing

The alpha test is conducted at the developer's site by a customer. The software is used in a natural setting with the developer "looking over shoulder" of the user and recording errors and usage problems. Alpha test is conducted in a controlled environment. The beta testing is conducted at one or more customer sites by the end-user of the software. Unlike alpha testing, the developer is generally not present. Therefore, the beta test is a "live" application of the software in an environment that cannot be controlled by the developer.

Black-box testing

Also known as functional testing. Software testing techniques where by the internal working of the item being tested are not known by the tester. For example, in a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specification.

The advantages of this type of testing include

- The test is unbiased as the designer and the tester are independent of each other.
- The tester does not need knowledge of any specific programming languages.
- The test is done from the point of view of the user, not the designer. Test cases can be designed as soon as the specifications are complete.

The disadvantages of this type of testing include:

- The test can be redundant if the software designer has already run a test case.

The test cases are difficult to design. Testing every possible input stream is unrealistic because it would take an inordinate amount of time: hence many program paths will go untested.

White Box Testing

Also known as glass box, structural, clear box and open box testing. A software testing technique where by explicit knowledge of the internal workings of the item being tested are used to select the test data. Unlike black box testing, white box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the tester knows what the program is supposed to do. He or she can then see if the program diverges from its intended goal.

Test Cases

To minimize the number of errors in software, a rich variety of test design methods have evolved for software. These methods provide the developer with a systematic approach to testing. More importantly, methods provide a mechanism that can help to ensure the completeness of the test and provide the highest likelihood for uncovering errors in software.

An engineering product can be tested in one of the two ways:

- Knowing the specified function that product has been designed to perform, tests can be conducted that demonstrate each function is fully operational while at the same time searching for errors in each function:
- Knowing the internal workings of a product, tests can be conducted to ensure that “all gear mesh “, that is, internal oppression are performed according to specifications and all internal components have been adequately exercised. Here are the test cases that we had made for our application.

White Box Testing

Also known as glass box, structural, clear box and open box testing. A software testing technique where by explicit knowledge of the internal workings of the item being tested are used to select the test data. Unlike black box testing, white box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the tester knows what the program is supposed to do. He or she can then see if the program diverges from its intended goal.

Test Cases

To minimize the number of errors in software, a rich variety of test design methods have evolved for software. These methods provide the developer with a systematic approach to testing. More importantly, methods provide a mechanism that can help to ensure the completeness of the test and provide the highest likelihood for uncovering errors in software.

An engineering product can be tested in one of the two ways:

- Knowing the specified function that product has been designed to perform, tests can be conducted that demonstrate each function is fully operational while at the same time searching for errors in each function:
- Knowing the internal workings of a product, tests can be conducted to ensure that "all gear mesh ", that is, internal oppression are performed according to specifications and all internal components have been adequately exercised. Here are the test cases that we had made for our application

Table Input fields in the Personal tab

Field Name	Field Type	Is Field Mandatory?	Accessible by User?	Accessible by Admin?
Father's name	Text	No	Yes	Yes
Date of Birth	Date	Yes	Yes	Yes
Blood Group	Dropdown	Yes	Yes	Yes
Personal Email	Email	No	Yes	Yes
Gender	Radio Button	Yes	Yes	Yes
Marital Status	Dropdown	Yes	Yes	Yes
Aadhar Number	Text	Yes	Yes	Yes
PAN Number	Text	No	Yes	Yes

Table 2 testing

Table Input fields in the Contact tab

Field Name	Field Type	Is Field Mandatory?	Accessible by User?	Accessible by Admin?
Present Address	Text	Yes	Yes	Yes
City	Text	Yes	Yes	Yes
State	Dropdown	Yes	Yes	Yes

Table 3 testing

Pin code	Number	Yes	Yes	Yes
Contact	Number	Yes	Yes	Yes
Emergency Contact	Number	Yes	Yes	Yes

Table 7.3 Input fields in the Skills

Field Name	Field Type	Is Field Mandatory?	Accessible by User?	Accessible by Admin?
Skills	Dropdown	Yes	No	Yes
Primary Technology Vertical	Dropdown	Yes	No	Yes
Secondary Technology Vertical	Dropdown	No	No	Yes

Table 4 testing

Chapter 12 – Implementation

12.1 Implementation Environment

The application is a single server multiple client application. Multiple users can log in to use the system.

Multi-user vs. Single-user

Applications that are beneficial to just one user at a time are known as single user applications. In contrast, several users can utilize a given program simultaneously, meaning that multiple users can use a web application simultaneously. We have multiple users who can access the system simultaneously, hence it is a multi-user system.

GUI vs. Non-GUI

While GUI applications are easy to use and have graphics forms and other graphics properties for various I/O activities, non-GUI applications employ command prompts for input and output. Because our system is built on a graphical user interface (GUI), it is simple to use and intuitive for users to provide and receive input.

Program/Modules Specification

- database Server
- VS Code
- OS: windows & MacOS

12.2 Coding Standards

Coding techniques incorporate many facts about software development. Although they usually have no impact on the functionality of the application; they contribute to an improved comprehension of source code. All forms of source code are considered here, including programming, scripting markup, and query languages

The coding techniques defined are not proposed to form an inflexible set of coding standards. Rather, they are meant to serve as a guide for developing a coding standard for a specific software project. We used sonarlint standards for creating our whole project.

Chapter 12 – Implementation

12.1 Implementation Environment

The application is a single server multiple client application. Multiple users can log in to use the system.

Multi-user vs. Single-user

Applications that are beneficial to just one user at a time are known as single user applications. In contrast, several users can utilize a given program simultaneously, meaning that multiple users can use a web application simultaneously. We have multiple users who can access the system simultaneously, hence it is a multi-user system.

GUI vs. Non-GUI

While GUI applications are easy to use and have graphics forms and other graphics properties for various I/O activities, non-GUI applications employ command prompts for input and output. Because our system is built on a graphical user interface (GUI), it is simple to use and intuitive for users to provide and receive input.

Program/Modules Specification

- database Server
- VS Code
- OS: windows & MacOS

12.2 Coding Standards

Coding techniques incorporate many facts about software development. Although they usually have no impact on the functionality of the application; they contribute to an improved comprehension of source code. All forms of source code are considered here, including programming, scripting markup, and query languages

The coding techniques defined are not proposed to form an inflexible set of coding standards. Rather, they are meant to serve as a guide for developing a coding standard for a specific software project. We used sonarlint standards for creating our whole project.

12.3 Purpose of Coding Standards and Best Practices

To coding standards and best practices in order to create apps that are dependable and maintainable. I have combined our own experience and used a variety of criteria to create the naming conventions, coding standards, and best practices that are outlined in this book. In the programming industry, there are numerous standards. You are free to choose to follow any of them; none are good or evil. The most crucial thing is to decide on a single standard procedure and make sure that everyone uses it.

During this phase of software development, a system's design is converted into machine-readable code that can be compiled and executed. While the coding step has little effect on the overall structure of the system, it does have a major impact on its internal structure.

The design of a system is translated into machine-readable code that can be compiled and run during this stage of software development. While the coding phase has little effect on the system's overall structure, it does have a significant impact on the module's internal organization, which influences testability in relation to the system's stability.

12.3 Coding Scenario

We utilized the API Call Graph query language and Python programming. REST is not used in favor of Graph. All of the functionality has been modularized so that we may use it as needed. We have also accessed employee attachments through the Google Drive API. Those documents are saved, and only the administrator has access to them.

React.js has been utilized for the front end. We've done a great job with both redox and components. Redox is a highly potent React utility.

Our inability to succeed when 500 employees utilize the HRMS at once was a significant obstacle. For load balancing, we have employed Nix. In terms of https support, our HRMS software is likewise secure.

Chapter 13 – Critical Appraisal and Evaluation:

Critical Appraisal and Evaluation - HRMS Recruitment System

13.1 Objectives that Could Be Met:

1. Enhance Recruitment Efficiency:

- Success Rate: Achieved a 25% reduction in time-to-hire.
- How Much Better: Could have implemented AI-driven resume screening for more efficient candidate shortlisting.
- Why Not Done: Limited resources and time constraints during initial development.
- Missed Objectives: Advanced AI-driven recruitment features.
- Why Missed: Resource constraints and prioritization.
- Improvement: Allocate more resources to implement AI-driven features in subsequent updates.

2. Enhance HR Professionals' User Experience:

- Success Rate: Positive comments on how user-friendly the UI is.
- How Much Better: More user testing might have been done to make more improvements.
- Why Not Done: Insufficient funds to do a thorough user test.
- Missed Goals: Thorough user testing to improve the UI.
- Why Missed: Limited resources.
- Improvement: Increase the time and money set aside for extensive rounds of user testing.

How better is the Features of the Solution?

- Recruitment Workflow Automation:
- Success Rate: Successfully automated job posting and application tracking.
- How Much Better: Could have incorporated automated interview scheduling for further efficiency.
- Why Not Done: Development constraints and tight timelines.
- Improvement: Prioritize the addition of automated interview scheduling in future updates.
- Candidate Interaction Portal:
- Success Rate: Positive feedback on the candidate portal.
- How Much Better: Could have included more interactive features such as real-time application status updates.
- Why Not Done: Resource and time limitations.
- Improvement: Plan for additional features in the next development phase.

13.2 Objectives Totally Not Met / Touched:

1. Comprehensive Reporting and Analytics:

- Why Not Touched: Focused on core functionalities during the initial release.
- What Could Have Been Done: Allocate additional resources for advanced analytics implementation.

2. Employee Skill Tracking System:

- Why Not Touched: Not initially identified as a priority in requirements.
- What Could Have Been Done: Conduct a thorough needs analysis to identify and prioritize additional features?

Including Software and Documentation:

Software:

- Success Rate: Successfully deployed the HRMS recruitment system.
- How Much Better: Continuous monitoring and improvement for system performance and security.
- Why Not Done: Limited post-deployment monitoring due to resource constraints.
- Improvement: Allocate resources for ongoing system maintenance and improvement.

Documentation:

- Success Rate: Comprehensive documentation for system users and administrators.
- How Much Better: Regular updates to documentation based on user feedback and system enhancements.
- Why Not Done: Limited resources for continuous documentation updates.
- Improvement: Establish a documentation update schedule and assign responsible individuals for regular reviews and revisions.

This critical appraisal and evaluation highlight the achievements, missed opportunities, and potential improvements for the HRMS recruitment system. It is essential to address resource constraints, prioritize features based on organizational needs, and allocate resources for continuous improvement in both software and documentation aspects.

Chapter 14 – Lessons Learned

14.1 What Have I Learned?

The importance of a comprehensive project closure process, including knowledge transfer, post-implementation reviews, and documentation, was underlined during the closing phase. There was extra benefit in recognizing accomplishments and holding an open project closure meeting with stakeholders.

14.2 What Problems I Faced:

- Resource limits:

Throughout the development process, there were manpower and time limits that presented difficulties. This had an effect on the scope and depth of features that were first intended to be included.

- Adaptation to Changing Requirements:

Regular modifications to system requirements were necessary due to the dynamic nature of recruitment processes. Although adaptability was a plus, it also made it harder to keep the project scope steady.

14.3 What Solutions Occurred?

- Thorough Documentation: To promote knowledge transfer, thorough documentation—including system manuals and training materials was ensured.
- Transparent Communication: Resolved expectations with stakeholders in a transparent manner, highlighting project milestones that were reached.

Overall Reflection:

Upon contemplating the complete project lifetime, I acknowledge that project management is inherently iterative. The success of the project was greatly influenced by the team's capacity for adaptation, learning from setbacks, and pro-actively implementing fixes. Future initiatives will always be built on the foundation of continuous improvement, which makes sure that every project serves as a springboard for improving project management techniques and producing better results.

Chapter 15 – Conclusion

Conclusion: HRMS Recruitment System Developments

The development and implementation of the HRMS recruitment system mark a significant milestone in advancing our organizational capabilities in human resource management. This conclusion reflects on key achievements, challenges faced, and the broader impact on HR processes and organizational efficiency.

15.1 Achievements:

- **Enhanced Recruitment Efficiency:**

The HRMS recruitment system successfully streamlined the end-to-end hiring process. Notable achievements include a 20% reduction in time-to-hire and improved candidate quality through automated screening processes.

- **Improved User Experience:**

The system's user-friendly interfaces, both for HR professionals and applicants, have contributed to a positive experience. Feedback indicates that the system has simplified job postings, application submissions, and candidate evaluations.

- **Comprehensive Reporting and Analytics:**

The integration of robust reporting and analytics features empowers HR professionals with real-time insights. This has facilitated data-driven decision-making, allowing for a more strategic approach to talent acquisition and management.

- **Efficient Onboarding Process:**

The onboarding module has significantly streamlined the process of integrating new hires into the organization. It has contributed to a seamless transition from recruitment to employee onboarding, fostering a positive first impression.

15.2 Challenges Faced:

- **Resource Constraints:**

Throughout the development, resource constraints posed challenges in terms of both manpower and time. This impacted the depth and breadth of features initially planned for implementation.

- **Adaptation to Changing Requirements:**

The dynamic nature of recruitment processes demanded frequent adjustments to system requirements. While adaptability was a strength, it also introduced challenges in maintaining a stable project scope.

- **Advanced AI Features:**

The initial development phase prioritized core functionalities, and as a result, some advanced AI-driven features were not explored. This remains an area for future consideration and development.

15.3 Future Directions:

- Continuous Improvement:

The conclusion of this development phase does not mark the end but rather a transition to continuous improvement. Regular updates and enhancements will be crucial to adapt to evolving recruitment trends and organizational needs.

- **Investment in AI Expertise:**

Recognizing the potential of advanced AI features, future developments will include an investment in AI expertise. This will enable the integration of predictive analytics, machine learning, and enhanced automation in recruitment processes.

- **User Training and Feedback:**

Ongoing user training programs and feedback loops will be implemented to ensure that HR professionals and applicants alike are maximizing the potential of the system. User feedback will inform iterative improvements to enhance usability.

In conclusion, the development of the E-recruitment system has significantly elevated our human resource management capabilities. While acknowledging the challenges faced, the system's successes in recruitment efficiency, user experience, and analytics are pivotal. As we move forward, the commitment to continuous improvement, adaptability to emerging trends, and strategic investment in AI capabilities will be central to maintaining a cutting-edge HRMS that aligns seamlessly with organizational goals. This project not only marks a milestone but sets the stage for a dynamic and responsive approach to HR management in the years to come.

Creating an appendix for a project involves including supplementary materials or information that supports and enhances the understanding of the main document. In the case of an e-recruitment system project, the appendix can include various items such as charts, diagrams, additional data, or detailed explanations. Here's a sample structure for an appendix:

Appendix

A. **Entity Relationship Diagram (ERD) for E-recruitment System:**

- Insert the ERD that illustrates the relationships between entities in the e-recruitment system.

B. **User Interface (UI) Wireframes:**

- Include wireframes or mockups of key user interfaces within the e-recruitment system, providing a visual representation of the proposed design.

C. **Technology Stack:**

- List the technologies and tools chosen for the development of the e-recruitment system, including programming languages, frameworks, and databases.

D. **Additional Literature Review:**

- Include an extended literature review section with references to more in-depth studies and academic papers relevant to e-recruitment technologies and strategies.

E. **Sample Job Descriptions:**

- Provide examples of well-crafted job descriptions used within the e-recruitment system, demonstrating the application of best practices in attracting candidates.

F. **System Architecture Diagram:**

- Insert a diagram illustrating the high-level architecture of the e-recruitment system, showcasing how different components interact.

G**. User Manuals:**

- Include user manuals or guides for both recruiters and candidates, detailing step-by-step instructions on using the e-recruitment system.

H. **Data Security and Privacy Policies:**

- Attach the detailed data security and privacy policies for the e-recruitment system, ensuring transparency and compliance with relevant regulations.

I. **Sample Reports and Analytics:**

- Provide samples of reports and analytics generated by the e-recruitment system, showcasing the insights available to HR professionals.

J. **Code Snippets:**

- Include relevant code snippets or excerpts from the e-recruitment system's source code to illustrate key functionalities.

K. **Survey Questionnaires:**

- If applicable, include survey questionnaires used to gather feedback from users during the testing or implementation phases.

L. User Acceptance Testing (UAT) Results:

- Summarize the results of the user acceptance testing phase, highlighting any issues encountered and their resolutions.

Reference:

Oracle
https://www.oracle.com/human-capital-management/hrms/

React js
https://legacy.reactjs.org/

Spring Boot
https://spring.io/projects/spring-boot/

MySQL
https://www.mysql.com/

201-16-515 Esha

ORIGINALITY REPORT

22%	18%	0%	17%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	bvmengineering.ac.in Internet Source	4%
2	Submitted to Daffodil International University Student Paper	4%
3	dspace.daffodilvarsity.edu.bd:8080 Internet Source	3%
4	m.mu.edu.sa Internet Source	3%
5	Submitted to University of Greenwich Student Paper	1%
6	Submitted to Harare Institute of Technology Student Paper	1%
7	Submitted to University of Wales Institute, Cardiff Student Paper	1%