**JOB PORTAL MANAGEMENT SYSTEM**

**SUBMITTED BY : *LAILA***

TABLE OF CONTENTS

**1.INTRODUCTION…………………………………………………….2**

* 1. PURPOSE………………………………………………………………………………………………………2
  2. SCOPE……………………………………………………………………………………………………………3
  3. OBJECTIVES……………………………………………………………………………………………………3

**2.FUNCTIONAL REQUIREMENTS…………………………………4**

2.1 PRIMARY ACTORS OF THE SYSTEMS………………………………………………………………..4

2.2 USER CASE DIAGRAMS…………………………………………………………………………………….5

2.2.1 ADMIN…………………………………………………………………………………………………………..5

2.2.2 JOB PROVIDER……………………………………………………………………………………………….6

2.2.3 JOB SEEKER……………………………………………………………………………………………………7

2.3 USER STORIES……………………………………………………………………………………………………8

2.3.1 ADMIN……………………………………………………………………………………………………………8

2.3.2 JOB PROVIDER………………………………………………………………………………………………..9

2.3.3 JOB SEEKER……………………………………………………………………………………………………10

2.4 BUSINESS PROCESS DIAGRAM………………………………………………………………………….11

3 **NON-FUNCTIONAL REQUIREMENTS**…………………………………………………11

4 **TECHENICAL REQUIREMENTS**…………………………………………………………….12

5 CONCLUSION ……………………………………………………………………………………..13

1.INTRODUCTION

**1.1 scope**

The Scope section of a Software Requirements Specification (SRS) document outlines the boundaries and objectives of the software system being developed. It provides a high-level overview of what the system is intended to achieve, the key features it will include, and the specific problems it aims to solve. This section identifies the primary users and stakeholders of the system, describing how they will interact with it. It also clearly defines what functionalities are within the project's scope and what is deliberately excluded, helping to set expectations and avoid misunderstandings. Additionally, it may mention any assumptions, constraints, or dependencies that impact the project. By establishing a well-defined scope, the development team can stay focused on agreed goals, reduce the risk of scope creep, and ensure that all stakeholders have a common understanding of the system's purpose and limitations.

**1.2 purpose**

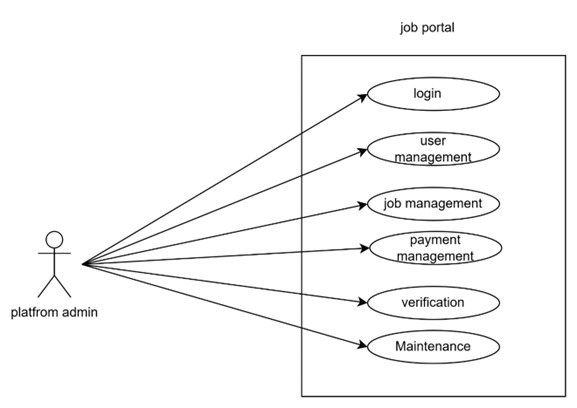
The Purpose section of a Software Requirements Specification (SRS) document describes the reason for developing the software system and what it is expected to achieve. It explains the overall objectives of the system from a business or user perspective, outlining the problems it intends to solve or the improvements it aims to bring. This section also identifies the intended audience of the document, such as developers, testers, project managers, or clients. Clearly defining the purpose ensures that all stakeholders understand the motivation behind the project and the value it will deliver.

**1.3 objectives**

The **Objectives** section of a Software Requirements Specification (SRS) document outlines the specific goals that the software system is expected to accomplish. These goals are typically aligned with solving a particular problem, improving a process, or meeting user or business needs. Objectives should be clear, measurable, and achievable, guiding the development team throughout the project. This section helps ensure that all stakeholders have a shared understanding of what the system is meant to deliver. Examples of objectives might include automating manual tasks, reducing processing time, improving data accuracy, or enhancing user experience.

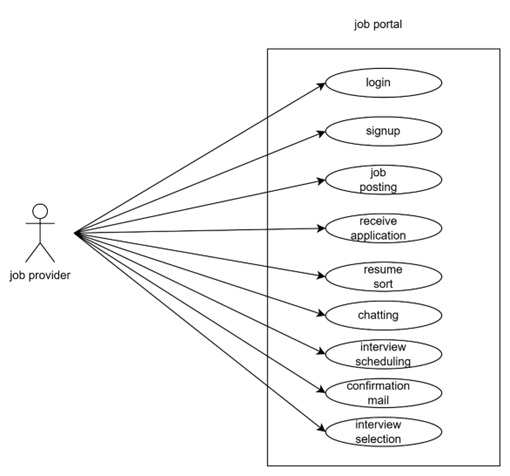
2.FUNCTIONAL REQUIREMENTS

2.1 PRIMARY ACTORS OF THE SYSTEM

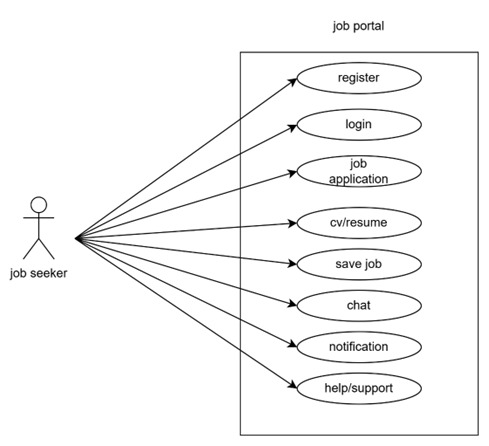
The **Primary Actors** of the system are the main users or entities that directly interact with the software to perform specific tasks or achieve particular goals. These actors are essential for defining how the system will be used and for identifying the various use cases the system must support. Typically, primary actors include individuals such as end users, administrators, and managers, as well as external systems that interface with the application. For example, in an e-commerce platform, the primary actors might include customers who browse and purchase products, administrators who manage product listings and orders, and external payment gateways that handle transaction processing. Clearly identifying these actors helps ensure the system is designed to meet their needs and supports all necessary interactions efficiently

2.2 USE CASE DAIGRAMS

2.2.1 ADMIN

2.2.2 JOB PROVIDER

2.2.3 JOB SEEKER

****

**2.3 USER STORIES**

2.3.1 JOB SEEKER

|  |  |  |
| --- | --- | --- |
| AS a (type of user) | I need to(do some task) | So that I can(get some result) |
| Job seeker | Login into the application by entering personal details | access the platform to find oppurtunities |
| Job seeker | upload my CV/Resume | Showcase my skills and provide details to the job provider |
| Job seeker | Search and apply for the job | Be eligible in the eyes of the employer in order to obtain the desired position |
| Job seeker | Save jobs | I can apply later or track interesting jobs |
| Job seeker | Message and Get notifications | I can keep contact with the employers |
| Job seeker | Access help/support | I can resolve any issues with the account |
|  |  |  |

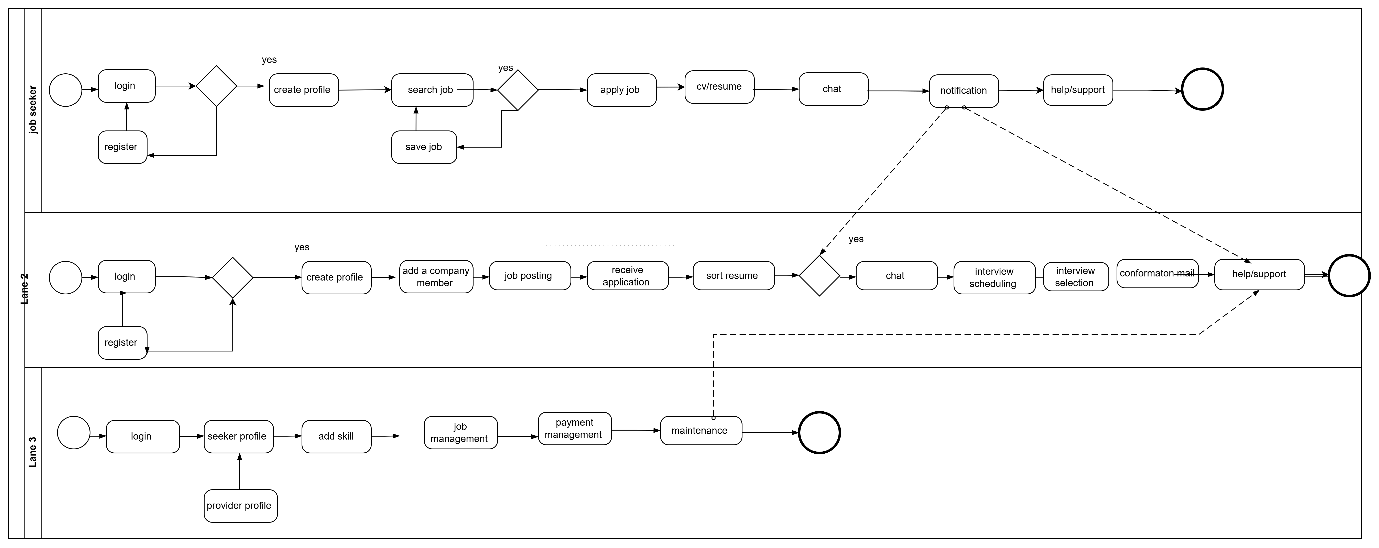
|  |  |  |
| --- | --- | --- |
| AS A [type of user] | I NEED TO [ do some task] | SO THAT I CAN [get some result] |
| Job provider | Login | Register and job details |
| Job provider | Finding candidates | Searching people who have the right skill and experience |
| Job provider | Posting job advertisements | Sharing with website, newspapers, or social media. |
| Job provider | Receive application | Checking resumes and application to shortlist |
| Job provider | Coordinating interviews | Scheduling meeting with candidates. |
| Job provider | Conformation mail | Helping candidates improve if they don’t get selected or select |

2.3.2 JOB PROVIDER

2.3.3 PLATFORM ADMIN

|  |  |  |
| --- | --- | --- |
| **AS A [type of user]** | **I NEED TO [do some task]** | **SO THAT I CAN [get some result]** |
| **Platform Admin** | **Login to the portal by entering details** | **Ensure a authorized access to the platform, different user have different login details** |
| **Platform Admin** | **Maintain user accounts and permission** | **Allows job seekers and employers to create profiles, providing essential details, job seekers and job providers can track application status** |
| **Platform Admin** | **Maintain payments and subscription plans** | **Allows users to unlock premium levels, users can upgrade or cancel subscription, improves the growth** |
| **Platform Admin** | **Job management and details** | **Ensure accurate, up-to-date listings for job seekers, help users to get the job opportunities easily** |
| **Platform Admin** | **Perform Regular updates and maintenance** | **Keep the platform running smoothly with out any errors or bugs** |

2.4 BUSINESS PROCESS DIAGRAM



1. NON-FUNCTIONAL REQUIREMENTS
2. **Performance**

The system should respond to user queries (e.g., job search, login) within 2 seconds under normal load.

It should be able to handle at least 1,000 concurrent users without degradation in performance.

1. **Scalability**

The system must support future growth in terms of users, job postings, and data without requiring major architectural changes.

It should be scalable both horizontally (adding servers) and vertically (upgrading system resources).

1. **Reliability**

The system should maintain at least 99.9% uptime and ensure that job listings and applications are not lost due to system failure.

It should be able to recover automatically from minor failures without user intervention.

1. **Security**

* All user data must be encrypted and stored securely.
* The system must implement role-based access control (RBAC) for admins, employers, and job seekers.
* It should protect against common vulnerabilities like SQL injection, XSS, and CSRF.

1. **Usability**

* The user interface should be intuitive and accessible for users of all technical levels.
* The system should follow standard UI/UX guidelines to ensure ease of navigation and interaction.

1. **Maintainability**

* The codebase should be modular and well-documented to facilitate easy updates and bug fixes.
* The system should support regular updates with minimal downtime.

1. **Availability**

* The system should be available 24/7, with scheduled maintenance communicated in advance.
* Backup mechanisms should be in place to ensure data availability in case of system crashes.

1. **Portability**

* The application should be accessible across different platforms (Windows, macOS, Linux) and devices (desktops, tablets, smartphones).
* It should be compatible with major web browsers like Chrome, Firefox, Safari, and Edge.

1. **Compliance**

* The system must comply with data protection regulations such as GDPR or other relevant laws depending on the deployment region

4 TECHNICAL REQUIREMENTS

|  |  |
| --- | --- |
| Processor | Intel Core i5 |
| Hard Disk | 40GB to 80GB |
| Memory | 8GB |
| OS | Windows 11 |
| Front end | HTML, CSS, JavaScript, Bootstrap and Angular |
| Back end | ASP.NET CORE |
| Database | MS SQL |
| IDE | Visual Studio ,visual studio code |

5 CONCLUSION

The Job Portal Management System project successfully provides a platform that bridges the gap between job seekers and employers through an efficient, user-friendly, and secure online interface. It streamlines the recruitment process by enabling job postings, resume submissions, and application tracking. With its focus on functionality, performance, and scalability, the system is designed to meet the evolving needs of the employment market. This project lays a solid foundation for future enhancements and can be further expanded to include advanced features such as AI-based job matching and real-time communication between users.