MEKDELA AMBA UNIVERSITY

COLLEGE OF COMPUTING AND INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

Industrial Project on:

Online Job Tracking System

Submitted to the College of computing and Informatics in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Group members full name	ID No.
1. Alemayehu Demissie	1400206
2. Jeginaw Minale	1401446
3. Rahel Fikre	1402024
4. Mekides Mequanent	1401718

Advisor: Mebitu K. (MSc.)

June 2, 2025G.C

Tulu Awuliya, Ethiopia

Approval sheet or declaration

Our project is original and has not been submitted for a degree at any other university. We have appropriately cited all sources of information used in the project.

Student full name	signature	date
1. Alemayehu Demissie	___________	_______________
2. Jeginaw Minale	___________	
3. Rahel Fikre	___________	
4. Mekides Mequanent	___________	

College: Computing And Informatics

Department: Computer Science

Project title: Online Job Tracking System

Approval of advisor and examiners

This is to certify that I have read this project and that in my opinion it is fully adequate, in scope and quality, as a project for the degree of bachelor of computer science.

Mebitu K.	____________	___________
Name of advisor	signature	date

Examining committee members:	signature	date
1. ________________________	______________	_______________
2. __________________________	_____________	
3. __________________________	________________	

It is approved that this project has been written in compliance with the formatting rules laid down by the college.

Acknowledgment

First of all, we would like to thank our God almighty for his daily protection, endless blessing, and giving us health. Secondly, a huge thank you to our advisor, Mr Mebitu Kefale, his advice and support were key to getting this project done. We also want to thank all computer science department teachers at Mekdela Amba University. They taught us everything we needed to know, and the university gave us the right environment to learn and grow. Our friends and classmates were awesome too, we like to thank them for supporting us when we need peer advice. And finally, We’re so grateful to our family for their emotional support and help with different things, We couldn't have done this without their love and sacrifices.

List of acronyms

CRC:	class-responsibility-collaboration
GDPR:	general data protection regulation
HR:	human resource
IDE:	integrated development environment
OS:	Operating system

List of table

List Of Figure

Table of Contents

Abstract

This project aims to develop an integrated online job portal website and mobile application to facilitate the job search process for both job seekers and employers. The platform will offer a user-friendly interface with features such as job searching, resume upload, and job posting. By using programming languages such as HTML, CSS, JavaScript, PHP, and MySQL, the project will ensure efficient data management. The mobile app will be designed for Android to enable users to access the job portal easily. The significance of this project lies in providing a centralized platform that simplifies the recruitment process, connects job seekers with relevant opportunities, and helps employers find qualified candidates. The development of this online job portal website and mobile app aims to enhance the efficiency and effectiveness of the job search and recruitment process, ultimately benefiting both job seekers and employers.

Chapter One: Introduction

1.1. Background

In today's life everybody can communicate each other in everywhere. They share information, technology and knowledge. More recently it is the use of the computers and information technology (IT) to improve the efficiency and competitiveness of businesses that has led to technological change. Since technology is so rapid, there are important implications for businesses. Websites are one of the way flows of information. People can get service by visiting this websites. In our country there are few job tracking websites. It is recent phenomena in our country to be incorporated. This is the result of the number of educational institutions increment. In previous time to recruit an employee it has tedious processes from both sides of the employee and the organization which hire the employee. So this project is one contribution to the small number of websites that are useful in announcing of different jobs.

1.2. Review Of Related Work

The followings are some examples of existing research, projects, and systems related to online job portal:

Jobify: An open-source job portal website project on GitHub that offers features for job seekers to search and apply for jobs, and for employers to post job listings.

JobFinder App: A mobile application project that provides job recommendations based on user preferences and location, with features for job application tracking and notifications.

Talent Acquisition System: An enterprise-level recruitment system used by companies to manage job postings, applicant tracking, and candidate screening processes.

Job Search Aggregator: A web-based system that aggregates job listings from multiple sources and presents them to users in a unified interface for easier job search.

Strength for the above existing systems and projects

Systems can handle a large volume of users and job postings without compromising performance are considered strong, especially in the context of popular job portal websites.

Weakness for the above existing systems and projects:

Complicated navigation, slow loading times, and unclear instructions can lead to a poor user experience, resulting in decreased user engagement.

A limited number of job listings.

1.3. Problem Statement

A significant problem in the job market today is the limited connection between job seekers and employers. Job seekers often lack access to comprehensive information about job vacancies to enhance their employability. At the same time, employers struggle to find candidates who meet their specific requirements, leading to delays and increased recruitment costs.

The extent of the problem varies by region and economic sector. Globally, millions of job vacancies go unfilled due to inefficiencies in recruitment processes. These challenges particularly pronounced rural and underserved communities, where infrastructures and transportations are limited. Small and medium enterprises, which form the backbone of many economies, are disproportionately affected by these issues, as they often lack the resources to implement effective recruitment strategies.

1.4. Proposed solution

- ➢ We will develop platform that provide comprehensive job listings to empower job seekers in their search for employment.
- ➢ This online job tracking website is technology-driven solution to improve recruitment processes and bridge the gap between job seekers and employers.
- ➢ The platform will provide a centralized and user-friendly space where individuals can access job openings across various industries and locations, breaking down geographical barriers.
- ➢ Our project will support small and medium enterprises by make the system cost affordable.

1.5. Objective of the project

1.5.1. General objective

The general objective of this project is developing an online job portal website and mobile application platform that connects job seekers with employers, facilitating the recruitment process and improving employment opportunities.

1.5.2. Specific Objective

- Define Project Scope and Objectives
- Create a Detailed Project Plan
- Design User Interfaces
- Develop Backend Infrastructure
- Implement Frontend Development
- Mobile App Development
- Deployment
- User Testing and Feedback

1.6. Methodology

1.6.1. Requirement gathering methods

To ensure robust data gathering for the project, we have utilized the following tools:

Questionnaires: Distributed online to job seekers, employers, and administrators to understand their needs and preferences.

Interviews: Conducted with industry professionals, HR experts, and target users for in-depth insights.

Focus Groups: Sessions involving small groups of end-users to discuss features, usability, and desired improvements in job portals.

1.6.2. Analysis and design methodology

To analysis the gathered information, we were used the following methods:

Use Case Scenarios: Documenting the specific scenarios in which different user types (job seekers, employers, and administrators) will interact with the system.

Competitor Analysis: Reviewing existing job portals to gather insights into best practices and gaps in current offerings.

Observation: Monitoring the workflow of recruitment processes in organizations to identify inefficiencies that the portal can address.

Design methodology:

Software development methodology: The Agile methodology more appropriate fit for our system. Because Agile emphasizes continuous collaboration and feedback loops with stakeholders (job seekers, employers, admins). This allows the development team to quickly incorporate user insights, ensuring the delivered features truly meet their needs.

Iterative development helps in identifying and addressing issues or challenges early in the process, reducing the risk of large-scale failures later on[4].

1.6.3. Implementation methodology

The development environment provides different programming tools, from those choices we have chosen the following programming tools by considering the solution to meet the user's requirement in web site development, the availability of the technology and by considering the knowledge that we have.

Front end: HTML, CSS, JavaScript

IDE: visual studio

Back end: PHP (for website development), SQL server (for data retrieve), java (for mobile application).

Documentation tools: MS-Word 2010

1.7. Scope and limitation

1.7.1. Scope

This project focuses on creating a platform that connects job seekers and employers, ensuring a smooth recruitment process. Key areas of coverage include:

Geographical Scope:

The portal will initially target a specific region (Ethiopia)

Target Audience:

Job Seekers: Students, fresh graduates, professionals seeking career growth.

Employers: Companies of all sizes, and recruitment agencies.

Functional Scope:

The project covers the following core functionalities:

For Job Seekers:

- User registration and profile creation.
- Search and apply for jobs using advanced filters (e.g., location, industry, salary).
- Resume upload and generation tools.

For Employers:

- Employer account creation and management.
- Job posting with detailed descriptions, requirements, and benefits.
- Access to a database of job seekers and resume searches.

For Administrators:

- Manage users (both job seekers and employers).
- Monitor job postings and applications to ensure compliance.
- Generate analytics and reports on portal usage.

1.7.2. Limitation

Advanced AI Matching: developing sophisticated artificial intelligence algorithms for precise job seeker-employer matching is challenging within the project's timeframe.

Comprehensive security measures: implementing highly advanced security protocols to protect user data comprehensively is complex and needs more experts than our project team possess.

Long-term maintenance planning: panning for long-term maintenance and update excluded due to the focus on development within the academic timeframe.

1.8. significance and beneficiaries of the project

This project is more than just a tool for connecting job seekers with employers; it is a catalyst for positive change in the employment ecosystem. By addressing critical gaps in accessibility, efficiency, and inclusivity, the portal has the potential to transform the way people find jobs. Its contributions to economic growth, social equity, and technological innovation underscore its significance as a project that can make a meaningful difference for individuals, organizations, and society as a whole.[3]

1.8.1. Benefits to Job Seekers

This project will serve as a lifeline for job seekers, particularly those with limited access to traditional recruitment channels. Key benefits for job seekers include:

Equal Opportunities: This project will prioritize inclusivity, ensuring that marginalized groups, such as individuals with disabilities or those from rural areas, have access to the same opportunities as others.

1.8.2. Benefits to Employers

This project will significantly enhance the recruitment process for employers, offering a range of tools and features to connect employers with job seekers. Key contributions include:

Cost reduction: The portal will reduce the costs associated with traditional recruitment methods.

Access to a wider talent pool: The platform will connect employers with a diverse range of candidates from various regions and backgrounds.

1.8.3. Benefits to Recruitment Agencies

Recruitment agencies play a vital role in connecting job seekers and employers, and this project will enhance their efficiency and effectiveness. Specific benefits include:

Improved Client Satisfaction: By enabling agencies to deliver quicker and more accurate results, the platform will help them build stronger relationships with their clients.

Expanded Reach: Recruitment agencies will be able to access a broader pool of candidates and job postings, enhancing their ability to meet the needs of both employers and job seekers.

1.8.4. Contributions to the Economy

This online job portal has the potential to create significant economic benefits by addressing inefficiencies in the job market. Its contributions include:

Enhancing Workforce Productivity: The platform will ensure that individuals are placed in roles where they can contribute effectively, boosting overall workforce productivity.

Supporting Small and Medium Enterprises: which often lack the resources to invest in extensive recruitment efforts, will benefit from the affordable and efficient solutions offered by the portal, enabling them to grow and thrive.

1.9. Project schedule

Project Schedule												
	2025/2026											
SDLS Phase	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**	**Jan**	**Feb**
Project Proposal	■											
Requirement and Analysis		■	■									
Design				■	■							
Implantation/Coding					■	■	■					
Testing								■				
Deployment/Hosting									■	■	■	■
Training											■	■

Table 1: Time schedule for work plan

1.10. Organization of the Project

This project documentation is organized into four chapters. The first chapter provides an introduction to the project. The second chapter focuses on system requirements and specifications. The third chapter covers system modeling. The fourth chapter deals with the system design.

Chapter Two: System Requirement And Specification

2.1 Existing System

Overview

In today's digital era, there is an increasing demand for online platforms that connect job seekers and employers. Existing online job portal systems typically offer a variety of web-based and mobile solutions, serving as intermediaries between job seekers and recruiters. However, despite the multitude of platforms available, they often present limitations in terms of usability, simplicity, and cost-effectiveness. Consequently, many job seekers and employers (specially in Ethiopia) still rely on traditional systems like job boards and recruitment agencies.

Key Characteristics of currently existing systems

Digital and Accessible: These systems operate online, allowing users to access them with an internet connection via websites. This broadens the reach for both job seekers and employers.

Database-Driven: They rely on robust databases to store and manage vast amounts of information, including job postings, user profiles (resumes, company details), applications, and communication records. This enables efficient searching, filtering, and matching.

Self-Service Functionality: They empower both job seekers and employers to manage their activities independently. Job seekers can create profiles, search for jobs, apply directly, and track their applications. Employers can post jobs, review applications, and manage candidates.

2.1.1. Functionality Of Existing System

An existing job portal system typically functions as a digital marketplace connecting job seekers, employers, and administrators. For **job seekers**, the core functionalities include the ability to create personal profiles and upload resumes, search and apply filters to browse job listings, submit online applications, track the status of their applications, and save preferred jobs while setting up alerts for new opportunities. **Employers** are provided with features to establish company profiles, post and manage job openings, view and manage submitted applications, and sometimes access a candidate database for proactive searching, alongside tools for communicating with applicants. Finally, **administrators** oversee the entire platform by managing users and accounts, monitoring job postings for compliance, configuring various platform settings, and generating basic reports to assess system activity.

Existing system workflow structure:

Job Seeker:

1. Sign up/in.
2. Create/update profile.
3. Search jobs.
4. View job details.
5. Apply for jobs.
6. Track applications.
7. Get job alerts.

Employer:

1. Sign up or log in.
2. Create or update company profile.
3. Post jobs.
4. Manage applications.
5. Contact candidates.
6. Search candidates (sometimes).

Administrator:

1. Log in.
2. Manage users.
3. Manage content.
4. Moderate jobs.
5. Configure system.

6. Monitor platform.

The way how existing system generate report

Online job tracking systems generate reports through a systematic process of data collection, processing, and presentation.

1. **Data Collection**: The system continuously gathers data from user interactions, database records, and system logs. Key data points include:

- ✓ User actions (logins, searches, applications, job postings)
- ✓ User profile information (skills, experience, location).
- ✓ Job posting details (views, applications, status).
- ✓ System performance metrics (page load times, error rates).
- ✓ This data is stored in the system's database, often organized into tables that track specific entities and activities.

2. Data Processing:

When a report is requested (e.g., an employer wants to see application statistics for a job posting), the system processes the raw data. This involves querying the database to retrieve the relevant information. The system may perform calculations, aggregations, and filtering to transform the data into a meaningful format[5]. For example: Counting the number of applications for each job, Calculating the average time-to-fill a position, Grouping users by location or industry. Data processing may be handled by database queries (e.g., SQL) or by server-side programming languages (e.g., Python, Java, PHP) that manipulate the data.

3. Report Generation:

Once the data is processed, the system formats it into a readable report. The report can be generated in various formats: HTML (for viewing within the web interface), PDF (for printing or downloading), CSV or Excel (for exporting the raw data). Use reporting libraries or tools to automate the report generation process.

4. **Report Delivery:** The generated report is then delivered to the user. This can happen in several ways:

- Displaying the report directly on the user's screen within the web application.
- Sending the report as an email attachment.
- Making the report available for download.
- Scheduling reports to be generated and delivered automatically at regular intervals.

2.1.2. Business Rule

1. User Privacy and Data Protection Rule:

When a user creates a profile, their resume and contact information are stored securely on its servers. It will only share their resume with employers the user has explicitly applied to, and it will never sell their personal data to third-party marketing agencies. The user can control the visibility of their profile settings in their account.

2. Content Moderation Policy:

If a job posting contains language that discriminates against candidates based on their race, gender, or disability, a user can report it using the 'Report Job' button. Its moderation team will review the report within 24 hours, and if the posting violates its policy, it will be removed, and the employer may face a warning or account suspension.

3. Job Posting Creation and Approval Rule:

When an employer submits a new job posting, the system checks for required fields (job title, description, location). If all mandatory fields are complete, the posting is typically published immediately. However, if the system flags potential issues, it may be held for manual review by its administrator before going live.

4. Customer Support Rule:

If a user has a question, they can first check its comprehensive FAQ section. If they still need help, they can click the 'Contact Us' button to send an email to its support team. It aims to respond to all inquiries within one business day. For urgent technical issues, the user can call its dedicated support hotline during business hours.

5. Accessibility Policy:

It is committed to making its platform accessible to everyone. All images on its site include descriptive 'alt text' for screen readers. Its website is designed to be navigable using only a keyboard, and it is continuously working to meet WCAG 2.1 Level AA guidelines.

2.1.3. Drawbacks The Existing Systems

Despite being feature-rich, existing systems often face several drawbacks:

User Experience Issues:

- Overwhelming interfaces, especially for mobile users.
- Cluttered navigation.
- Complex and time-consuming application processes.

Limited Support for Niche Markets:

- Focus on large industries, with fewer opportunities for blue-collar jobs.
- Regional limitations.

Mobile App Shortcomings

- Some portals lack robust, responsive mobile applications.
- Mobile versions may not offer full functionality as desktop platforms.

2.2. Proposed System

2.2.1. Overview

Given the limitations of the existing job portals, there is a strong need for a more user-friendly, and mobile-optimized platform that bridges the gap between job seekers and employers more efficiently. Our system will be a responsive web portal and mobile application (for Android) for job seekers and employers with the following capabilities:

- Provide an intuitive user interface across web and mobile.
- Provide mobile application for android users.
- Support blue-collar, and niche job markets (jobs that narrow their focus to a more defined area).
- Offer location-based personalization by Allowing users to manually set their preferred location (city, region, or postal code).

2.2.2. Functional Requirements

A functional requirement defines the specific actions the system *must perform* to serve its users, encompassing capabilities like enabling job seekers to register, build profiles, search for jobs, and track applications; allowing employers to post job listings, manage company profiles, and view applicant details; and empowering administrators to oversee user accounts, moderate content, and generate reports. The functional requirements for our system, categorized by user role as follows:

Job Seeker Functions:

- Register and log in via email or phone.
- Create and edit a user profile with skills, education, and experience.
- Upload or build a resume or CV.
- Search and filter jobs.
- Apply to jobs.
- View application status or history.

Employer or Recruiter Functions:

- Register and log in as a company or recruiter.
- Create or edit company profile.
- Post new job listings with description, qualifications, salary, and location.

- Search and filter applicants.
- View applicant profiles and resumes.
- Communicate with applicants via email.
- Manage job postings (edit, delete, expire).
- View job application statistics and insights.

Admin Functions:

- Manage user accounts (job seekers and employers).
- Approve or reject employer registrations (if needed).
- Monitor and manage posted jobs.
- Generate reports (user activity, job applications, etc.).
- Handle complaints or abuse reports.
- Manage site content (FAQs, help, about us, terms).

2.2.3. Non-Functional Requirements

Non-functional requirements specify how the system performs rather than what it does, defining critical qualities such as its responsiveness, reliability, and security[7]. This includes ensuring the system supports thousands of concurrent users with sub-5-second page loads, maintains high uptime with daily data backups, offers a user-friendly interface across devices, protects sensitive data through encryption and role-based access control, is easily scalable for future growth, boasts a maintainable codebase, and ensures compatibility with various browsers and mobile platforms.

Performance Requirements:

When it comes to performance requirements, the our system robust enough to handle high demand, specifically designed to support at least 5000 concurrent users without degradation. This means that under normal load, critical actions like page load times should not exceed 5 seconds. Furthermore, the system ensure rapid response times for key user interactions such as submitting applications or searching job listings, with data processing occurring efficiently to prevent any noticeable delays.

Reliability and Availability:

This system is designed for robust operation, targeting a minimum of 99.9% uptime to ensure consistent access for all users. To safeguard against data loss, comprehensive data backups will occur daily, enabling rapid recovery within an hour in the event of any system failure or disaster.

Usability:

Our system prioritizes a highly user-friendly and responsive UI, ensuring a seamless experience across all web browsers and diverse mobile devices. This is complemented by an intuitive navigation structure that allows users to effortlessly find information and complete tasks with minimal clicks. To further enhance the user experience, the system will provide clear and constructive feedback for all actions, employ a consistent design language throughout, and offer accessible features to accommodate users with varying needs. Additionally, features like onboarding tutorials for new users and readily available contextual help will ensure a low learning curve and high user satisfaction.

Security:

The system will implement robust measures to protect all user data and system integrity. This includes mandatory data encryption for sensitive information such as passwords, resumes, and private messages, both in transit and at rest. Access to functionalities and data will be strictly enforced through role-based access control, segmenting permissions for administrators, employers, and job seekers. The system will also incorporate comprehensive defenses against common web vulnerabilities, including protection against SQL injection, Cross-Site Scripting (XSS), and other risks. Furthermore, secure API design, regular security audits and penetration testing, and the implementation of multi-factor authentication (MFA) for privileged accounts will safeguard against unauthorized access and maintain data confidentiality and integrity. User sessions will be securely managed, and system activity will be continuously monitored for suspicious patterns.

Scalability:

This online job tracking system is engineered to adapt to increasing demand, ensuring it can easily accommodate a growing number of users, job postings, and employers without performance degradation. This is supported by a modular architecture that facilitates the seamless integration of new features. The use of cloud-native technologies, will ensure the system remains resilient and performs optimally even under significant data and user growth.

Maintainability:

The system will be built upon a foundation of clean, well-documented code, adhering to consistent coding standards to ensure ease of understanding and modification by development teams. This commitment extends to making content and features easy to update, streamlining future enhancements and bug fixes. The entire development process will rigorously utilize version control systems like Git, fostering collaborative development and providing a robust history for tracking changes.

Compatibility:

This system is designed to provide a consistent and accessible experience across various platforms. The web application will be fully compatible with all major browsers, including Chrome, Firefox, and Microsoft Edge, ensuring a broad reach. For mobile users, dedicated native applications will be compatible with Android devices.

2.3. Feasibility Study

2.3.1. Technical Feasibility

This assesses whether the technology required to develop the portal is available and suitable.

Skill set: The project team is possesses skills in web development, database management, and cyber security to implement and maintain the portal.

Technology Stack: Proven and widely-used tools such as HTML5, CSS, JavaScript (frontend), PHP/Node.js (backend), and SQL server (database) ensure reliability and scalability.

Resources: Availability of tools like IDEs (Visual Studio Code) and version control (Git) supports efficient development.

2.3.2. Operational Feasibility

This measures how well the project aligns with the operational needs of users and stakeholders.

Target Users: The portal must meet the needs of job seekers, employers, and administrators. Features like easy registration and job searches ensure usability.

User Adoption: High operational feasibility depends on the ease of use and intuitive design. The platform will offer a user-friendly interface ensures smooth adoption.

2.3.3. Economic Feasibility

This evaluates whether the project is financially compatible and cost-effective.

Initial Costs: Includes development costs, domain registration, hosting fees, and licensing for software tools.

Operational Costs: Includes employee salaries, and marketing expenses.

2.3.4. Legal Feasibility

This ensures the project complies with relevant laws and regulations.

Data Protection Laws: Meet with privacy laws, such as GDPR (General Data Protection Regulation) and local data protection act is critical to protect user data.

Intellectual Property: Avoiding copyright by using licensed technologies and content.

Terms and Conditions: Clearly defined user agreements and policies to govern platform usage.

2.3.5. Scheduling Feasibility

This aspect evaluates whether the project can be completed within the proposed timeline.

Project Timeline: A realistic timeline is developed based on project phases (data collection, requirement analysis, design, development, testing, deployment, and evaluation).

Resource Allocation: Adequate allocation of human resources and tools for each phase minimizes delays.

Chapter Three: System modelling

3.1. Use case model

- **Actors:**
  - ✓ Job Seeker
  - ✓ Employer
  - ✓ Administrator
- **Use cases:**

Job Seeker Actor:

- ✓ Register/Login
- ✓ Search Jobs
- ✓ Apply for Job
- ✓ Upload resume
- ✓ Log out

Employer Actor:

- ✓ Register/Login
- ✓ Register Company
- ✓ Post Job
- ✓ View Applications
- ✓ Log out

Administrator Actor:

- ✓ Login
- ✓ View data
- ✓ Update data
- ✓ Delete data

- ✓ Generate report
- ✓ Log out

Use case diagram:

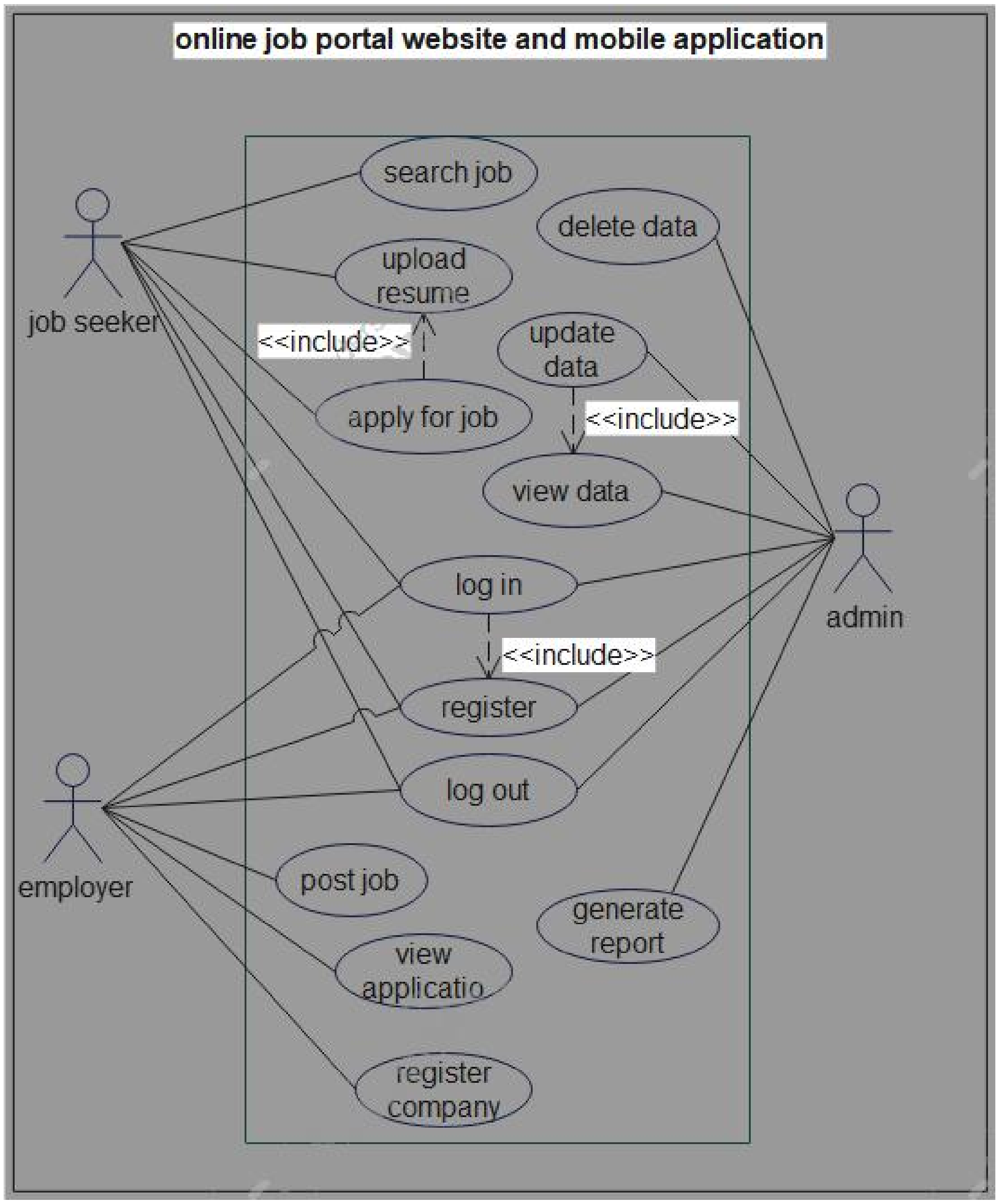

Figure 1: Use case diagram

System use case documentation:

Use case name	Register/Login	
Actor	Job seekr, Employer	
Description	Both Job Seekers and Employers either create a new account (Register) or access their existing account (Login) on the Online Job Portal through the website and mobile application.	
Precondition	The user (Job Seeker or Employer) has accessed the registration/login page or screen on either the website or mobile application. For registration, the user does not have an existing account. For login, the user has a previously registered account with valid credentials.	
Basic flow	User action 1. User Accesses Login Page/Screen 2. User Chooses Action: • Option A: New User Registration: The user selects the "Register" option. • Option B: Existing User Login: The user proceeds to enter their login credentials. 3. Flow A: New User Registration User Enters Registration Details: The system prompts the user to enter required information such as: o Name o Email Address o Password o User Type (Job Seeker/Employer) 4. Flow B: Existing User Login: User Enters information to login	System response : Flow A System Validates Input System Checks for Existing Account (Email): System Creates New Account System Sends Verification Email (Optional) System Displays User Dashboard Flow B System Authenticates User: The system compares the entered credentials with the stored credentials in the database. ✓ Login Successful: If the credentials match, the user is successfully logged in.

		✓ Login Failed: If the credentials do not match, the system displays an error message (e.g., "Invalid email or password") and may allow the user to attempt to log in again or use a "Forgot Password" option. ✓ System Displays User Dashboard
Post condition	The job seeker is authenticated and has access to their personalized dashboard.	
Alternative flow	Username Login: Instead of email, the system allow users to log in using a unique username. Separate Registration Pages	

Table 3: Use case documentation for Register/Login

Use case name	Search jobs	
Actor	Job seekr	
Description	Enables job seekers to find relevant job openings based on various criteria.	
Precondition	The job seeker is logged into the system and has access to their dashboard. The system contains a database of job postings.	
Basic flow	User action Job Seeker Accesses Search Interface Job Seeker Enters Search Criteria Job Seeker Initiates Search	System response System Displays Search Options System Processes Search Request System Retrieves Matching Job Postings

	Job Seeker Reviews Results Job Seeker Selects Job Posting	System Displays Search Results
Post condition	A list of job openings matching the search criteria is displayed to the job seeker. If no jobs match the criteria, a message indicating this is displayed.	
Alternative flow	Suggested Searches: The system provide suggestions for related job titles or keywords based on the job seeker's profile or previous searches. Search History: The system keep a history of the job seeker's previous searches.	

Table 4: Use case documentation for Search job

Use case name	Apply for job	
Actor	Job seeker	
Descriptiom	Allows job seekers to submit their application for a selected job.	
Precondition	The job seeker is logged into the system and has viewed the full details of a job they wish to apply for. The job seeker has a resume uploaded to their profile (or the system allows uploading during the application process). The job posting is currently open for applications.	
Basic flow	User action Job Seeker Views Job Details Job Seeker Initiates Application. Job Seeker Reviews Application Details Job Seeker Submits Application	System response System Presents Application Options System Retrieves Job Seeker's Resume System Validates Application System Saves Application Data

Post condition	The job seeker's application, including their resume and any other required information, is submitted to the system and associated with the specific job posting. The job seeker receives confirmation that their application has been submitted.
Alternative flow	Application Limits: The system limits the number of applications a job seeker can submit for a specific job or within a certain time frame. Withdraw Application: The system allow job seekers to withdraw their application before the employer takes action.

Table 5: Use case documentation for Apply for job

Use case name	Upload resume	
Actor	Job seeker	
Description	Allows a logged-in job seeker to upload their resume file to the system. This resume will be stored and associated with their profile and can be used when applying for jobs.	
Precondition	The Job Seeker must be successfully logged into the system.	
Basic flow	User action The Job Seeker enters to their profile settings to upload a resume during the job application process (if no resume is already uploaded). The Job Seeker selects a resume file from their local device. The Job Seeker initiates the upload process (e.g., by clicking an "Upload" button). The Job Seeker can now view or manage their uploaded resume (e.g., download, replace) through their profile settings.	System response The system displays the resume upload interface. The system receives the selected file. The system validates the file: Checks if the file size is within the allowed limit. Checks if the file format is among the supported formats (e.g., PDF, DOC, DOCX). If the file is valid, the system stores the resume file securely on the server and links it to the Job Seeker's user account. The system displays a confirmation message to the Job Seeker indicating that

		the resume has been uploaded successfully.
Post condition	The resume file is successfully uploaded and stored in the system. The uploaded resume is associated with the Job Seeker's profile. The Job Seeker receives confirmation of the successful upload.	
Alternative flow	Replacing Existing Resume. Upload Error: unexpected error during uploading resume. Unsupported File Format Invalid file size	

Table 6: Use case documentation for Upload Resume

Use case name	Log Out	
Actor	Job seeker, Employer, Adminstrator	
Description	This use case allows a currently logged-in user (regardless of their role) to securely terminate their active session on the online job portal website and/or mobile application.	
Precondition	The actor (Job Seeker, Employer, or Administrator) must be currently logged into the system.	
Basic flow	User action The logged-in actor (Job Seeker, Employer, or Administrator) initiates the log-out action.	System response The system receives the log-out request. The system logs the actor out, preventing further access to authenticated areas of the platform without logging in again. The system redirects the actor to a public page.
Post condition	The actor's current session is successfully terminated. Any session-specific data or cookies associated with the logged-in session are	

	invalidated or cleared. The system redirects the actor to a public area of the platform, typically the login page or the homepage in a logged-out state.
Alternative flow	No Active Session System Error

Table 7: Use case documentation for Log Out

Use case name	Register company	
Actor	Employer	
Description	This use case enables a logged-in employer to register their company within the online job portal. This registration process is typically required before the employer can post job openings.	
Precondition	The Employer actor must be successfully logged into the system. The Employer should not have already registered a company profile	
Basic flow	User action	System response
	The Employer logs in for the first time or enters to the "Register Company," of the platform. The Employer fills out the required fields in the registration form. The Employer submits the form by clicking a "submit" button.	The system displays the company registration form. This form typically includes fields for:(Company Name, Industry/Sector, Company Size (number of employees), Company Location (address, city, country), Company Description (a brief overview of the company), Company Website (optional) , Contact Information (for the company representative), Company Logo (image upload) - Optional) The system receives the submitted data. The system validates the entered information: If the data is valid, the system stores the company information in the database, associating it with the Employer's user account. The system displays a confirmation message to

		the Employer, indicating that the company registration was successful.
Post condition	The company's information is successfully stored in the system's database. The company profile is associated with the Employer's user account. The Employer receives confirmation of the successful company registration. The Employer is typically granted access to features that require a registered company, such as posting jobs.	
Alternative flow	Missing Required Fields Invalid Data Format Duplicate Company Name System Error	

Table 8: Use case documentation for Register Company

Use case name	Post job	
Actor	Employer	
Description	To create and publish a new job advertisement on the online job portal, making it visible to potential candidates.	
Precondition	The Employer has a registered and active account on the job portal (website or mobile application). The Employer is logged into their account. The Employer has sufficient permissions to post jobs (if applicable based on subscription or account type).	
Basic flow	User action	System response
	The Employer navigates to the "Post a Job" section on the website or mobile application.	The system presents the Employer with a job posting form.
	The Employer fills in the required job details, including: (Job Title, Company	The system performs client-side validation on the entered data

	Name, Job Location (city, state/region, country), Job Category/Industry, Job Type (e.g., Full-time, Part-time, Contract, Internship), Salary/Compensation (optional but recommended), Job Description (detailed responsibilities, requirements, benefits), Required Skills, Education Level, Experience Level, Application Deadline, Contact Information (email, application URL, etc.) The Employer reviews the entered job details. The Employer clicks the "Submit" or "Post Job" button.	(e.g., required fields are filled, email format is correct). The system performs server-side validation on the submitted data. The system saves the job advertisement details in the database. The system displays a success message to the Employer, confirming the job has been posted. The system makes the job advertisement searchable and visible to job seeker.
Post condition	The new job advertisement is successfully created and stored in the system's database. The job advertisement is visible to job seekers according to the portal's visibility settings.	
Alternative flow	Incomplete Required Fields Invalid Data Format System Error Employer Cancels Posting	

Table 9: Use case documentation for Post Job

Use case name	View application
Actor	Employer
Description	To access and review the details of a specific job application submitted by a job seeker for a job posted by the Employer.
Precondition	The Employer has a registered and active account on the job portal (website or mobile application).

	The Employer is logged into their account. The Employer has posted at least one job that has received applications. The Employer has navigated to a list of applications for one of their posted jobs. The specific application the Employer wants to view is present in the list of applications.	
Basic flow	User action The Employer navigates to the "Applications" section on the website or mobile application. The Employer selects a specific job for which they want to view applications. The Employer selects a specific application from the list to view its details. The Employer reviews the application information.	System response The system displays a list of jobs posted by the Employer. The system displays a list of applications received for the selected job. The system retrieves the application details. The system displays the retrieved application details to the Employer in a readable format.
Post condition	The system displays the detailed information of the selected job application to the Employer. The Employer can review the applicant's profile information, submitted documents (e.g., resume, cover letter), and answers to any screening questions.	
Alternative flow	No Applications Received Application Not Found Error Retrieving Application Details	

Table 10: Use case documentation for View application

Use case name	View data
Actor	Adminstrator

Description	To allow the administrator to view various data stored within the system for monitoring, analysis, and management purposes.	
Precondition	The Administrator is logged into the administrative panel.	
Basic flow	User action Administrator Navigates to Data Section Administrator Selects Data View Administrator Interacts with Data Display (Optional)	System response System Retrieves Data System Displays Data
Post condition	The Administrator is presented with the requested system data in a readable and navigable format. The Administrator interact with the data display (filtered, sorted, searched, viewed details).	
Alternative flow	No Data Available Invalid Data Selection	

Table 11: Use case documentation for View data

Use case name	Update data	
Actor	Administrator	
Description	To allow the administrator to modify existing data records within the system to ensure accuracy, make necessary adjustments, or manage the platform's content.	
Precondition	The Administrator is logged into the administrative panel and has successfully viewed the data record they intends to update.	
Basic flow	User action Administrator Selects Data Record for Update Administrator Modifies Data Fields	System response System Presents Data Modification Interface System Updates Data in Database System Provides Feedback to

		Administrator
Post condition	Successful Update Unsuccessful Update Update Cancelled	
Alternative flow	System Error During Update Administrator Cancels Update Invalid input	

Table 12: Use case documentation for Update data

Use case name	Delete data	
Actor	Adminstrator	
Description	To allow the administrator to permanently remove existing data records from the system when they are no longer needed, are inaccurate, or for data management purposes.	
Precondition	The Administrator is logged into the administrative panel and has successfully viewed the data record they intends to delete.	
Basic flow	User action Administrator Selects Data Record for Deletion Administrator Confirms Deletion	System response System Presents Confirmation Prompt Before proceeding with the deletion. System Deletes Data from Database System Provides Feedback to Administrator.
Post condition	Successful Deletion Deletion Cancelled Deletion Prevented (Due to Constraints or Authorization)	
Alternative flow	Administrator Cancels Deletion Deletion of Multiple Records (Bulk Delete)	

	System Error During Deletion

Table 13: Use case documentation for Delete data

Use case name	Generate report	
Actor	Administrator	
Description	To allow the administrator to create and retrieve reports on various aspects of the platform's data and usage for analysis, monitoring, and decision-making.	
Precondition	The Administrator is logged into the administrative panel.	
Basic flow	User action Administrator Navigates to Report Generation Section. Administrator Selects Report Type. Administrator Initiates Report Generation.	System response System Presents Report Selection Options User Statistics (e.g., total registered users, new users over a period, active users). Job Posting Statistics (e.g., total job postings, new postings over a period, top industries, top locations). Application Statistics (e.g., total applications, applications per job, application status trends). Employer Statistics (e.g., total registered employers, new employers, employers by industry). System Usage Reports (e.g., login frequency, most used features). System Retrieves and Processes Data. System Presents the Generated Report
Post condition	Successful Report Generation Report Generation Failed No Data Available:	

Alternative flow	No Data for Report Error During Report Generation Long Report Generation Time (Large Datasets)

Table 14: Use case documentation for Generate report

3.2. Conceptual Modeling (Analysis Level Class Diagram)

Conceptual modeling (analysis-level class diagram) is about creating a simple visual map of the essential things (concepts) in the real-world problem trying to solve with software and how they relate to each other. It helps everyone understand the basics before getting into technical details.[1]

Classes:

User: Represents any person interacting with the system.

Attributes: userID, name, email, password, phone, address, userType

Operations: register, login, updateProfile, viewProfile, changePassword, logout

JobSeeker: A specialized User seeking employment.

Attributes: jobSeekerID, resume, skills, education, experience

Operations: searchJobs, applyForJob, viewApplications, uploadResume, manageResume

Employer: A specialized User who posts job openings.

Attributes: employerID, companyName, companyWebsite, contactPerson

Operations: registerCompany, postJob, viewApplication, manageJobs, contactCandidates

Admin: A specialized User who manages the system.

Operations: viewData, updateData, deleteData, generateReport, manageUsers, viewStatistics

Job: Represents a job posting by an employer.

Attributes: jobID, title, description, company, location, salary, jobType, postedDate, status, skillsRequired

Operations: viewDetail, apply

Application: Represents a job seeker's application for a specific job.
Attributes: applicationID, jobSeeker, job, applicationDate, status
Operations: viewApplication, updateStatus

Resume: Holds the detailed information of a JobSeeker's resume.
Attributes: resumeId, objective, education, experience, skills, certifications, projects
Operations: addEducation, addExperience, addSkills, addCertification, addProject, updateResume

Relationships:

- ✓ JobSeeker, Employer, and Admin inherit from User.
- ✓ A User can be associated with many Jobs.
- ✓ An Employer posts many Jobs.
- ✓ A JobSeeker creates many Applications.
- ✓ An Application belongs to one JobSeeker and one Job.
- ✓ A JobSeeker has one Resume.
- ✓ An Employer is associated with a Company.

3.3. key abstractions with CRC analysis

In the development of a robust and scalable online job tracking website and mobile application, identifying and defining is the system's fundamental building blocks. This involves the process of abstraction, where complex system functionalities are simplified into manageable components.[2] This document outlines the key abstractions identified within our online job portal project and details the application of Class-Responsibility-Collaboration (CRC) analysis in defining these abstractions. CRC analysis is an object-oriented design technique that facilitates the identification of classes, the assignment of responsibilities to those classes, and the understanding of how those classes collaborate to fulfill system requirements.

To clarify the essential components and their responsibilities, we employed CRC cards, a technique used in CRC analysis.

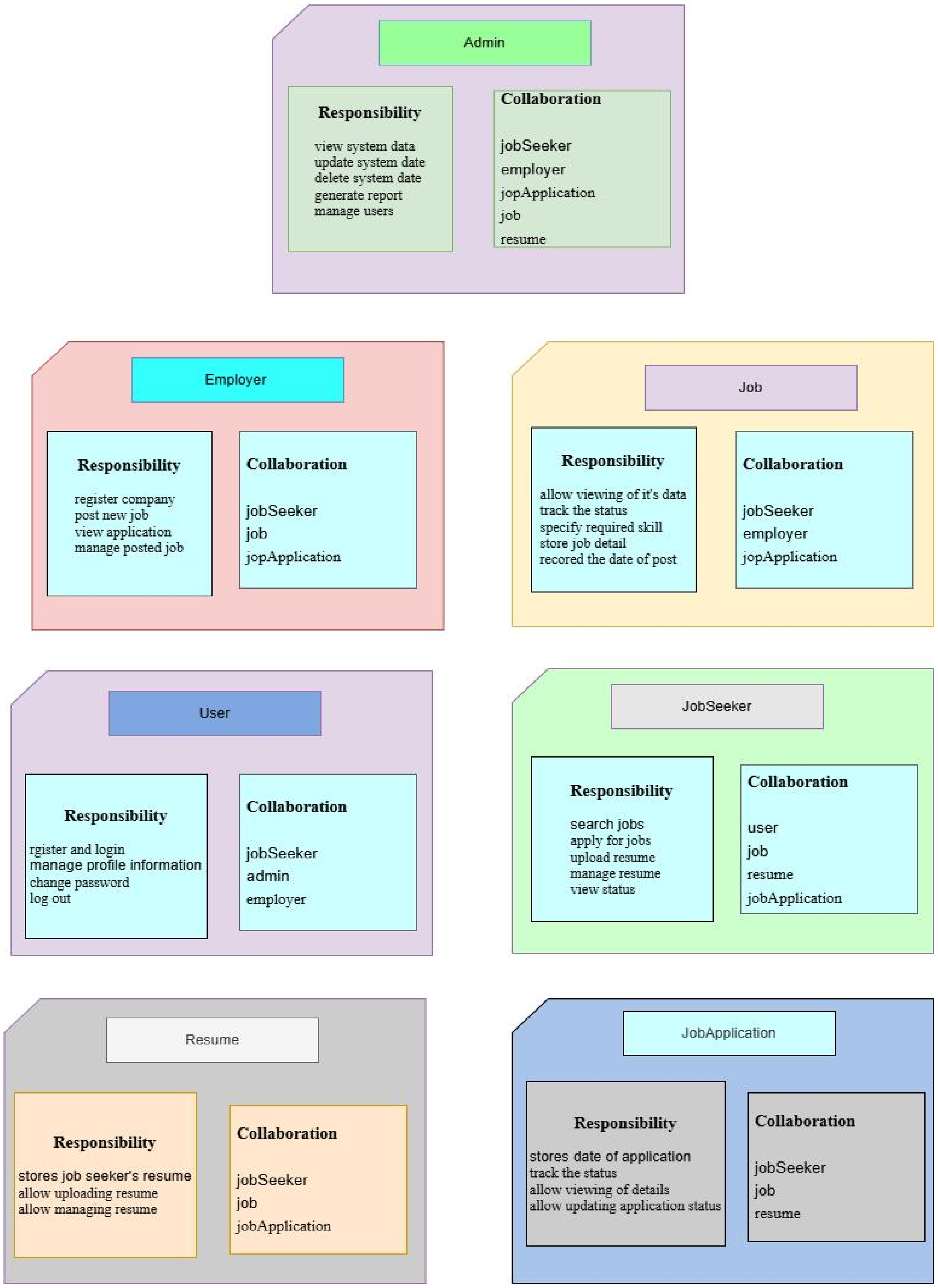

Figure 2: CRC cards

3.4 Object diagram

An object diagram is a snapshot of the instances (objects) within a system at a specific point in time. It is a photograph showing the objects and their relationships as they exist during a particular moment of the system's execution. Unlike class diagrams, which describe the blueprint or structure of objects, object diagrams show actual, concrete instances of those classes.

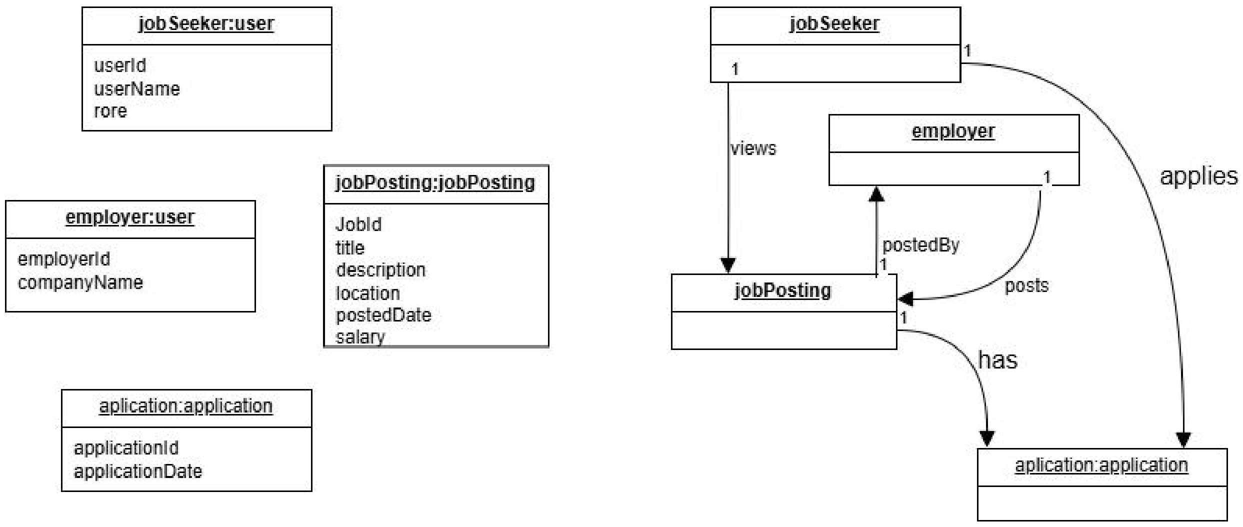

Figure 3: Object diagram

3.5. Dynamic Modeling

Dynamic modeling focuses on illustrating how a software system changes and behaves over time in response to internal and external events. Unlike static modeling, which describes the structure of a system at a particular point in time, dynamic modeling depicts the system's behavior and interactions of its components as it evolves.[4]

3.5.1. Sequence diagram

A **sequence diagram** is a type of behavioral diagram within the Unified Modeling Language (UML) that visually represents the interactions between different objects in a system over time. It focuses on illustrating the chronological order of messages exchanged between objects to accomplish a specific task or scenario. It is a timeline for each participating object, showing the sequence of messages they send and receive. This makes it excellent for understanding the dynamic behavior of a system, particularly how different components collaborate to achieve a particular goal, like fulfilling a use case.[4]

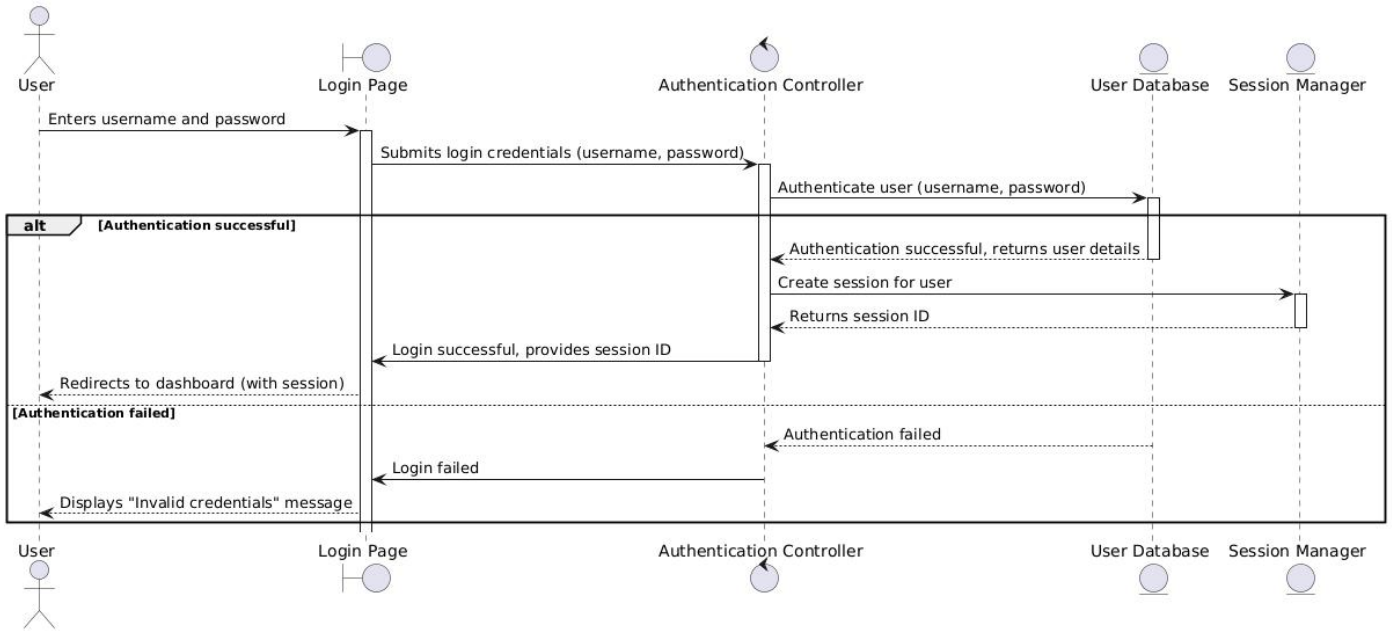
User
Login Page
Authentication Controller
User Database
Session Manager
Enters username and password
Submits login credentials (username, password)
Authenticate user (username, password)
alt
[Authentication successful]
Authentication successful, returns user details
Create session for user
Returns session ID
Login successful, provides session ID
Redirects to dashboard (with session)
[Authentication failed]
Authentication failed
Login failed
Displays "Invalid credentials" message

Figure 4: Sequence diagram for login

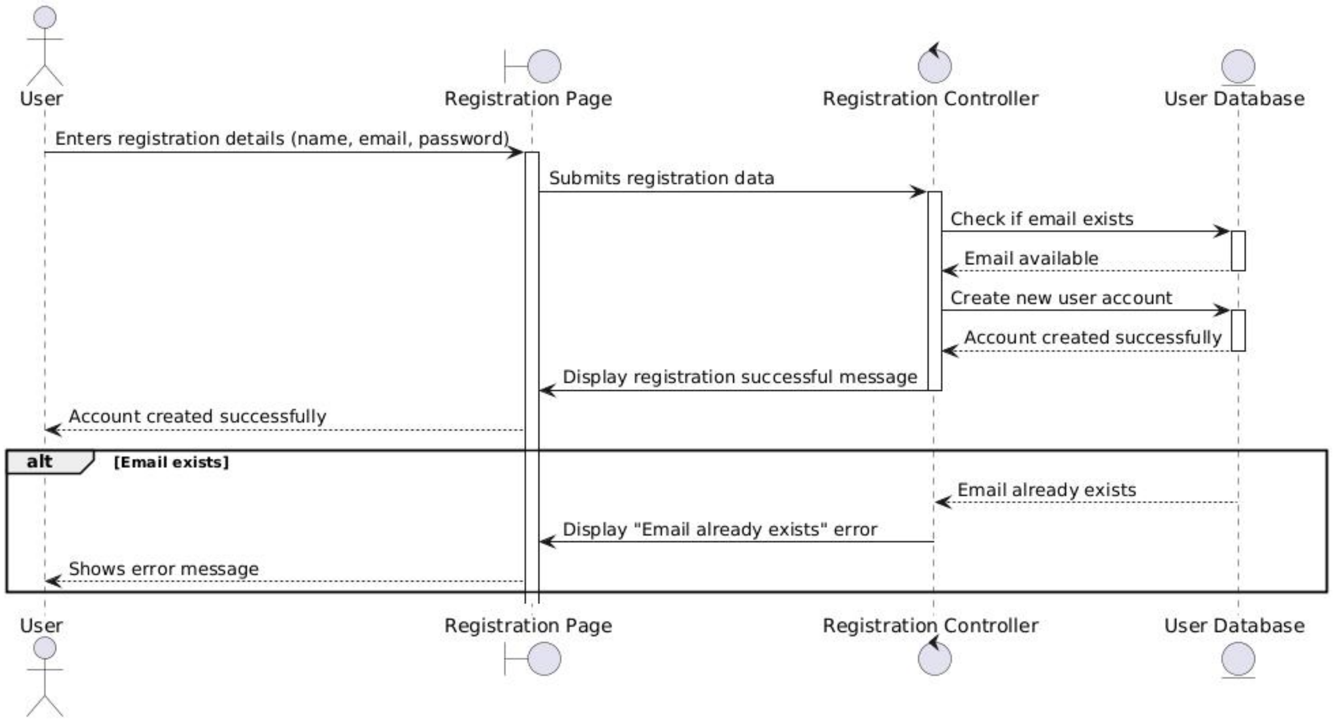
User
Registration Page
Registration Controller
User Database
Enters registration details (name, email, password)
Submits registration data
Check if email exists
Email available
Create new user account
Account created successfully
Display registration successful message
Account created successfully
alt
[Email exists]
Email already exists
Display "Email already exists" error
Shows error message

Figure 5: Sequence diagram for registration

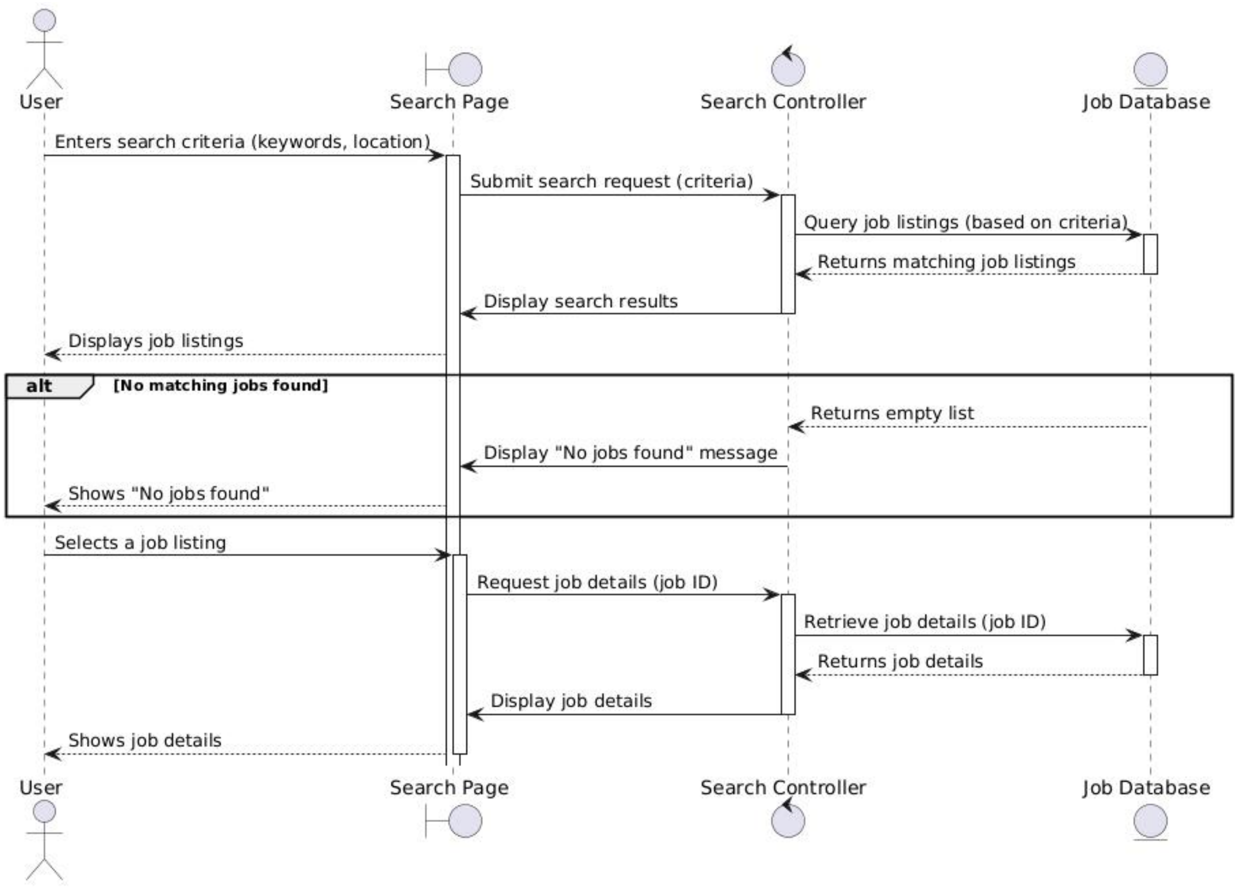
User
Search Page
Search Controller
Job Database
Enters search criteria (keywords, location)
Submit search request (criteria)
Query job listings (based on criteria)
Returns matching job listings
Display search results
Displays job listings
alt
[No matching jobs found]
Returns empty list
Display "No jobs found" message
Shows "No jobs found"
Selects a job listing
Request job details (job ID)
Retrieve job details (job ID)
Returns job details
Display job details
Shows job details

Figure6: Sequence diagram for searching job

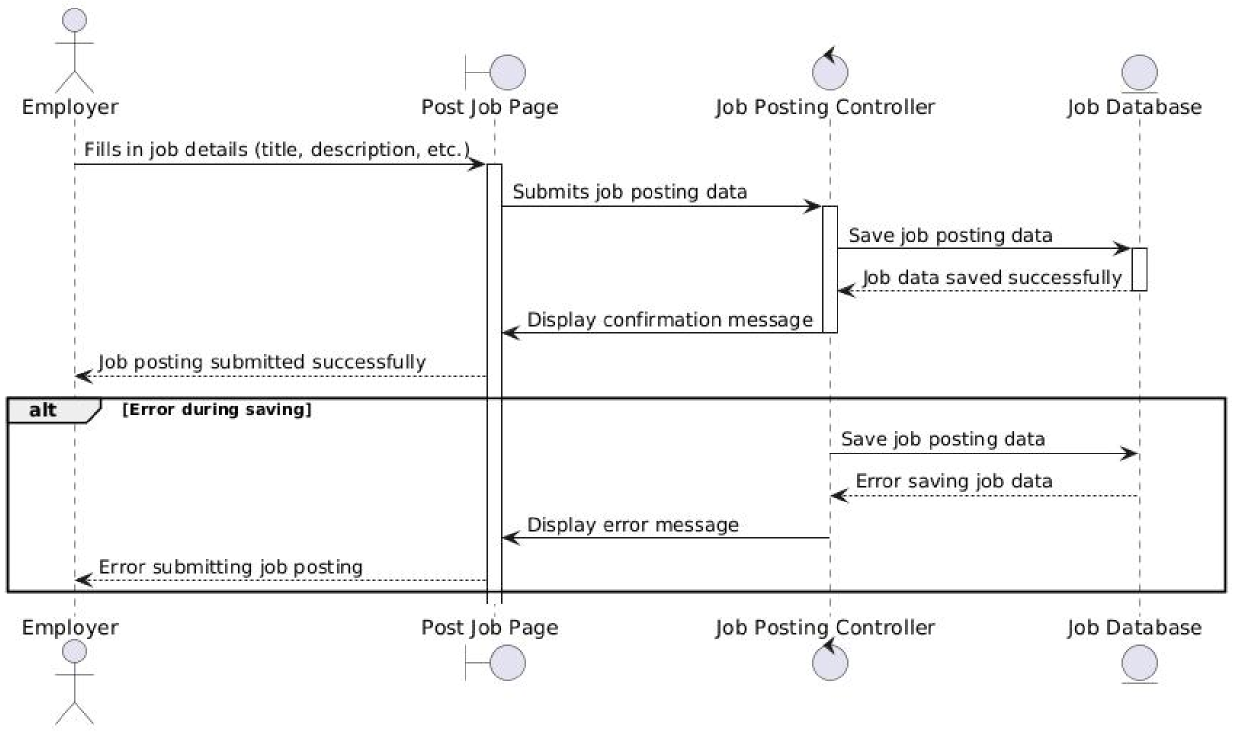

Figure 7: Sequence diagram for posting job

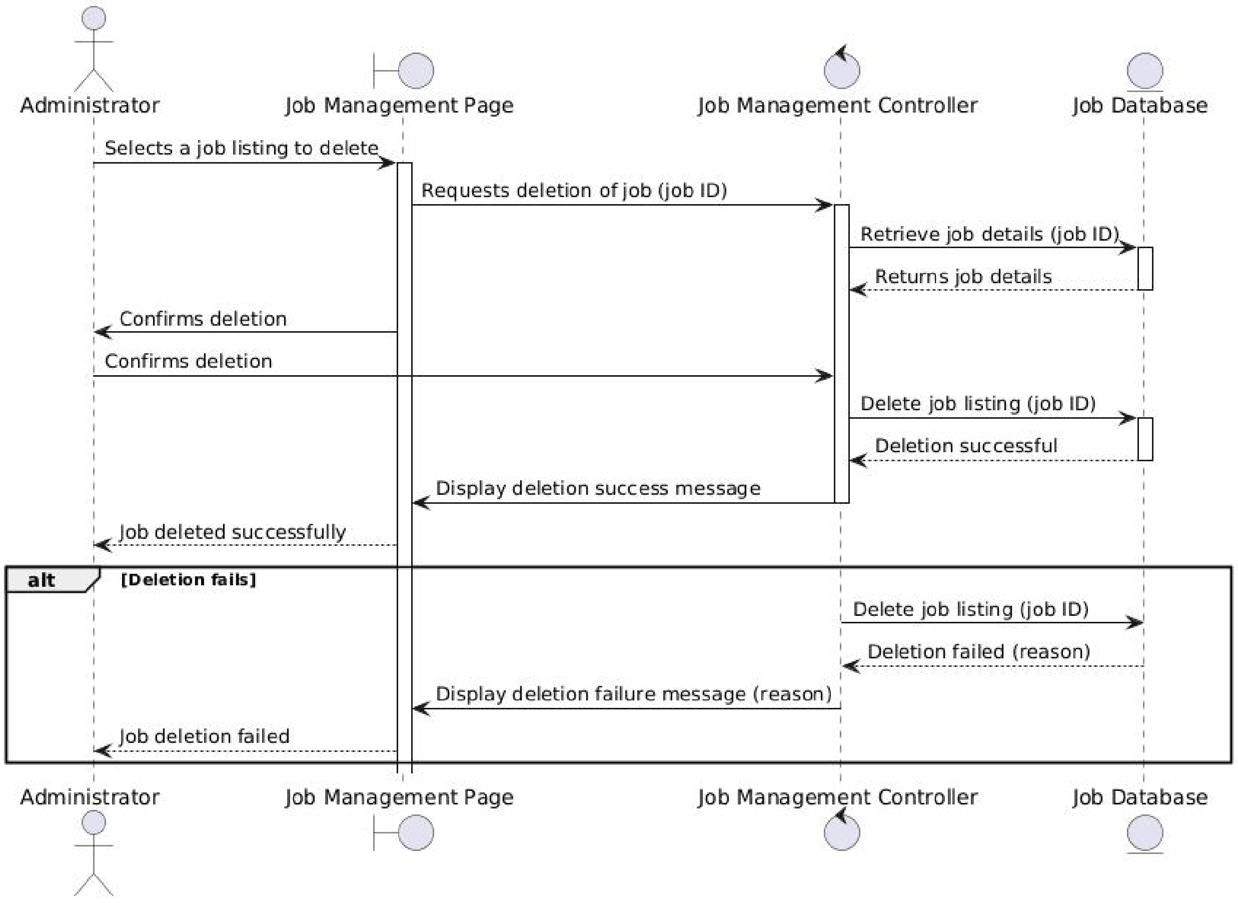
Administrator
Job Management Page
Job Management Controller
Job Database
Selects a job listing to delete
Requests deletion of job (job ID)
Retrieve job details (job ID)
Returns job details
Confirms deletion
Confirms deletion
Delete job listing (job ID)
Deletion successful
Display deletion success message
Job deleted successfully
alt
[Deletion fails]
Delete job listing (job ID)
Deletion failed (reason)
Display deletion failure message (reason)
Job deletion failed
Administrator
Job Management Page
Job Management Controller
Job Database

Figure 8: Sequence diagram for deleting job

3.5.2. Activity Diagram

An activity diagram is another type of behavioral diagram within the Unified Modeling Language (UML) that visually represents the flow of activities within a system or process. It's essentially a flowchart that illustrates the sequence of actions, decisions, parallel processing, and control flow as the system performs its tasks. It is a way to map out the different steps involved in a specific use case, businss process, or even the internal logic of a complex operation.[4]

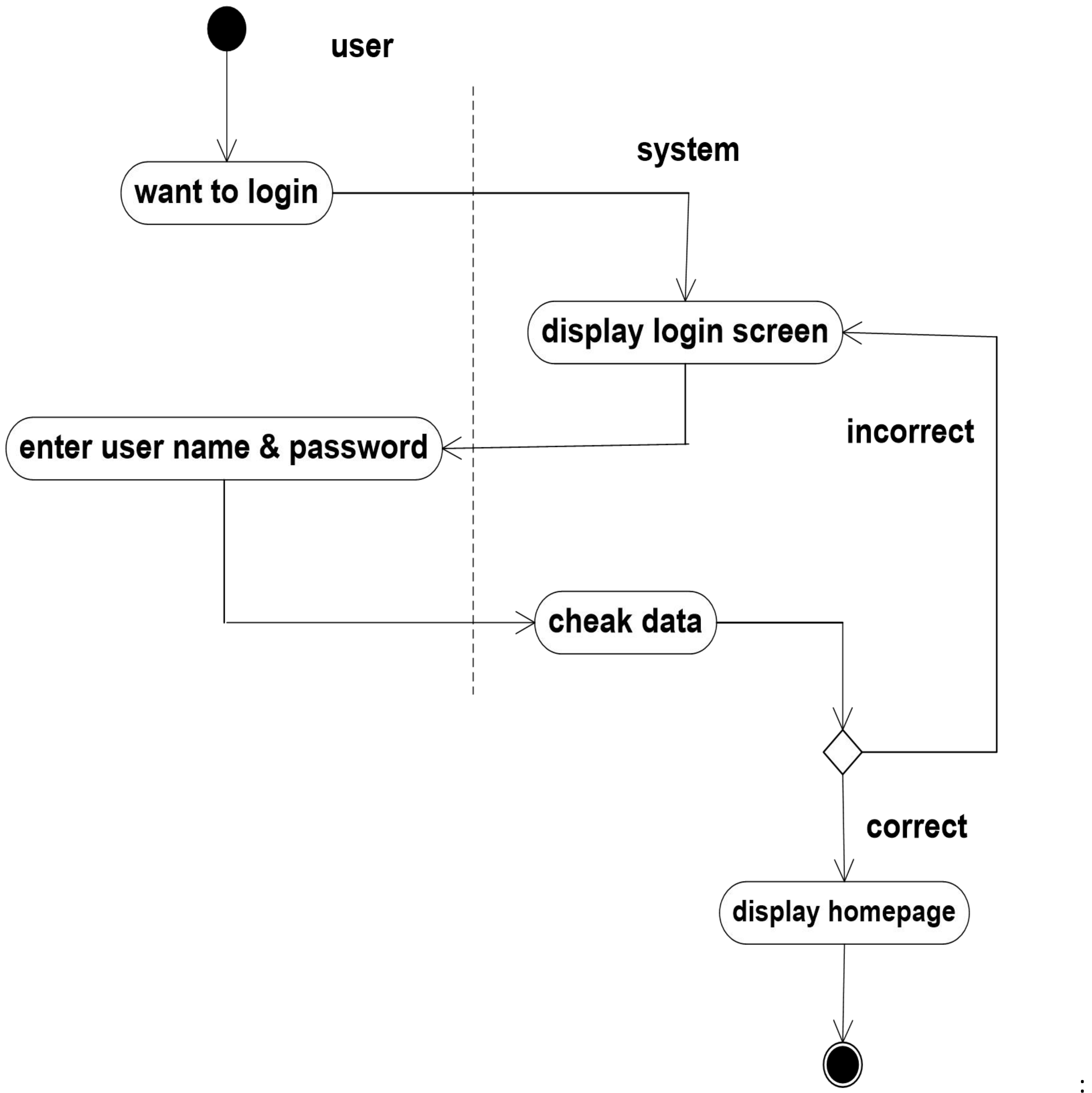

Figure 9: Activity diagram for login

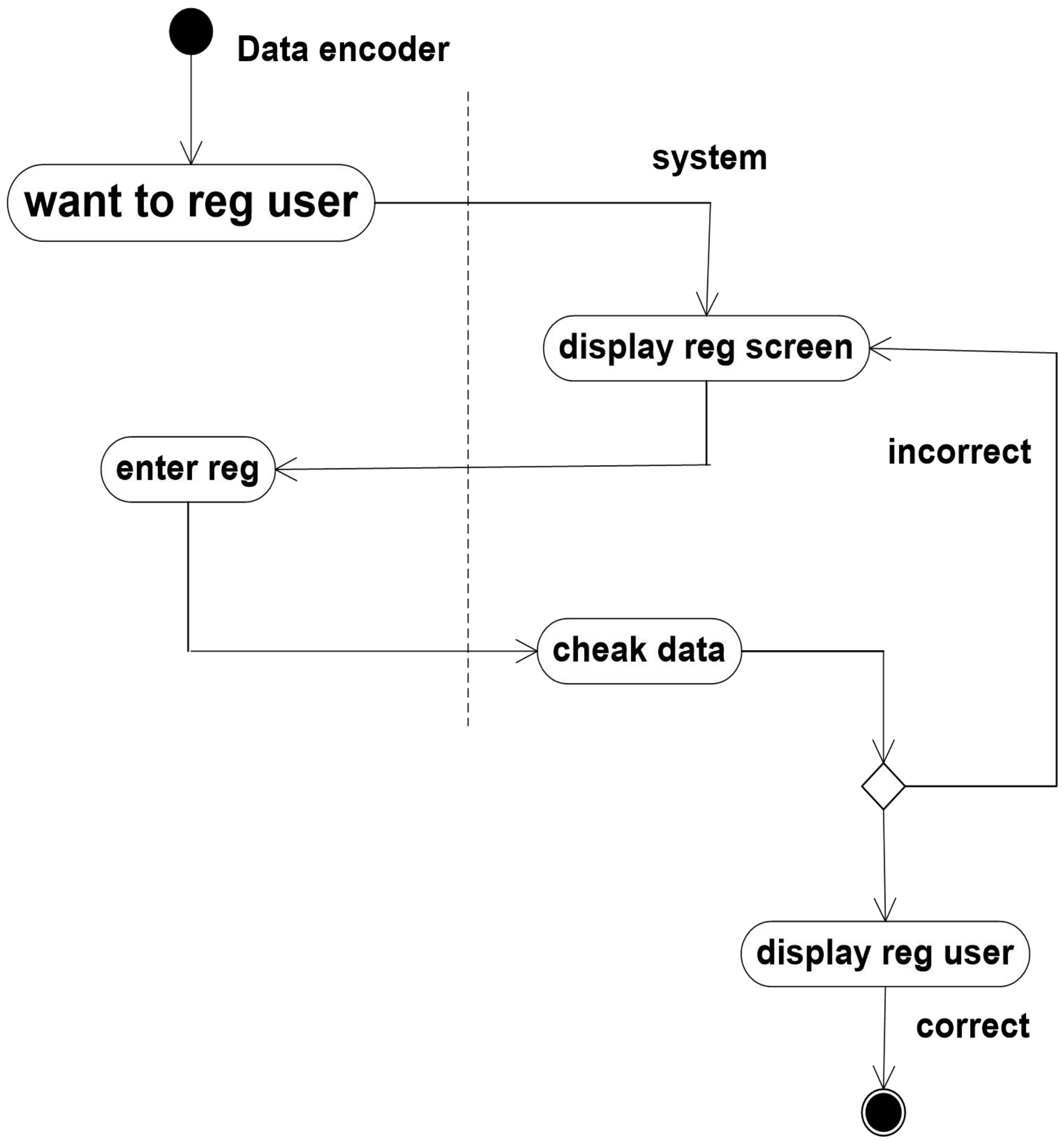

Figure 10: Activity diagram for register

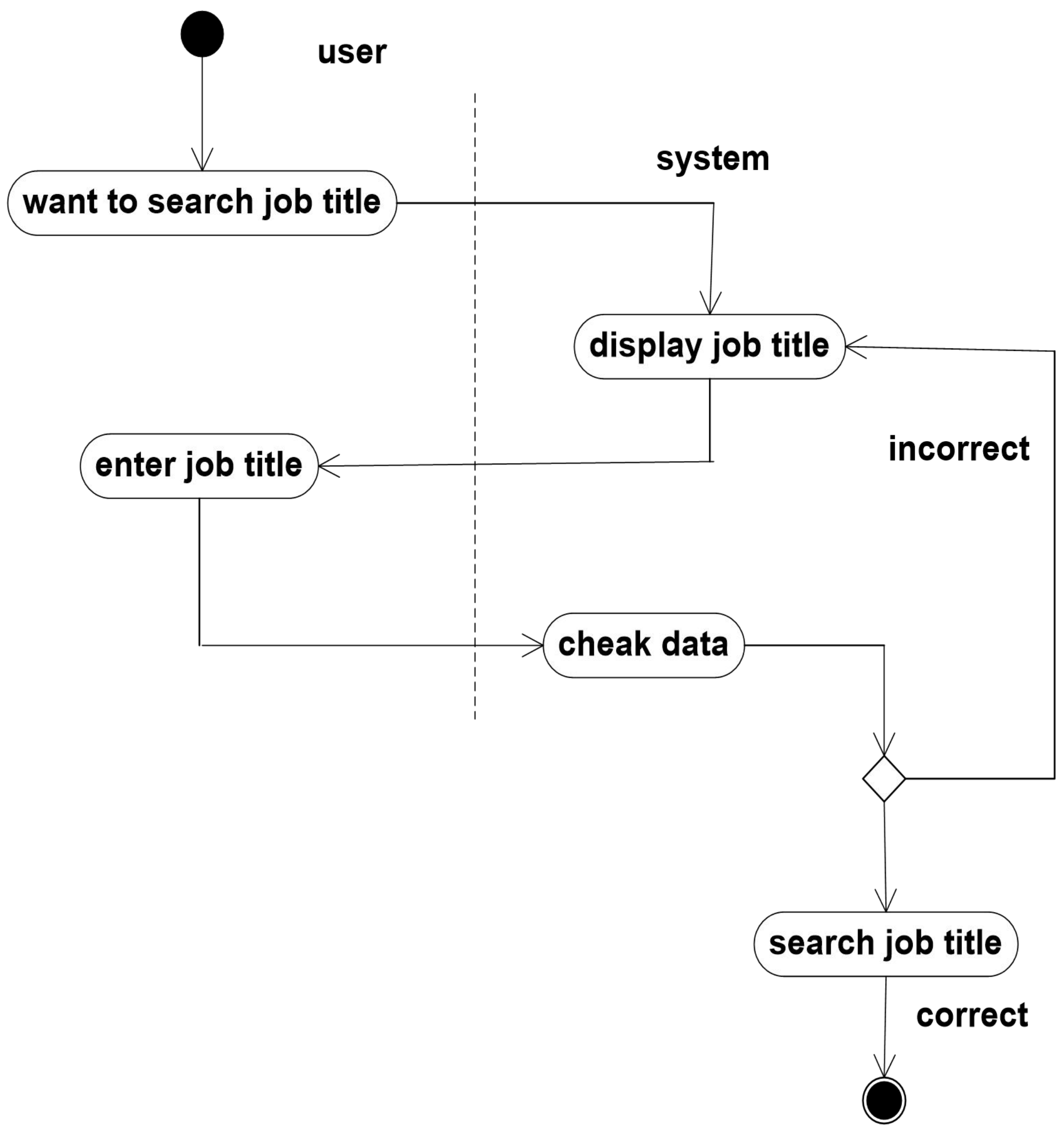

Figure 11: Activity diagram for search job

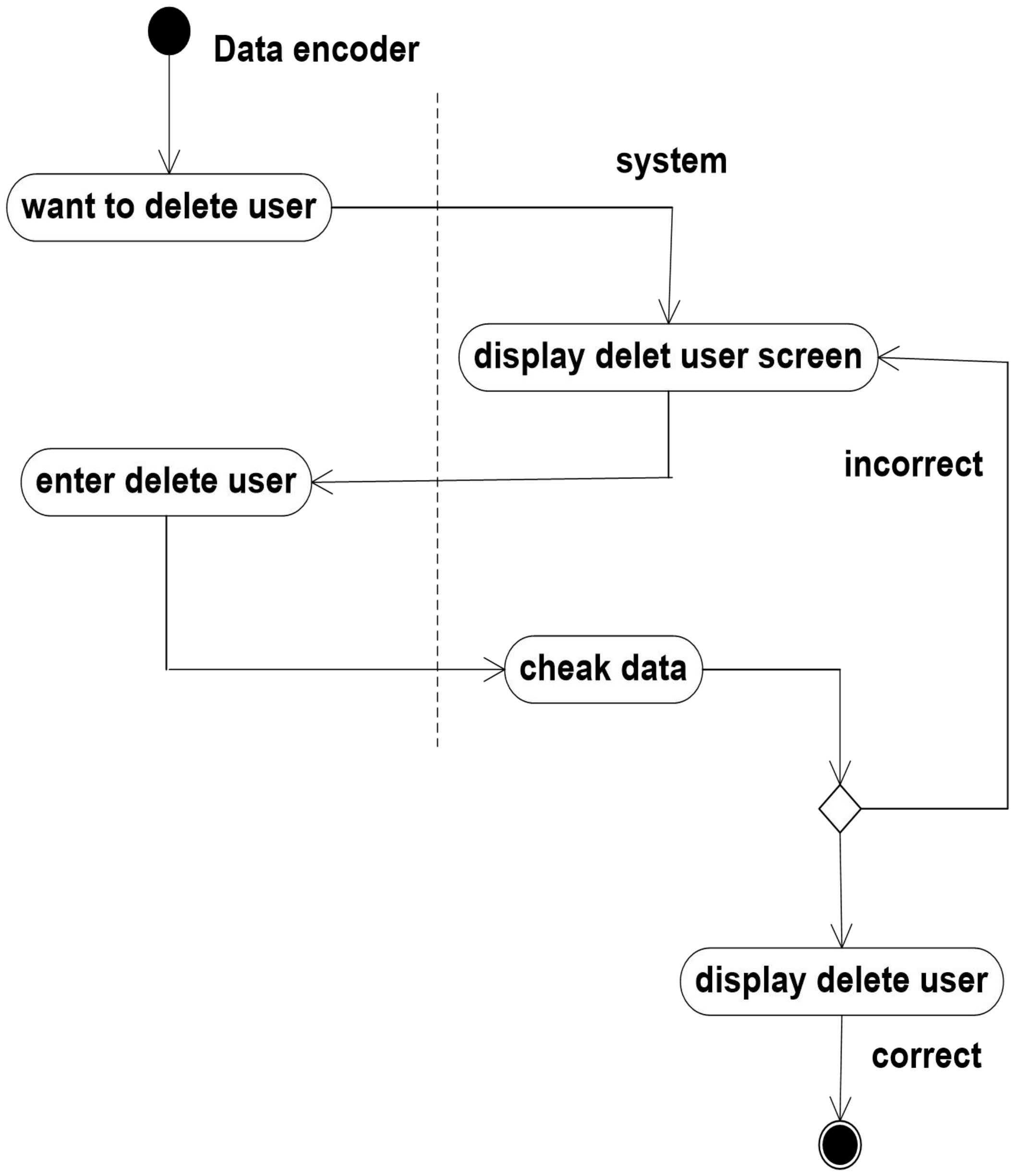

Figure 12: Activity diagram for delete

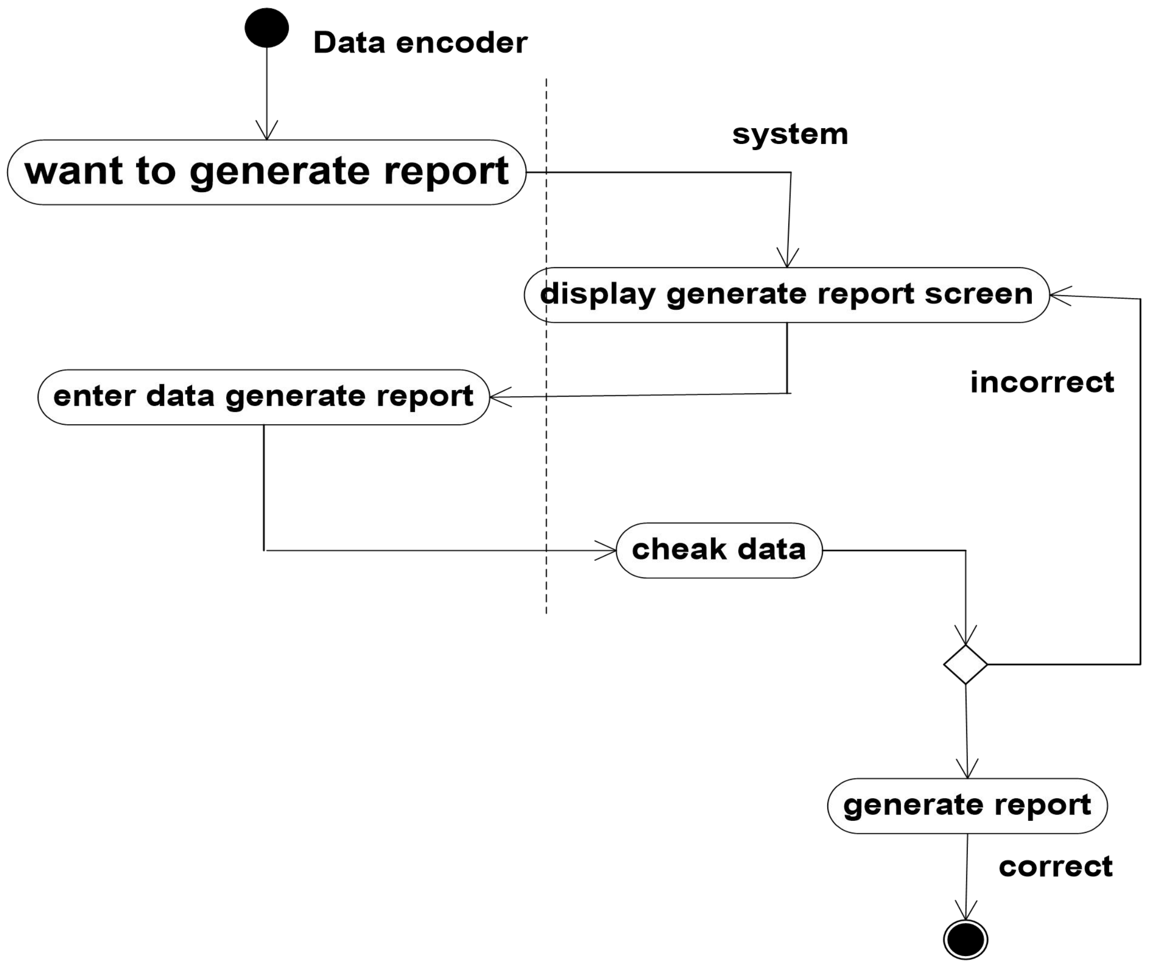

Figure 13: Activity diagram for generate report

3.5.3. State Chart Diagram

A State Chart Diagram is a visual modeling technique used to describe the different states that an object or a system can be in, and how it transitions between these states in response to various events. It's a crucial part of the Unified Modeling Language (UML) and is particularly useful for understanding and designing the dynamic behavior of software components, user interfaces, or entire systems that exhibit complex, event-driven behavior. It is like a roadmap for the lifecycle of a software entity**.** It shows all the possible conditions that entity can be in and what causes it to move from one condition to another.

State Chart Diagrams, in contrast to Activity Diagrams, model the lifecycle of an object, emphasizing its various states and the transitions between them triggered by events, illustrating how an object's behavior changes over time in response to stimuli. Whereas Activity Diagrams model the flow of activities within a process or workflow, focusing on the sequence of tasks and control flow, including parallel and conditional execution, to visualize how a process unfolds from start to finish. Unlike State Charts, which are object-centric and event-driven, detailing an object's behavioral states, Activity Diagrams are process-centric and flow-driven, outlining the steps of a procedure.

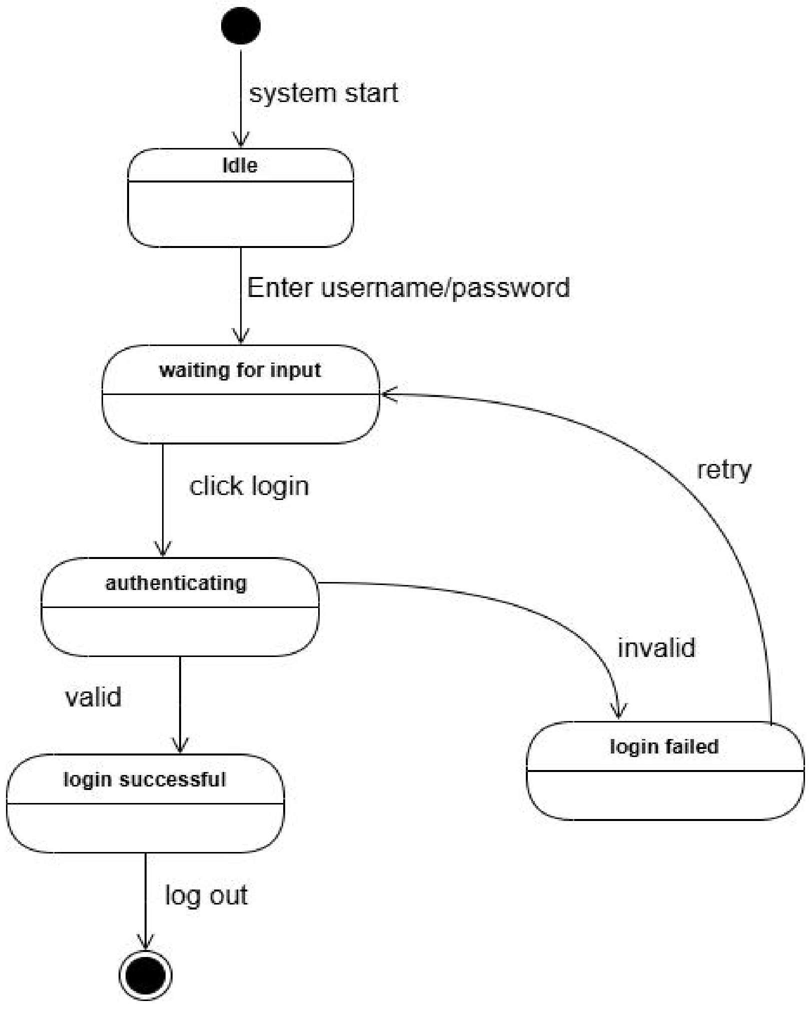

Figure 14: State chart diagram for login

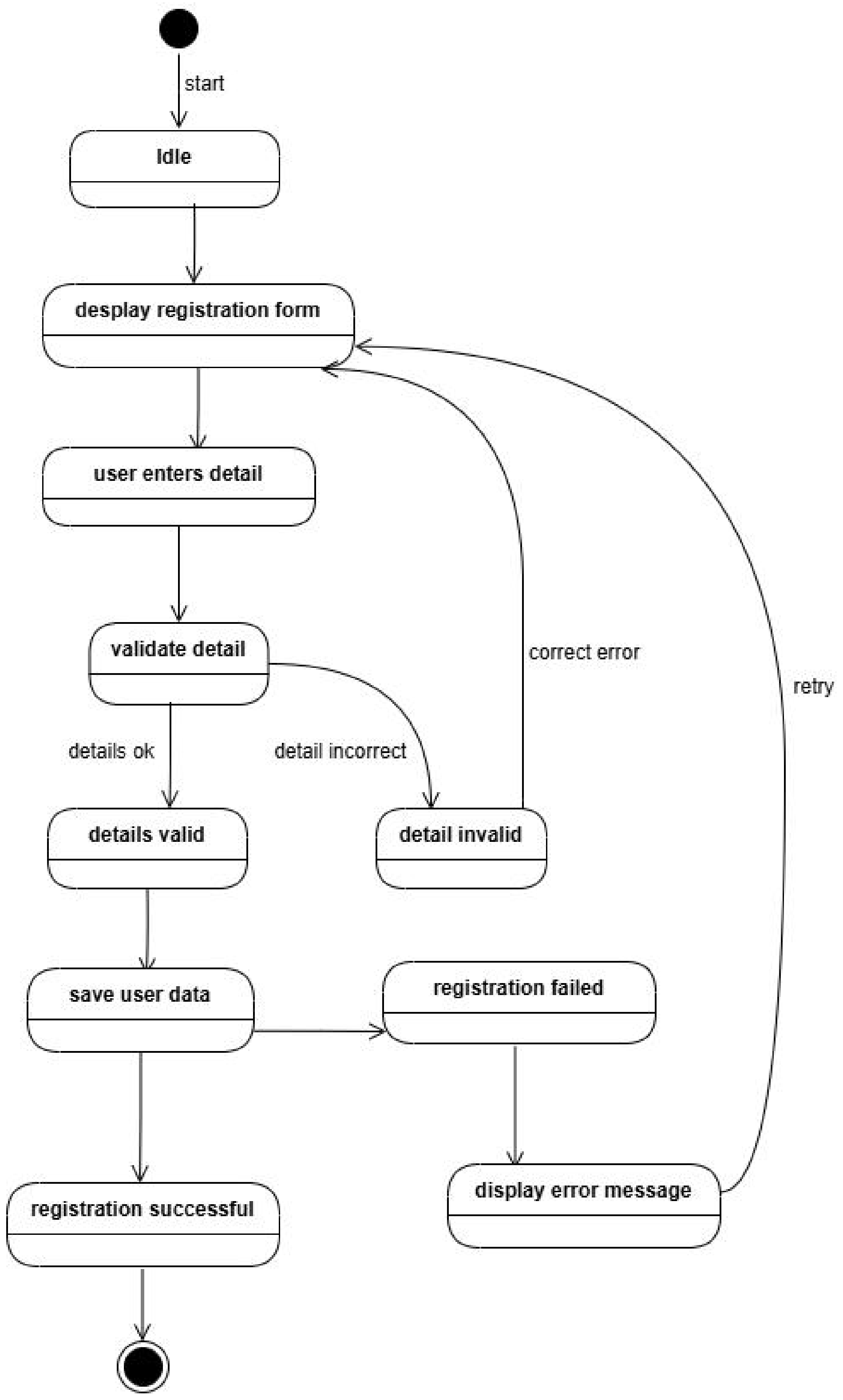

Figure 15: State chart for regiter

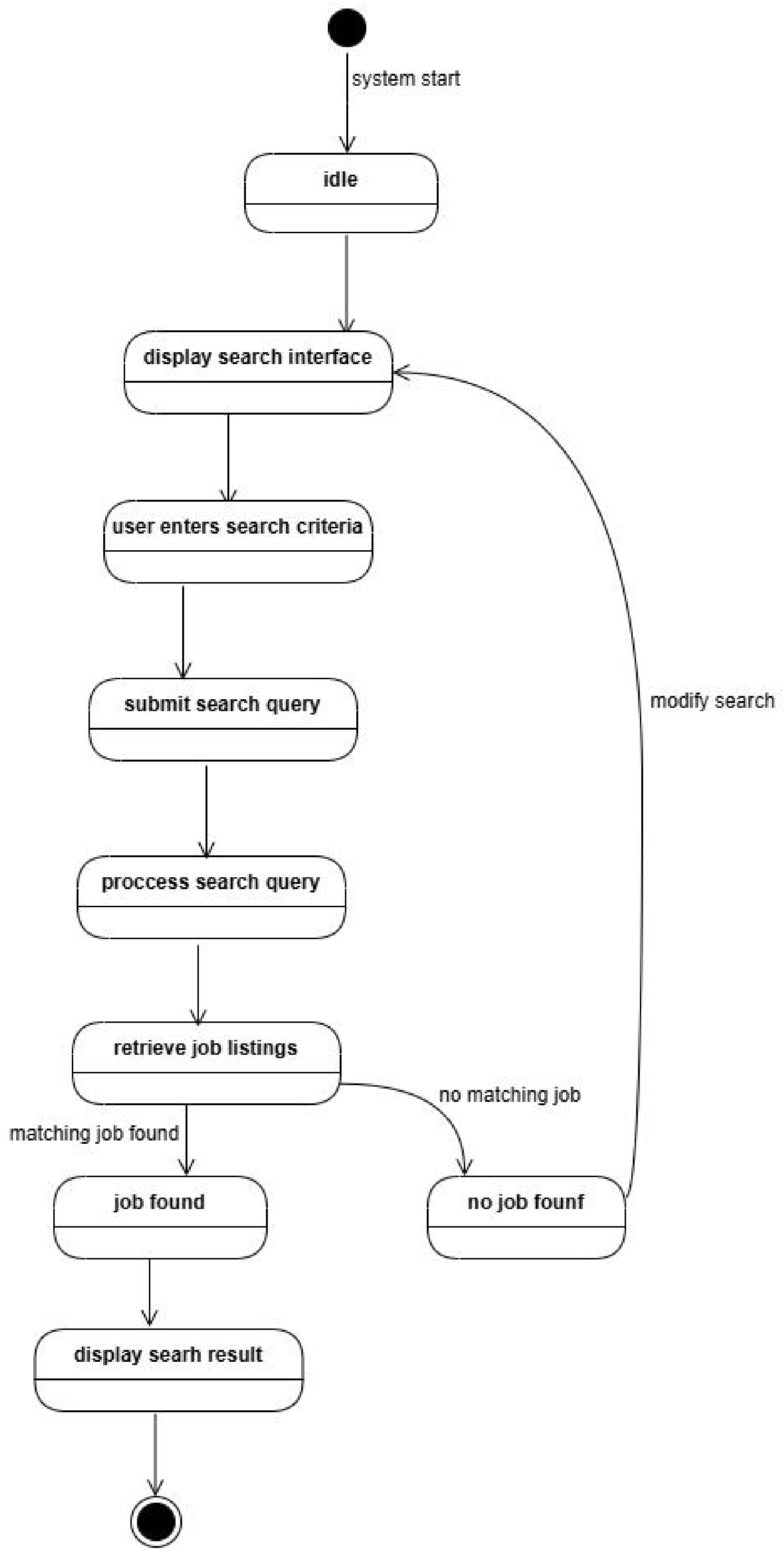

Figure 16: State chart diagram for search job

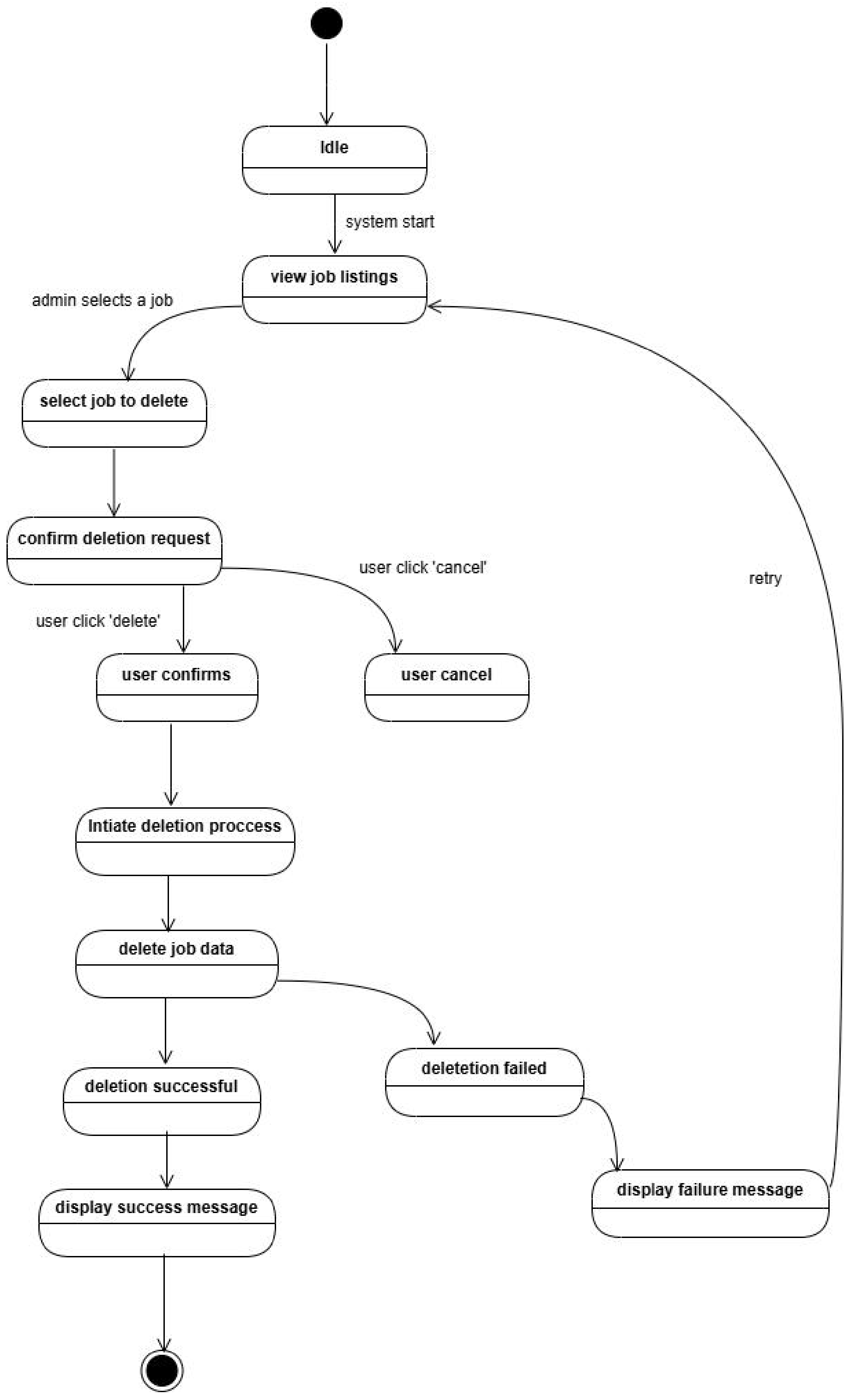

Figure 17: State chart diagram for delelte job

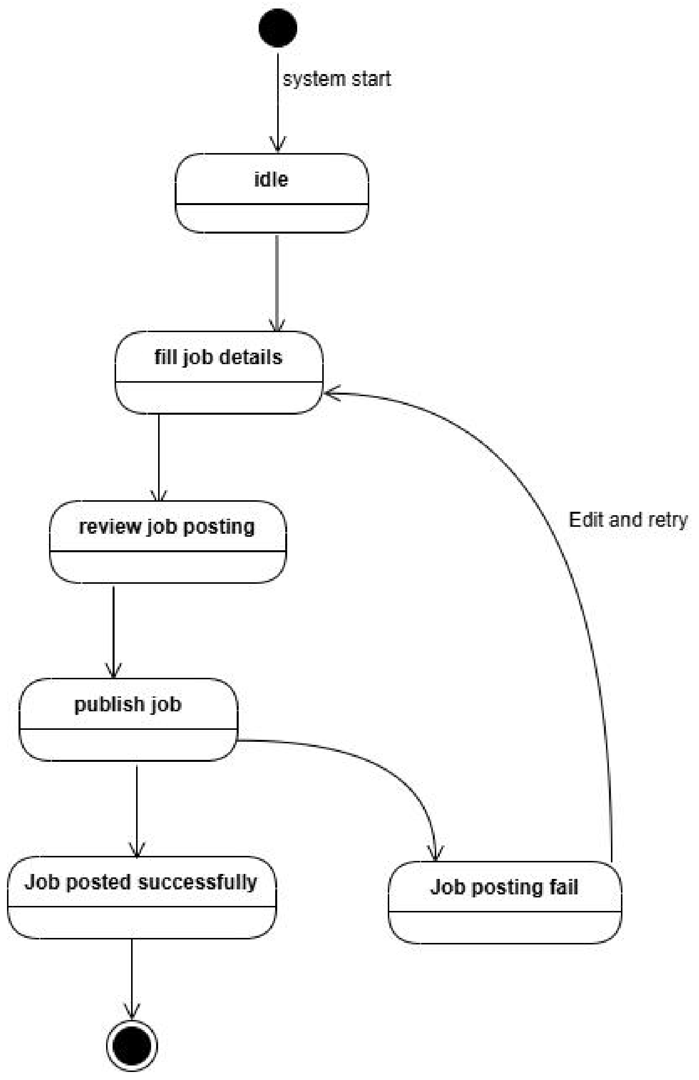

Figure 18: State chart diagram for post job

3.5.4. User Interface Prototype

A user interface (UI) prototype is a preliminary version of the user interface that focuses on visualizing the layout, interaction flow, and overall user experience of the software application. It is a blueprint or a non-functional model that allows stakeholders and the development team to get a tangible feel for how the final product will look and behave.

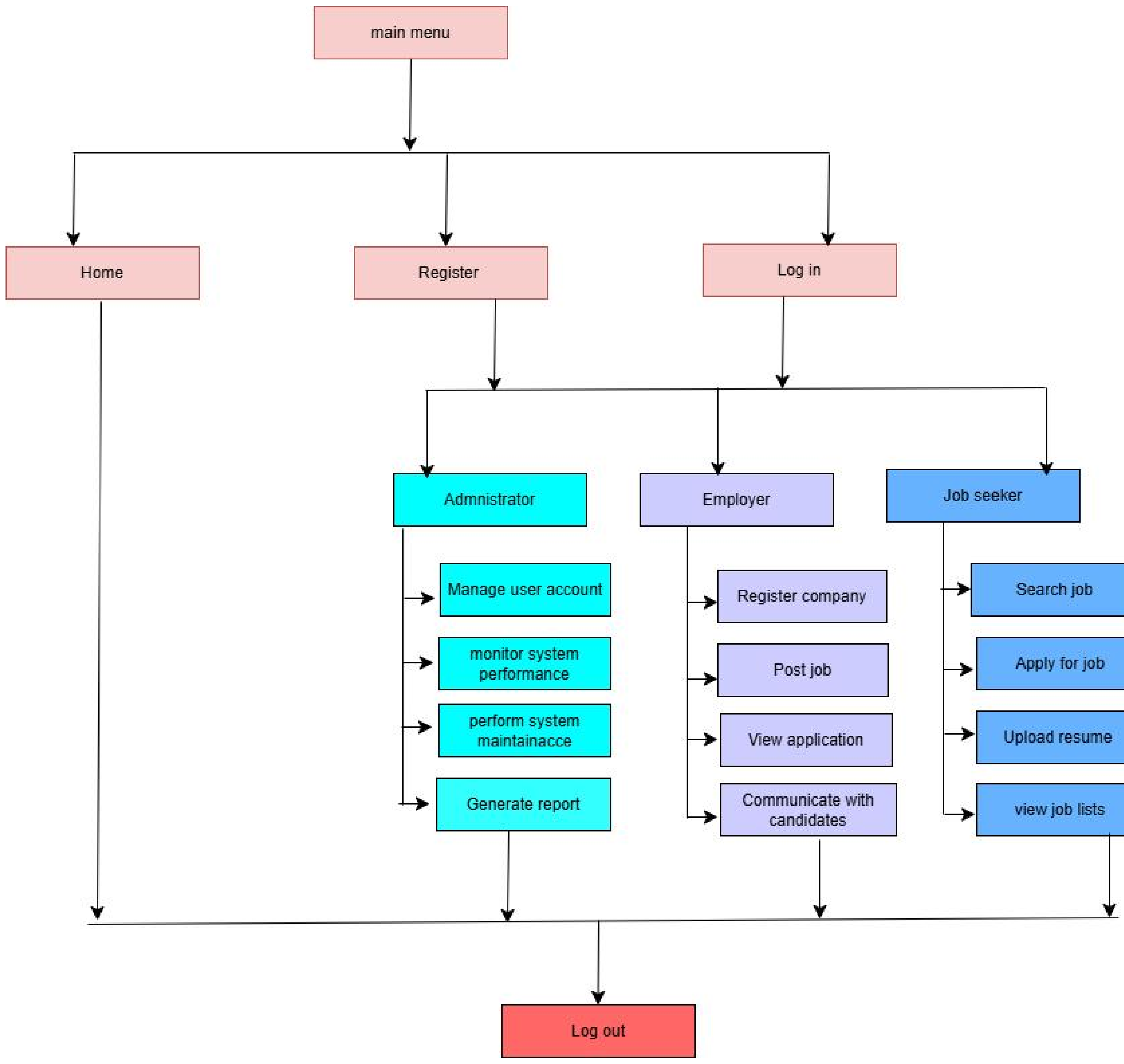

Figure 19: User interface prototype

Chapter Four: System Design

4.1. Introduction

In today's dynamic job market, connecting job seekers with the right opportunities efficiently and effectively is essential. An online job portal, accessible through both web and mobile platforms, serves as a bridge, facilitating the recruitment process for both individuals and organizations. Designing such a system requires careful consideration of various factors, from handling a large volume of user data and job postings to ensuring a seamless and intuitive user experience across different devices. This involves architecting a robust, scalable, and maintainable system capable of supporting core functionalities like user registration and profiles, job posting and searching and application management. This introduction sets the stage for exploring the key architectural components, design considerations, and technological choices involved in building a comprehensive online job tracking ecosystem.

4.1.1. Overview Of System Design Goal

Our system aims to efficiently connect job seekers with employers. Key design goals include:

- ✓ Provide comprehensive features for job seekers and recruiters.
- ✓ Offer intuitive and user-friendly interfaces on both web and mobile platforms.
- ✓ Ensure fast loading times and responsiveness for all key actions across both platforms.
- ✓ Maintain high availability and minimize downtime to ensure continuous access for users.
- ✓ Design a modular and well-documented system that is easy to understand, update, and extend.
- ✓ Optimize resource utilization and infrastructure costs without compromising performance or reliability.
- ✓ Allow for the easy integration of new features and services in the future.

4.1.2. Design Goal

Our system aim to achieve the following design goal:

End-User Criteria: Our system prioritizes effective experience for both job seekers and employers. This is achieved through intuitive and simple navigation for all technical skill levels, ensuring the platform is accessible to every user. The system focuses on delivering relevant job postings to seekers and suitable candidates to employers, often incorporating personalization features like saved searches and recommendations. Clear guidance and support resources are also provided to assist users with how to use the system.[3]

Maintenance Criteria: This system is designed for long-term viability and ease of maintenance. A modular architecture allows for focused updates and feature additions without disrupting the entire system. Comprehensive documentation of the code ensures that developers can easily understand and modify the platform. The design incorporates scalability to accommodate increasing user loads and data without significant architectural changes.

Performance Criteria: A responsive and efficient system is essential for a positive user experience. This requires pages and search results to load quickly, preventing user frustration. The search functionality is optimized to deliver accurate and relevant results rapidly. Efficient database design and optimized queries ensure quick data retrieval and storage. Finally, the system efficiently manages server resources to maintain optimal performance levels.

Cost Criteria: The development and operation of this system are approached with cost-effectiveness in mind. By strategically applying our established development methodologies and selecting appropriate technologies aligned with our team's skills, we aim to control the initial expenses of the system. Scalable and cost-effective hosting solutions are selected to manage operational costs. A well-designed and maintainable system reduces ongoing expenses related to errors and updates. The architecture allows for cost-effective scaling as the platform grows, and the selection of technologies considers both functionality and cost.

Dependability Criteria: Trust and reliability are essential for the job portal. Our system is engineered for consistent availability with minimal downtime, often through redundancy and failover mechanisms. Security measures are implemented to protect user data from unauthorized access and cyber threats[6]. The accuracy and consistency of stored data are maintained to ensure integrity. The system is designed to handle errors gracefully, minimizing disruptions and preventing data loss. Regular data backups and a comprehensive recovery plan are in place to ensure business continuity in case of significant failures.

4.2. Detailed Class Model

A class diagram is used to show a sub set of the class, interfaces, packages of classes and relationships in the system. A typical system will have many different class diagrams. A class diagram gives an over view of a system by showing its classes and the relationships among them class diagrams are static they display what interacts but not what happen when they do interact.

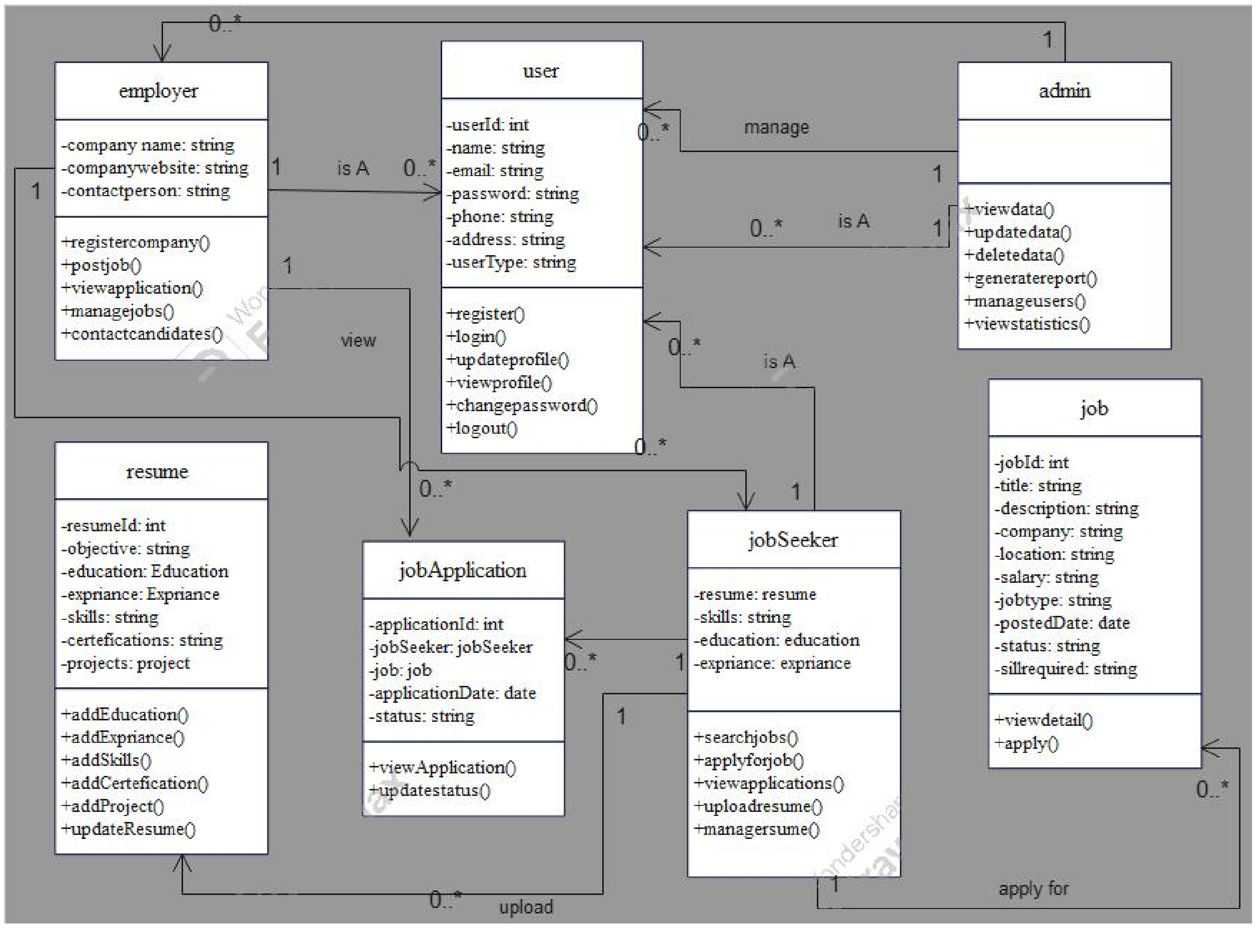

Figure 20: Class diagram

4.3. Current Software Architecture

While the specific software architecture can vary depending on the scale, features, and technology choices of a particular online job portal, a common and robust architecture for both the website and mobile application often follows a multi-tier structure, typically a three-tier architecture, with considerations for micro services and API-driven communication.

I. Website Architecture:

The website is employs a standard three-tier architecture:

1. Presentation Tier (Frontend): This is what the users directly interact with, the user interface (UI).

Technologies: HTML, CSS, JavaScript frameworks (React, Angular, Vue.js), and potentially Progressive Web App (PWA) techniques for enhanced mobile experience directly through the browser.

Responsibilities:

- ✓ Displaying information to users.
- ✓ Handling user input and interactions.
- ✓ Making asynchronous calls (AJAX/Fetch) to the application tier to retrieve and send data.
- ✓ Managing user sessions and authentication tokens.
- ✓ Providing a responsive design that adapts to different screen sizes.

2. Application Tier (Backend / Business Logic): This layer handles the core logic and functionality of the job portal.

Technologies: Programming languages (Java, Python, Node.js, Ruby on Rails, PHP, C#), backend frameworks (Spring, Django, Express.js, Laravel, .NET), and potentially a microservices architecture where different functionalities (user management, job postings, search, application processing) are handled by independent services.

Responsibilities:

- ✓ Handling user requests from the presentation tier.
- ✓ Implementing business rules and workflows (e.g., job posting validation, application processing logic, user role management).

- ✓ Interacting with the data tier to retrieve and store information.
- ✓ Performing data validation and manipulation.
- ✓ Implementing security measures, including authentication and authorization.
- ✓ Managing background tasks and job queues (e.g., sending email notifications).
- ✓ Providing APIs (RESTful or GraphQL) for the frontend (both web and mobile) to communicate with.

3. Data Tier: This layer is responsible for the persistent storage and retrieval of data.

Technologies: Relational databases (MySQL, PostgreSQL, SQL Server), NoSQL databases (MongoDB, Cassandra) depending on the data requirements (structured vs. unstructured), caching systems (Redis, Memcached) for performance enhancement, and potentially cloud storage (AWS S3, Google Cloud Storage) for file uploads (resumes, etc.).

Responsibilities:

- ✓ Storing all application data (user information, job postings, applications, company profiles, etc.).
- ✓ Providing mechanisms for data querying, indexing, and retrieval.
- ✓ Ensuring data integrity and consistency.
- ✓ Managing database connections and transactions.
- ✓ Implementing data backups and recovery strategies.

II. Mobile Application Architecture:

The mobile application (Android and/or iOS) will also typically follow a layered architecture, adapted for the mobile environment:

1. Presentation Layer (Mobile Frontend): This is the user interface of the mobile app.

Technologies: Native development (Kotlin/Java for Android, Swift/Objective-C for iOS), cross-platform frameworks (React Native, Flutter, Ionic) depending on the development strategy.

Responsibilities:

- ✓ Displaying information in a mobile-friendly format.
- ✓ Handling user interactions (taps, swipes, etc.).
- ✓ Making API calls to the backend application tier to fetch and send data.
- ✓ Managing the user interface state.
- ✓ Handling user authentication and storing session information securely.
- ✓ Providing offline capabilities where appropriate.

2. Domain or Business Logic Layer (Backend): In many modern mobile architectures, the core business logic is centralized in the backend application tier. The mobile app primarily acts as a client, consuming APIs.

Responsibilities (in the context of the mobile app):

- ✓ Orchestrating API calls to the backend.
- ✓ Handling data received from the backend and transforming it for UI display.
- ✓ Implementing UI-specific logic and state management.
- ✓ Managing local data storage (if any) for offline use or caching.

3. Data Access Layer (Abstraction for Backend Communication): This layer handles how the mobile app interacts with the backend APIs.

Technologies: Libraries for making network requests (Retrofit for Android, Alamofire/URLSession for iOS, or the networking capabilities of cross-platform frameworks).

Responsibilities:

- ✓ Abstracting the details of API communication.
- ✓ Handling request formatting and response parsing (JSON, XML, etc.).
- ✓ Implementing error handling for network requests.
- ✓ Potentially caching API responses for better performance.

4.4. Sub-System Decomposition With Service

Sub-system decomposition for our system involves breaking down the system into logical functional areas like User Management, Job Posting Management, Job Seeker Management, and Employer Management, each of which is further composed of individual services. These services, such as User Service, Job Posting Service, and Application Service, encapsulate specific functionalities and communicate with each other through APIs to deliver the overall features of the platform. Additionally, cross-cutting sub-systems like Search & Indexing and Core Platform Services (including API Gateway) provide essential functionalities that support the entire system. This modular approach, utilizing micro-services, enhances maintainability, scalability, and independent development of different parts of the system.[3]

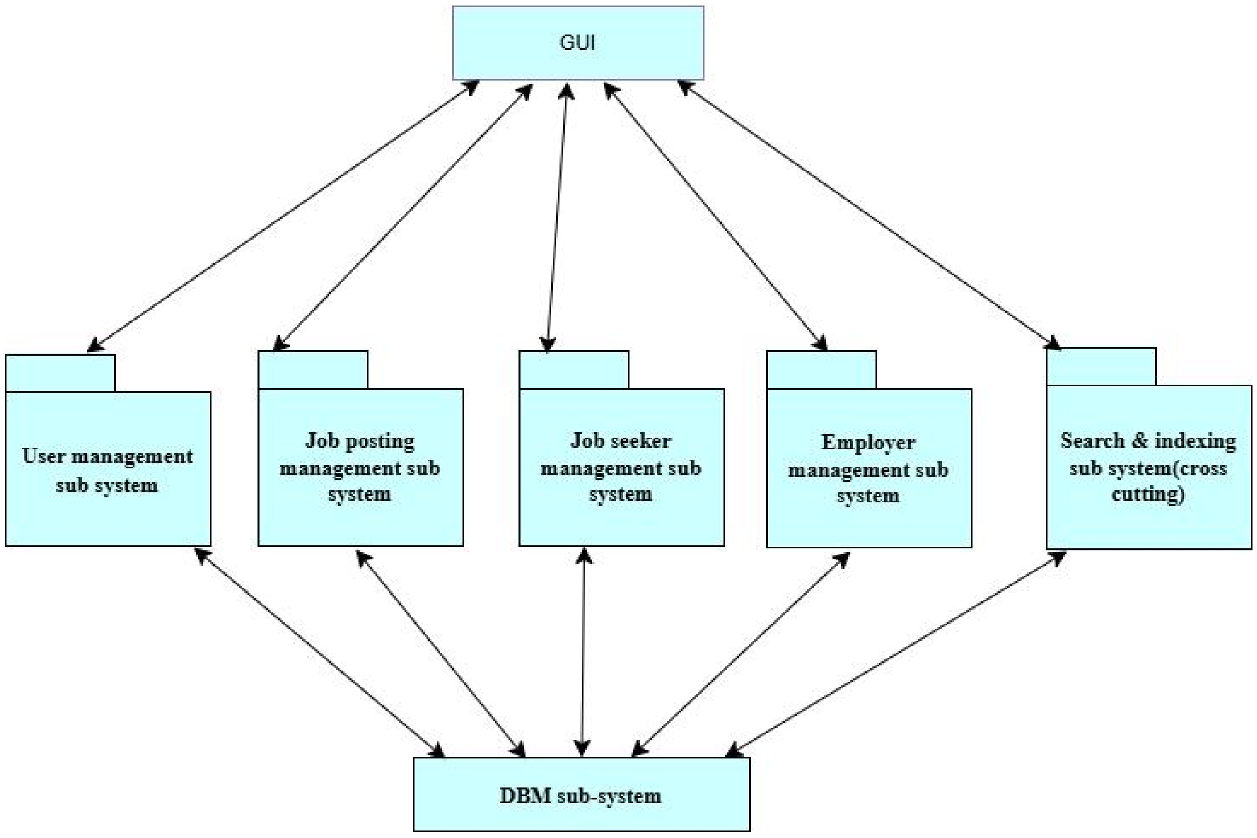
GUI
User management sub system
Job posting management sub system
Job seeker management sub system
Employer management sub system
Search & indexing sub system(cross cutting)
DBM sub-system

Figure 21: Sub-system decompositio

4.5. Proposed System Architecture

Our system architecture is designed as a multi-tier application, with a Presentation Layer for user interaction (both web and mobile), an Application Layer to handle the business logic, and a Data Layer for managing data. The web interface will be developed using HTML, CSS, and JavaScript, while the mobile application will be built using Java for Android. PHP will be used for the website's backend, and SQL server will serve as the database. This architecture ensures a scalable, robust, and user-friendly platform for connecting job seekers and employers. The following is the detailed architecture for the system we have proposed.

Overall System Architecture: Employs a multi-tier architecture with a Presentation Layer (user interfaces for web and mobile), an Application Layer (handles business logic and processes), and a Data Layer (manages data storage and retrieval).

Components:

Presentation Layer: Features a Web Interface (HTML, CSS, JavaScript; user-friendly design for job seekers, employers, and admins; includes job searching, profile or posting creation, user management) and a Mobile Application (Android; user-friendly interface for job seekers and employers; includes job searching, application submission, job posting).

Application Layer: Utilizes a Web Application Server (PHP backend; manages authentication, job searching, job posting, application management, and business logic).

Data Layer: Includes a Database Server (SQL server for storing user, job posting, application, and system data).

Key Technologies: Frontend (HTML, CSS, JavaScript), Backend (PHP for website), Mobile App (Java for Android), Database (SQL server), IDE (Visual Studio), Documentation Tools (MS-Word 2010).

System Interactions:

Job Seeker: Searches for jobs via web or mobile, views details and applies, creates profile, manages profiles.

Employer: Creates/manages company profiles via web, posts job openings, reviews applications and manages candidates.

Administrator: Manages users, job postings, and system data via web; monitors activity and generates reports.

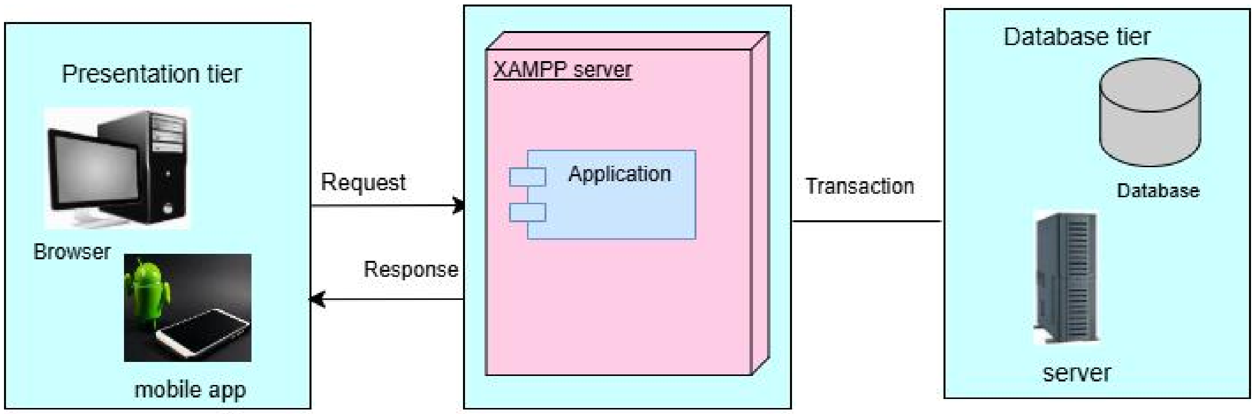

Figure 22: System architecture

4.6. Component Diagram

The component diagram below illustrates the major building blocks of the online job portal system and their interrelationships. It decomposes the system into key components such as User Interface, User Management, Job Management, Application Management, Resume Management, Search and Filter, Database, Reporting, and Communication. The diagram highlights how these components interact to deliver the overall functionality of the platform, showing data flow and dependencies. For example, it shows how the User Interface relies on other components to handle user-related actions, job postings, applications, and searches, while the Database component provides persistent storage for all the system's data.

Components:

User Interface (UI) Component: This component is responsible for the presentation layer, handling user interactions.It will have sub-components for:

- ✓ **Web Interface:** Handles the website part of the portal.
- ✓ **Mobile Interface:** Handles the Android mobile application.

User Management Component: Manages user registration, login, profiles, and authentication.

Job Management Component: Handles the creation, storage, retrieval, and display of job postings.

Application Management Component: Manages job applications submitted by job seekers, including submission, tracking, and notifications.

Resume Management Component: Handles the uploading, storage, and retrieval of resumes.

Search and Filter Component: Provides the functionality to search for jobs based on various criteria.

Database Component: Provides data storage and retrieval for all components.

Reporting Component: Generates reports for administrators on user activity, job applications, etc.

Communication Component: Handles communication between the system and users.

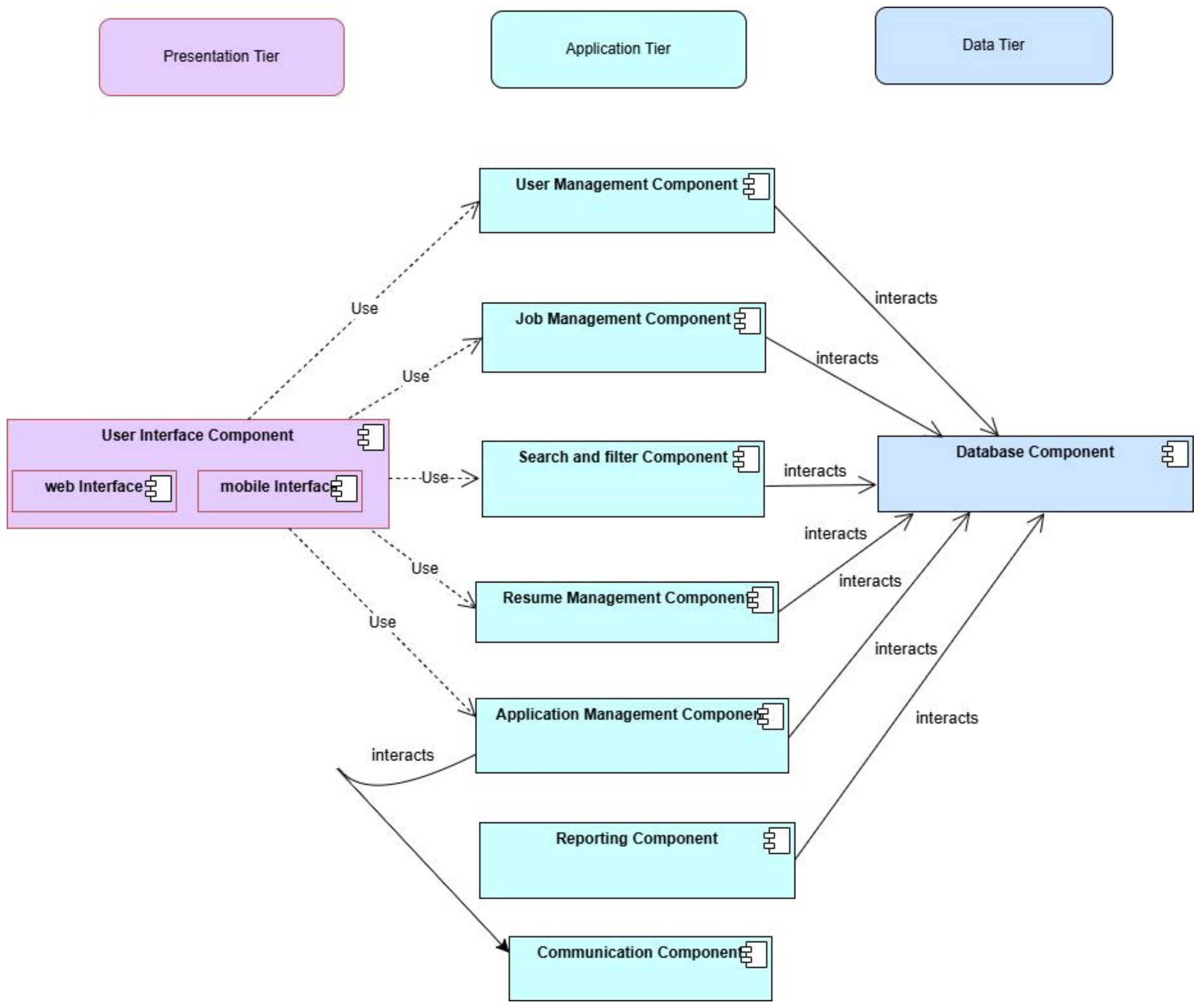

Figure 23: Component diagram

4.7. Deployment Diagram (Hardware/Software Mapping)

Deployment diagram of our system illustrates the physical architecture of an online job tracking system. It shows how the software components of the system are deployed onto the underlying hardware infrastructure. The diagram includes key elements such as client devices (desktop computers and mobile phones), servers (web, application, database, and mobile app), and network infrastructure (Internet). It also depicts the artifacts, which are the software packages and files that are deployed onto the nodes. The connections between the nodes represent the communication paths and protocols used for interaction.

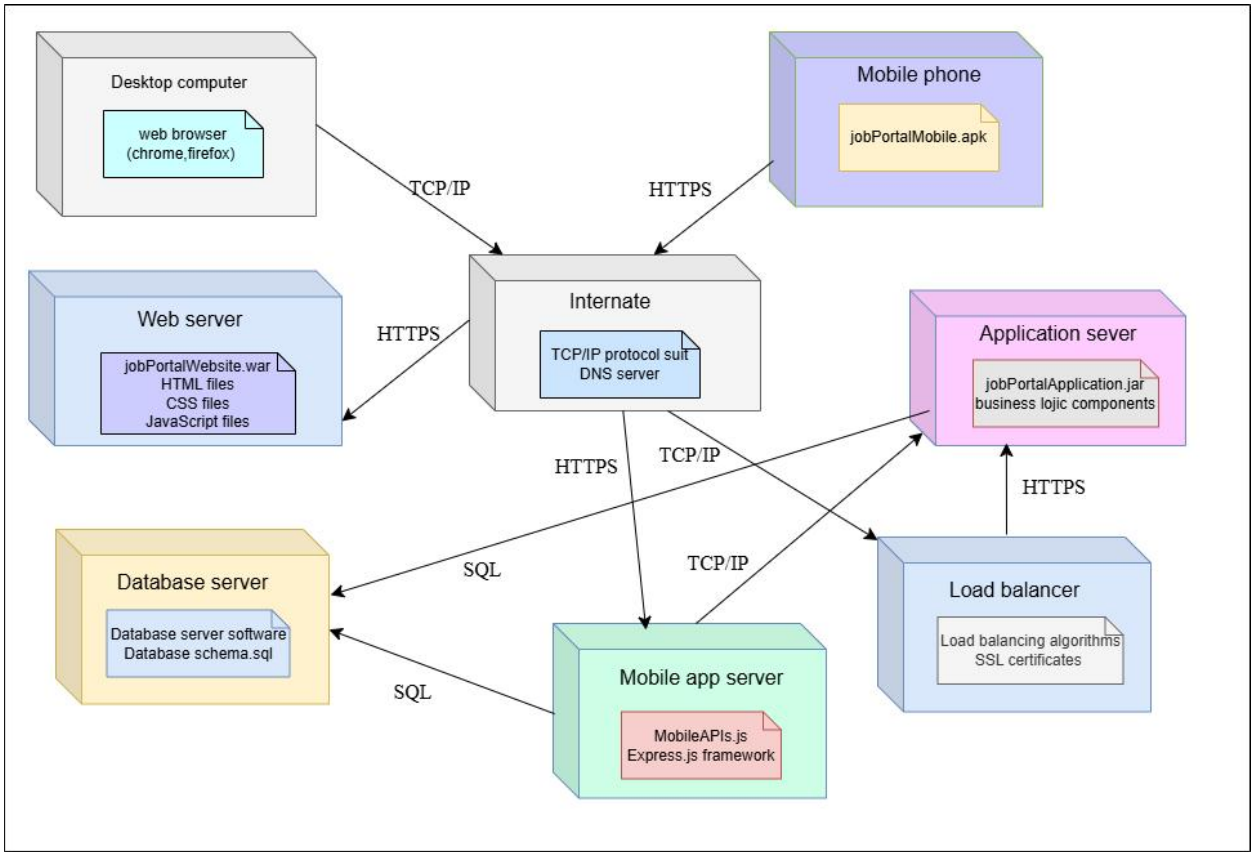

Figure 24: Deployment diagram

4.8. Persistence Model

In a database-driven application, almost all system interactions deal with persistent data. This means that instead of data existing only temporarily in the application's memory, it is stored in a way that it remains available even after the application is closed or the server is restarted. In our systems, information related to job seekers (such as profiles, resumes, and application history), employers (including company details and job postings), job postings (with descriptions, requirements, and locations), applications (tracking which job seeker applied for which job), and user accounts (containing login credentials and user roles) are persistent data, which should be stored in the Database Management System (DBMS). The DBMS provides the necessary structure, organization, and durability to manage this data effectively, ensuring that the job portal can function correctly and provide a consistent experience for its users.[5][7]

Mapping:

Mapping is the process of translating object-oriented program elements into relational database structures; this involves representing objects as tables and their attributes as fields within those tables, which is essential for the persistent storage of data such as job seeker information, employer details, and job postings, ensuring that this information is stored and can be retrieved effectively by the Database Management System.[5] The mapping of objects to tables is displayed as follows:

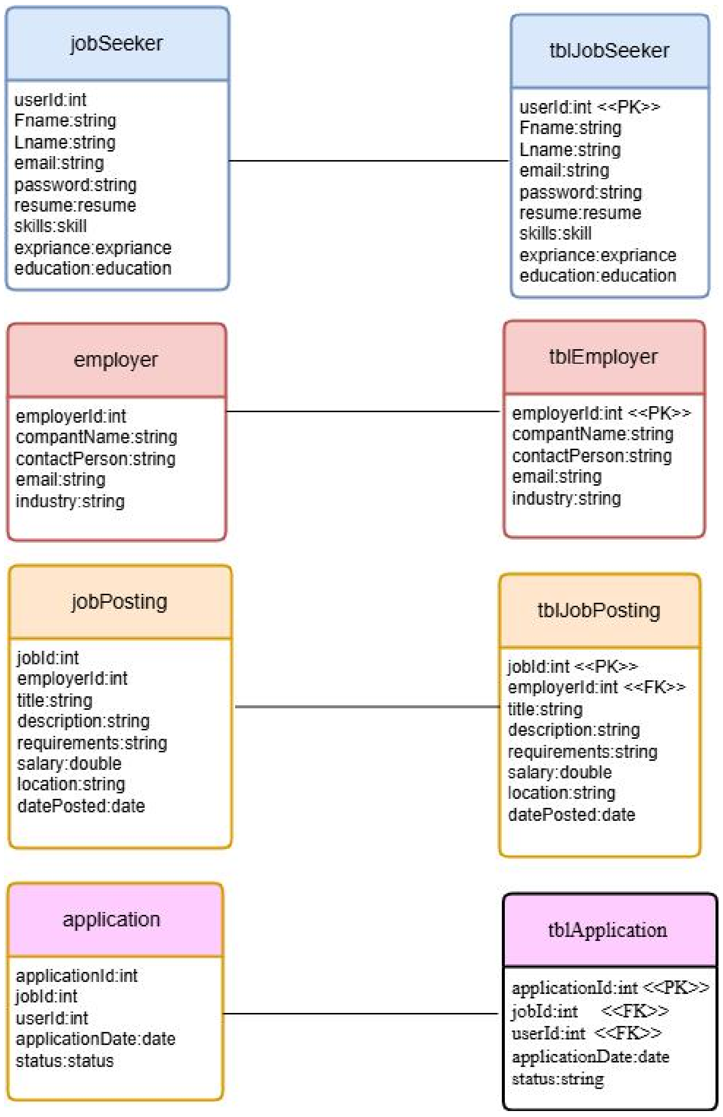

Figure 25: Mapping objects to tables

Relationships among tables

- An Employer can have many Job Postings (One-to-Many)
- A Job Seeker can apply for many Job Postings (Many-to-Many - requires an intermediate table like Application)
- Each Application is for one Job Posting and belongs to one Job Seeker (Many-to-One)

4.9. Access control And Security

This section details the measures implemented to control access to the job portal and ensure the security of data. It involves defining user roles, authentication mechanisms, authorization protocols, and security practices to protect against unauthorized access, data breaches, and other security threats.[6]

1. User Roles and Permissions: Define the various roles within the system (e.g., Job Seeker, Employer, Administrator) and specify the permissions associated with each role.

Implementation:

- ✓ Job Seekers able to create profiles, search for jobs, apply for positions, and manage their applications.
- ✓ Employers able to post jobs, view applications, manage candidates, and communicate with applicants.
- ✓ Administrators have full access to manage users, roles, system settings, and monitor system activity.

2. Authentication Mechanisms: The methods used to verify the identity of users.

Implementation:

- ✓ Secure password storage with hashing and salting.
- ✓ Session management to control user access duration.
- ✓ Consider social media login integration with appropriate security measures.

3. Authorization Protocsols: Detail how the system determines what actions a user is allowed to perform.

Implementation:

- ✓ Role-Based Access Control (RBAC) to manage user permissions.
- ✓ Access control lists (ACLs) for specific resources or data.
- ✓ Workflow authorization for sensitive operations (e.g., approving job postings).

4. Data Security: The measures taken to protect data at rest and in transit.

Implementation:

- ✓ Secure storage of data with appropriate database security measures.
- ✓ Regular backups and disaster recovery plans.

5. Security Practices: The ongoing practices to maintain system security.

Implementation:

- ✓ Regular security audits and vulnerability assessments.
- ✓ Monitoring and logging of system activity.

4.10. Global Control Flow

4.10.1. Procedural Driven Control Flow

Procedural Driven Control Flow as an approach that emphasizes a pre-defined, step-by-step user interaction with the system[7]. Users navigate through a series of prompts and screens, completing each step before proceeding to the next[2]. The system validates and processes user input at each stage, ensuring accuracy and preventing errors.

Procedural Driven Control Flow applied to our system as follows:

1. User Registration:

- ✓ The user accesses the registration page or screen.
- ✓ The system presents a series of fields for the user to enter their information (e.g., name, email, password, contact details).
- ✓ The user fills in each field, one by one.
- ✓ The system validates each field upon completion (e.g., checking for a valid email format, password strength).
- ✓ The user proceeds to the next step, by clicking a "Next" button.
- ✓ Once all required fields are filled, the user submits the registration form, by clicking a "Submit" button.
- ✓ The system validates all the entered information.
- ✓ If all is correct, the system creates a user account and notifies the user of successful registration.

2. Job Application:

- ✓ The user selects a job to apply for.
- ✓ The system displays an application form with a series of sections (personal information, education, work experience, cover letter).
- ✓ The user fills in each section sequentially.
- ✓ The user uploads required documents (e.g., resume, cover letter) following specific format guidelines.
- ✓ The system validates the uploaded files and the information provided in each section.
- ✓ The user reviews the completed application.
- ✓ The user submits the application.
- ✓ The system confirms the submission and provides a status update.

3. Employer Job Posting:

- ✓ The employer logs in to their account.
- ✓ The system presents an interface to create a new job posting.
- ✓ The employer fills in fields such as job title, description, requirements, salary, and application deadline in a step-by-step manner.
- ✓ The employer previews the job posting.

- ✓ The employer confirms and submits the job posting.
- ✓ The system publishes the job posting and provides confirmation.

4.10.2. Event Driven Control Flow

Event Driven Control Flow is an approach that focuses on dynamic responses to user actions. Instead of following a pre-defined path, the system reacts to user choices and triggers relevant events or functionalities.[1]

1. Job Search: When a user enters keywords in the search bar, each character typed on the "search" button triggers the system to react by suggesting search terms in real-time, dynamically filtering job listings, and updating search results as filters are applied; upon clicking a specific job listing, the system responds by displaying the job details page.

2. User Profile Management: Navigating to profile settings and clicking "Edit Profile" prompts the system to display editable fields; uploading a new profile picture triggers a preview; changing contact information fields leads to real-time validation, with the updated information being saved upon clicking "Save Changes".

4.11. Boundary Conditions And Exception Handling

This section deals with how the system is designed to handle boundary conditions and exceptions. Boundary conditions refer to the limits or constraints under which the system operates, such as maximum user limits and data entries. Exception handling involves the strategies and mechanisms implemented to manage errors and unexpected events that may occur during system operation. This is essential for ensuring system stability, data integrity, and a positive user experience.

4.11.1. Boundary Conditions

Maximum Number of Concurrent Users: The system is designed to handle up to 5000 concurrent users.

Maximum Data Entries:

Job postings: Employers can post a maximum of 10 active job listings at any given time.

Resumes: Job seekers can upload a maximum of 5 resumes to their profile.

Applications: Job seekers can apply for a maximum of 10 jobs per day.

File Upload Size: The maximum file size for resume uploads is 10MB. Users will be prompted to upload a smaller file if they exceed this limit.

Session Timeout: User sessions will automatically time out after 30 minutes of inactivity. Users will be notified 2 minutes prior to timeout and given the option to extend their session.

System Architecture: The system follows a Client-Server architecture and allows remote access.

Client-Side Requirements:

- Internet connection.
- Web browser.
- Legitimate user account.
- Correct URL.
- Ability to navigate hyperlinks or pages.
- Ability to use services like viewing available jobs, applying, etc.

Server-Side Requirements:

- System administrator/Web Master privileges to initiate and update the system.
- Server registration with a service provider.
- Internet connection and database-driven website for remote access.
- Automatic saving of changes and backups.

4.11.2. Exception Handling

➢ When a user attempts to register with an email address already present in the database, the system catch the resulting duplicate entry exception, and display the error message "This email address is already registered. Please log in or use a different email." to the user, and concurrently log the error for debugging.

- If a user attempts to submit an application without an attached resume, the system detect the missing file, then display an error message stating, "Please upload your resume," and consequently prevent the application from being submitted.
- When a user's search query yields no results, the system will display a message, "No jobs found matching your search criteria," and importantly, suggest alternative search terms or filters to help them refine their search and find relevant information.
- When an employer attempts to post a job with invalid input, such as missing fields or an incorrect date format, the system will validate the input to identify all errors, then display informative error messages that indicate which specific fields require correction, and ultimately prevent the job posting from being saved until all identified errors are fully resolved.
- If a user attempts to access a job posting with an invalid ID, such as one that has been deleted, the system will catch the corresponding error (e.g., a "Job Not Found" exception), display a user-friendly message like "Sorry, that job posting could not be found," and log the error for administrative review.

4.12. User Interface Design

The UI design provides simple login and account creation screens for a job portal. The "Log in" screen has email and password fields with a "Forgot password?" option. The "Create account" screen requires full name, email, and password. Both designs use clear buttons and links for easy navigation.

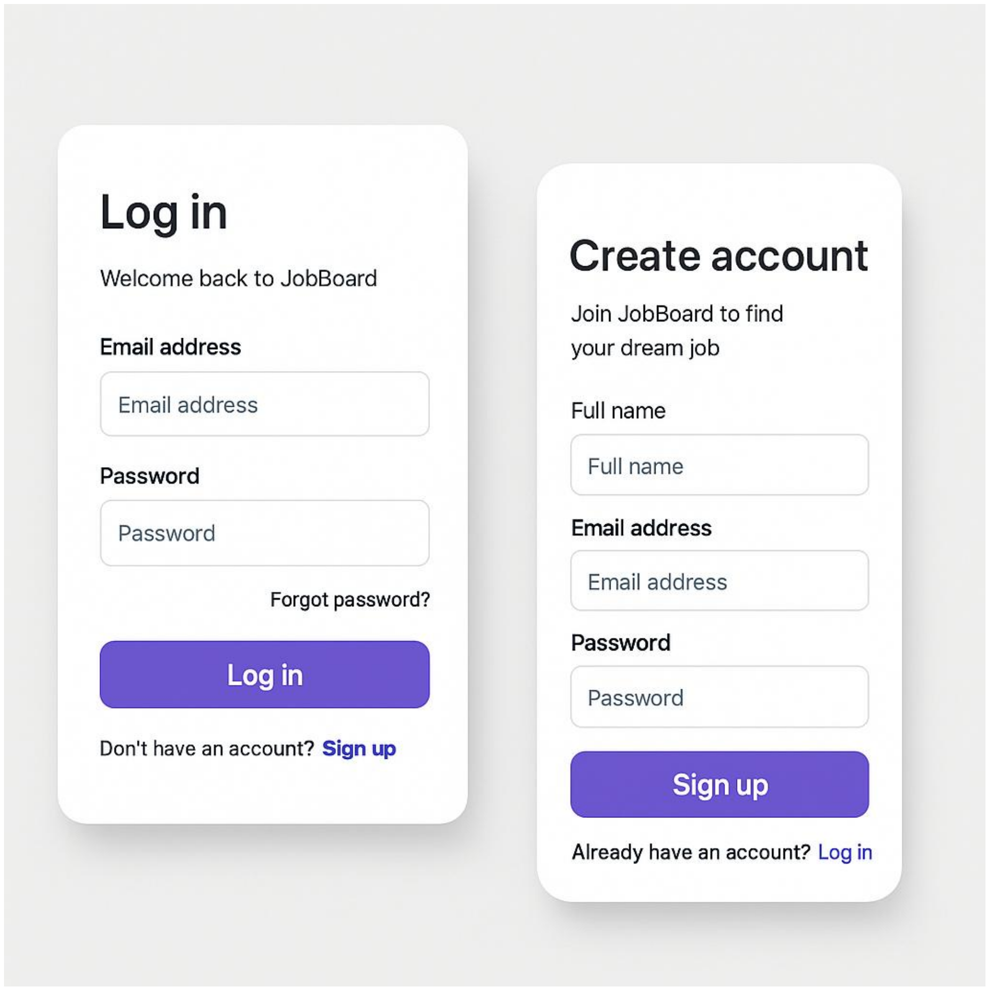

Figure 26: User interface design

Reference

[1] Michael. Mendez, *An introduction to Web development and programming*. Open SUNY textbooks, Milne Library, 2014.

[2] G. Gay and I. Karasyov, "Web Accessibility for Developers Essential Skills for Web Developers DIGITAL EDUCATION STRATEGIES, THE CHANG SCHOOL."

[3] A. Professor, M. K. Kumari, and M. B. Chandran, "A descriptive study on Applicant Tracking System: Automation software for Recruitment and Selection," *IJRAR19VP033 International Journal of Research and Analytical Reviews*, 2019, [Online]. Available: www.ijrar.org

[4] "A Complete Web Development Guide."

[5] "3. Database Management Systems Author Abhishek Taneja".

[6] "3 ESSENTIAL TYPES OF CYBERSECURIT Y SOLUTIONS YOUR BUSINESS MUST HAVE."

[7] "Frontend Developer Manual COREMEDIA CONTENT CLOUD Frontend Developer Manual," 2021.