WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM

REBAZ FARID NOORI

UNIVERSITI TEKNOLOGI MALAYSIA

UNIVERSITI TEKNOLOGI MALAYSIA

DECLARATION OF THESIS / UNDERGRADUATE PROJECT REPORT AND COPYRIGHT

Author's full name :REBAZ FARID NOORI

Date of Birth : 20 Feb 1998

Title : WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM

Academic Session :

I declare that this thesis is classified as:

☐	**CONFIDENTIAL**	(Contains confidential information under the Official Secret Act 1972)*
☐	**RESTRICTED**	(Contains restricted information as specified by the organization where research was done)*
☑	**OPEN ACCESS**	I agree that my thesis to be published as online open access (full text)

1. I acknowledged that Universiti Teknologi Malaysia reserves the right as follows:
2. The thesis is the property of Universiti Teknologi Malaysia
3. The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of research only.
4. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

SIGNATURE OF STUDENT	**SIGNATURE OF SUPERVISOR**
QU180SCSR013	**MR.AKO ABUBAKR JAFFAR**
MATRIX NUMBER	**NAME OF SUPERVISOR**
Date: 25 JUNE 2022	Date: 25 JUNE 2022

NOTES : If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter from the organization with period and reasons for confidentiality or restriction

May 2023

QIU Library

Sir,

CLASSIFICATION OF THESIS AS OPEN
WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM
REBAZ FARID NOORI

Please be informed that the above-mentioned thesis entitled “WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM” be classified as OPEN ACCESS.

Thank you.

Sincerely yours.

AKO ABUBAKR JAAFAR, As Sulaymaniyah Iraq, +964 770 248 5353

"I hereby declare that we have read this thesis and in my
opinion this thesis is suffcient in term of scope and quality for the
award of the degree of BSc of Computer Science (Computer Network & Security)"

Signature	:	_____
Name of Supervisor	:	AKO ABUBAKR JAAFAR
Date	:	25 JUNE 2022

WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM

REBAZ FARID NOORI

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Bachelor of Computer Science (Computer Network & Security)

School of Computing
Faculty of Engineering
Universiti Teknologi Malaysia

JUNE 2022

DECLARATION

I declare that this thesis entitled *"WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM "* is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature	:	.....
Name	:	REBAZ FARID NOORI
Date	:	25 JUNE 2022

DEDICATION

I dedicate this project to God Almighty my creator, my strong pillar, my source of inspiration, wisdom, knowledge and understanding. He has been the source of my strength throughout this program and on His wings only have I soared. I would also like to thank my family. A special feeling of gratitude to my loving parents, whose words of encouragement and push for tenacity ring in my ears.

Thank you. My love for you all can never be quantified. God bless you.

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed to my understanding and thoughts. In particular, I wish to express my sincere appreciation to my main thesis supervisor, Mr. Ako, for encouragement, guidance, critics and friendship. I am also very thankful to my co-supervisor Ts. Dr. Sarina Sulaiman for their guidance, advices and motivation. I would also like to give special thanks to my fellow friends, Mr. Siroz, Mr. Renas, and lastly Mr. Saman Without their continued support and interest, this thesis would not have been the same as presented here.

My fellow student should also be recognised for their support. My sincere appreciation also extends to all my colleagues and others who have provided assistance at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. I am grateful to all my family member.

ABSTRACT

Employments systems play a vital role in today's communities it allows for a much more diverse range of applications and this serves both employers and job-seekers within a community. Since normal hiring processes had to rely on personal connections the employer had a much less diverse range of people to choose from, however with today's technologies recruiters can reach out to countries abroad just to select the right candidate which has helped employers and job-seekers with the right skills grow further. The purpose of this project is to review the current methods and systems used for publishing open job vacancies and hiring job-seekers or allow them to search for new opportunities within Kurdistan Regional Governate (KRG). The main goal is to Develop an alternative system to fix the addressed issues found while reviewing the existing systems. The process starts with conducting several case studies to better understand the existing system and its issues. There are several methodologies used for system development, it'll analyse and determine the best possible methodology for the system development. Then started designing the basics of the system for better understanding. The overall implementation which takes place in FYP2 is conducted after all aspects of the problem is considered. This system focuses on providing a better solution that provides more reliable and easy to use, for both users of the system. This system will be implemented using React.Js Framework a combination of JavaScript, HTML, and CSS that allows creating dynamic web applications. The system should be reliable to use by Employers and Job-Seekers to search, and find jobs or add new vacancies and find new employers.

ABSTRAK

Sistem pekerjaan memainkan peranan penting dalam komuniti hari ini yang membolehkan pelbagai aplikasi yang lebih pelbagai dan ini memberi perkhidmatan kepada majikan dan pencari kerja dalam komuniti. Memandangkan proses pengambilan biasa terpaksa bergantung pada hubungan peribadi, majikan mempunyai lebih kurang pelbagai orang untuk dipilih, namun dengan teknologi hari ini pereku boleh menjangkau negara di luar negara hanya untuk memilih calon yang tepat yang telah membantu majikan dan pencari kerja kemahiran yang betul berkembang lagi. Tujuan projek ini adalah untuk mengkaji kaedah dan sistem semasa yang digunakan untuk menerbitkan kekosongan kerja terbuka dan mengambil pencari kerja atau membenarkan mereka mencari peluang baharu dalam Kurdistan Regional Governate (KRG). Matlamat utama adalah untuk Membangunkan sistem alternatif untuk membetulkan isu yang ditangani yang ditemui semasa menyemak sistem sedia ada. Proses ini bermula dengan menjalankan beberapa kajian kes untuk lebih memahami sistem sedia ada dan isu-isunya. Terdapat beberapa metodologi yang digunakan untuk pembangunan sistem, kami akan menganalisis dan menentukan kaedah terbaik yang mungkin untuk pembangunan sistem kami. Kemudian mula mereka bentuk asas sistem untuk pemahaman yang lebih baik. Pelaksanaan keseluruhan yang berlaku dalam PSM2 dijalankan selepas semua aspek masalah dipertimbangkan. Sistem ini memberi tumpuan kepada menyediakan penyelesaian yang lebih baik yang menyediakan lebih dipercayai dan mudah untuk digunakan, untuk kedua-dua pengguna sistem. Sistem ini akan dilaksanakan menggunakan Rangka Kerja React.Js gabungan JavaScript, HTML dan CSS yang membolehkan mencipta aplikasi web dinamik. Sistem ini harus boleh dipercayai untuk digunakan oleh Majikan dan Pencari Kerja untuk mencari, dan mencari pekerjaan atau menambah kekosongan baru dan mencari majikan baharu.

Table of Contents

LIST OF TABLES

TABLE NO. **TITLE** **PAGE**

LIST OF FIGURES

FIGURE NO. TITLE PAGE

LIST OF ABBREVIATIONS

UN SDGs	-	United Nations Sustainable Development Goals
CV	-	Curriculum Vitae
KRG	-	Kurdistan Regional Governate
NGO	-	Non-Profit Organizations
IT	-	Information Technology
JS	-	JavaScript
AJAX	-	Asynchronous JavaScript and XML
RAM	-	Random Access Memory
JSX	-	JavaScript Syntax Extension
IDE	-	integrated development environment
Gen	-	Generation
QA	-	Question & Answer
DOM	-	Distributed Object Management
CSS	-	Cascading Style Sheets
HTTP	-	Hypertext Transfer Protocol
OS	-	Operating System
VS Code	-	Visual Studio Code
UI/UX	-	User Interface / User Experience
UML	-	Unified Modeling Language
API	-	Application Programming Interface
MERN	-	MongoDB, Express, React, Node
NCCI	-	NGO Coordination Committee for Iraq
GUI	-	graphical user interface
SRD	-	Software Requirement Documentation
STD	-	Software Testing Documentation
SDD	-	Software Design Documentation

LIST OF APPENDIX

CHAPTER 1

INTRODUCTION

1.1. Overview

There is always a constant search for jobs and opportunities inside every community. This consists of people looking to get their first job or aiming for something better. When looking at the unemployment ratio, it can estimate the potential number of people seeking employment. Furthermore, this number has been rising in the past few years. This means that more and more people are out there every year seeking job opportunities and employment, which brings us to the issue.

> "*Hiring within Iraq is based primarily on connections and networks; the priority for filling an open position is often to gain security for a family member or friend.*" (Caldwell, 2013)

There is no practical way for people to communicate with employers and find their jobs in Kurdistan Regional Governate's (KRG) community. This has forced the community to open up social groups and accounts to ease the accessibility to those with the qualified skills with a job that matches them. Of course, this has many issues as the jobs are random, making it harder for users to keep and find a job that meets their skills. Moreover, since these platforms are not well structured and designed to meet the requirements and needs of employers and Job-seekers, now more than ever, people are struggling with finding opportunities. This has also forced many to seek a new life and start fresh in other countries by smuggling into the European countries to find better opportunities that accommodate their living expenses. The objective focuses on sustainable economic growth under UN.

> "United Nation's Sustainable Development Goals (*UN SDGs), GOAL 8: DECENT WORK AND ECONOMIC GROWTH. This will require societies to create conditions that allow people to have quality jobs.*" (UN, 2015)

Nevertheless, as much as the importance of building quality job opportunities, it is crucial to give equal opportunities to everyone for the position. To solve this, the idea is to design and develop a web-based platform where users can easily search and find jobs using several filtering options. This platform gives the users the ability also to create a standardized Curriculum Vitae (CV). This provides efficiency and accessibility and allows users to find their next opportunity in a shorter time. This means that their search timings or quality of making a CV no longer limit users, and the decision made by the company goes back to how skilled and qualified the user is.

1.2. Problem background

The current methods people use to search and find jobs have always been the subject of improvement. It begins with time it consumes plus further fueling this issue is that in a less advanced society like the people within the (KRGs) community where usually find jobs based on personal political connections or they know someone from the work area they wish to work in rather than their actual skills and qualifications, previously the damage it caused was not understood. Still, as the society grows bigger and bigger, unqualified people can be seen in different positions due to many inefficiencies in various work environments.

> "*The increase in the population in Iraq, the lack of effective programs to deal with unemployment and positive control of the number of new entrants to the labor market, the breakdown in the appointment of graduates, the decline in educational planning and the low link of educational institutions to the labor market and the inability of the private sector to absorb unemployment led to the rise of the unemployment rate."*
>
> *(unity, 2018)*

This problem in the lack of job accessibility has made the community struggle. This has affected the young generation heavenly, to a point where many people seek refuge and migration to other countries in the hope of finding new opportunities. A stable platform that allows users to search and find job openings to start their journey would seriously affect this problem. Because of the rise in the unemployment ratio, Kurdistan Regional Governate requires job accessibility more than ever, since the workplace environments need equality and equity now more than ever. Many issues with employment and job opportunities lie within connecting the right people with the right employer. While in the current system, many unqualified people aim to apply to any positions they find interesting, this gives the employers the struggle of going through all of the applications to find their right candidate. Since there is no field-specific platforms that focuses on people and employers with specific talents and qualification. While also guiding their journey if they were starting and did not meet the requirements for a particular position.

As an alternative solution, companies and employers took it to social media platforms to post and announce their job openings, which helped promote job opportunities widely but lacked specificity. Since the audience was too general, it often caused problems finding the right candidate for the correct position. While another issue was the lack of verifications wherein a social media platform like Facebook, anyone can see a post or make a post. Due to many random and inexperienced people being among the applicants, companies started to rely less on this method and take their matters into their hands, which is one of the main reasons most employments are made through personal connections nowadays. It provides trust and reliability to the employer but only ranges within the known relations of that person.

1.2.1. Problem Statement

Due to the increase of job opportunities in the technology industries, the current platforms for job seekers and providers in Kurdistan Regional Governate (KRG) is not suitable or effective in finding jobs and professions, Figure 1.1 & 1.2 shows the rise in unemployment rate since 2008(WorldBank, 2008 - 2022); therefore, the current problem with existing platforms in KRG can be classified as follow:

1- Current systems cannot meet the rapid growth in demand for skilled individuals in the technology industry.
2- Lack of a reliable and effective method to create effective communication between job seekers and job providers.

The figures from (Macrotrends) indicate the severeness of the issue and how it has kept rising over the past few years, the poor effort of the governate on this issue has led it to get worse day by day. This has made many people seek new opportunities in other countries and among those the majority are among the youth.

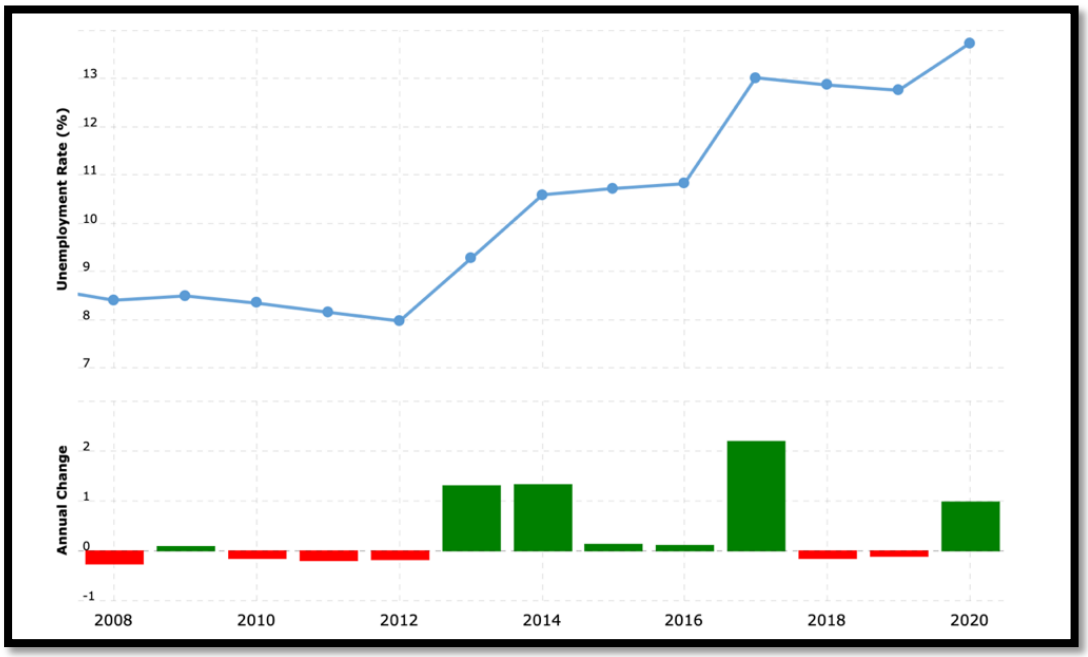

Figure 1-1 The Rise of Unemployment in IRAQ over the last years Source: Macrotrends

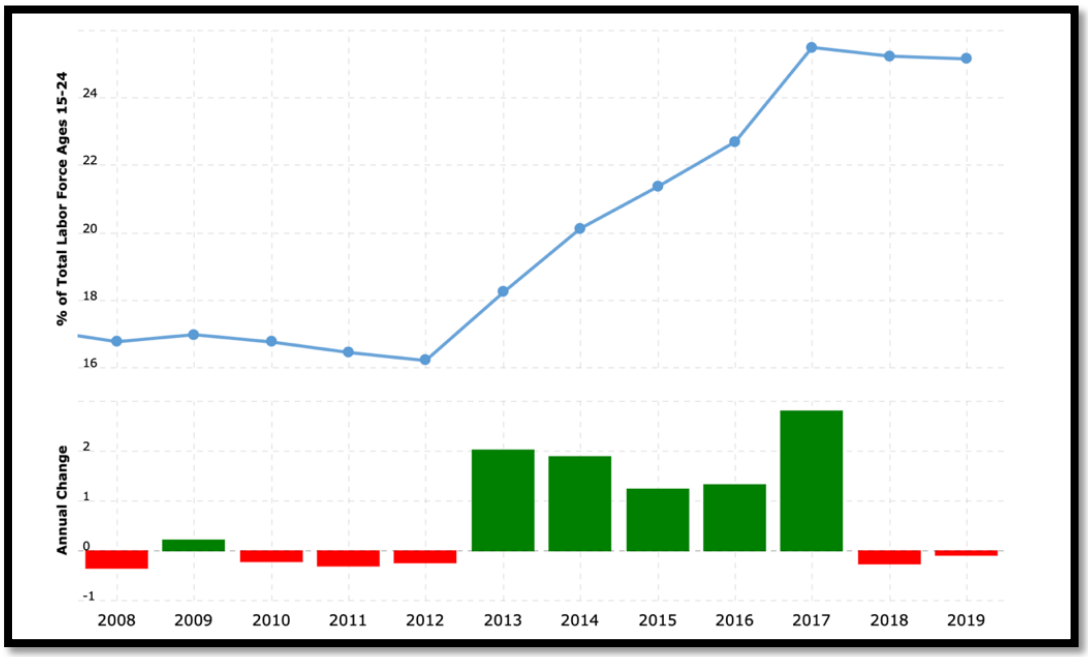

Figure 1-2 The Rise of Youth Unemployment in IRAQ over the last years Source: Macrotrends

1.3. Project aim

The development of this project aims to provide a more efficient, and accessible system to Job-Seekers and Employers, to allow better communication and reachability between them through a more straightforward process by using a dedicated employment system.

1.4. Objectives

i. To Analyze the existing and current job search and employment systems.
ii. To design and develop the proposed Web-Based Professional Employment system.
iii. To test and evaluate the developed Web-Based Professional Employment System's Performance, accessibility.

1.5. Scopes

i. The system will focus on job seekers and employers in the technological industry.
ii. The system will use React.js for Front-end and Firebase for the Backend.
iii. The system will consist of necessary functionalities for job seekers and Employers when seeking one another.
iv. The system will be developed as a website accessed through the internet.
v. The system will be developed to operate within Kurdistan Regional Governate (KRG).

1.6. Importance of the Project

People expect comfort and high-value services all the time, and finding a job is no exception. Having bad experiences and difficulties finding a job will drastically affect how people preserve getting a job and being stable.

> "*Unemployment in Iraq has reached an unprecedented level, despite its vast oil wealth and abundant agricultural land."* (unity, 2018)

This can easily cause them depression and frustration toward their future, which directly affects the growth of the community and the country's development process because this is highly dependent on the workforce and especially the youth when they are just starting to look for jobs and enter their life of self-government. Due to the difficulties the country has faced caused by the government and several other parties; the community has become hopeless in finding a job.

> " *The number of unemployed stood at 653,000, which is the number of registered, not including non-registered, which will double the numbers. There is chaos in statistics and information, and inaccuracy "*(unity, 2018)

This has led to their disappointment, and lack of experience since the only way to raise their knowledge is through hands-on experience. The objective here is to focus on communication between Job-seekers and employers. To allow Job-seekers to find work more straightforward. Hence, bringing hope back to their life so that they can stand on their own and start making dreams and shaping the country's features. They aim to provide equality and equity to people looking for job opportunities, drastically affecting youth unemployment rates.

1.7. Organization of the Report

Chapter 1 (Introduction): the chapter focused on providing an overall view of the problem, why it is essential, and what benefits it can gain by solving the issue. Then it explained the proposed solution and how it intended to solve the given problem.

Chapter 2 (Literature Review): the chapter focused on research, and findings of literature review.

Chapter 3 (Methodology): the chapter focused on describing the overall method and methodology chosen for the development of this system. While also justifying the choice of methods, techniques, and frameworks.

Chapter 4 (Requirements and Analysis Design): the chapter focused on explaining and finalizing the requirement analysis, Design, Database, and interface design.

Chapter 5 (Conclusion): concluding and finalizing remarks in the Final Year Project (FYP) report.

CHAPTER 2

LITERATURE REVIEW

2.1. Introduction

This chapter conducts a detailed literature review. The purpose is to isolate the relevant analyses to support FYP1. This chapter has conducted a literature review to analyze the existing system. This literature review is an essential section because it allows a better understanding of the execution of the project by including logical fragments of knowledge. The analysis involved conducting research, survey and observation on the existing systems. Besides, this chapter also involves finding similar job-listing web-based systems and methods users take. In addition, identifying and analyzing the current system will give a clear view of the proposed system and its purpose. The essential topics discussed are the overview, and all found systems are compared with the proposed system for analysis. Furthermore, a brief literature review is conducted on the tools and technologies used to determine their benefits to the project development and implementation, Figure 2.1 shows the content structure of Chapter 2.

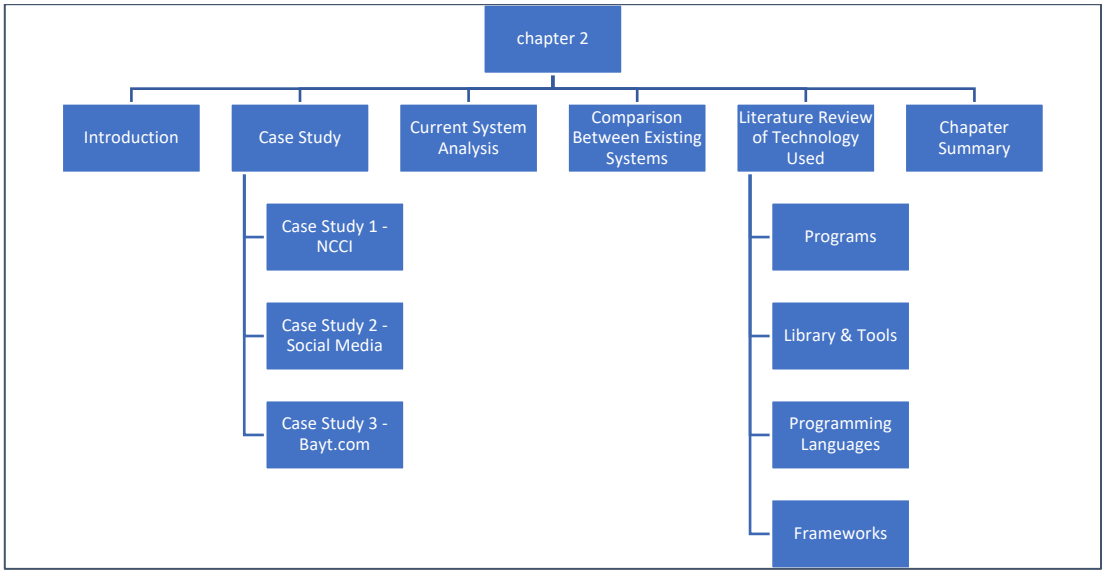

Figure 2-1 Content of Chapter 2

2.2. Case Study

This section will illustrate the case study for the existing systems that have been used within Kurdistan Regional Governate (KRG). A few systems and methods are currently being used to manifest the need for employment.

> "*LinkedIn will help you find a job faster because most hiring managers and recruiters are already using it. A whopping 87% of recruiters find LinkedIn to be the most effective when vetting candidates during the hiring process – especially those under 45 (90%).*" (MyComputerCareer, 2020)

Moreover, there are discussions and explanation included for a few systems that are known and used in (KRG). Furthermore, this case study only focuses on Kurdistan Regional Governate (KRG) and its community. While many existing systems are used in different Iraq cities, this project only focuses on Kurdistan Regional Governate (KRG). The justifications, needs, and results are only intended for (KRG) and its community.

2.2.1. Case Study 1 – NGO Coordination Committee for Iraq (NCCI)

Non-Profit Organizations (NGOs) Coordination Committee for Iraq (NCCI) system is a Web-based system developed by Non-profit Organizations (NGOs) committee members to allow members from (NGOs) that are also a registered member of (NCCI) to post and publish their job vacancies. The objective of this web-based system is to open doors to members within the community to stay informed and communicate with the (NGOs) regarding their open vacancies.

The web-based system is straightforward and lacks a good User Interface and User Experience with its simple design. It is also functionally slow and unreliable when loading different pages. Regular operational tasks involved managing the open vacancies by the (NGOs) and Job-seekers registering and applying for the available vacancies. So, the whole operations rounds around the (NGOs) 's infrastructure and management, and only

they can use it. Besides, since this web-based system is not well known, is limited to (NGOs), and lacks extensive features such as being accessible and allow employers from different backgrounds to key in their open vacancies. Thus, resulting in not being the ideal destination for job seekers. Figure 2.2 shows the main page of listed vacancies available from the (NGOs).

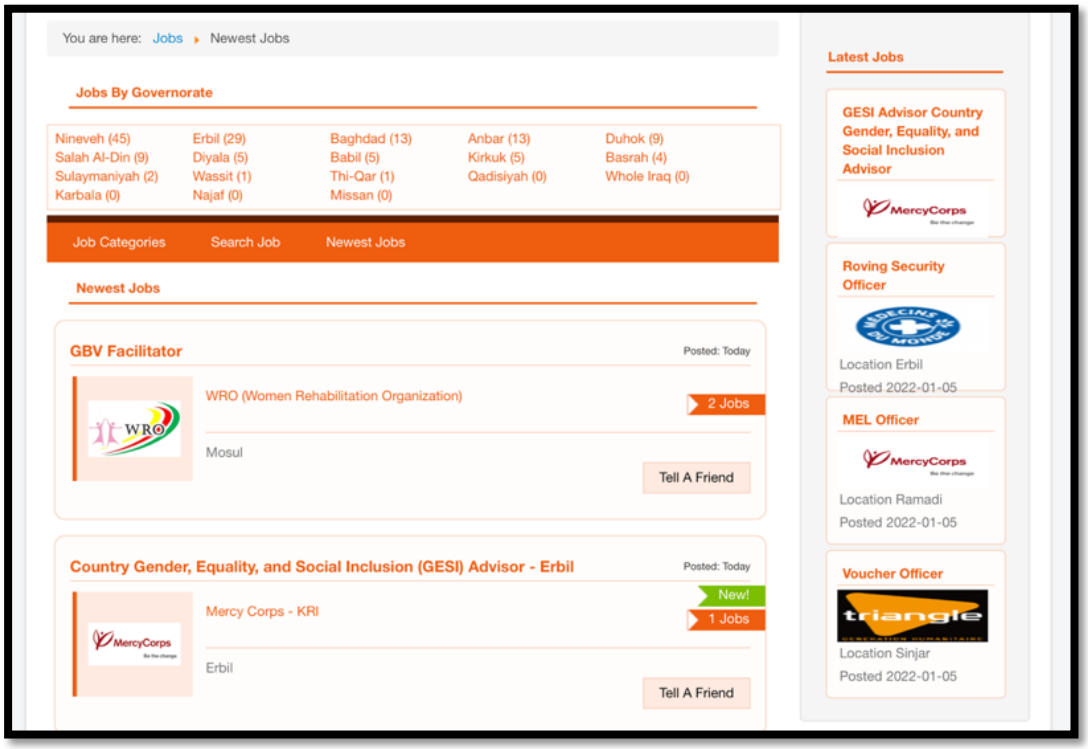

Figure 2-2 NCCI Website - Home Page

2.2.2. Case Study 2 – Social Media Advertisement

As an alternative, the (KRG) uses social media such as Facebook & Telegram groups or advertisements to find the right candidate. After Interviewing some Employers and Job-seekers who have used the platforms for employment or job search, it was concluded that, Regular operations in this method include posting jobs in job listing groups or companies or posting a position through their account and advertising to it. This method allows them to reach many people from different backgrounds, but it cannot meet the company or employer's needs. Many users on Facebook are usually just them to spend their free time watching & reading things they like with no intention to find a job.

Even though the purpose of this method is to connect employers and job seekers, it is sometimes costly and, in most cases, fails to achieve what the employer or company was aiming for mainly because it is targeting the wrong audience. Which makes finding a job for job seekers or finding the right candidate challenging and time-consuming where it was usually resulting in a failed attempt. This is all because these platforms are Social Media accounts.

These systems are not made for a specific purpose such as finding a job or a potential candidate, which is why Facebook & Telegram fails in helping users achieve this even though it is well known among their users. This is why the method is not ideal when job seekers or employers need to find an opportunity or a potential candidate Figure 2.3 shows Facebook groups and Telegram channels used for job-listing purposes the pictures were taken from Facebook and Telegram Applications.

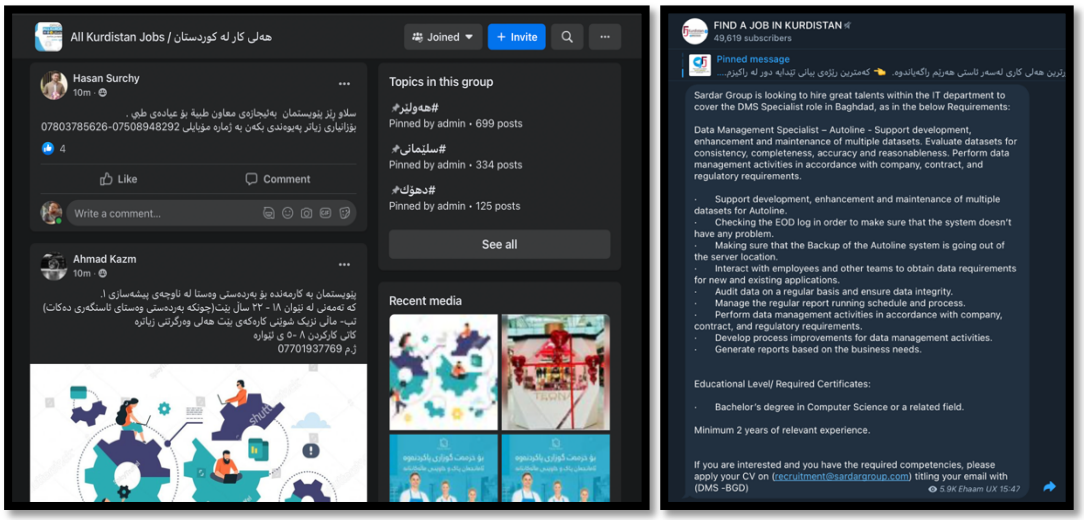

Figure 2-3 Job-Listing channel & Group on Telegram & Facebook

2.2.3. Case Study 3 – Bayt.com

Bayt.com is yet another platform used for job listing purposes made by Bayt.com. An Information Technology (IT) Solution company; this platform is available to use in most countries and regions of the Middle East. To allow companies and employers to post vacancies and job seekers applying to them. The objective of this platform is to enable cross-country applicants, so those job seekers can find and apply for jobs abroad and

outside of their region and country, which opens doors to a whole new set of opportunities. However, not everyone is willing to go to that length, and Job-seekers usually seek jobs within their country, regions, or a particular city.

The web-based system's design and structure retain simplicity and are materialistic, which gives the whole website a good User Interface and User Experience. The functionality of the website is fast and reliable. At the same time, the downside is that it is only used and well-known in a few countries of the Middle East. This makes the website inconsistent and not ideal, especially when users look for jobs in a particular place instead of looking around for new opportunities. Figure 2.4 (Bayt.com, 2022) shows the Bayt.com system.

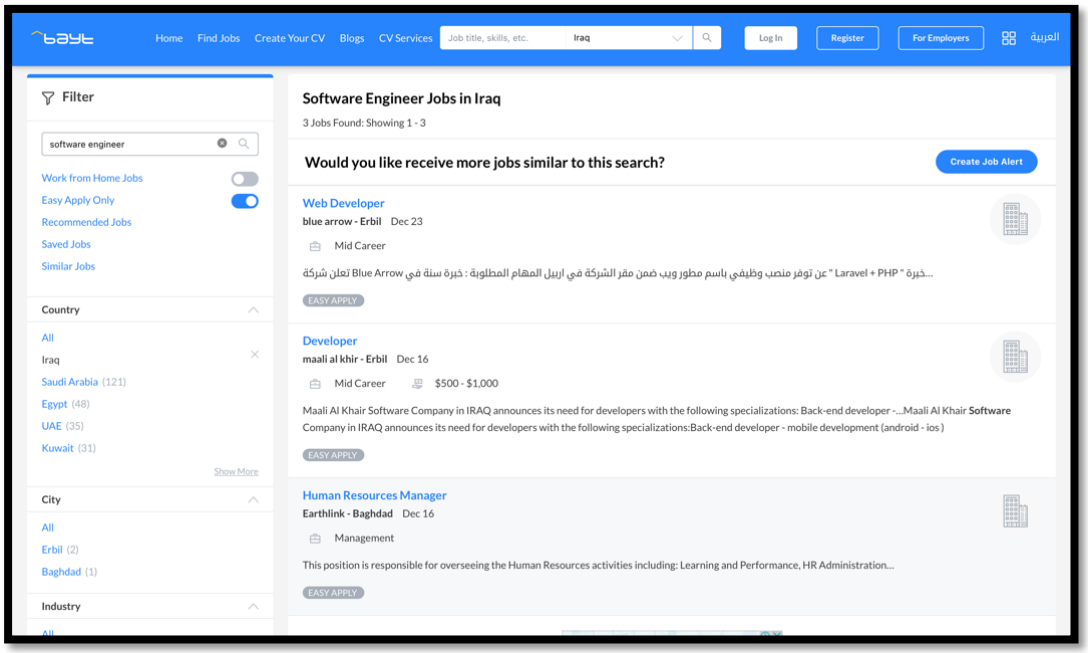

Figure 2-4 Listed jobs on Bayt.com Website

2.3. Current System Analysis

After interviewing with a few employers and job seekers analyzing and reviewing the current systems, it was found that the existing systems and methods available for Job Seekers and Employers are very inefficient. They lack accuracy and precision when

connecting job seekers and employers. This is because of one or more of the following three main reasons.

A. The system is not well known inside the (KRG) communities, and thus not all opportunities are listed there, such as bayt.com.
B. The system only allows a particular community from job seekers and employers to use its functionality where others are restricted and not allowed to access its content.
C. The system is not made for job listing and employment purposes.

Employers and job seekers would appreciate a more easy-to-use and consistent system that makes their process simpler and more efficient. Moreover, it would be much easier to rely on one approach than searching in many places to find what they are looking for, which counts for both employers and job seekers. Since the current methods are inefficient and not well known or used among job seekers, it forces both job seekers and employers to use multiple ways. For employers, this includes posting and advertising in various systems and platforms to find the right candidate while also forcing job seekers to have to search several places with the hope of finding a job that fits them. This is a very time-consuming process to maintain.

This is why employers prefer the hiring process through their connections most of the time. It saves their time, but as a result, it limits the opportunities made available for the public. After the interview, it concluded that there were many weaknesses with using current methods and systems, from lacking consistency to making it more time-consuming, giving both sides a hard time throughout the whole process.

2.4. Comparison Between Existing Systems

These current systems and methods that users can use include but are not limited to (Facebook & Telegram, Bayt.com, and NCCI – NGO Coordination Committee for Iraq). Table 2.1 Below is a detailed comparison between the existing systems, and this is

necessary because it can help developers develop great products with great functionalities. Table 2.1 below showcases the comparison between the proposed approach and other existing systems, and methods, related to job listing and job seeking. Since the main scope of the project is the proposed solution, it only needs to showcase several components and aspects to compare that lead to the proposal of this system.

Table 2.1 Existing system and Proposed system comparison

Characteristics	**Social Media**	**Bayt.com**	**NCCI**	**Jobie**
Search Engine	X	√	√	√
Categorization	X	√	X	√
Built-in CV	X	√	X	√
User Verification	√	X	X	√
Open for all user	√	√	X	√
Built-in notification	X	X	X	√
Company's profile	X	X	X	√
Position details	X	√	√	√
Well-known in (KRG)	√	X	X	X

Analyzing Table 2.1, concludes that the existing systems have several flaws and issues. The objective of the proposed method is to fix them, such as Providing a well-Designed search Engine, Categorization based on the type of job, allowing users to build their CV within the system, Validate the information of the user, open access for all kinds of users (Employers & Job-seekers), Receiving notification upon application process, providing employers profile, Detailed information about the position. By fixing these aspects, the newly developed system, it provides users with a better experience and reliability when they consider looking for jobs online.

2.5. Literature Review of Technologies Used

For the implementation of this project, MongoDB, Express, React, Node (MERN) Stack is used; the technology stack presents the solution stack, including technology infrastructure and a list of all the technology services used to build and run the Web-based system. The system is written using React and Redux technologies. It will be deployed

using Netlify, a hosting and serverless backend service for web-based systems and static websites. This deployment requires extra consideration during development of the architecture.

Table 2.2 Programs used in System Development

No	Name	Description
1	**VS Code**	Visual Studio Code is a free application by Microsoft available for all available operating systems, an online version of the application was also released by Microsoft recently to allow users to access the unique and advanced features everywhere.
2	**Node.js**	Node.js is an open-source environment that runs on multi platforms. It uses JavaScript on the server as asynchronous programming, allowing dynamic page contents that creates, open, read, write, deletes and closes files on the server. It can collect form data or add, delete, and modify the data stored in database.

Table 2.2 shows the programs that are being used in the development of the System, while Table 2.3 shows the tools, and libraries. As for the programming languages they're shown in the last table of this chapter Table 2.4.

Table 2.3 Libraries and Tools used in System Development

No	Name	Description
1	**MongoDB**	MongoDB stores data in JSON format, meaning separate fields belongs to separate documents this structure can also be changed by the user which allows for a more flexible management.
2	**Express.JS**	Express is an optimized web application framework, providing robust features for both web-based and mobile-based applications.
3	**Yarn Package Manager**	Yarn is a global package manager for repositories and source codes. Yarn is more superior to NPM in terms of performance, security, and reliable.
4	**GitHub and Git**	A global source code hosting platform, that allows for a great deal of features for version control and collaboration. While Git runs in command lines and allows tracking changes in open-source code files.
5	**Netlify**	Traditional web apps are all served with standard origin servers authenticated every request. Building and delivering the application interface every time for every user is computationally expensive, and the latency makes the app less responsive. On Netlify, Jam Stack architecture is used for a modern serverless way to deliver web applications that give the viewers a very responsive interface.

6	**EmailJS**	EmailJS is a public library used to send emails using server-side technologies, it connects to the user's email allows template creation and is easy to use.
7	**React-Icons**	Easily include popular icons in React projects with react-icons, which utilizes ES6 imports that allow you to have only the icons the project uses.
8	**TailwindCSS**	It is an open-source Cascading Style Sheets (CSS) framework used for responsive interface components. TailwindCSS was chosen over Bootstrap because Tailwind is more Utility-based than Bootstrap, which is Component-based, which means it will allow more customization and flexibility in TailwindCSS.
9	**Redux**	It is used for state management in JavaScript apps that acts dependably across client, server, and native environments.
10	**React**	React is a front-end JavaScript library maintained by Facebook, it is used to build user interfaces and UI component. React also has a library, react native for mobile development which supports both iOS & Android.
11	**VS Code**	Visual Studio Code is a free application by Microsoft available for all available operating systems, an online version of the application was also released by Microsoft recently to allow users to access it's unique and advanced features everywhere.

Table 2.4 Programming Languages

No	Name	Description
1	**JavaScript (JS)**	JavaScript is a programming language used for client-side and server-side. It allows web-pages to be interactive or dynamic in terms of functionality.
2	**JSON**	JSON is a standard file format that uses readable text to store and transmit data objects consisting of attribute-value pairs and arrays.
3	**HTML**	HTML is a formatting Language used to display content over the internet and browsers.
4	**CSS (Cascading Style Sheets)**	CSS is a simple design language used to simplify designing on web pages on the internet, it can control element colors, fonts, spaces, and size.
5	**JSX (JavaScript Syntax Extension)**	JSX is an extension of JavaScript allowing better implementation and a more dynamic use with HTML. This allows rendering logics to the Distributed Object Management (DOM) within the react, and XML elements.
6	**LucidChart**	LucidChart is a web-based design-modeling tool that has been used to create a class diagram of the system. It also has been used in creating a swim lane diagram of an existing system in Chapter 2. However, LucidChart requires a premium account to create a complex design such as the use case diagram, activity diagram, and system architecture diagram.

7	**Figma**	A prototyping tool for web and mobile applications. Figma has been used in this project to assist in designing the system interface before the development has started. It also has provided various design templates and resources.

2.6. Chapter Summary

This chapter has the existing systems to understand the characteristics and problems by comparing them with the proposed approach. As a result of the research, the issues that arise in posting jobs or searching and applying are recognized. After the study compared the existing systems, the advantages and disadvantages were identified, which were used to improve the project. As a result, the proposed approach can fix the weakness in the current system. It also covered the technology tools suitable for this project development. The relations were examined to improvise the shortcomings of this system. Besides, the MERN stack will be used to develop this project, including MongoDB, ExpressJS, React, and NodeJS. The next chapter discusses methodology of the system development.

CHAPTER 3

METHODOLOGY

3.1. Introduction

A system or Software development cycle is a set of numerous processes or phrases used in the cycle of developing software. Depending on the approach, this cycle includes certain Advantageous activities that help in the management and planning process of software development. Furthermore, this chapter discusses and elucidates several methodologies that represent a guideline to complete the system and ensure it can achieve and work well. lastly, Chapter 3 showcases several methods and then justifies the choice among the mentioned Methodologies.

An adequately defined methodology gives countless advantages to the software development cycle, such as permitting for a numerous time to adjust, identify risk beforehand, provide better valuations, deliver a protected system, and produce a well-defined understanding of the mission ahead. Choosing an appropriate methodology is critical for developers to tail since methods can progress the value of software and the complete development process. Otherwise, numerous concerns become more extensive as the progress lasts. For illustration, the built systems do not convene the anticipated needs of the users, produce unsteady systems, postponement projects, and may surpass the budget. Hence, following the methodology minimalizes these occurrences. Some methods include various agile methods, iterative and incremental development, spiral, and waterfall.

Besides, this chapter elaborates on the rationalization and methodology elected in system development. Classifying and clarifying the project phases will be defined in detail, such as how each stage affects the project development. Furthermore, describe all tools and technologies used in the project. Moreover, explain all tools and technologies used in

the project by analyzing the system requirement based on hardware and software. They conclude with the system requirement analysis at the completion of this section.

3.2. Methodology Choice and Justification

Today, many approaches can be used for system progress. There are advantages and disadvantages correlated with choosing any of the following methods to compare. The comparison helps us determine the correct methodology that is most suitable for developing the Web-based Employment System. The purpose of the comparison is to ensure selecting the right methodology because selecting the proper methodology plays a vibrant role in developing software or system.

3.2.1. Prototyping Model

Developers can use the prototyping process to create simply a solution prototype to demonstrate its functionality to consumers and make necessary improvements before developing the system. Users can be included in the development process using prototype approaches, which raises the likelihood of user agreement of the ultimate product. Several of the prototype method's advantages is the fact that it can be used to accurately model key components of a system at each stage of the typical life cycle. User participation in system development can improve communication among project stakeholders because the user is active in the process.

This is due to the user's participation in resolving ambiguous objectives and validating user needs. While this process aids developers in demonstrating the system model, it can also lead to incorrect suppositions. Clients could suppose the system is "finished" while it is not. In this phase the system is only complete in terms of interfaces, while none of the functionality is complete.

3.2.2. Methodology Comparison

	Waterfall Model	Agile	Prototyping
Advantages	• Simple and easy-going to comprehend and practice. • Stress-free to cope due to the strictness of the model. Every stage has detailed deliverables and a evaluation procedure. • Stages are handled and finalized step by step. • The Mechanism works satisfactory for slighter projects where necessities are comprehended well. • Distinctly designated steps, well- comprehended indicators, easy to assemble duties, processes, and outcomes are well documented.	• A large project can divide into manageable sprints. • Reviews are done at each sprint. • You can implement software faster, so client can acquire significance earlier. • Developers can improve their skills based on Question & Answer (Q&A) feedback • You can investigate and test concepts since their expenses are little.	• Increased user involvement in the product even before implementation. • Since a working model of the system is displayed, the users get a better understanding of the system being developed. • Reduces time and cost as the defects can be detected much earlier. • Quicker user feedback is available, leading to better solutions, and Confusing or complex functions can be identified. • Confusing or difficult functions can be identified.
Disadvantages	• Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty are high with this process model. • Poor model for long and ongoing projects. • Not a good model for complex and object-oriented projects. • High amounts of risk and uncertainty. Integration is done as a "big bang" at the end, which does not allow identifying any technological or business bottlenecks or challenges early. • No working software is produced until late during the life cycle.	• Need experienced team members. • Non-stop reviews and meetings required reliable resources. • Documentation tends to get sidetracked, which makes it harder for new members to get up to speed. • Projects can become ever-lasting because there is no clear end. • You need a long-term vision for the product and actively communicate it.	• The effort invested in building prototypes may be too much if not appropriately monitored. • Developers may try to reuse the existing prototypes to build the system, even when it is not technically feasible. • Practically, this methodology may increase the system's complexity as the scope may expand beyond original plans. • Users may get confused about the prototypes and existing systems. • Risk of preliminary requirement analysis owing to too much dependency on the prototype.

Figure 3-1 Methodology Comparison

After Researching and investigating Several System Development Methodologies, taking their advantages and Disadvantages into consideration and the overall approach. They take in developing the system as shown in Figure 3.1 As a result, the decision was to proceed with Prototyping Model to develop the Web-Based Employment System. The

prototyping model was chosen because building a prototype first, can allow to understand the system's requirements, better receive feedback from users, and allow hands-on experience with limitations to actual functionality to understand the UI/UX. Moreover, this will allow the project development to have the best possible system per feedback given from various users.

3.3. Phases of Methodology

Prototyping model is a system progress life-cycle where a prototype is developed, examined, and revised till a satisfactory prototype is accomplished. It also produces a base for the ultimate system. This model works finest when the project needs are not recognized in detail. The prototyping model is a reiterative, test and mistake method among developers and clients or targeted users. This model proposes constructing a working system prototype before developing the actual software as shown in Figure 3.2 (Martin, 2022). A prototype is a dummy implementation of a system with inadequate practical abilities, little consistency, or incompetent functioning rivalled to the current software. A prototype can be built speedily. Fast Throwaway, Evolutionary, Incremental, and Extreme prototypes are popular prototyping methods. The development of this system will tail an Evolutionary prototype, one of the executions of the Prototyping Model. The prototype developed is incrementally polished based on customer criticism until accepted. There are several stages in the prototype. The methodology stage starts with determinant objectives, developing the prototype, showing the prototype to the user for the evaluation process, improving the prototype after receiving feedback from users, and advancing to the last stage, which implements the recommended system and maintains it.

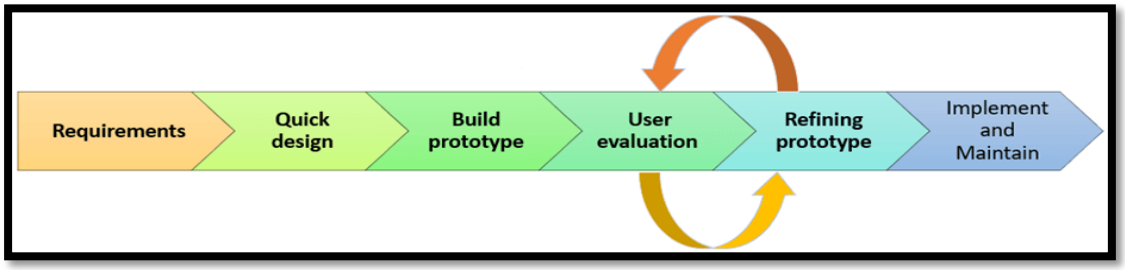

Figure 3-2 Prototyping Model

3.3.1. Phase 1: Requirement Gathering and Analysis

Requirement Gathering and Analysis is the first phase, which is the foundation of the Prototyping Model.

"*This phase is where the requirements of the system are defined in detail*"

(Matthew, 2022)

During this phase, there will be several interviews with potential users within targeted users for the Web-based Employment System. The internet will also be used to search and gather information. The goal of the interview is to know and understand their expectation from the system directly from the user. This allows for a better understanding of the user's needs. During this interview, there will be discussions of their normal operations when looking for jobs and applying to hire someone. This would allow a better understanding of the weak aspect and operations that makes the whole process hard to manage and how it can fix them through the development of the Web-Based Employment System. While the drive of surfing the internet is to get an overall indication and collect knowledge about employment systems from internet, such as common steps and functionalities expected by the public to be in an employment system.

3.3.2. Phase 2: Quick Design

The Design phase helps in developing the prototype. This design is presented to targeted users for feedback. This phase includes software Design and UI/UX design. For Software Design, brainstorming and gather ideas to convert the requirements for the Web-Based Employment system into Functionalities and decide on the best tools such as programming languages, Frameworks, Libraries, and databases to achieve the best possible results for the system. As for the UI/UX Design, it will allow discussions of the fundamentals of UI/UX and how they can be implemented in the system to result in the best possible User Experience and User Interface. This part would also include getting feedback from users to ensure the quality of UI/UX from a user's experience.

"*The second phase is Quick Design or Preliminary Design. In this stage, the simple design of the system is created. However, it is not a complete or finalized design. It gives a brief idea of the system's interface to the users."*

(Angela, 2016)

3.3.3. Phase 3: Build Prototype

In this phase, a simple prototype of the Web-Based Employment System is built with Figma based on the knowledge and requirements collected from phase One. This early version of the prototype is shown to the users for feedback and comments on the current progress in the next phase.

3.3.4. Phase 4: User Evaluation

After the simple prototype has been built using Figma Prototyping Tool, it is presented to the users for the initial evaluation process. Through this process, the project will have a reduction of the risk of failure by detecting the strength and weaknesses of the early built design.

"*Comments and Suggestions are collected from the users to ensure the developers implementation of the necessary changes and additions to the system*" (Angela, 2016)

3.3.5. Phase 5: Refining Prototype

This phase ensures that the system meets their needs. The process between Phase 4 User Evaluation and Phase 5 Refining Prototype would continue to loop until it has concluded the prototype where the users are delighted with the prototype's functionality

and UI/UX. This process can include removing and deleting features and modifications to the UI/UX. After receiving the users' agreement on the prototype, the implementation of the Web-Based Employment System will begin.

> "*If users are not satisfied with the current prototype, the prototype will be modified and changed based on the user's feedback and suggestions.*"
>
> (Angela, 2016)

3.3.6. Phase 6: Implement and Maintain

After the users agreed and were content with the refined prototype, an ultimate system is progressed based on the agreed final version of the Web-Based Employment system's prototype. The system is thoroughly examined and utilized to production. The testing process involves the users collecting their feedback after they experience the system. In addition,

> "*The system undergoes routine maintenance to minimize downtime and prevent large-scale failures*" (Matthew, 2018)

3.4. FYP 1 and 2 Gantt Chart

Gantt chart has been used as a guide to execute all tasks within a set timeline and illustrate how the project has worked. It also provides an overall look at the project timeline with the completion date for each task. By using the Gantt chart, milestones have been used for specific tasks to ensure they are done on time. Figure 3.3 showcases the Gantt chart for FYP1, which includes a detailed timeline of the progress of FYP1 from choosing the title until the submission of the final FYP1 report.

MONTH/WEEK	NOVERMBER		DECEMBER				January				FEBRUARY			
	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Prototypying Model														
Choosing PSM1 Title														
Proposal														
Chapter 1: Proposal														
Project Proposal														
Refine Proposal														
Chapter 1 Report														
Chapter 2: Literature Review														
Study on Existing System														
Study on Technologies Used														
Chapter 2 Report														
Chapter 3: Methodology														
Compare Related Methodologies														
Choosing Methodology														
Chapter 3 Report														
Chapter 4: Design														
SRS														
SDD														
STD														
Chapter 4 Report														
Chapter 5: Conclusion														
Chapter 5 Report														
Draft Full Report														
Report first Correction														
PSM1 Presentation														
Report Second Correction														
Report Submissio														

Figure 3-3 FYP1 GANTT CHART

MONTH/WEEK	April				May				Jun				July			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Requirement Gathering																
User Interview																
Internet Research																
Define Requiremnets in Detail																
Analysis The Requiremnts																
Quick Design																
Design Sketch																
WireFrame Design																
First Prototype Design																
Build Prototype																
Improved Prototype Design																
Studying the Prototype Design																
User Evaluation																
Prototype Evaulation																
Analysing User Feedback																
Refine Prototype																
Refine the Prototype Based on User's Feedback																
Conduct User Evaluation and Refine the Prototype until User Satisfaction is met																
Implement and Maintain																
Front-End Development																
Database Development																
Front-End and Database Connection																
Demo to SV and Improvise																
Conduct Tests and Improvise																
Deployment																
Documentation																
Document Progress of FYP2																
Presentation of FYP2																
FYP2 Report Correction																
Project and Report Submission																

Figure 3-4 FYP2 GANTT CHART

3.5. Technologies Used Description

A Web-Based Structure is a technology used to create the Web-Based Employment System. This system involves three users: Admin, Employers, and Jobseekers. Admin will Maintain and Develop the system while ensuring it is always functional and runs smoothly. Employers can post jobs and receive applications from Jobseekers. Job-seekers can search and find Jobs posted by Employers and Apply for Them. More information of Technologies used and their Description are given in Chapter 2.5 Literature Review.

3.6. System Development Requirement Analysis

The key objective of system constraint analysis is to tackle the hardware and software needs necessary to implement the system. Choosing appropriate hardware and software is essential to ensure the ultimate outcome meets user needs and specifications.

3.6.1. Hardware Requirements

Laptop, Desktop, Smartphones, or Tablets can be used to access the Web-Based Employment System. To ensure optimum performance while developing the Web-Based system, the following Minimum Hardware is required to support the web-based system. Table 3.2 shows the minimum hardware requirements when developing the system.

Table 3.1 Minimum Hardware Requirement for Development

NO	Hardware	Specification
1	Random Access Memory (RAM)	8GB or above
2	Operating System (OS)	macOS or Windows
3	Network Connection	Wi-Fi or 4G
4	Storage Space	100GB and above
5	Processor	I5 6Gen, Ryzen 5 4800h, M1 and above

3.6.2. Software Requirements

Software constraint is very critical in developing a proposed system. Selecting the best software requirement will lead to a plane process of development. The lowest Requirements for software are listed below. Table 3.3 shows lowest software requirements for development of the system.

Table 3.2 Minimum Software Requirement for Development

NO	Software	Specification
1	integrated development environment (IDE) - VS Code v1.6	For coding
2	Web Browser (Safari, Chrome)	For output
3	Microsoft Office	For Documentation Purposes.
4	NodeJS	Runtime environment
5	MongoDB	Database Management

3.7. Chapter Summary

Overall, Chapter 3 explained and compared the methodologies in detail since selecting the best system methodology is critical to smoothly developing the development. In addition, picking the suitable methodology will lead to the accomplishment of developing the proposed system. Beforehand picking the system methodology, several factors and evaluations need to be measured, such as the project requirements, expected end product, and project difficulty. Besides, the chosen methodology has more advantages than disadvantages. Moreover, this chapter also discussed the hardware and software needs to ensure process development is smooth and on time. Next, Chapter 4 will be concerned about the system design of the proposed system.

CHAPTER 4

REQUIREMENTS ANALYSIS AND DESIGN

4.1. Introduction

This chapter analyzes the system design of the project based on the gathered requirements; requirement analysis is the first step into identifying plans and goals with subsequent determination of information needs. It consists of describing the workflow of requirements analysis results and design phase. The first part of this chapter is requirements analysis that contains Use Case Diagram, Sequence Diagram, and Activity Diagrams followed by project design consisting of Unified Modeling Language (UML) and System Architecture Diagrams. Then it moves on to Database Design Interface Design and end the chapter with a summary.

4.2. Requirements Analysis

All descriptions of actors, use case diagrams, activity diagrams, and sequence diagrams are clearly explained in this section. Requirement analysis is an essential phase for the development project to prevent errors in the system. One of the required analyses is user requirement that allows the user to understand the system activity and process.

4.2.1. Use Case Diagram

Use case diagram summarizes some of the relationships between use cases, actors, and systems. Actors represent an entity that will trigger the system. On the other hand, use cases represent the functions, mainly functional requirements of the system, after getting

the requirements and analyzing them. The Figure 4.1 below showcases the actors and use cases of the system.

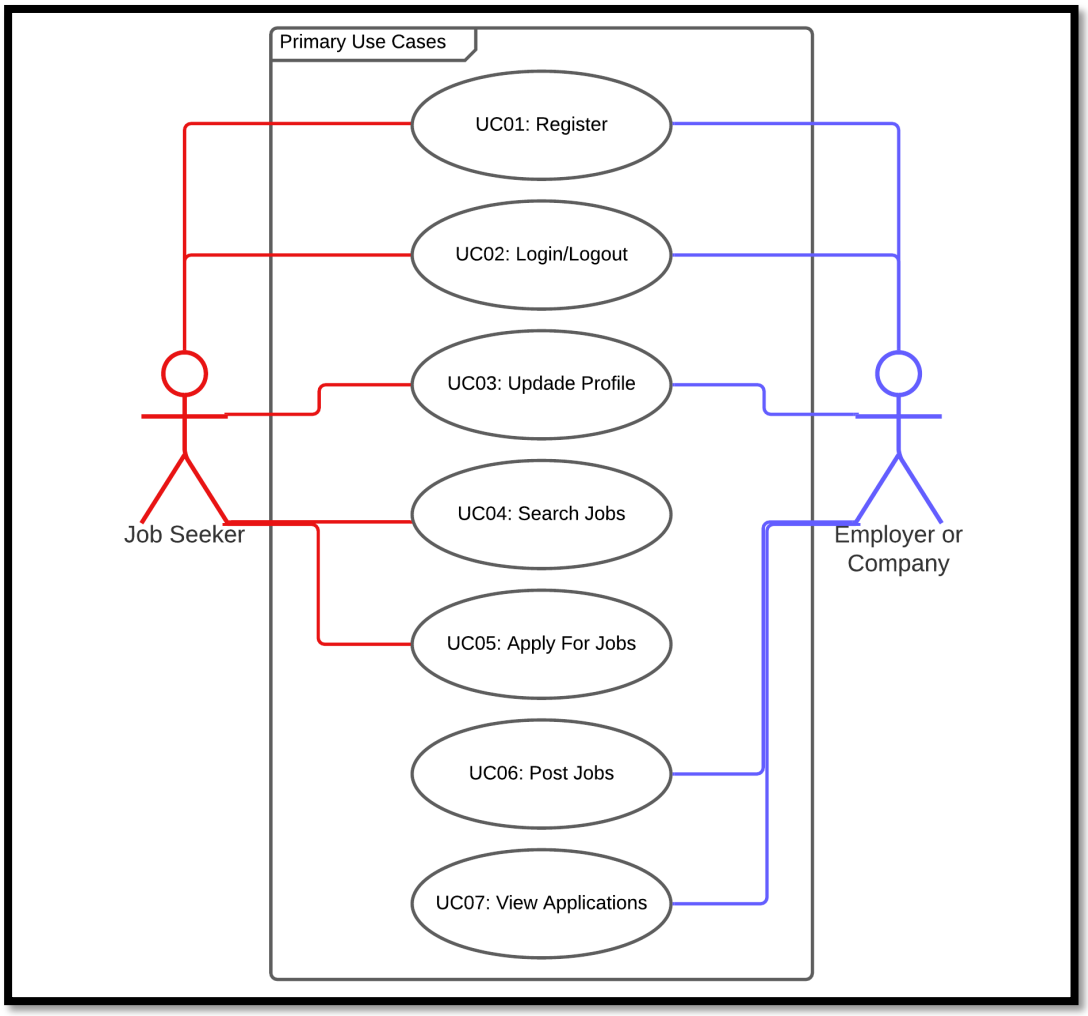

Figure 4-1 Use Case Diagram

4.2.1.1. Actor Description

Referring to the Figure above, three actors interact with the system: job-seeker, employ or company, and admin. These actors play different roles within the system. Table 4.1 Actor Description describes actors' roles within the system.

Table 4.1 Actor Description

No	Actor	Role
1	Job-seeker	Job-seekers can use the system to search for jobs and apply to them.
2	Company or Employer	Company or employer can use the system to post open vacancies and allow job-seekers to apply.

4.2.1.2. Use Case Description

The actors of the system can perform Several functions. Table 4.2 Use Case Description describes the use cases of the employment system based on the actor.

Table 4.2 Use Case Description

No	Actor	Use Case	Description
1	Employer	Register	To allow users to register and select their account type.
2		Login/Logout	To allow users to log in and log out from the system.
3		Update Profile	To allow users to set up their profile and update it later on.
4		Post Jobs	To allow employers to post their open vacancies to the system.
5		View Application	To allow employers to view the applications submitted to their open positions.
6	Job-seeker	Register	To allow users to register and select their account type.
7		Login/Logout	To allow users to log in and log out from the system.
8		Update Profile	To allow users to set up their profile and update it later on.
9		Search Jobs	To allow users to search for jobs based on their preference.
10		Apply for jobs	To allow job-seekers to apply for the jobs posted by the employers.

4.2.2. Sequence Diagram

Sequence diagrams are interaction diagrams that show how the system activities, processes, or tasks are being carried out for each function. The sequence diagram also shows in detail how each function work.

4.2.2.1. Shared Sequence Diagram

There are several functions that the two users share, which are login, register, and update profile. Figure 4.2 - 4.4 below shows the Sequence Diagram for the shared functions of job-seekers and employers. And then Figure 4.5 – 4.8 shows the sequence Diagram for the functions belonging to job-seeker and Employers separately.

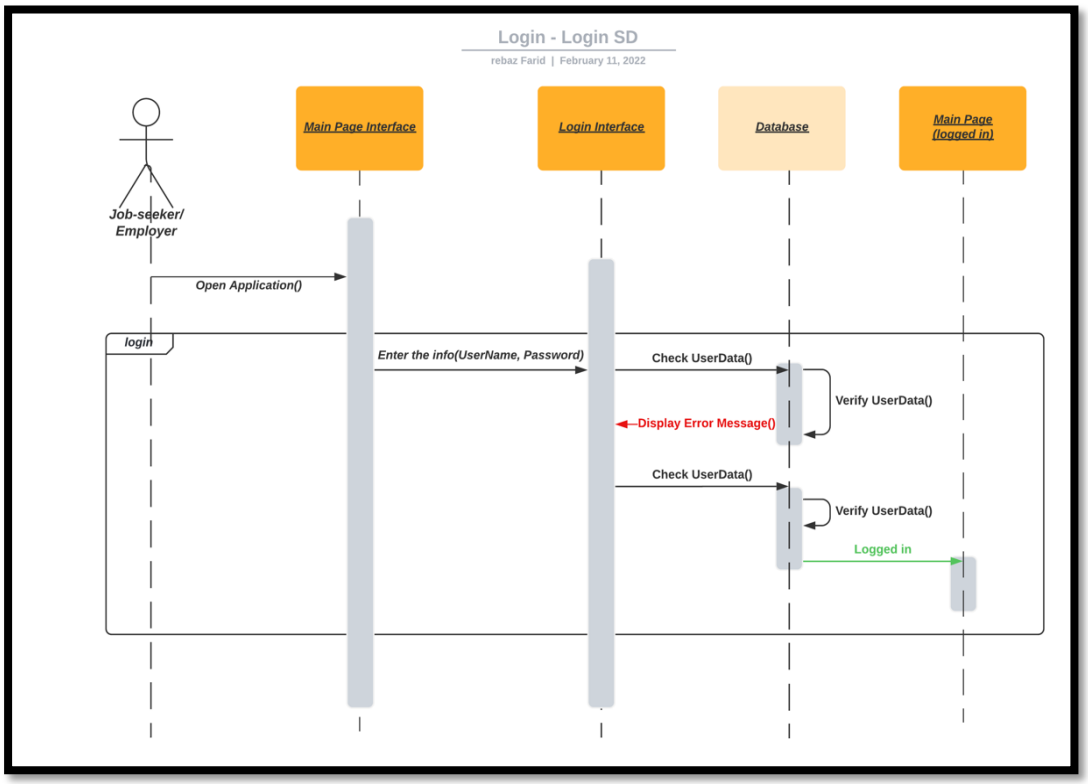

Figure 4-2 Login Sequence diagram

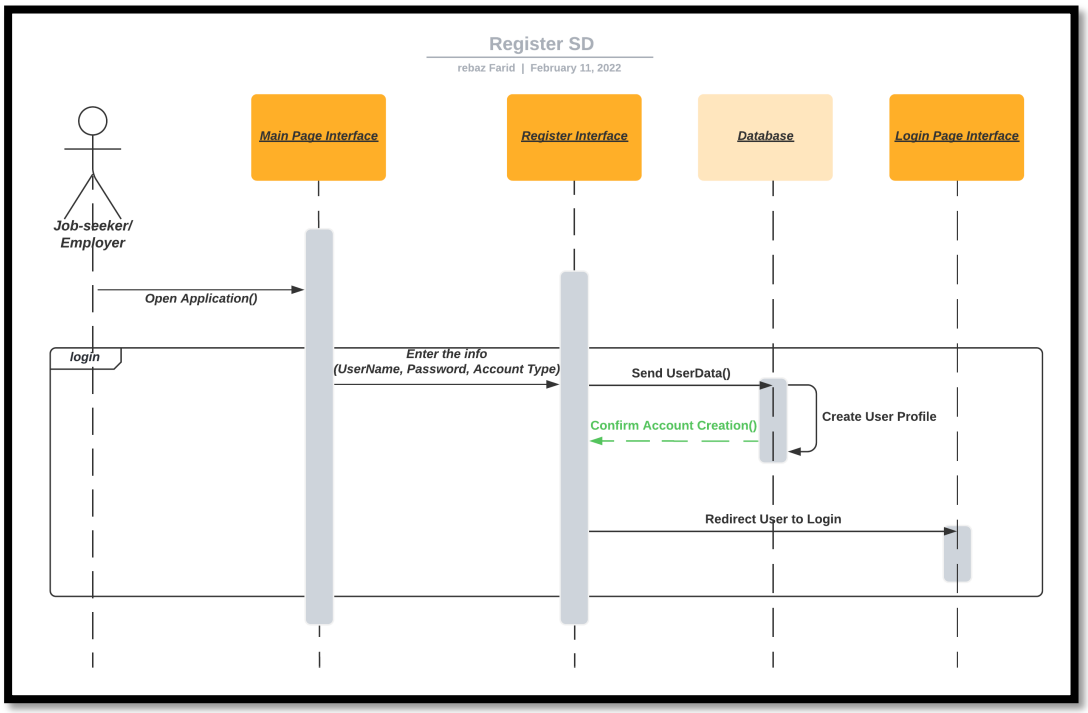

Figure 4-3 Register Sequence Diagram

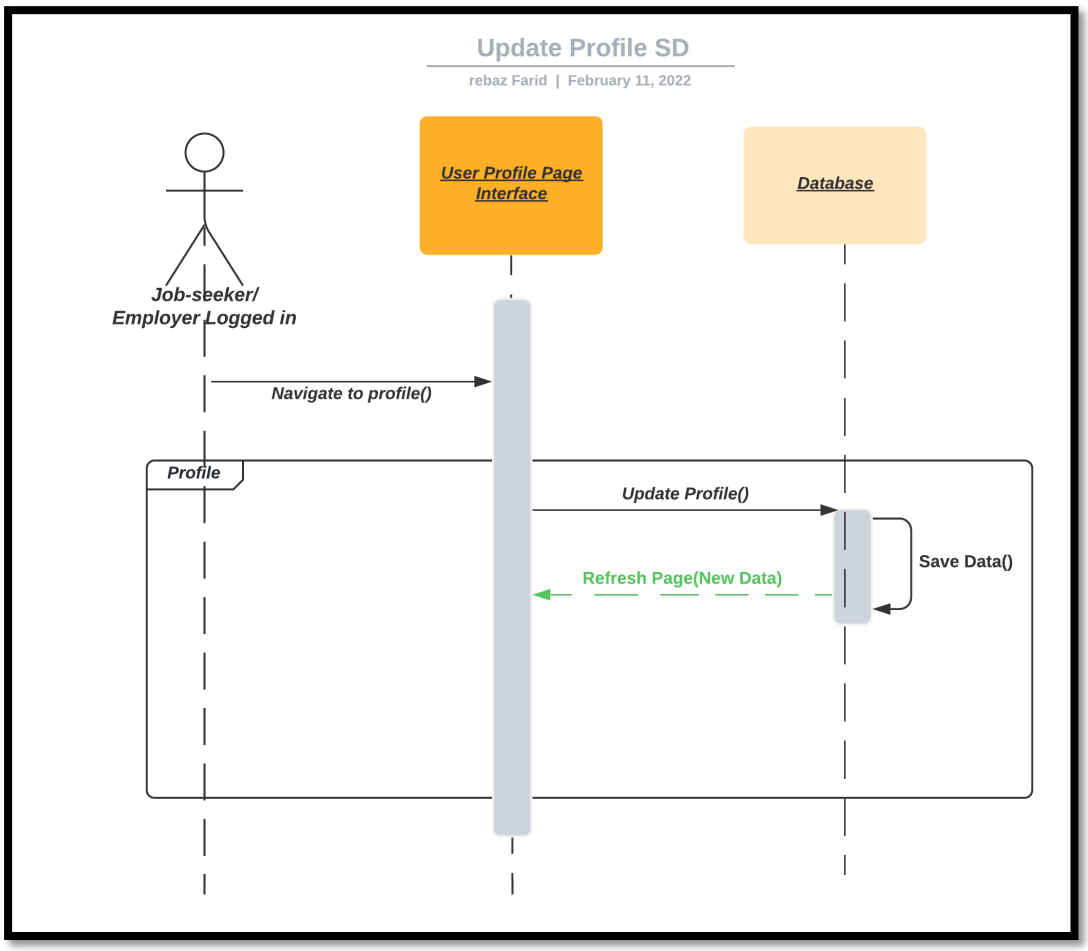

Figure 4-4 Update Profile Sequence Diagram

4.2.2.2. Sequence Diagram – Employer

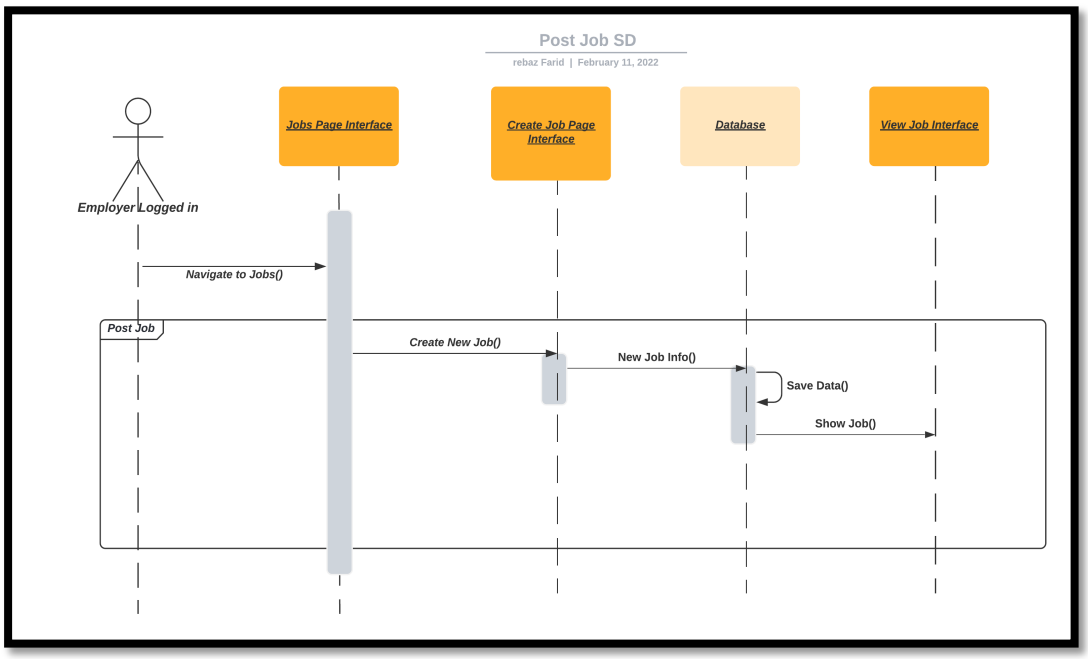

Figure 4-5 Post Job Sequence diagram

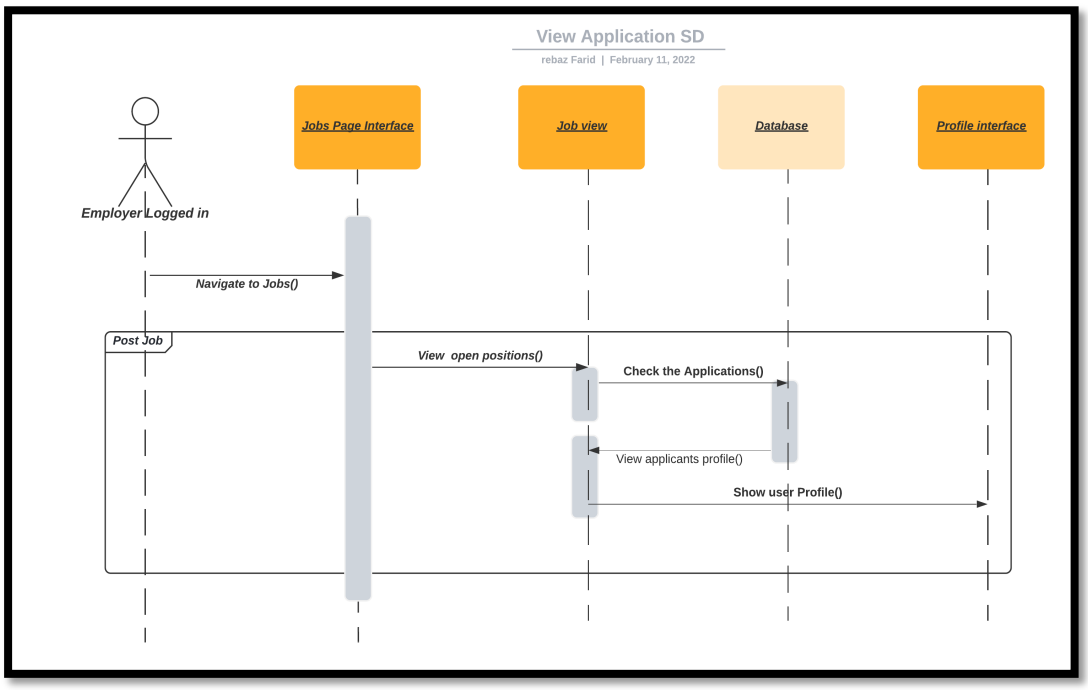

Figure 4-6 View Application Sequence diagram

4.2.2.3. Sequence Diagram – Job-Seeker

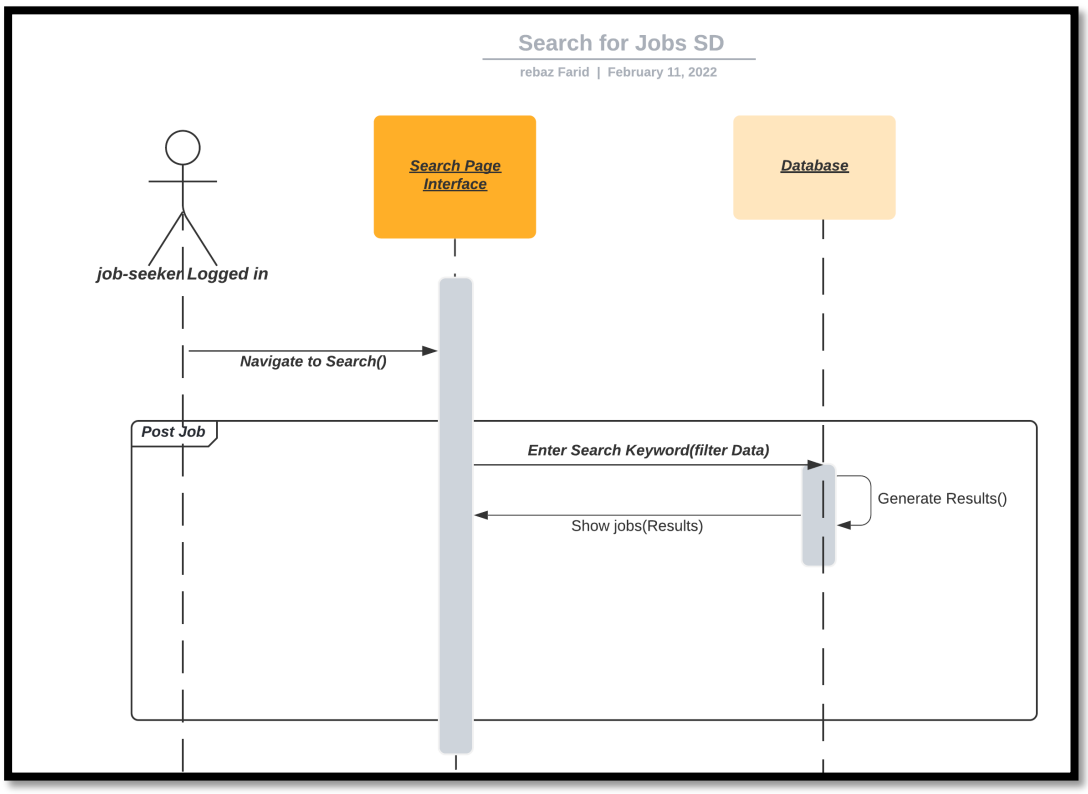

Figure 4-7 Search for Jobs Sequence diagram

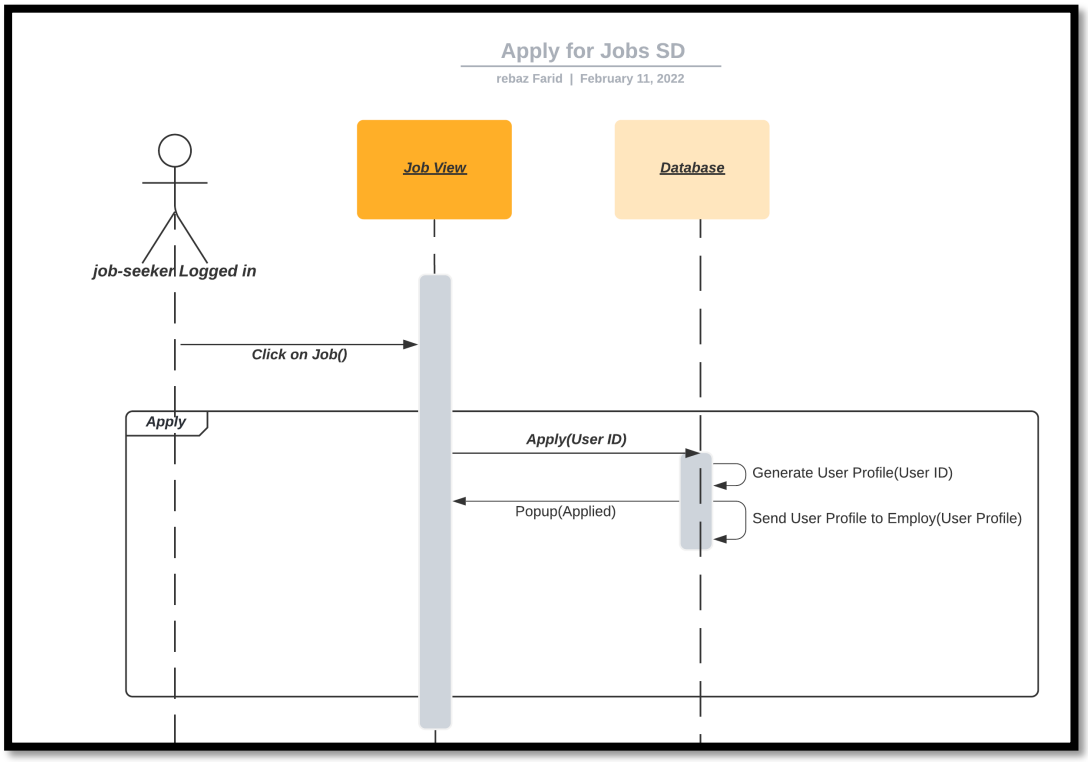

Figure 4-8 Search for Jobs Sequence diagram

4.2.3. Activity Diagram

Another critical diagram is the activity diagram that defines the system's dynamical properties. The diagram is an innovative flow chart form, which models the flow from one activity to another. In addition, this part will explain the activity diagram for each user in the web-based system. The activity diagram describes the process flow from one state to another. Each shape represents an activity of the system. Figure 4.9 shows Activity Diagram for job-seeker.

Job-Seeker:

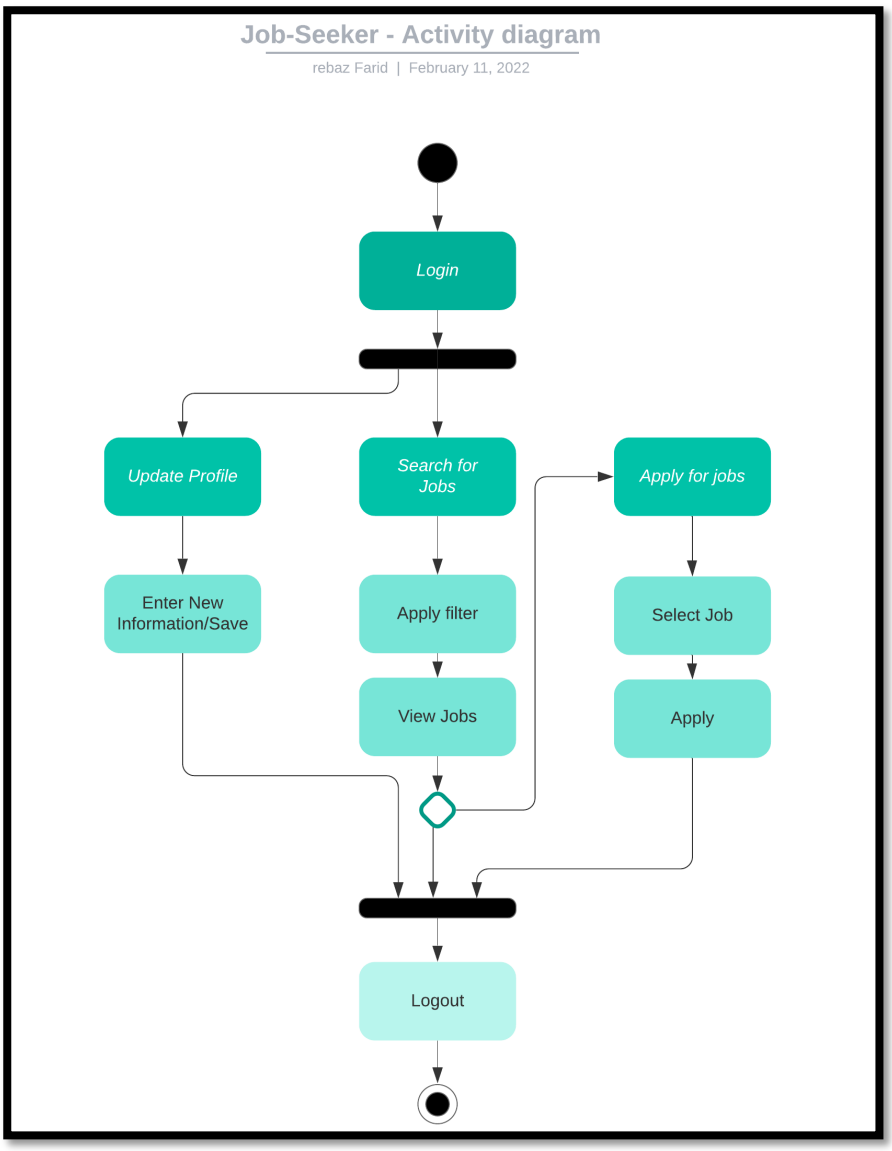

Figure 4-9 Job-Seeker Activity diagram

Employer:

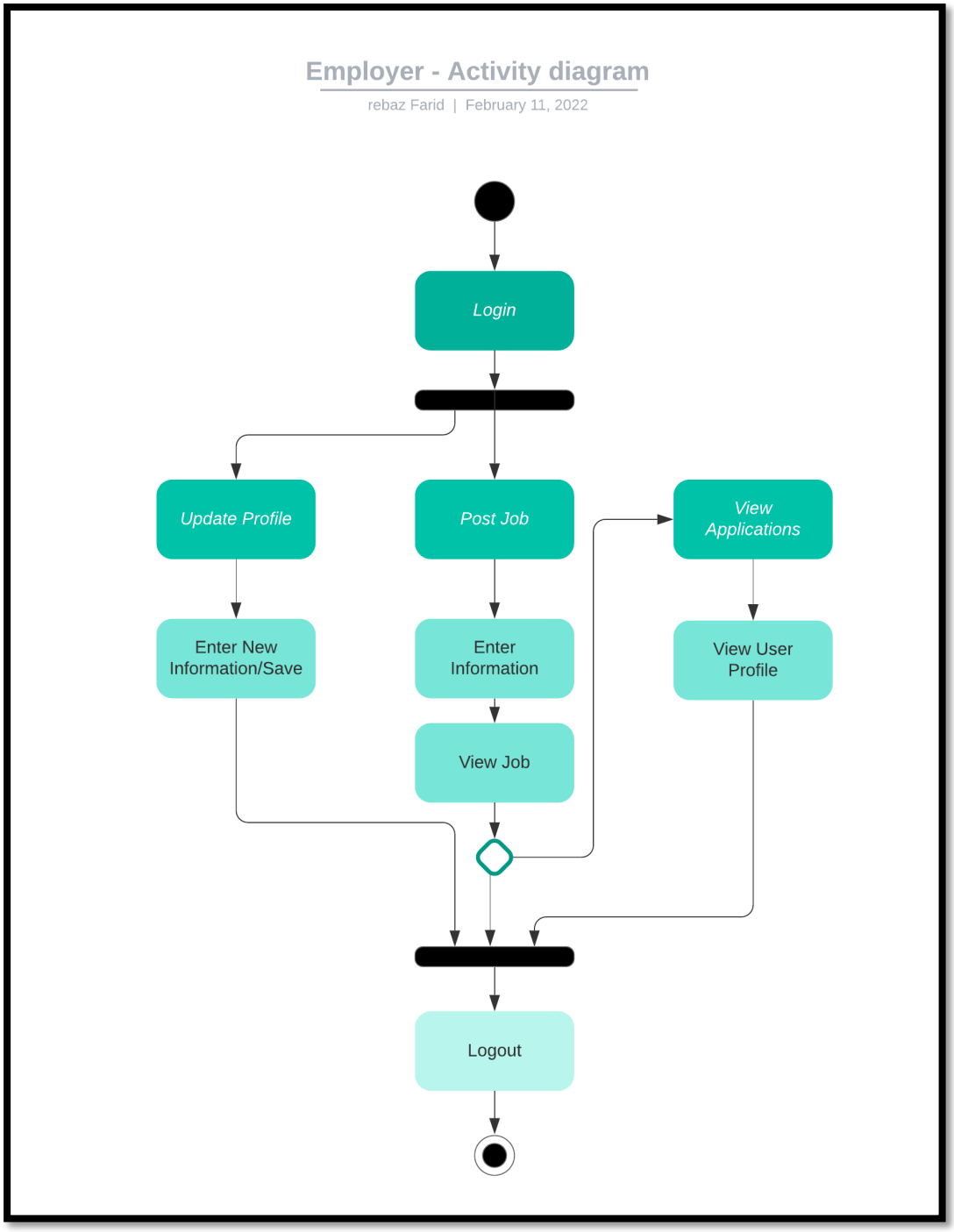

Figure 4-10 Employer Activity Diagram

As shown in Figure 4.10, the activity diagram for job-seeker and employer explains the functions and ways that the two users can interact with the system. Both users can register, log in, and update their profiles. While the job-seeker can search and apply for jobs, employers can post jobs and view applications submitted to their jobs.

4.3. Project Design

This section will focus on the UML class diagram and system architecture diagram. UML class diagram showcases how the data that belong to a different part of the system are connected and have relations to each other. However, the architecture diagram showcases how hardware and software are connected and works in real-time. Figure 4.11 shows the UML Diagram and 4.12 showcases the Architectural Design.

4.3.1. UML Class Diagram

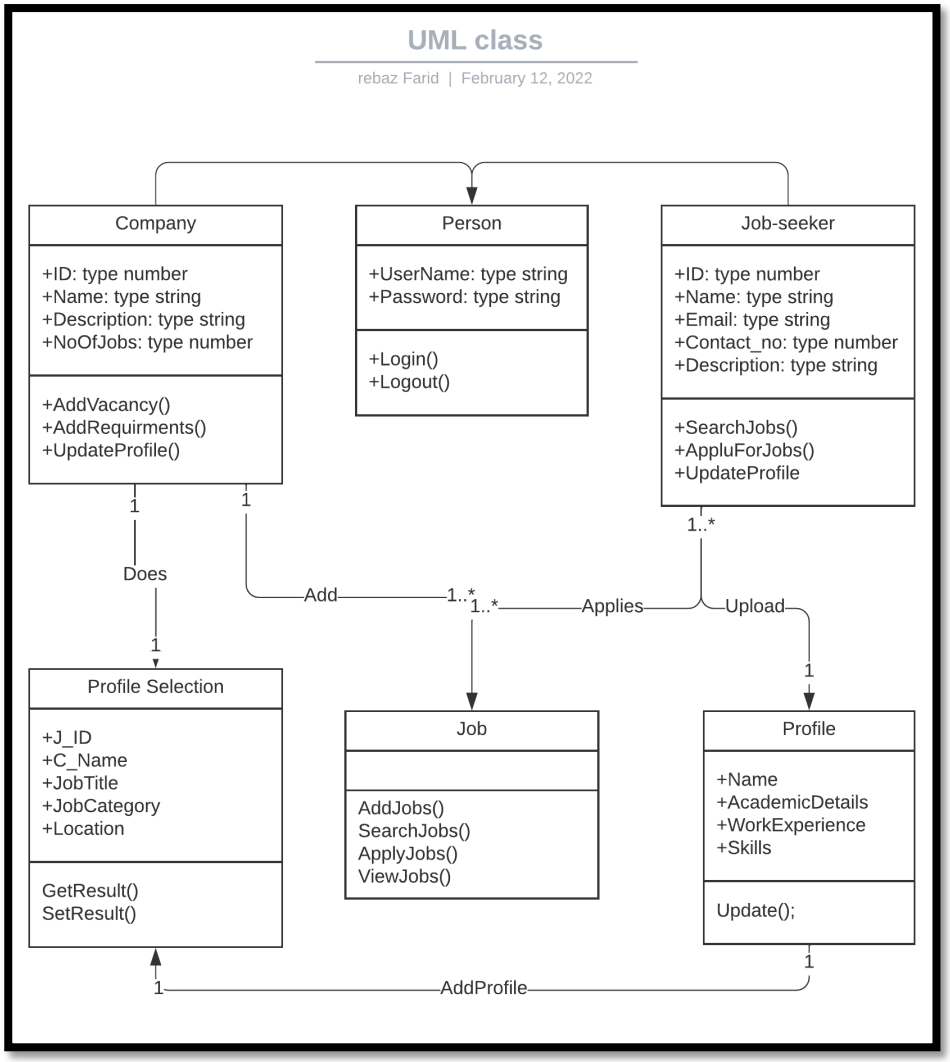

Figure 4-11 UML Class Diagram

4.3.2. System Architecture Diagram

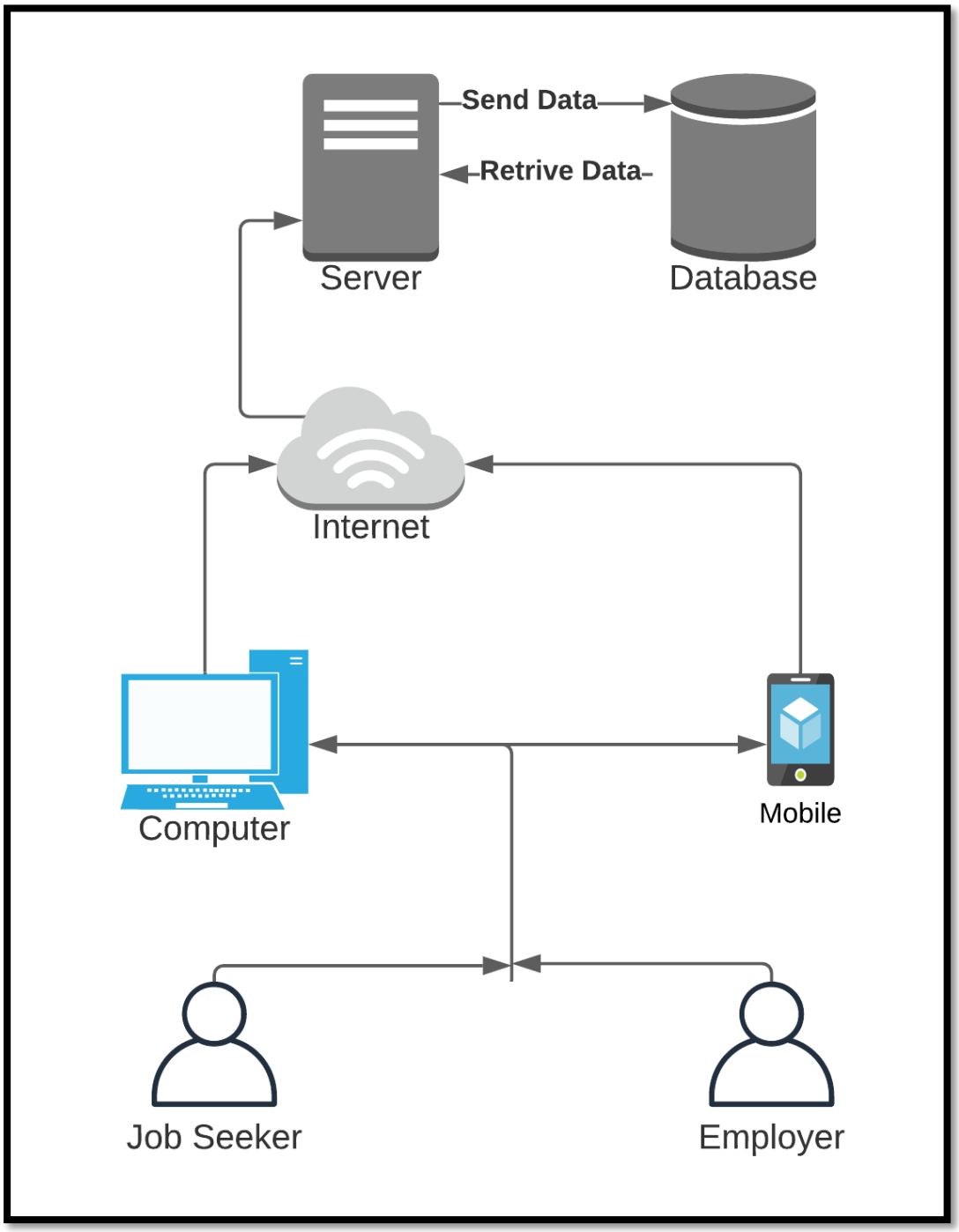

Figure 4-12 Architecture Diagram

The system architecture and design show the connection among every part of the related hardware and software used to develop the system. The diagram shows all the connected hardware and software that have a relationship. Such as how the user can use mobile or computer to access the system and how the device needs to be connected to the

internet. Only then can the device communicate with the server to load the web-based system. The server communicates with the database to retrieve and update stored data within the database. Lastly, when the user uses the system application, all the user needs information will be displayed on the web-based system.

4.4. Interface Design

Interface design is a graphic user interface (GUI) where users interact with the web-based system. The interface design must be user-friendly and straightforward to make the user easy to understand to use the system. Below is the user interface for each function in this web-based system. More information on System and interface Design is given on Appendix A System Design Documentation

4.5. Chapter Summary

As a conclusion of the chapter, the requirement analysis and design for developing the web-based system, there are five types of system designs created based on the developed system. Each system design is uniquely portraying how the developed system is working. The Unified Modelling Language (UML) is used on the requirement analysis part, a use cases diagram, sequence diagram, and activity diagram. In addition, Architecture Diagram and UML class Diagram were designed. Lastly, the interface for the web-based system was designed.

CHAPTER 5

IMPLEMENTATION, TESTING, AND DISCUSSION

5.1. Introduction

Chapter 5 highlights the implementation, testing, results and discussion of the developed system. The implementation is a process of writing code in a computer language. This is where we conclude how the system is coded, showcasing the main functions and the methods used to structure the code base for the system in accordance to the selected architecture design. Furthermore, after completion of the system implementation and development we'll conduct testing where we'll use (Black Box Testing, White Box Testing, and User Acceptance Testing). This is one of the most crucial phases of the project lifecycle to ensure that the system is fully developed based on user's Requirements and meets their expectation and satisfaction. This will ensure the maintenance of the system for a longer period of time, while also increasing the chance of a good user experience and their level of satisfaction with the developed system.

5.2. System's Core Function: User Interface

During the development cycle of this project, we've used several different programming languages among them are, HTML5, CSS, and JavaScript. Libraries include Tailwind CSS, EmailJS, ReactStrap. Below we'll discuss and elaborate on the main functions of the proposed system while also including the interface and UI/UX Design and Code Base of project. More information on system User interface is given on **Appendix A**

5.3. System's Core Function: Code Base

In this section we'll introduce some of the main functions of the system, to elaborate the system is written in JavaScript, HTML5, and CSS using React.Js which is one of the Modern technologies used today that is maintained by Facebook.

5.3.1. Contact Us

Figure 5.1 showcases the code snippet written for contact us page, this page allows the users to see contact information for the System Creators. This page also includes a section that allows users to contact the creators using a form that send the information to the creators through an email. To implement this, we've used a library called EmailJS which is easy to implement and reliable to use for small Systems. The implementation can be seen below for more details refer to the code base.

```
import { useRef } from "react";
import { sendFeedback } from "../config/emailConfig";
import { IoLocationOutline } from "react-icons/io5";
import { IoCallOutline } from "react-icons/io5";
import { IoMailOutline } from "react-icons/io5";
import { IoTimeOutline } from "react-icons/io5";
import { useRouter } from "next/router";
import en from "../locales/en";
import ar from "../locales/ar";

import pic from "../assets/img_contact.png";
export default function Home() {
  const form = useRef();
  const formref = form.current;
  const Router = useRouter();
  const { locale } = Router;
  const t = locale === "ar" ? ar : en;

  const handleSubmit = (e) => {
    sendFeedback(e, formref);
    Array.from(document.querySelectorAll("input")).forEach(
      (input) => (input.value = "")
    );
    Array.from(document.querySelectorAll("textarea")).forEach(
      (textarea) => (textarea.value = "")
    );
  };

  return (
    <div className=" bg-body text-dark px-4 lg:px-48 w-full">
      <div className="pb-10">
        <img alt="section image" src={pic.src} className="w-full" />
```

```
      </div>

      <div className="grid grid-cols-2 font-medium p-6 ">
        <div className="col-1">
          <h1 className="pb-8 lg:text-4xl">{t.contact.GetInTouch}</h1>

          <form className="form mr-8 " ref={form}>
            <div className="flex space-x-2 my-3">
              <label className="w-full">
                {t.contact.FirstName} <br />
                <input
                  name="first-name"
                  type="text"
                  placeholder={t.contact.FirstName}
                  className="border p-2 mt-3 w-full  border-dark"
                  required
                />
              </label>

              <label className="w-full">
                {t.contact.LastName} <br />
                <input
                  name="last-name"
                  type="text"
                  placeholder={t.contact.LastName}
                  className="border p-2 mt-3  w-full border-dark"
                  required
                />
              </label>
            </div>

            <label>{t.contact.Email}</label>
            <input
              type="email"
              name="email"
              placeholder={t.contact.emailHolder}
              className="border p-2 w-full my-3 border-dark"
              required
            />
            <label> {t.contact.Message} </label>
            <textarea
              name="message"
              placeholder={t.contact.messageHolder}
              className="border p-2 mt-3 w-full border-dark"
            />

            <button
              type="submit"
              value="Send"
              className=" text-xl lg:text-2xl  rounded-full  mt-6 p-2 pr-10
pl-10  text-white font-semibold  bg-accent"
              onClick={handleSubmit}
            >
              {t.contact.Send}
            </button>
          </form>
        </div>

        <div className="col-2 ml-auto w-3/4 lg:w-4/5 xl:w-2/3 ">
          <h1 className="pb-8 lg:text-4xl">{t.contact.ContactUs}</h1>

          <div className="grid grid-cols-3  mt-3 p-6 lg:p-6 xl:p-12 bg-
lightgrey ">
            <div className="col-1 grid grid-rows-4 gap-6 justify-around">
              <IoLocationOutline className="row-1 w-10 h-10 " />
              <IoCallOutline className=" row-2 w-10 h-10 " />
```

```
              <IoMailOutline className="row-3 w-10 h-10 " />
              <IoTimeOutline className="row-4 w-10 h-10 " />
            </div>

            <div className="col-2 col-span-2 grid grid-rows-4 gap-6
justify-around ">
              <h3 className="row-1 ">{t.contact.Location} </h3>
              <h3 className="row-2"> {t.contact.Phone} </h3>
              <h3 className="row-3"> {t.contact.EmailUs}</h3>
              <h3 className="row-4">{t.contact.OpenHours}</h3>
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}
```

Figure 5-1 Contact Us Code

5.3.2. Login

Login Authentication needs to be reliable and secure inside every system to prevent against attacks. To achieve reliability and secure authentication we've used google authentication that saves the trouble of authentication and secure connections from system administrators. Since google does it when user login in into their account to authenticate with our system. Details of the implementation is shown in Figure 5.2

```
import logo from "../assets/img_logo.png";
import { FcGoogle } from "react-icons/fc";
import { FaTimes } from "react-icons/fa";
import { useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { registerWithGoogle } from "../store/auth/authSlice";

export default function LoginPopup({ handleLoginModal, showLoginModal }) {
  const dispatch = useDispatch();
  const handleLoginClick = () => {
    dispatch(registerWithGoogle());
    handleLoginModal();
  };
  return (
    <div
      className={`bg-black bg-opacity-80 inset-0 fixed flex justify-center
items-center z-50 h-full w-full ${
        !showLoginModal && "hidden"
      }`}
    >
      <div className="rounded-md shadow-md w-1/4 bg-white">
        <div className="flex items-end  justify-end">
          <button
            type="button"
            className="p-4 text-xl"
```

```
            onClick={handleLoginModal}
          >
            <FaTimes />
          </button>
        </div>
        <div className="flex flex-col items-center justify-center px-6
space-y-8 pb-12">
          <div className="text-center w-32">
            <img
              src={logo.src}
              alt="Jobie Logo"
              className="object-center w-full h-full "
            />
          </div>
          <button
            className="my-15 bg-googleBtnBlue rounded-sm flex items-center
justify-between py-0.25 px-0.25"
            type="button"
            onClick={handleLoginClick}
          >
            <span className="bg-white p-1 m-1 rounded-sm ">
              <FcGoogle size={45} />
            </span>

            <span className="text-white px-2 ">Sign in with Google</span>
          </button>
        </div>
      </div>
    </div>
  );
}
```

Figure 5-2 Login Code

5.3.3. (Company & Job-Seeker) Profile Edit

The users (Company & Job-Seeker) needs to be able to showcase their profiles to one another, to achieve this user needs to be able to adjust and edit their profiles accordingly. Thus, we have implemented Profile edit to allow users to change or set up their profile accordingly. Details of this implementation is shown in the code snippet below on Figure 5.3 and 5.4.

```
import Select from "react-select";
import { useSelector, useDispatch } from "react-redux";
import { addUserProfile, fetchProfile } from
"../store/profiles/profileSlice";
import { useState, useEffect, useRef } from "react";
import { FaPlus } from "react-icons/fa";
import Education from "../components/Education";
import WorkExperience from "../components/WorkExperience";
import { cities } from "../selectData";
import { nanoid } from "@reduxjs/toolkit";
import { useRouter } from "next/router";
```

```
import useIsLoggedIn from "../config/useIsLoggedIn";
import Loading from "../components/Loading";

import en from "../locales/en";
import ar from "../locales/ar";

export default function Edit() {
  const Router = useRouter();
  const { locale } = Router;
  const t = locale === "ar" ? ar : en;

  const router = useRouter();
  const [loading, setLoading] = useState(true);
  useIsLoggedIn().then((value) => {
    setLoading(value);
  });
  const userProfile = useSelector((state) => state.profile.profile);
  const [profileData, setProfileData] = useState(userProfile);
  const [imgPreview, setImgPreview] = useState(profileData.img);
  const [cvPreview, setCvPreview] = useState();

  const imagePreviewRef = useRef();
  const dispatch = useDispatch();

  const style = {
    control: (base) => ({
      ...base,
      border: 0,
      // This line disable the blue border
      boxShadow: "none",
    }),
  };

  const handleChange = (e) => {
    setProfileData({
      ...profileData,
      [e.target.name]: e.target.value,
    });
  };

  const handleWorkAdd = (e) => {
    const workItem = {
      id: nanoid(),
      company: "",
      location: "",
      position: "",
      date: "",
      employment_type: "",
    };
    const workItems = [...profileData.workExperience, workItem];
    setProfileData({
      ...profileData,
      workExperience: workItems,
    });
  };

  const handleWorkRemove = (id) => {
    const workItems = profileData.workExperience.filter(
      (item) => item.id !== id
    );
    setProfileData({ ...profileData, workExperience: workItems });
  };

  const handleWorkChange = (e, id) => {
    const workItem = profileData.workExperience.filter(
      (item) => item.id === id
```

```
    );

    const newItem = { ...workItem[0], [e.target.name]: e.target.value };
    const newExp = profileData.workExperience.map((work) =>
      work.id === id ? newItem : work
    );

    setProfileData({ ...profileData, workExperience: newExp });
  };

  const handleEducationAdd = (e) => {
    // console.log("click");
    const eduItem = { id: nanoid(), school: "", major: "", date: "" };
    const eduItems = [...profileData.education, eduItem];

    setProfileData({
      ...profileData,
      education: eduItems,
    });
  };

  const handleEducationChange = (e, id) => {
    const eduItem = profileData.education.filter((item) => item.id === id);

    const newItem = { ...eduItem[0], [e.target.name]: e.target.value };
    const newExp = profileData.education.map((work) =>
      work.id === id ? newItem : work
    );

    setProfileData({ ...profileData, education: newExp });
  };

  const handleEducationRemove = (id) => {
    const eduItems = profileData.education.filter((item) => item.id !==
id);
    setProfileData({ ...profileData, education: eduItems });
  };

  const handleDropDownChange = (e, id, action) => {
    const workItem = profileData.workExperience.filter(
      (item) => item.id === id
    );
    const newItem = { ...workItem[0], [action]: e.value };
    const newExp = profileData.workExperience.map((work) =>
      work.id === id ? newItem : work
    );

    setProfileData({ ...profileData, workExperience: newExp });
  };

  const handleUpload = (e) => {
    e.preventDefault();
    setProfileData({
      ...profileData,
      img: e.target.files[0],
    });
    setImgPreview(URL.createObjectURL(e.target.files[0]));
  };
```

Figure 5-3 User Edit code

```
import Select from "react-select";
import { useSelector, useDispatch } from "react-redux";
import { addProfile } from "../store/profiles/profileSlice";
import { useState, useEffect } from "react";
import { useRef } from "react";
```

```
import { useRouter } from "next/router";
import useIsLoggedIn from "../config/useIsLoggedIn";
import Loading from "../components/Loading";

export default function Edit() {
  const router = useRouter();
  const [loading, setLoading] = useState(true);
  useIsLoggedIn().then((value) => {
    setLoading(value);
  });
  const category = [
    { value: "Design", label: "Design" },
    { value: "Frontend Developer", label: "Frontend Developer" },
    { value: "Backend Developer", label: "Backend Developer" },
    { value: "Full Stack Developer", label: "Full Stack Developer" },
    { value: "Web Developer", label: "Web Developer" },
    { value: "Network", label: "Network" },
    { value: "Project Manager", label: "Project Manager" },
    { value: "Data", label: "Data" },
  ];
  const cities = [
    { value: "Remote", label: "Remote" },
    { value: "Anbar", label: "Anbar" },
    { value: "Babylon", label: "Babylon" },
    { value: "Baghdad", label: "Baghdad" },
    { value: "Basrah", label: "Basrah" },
    { value: "Dahuk", label: "Dahuk" },
    { value: "Diyala", label: "Diyala" },
    { value: "Erbil", label: "Erbil" },
    { value: "Kerbala", label: "Kerbala" },
    { value: "Missan", label: "Missan" },
    { value: "Muthanna", label: "Muthanna" },
    { value: "Najaf", label: "Najaf" },
    { value: "Ninewa", label: "Ninewa" },
    { value: "Qadissiya", label: "Qadissiya" },
    { value: "Salah al-Din", label: "Salah al-Din" },
    { value: "Sulaymaniyah", label: "Sulaymaniyah" },
    { value: "Tameem", label: "Tameem" },
    { value: "Thi-Qar", label: "Thi-Qar" },
    { value: "Wassit", label: "Wassit" },
  ];
  const style = {
    control: (base) => ({
      ...base,
      border: 0,
      // This line disable the blue border
      boxShadow: "none",
    }),
  };
  const profile = useSelector((state) => state.profile.profile);
  const dispatch = useDispatch();
  const imagePreviewRef = useRef();

  const [profileData, setProfileData] = useState(profile);
  const [logoPreview, setLogoPreview] = useState(profileData.logo);

  const handleChange = (e) => {
    setProfileData({
      ...profileData,
      [e.target.name]: e.target.value,
    });
  };
```

Figure 5-4 Company Edit Code

5.3.4. Posting Jobs

Posting jobs is one of the core functions with the system, without it the system becomes utterly useless. This section consists of several mini function that overall contributes toward posting the jobs such as fetching data, and storing information with the database. The details for the implementation are shown within the code snippets below on figure 5.5 and 5.6.

```
import Select from "react-select";
import { useSelector, useDispatch } from "react-redux";
import { createJob } from "../store/jobs/jobsSlice";
import { useState } from "react";
import { useRouter } from "next/router";
import useIsLoggedIn from "../config/useIsLoggedIn";
import Loading from "../components/Loading";

export default function Createjob() {
  const [loading, setLoading] = useState(true);
  useIsLoggedIn().then((value) => {
    setLoading(value);
  });
  const cities = [
    { value: "Remote", label: "Remote" },
    { value: "Anbar", label: "Anbar" },
    { value: "Babylon", label: "Babylon" },
    { value: "Baghdad", label: "Baghdad" },
    { value: "Basrah", label: "Basrah" },
    { value: "Dahuk", label: "Dahuk" },
    { value: "Diyala", label: "Diyala" },
    { value: "Erbil", label: "Erbil" },
    { value: "Kerbala", label: "Kerbala" },
    { value: "Missan", label: "Missan" },
    { value: "Muthanna", label: "Muthanna" },
    { value: "Najaf", label: "Najaf" },
    { value: "Ninewa", label: "Ninewa" },
    { value: "Qadissiya", label: "Qadissiya" },
    { value: "Salah al-Din", label: "Salah al-Din" },
    { value: "Sulaymaniyah", label: "Sulaymaniyah" },
    { value: "Tameem", label: "Tameem" },
    { value: "Thi-Qar", label: "Thi-Qar" },
    { value: "Wassit", label: "Wassit" },
  ];

  const employment = [
    { value: "Full Time", label: "Full Time" },
    { value: "Part Time", label: "Part Time" },
    { value: "Contract", label: "Contract" },
    { value: "Internship", label: "Internship" },
  ];

  const category = [
    { value: "Design", label: "Design" },
    { value: "Frontend Developer", label: "Frontend Developer" },
    { value: "Backend Developer", label: "Backend Developer" },
    { value: "Full Stack Developer", label: "Full Stack Developer" },
    { value: "Web Developer", label: "Web Developer" },
    { value: "Network", label: "Network" },
    { value: "Project Manager", label: "Project Manager" },
```

```
    { value: "Data", label: "Data" },
  ];

  const experience = [
    { value: "1", label: "1" },
    { value: "2", label: "2" },
    { value: "3", label: "3" },
    { value: "4+", label: "4+" },
  ];

  const level = [
    { value: "Entry-Level", label: "Entry-Level" },
    { value: "Mid-Level", label: "Mid-Level" },
    { value: "Senior-Level", label: "Senior-Level" },
  ];

  const gender = [
    { value: "Male", label: "Male" },
    { value: "Female", label: "Female" },
    { value: "Both", label: "Both" },
  ];

  const style = {
    control: (base) => ({
      ...base,
      border: 0,
      // This line disable the blue border
      boxShadow: "none",
    }),
  };
```

Figure 5-5 Posting Job Code

```
const router = useRouter();

  const [formData, setFormData] = useState({
    position: "",
    address: "",
    description: "",
    responsibilities: "",
    experience: "",
    salary_from: "",
    salary_to: "",
    location: "",
    employment_type: [],
    category: "",
    work_level: [],
    gender: "",
    experience_years: "",
  });
  const dispatch = useDispatch();

  const handleFormChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleDropDownChange = (values, action) => {
    if (Array.isArray(values)) {
      const selectedData = values.map((value) => {
        return value.value;
      });
      setFormData({ ...formData, [action.name]: selectedData });
    } else {
      setFormData({ ...formData, [action.name]: values.value });
    }
  };
```

```
  const handleSave = (e) => {
    e.preventDefault();
    dispatch(createJob(formData));

    setFormData({
      position: "",
      address: "",
      description: "",
      responsibilities: "",
      experience: "",
      salary_from: "",
      salary_to: "",
      location: "",
      employment_type: "",
      category: "",
      work_level: "",
      gender: "",
      experience_years: "",
    });
  };
```

Figure 5-6 Posting Job Code

5.3.5. Finding Jobs

Another feature of the system is to allow users to find jobs based on their search and filter elements. This function is implemented within the jobs.js file and the details can be seen in Figure 5.7, for a more details code snippet refers to the code base as the code for this function is larger and cannot be showcased here. The system implements finding jobs through fetching data to and from the database and then analyzing and filtering the data that is received from the data base to be able to show the user the jobs that matched their searching elements.

```
import JobListing from "../components/JobListing";
import FilterSidebar from "../components/Filter/FilterSidebar";
import SearchButton from "../components/Home/SearchButton";
import Loading from "../components/Loading";

import { fetchJobs } from "../store/jobs/jobsSlice";
import { useDispatch, useSelector } from "react-redux";
import { fetchCompany } from "../store/tempStorage/tempStorageSlice";
import { useEffect } from "react";
import { useRouter } from "next/router";
import en from "../locales/en";
import ar from "../locales/ar";

function JobFinder() {
  const jobs = useSelector((state) => state.jobs.jobs);
  const companies = useSelector((state) => state.tempStorage.company);
  const dispatch = useDispatch();
  const Router = useRouter();
```

```
  const { locale } = Router;
  const t = locale === "ar" ? ar : en;
  useEffect(() => {
    dispatch(fetchJobs());
    dispatch(fetchCompany());
  }, [dispatch]);
  if (!jobs || !companies) return <Loading />;
  return (
    <div>
      <div className="bg-light">
        <div className="px-4 lg:px-48 w-full py-20 ">
          <div className="flex flex-col justify-center items-center">
            <h1 className="text-primary text-4xl mb-
1">{t.jobs.JobsFider}</h1>
            <h2 className="text-secondary">{t.jobs.JobsFinerDesc}</h2>
            <div className="mt-5">
              <SearchButton />
            </div>
          </div>
        </div>
      </div>

      <div className="grid grid-cols-3">
        <div className="bg-lightgrey px-5 lg:pl-48 w-full py-10 pr-8">
          <FilterSidebar
            filter={t.jobs.Filter}
            salaryRange={t.jobs.SalaryRanges}
            from={t.jobs.From}
            to={t.jobs.To}
            location={t.jobs.Location}
            TypeOfEmployment={t.jobs.TypeOfEmployment}
            cetegory={t.jobs.Category}
            Experience={t.jobs.Experience}
            WrokLevel={t.jobs.WrokLevel}
            ChooseAllThatApplies={t.jobs.ChooseAllThatApplies}
          />
        </div>
        <div className="bg-body col-span-2">
          <div className="pl-8 lg:pr-48 w-full py-10 ">
            {" "}
            <div className="flex justify-between mb-5">
              <p>
                {t.jobs.Total} {jobs.length} {t.jobs.Results}
              </p>
              <p>{t.jobs.SortBy}</p>
            </div>
            {jobs.map((jobsData, index) => {
              // console.log(jobsData);
              const company = companies.filter(
                (item) => item.id === jobsData.company_id
              );
              return (
                <div key={index}>
                  <JobListing job={jobsData} company={company[0]} />
                </div>
              );
            })}
          </div>
        </div>
      </div>
    </div>
  );
}
export default JobFinder;
```

Figure 5-7 Finding Job Code

5.3.6. Recent Jobs

This function allows the users to see the latest jobs added to the system, they can also use it to see the latest jobs posted by a specific company or employer when viewing their profile. The implementation is as shown in Figure 5.8.

```
        <div className="bg-light px-4 lg:px-48 w-full py-16">
          <h1 className="text-primary font-semibold  pb-8 text-lg lg:text-
2xl xl:text-4xl">
            {" "}
            {t.home.LatestJobs}
          </h1>
          {jobs.length !== 0 ? <HomeTable jobs={jobs} /> : <p>No jobs
found</p>}
        </div>

        <div className="bg-body px-4 lg:px-48 w-full py-20">
          <h1 className="text-primary font-semibold text-lg lg:text-2xl
xl:text-4xl">
            {" "}
            {t.home.OurCustomers}
          </h1>
          <h2 className="text-primary font-semibold py-8 text-base lg:text-
xl xl:text-2xl">
            {" "}
            {t.home.OurCustomersDescription}
          </h2>
```

Figure 5-8 Recent Job Code

5.3.7. Redux store

Redux is a JavaScript store management library that indicates how data should be accessed, managed, and modified. it supports state management through centralized store that is accessible by the whole application. There are other libraries that offer state management but none is like Redux, simple, easy to use, and implement. as shown in Figure 5.9 the redux implementation is a bit complicated but easy to use.

```
import { configureStore } from "@reduxjs/toolkit";
import { createWrapper } from "next-redux-wrapper";
import counterSlice from "./counter/counterSlice";
import { getFirestore, reduxFirestore } from "redux-firestore";
import { getFirebase, reactReduxFirebase } from "react-redux-firebase";
import app from "../config/dbConfig";

import jobsSlice from "./jobs/jobsSlice";
import profileSlice from "./profiles/profileSlice";
```

```
import notificationSlice from "./notification/notificationSlice";
import authSlice from "./auth/authSlice";
import tempStorageSlice from "./tempStorage/tempStorageSlice";

const makeStore = () =>
  configureStore({
    reducer: {
      [counterSlice.name]: counterSlice.reducer,
      [jobsSlice.name]: jobsSlice.reducer,
      [profileSlice.name]: profileSlice.reducer,
      [notificationSlice.name]: notificationSlice.reducer,
      [authSlice.name]: authSlice.reducer,
      [tempStorageSlice.name]: tempStorageSlice.reducer,
    },
    devTools: true,
    middleware: (getDefaultMiddleware) =>
      getDefaultMiddleware({
        thunk: {
          extraArgument: { getFirebase, getFirestore },
        },
        serializableCheck: false,
      }),
    enhancers: [
      reactReduxFirebase(app.firebase), // access the inner .firebase
instance
      reduxFirestore(app.firebase),
    ],
  });

export const wrapper = createWrapper(makeStore);
export const selectCounter = () => (state) =>
state?.[counterSlice.name]?.value;
```

Figure 5-9 Redux Store Code

5.3.8. Firebase & EmailJS Configuration

In our project we've used firebase to ease a lot of the implementations of features this includes real time database, database, and storage. Firebase has also made authentication easier to implement, using this modern technology was very useful in the implementation. Below in Figure 5.10 we can see the configuration made for firebase and how it's done, which allows us to access all its functionalities and features. EmailJS is another library that we used in this project. It is used to make sending emails easier; email JS is a library that allows applications to generate template and send automated emails. This was used to ensure that both users receive proper information about their updates within the system without having to check the system all the time to ensure that they're not missing anything.

```
import emailjs from "emailjs-com";

// to send user data when applying
export function sendUserData(data) {
  emailjs.init("user_sRP5iBhZmFEbxe9NtZU2b");
  emailjs.send("service_m2azucq", "template_6qnh9fc", data).then(
    (result) => {
      // alert("Application was submitted successfully");
    },
    (error) => {
      alert("Application was not submitted successfully");
    }
  );
}

// contact us page
export function sendFeedback(e, formref) {
  e.preventDefault();
  emailjs.init("user_sRP5iBhZmFEbxe9NtZU2b");
  emailjs
    .sendForm(
      "service_m2azucq",
      "template_gp268sw",
      formref
    )
    .then(
      (result) => {
        alert("Message was sent successfully");
      },
      (error) => {
        alert("Message was not sent successfully");
      }
    );
}
```

Figure 5-10 Configuration Code

5.4. Testing The System

System testing is a critical step to ensure that the system or application works well and as expected to be before we deploy or lunch the system to the public. This section focusses on testing the main features of the system and analysis how they work to ensure that everything is the way they are meant to be for users to use. This testing is a procedure of analyzing and detection of problems and errors in code, application, design, and implementation. It helps verifying software for correctness, completion, security and or quality. In terms of evaluation in quality and proper use. Testing of the system is divided into three parts which are black box testing, and white box testing.

5.4.1. Black-Box Testing

Black box testing is a testing method used for validation of a system's input and output. Its main objective is to focus on looks and, responses of the system while being used and ensure it's doing what it's supposed to do when user is entering any inputs or outputs. Black box testing is required to make sure that the functionality of the requirements is working properly with no errors, things that we're evaluating are interface, system's functions, input and output. The testing for this section is shown in detail on **Appendix C**.

5.4.2. User Acceptance Testing

To find out how satisfied users are with the proposed system, User Acceptance Testing is conducted. This method of testing is optimized to get feedback from the end-users, with the purpose of improvising the system in the future. User Acceptance Testing is necessary to ensure that the all-system's features and functionalities meet the specification and requirements gathered beforehand. To conclude the user acceptance testing, after testing all functions and features and receiving feedbacks from users, it was clear that our systems meet all the requirements expected by the users inside KRG from both (company & Employers) and Job-Seekers.

The users that we reached out to has given constructive and positive feedback regarding our system. Such as user-to-user communication, ensuring use of the platform by local companies, these users included several companies and employers that owned their own business. And job seekers that ranged from people who just quit their job and was looking for new opportunities, or fresh graduates that were just starting their career.

This acceptance testing was conducted by having potential users use our system and then requesting them to fill out our user feedback form that allowed us to understand how satisfied users are with using our system in terms of usefulness, ease of use, need to use, and actual use. The structure of the questions was based in a way to evaluate the users

on 4 core aspects that are mentioned above. Details of this test can be seen in **Appendix C** System Testing Documentation.

5.5. Chapter Summary

To conclude, chapter 5 Implementation, Testing, Results and discussion showcases the core functions of the employment system. the chapter starts by showcasing the interface of the project after actual implementation, indicating how they look on an actual website after production and deployment, to ensure that they are user-friendly and easy to use or navigate. To prevent cluttering this chapter only the interfaces of the core parts of the system is included. After this the code base is given for all the main functions inside the system, giving a brief intro to how they are implemented and work together to achieve the goal of the system. last but not least a brief on testing has been conducted that covers black box testing and user acceptance testing with more information given on **Appendix C.** the testing was done by reaching out to potential users to test and use the system and then receiving anonymous feedback to prevent biased information. This chapter concluded a critical step of development and deployment of the system that ensures success on lunch by providing how impactful it can be on potential users, while also making sure the system is error free and runs smoothly on any given browser.

CHAPTER 6

CONCLUSION

6.1. Introduction

People expect comfort and high-value services all the time, and finding a job is no exception. Having bad experiences and difficulties finding a job will drastically affect how people preserve getting a job and being stable. This can easily cause them depression and frustration toward their future, which directly affects the growth of the community and the country's development process because this is highly dependent on the workforce and especially the youth when they are just starting to look for jobs and enter their life of self-government.

Due to the difficulties the country has faced caused by the government and several other parties; the community has become hopeless in finding a job. This has led to their disappointment, and lack of experience since the only way to raise their knowledge is through hands-on experience. The objective here is communication between Job-seekers and employers. To allow Job-seekers to find work more straightforward. Hence, bringing hope back to their life so that they can stand on their own and start making dreams and shaping the country's features. They aim to provide equality and equity to people looking for job opportunities, drastically affecting youth unemployment rates.

Chapter one, redefined the idea behind the title explained its importance and why it's important to develop this system. What problems can the proposed system solve and why they are important.

Chapter two, analyzed several existing systems similar to the proposed system to understand the system's needs better and determine the issues that it can solve within the current systems that prevent them from solving the problems it tends to solve with the system.

Chapter three, compared several methodologies to understand them better and choose the best one for the development of the system. It also planned out and decided on the technologies and methods to develop in the proposed system.

Chapter four, dived into the requirements of the system a bit further and determined all the functionalities that needs to be within the website based on the user requirement analysis in chapter 2. It also indicated the data flow within the system and how they are connected. Last but not least, designing the system's interface.

Chapter five, span around the actual implementation during FYP 2, it describes, and showcases the interface of the project, code base of the core function within the system. and lastly it dives into testing of the system after completion where we test the system using black box testing and User Acceptance Testing.

6.2. Achievements

After conducting the literature review and analyzing the results, it gave a clear idea of the needs of the proposed system. This helped in better diagnose and determine the functionalities that need to be available within the system while also choosing the user flow and how they interact with the system, the data generated through the system, how they are stored, and how they are connected. Thanks to the analysis, now there is a good grasp of the system and how it should be implemented.

One of the goals for developing this system was (To analyze and review the existing and current job search and employment systems). Through FYP1, the exploration of the existing systems, taking feedback from users, and understood their flows. This helped in better understand what is lacking in the community and what it needs to do to fix these issues with the current system.

Another one of the goals for developing this system was (To design and develop the proposed web-based job listing system). While this goal is not fully achieved, it gave

an understanding of how the system should work and interact with the users and within itself. Then it also designed the interface, which gave a head start and an idea of what the system to look like.

Throughout FYP 2, the system was developed, tested and deployed. In conclusion it was clear that the system was a good alternative to the other options available within Kurdistan Regional Governate. The developed system gives the necessary functions and features to overcome the issue that arise with the use of the existing systems. Although the system requires intensive advertisement and marketing strategy to ensure that it is used by the companies within KRG, it still did not fail to accomplish its objective of solving the problems that has occurred with the user of existing system through FYP 1 and FYP 2. Below is the list of the objectives that has been accomplished.

I. Studying existing systems has been accomplished, while also listing the functionalities from other system for comparison.
II. The system has accomplished the main objectives by collecting the user requirements through interviews, calls, and surveys with potential users.
III. The functionalities proposed after analyzing user needs has been implemented and functions as expected.
IV. Several testing methods has been used such as black box testing and user acceptance testing to ensure the completion and error free of the system before deployment.
V. The system is deployed to allow users to test and see the potential and capability of the developed system in real time.

6.3. Suggested Plans for Improvement in the future

While the implementation of the project was achieved successfully, a few improvements can be done to raise the success of this project which has not been considered due to time limitation. For instance, by providing a communication feature that allows users to communicate with each other, this can ensure better communication in-between Job-Seekers and employers while looking for one another. The system can also include a recognition system that can automatically fill user's profile from their previously

made CVs or accounts such as LinkedIn. Another feature that can be implemented is an admin panel that can manage the system and profiles, this allows for better management. An AI algorithm can also be included to prevent or delete fake accounts, or inactive accounts to ensure that the users on the system are people that are actively hiring or looking for jobs.

REFERENCES

Angela. (2016). *Prototyping Methodology*. lamintang. Retrieved 20th February from http://lamintang.org/Journals

Aziz, H. (2014). *Factors Affecting Unemployment in Iraq* [Dissertation University of Southampton]. Academia. https://www.academia.edu/19600530/Factors_Affecting_Unemployment_in_Iraq

Caldwell, C. (2013, April 19, 2013). *Key Talent Considerations for Working in Iraq*. shrm Org. https://www.shrm.org/resourcesandtools/hr-topics/global-hr/pages/talent-management-iraq.aspx

Department, E. R. (2021, 31/08/2021). *The unemployment problem in Iraq after 2003, realities, causes, implications, and public policy options*. bayancenter. Retrieved 20th February from https://www.bayancenter.org/en/2021/08/2669/

ilo. (2020, Jan 2020). *Promoting decent work in Iraq*. ilo. https://www.ilo.org/beirut/countries/iraq/WCMS_433682/lang--en/index.htm

Matthew. (2018). *Software Development Life-Cycle* Colardo Technical University].

Matthew, M. (2022). *Prototyping Model in Software Engineering: Methodology, Process, Approach.* guru99. Retrieved 13thth February from https://www.guru99.com/software-engineering-prototyping-model.html

MyComputerCareer. (2020). *4 Reasons Why to Use LinkedIn for Your Professional Success*. MyComputerCareer. Retrieved 23rd February from https://www.mycomputercareer.edu/news/4-reasons-linkedin-important-professional-success/

Rodriguez, M. (2017, MAY 31, 2017). *THE ONGOING IMPACTS OF UNEMPLOYMENT IN IRAQ*. borgenproject. https://borgenproject.org/unemployment-iraq/

UN. (2015). *United Nations Sustainable Development Goals*. UN. Retrieved February from https://sdgs.un.org/goals

unity, E. (2018). *Eliminating unemployment in Iraq ... Eliminating terrorism*. Rawabet Center for Research and Strategic Studies. Retrieved 17th February from https://rawabetcenter.com/en/?p=5142

WorldBank. (2008 - 2022). MacroTrends. Retrieved 20th February from https://www.macrotrends.net/countries/IRQ/iraq/youth-unemployment-rate

APPENDIX A
SOFTWARE DESIGN DOCUMENTATION

SCSR3323: Software Design and Architecture

Software Design Document

WEB-BASED PROFESSIONAL
EMPLOYMENT-ORIENTED SYSTEM

Version 1.0

25/6/2022

Computing Department
Computer Networks and Security

Prepared by: Rebaz Farid Noori

Revision Page

a. Overview

Version 1.0 of SDD includes a brief description of the Architecture Design for the developed web-based application and database descriptions in detail.

b. Target Audience

The developed web-based application has two actors that are as follows.

- Job-Seekers
- Employers

c. Version Control History

Version	**Primary Author(s)**	**Description of Version**	**Date Completed**
1.0	Rebaz Farid Noori	Final Version	25/6/2022

Table of Contents

1. Introduction

1.1 Purpose

This SDD describes or showcases a detailed view on how the WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM is developed with a brief on the flow of the system. it also includes Design description such as architecture, database management, and interface design is also highlighted in this document.

1.2 Scope

The Scope of the software product is as below

vi. The system will focus on job seekers and employers in the technological industry.
vii. The system will use React.js for Front-end and Firebase for the Backend.
viii. The system will consist of necessary functionalities for job seekers and Employers when seeking one another.
ix. The system will be developed as a website accessed through the internet.
x. The system will be developed to operate within Kurdistan Regional Governate (KRG)

1.3 Overview

This SDD Document elaborates on the overview of the system in several sections that is as below

1. Introduction
2. System Overview
3. System Architecture
4. Database Design
5. Interface Design

2. System Overview

WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM, is a web-based application aiming to provide a platform for employers and job-seekers to connect and find one another. This helps reducing the waste of time and effort by providing an effective platform for users to connect. Moreover, the proposed web-based application allows job- seekers to create their accounts upload CVs and generate standardized profiles and search among jobs posted by employers while employers can also create profiles and post job positions that can be accessed and seen by the users.

3. Data Design

This section will focus on the UML class diagram and system architecture diagram. UML class diagram showcases how the data that belong to a different part of the system are connected and have relations to each other.

3.1 Entity Relation Diagram

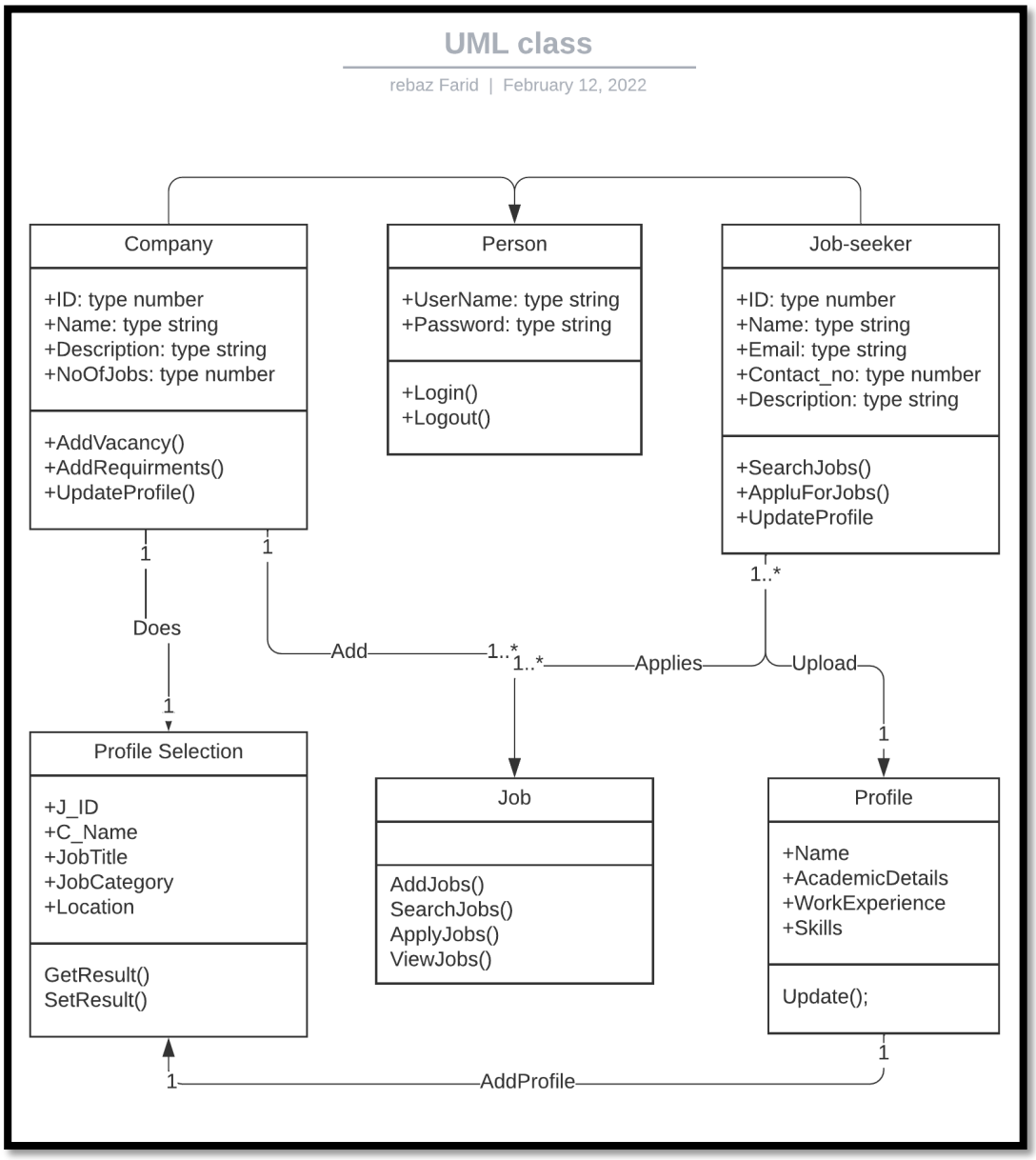

Figure 11 ERD for the Developed System

3.2 Data management in Firebase

In our project we've used firebase to ease a lot of the implementations of features this includes real time database, database, and storage. Firebase has also made authentication easier to implement, using this modern technology was very useful in

the implementation. A simple configuration once allows us to access all its functionalities and features. It is a great tool to use for projects with intensive time limitations.

4. User Interface Design

4.1 Overview of User Interface

Interface design is a graphic user interface (GUI) where users interact with the web-based system. The interface design must be user-friendly and straightforward to make the user easy to understand to use the system. Below is the user interface for each function in this web-based system. During the development cycle of this project, we've used several different programming languages among them are, HTML5, CSS, and JavaScript. Libraries include TailwindCSS, EmailJS, ReactStrap. Below we'll discuss and elaborate on the main functions of the proposed system while also including the interface and UI/UX Design and Code Base of project.

4.2 Screen Images

in this section we'll showcase the interface of the developed system.

4.2.1 Landing Page UI

Landing page is the root of every website it's how every person who visits your website gets their first impressions, it's very important to ensure a good and eye-

catching design to make sure that the user doesn't get uncomfortable or can understand what your website is about. It's necessary to provide guidance and supplementary information in order to allow the user to navigate to their needs. As seen below in Figure 2 and 3 The landing page design has a good design and color set and gives good first impressions to users at first glance. The landing page also includes several feedback section and customer review as well as an overview of the latest jobs. To assure users if they can find a job in the category that they're looking for a section with the sub-categories is provided within the system that shows the users what kinds of jobs the platform offers.

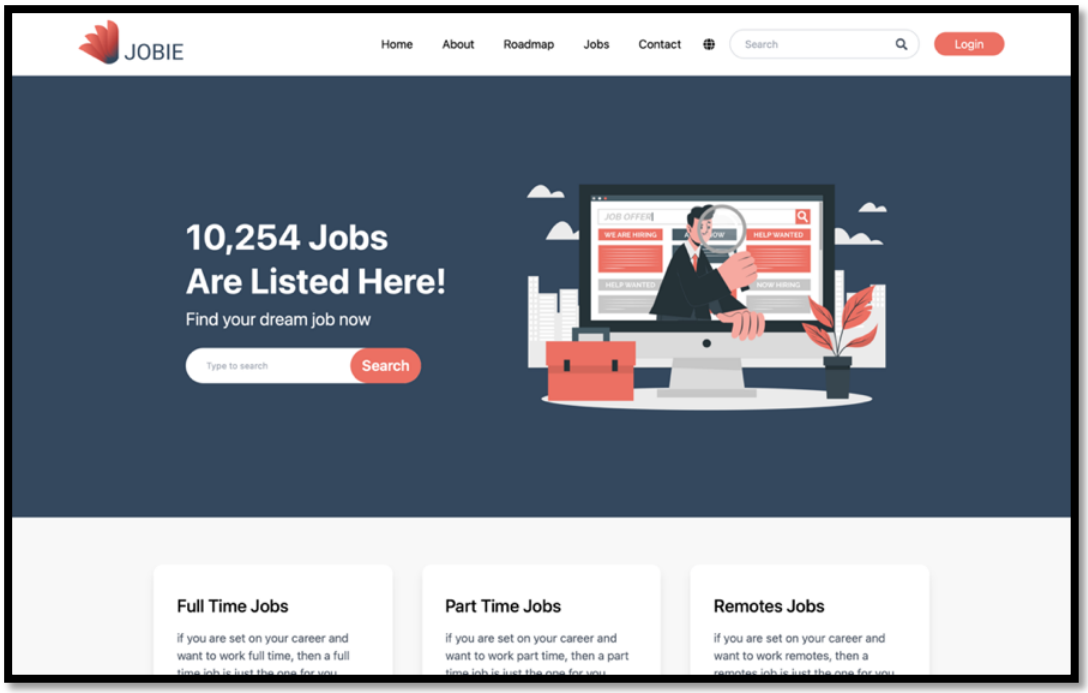

Figure 12 Landing Page UI 1

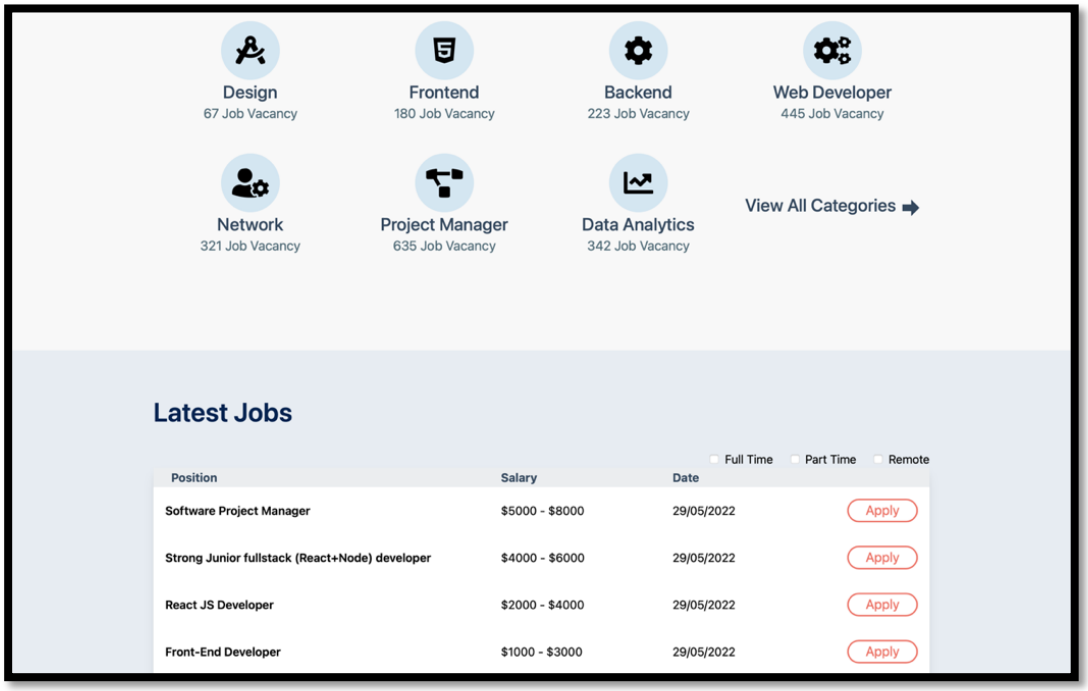

Figure 13 Landing Page UI 2

4.2.2 Contact Us UI

Feedback is necessary in the implementation of every software it's a critical way of receiving information about things that needs to be improved, features that can be added, or bugs that needs to be dealt with. For this a simple and interactive Contact, Us Form has been provided to the users while also including contact information such as phone number and email, as shown in Figure 4.

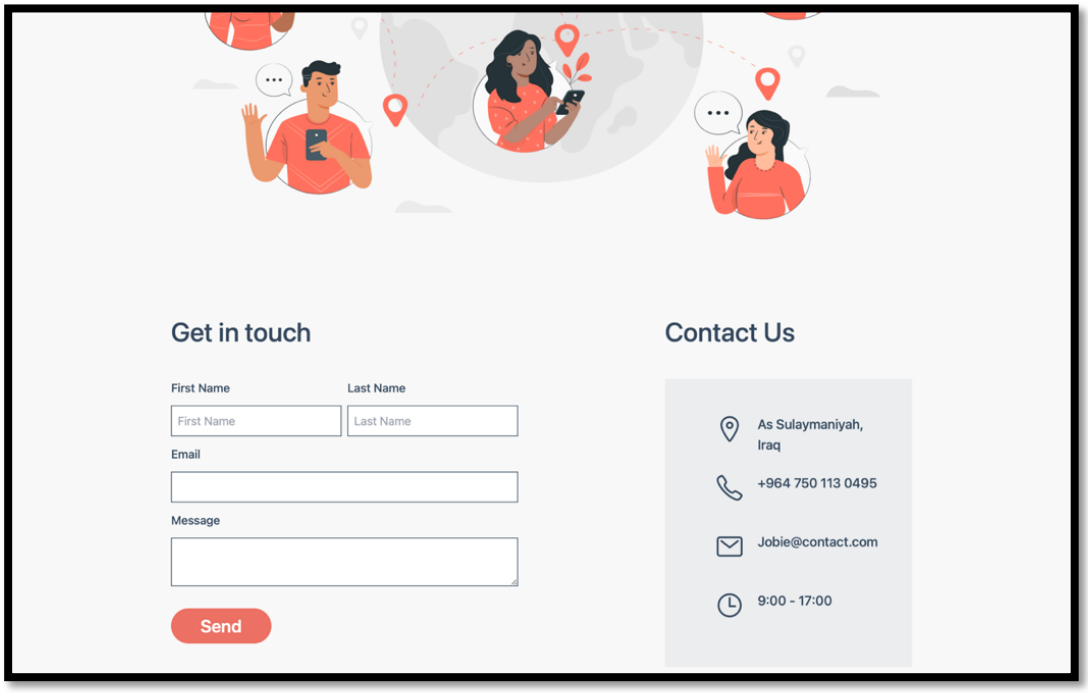

Figure 14 Contact Us UI

4.2.3 Login UI

Login or signing up is the second step in a user's journey into the system. since Gmail is widely used among users, and to make it simple and easy to access we've provided login authentication in pair with google accounts. Once user authenticates using their Gmail account, they are asked to choose their account type that defines what they want to use the system for, the visuals are shown in Figure 5 and 6.

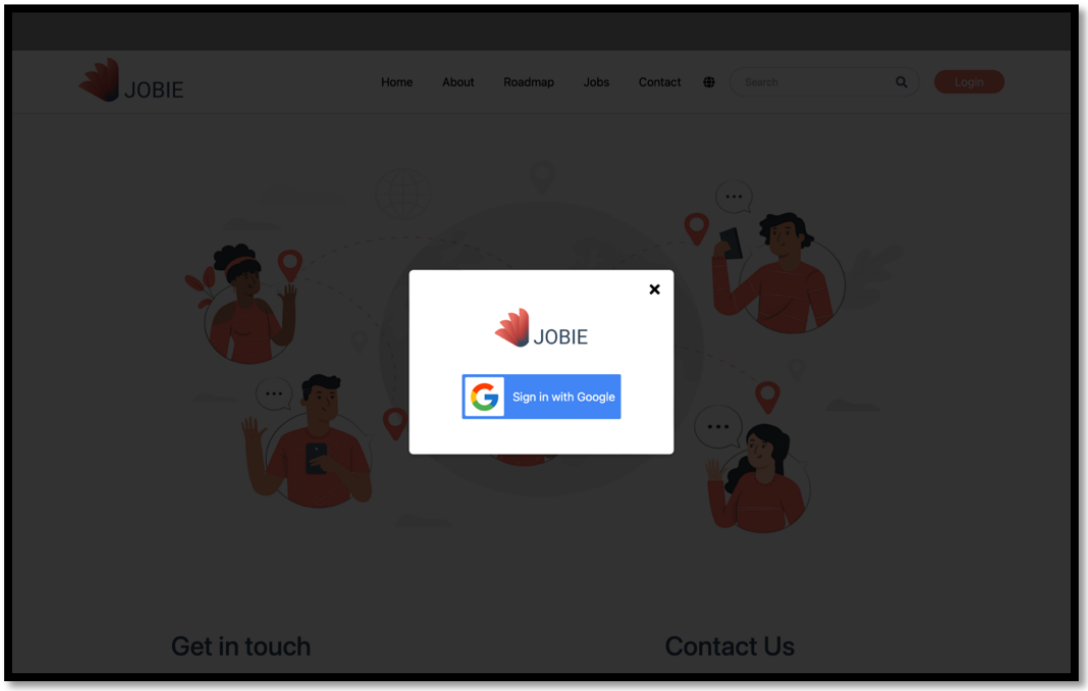

Figure 15 Login UI

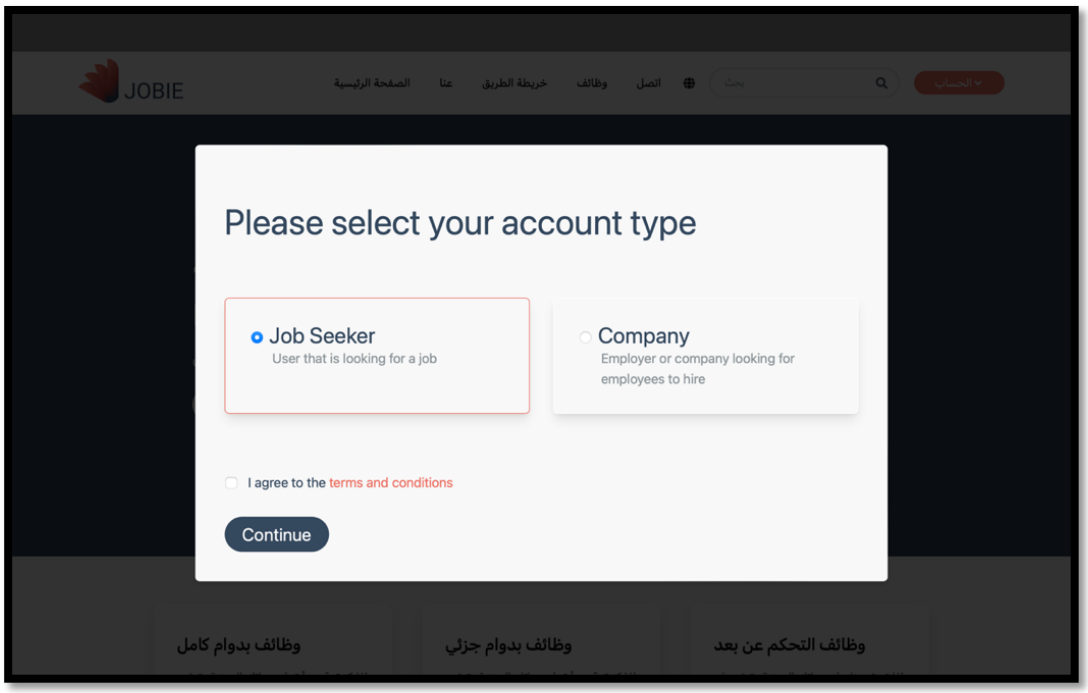

Figure 16 Account Type Selection UI

4.2.4 JOB Search UI

The job Search Interface gives quick access to many functionalities that users need when looking for jobs such as keyword, salary range, location, type of employment, and category. This allows users to easily find the job they're looking for and apply with ease. The users are also given the ability to save jobs so that they can come back to later. While searching users see a quick look at the critical aspects of any job position such as company name, location, job type, and salary range as shown in Figure 7.

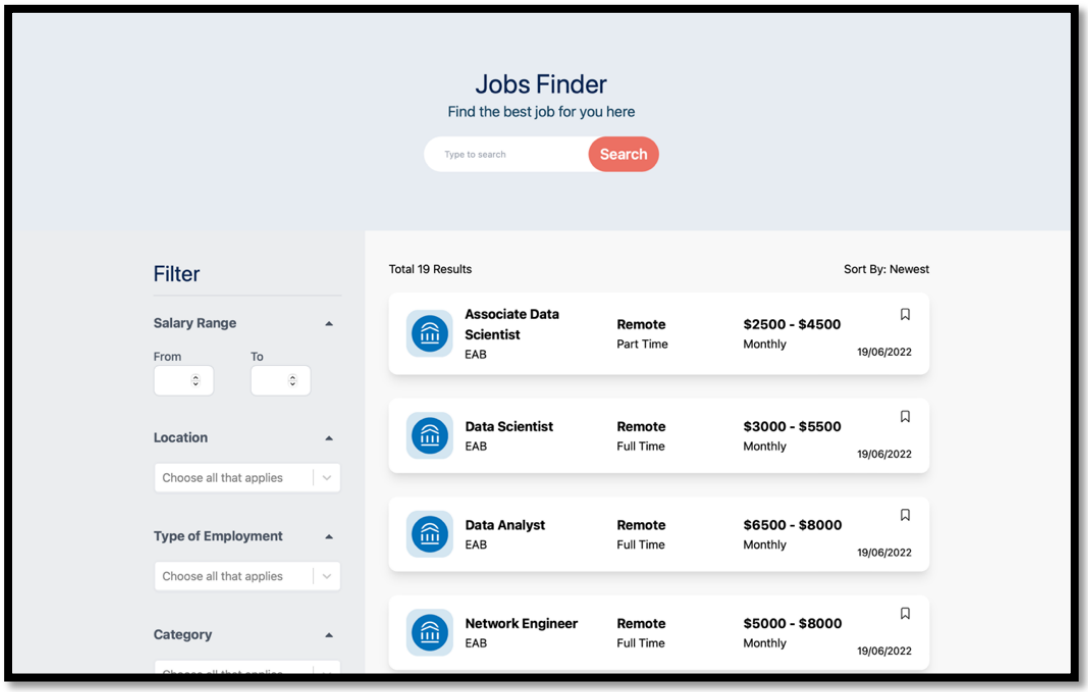

Figure 17 Job Search UI

4.2.5 JOB View UI

The job view Module shows the details of any given job position and allows users to read the information such as job description, responsibilities, and requirements. With a quick look into salary range, experience, work level, and

category. As shown in Figure 8, and 9 when the user click apply a green alert is given when their application is submitted successfully.

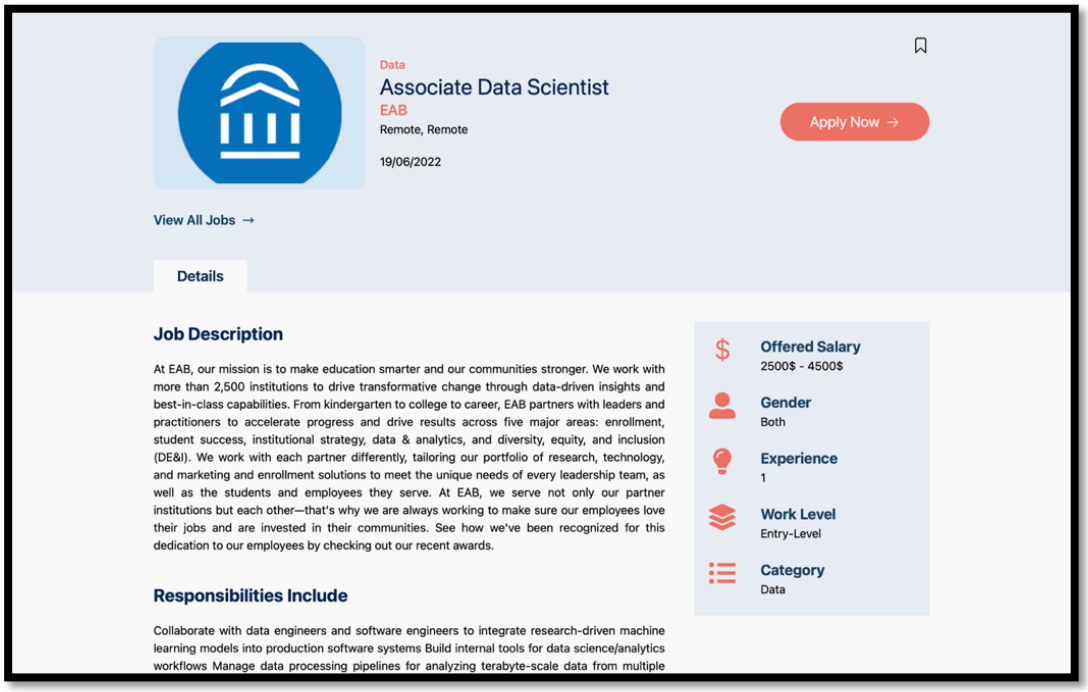

Figure 18 Job View UI

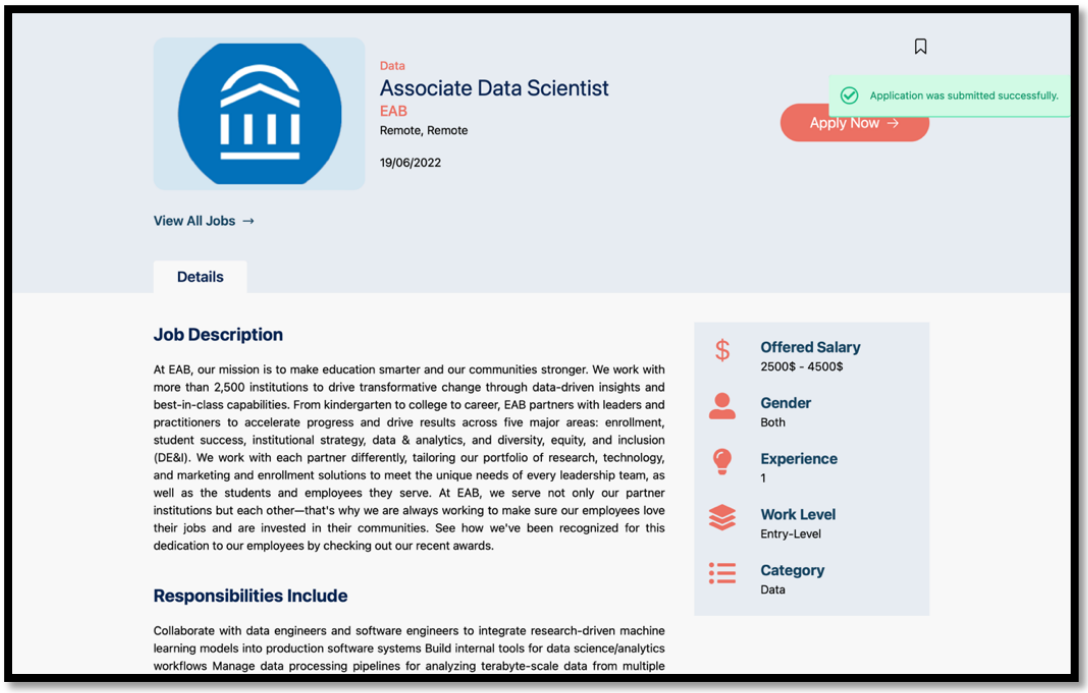

Figure 19 Job Applying Alert UI

4.2.6 Notification and User Profile Access for Company

The whole process of finding a job or an applicant is exhausting due to the vast number of options and the difficulty of the selection process, whether if it's a company, employer, or Job-Seeker. Because of this the system should tend to make every other process easier. To achieve this, we've implemented email notification so that users don't need to keep up with the system on a daily basis. When a job is posted it's available in the system to be found by potential job-seekers and when an application is submitted employer or company is informed through an email as shown in Figure 10 and 11.

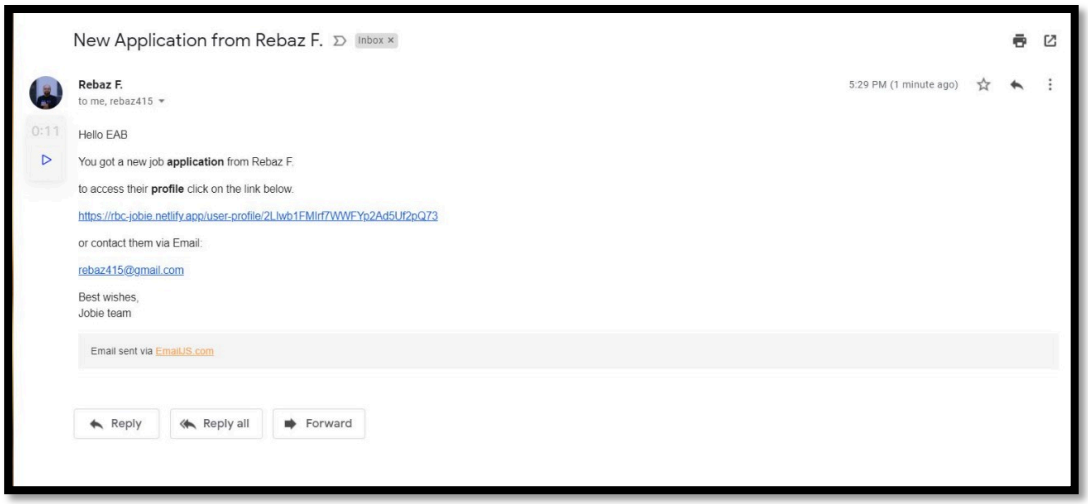

Figure 20 Email Notification

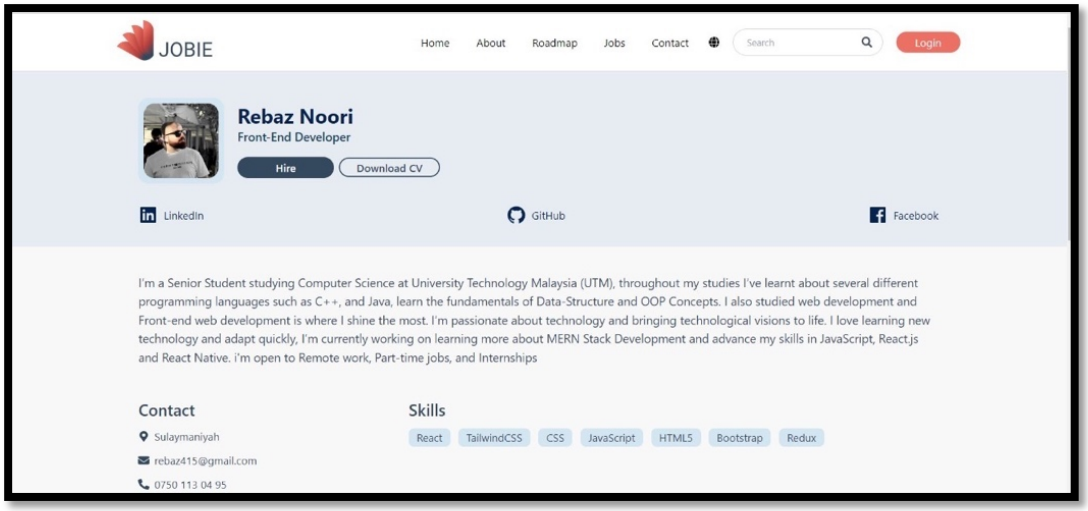

Figure 21 Applicants Profile View for Employers

4.2.7 Multi Language Support

To allow accessibility for the users of the program, another feature has been implemented which allows the users to access the system in multi languages to support a wider range of people within the community, since KRG has a very diverse community ranging from Kurds, Arabs, Turks and more. As shown in Figure 12 the users can view the system in Arabic language.

Figure 22 Multi Language Support

4.2.8 Company & User View

we've put a lot of effort into giving user profile a nice and modern look, by analyzing the necessary information needed when viewing either profile. We've included websites for both users, and for job seekers we've also added the ability to upload and view their CV. Details can be seen below in Figure 13 and 14.

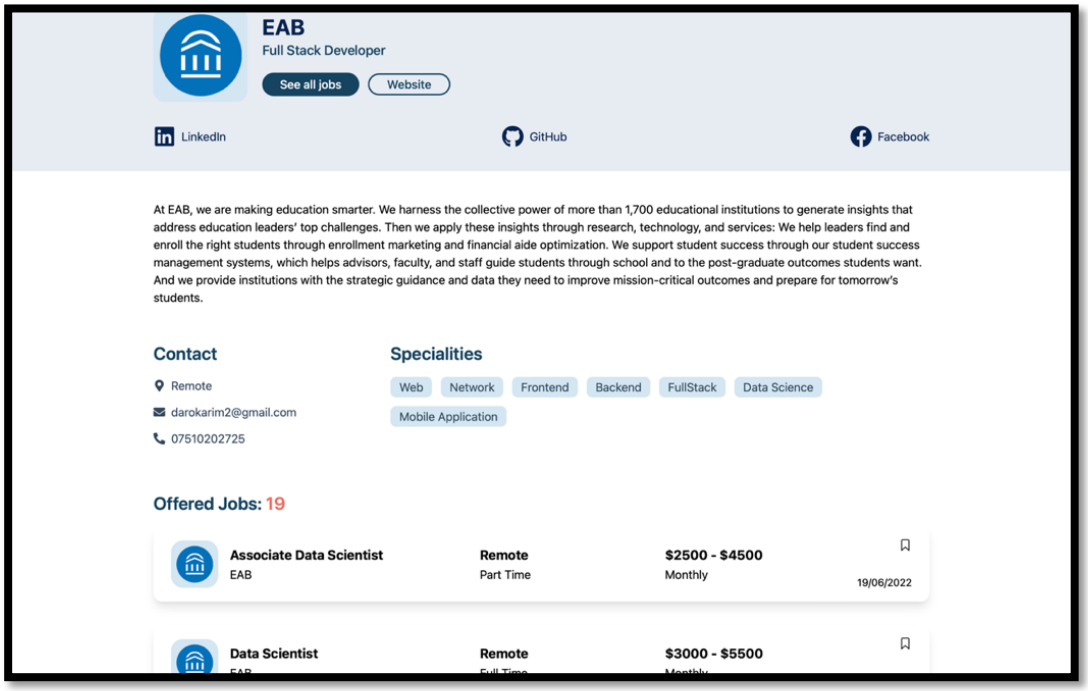

Figure 23 Company View

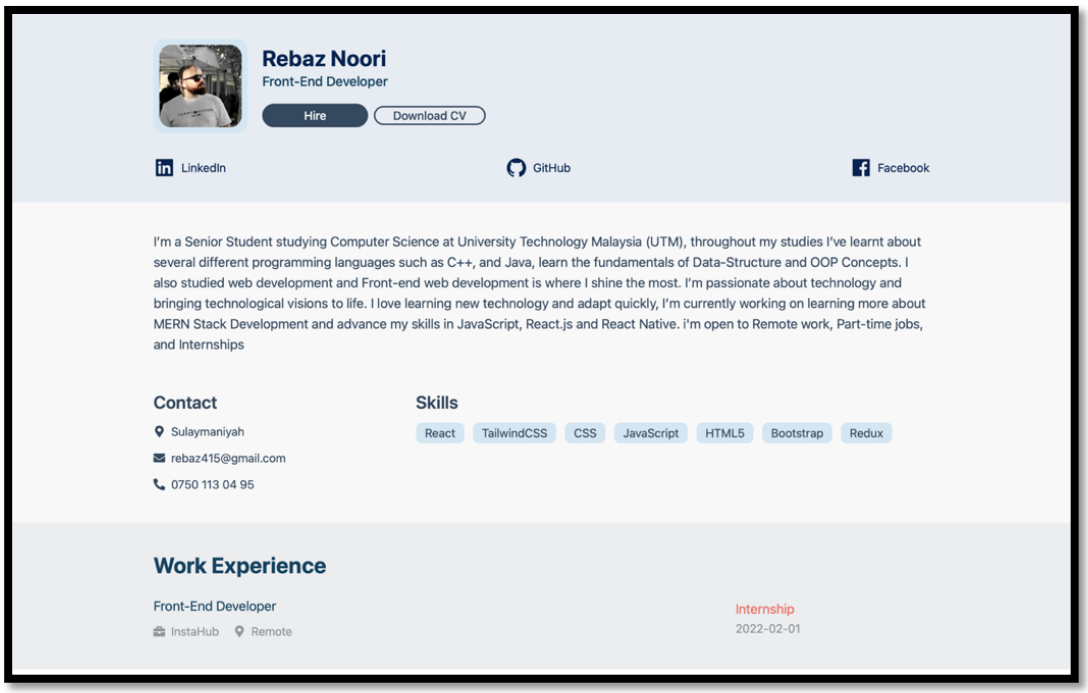

Figure 24 User View

APPENDIX B
SOFTWARE REQUIRMENT DOCUMENTATION

Software Requirements Specification

Project Title

Version 1.0

25/6/2022

Computing Department/ Computer Networks and Security

Revision Page

d. Overview

This paper aims to elaborate on the software requirements for the development of WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM.

e. Target Audience

The target audience for this system are as follows.

- Job-Seeker
- Company/Employer

f. Project Team Members

Rebaz Farid Noori.

g. Version Control History

Version	**Primary Author(s)**	**Description of Version**	**Date Completed**
1.0	**Rebaz Farid Noori**		**25th/6/2022**

Table of Contents

1. Introduction

Software Requirements Specification (SRS) aims to provide a description of the functions, features, and limitations of the WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM. The developed System is a web-based Application. Moreover, this document describes the system's target users in detail.

1.1 Project Scope

The scope of this project are as follows:

xi. The system will focus on job seekers and employers in the technological industry.
xii. The system will use React.js for Front-end and Firebase for the Backend.
xiii. The system will consist of necessary functionalities for job seekers and Employers when seeking one another.
xiv. The system will be developed as a website accessed through the internet.
xv. The system will be developed to operate within Kurdistan Regional Governate (KRG).

2. Overall Description

2.1 Product Perspective

WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM, is a web-based application aiming to provide a platform for employers and job-seekers to connect and find one another. This helps reducing the waste of time and effort by

providing an effective platform for users to connect. Moreover, the proposed web-based application allows job-seekers to create their accounts upload CVs and generate standardized profiles and search among jobs posted by employers while employers can also create profiles and post job positions that can be accessed and seen by the users.

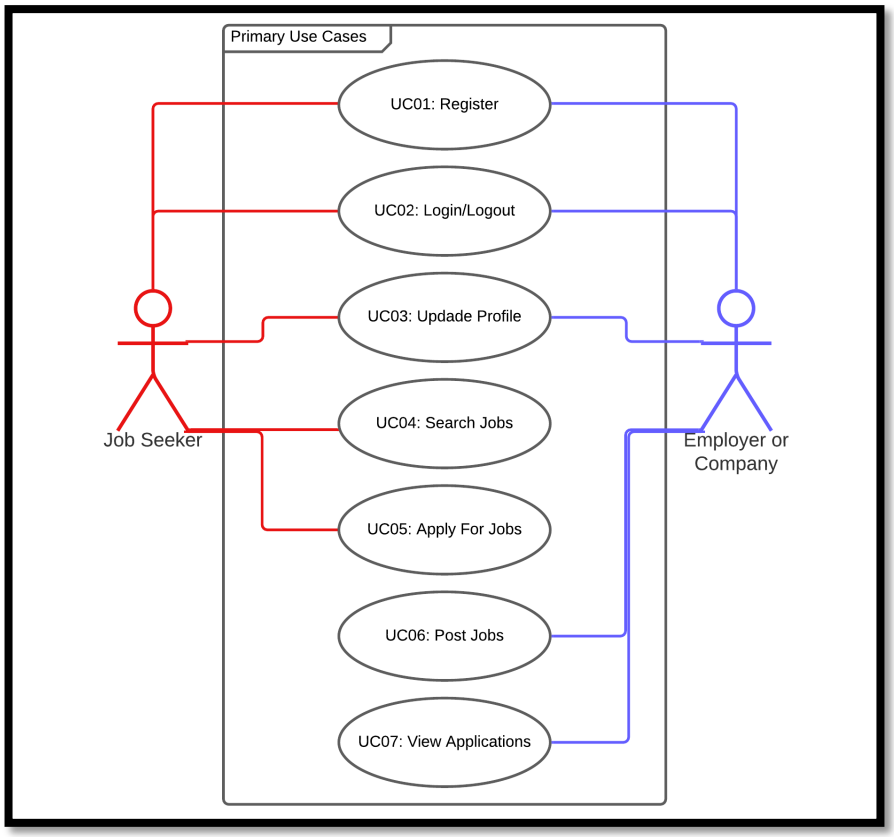

Figure 25 Use-Case Diagram for WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM

2.2 Product Functions

There are 7 Use Cases that resolves around the functions implemented within the web-based Application.

Table 3 Use-Case Description

No	Actor	Use Case	Description
1	Employer	Register	To allow users to register and select their account type.
2		Login/Logout	To allow users to log in and log out from the system.
3		Update Profile	To allow users to set up their profile and update it later on.
4		Post Jobs	To allow employers to post their open vacancies to the system.
5		View Application	To allow employers to view the applications submitted to their open positions.
6	Job-seeker	Register	To allow users to register and select their account type.
7		Login/Logout	To allow users to log in and log out from the system.
8		Update Profile	To allow users to set up their profile and update it later on.
9		Search Jobs	To allow users to search for jobs based on their preference.
10		Apply for jobs	To allow job-seekers to apply for the jobs posted by the employers.

Table 1 shows in detail the use-case diagrams for the developed web-based application, furthermore some of the main use-cases are applying for jobs, searching jobs, viewing application, and posting jobs. That represents the core function of the developed system.

2.3 User Characteristics and Constraints

Based on table 2 there are 2 main actors within the developed web-based application, they are Job-Seekers and Employers. Detailed description for the actors is as given below.

Table 4 Actor Description

No	Actor	Role
1	Job-seeker	Job-seekers can use the system to search for jobs and apply to them.
2	Company or Employer	Company or employer can use the system to post open vacancies and allow job-seekers to apply.

2.4 Operating Environments

Software and hardware Requirements of Developed System are shown in Table 3 & 4.

Table 5 Hardware Requirements

NO	Hardware	Specification
1	Random Access Memory (RAM)	8GB or above
2	Operating System (OS)	macOS or Windows
3	Network Connection	Wi-Fi or 4G
4	Storage Space	100GB and above
5	Processor	I5 6Gen, Ryzen 5 4800h, M1 and above

Laptop, Desktop, Smartphones, or Tablets can be used to access the Web-Based Employment System. To ensure optimum performance while developing the Web-Based system, the following Minimum Hardware is required to support the web-based system. Table 3 shows the minimum hardware requirements when developing the system.

Table 6 Software Requirements

NO	Software	Specification
1	integrated development environment (IDE) - VS Code v1.6	For coding
2	Web Browser (Safari, Chrome)	For output
3	Microsoft Office	For Documentation Purposes.
4	NodeJS	Runtime environment
5	MongoDB	Database Management

Software requirement is very crucial in developing a proposed system. Choosing the best software specification will lead to a smooth process of development. The minimum for software requirements is listed below. Table 4 shows minimum software requirements for development of the system.

3. System Features

3.1 UC01 Register

This section provides information about Register use case

Table 7 UC01 Register

Use Case	Register
ID	UC01
Actors	Job-Seeker/Employer
Description	To allow users to register and select their account type.
Pre-Condition	Have google account
Normal-Flow	User opens the website. User clicks on the Login/Register Button. User Logins using google account. User Selects Account Type.
Exception-Flow	If user doesn't have a google account, they are required to create a google account before registration
Post-Conditions	Actor has a Google account

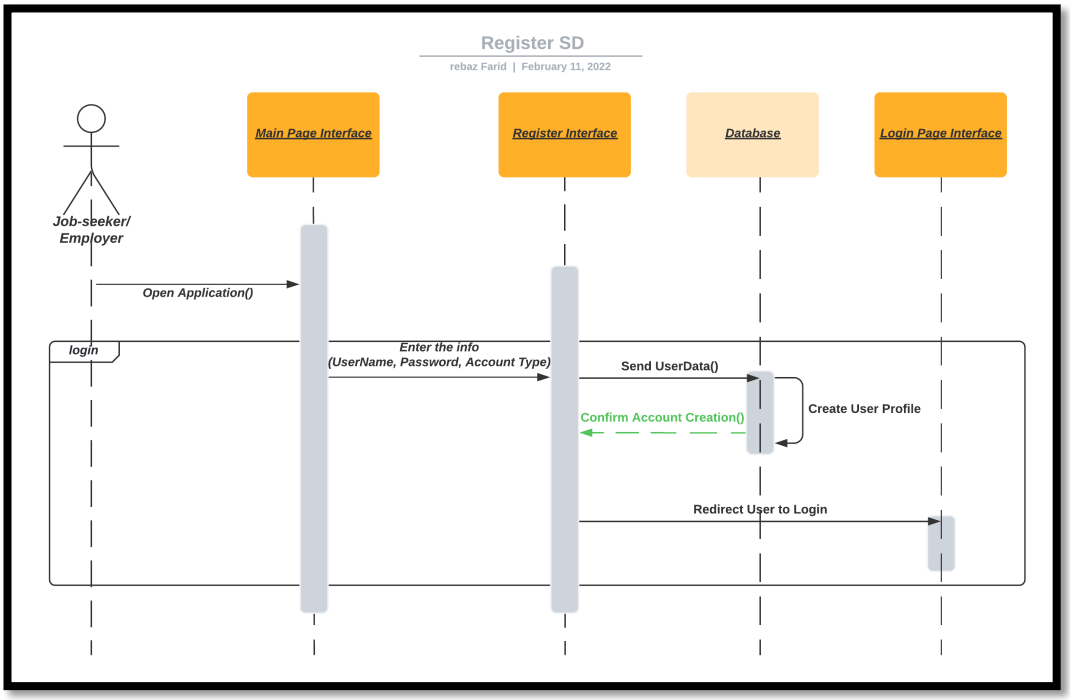

Figure 26 Register use case

3.2 UC02 Login/Logout

This section provides information about Login/Logout use case

Table 8 UC02 Login/Logout

Use Case	Login/Logout
ID	UC02
Actors	Job-Seeker/Employer
Description	To allow users to log in and log out from the system.
Pre-Condition	User Already Registered
Normal-Flow	User opens the website. User clicks on the Login/Register Button. User Logins using google account.
Alternative-Flow	If user doesn't have an account, they have to choose their account type after Login-in
Related-Requirements	Related With UC01
Post-Conditions	Actor has Registered Previously

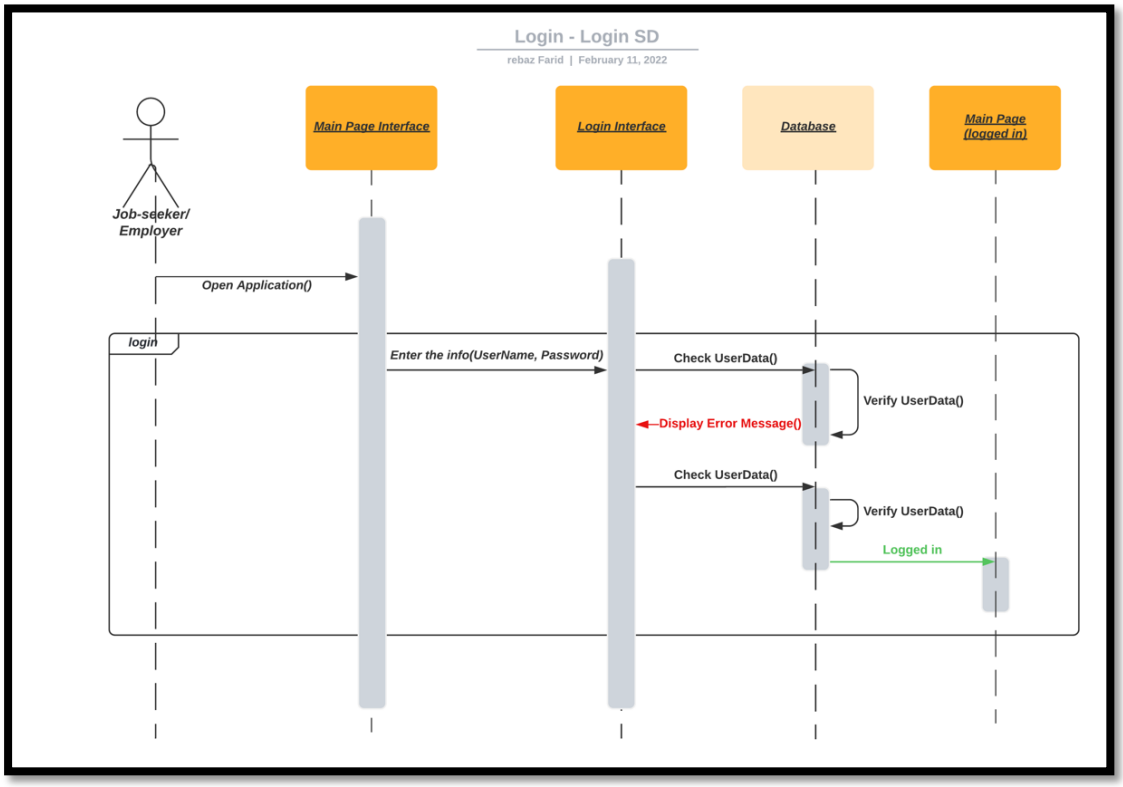

Figure 27 Login/Logout use case

3.3 UC03 Update Profile

This section provides information about Update-Profile use case

Table 9 UC03 Update Profile

Use Case	Update Profile
ID	UC03
Actors	Job-Seeker/Employer
Description	To allow users to set up their profile and update it later on.
Pre-Condition	User has an account
Normal-Flow	User logs in User navigates to edit Profile User Makes modification User Saves The newly entered information
Related-Requirements	Related with UC01
Post-Conditions	Actor is previously Registered and Logged in

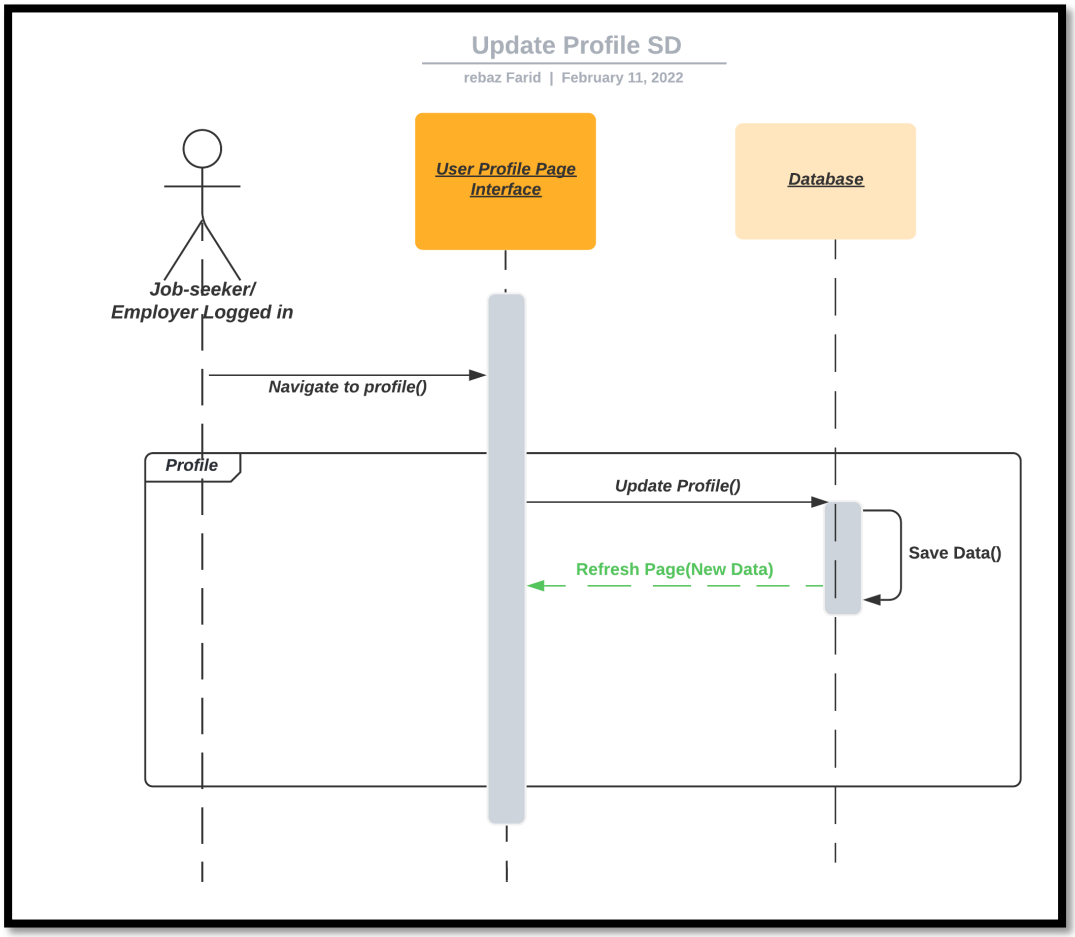

Figure 28 Update-Profile use case

3.4 UC04 Post Jobs

This section provides information about post-Jobs use case

Table 10 UC04 Post Jobs

Use Case	Post Jobs
ID	UC04
Actors	Employer
Description	To allow employers to post their open vacancies to the system.
Pre-Condition	User has an account of Employer Type
Normal-Flow	User logs in User navigates to post a job page User Enters Job Detail

	User Saves the job
Alternative-Flow	If user account is of type job seeker users need to create a different account using a different google account to post jobs
Related-Requirements	Related with UC01
Post-Conditions	Actor has an Account type of Employer and is logged in

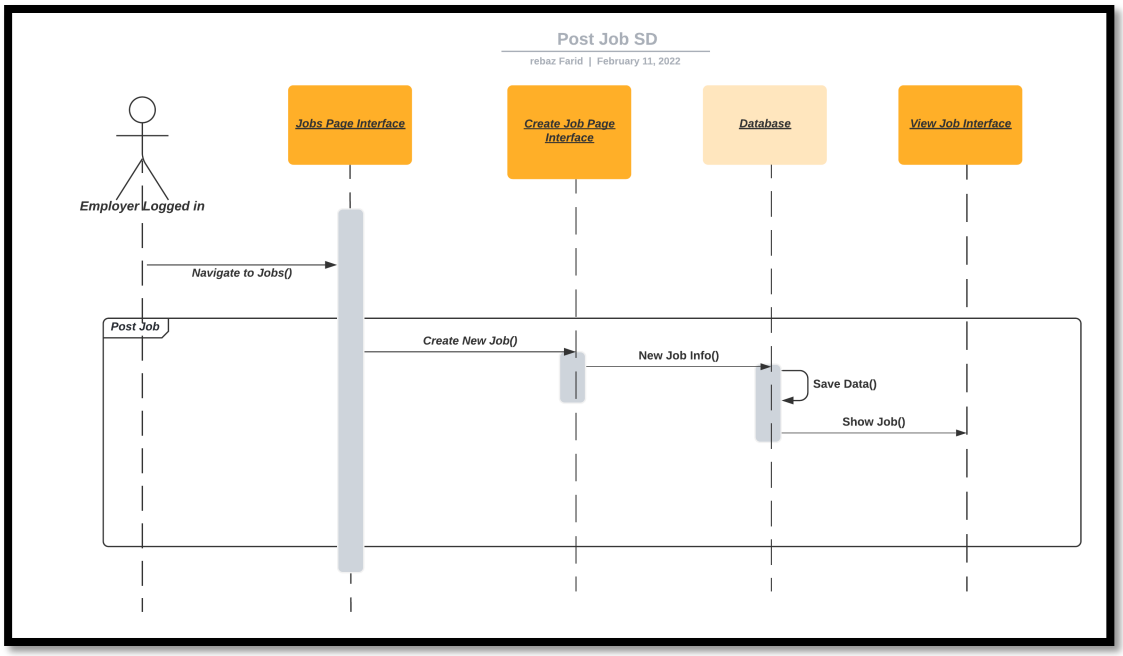

Figure 29 post-Jobs use case

3.5 UC05 View Application

This section provides information about View Application use case

Table 11 UC05 View Application

Use Case	View Application
ID	UC05
Actors	Employer
Description	To allow employers to view the applications submitted to their open positions.
Pre-Condition	User has an account of Employer Type
Normal-Flow	User Posts a job. Job-Seekers Apply for the job.

	User Views Job Seeker's Application through the link received in the email.
Related-Requirements	Related with UC01 & UC07
Post-Conditions	Actor has an active job Position that has one or more applications submitted

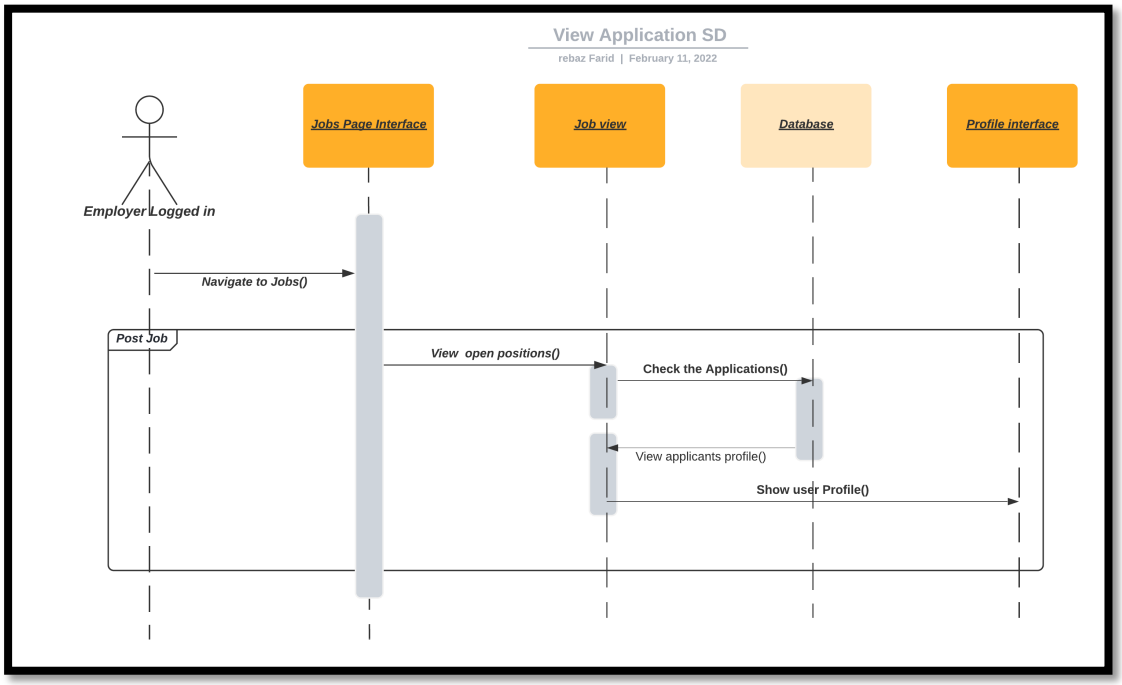

Figure 30 View Application use case

3.6 UC06 Search Jobs

This section provides information about Search-Jobs use case

Table 12 UC06 Search Jobs

Use Case	Search Jobs
ID	UC06
Actors	Job-Seeker
Description	To allow users to search for jobs based on their preference.
Pre-Condition	User has an account of Job-Seeker Type
Normal-Flow	User Logs in to their account

	User Navigates to jobs page User enters key word to search for their desired position
Related-Requirements	Related with UC05
Post-Conditions	Actor has an account type of Job-Seeker

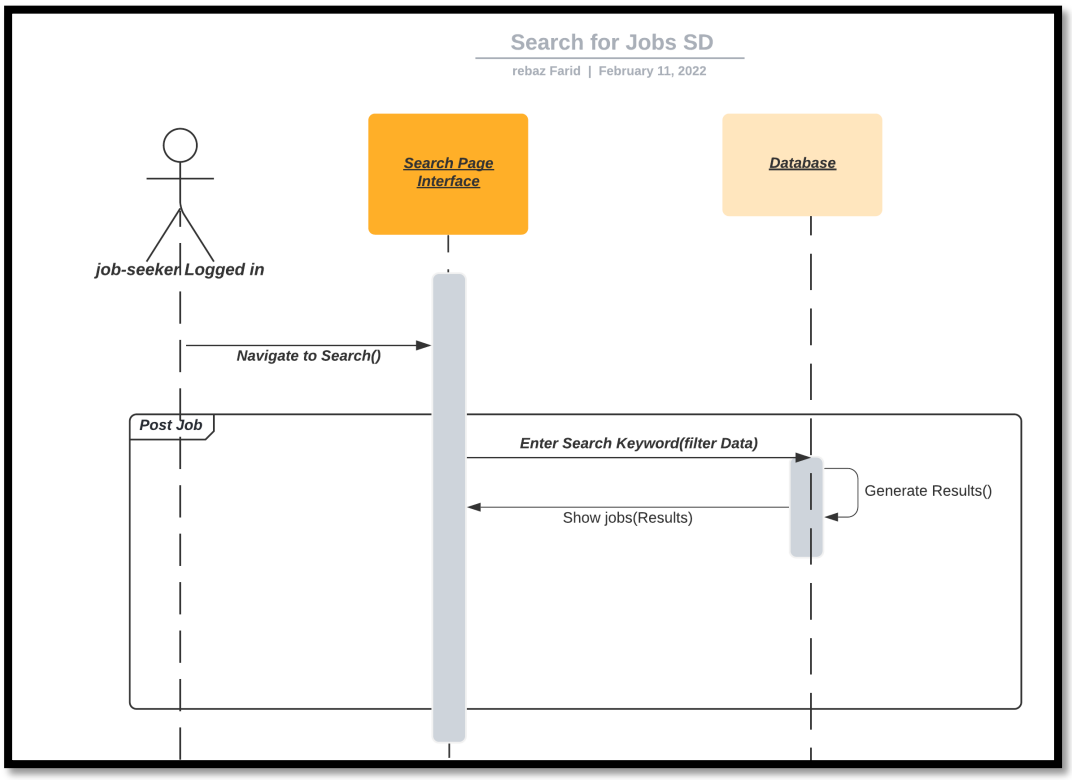

Figure 31 Search-Jobs use case

3.7 UC07 Apply for jobs

This section provides information about Apply-for-Jobs use case

Table 13 UC07 Apply for jobs

Use Case	Apply for jobs
ID	UC07
Actors	Job-Seeker
Description	To allow job-seekers to apply for the jobs posted by the employers.
Pre-Condition	User has an account of Job-Seeker Type

	User is Logged in User is Viewing a Job
Normal-Flow	User Clicks Apply after finding and viewing their desired job in the system
Related-Requirements	Related with UC06
Post-Conditions	Actor has an account type of Job-Seeker and has found their desired Position within the system while also has an active profile with up-to-date information

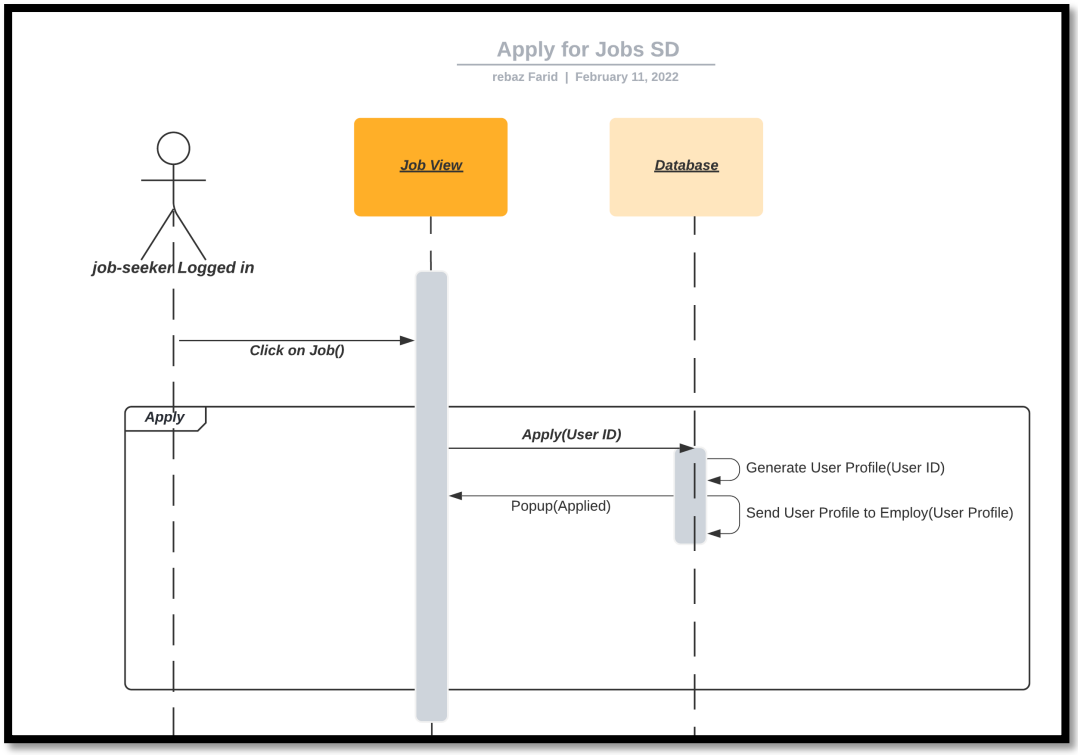

Figure 32 Apply-for-Jobs use case

3.8 Activity diagram

The activity diagram shows the flow of the users and their possible actions, since our system has two users, we have two activity diagram which are listed below.

3.8.1 Employer

The employer activity diagram shows the flow of actions for the employer actor within our system.

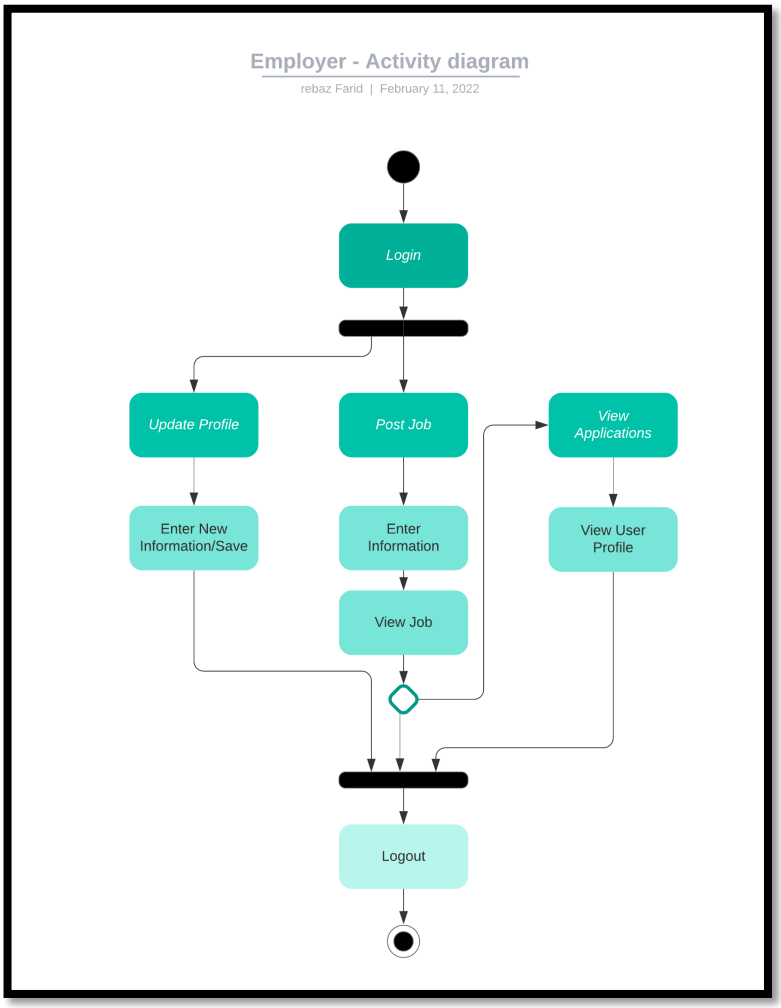

Figure 33 Employer Activity Diagram

3.8.2 Job-Seeker

The Job-Seeker Activity Diagram shows the flow of possibly actions and events for Job-Seeker Actor within the developed system.

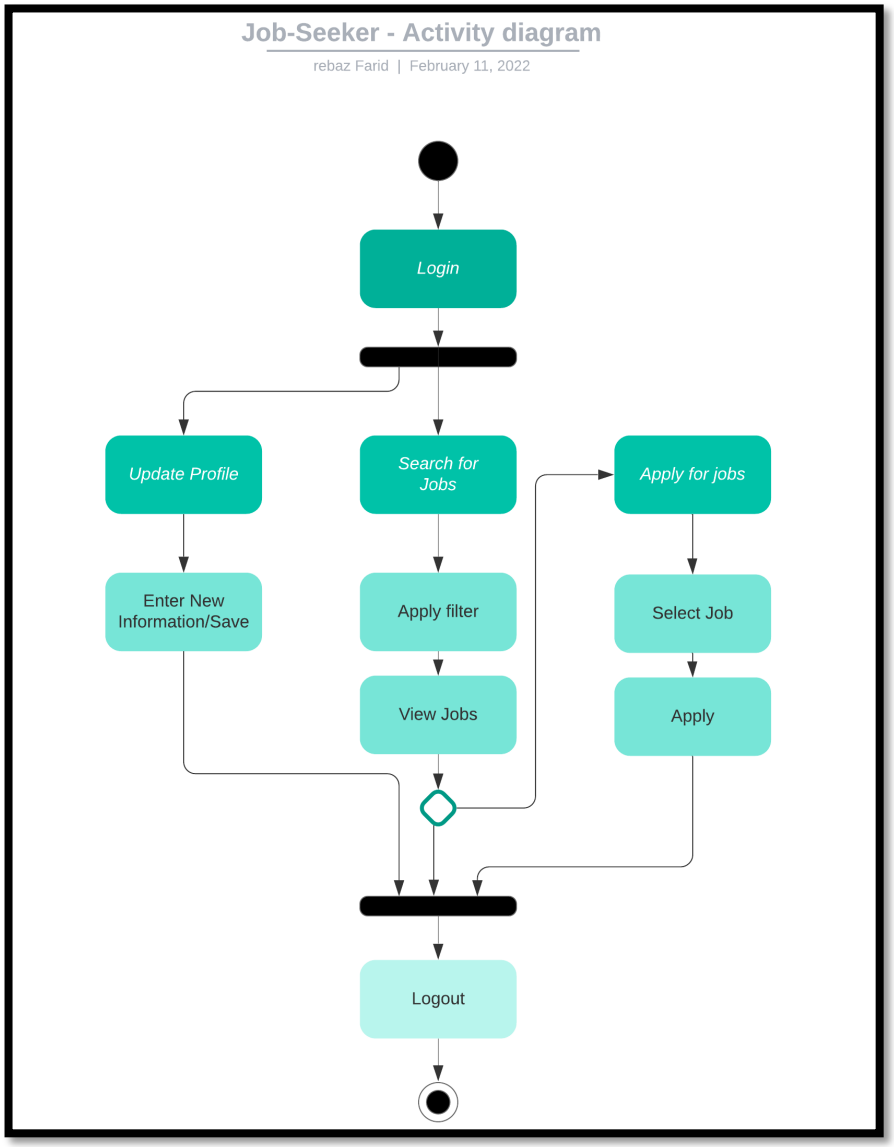

Figure 34 Job-Seeker Activity Diagram

4. NON-Functional Requirements

Non-Functional Requirements	Description
Availability	The system is available for businesses and charity organization its goal is to act as a bridge of communication between them.
Usability	The system should be usability to all types of users in terms of navigation and accessibility of features so that it can benefit all types of users.

Performance	It could make a user want to or not want to use your service based on the performance because one thing for sure users don't like to wait is why performance could have a high impact on how likely it is for a user to use your webapp. We have thought of this and ensured the system would be as responsive as answering to a user request <2s while it maintain over 1,000 active users.
Security and Authentication	To prevent user data manipulation, we have added accounts in which a user can access and modify only their data, and all data can be altered by the developers. This will allow us to prevent unauthorized access to user data.
Portability	The project is going to be built as a webapp which makes it available on almost every platform with a browser. Such as (Android, IOS, MacOS, IPadOS, Windows, Linux, ChromeOS) this is a very crucial point since it gives accessibility to the most people, and that is portability at its best.

APPENDIX C
SOFTWARE TESTING DOCUMENTATION

Software Testing Documentation

WEB-BASED PROFESSIONAL EMPLOYMENT-ORIENTED SYSTEM

Version 1.0

Printing Date

25/06/2022

Department and Faculty

Computer Networks and Security

Prepared by:

Rebaz Farid Noori

Revision Page

h. Overview

This Version of the Software Testing Documentation (STD) Contains the final results and analysis for the developed Web-Based application.

i. Target Audience

The target Audiences of the web-based applications are Job-Seekers & Employers, they will also be reached out to in order to analysis and perform the test cases of the system.

j. Version Control History

Version	Primary Author(s)	Description of Version	Date Completed
1.0	**Rebaz Farid Noori**	**Final Version**	**25th June 2022**

Table of Contents

1. Introduction

STD is the practice of detecting issues in software requirements, Design, and implementation. STD is used to verify Software Development for correctness, completeness, security, and quality In terms of specification. This testing evaluates the value of developed software. It showcases all the issues within the system such as flows, issues, errors, and weaknesses. Various ways can be taken to conduct STD such as White Box Testing and Black Box Testing, Grey Box Testing, and Agile Testing. While there are testing levels such as unit testing, system testing, acceptance testing, integration testing. For the developed software we're conducting black box testing and User Acceptance Testing.

1.1 Purpose

This STD Report showcases the results and discussions for the testing procedures of the developed software while also recording User Acceptance Testing Perform by the targeted Users

1.2 Scope

xvi. The system will focus on job seekers and employers in the technological industry.

xvii. The system will use React.js for Front-end and Firebase for the Backend.

xviii. The system will consist of necessary functionalities for job seekers and Employers when seeking one another.

xix. The system will be developed as a website accessed through the internet.

xx. The system will be developed to operate within Kurdistan Regional Governate (KRG).

1.3 System Overview

This STD Document elaborates and showcases the results and procedures in detail for the following points.

1. Test Cases, Data, and Expected Results.
2. Black Box Testing.
3. User Acceptance Testing.

2. Test Cases, Data and Expected Results

2.1 Test TC001 for Account Registration: Register UC01

This section consist of login attempts tests for the developed system

Test Case ID	TC001_01_01	TC001_01_02	TC001_01_03	TC001_01_04
Action/Input				
Used Email	Rebaz415@gmail.com	Daro.21@hotmail.com	Renwar33@gmail.com	Sana 366@yahoo.com
Used Password	Re57baz	Daro21	Sima@76	Sana2
Password Sufficient	Yes	yes	yes	no
Output				
Login Successful	Yes	Yes	Yes	
Login Failed				No
Expected Result				
Redirect to Landing Page	Yes	Yes	Yes	No
Testing Result	Pass	Pass	Pass	Pass

2.2 Test TC002 for Account Login: Login UC02

This section consist of login attempts tests for the developed system

Test Case ID	TC002_01_01	TC002_01_02	TC002_01_03	TC002_01_04
Action/Input				
Used Existing User Email	Yes	Yes	NO	Yes
Used Correct Password	Yes	No	NO	No
Expected Result				
Redirect to main page	Yes	No	No	NO
Testing Result	Pass	Pass	Pass	Pass

2.3 Test TC003 for Manage Course: Manage Course UC03

This section consists of Manage Course tests for the developed system

Test Case ID	TC003_01_01	TC003_01_02	TC003_01_03	TC003_01_04
Action/Input				
User has account type of teacher	Yes	Yes	NO	Yes
User adds new course information	Yes	Yes	NO	Yes
User submits new information	Yes	Yes	NO	Yes
Output				
New Course Added	Yes	Yes	No	Yes
Expected Result				
New Course Saved	Yes	Yes	No	Yes
Testing Result	Pass	Pass	Pass	Pass

Test Case ID	TC003_02_01	TC003_02_02	TC003_02_03	TC003_02_04
Action/Input				
User has account type of teacher	Yes	Yes	NO	Yes
User Edits Course	Yes	Yes	NO	Yes
User Submits new information	Yes	Yes	NO	No
System Validates new Course Data	Yes	Yes	NO	NO
Output				

Course Modified	Yes	Yes	No	NO
Expected Result				
Testing Result	Pass	Pass	Pass	Pass

Test Case ID	TC003_03_01	TC003_03_02	TC003_03_03	TC003_03_04
Action/Input				
User has account type of teacher	Yes	Yes	NO	Yes
User selects course to edit	Yes	Yes	NO	No
User confirms deletion of course information	Yes	No	NO	No
System Deletes course Data	Yes	No	NO	NO
Output				
Course Deleted	Yes	No	No	NO
Expected Result				
Testing Result	Pass	Pass	Pass	Pass

2.4 Test TC004 for Upload Course Material: Upload Course Material UC04

This section consists of Upload Course Material tests for the developed system

Test Case ID	TC004_01_01	TC004_01_02	TC004_01_03	TC004_01_04
Action/Input				
User Account Type is Teacher	Yes	Yes	No	Yes
User Navigates to upload new Material	Yes	Yes	No	Yes
User adds new course material	Yes	Yes	No	No
Output				
Course Material Saved	Yes	Yes	No	No
Expected Result				
Course Material Saved	Yes	Yes	No	No
Testing Result	Pass	Pass	Fail	Pass

2.5 Test TC005 for View Course: View Course UC05

This section consists of View Course tests for the developed system

Test Case ID	TC005_01_01	TC005_01_02	TC005_01_03	TC005_01_04
Action/Input				
User Account Type is Teacher	Yes	Yes	Yes	No
User Selects Course Card	Yes	Yes	yes	No
System Displays Course Detail	Yes	Yes	Error_Message	No
Output				
User can view course detail	Yes	Yes	No	No
Testing Result	Pass	Pass	Fail	Fail

2.6 Test TC006 for Submit Assignment: Submit Assignment UC06

This section consists of Submit Assignment tests for the developed system

Test Case ID	TC006_01_01	TC006_01_02	TC006_01_03	TC006_01_04
Action/Input				
User Account Type is Student	Yes	Yes	Yes	No
User Clicks submit Assignment	Yes	Yes	yes	No
User Fills the form	Yes	Yes	No	No
System Validates Assignment Data	Yes	Yes	No	No
Output				
User Submitted Assignment	Yes	Yes	No	No
Testing Result	Pass	Pass	Fail	Fail

2.7 Test TC007 for Chat: Chat UC07

This section consists of Submit Assignment tests for the developed system

Test Case ID	TC007_01_01	TC007_01_02	TC007_01_03	TC007_01_04
Action/Input				
User Account Type is Student, and Teacher	Yes	Yes	Yes	Yes
User Navigates to chat	Yes	Yes	yes	Yes
System Displays chat Form	Yes	Yes	Yes	Yes
User can send enter and send a message	Yes	Yes	Yes	No
Output				
User can communicate using chat function	Yes	Yes	Yes	Show Error Message
Testing Result	Pass	Pass	Pass	Fail

3. User Acceptance Test

The Purpose of the User Acceptance Test is to Evaluate the Developed System Based on the User's Experience and feedback. The result of this section was determined through a survey that was answered by potential users that are either job-seekers or employers.

3.1 Survey

Scale of the Survey that is used to Determine the User's Satisfaction is 1-5 and they mean as follows.

1- Very Bad 2- Bad 3- Neutral 4- Good 5- Very Good

Our user acceptance test targeted two user's Job-Seekers and Employers individually, the ratio between the participants is as described in the following figure.

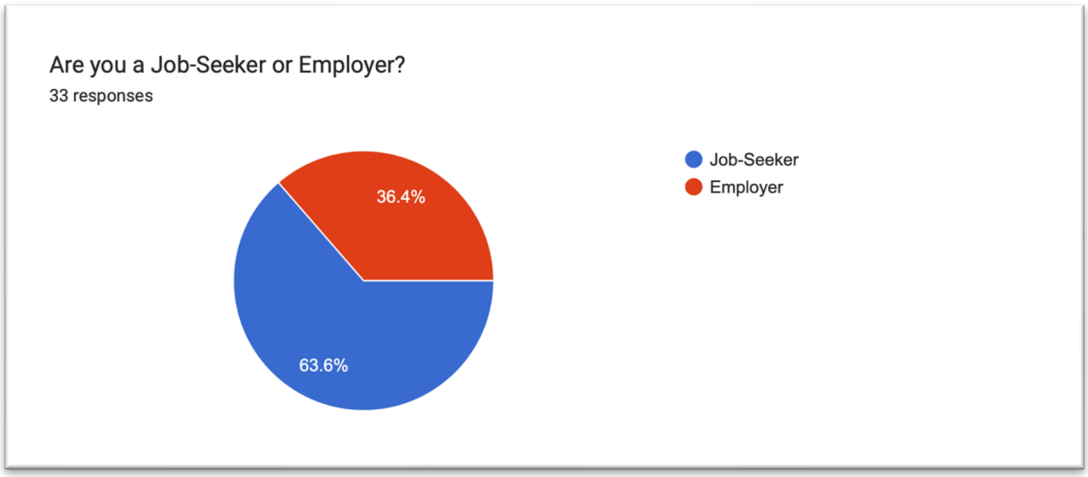

Figure 35 Ratio of Employer Vs Job-Seeker

3.1.1 Employer

In this section we'll elaborate on the User Acceptance Survey and discovery based on the tests we have made, that was conducted on Employers who have tested our system and has answered our survey.

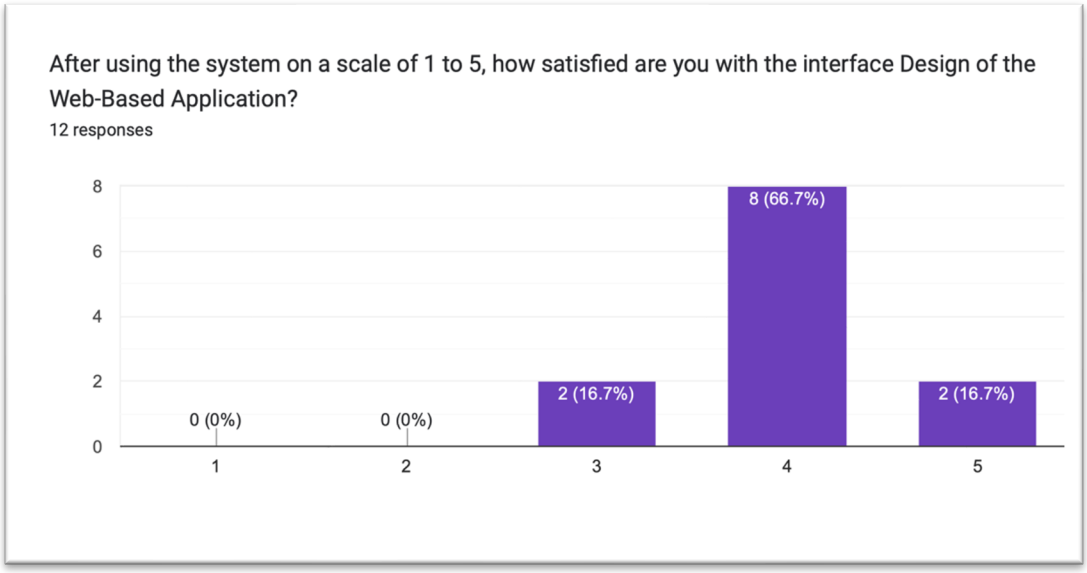

Figure 36 Employer's Satisfaction with Design

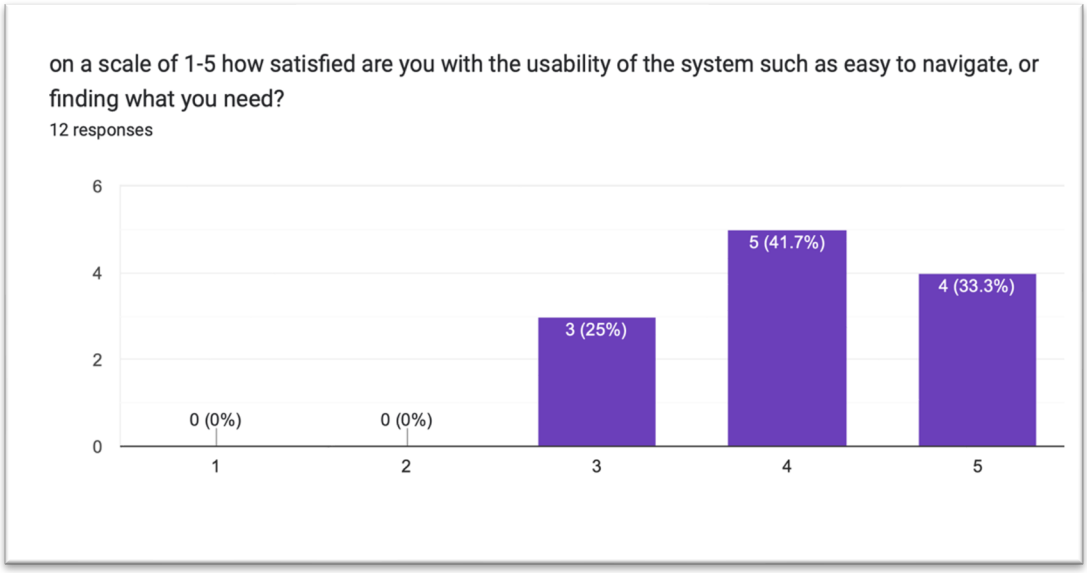

Figure 37 Employer's Satisfaction with Navigation

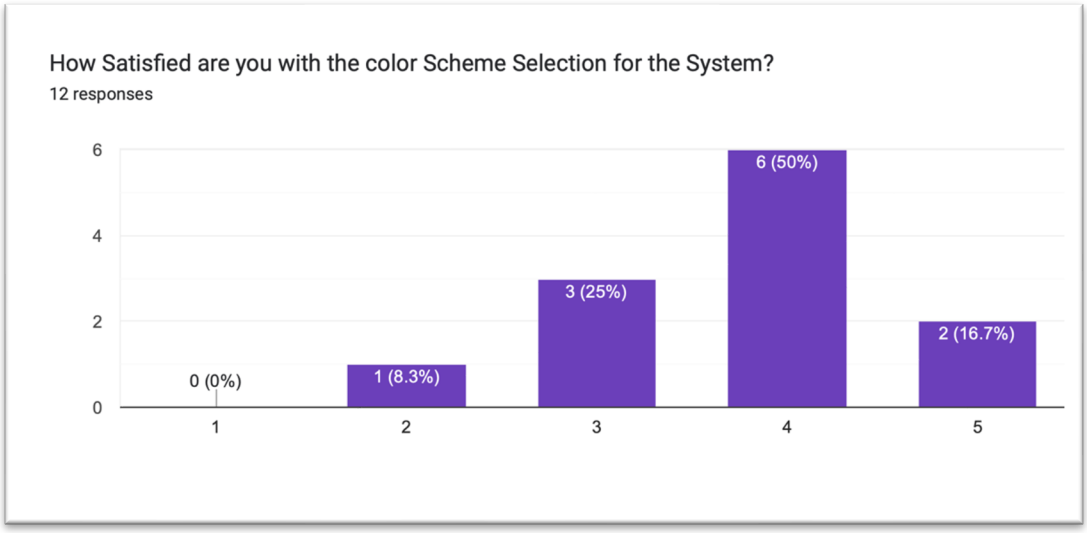

Figure 38 Employer's Satisfaction with color Scheme

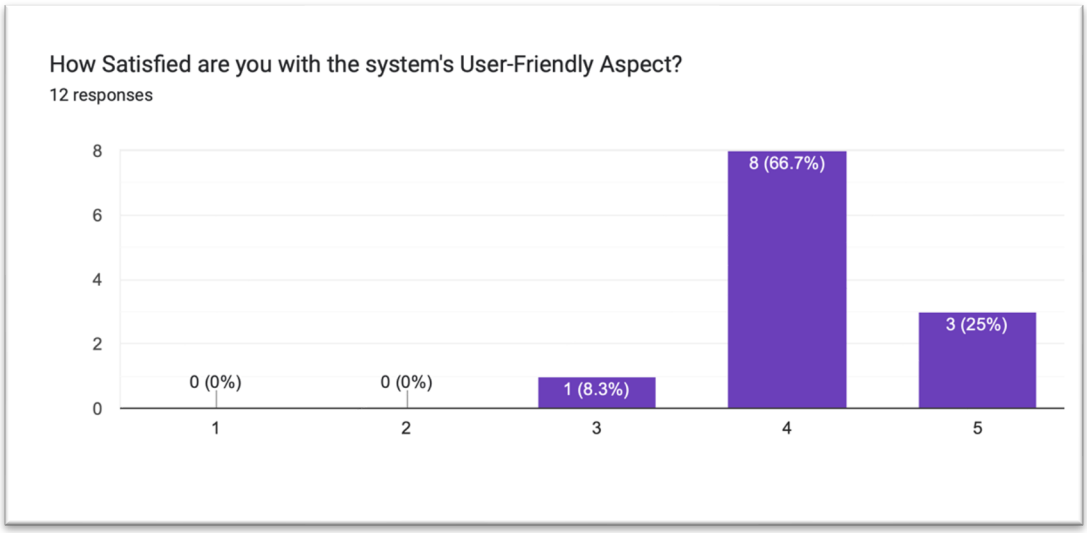

Figure 39 Employer's Satisfaction with user friendliness

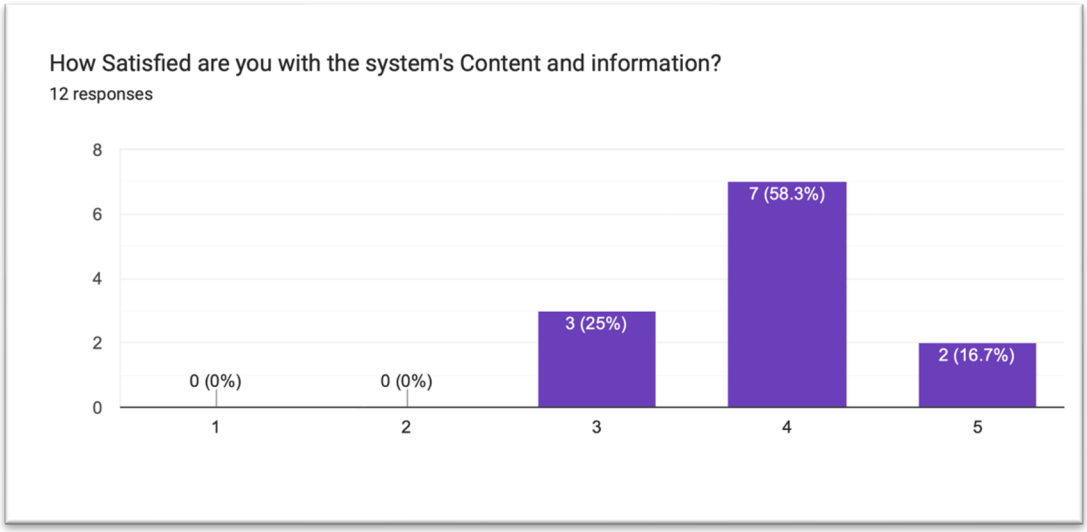

Figure 40 Employer's Satisfaction with information and content

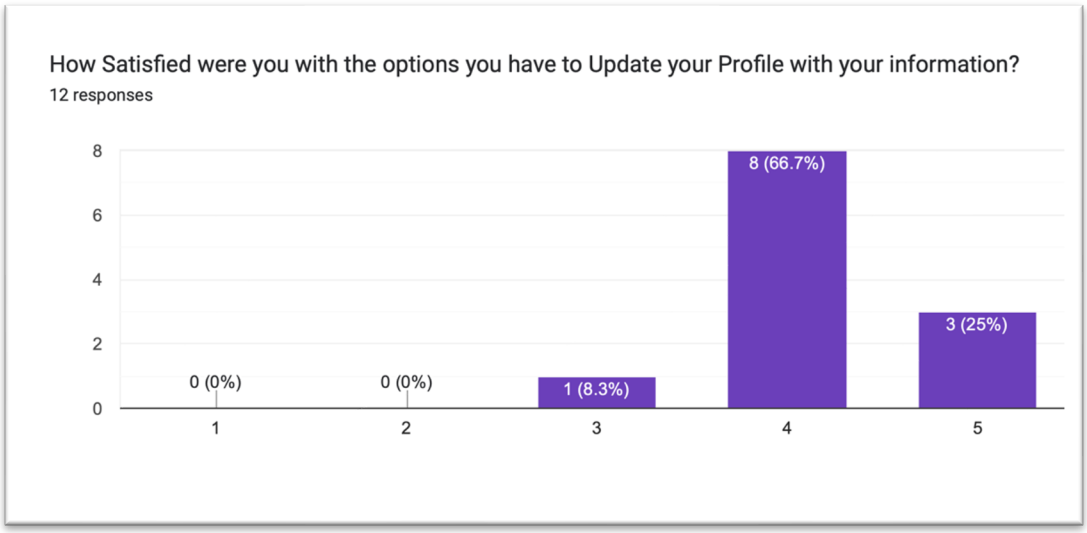

Figure 41 Employer's Satisfaction with update profile feature

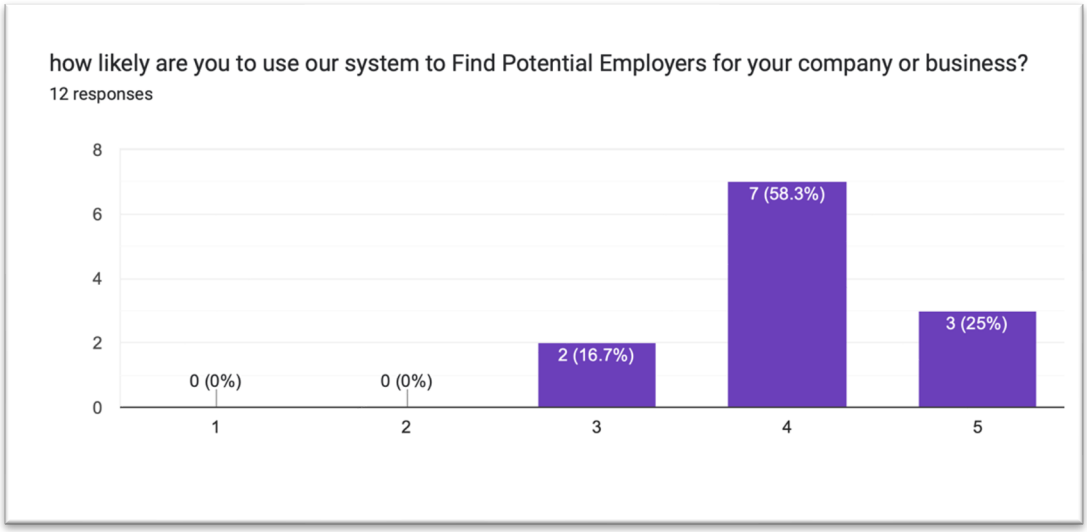

Figure 42 Employer's intention to use

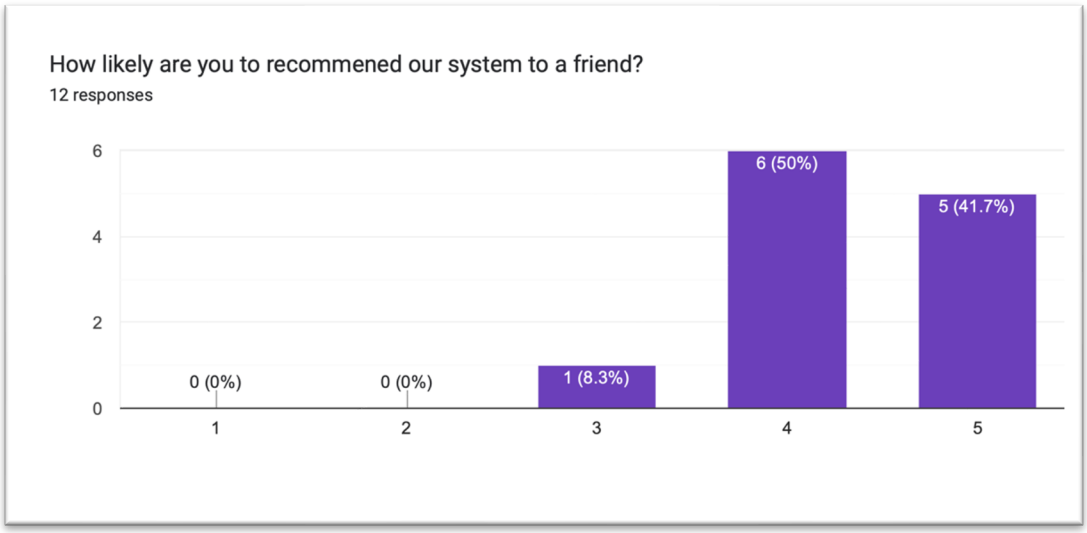

Figure 43 Employer's Intention to recommend to use

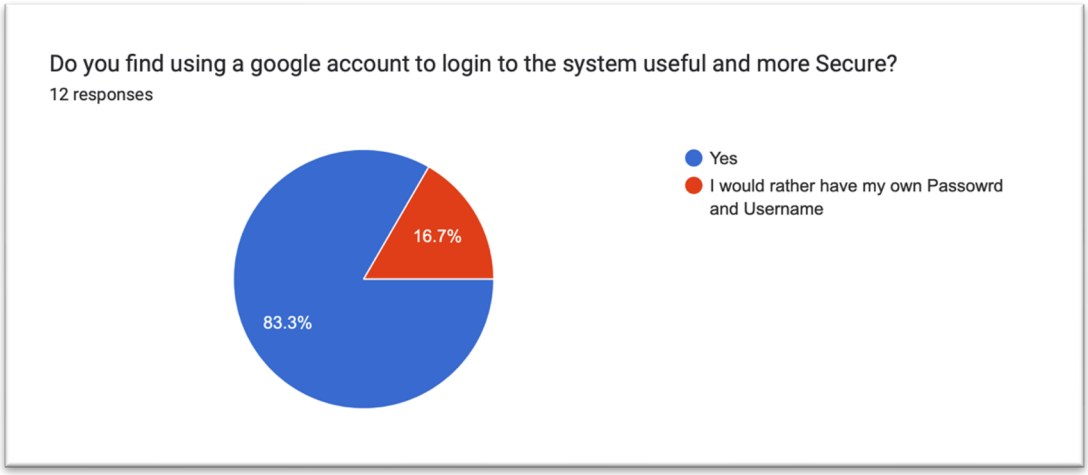

Figure 44 Employer's feel of safety

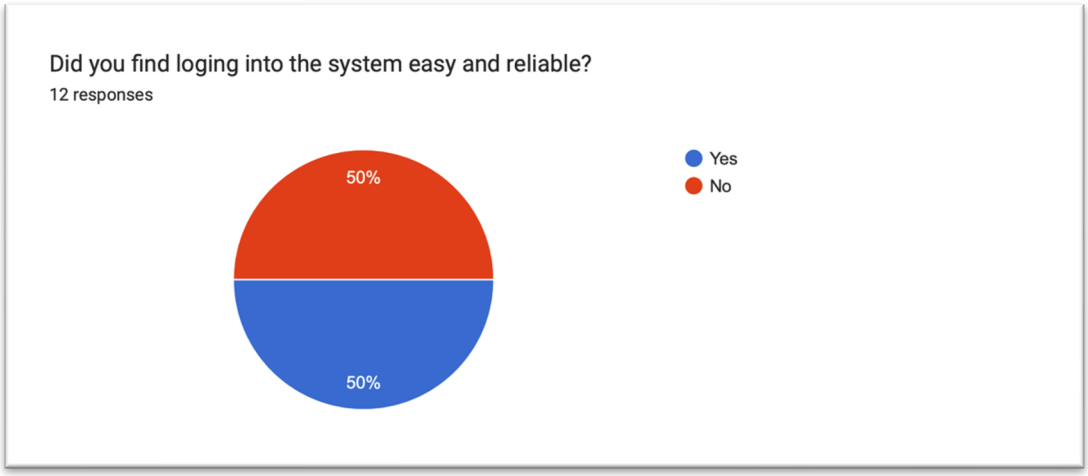

Figure 45 Employer's Satisfaction with Login Method

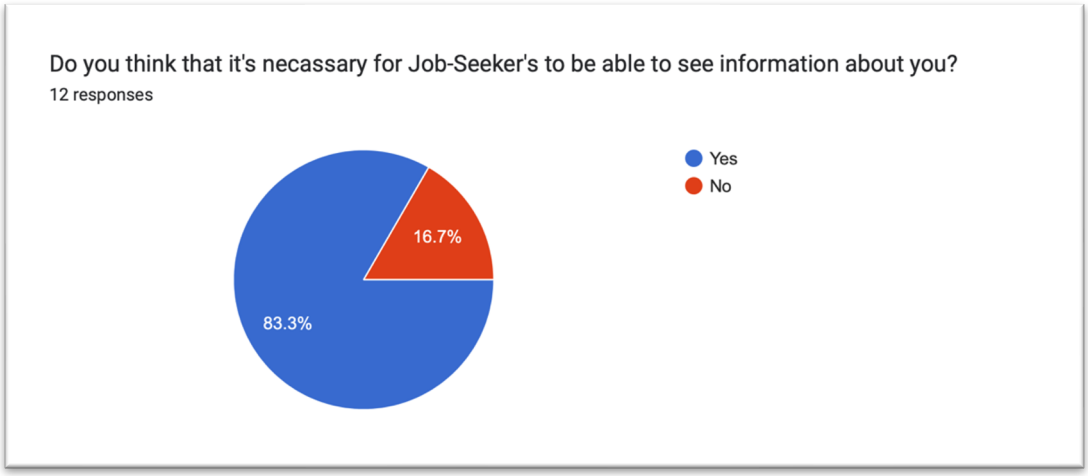

Figure 46 Employer's Satisfaction with sharing information

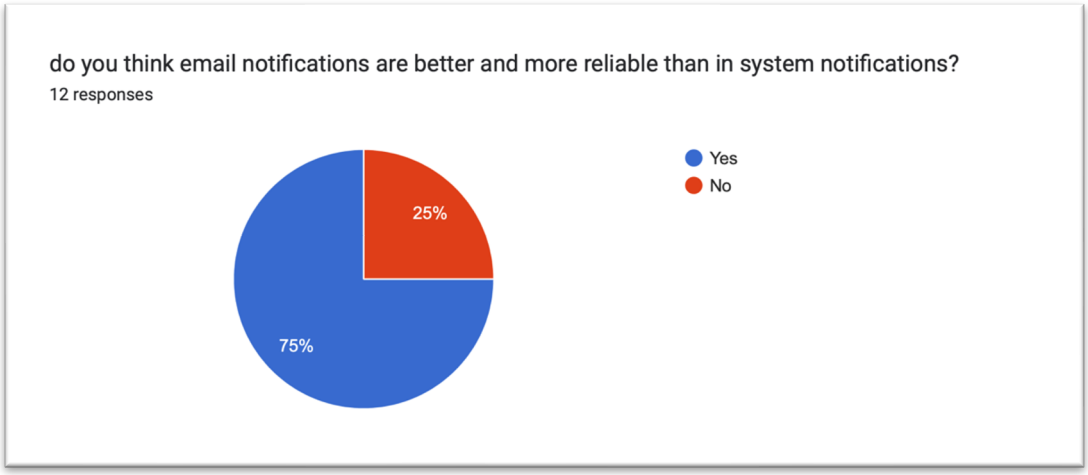

Figure 47 Employer's Satisfaction with notification method

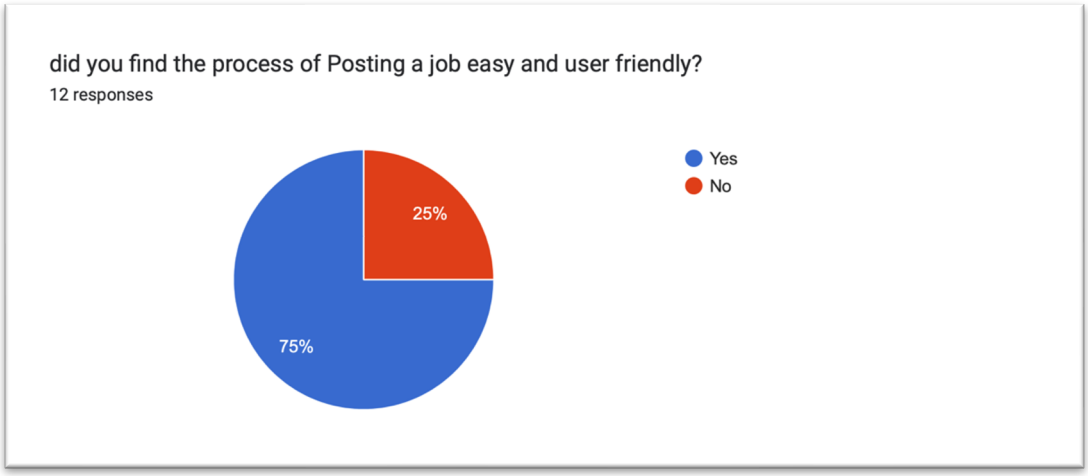

Figure 48 Employer's Satisfaction with posting a job

3.1.2 Job-Seeker

In this section we'll elaborate on the User Acceptance Survey and discovery based on the tests we have made, that was conducted on Job-Seekers who have tested our system and has answered our survey.

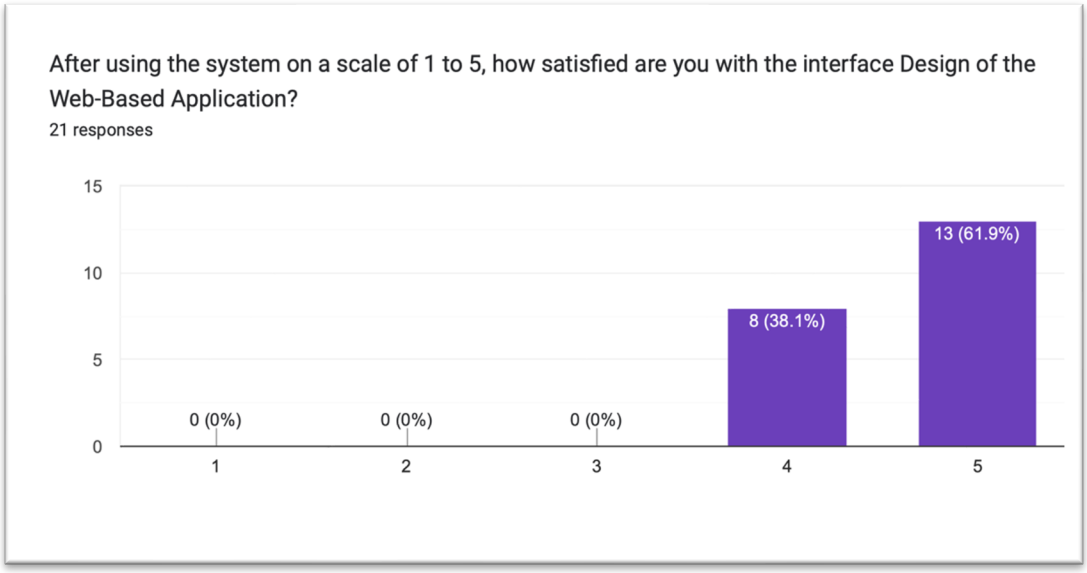

Figure 49 Job-Seekers Satisfaction with Design

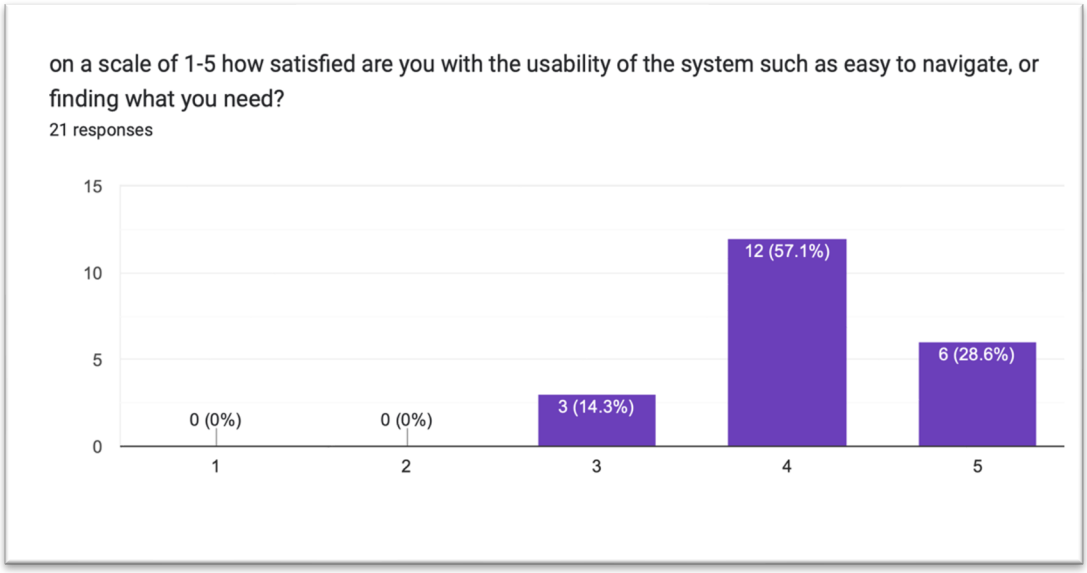

Figure 50 Job-Seekers Satisfaction with navigation

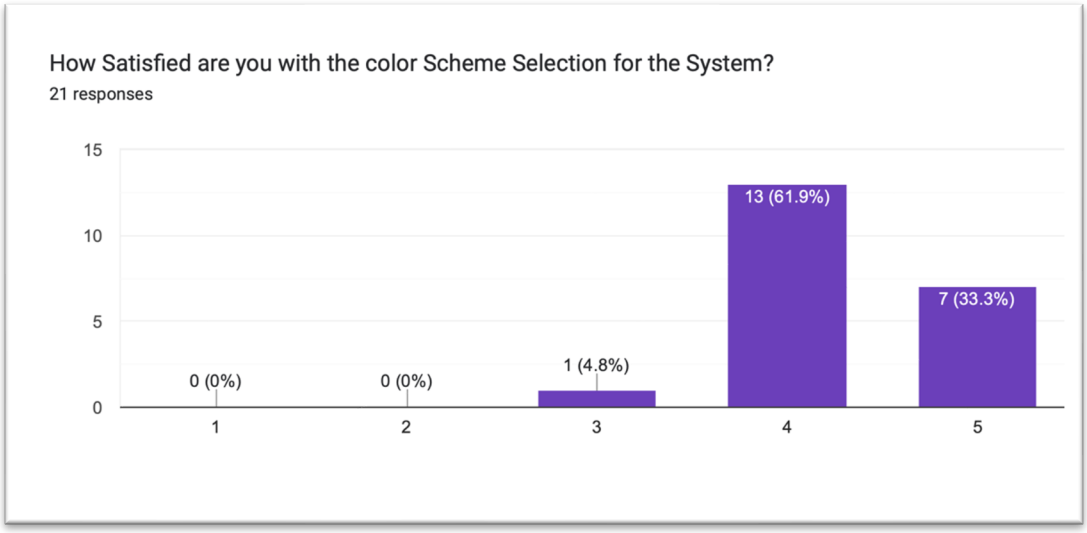

Figure 51 Job-Seekers Satisfaction with color Scheme

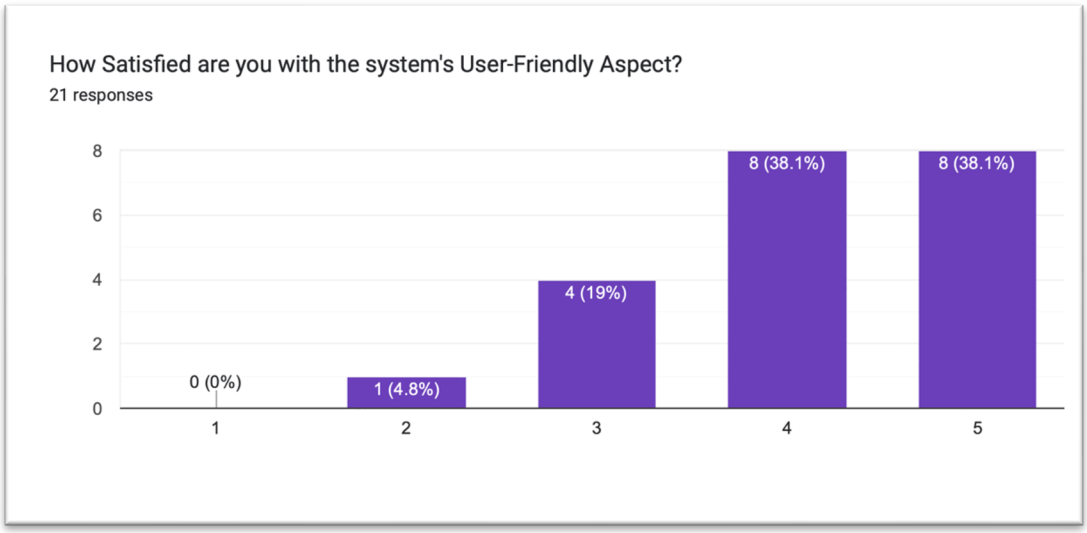

Figure 52 Job-Seekers Satisfaction with User friendliness

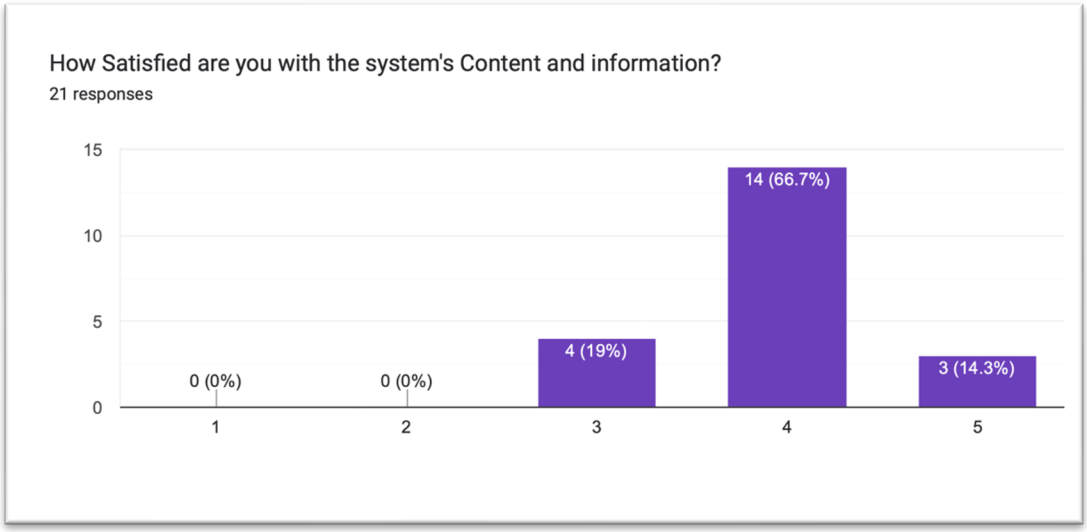

Figure 53 Job-Seekers Satisfaction with Content and information

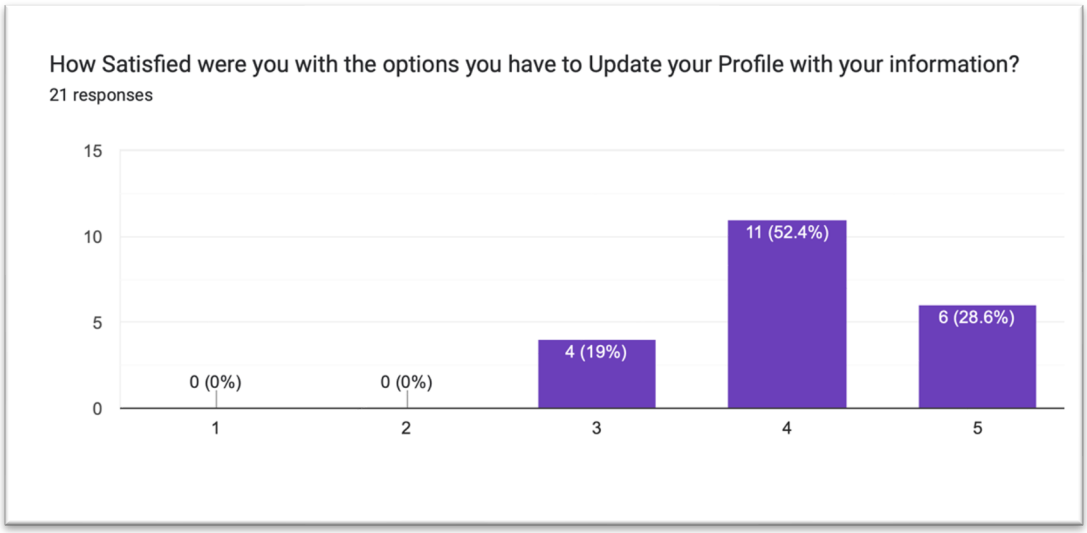

Figure 54 Job-Seekers Satisfaction with update profile feature

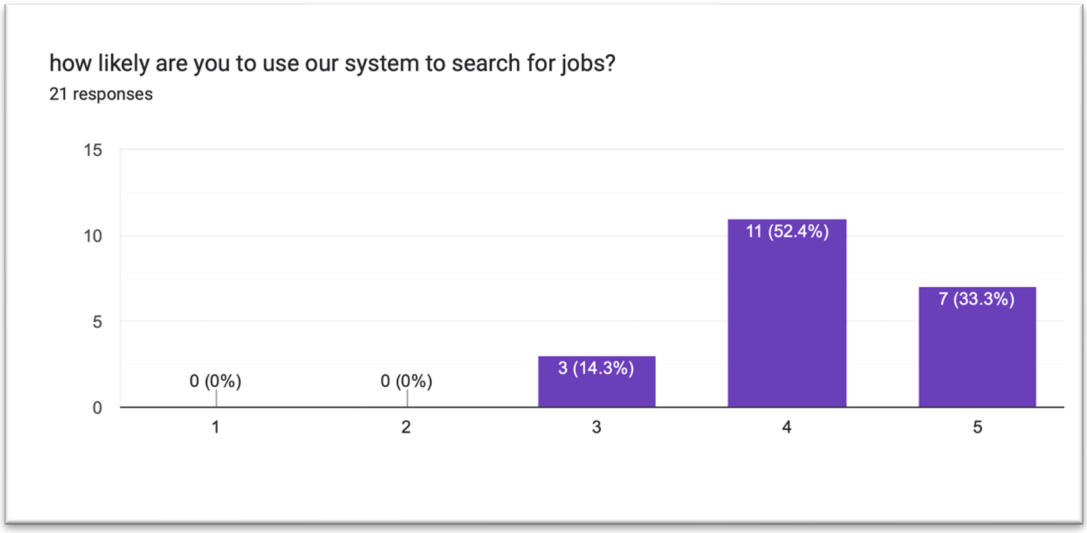

Figure 55 Job-Seekers Intention to use

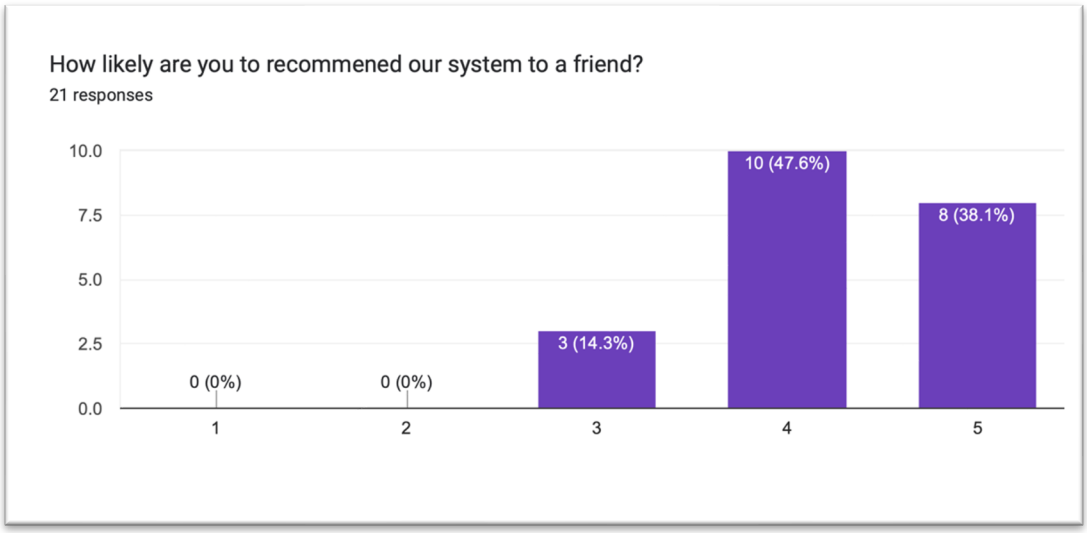

Figure 56 Job-Seekers intention to recommend to use

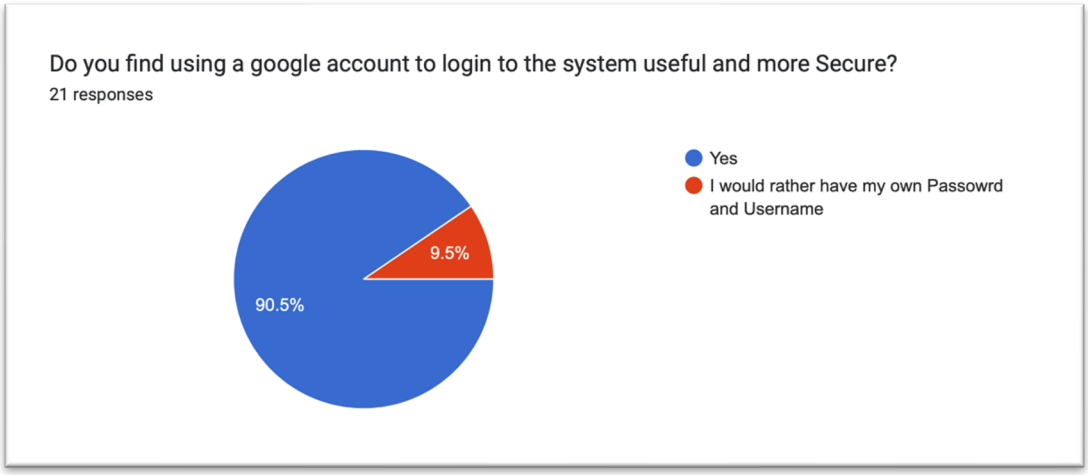

Figure 57 Job-Seekers Satisfaction with security of the system

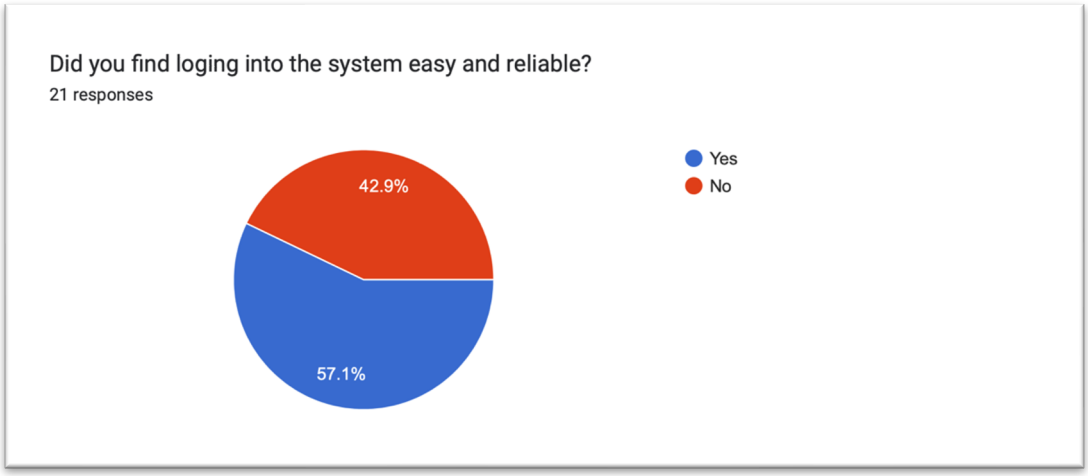

Figure 58 Job-Seekers Satisfaction with ease of use and reliability

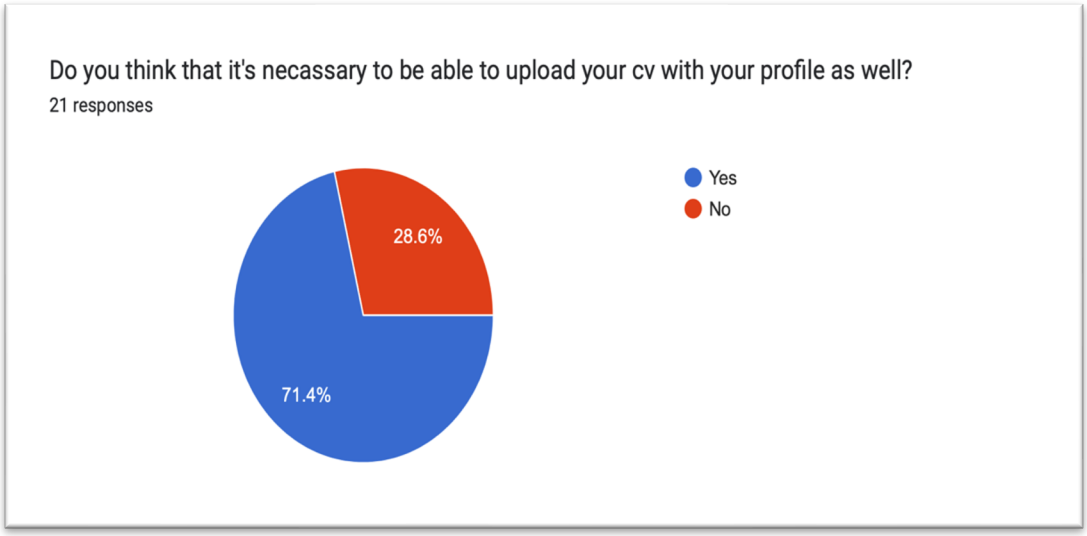

Figure 59 Job-Seekers Satisfaction with the option to include CVs

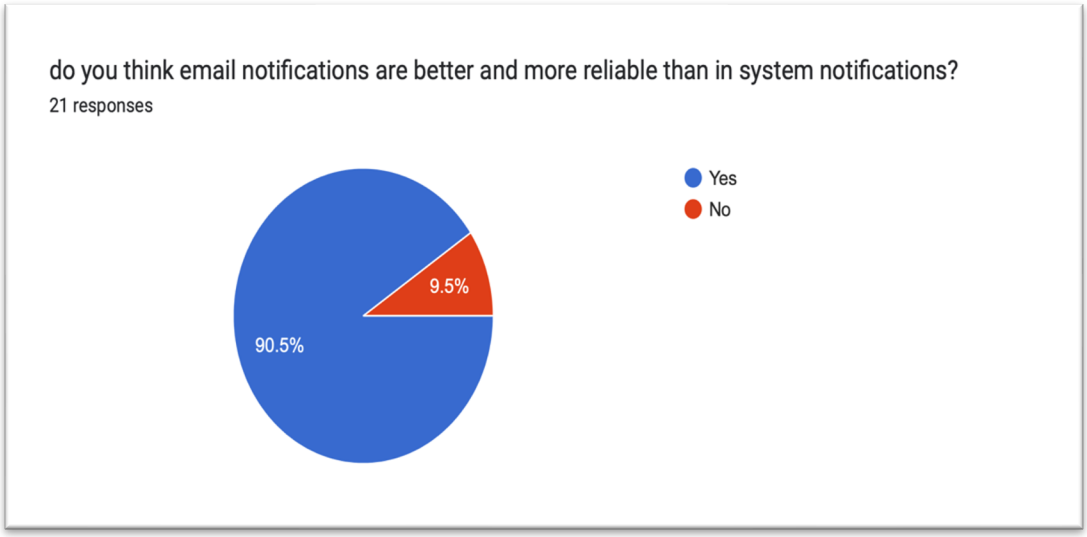

Figure 60 Job-Seekers Satisfaction with Notification Method

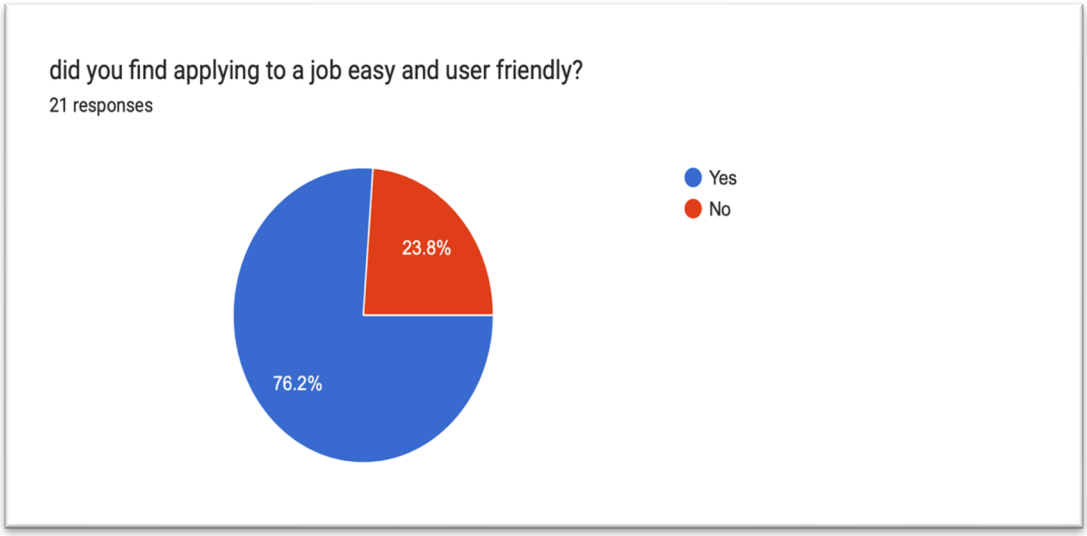

Figure 61 Job-Seekers Satisfaction with ease to use and friendliness