JOB PORTAL SYSTEM

A PROJECT REPORT

Submitted by

SACHIN KUMAR A	(2303811710421132)
SHARUKESH DM	(2303811710421140)
SRINIVAS R	(2303811710421157)

in partial fulfillment of requirements for the award degree of Bachelor in Engineering

CSB1303 – OBJECT ORIENTED ANALYSIS AND DESIGN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

DECEMBER - 2025

JOB PORTAL SYSTEM

A PROJECT REPORT

Submitted by

SACHIN KUMAR A	**(2303811710421132)**
SHARUKESH DM	**(2303811710421140)**
SRINIVAS R	**(2303811710421157)**

in partial fulfillment of requirements for the award degree of Bachelor in Engineering

CSB1303 – OBJECT ORIENTED ANALYSIS AND DESIGN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

DECEMBER - 2025

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"JOB PORTAL SYSTEM"** is the bonafide work of **SACHIN KUMAR A, SHAHRUKESH DM, SRINIVAS R** who carried out the project work and we declare that the project is their own work, and has not been copied from other sources.

Date of Viva Voce:…………….

SIGNATURE	**SIGNATURE**
Mrs. V. KALPANA , M.E.,(Ph.D.,)	**Mr. R. RAJAVARMAN, M.E., (Ph.D.,)**
SUPERVISOR	**HEAD OF THE DEPARTMENT**
Assistant Professor	Assistant Professor (Sr. Grade)
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The Job Portal System is an innovative, feature-rich web application designed to modernize and streamline the process of job searching, recruitment, and career development. Traditional job platforms often face challenges such as limited personalization, inefficient matching algorithms, and cumbersome application processes. This platform aims to resolve these issues by providing an intelligent, user-centric environment that supports seamless job discovery, personalized recommendations, and efficient application management. Built on modern web technologies, the platform incorporates a robust client-side architecture to manage user profiles, job listings, applications, and matching algorithms. Core functionalities include multi-tier user authentication, intelligent job matching based on skills and preferences, real-time notifications, comprehensive application tracking, and dynamic user engagement. The system leverages localStorage API with structured JSON schemas to ensure efficient data management and data integrity. To enhance security, the platform incorporates authentication and role-based access control, ensuring that only authorized users can perform sensitive actions such as posting jobs or accessing personal data.

Keywords:

Intelligent Job Matching ,Personalized Recommendations, Role-based Access Control , Application Tracking

ACKNOWLEDGEMENT

We thank our **Dr. N. Vasudevan,** Principal, for his valuable suggestions and support during the course of my research work.

We thank our **Mr. R. Rajavarman**, Head of the Department, Assistant Professor (Sr. Grade), Department of Computer Science and Engineering, for his valuable suggestions and support during the course of my research work.

We wish to record my deep sense of gratitude and profound thanks to my Guide **Mrs. V. Kalpana**, Assistant Professor , Department of Computer Science and Engineering, for her keen interest, inspiring guidance, constant encouragement with my work during all stages, to bring this thesis into fruition.

We are extremely indebted to our project coordinator **Mrs. V. Kalpana**, Assistant Professor, Department of Computer Science and Engineering, for her valuable suggestions and support during the course of my research work.

We also thank the faculty and non-teaching staff members of the Department of Computer Science And Engineering, K.Ramakrishnan College Of Technology, Samayapuram, for their valuable support throughout the course of my research work.

Finally, we thank our parents, friends and our well wishes for their kind support.

SIGNATURE

____

____

____

____

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

API	-	Application Programming Interface
DB	-	Database
GOF	-	Gang of Four (Design Patterns)
GRASP	-	General Responsibility Assignment Software Patterns
GUI	-	Graphical User Interface
HTML	-	Hypertext Markup Language
JSON	-	JavaScript Object Notation
PHP	-	Hypertext Preprocessor
SQL	-	Structured Query Language

CHAPTER 1
INTRODUCTION

1.1 Introduction about Domain

The Job Portal System is a modern web-based platform designed to revolutionize job searching and recruitment processes. It connects job seekers and employers through an intelligent, user-friendly interface that simplifies employment connectivity. The system features advanced matching algorithms that personalize job recommendations based on skills, preferences, and profiles. Job seekers benefit from streamlined applications, real-time tracking, and personalized dashboards. Employers gain efficient tools for posting jobs, managing candidates, and accessing qualified talent. With secure role-based authentication, the platform ensures data protection for all users. Its responsive design provides seamless accessibility across all devices. Real-time notifications keep users engaged and informed throughout their journey. Built with HTML5, CSS3, and JavaScript.

1.2 Problem Description

The current job market faces significant inefficiencies in connecting job seekers with suitable employment opportunities through traditional and digital platforms. Job seekers struggle with overwhelming numbers of irrelevant listings, generic search results, and cumbersome application processes that lack personalization. Employers face challenges in finding qualified candidates amidst unqualified applications and inefficient screening methods. Existing platforms often provide poor user experiences with complicated interfaces and limited mobile compatibility. The absence of intelligent matching algorithms leads to missed connections between qualified candidates and appropriate positions. Both job seekers and employers lack real-time communication channels and application tracking capabilities. Security concerns regarding personal data and fraudulent listings further complicate the digital recruitment landscape. These utilities promote code reuse and reduce duplication.

1.3 Objective of the Project

The primary objective of this Job Portal System is to develop a comprehensive web-based platform that effectively bridges the gap between job seekers and employers through an intuitive, efficient, and personalized digital ecosystem. The system aims to revolutionize traditional job search and recruitment processes by implementing intelligent matching algorithms that connect candidates with relevant opportunities based on their skills, experience, and preferences. The project focuses on establishing a robust user management system with secure authentication and role-based access control, safeguarding sensitive personal and professional information while maintaining data integrity. Another critical objective is to implement real-time notification systems and application tracking features that keep users informed and engaged throughout their job search or hiring journey. The platform strives to enhance user engagement through personalized job recommendations, interactive application processes.

1.4 Scope of the Project

The Job Portal System is a comprehensive web-based platform designed to transform the traditional job search and recruitment landscape. This innovative application connects job seekers with employers through an intelligent, user-friendly interface that streamlines the entire employment process. The system features a multi-tier authentication system supporting job seekers, employers, and administrators with customized dashboards and functionalities. At its core, the platform employs sophisticated matching algorithms that analyze user profiles, skills, and preferences to deliver personalized job recommendations. Job seekers benefit from advanced search capabilities, one-click applications, and real-time tracking of their submission status. Employers gain access to powerful tools for posting opportunities, managing applications, and discovering qualified candidates. Security is maintained through role-based access controls and secure data management practices.

CHAPTER 2
SYSTEM REQUIREMENT SPECIFICATION

2.1 Functional Requirements

The functional requirements of the Job Portal System describe the essential operations the system must perform to support job seekers, employers, and administrators. The system must allow users to register, log in, and manage their profiles securely. Job seekers should be able to search for jobs using keywords or filters, view job details, and submit applications. Employers must be able to post new job openings, update or delete existing postings, and review applications received. The system should store and display application status so users can track their submissions. Administrative functions include monitoring user activities, verifying employer accounts, and ensuring data integrity across the portal.

2.2 Non Functional Requirements

The non-functional requirements focus on how the system should perform rather than what it should do. The Job Portal System must provide a fast and responsive user interface with minimal loading time to ensure a smooth experience. The portal should maintain high usability, offering intuitive navigation and clear feedback messages. It should be compatible across multiple devices and browsers through responsive design. Security is essential, requiring proper validation of user input, protection of sensitive data, and prevention of unauthorized access. Reliability must be ensured by preventing system crashes and providing stable performance even with multiple users. The system should also be maintainable so that future updates, enhancements, or integrations can be added easily. These non-functional requirements help ensure the system's quality and performance. Security is essential, requiring proper validation of user input, protection of sensitive data, and prevention of unauthorized access. Reliability must be ensured by preventing system crashes and providing stable performance even with multiple users. The system should also be maintainable so that future updates, enhancements, or integrations can be added easily.

2.3 Hardware Requirements

The hardware requirements for running the Job Portal System are minimal since it is a web-based application. A standard computing device such as a laptop or desktop with at least 4 GB of RAM and a dual-core processor is sufficient for development and usage. Users need a device capable of running a modern web browser, which can include PCs, tablets, or smartphones. If implemented with a backend, the system may require a hosting server with adequate storage and processing capacity based on expected user load. Internet connectivity is essential for accessing the portal, uploading files such as resumes, and interacting with online resources.

2.4 Software Requirements

The software requirements include the tools, platforms, and technologies needed to develop and use the Job Portal System. For frontend development, HTML, CSS, Bootstrap, and JavaScript are required to design the interface and provide user interactivity. Any modern web browser such as Chrome, Firefox, or Edge is necessary for running and testing the application. Developers may use code editors like Visual Studio Code, Sublime Text, or Notepad++. If backend implementation is added, additional software such as PHP, Node.js, or Python, along with a database like MySQL or MongoDB, may be required. Supporting packages for version control (Git) and hosting (XAMPP, WAMP, or cloud services) can enhance development efficiency.

2.5 User Characteristics

Users of the Job Portal System include job seekers, employers, and administrators, each having different characteristics and levels of technical understanding. Job seekers may have basic computer skills and expect simple navigation while searching for jobs, creating profiles, and submitting applications. Employers typically possess moderate technical knowledge and require efficient tools for posting job openings and managing applicants. Administrators are usually more experienced users who handle system maintenance, data monitoring, and user verification. The system is designed to be intuitive.

2.6 Constraints

The Job Portal System faces certain constraints that influence its design and implementation. Since the presented system is a frontend prototype, it does not include real-time backend processing or database storage, limiting its ability to handle live data and dynamic updates. The system depends on internet connectivity for full functionality, especially when integrated with backend services. Another constraint is browser compatibility, as the interface must adapt correctly to different screen sizes and devices. Security limitations exist in the absence of server-side validation or encryption. Time constraints during development may restrict the inclusion of advanced features such as AI recommendations or real-time notifications. These constraints highlight considerations that must be addressed in future development stages.

Another major constraint involves the system's **dependency on internet connectivity**, particularly once backend services and APIs are introduced in the future. Any disruption in connectivity would impact the user experience, restricting system usability. Additionally, the prototype must account for **browser compatibility issues**, as modern web applications need to function seamlessly across multiple browsers such as Chrome, Firefox, Edge, and Safari. Differences in rendering engines, JavaScript execution, and CSS support can lead to inconsistent layouts, misaligned components, or functional discrepancies. Ensuring cross-browser responsiveness and compatibility across mobile, tablet, and desktop devices increases design complexity and testing requirements.

CHAPTER 3
ANALYSIS AND DESIGN

3.1 Use Case Model

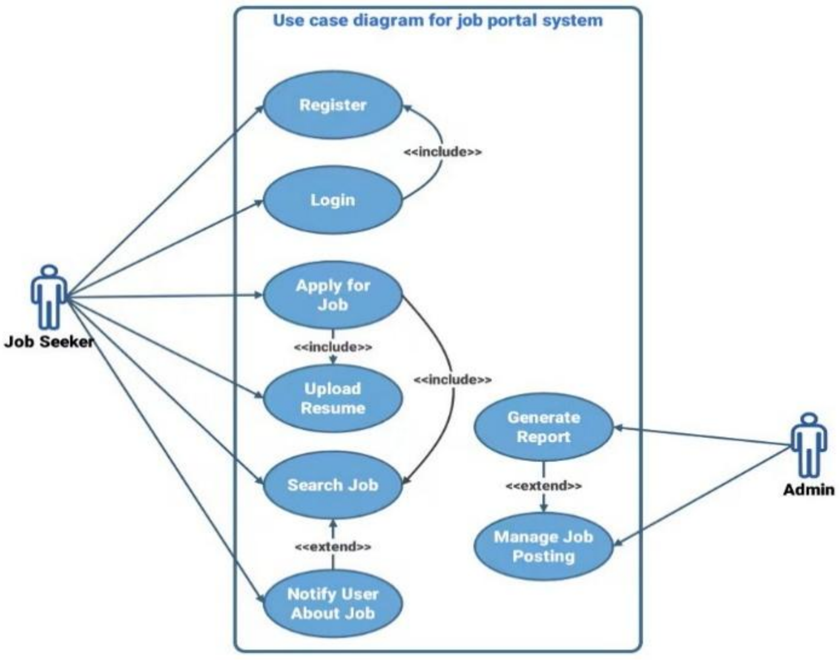


Figure.3.1 Use Case Model

3.1.1 Use Case Description

Each use case in the Job Portal System provides a detailed description of individual interactions between users and the system. For example, the "Job Search and Application" use case describes how a Job Seeker enters keywords, selects filters, browses results, views job details, and submits an application. The "Post Job Vacancy" use case explains how an Employer logs in, fills in job details, uploads company information, and publishes a job listing. The "User Registration and Authentication" use case outlines how the system validates user information, verifies credentials, and grants access. These use case descriptions present preconditions, postconditions, triggers, normal flows, and alternate flows to ensure that each confirming the application. Conditional branches such as "login success or failure," "job available or not," and "application already submitted or new" help visualize real-time decision-making. Similarly, the Employer workflow includes posting a job,

3.2 Activity Diagram

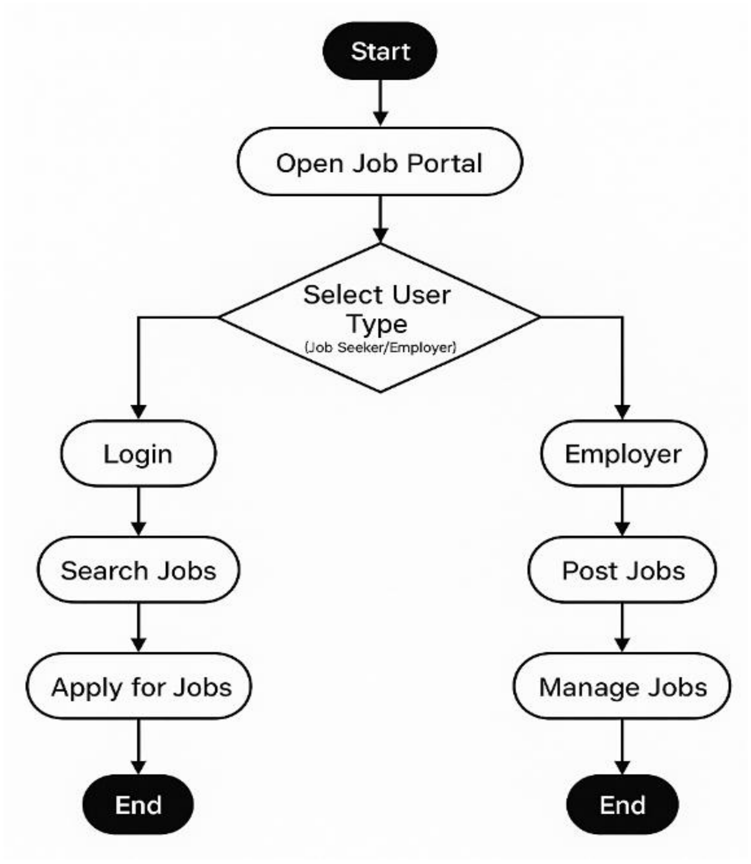


Figure.3.2 Activity Diagram

3.2.1 Activity Diagram Description

The activity diagram represents the high-level workflow of a Job Portal system. The process begins when the user opens the portal and chooses their user type: Job Seeker or Employer. For a Job Seeker, the system supports activities such as logging in, searching available jobs, and applying for selected positions. For an Employer, the system enables posting new job openings and managing existing listings. Each user flow terminates after completing their respective actions. This diagram clearly illustrates conditional navigation, independent user workflows, and the overall behaviour of the system. For the **Job Seeker**, the system enables authentication through the Login activity, followed by the Search Jobs activity where available job listings are retrieved from the system. Upon selecting a suitable job, the user proceeds to Apply for Jobs, completing their functional flow.

3.3 Class Diagram

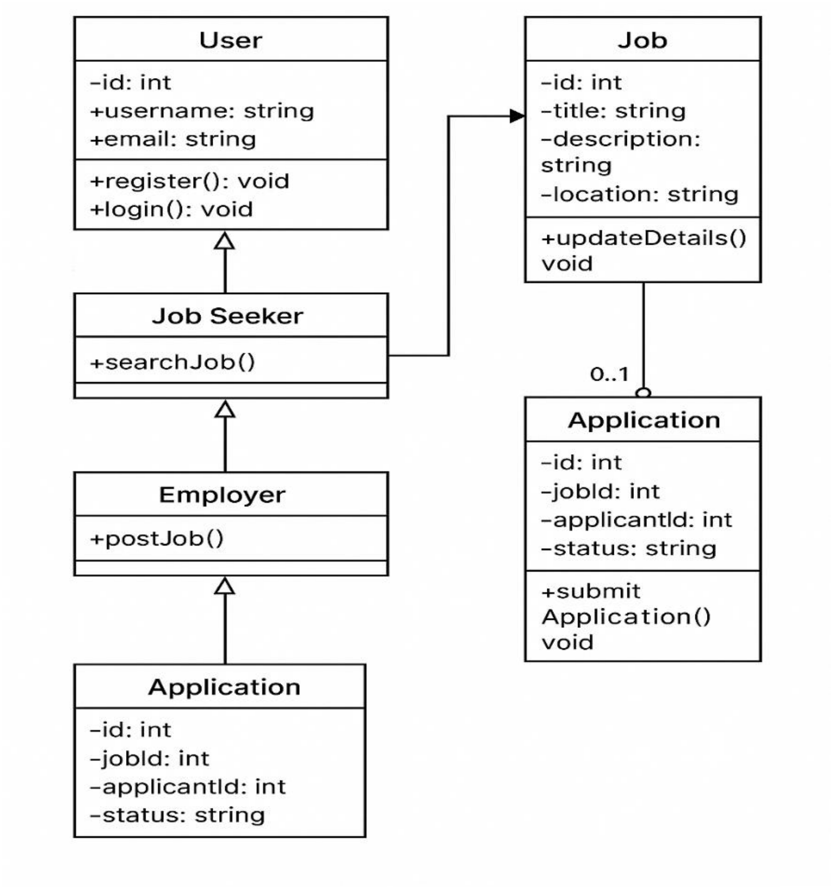


Figure.3.3 Class Diagram

3.3.1 Class Diagram Description

The class diagram models the core structure of a Job Portal system. The **User** class acts as a base class with attributes such as id, username, and email, along with operations for registration and login. Two specialized subclasses extend it: **Job Seeker**, which searches for jobs, and **Employer**, which posts job listings. The **Job** class represents job postings with attributes like title, description, and location, and provides an operation to update job details. The **Application** class captures the job application process, linking a Job Seeker to a Job through attributes such as jobId, applicantId, and status, and includes a method to submit an application. Relationships illustrate how users interact with jobs and applications, showing clear separation of responsibilities. The design supports modularity, reusability through inheritance, and well-defined associations between core system entities.

3.4 Sequence Diagram

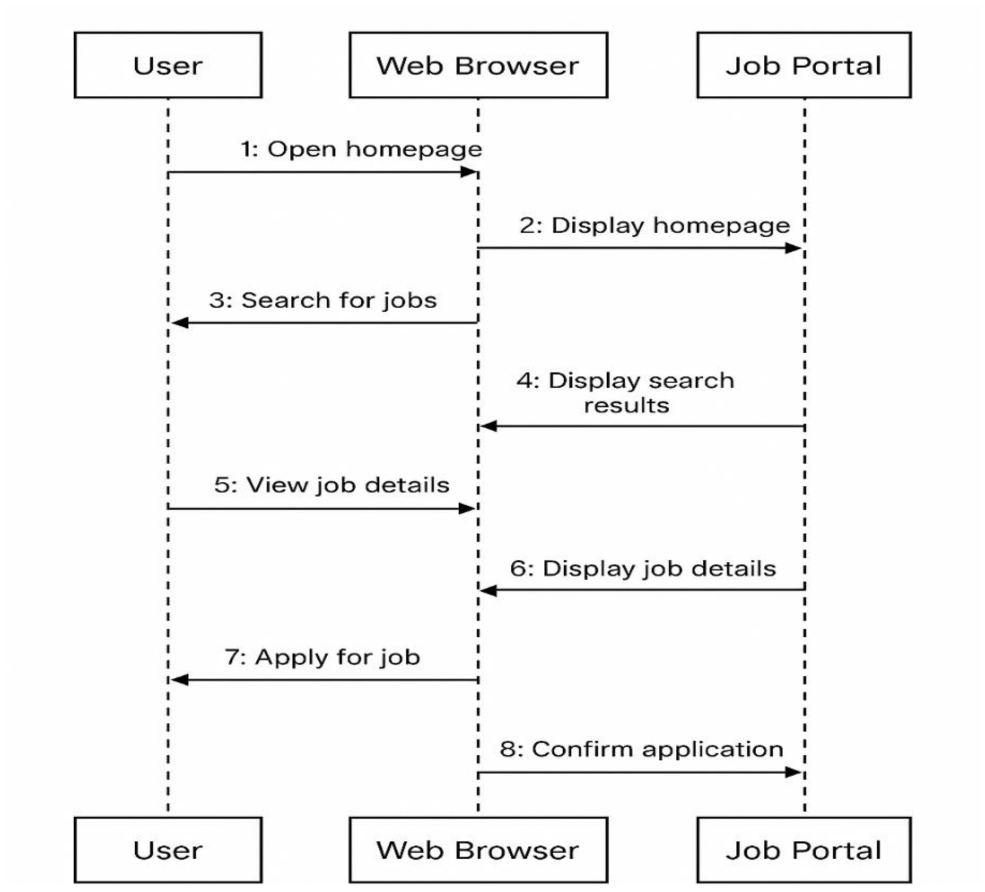


Figure.3.4 Sequence Diagram

3.4.1 Sequence Diagram Description

The sequence diagram illustrates the interaction flow between the User, Web Browser, and Job Portal during a job application process. The user begins by opening the homepage, which the browser requests and the portal displays. The user then searches for jobs, prompting the portal to return matching search results. After selecting a job, the browser requests job details, which the portal displays. Finally, the user applies for the job, and the Job Portal confirms the submission. The diagram effectively shows the message sequence, request–response interactions, and the logical progression of actions in the job application workflow. The portal validates the request, records the application, and sends a confirmation message back to the browser for display. This sequence clearly represents synchronous communication, ordered message flow, and the step-by-step behaviour of the system when handling a job application.

3.5 Collaboration Diagram

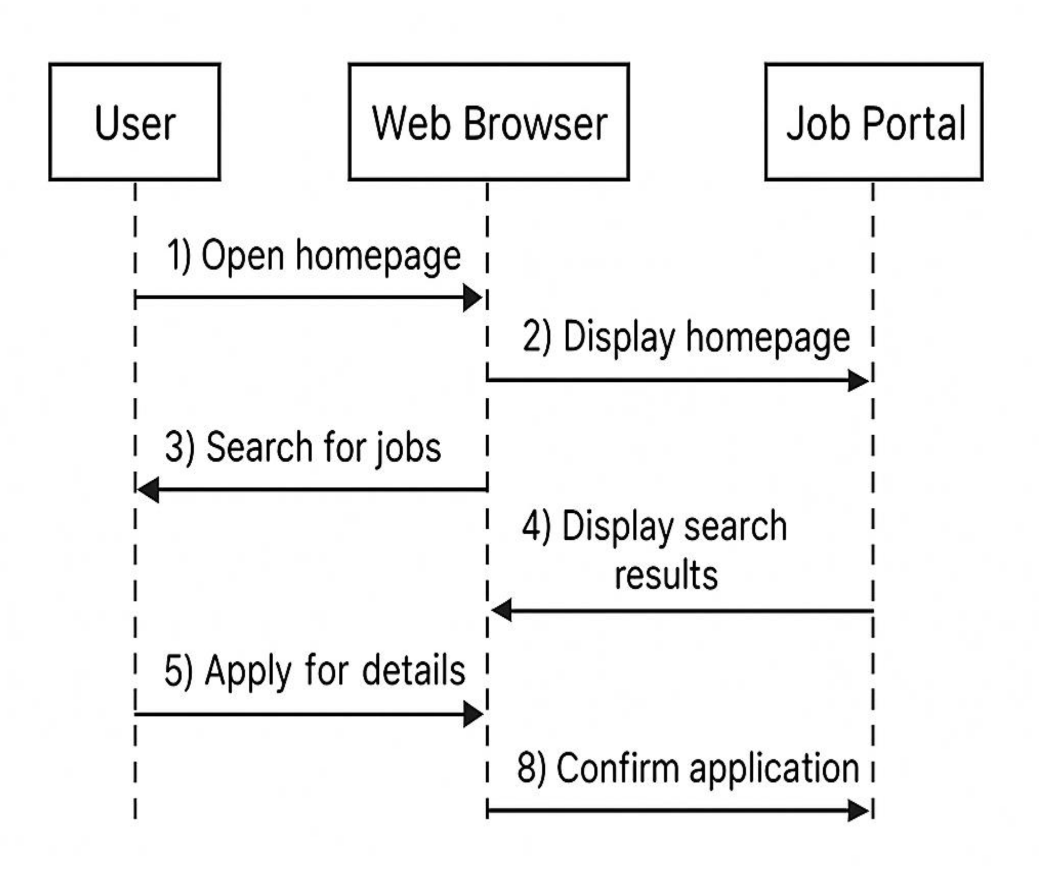


Figure.3.5 Collaboration Diagram

3.5.1 Collaboration Diagram Description

The sequence diagram illustrates the interaction flow between the User, the Web Browser interface, and the Job Portal backend during the job search and application process. The process begins when the user initiates a request to open the homepage. The Web Browser forwards this request to the Job Portal, which responds by sending the homepage content back to be displayed Next, the user searches for available jobs. The browser sends the search query to the Job Portal, which processes the request and returns the relevant search results. This demonstrates a typical request–response pattern between the client interface and the server. After reviewing the results, the user requests job details or proceeds to apply for a job. The browser sends this request to the Job Portal, which validates the action and returns an application confirmation message.

3.6 Component Diagram

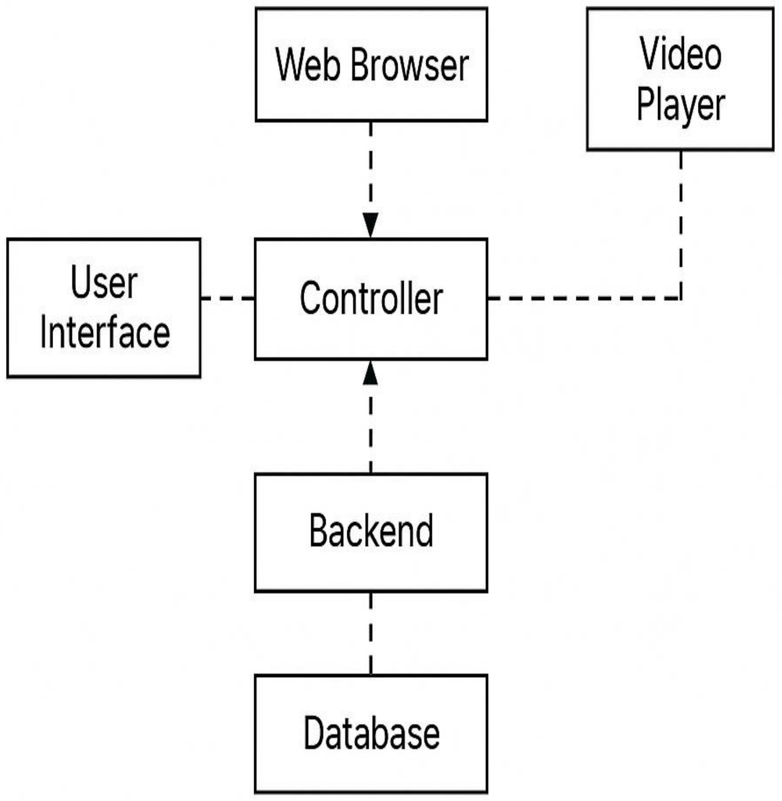


Figure.3.6 Component Diagram

3.6.1 Component Diagram Description

The diagram represents a simplified architecture showing the interaction between different components of a software system. The User Interface communicates with the Controller, which acts as the central coordinator for handling user actions and system requests. The Web Browser also interacts with the Controller, enabling clients to access the system through a web-based interface. The Backend component handles business logic and processes incoming requests from the Controller, while the Database stores and retrieves application data. Additionally, a Video Player component is loosely connected to the Controller, indicating optional or extended functionality such as media playback. The architecture clearly separates presentation, control, logic, and data layers, supporting modularity and easier system maintenance.

3.7 Deployment Diagram

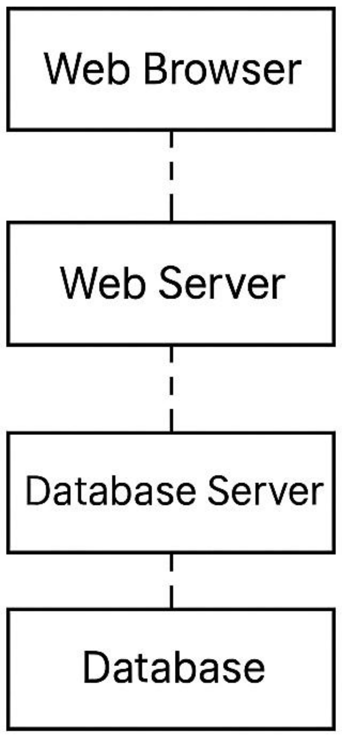


Figure.3.7 Deployment Diagram

3.7.1 Deployment Diagram Description

The diagram illustrates a simple three-tier architecture used in web-based applications. The Web Browser represents the client layer, where users interact with the system through a graphical interface. All user requests are sent to the Web Server, which handles application logic, processes incoming requests, and generates appropriate responses. The Web Server communicates with the Database Server, which manages database operations such as querying, updating, and retrieving stored data. The Database at the final layer stores all persistent system information. This layered structure separates presentation, application logic, and data management, ensuring Below the Web Server is the Database Server, which handles all operations related to data management. It processes SQL queries, ensures data integrity, manages transactions, and enforces security rules. The Database Server retrieves or updates information stored in the Database, which represents the persistent storage layer containing all structured application data.

3.8 Package Diagram

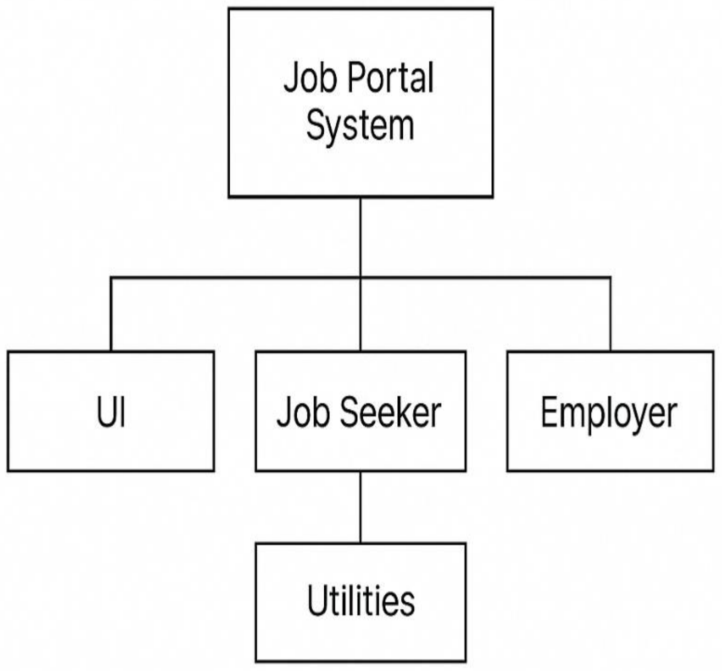


Figure.3.8 Package Diagram

3.8.1 Package Diagram Description

The diagram represents a high-level package structure for the Job Portal System. At the top, the Job Portal System package encapsulates the entire application, ensuring modular organization and separation of concerns. Beneath it, three primary functional packages are defined: UI Package – This package contains all presentation and interaction components. It manages screens, forms, and interfaces through which users access the system. It focuses on user experience and input/output handling. Job Seeker Package – This package contains all features related to job-seeking activities, including searching for jobs, viewing job details, and submitting applications. It represents one of the core user roles and its associated functionalities. Employer Package – This package handles employer-related operations such as posting jobs, managing listings, and reviewing applicants. It encapsulates all actions required for job providers within the system. Additionally, the Utilities Package is connected to the Job Seeker package. It contains reusable helper components such as validation functions, search filters, sorting utilities.

3.9 Design Patterns Used (GRASP, GoF)

In the Job Portal System, GRASP and GoF design patterns are applied to create a clean, maintainable, and scalable architecture. GRASP patterns such as Controller, Information Expert, Low Coupling, High Cohesion, and Polymorphism ensure proper responsibility assignment, reduced dependencies, focused class roles, and flexible behavior for different user types. GoF patterns further strengthen the design, with the Factory Method used to create user objects, Singleton managing a single database connection, Strategy enabling flexible job search filters, and Observer supporting user notifications. The MVC architectural pattern separates the system into Model, View, and Controller layers for better organization and easier maintenance. Together, these patterns improve system performance, flexibility, and future scalabilityGoF design patterns further strengthen the system, with the Factory Method Pattern used for creating user objects through a UserFactory to reduce tight coupling, the Singleton Pattern applied to the Database Connection Manager to ensure a single shared instance, and the Strategy Pattern enabling flexible job search filtering based on criteria such as location, salary, or experience. The Observer Pattern supports user notifications for job updates and application progress, enhancing dynamic interaction.. Together, these GRASP and GoF patterns contribute to a more scalable, robust, flexible, and efficient Job Portal System. Security limitations are also inherent in a frontend-only system. Without server-side validation, encryption mechanisms, or secure API handling, sensitive user actions cannot be protected effectively. Features such as secure login, password hashing, HTTPS enforcement, and data protection remain absent in the current prototype. As a result, the system cannot be deployed in a production environment without posing risks to user privacy and data integrity.

CHAPTER 4
IMPLEMENTATION

4.1 Module Description

The Job Portal System is organized into several functional modules that work together to provide a seamless experience for job seekers, employers, and administrators. The User Management Module handles registration, login, authentication, and profile management for all users. The Job Search Module allows job seekers to browse, filter, and view detailed job listings based on criteria such as location, role, and experience. The Job Posting Module enables employers to create, update, and remove job vacancies, ensuring that the portal always displays up-to-date opportunities. The Application Management Module manages application submissions, tracks application status, and allows employers to review applicants. The Admin Module oversees the entire platform by monitoring user activity, validating employer accounts, and ensuring system security.

4.2 Technology Description

The project is developed using a combination of web technologies that enable interactivity, responsiveness, and smooth navigation across pages. HTML is used for structuring the web pages, while CSS and Bootstrap enhance the visual presentation through layout styling, buttons, forms, and responsive components. JavaScript provides client-side interactivity, enabling form validation, dynamic content display, and interactive UI behavior. Image and video assets are included to enrich the user interface and improve user engagement. Although this prototype focuses on frontend development, it can be extended with backend technologies like PHP, Node.js, or Python Flask, along with a database such as MySQL or MongoDB for storing job postings, user profiles, and application data. These technologies together form a strong foundation for developing a fully functional job portal.

CHAPTER 5
TESTING

5.1 Testing Strategy

The testing strategy for the Job Portal System focuses on ensuring that all modules function correctly, user interactions behave as expected, and the platform provides a smooth and error-free experience. The system is tested using a combination of functional testing, usability testing, and interface validation. Functional testing checks whether core activities—such as user registration, login, job search, job posting, and application submission—work according to requirements. Usability testing ensures that all pages load correctly, navigation is easy, and users can complete tasks without confusion. Form validation testing verifies that incorrect or incomplete inputs trigger proper error messages. Compatibility testing is performed on different browsers and screen sizes to ensure responsiveness.

5.2 Sample Test Cases

Sample test cases for the Job Portal System cover major functionalities such as user authentication, job searching, job posting, and job application. A typical test case for login includes entering valid credentials to verify successful authentication, entering wrong credentials to confirm error messages, and checking that empty fields are not accepted. Job search test cases include verifying search results for specific keywords, filtering jobs by category or location, and checking whether unavailable jobs display proper messages. Job posting test cases validate if employers can successfully upload job information, edit listings, or delete postings. Application submission test cases ensure that job seekers can apply only once, required files like resumes upload correctly, and application status updates are visible. These test cases confirm that every user action leads to the correct system response. Following extensive functional, usability, and compatibility assessments, the final round of testing validated that the Job Portal System meets the defined performance benchmarks and satisfies all essential user and system requirements.

5.3 Test Module

The test results of the Job Portal System indicate that the major functionalities operated successfully, with most test cases passing on the first attempt and minor issues corrected through iterative debugging. User registration, login, and authentication performed reliably under different input conditions. Job search and filtering produced accurate results, while job posting and application submission modules handled data correctly without errors. Form validations worked as expected by preventing empty or incorrect inputs. Compatibility tests showed that the system displayed properly on multiple browsers and devices. After resolving small UI inconsistencies and validation issues, the final test cycle confirmed that the system meets the required functional and usability standards, demonstrating readiness for deployment as a frontend prototype.

Form validation components were also thoroughly tested, demonstrating effective prevention of empty, malformed, or invalid inputs. These controls ensured that users were guided to provide correct information, thereby enhancing overall usability and reducing the likelihood of operational errors. Compatibility testing across different browsers and device types confirmed that the system's layout, responsiveness, and visual elements rendered consistently, supporting smooth user experience regardless of the access platform. During testing, a few minor user interface irregularities and validation discrepancies were identified, but these were promptly addressed, leading to improved system coherence and polish The job search and filtering mechanisms delivered precise and relevant results, confirming that the search logic and interface interactions function correctly. Likewise, the modules responsible for job posting and application submission processed information without errors, maintaining the integrity and consistency of user-generated data..

CHAPTER 6
CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

The Job Portal System successfully demonstrates how a digital recruitment platform can streamline the interaction between job seekers and employers through an organized, user-friendly interface. By applying Object-Oriented Analysis and Design principles, the system achieves a structured flow from requirements to implementation, ensuring clarity and modularity in its design. The use of UML diagrams, design patterns, and layered architecture helps maintain a strong foundation for scalability and maintainability. Core features such as user registration, job search, job posting, and application submission function smoothly, validating the effectiveness of the system's design. As a frontend prototype, it provides a complete representation of how a real-world job portal operates and serves as a solid base.

6.2 Future Enhancement

The Job Portal System can be further enhanced by integrating backend technologies such as PHP, Node.js, or Python Flask, along with a database like MySQL or MongoDB to store user and job data securely. Additional features like AI-based job recommendations, automated resume screening, real-time chat between employers and applicants, and advanced search filters can significantly improve user experience. Implementing email or SMS notifications for application updates, employer verification systems, and secure authentication methods such as OTP or OAuth would strengthen reliability and security. Expanding the system into a mobile application and deploying it on cloud platforms would enhance accessibility and performance. These enhancements would transform the prototype into a full-scale, dynamic job portal capable of supporting a large number of users efficiently.

APPENDIX A – SOURE CODE

User Profile

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-width, initial-
scale=1.0"/>
  <title>My Profile</title>
  <link rel="icon" type="image/png" href="Images/icon.png">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/js/bootstrap.bundle.min.js"></s
cript>
</head>
<body>
  <div class="background-overlay"></div>
  <div class="content">
    <div class="container">
      <div class="row justify-content-center">
        <div class="col-md-8">
          <div class="profile-card">
            <div class="profile-header">
              <div class="avatar-upload-area">
                <img id="profileAvatar" src="Images/default-avatar.png" alt="Profile
Avatar" class="profile-avatar">
                <input type="file" id="avatarFile" accept="image/*" style="display: none;">
                <div class="overlay-text"
onclick="document.getElementById('avatarFile').click()">
                    Change Photo
                </div>
              </div>
              <h2 id="profileName">User Profile</h2>
              <p id="profileRole" class="text-muted"></p>
            </div>

            <div id="jobSeekerProfile">
              <h4>Personal Information</h4>
              <div class="row mb-3">
                <div class="col-md-6">
                  <label class="form-label">Full Name</label>
                  <input type="text" class="form-control" id="profileFullName">
  <div class="mb-3">
              <label class="form-label">Skills</label>
              <input type="text" class="form-control" id="profileSkills"
placeholder="Add skills separated by commas">
              <div id="skillsDisplay" class="mt-2"></div>
            </div>

            <div class="mb-3">
              <label class="form-label">Experience</label>
              <textarea class="form-control" id="profileExperience"
rows="3"></textarea>
            </div>
            <div class="mb-3">
              <label class="form-label">Education</label>
              <textarea class="form-control" id="profileEducation" rows="3"></textarea>
            </div>
            <div class="mb-3">
              <label class="form-label">Preferred Job Types</label>
              <input type="text" class="form-control" id="profileJobTypes"
placeholder="Add preferred job types separated by commas">
              <div id="jobTypesDisplay" class="mt-2"></div>
            </div>
          </div>
            <div class="mb-3">
              <label class="form-label">Company Description</label>
              <textarea class="form-control" id="companyDescription"
rows="4"></textarea>
            <div class="mb-3">
              <label class="form-label">Experience</label>
              <textarea class="form-control" id="profileExperience"
rows="3"></textarea>
            <div class="mb-3">
              <label class="form-label">Preferred Job Types</label>
              <input type="text" class="form-control" id="profileJobTypes"
placeholder="Add preferred job types separated by commas">
              <div id="jobTypesDisplay" class="mt-2"></div>

            </div>
          </div>
          <div class="d-flex justify-content-between mt-4">
            <a href="index.html" class="btn btn-coffee">Back to Home</a>
            <button id="saveProfile" class="btn btn-coffee">Save Profile</button>
          </div>
</body>
</html>
```

Index

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Job Portal - Home</title>
  <link rel="icon" type="image/png" href="Images/icon.png">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/js/bootstrap.bundle.min.js"></s
cript>
</head>
<body>
  <div class="user-profile" id="userProfileContainer" style="display: none;">
    <div class="position-relative">
      <div id="userAvatarContainer" class="user-avatar-placeholder">
        </div>
      <span id="notificationBadge" class="notification-badge" style="display:
none;">0</span>
    </div>
    <div id="userMenu" class="user-menu">
      <a href="user-profile.html">My Profile</a>
      <a href="applied-job.html">Applied Jobs</a>
      <a href="job-recommendations.html">Recommended Jobs</a>
      <a href="#" id="logoutBtn">Logout</a>
    </div>
  </div>
  <div class="container-wrapper">
    <div class="glass-card">
      <a href="about.html" class="btn btn-outline-light btn-custom about-btn">About
Us</a><br>
      <h1>Welcome to Job Portal</h1>
      <p>Turning <strong>dreams</strong> into <strong>careers</strong></p>

        <a href="post-job.html" class="btn btn-light btn-custom me-3 user-company-
link" style="display: none;">Post a Job</a>
        <a href="view-applicants.html" class="btn btn-outline-light btn-custom user-
company-link" style="display: none;">View Applicants</a>
```

Admin

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Admin Dashboard</title>
  <link rel="icon" type="image/png" href="Images/icon.png">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/js/bootstrap.bundle.min.js"></s
cript>
</head>
<body>


  <div class="background-overlay"></div>
  <div class="content">
    <div class="container">
      <h2 class="text-white mb-4"><i class="bi bi-gear-fill me-2"></i>Admin
Dashboard</h2>

      <div class="alert alert-danger" id="authError" style="display: none;">
        Access Denied. Only Administrators can view this page.
      </div>


      <ul class="nav nav-tabs" id="myTab" role="tablist">
        <li class="nav-item" role="presentation">
          <button class="nav-link active" id="jobs-tab" data-bs-toggle="tab" data-bs-
target="#jobs" type="button" role="tab">All Posted Jobs</button>
        </li>
        <li class="nav-item" role="presentation">
          <button class="nav-link" id="applications-tab" data-bs-toggle="tab" data-bs-
target="#applications" type="button" role="tab">All Applications</button>
        </li>
      </ul>


      <div class="tab-content admin-card" id="myTabContent">
```

```
          <h4>All Jobs in the Portal</h4>
          <table class="table table-striped" id="jobsTable">
            <thead>
              <tr>
                <th>Job Title</th>
                <th>Company</th>
                <th>Location</th>
                <th>Salary</th>
                <th>Posted By (ID)</th>
              </tr>
            </thead>
            <tbody>
              </tbody>
          </table>
        </div>


        <div class="tab-pane fade" id="applications" role="tabpanel" aria-
labelledby="applications-tab">
          <h4>All Submitted Applications</h4>
          <table class="table table-striped" id="applicationsTable">
            <thead>
              <tr>
                <th>Job Title</th>
                <th>Applicant Name</th>
                <th>Email</th>
                <th>Phone</th>
                <th>Applied At</th>
              </tr>
            </thead>
            <tbody>
              </tbody>
          </table>
        </div>


      </div>

      <div class="mt-4 text-center">
        <a href="index.html" class="btn btn-coffee"><i class="bi bi-house-fill me-
2"></i>Back to Home</a>
    </div>
</body>
</html>
```

APPENDIX B - SCREENSHOTS

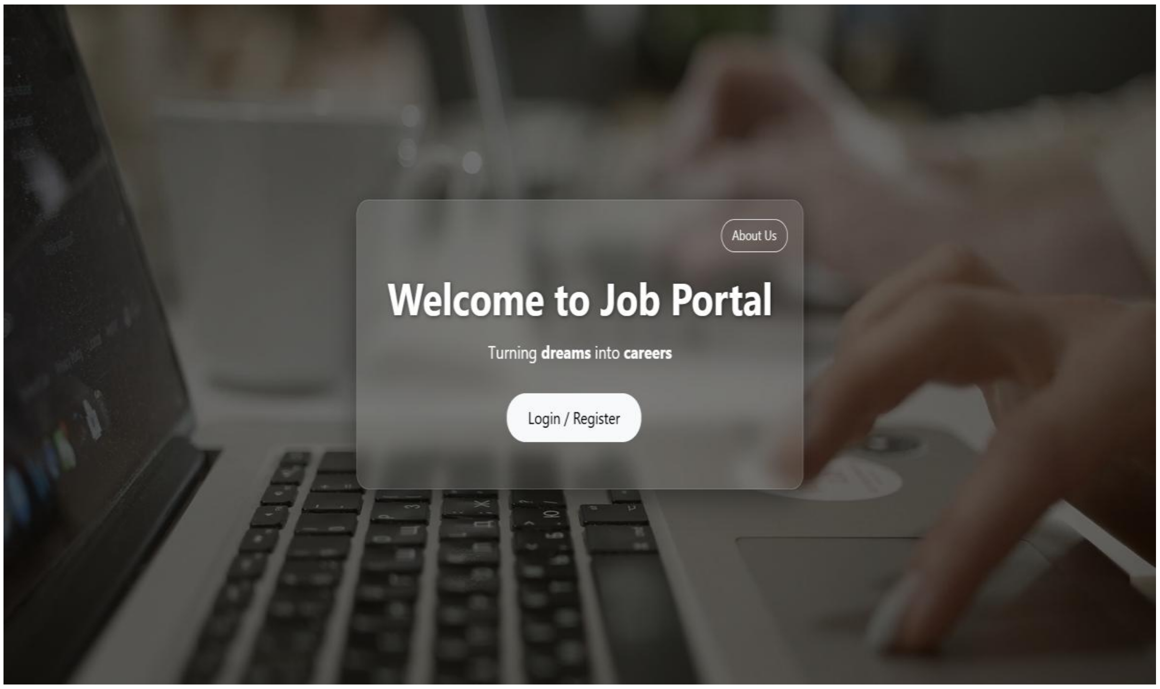


Figure.B.1 Index

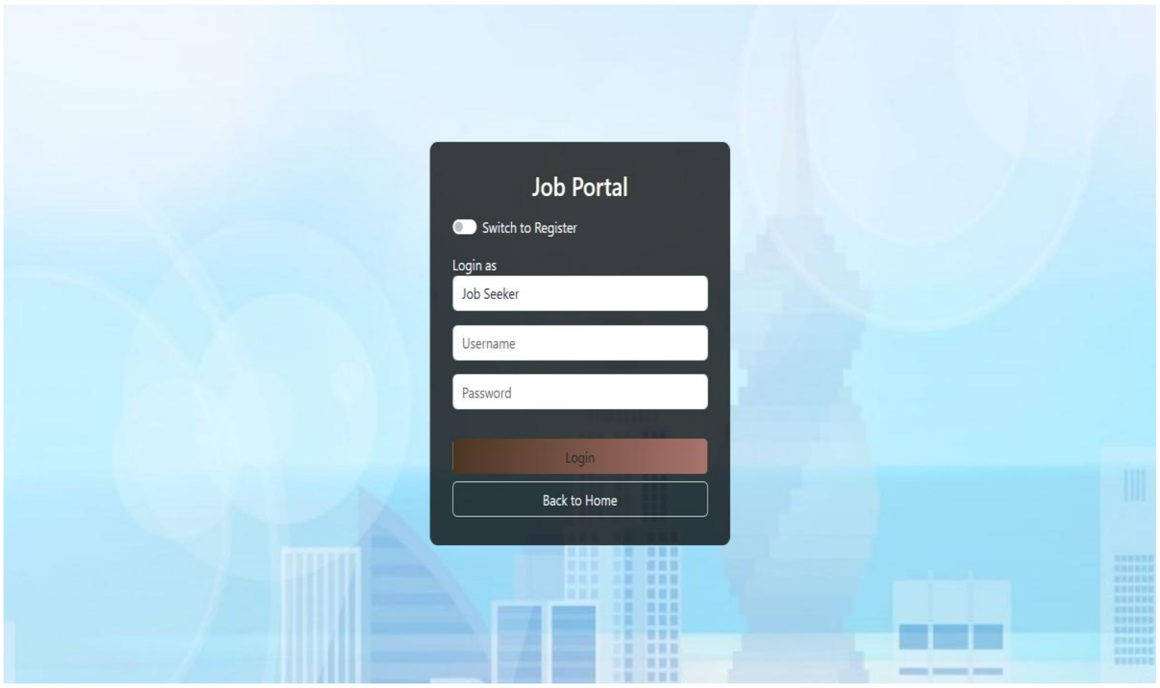
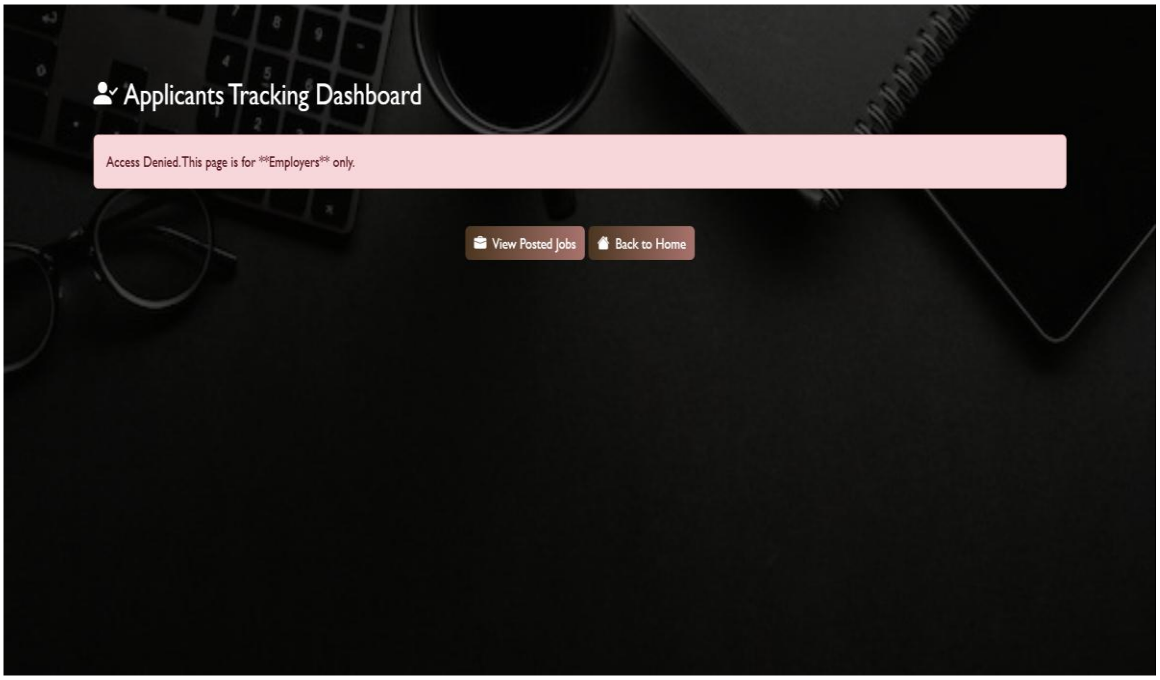
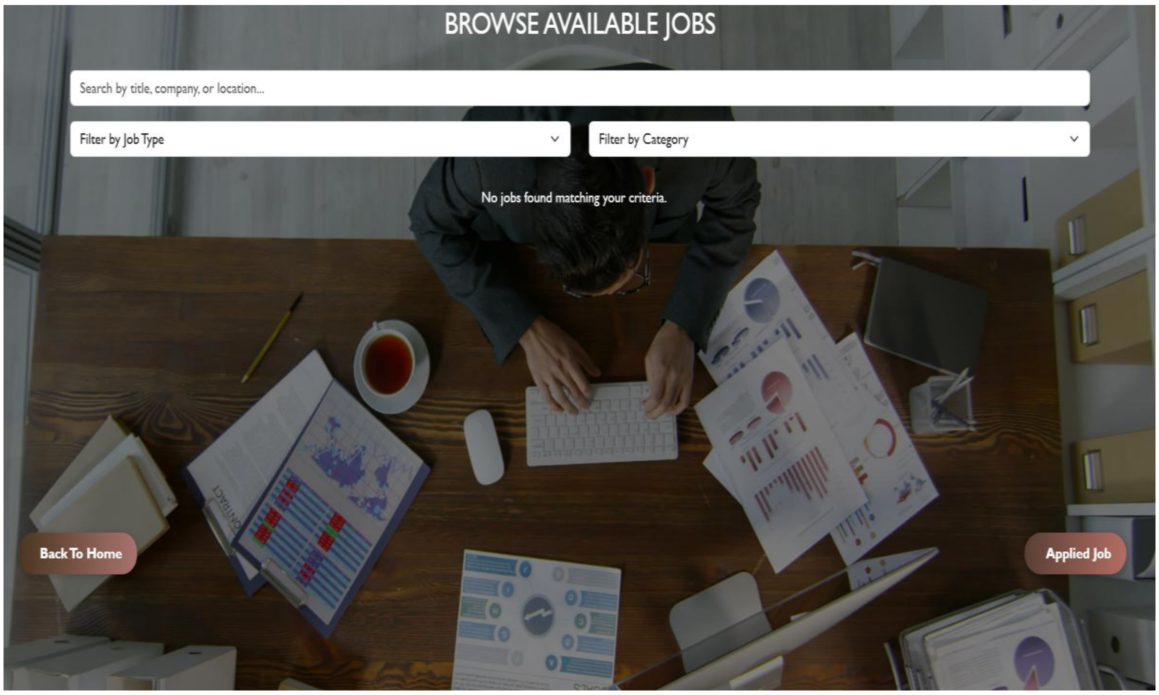


Figure.B.2 Login



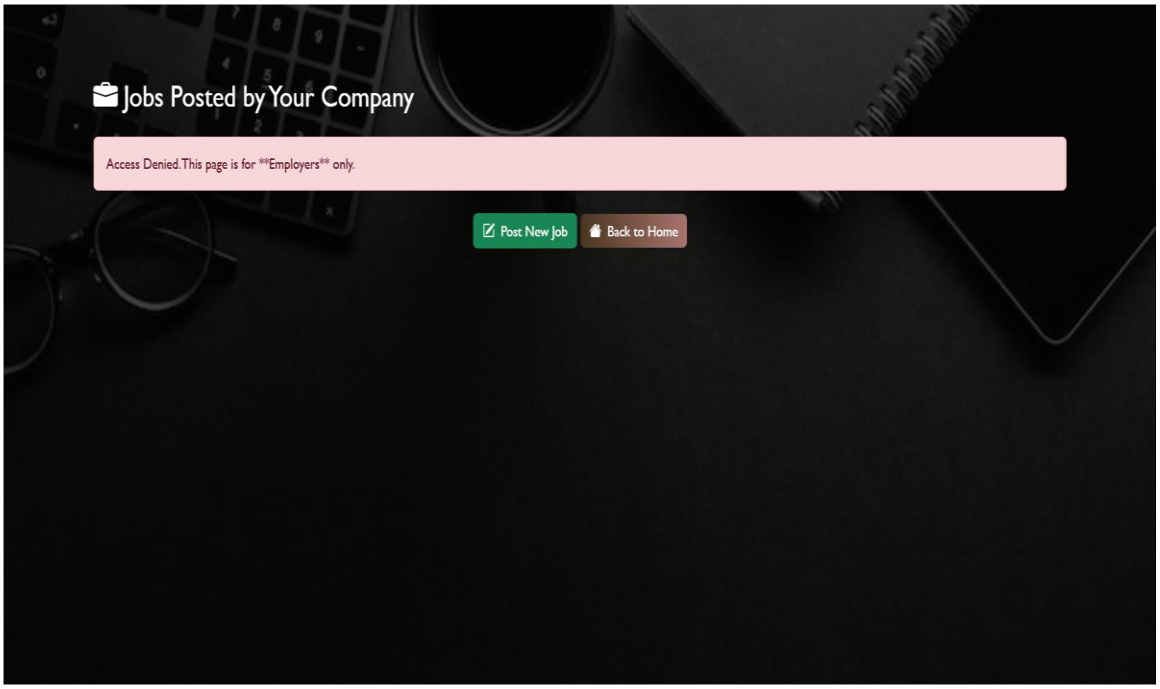
Access Denied.This page is for **Employers** only.

Figure.B.3 View Applicants



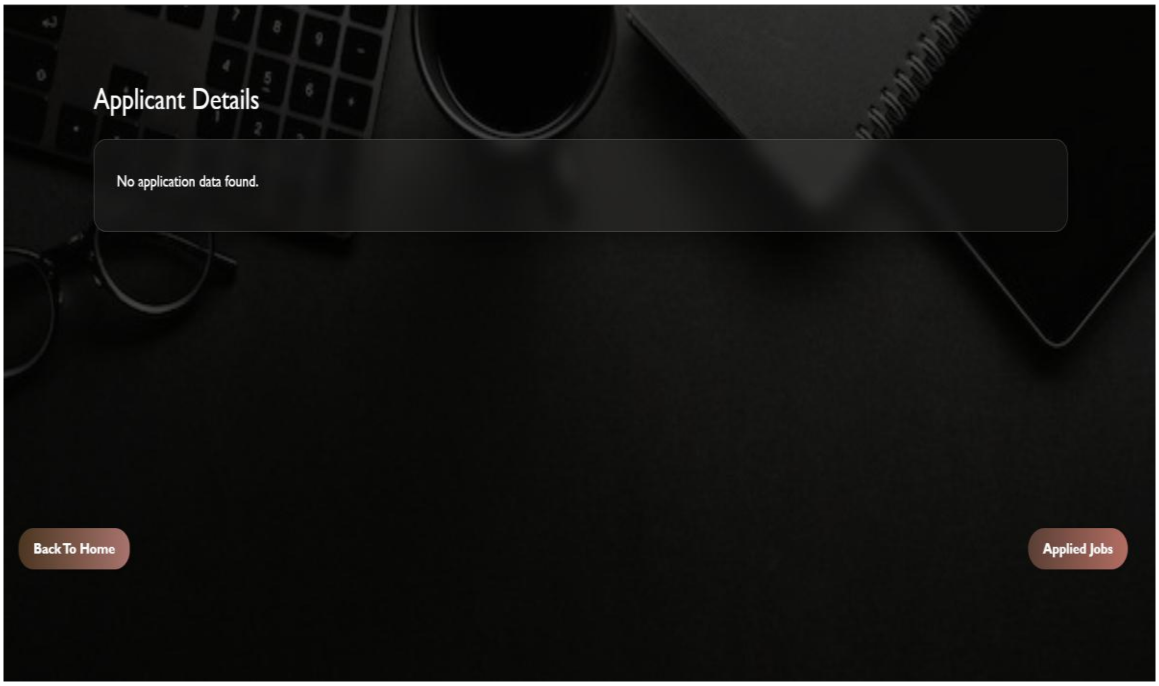
No jobs found matching your criteria.

Figure.B.4 Search Job


Jobs Posted by Your Company

Access Denied. This page is for **Employers** only.

Figure.B.5 Posted Jobs


Applicant Details

No application data found.

Figure.B.6 Info

REFERENCES

1. https://www.w3schools.com/

2. https://www.php.net/manual/en/

3. https://www.youtube.com/watch?v=OK_JCtrrv-c

4. https://www.youtube.com/watch?v=W6NZfCO5SIk

5. "Modern PHP: New Features and Good Practices" by Josh Lockhart

6. "Learning MySQL and MariaDB" by Russell J.T. Dyer

7. "JavaScript and JQuery: Interactive Front-End Web Development" by Jon Duckett .

8. "HTML and CSS: Design and Build Websites" by Jon Duckett