CVLyzer

Group Members

Hamza Asif (01-131182-012)

Habib Ur Reman (01-131182-049)

Supervisor: Dr. Raja M Suleman

A Final Year Project submitted to the Department of Software Engineering, Faculty of Engineering Sciences, Bahria University, Islamabad in the partial fulfillment for the award of degree in Bachelor of Software Engineering

July 2022

THESIS COMPLETION CERTIFICATE

Student Name: Habib Ur Rehman Enrolment No: 01-131182-049

Student Name: Hamza Asif Enrolment No: 01-131182-012

Programme of Study: Bachelor of Software Engineering

Project Title: CVLyzer

It is to certify that the above students' project has been completed to my satisfaction and to my belief, its standard is appropriate for submission for evaluation. I have also conducted plagiarism test of this thesis using HEC prescribed software and found similarity index at 9% that is within the permissible limit set by the HEC. I have also found the thesis in a format recognized by the department.

Supervisor's Signature: _____

Date: _____ Name: _____

CERTIFICATE OF ORIGINALITY

This is certified that the intellectual contents of the project CVLyzer are the product of my/our own work except, as cited properly and accurately in the acknowledgements and references, the material taken from such sources as research journals, books, internet, etc. solely to support, elaborate, compare, extend and/or implement the earlier work. Further, this work has not been submitted by me/us previously for any degree, nor it shall be submitted by me/us in the future for obtaining any degree from this University, or any other university or institution. The incorrectness of this information, if proved at any stage, shall authorities the University to cancel my/our degree.

Name of the Student: Habib Ur Rehman

Signature: _____ Date: _____

Name of the Student: Hamza Asif

Signature: _____ Date: _____

Abstract

With the accomplishment of digital recruitment, now it is easier to find a job and apply for it. At the same time organizations tends to receive a high volume of resumes for each job posting. Managing and processing these resumes to filter a suitable candidate is a time-consuming and costly task for Human Resource department. On average single hiring takes 30 to 40 days with an above 4000$ budget. Resumes contain information in many formats. CVLyzer helps the recruiters by providing automation of ranking resumes according to job descriptions in a flash of time without any biasness. The system also generates a report of ranking explanations for every individual. Besides that, CVLyzer also assist the job seeker by providing a tailored review of the resume with suggestions for improvements. The project is mainly based on Natural Language Processing a subfield of Artificial Intelligence. Information retrieval is done using named entity recognition. The model is trained using Spacy which uses word embeddings for its NER model which is multilayer CNN. The system also contains some rule-based models and text processors to extract data

Keywords: Artificial Intelligence, CNN, Information retrieval, Natural Language Processing, Named entity recognition, Spacy, Word embeddings

Dedication

We dedicated this project to our family and my supervisor. A special feeling of gratitude for our loving parent. We also dedicated this project to our friends who supported us throughout the process We will always appreciate all for what they have done.

Acknowledgments

First of all, we would like to express our greatest gratitude to Mr. Raja Suleman. This project would not have been possible without his help and active participation in every step of the process. Every time we attend his meeting, we feel motivated and encouraged. Without his support, this project would not have come about. The support of our friends was also a crucial success of the project. We are grateful for their constant support and help.

Table of Contents

List of Figures

Chapter 1
Introduction

The introduction covers the origin of the system. Problem statement and the objectives of the system that led us to build the CVLyzer. Introduction also contains the contribution that shows how it is beneficial for the society and what is new in the system. The chapter organization is also mentioned in the introduction

1.1. Motivation

Recruitment includes creating a wide pool of candidates and filtering the most qualified candidates for the job. This process requires on average 20 to 30 days and costs over $4000 according to the Society of Human Resource. These numbers will be reduced several times by automating the process. Auto recruitment increases productivity and quality. There is also a factor of biases in manual recruitment. Auto recruitment provides bias-free hiring and provides more analytics and reporting for a better experience. The time-saving capability of the auto recruitment systems is the key motivation. Auto recruitment systems perform lengthy and tedious manual tasks in a flash of time with more accuracy.

1.2. Problem statement

Currently, the hiring process in the industry is carried out by humans who spend time scanning resumes or other online profiles. This process is lengthy because of human limitations because the hiring requires a lo dedicated time. Biasness is also a human limitation that affects hiring. These factors can lead the organization to drop the better-fit candidates for the job.

Automated recruitment helps to minimize these factors and generates results faster. Automated recruitment systems are already available but as the technology is changing day by day these systems become obsolete. So, creating a state-of-art system requires the usage of more futuristic technology.

1.3. Objectives

The system provides service to both candidates and recruiters:

1. The system provides a tailored review of the resume to the candidate to fill the gaps in the resume and get more interview calls.
2. The system provides the sorted candidates against the job description to the recruiter with a detailed analysis and explanation.

1.4. Main contributions

Recruiters spend a lot of time sorting the candidates according to the job description. Other systems are providing similar services but they are not providing details of generated reports. Our system provides the sorted list of candidates against the job description with the sort report and a more detailed analysis for each candidate. Recruiters are allowed to build their evaluation framework so that the system uses this framework to sort the candidates.

Reducing time to hire and ranking more qualified candidates without any biases can lead the organizations to have more completive employees in less time. Candidates spend time improving their CVs. Our system provides a tailed review by passing the resume through an evaluation framework and provides a suggestion to improve the resume and get more interview calls.

Now organizations hire more qualified candidates in less time and for candidates now it is easy to improve their resumes according to the suggestions.

1.5. Report organization

Chapter 1 discusses an introduction of the document and the system

Chapter 2 discusses the background of the system.

Chapter 2 focuses on detailed system requirements.

Chapter 4 focuses on the design of the system.

Chapter 5 discusses the implementation and development of the system.

Chapter 6 consists of a discussion related to the testing of the system

Chapter 7 concludes the thesis by summarizing different aspects of the work.

Chapter 2
Background Study/Literature Review

Nowadays, the recruitment process implies searching huge amounts of usually unstructured documents. Resumes contain a lot of important information but they may differ in format. Therefore, it is difficult to extract data from resumes automatically.
This chapter discusses the literature review and background study of the system. A review of the literature is important to a successful system as it reveals problems with existing systems. This also helps in finding the best approach to achieve the goal. This chapter also involves the analysis of existing work.

2.1. Approaches

2.1.1. Named Entity Recognition based Resume Parser

Named Entity Recognition based Resume Parser and Summarizer [1] by Narendra G on how to parser the resume, summarize the content, and rank the resumes. The proposed system only works for skills however there are other important entities in the resume. Their experiment score is also not that good because of less amount of data and also the variety of the data is important. Spacy is used to train the model spacy produce good results depending upon the training data.

2.1.2. Semi-supervised deep learning-based named entity recognition

Semi-supervised deep learning-based named entity recognition model [2] by Bodhvi Gaur, Gurpreet Singh Saluja1, Hamsa Bharathi Sivakumar, and Sanjay Singh on parsing the education section of the resume. The approach used in this paper results in 92.06% of accuracy with a 73.28 f1 score which is good but the system is only to parser education section (degrees and institution names) of the resume. As there is limited work on resume parsing that’s why there is an absence of a larger and standard dataset.

2.1.3. Resume Ranking based on Job Description

Resume Ranking based on Job Description [3] by Dr.K. Satheesh uses the Spacy NER model. In this paper, spacy is used to train the information extraction or NER model but they worked on only four entities location, organization, miscellaneous, and person. The accuracy score is good but the system is not reliable as it missed most of the important information in the resume that is essential for ranking like skills, education, certifications, etc.

2.1.4. Resume Parser with Natural Language Processing

Resume Parser with Natural Language Processing [4] by Satyaki Sanyal. In this paper, information is retrieved using different analyses such as lexical, syntactic, and semantic analysis. These results are good and they extract almost all entities but this method is not generalized. This will not work as well for different formats.

2.1.5. A Low-Cost Named Entity Recognition Research Based on Active Learning

The paper [5] shows the active learning process of named entity recognisers at low cost. Their results are also good but the active learning require CPU as it is CPU intensive task. There is another paper [6] that shows the recent trends in the NER.

Peng sun has a paper [7] that shows the overview of the named entity recognition. It gives a good idea of developing custom NER.
A two step information extraction algorithm from resume is also proposed by Jie Chen [8] provides a new algorithm of extracting data and their results are also good.

2.2. Key Concepts

2.2.1. Natural language processing

Natural language processing is the subfield of artificial intelligence that deals with the interaction of computers and the human language. It takes the real-world input and processes it in a way that computers can understand. It can be used to analyse a large amount of textual data like social media.

2.2.2. Information retrieval

Information retrieval in the terms of NLP is the process of organization, storage, retrieval, and evaluation of the information from unstructured documents, especially through the use of computerized systems. A good information retrieval system extracts all relevant information from the document.

2.2.3. Named entity recognition

NER (Named entity recognition) is a subtask of information retrieval that extracts and seeks to locate and classify named entities mentioned in the unstructured data into pre-defined categories like email, name, country, time period, date, etc. It first detects the named entity and then categorizes it into a predefined class. Models are trained by providing data with entities tagged after that these models are used to extract information.

Chapter 3

System Requirements

3.1. Use Case Diagrams

3.1.1. Accounts

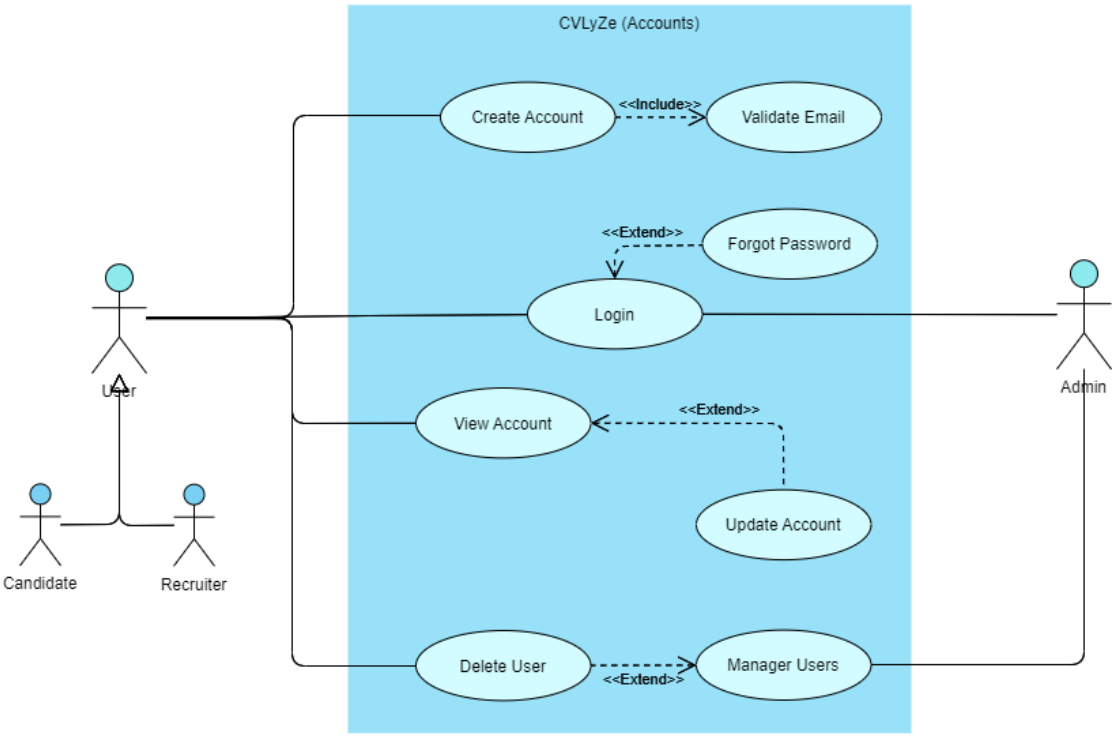

Figure 1 Accounts use case

3.1.2. Resume Report

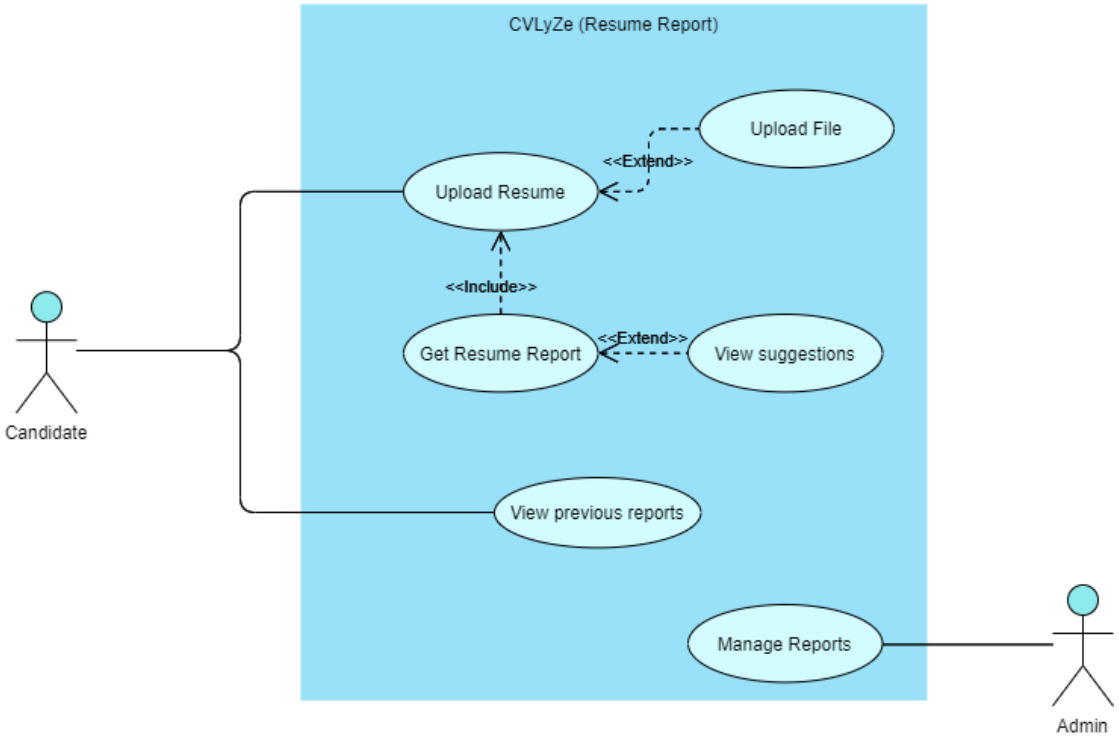

Figure 2 Resume report use case

3.1.3. Sort Report

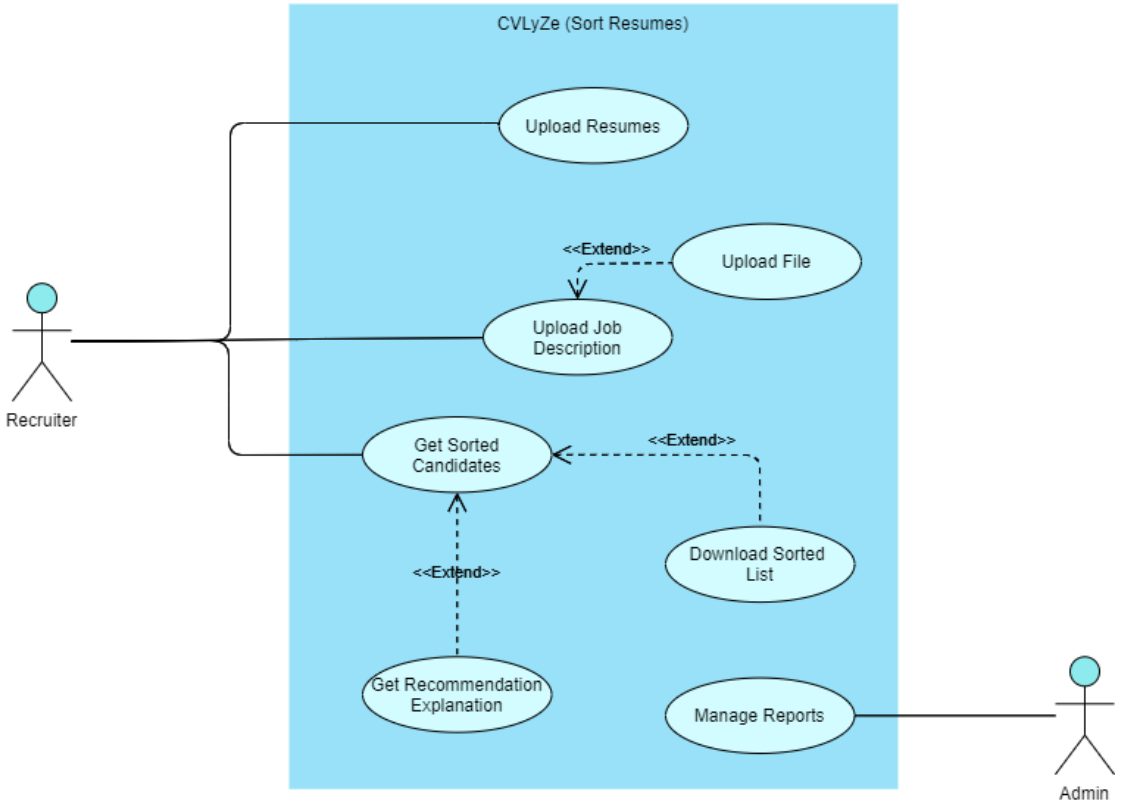

Figure 3 Sort report use case

3.2. Functional Requirements presented as Use Case Descriptions

3.2.1. Create account

<table>
<tr><td>Use Case ID</td><td colspan="2">A1</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Create Account</td></tr>
<tr><td>Actor(s)</td><td colspan="2">User (Candidate, Recruiter)</td></tr>
<tr><td>Pre-Conditions</td><td colspan="2">User wants to register and use the system</td></tr>
<tr><td>Priority</td><td colspan="2">High</td></tr>
<tr><td>Basic Flows</td><td colspan="2">The user clicks on create account button if he is not registered and he will be directed to signup page</td></tr>
<tr><td colspan="2">Actor Actions</td><td>System Response</td></tr>
<tr><td colspan="2">This use case starts when the User accesses the system feature (like uploading CV’s) that will enable the user to create an account by entering his information that will be maintained in the User’s account.
1. The User enters the required User Account information values and requests that the system saves the entered values.</td><td>2. The system validates the entered User Account information.
The values for the User Account information are stored in the User’s account. The system notifies the User (through email) that the account has been created.</td></tr>
<tr><td colspan="3">Alternate Course of Action</td></tr>
<tr><td>Title</td><td colspan="2">User Enters Invalid User Account Information</td></tr>
<tr><td>Actor Action</td><td colspan="2">System Response</td></tr>
</table>

	If during Create Account session, the system determines that the User has entered any invalid User Account information, the following occurs: 1. The system describes which entered data was invalid and presents the User with suggestions for entering valid data. 2. The system prompts the User to re-enter the invalid information. 3. The User re-enters the information and the system revalidates it. 4. If valid information is entered, the User Account Information is stored. 5. If invalid information is entered, the Entered Information is Invalid alternative flow is executed again. This continues until the User enters valid information. Invalid User Account information: - Missing information items - Not well-formed e-mail address Offending words in any part of the User Account information

Table 1 create account use case description

3.2.2. Login

Use Case ID	A2	
Use Case Name	Login	
Actor(s)	User (Candidate, Recruiter, Admin)	
Pre-Conditions	User must be registered	
Priority	High	
Basic Flows	The Create Account use case allows the User to login and become a regular User.	
Actor Actions		System Response
This use case starts when the User accesses the sign in feature of the system.		

2. The User enters his username and password 4. The User is signed in. The system displays a message indicating that the user is signed in.	1. The system prompts the User for his username and password. 3. The system validates the entered information, making sure that the entered username and password are valid for one user account in the system, and that the required password is entered for the entered username.
Alternate Course of Action	
Title	New User
Actor Action	System Response
	If the User does not have an account, the System will give the User the opportunity to create an account. See the Create Account use case. Once the account is created, the User is redirected to login page to sign in.
Title	User Fails Authentication
Actor Action	System Response
If the User entered an invalid username and/or password, the following occurs:	1. The system describes the reasons why the User failed authentication. 2. The system presents the User with suggestions for changes necessary to allow the User to pass authentication. 3. The system prompts the User to re-enter the valid information. 4. The Basic Flow continues where the User enters new information. 5. On success, the User is authenticated, and the system displays all features available for the role the user is associated with as defined in his/her user account.

Table 2 Login use case description

3.2.3. Update account

Use Case ID	A3	
Use Case Name	Update Account	
Actor(s)	User (candidate, Recruiter)	
Pre-Conditions	The User must be signed in before the User can edit his account	
Priority	High	
Basic Flows	User opens his account; the user makes changes to his information, and he saves his information. The system updates his account	
Actor Actions		System Response
This use case starts when the User accesses the feature (updating feature) that enables him to update the information that is maintained in the User's account.		
2. The User enters the desired User Account information values and requests that the system saves the entered values.		1. The system displays the User Account information which is stored for the User. 3. The system validates the entered User Account information. 4. The values for the User Account information are stored in the User's account if are validated. Else, the user will be notified through alerts to correct a specific field. **5.** The use case ends
Alternate Course of Action		
Title	User Enters Invalid User Account Information	
Actor Action	System Response	

	If during updating Account, the system determines that the User entered invalid User Account information, the following occurs: 1. The system describes which entered data was invalid and presents the User with suggestions for entering valid data. 2. The system prompts the User to re-enter the invalid information. 3. The User re-enters the information and the system revalidates it. 4. If valid information is entered, the User Account Information is stored. 5. If invalid information is entered, the Entered Information is Invalid alternative flow is executed again. This continues until the User enters valid information. Invalid User Account information: - Missing information items - Not well-formed e-mail address Offending words in any part of the User Account information

Table 3 Update account use case description

3.2.4. Delete account

Use Case ID	A4	
Use Case Name	Delete Account	
Actor(s)	User (candidate, Recruiter)	
Pre-Conditions	The User must be signed in before the User can delete his account	
Priority	High	
Basic Flows	User opens his account; the user clicks on delete account button in his profile. The system validates him by taking his password and delete his account.	
Actor Actions		System Response
This use case starts when the User accesses the feature (deletion feature) that enables him to delete his account.		

1. The user open's his profile. 3. The User requests that the system deletes the entered values. 5. User is asked to enter his password.	2. The system displays the User Account information which is stored for the User. 4. The system validates the entered User Account password. 6. If validated, the system deletes the user's account and notifies the User that the account has been updated.
Alternate Course Of Action	
Title	User Enters Invalid User Account Password
Actor Action	System Response
4. The User re-enters the password and the system revalidates it.	If during delete Account, the system determines that the User entered invalid User Account password, the following occurs: 1. The system describes the user that his password is invalid. 2. The system prompts the User to re-enter the invalid password. 5. If valid password is entered, the User Account will be deleted. 6. If invalid password is entered, the Entered password is Invalid alternative flow is executed again. This continues until the User enters valid password. Invalid User Account information: Not matching password which the user is having

Table 4 Delete account use case description

3.2.5. Upload resume

<table>
<tr><td>Use Case ID</td><td colspan="2">RR1</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Upload Resume</td></tr>
<tr><td>Actor(s)</td><td colspan="2">User (candidate)</td></tr>
<tr><td>Pre-Conditions</td><td colspan="2">The User must be signed in before the User can upload the resume.</td></tr>
<tr><td>Priority</td><td colspan="2">High</td></tr>
<tr><td>Basic Flows</td><td colspan="2">User opens the site, the user clicks on upload resume button, system take him to the page where the user can upload his resume.</td></tr>
<tr><td colspan="2">Actor Actions</td><td>System Response</td></tr>
<tr><td colspan="3">This use case starts when the User accesses the feature (uploading resume feature) that enables him to upload the resume.</td></tr>
<tr><td colspan="2">1. The user can adopt the following way to upload the resume.
a) The user can upload file.
• The User selects a file/directory on the Local File List Window
2. The user clicks the resume upload button.
3. The User then uploads the resume on the system.
4. The user can only upload one resume at a time.</td><td></td></tr>
<tr><td colspan="3">Alternate Course of Action</td></tr>
<tr><td>Title</td><td colspan="2">Upload Resume fail</td></tr>
<tr><td>Actor Action</td><td colspan="2">System Response</td></tr>
<tr><td>3. The User reuploads the resume.</td><td colspan="2">If during upload resume, the resume was failed to upload, the system determines it, and the following occurs:
1. The system describes the user that the resume was not uploaded.
2. The system prompts the User to re-upload the resume.
4. If resume is uploaded successfully, the user will be continued. Otherwise, he will repeat the steps again.</td></tr>
</table>

Table 5 Upload resume use case description

3.2.6. Resume report

Use Case ID	**RR2**
Use Case Name	Get Resume Report
Actor(s)	User (candidate)
Pre-Conditions	The user must be signed in and the resume should be uploaded successfully in order to get the resume report.
Priority	High
Basic Flows	This use case allows the User to get the resume report after successful resume upload.
Actor Actions	**System Response**
This use case starts after successful upload of resume	
1. The user gets the report of his uploaded resume. **2.** The use case ends	
Alternate Course Of Action	
Title	None

Table 6 resume report use case description

3.2.7. Resume sorting

Use Case ID	SR1
Use Case Name	Upload Resumes
Actor(s)	User (recruiter)
Pre-Conditions	The User must be signed in before the User can upload resumes.
Priority	High
Basic Flows	The user clicks on upload resume button, system take him to the page where the user can upload resume(s).
Actor Actions	System Response
This use case starts when the User accesses the feature (uploading resumes feature) that enables him to upload resumes.	

1. The user clicks the resume uploads button. 2. User then uploads the resume on the system. 3. The user can upload multiple resumes at a time.	
Alternate Course of Action	
Title	Upload Resume(s) fail
Actor Action	System Response
3. The User re-uploads the resume.	If during upload resumes, the resume was failed to upload the system determines it and the following occurs: 1. The system describes the user that the resume was not uploaded successfully. 2. The system prompts the User to re-upload the resume. 4. If resume is uploaded successfully, the user will be continued. Otherwise, he will repeat the steps again.

Table 7 Resume sorting use case description

3.2.8. Upload job description

Use Case ID	SR2
Use Case Name	Upload Job description
Actor(s)	User (recruiter)
Pre-Conditions	The User must be signed in before the User can upload job description.
Priority	High
Basic Flows	The user clicks on upload job description button; system opens the files prompt from where the user can upload job description.
Actor Actions	System Response
This use case starts when the User accesses the feature (uploading job description feature) that enables him to upload the job description against the resume.	

1. The user can adopt the following way to upload the job description. **a)** The user can upload file. • The User selects a file/directory on the Local File List Window **2.** The user clicks the resume job description button. **3.** The User then uploads the job description on the system (from systems directories). **4.** The use case ends.	
Alternate Course Of Action	
Title	Upload descriptions fail
Actor Action	System Response
3. The User re-uploads the resume.	If during Upload Job description, if the job description was failed to upload, the system determines it, and the following occurs: 1. The system describes the user that the job description was not uploaded. 2. The system prompts the User to re-upload the job description. 4 If job description is uploaded successfully, the user will be continued. Otherwise, he will repeat the steps again.

Table 8 Upload job description use case description

3.3. Non-Functional Requirements

3.3.1. Portability

The system should be portable. That is, while running on one platform, the system could be easily used to run on another platform.

3.3.2. Performance

The system should be fast. That is, the system should be fast enough so that the users using it didn't get frustrated.

3.3.3. Availability

The system should be available all the time. That is, the system could be accessed by user 24/7.

3.3.4. User friendly

The system should be easily used by the users.

3.3.5. Reliability

The system should be reliable. That is, the ability of the system to behave consistently in a user-acceptable manner when operating with the environment.

3.3.6. Security

- Secure access to confidential data.
- Proper user authentication should be provided
- System should have different type of users and every particular user has access constraints

3.3.7. Privacy

Personal information of the data should not be disclosed to anyone

3.3.8. Scalability

The system should be scalable. That is, in case of increase of usage (i.e., if more resources are added) the system should still be able to meet its goals.

3.4. Interface Requirements

The system should be a web-based interface. Three types of users are allowed in the system, one is the candidate, second is the recruiter and third one is administrator. The system provides an elegant interface for the user (candidate & recruiter) and the admin can operate on the system, performing the required task such as:

- Before being able to use the system, a username and password should be entered in login screen by the user in order to be authenticated.
- Once the user is logged in, then based on roles the user will be redirected to his dashboard. That is,
    - If the user is admin, then user will be redirected to admin dashboard
    - If the user is candidate, then user will be redirected to candidate dashboard
    - If the user is recruiter, then user will be redirected to recruiter dashboard
- By clicking on “upload resume” button on home page user will be redirected to upload resume section (if user is logged in or else the user will be redirected to login page) where the user can upload resume(s).
    - If the user is a candidate, user can upload only one resume at a time.
    - If the user is a recruiter, then user can upload multiple resumes.
- Candidate should upload a single resume at a time, candidate will then get analysis report of that particular resume. The candidate can also view history where he can see the list of previous reports along with their score. Candidate can also download a specific file (which he uploads earlier) from the list and he can also see the analysis report of a specific previous report by click the “view report” button.
- Recruiter can upload multiple resumes at a time. He has to upload job description as well against those resumes to get the sorted resumes. After uploading, the recruiter will be redirected to results page where recruiter gets the list of sorted candidates. recruiter could download any resume from the list. The recruiter could visit any specific candidate report by clicking on the view details button. He will be redirected to report page of that particular candidate

- Admin can manage users. An admin can see how many users (both candidates and recruiters) are using the system. Within in that, admin can see how many analyses a candidate has done. An admin can check if the user is verified or not. Admin can also delete a user.
- At the end, if the user logout. User will be redirected to homepage.

3.5. Database Requirements

MySQL server is used as database for the system

3.6. Project Feasibility

3.6.1. Technical Feasibility

The technical feasibility of the system is that any device having an access to browser can use the application as the system is web-based application, the system should be available 24/7.

The required technical skills for the project are also present and will be sufficient for the completion of the project.

3.6.2. Operational Feasibility

The system will be easily accessible and easy to use for all kinds of users as it is based on web services that people are familiar with.

3.6.3. Legal & Ethical Feasibility

The project should be within legal and ethical boundaries and is legally and ethically feasible.

3.7. Conclusion

In this section of report, we have covered the various requirements of our system which are sufficient to create a feasible project.

Chapter 4

System Design

System design includes every aspect of the system design

4.1. Design Approach

The system design approach is Object Oriented. The system is divided into small modules that include both processes and data. The whole system is based on the concept of objects which makes it more flexible, reusable, and portable.

4.2. System Architecture

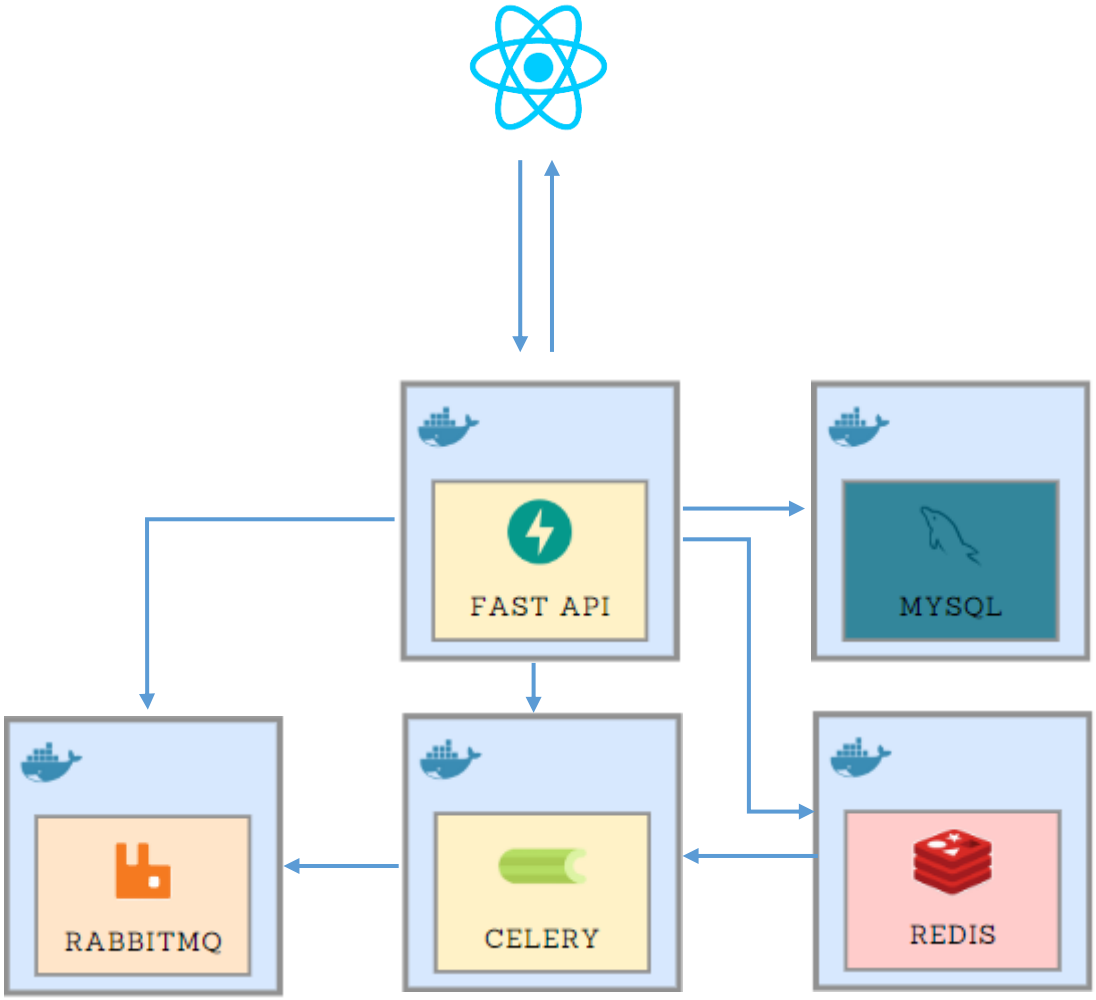

Figure 4 System architecture

4.2.1. React

React is an open-source front-end JavaScript library to build user interfaces. Our system UI is built using React and Redux. Redux is used to behave application consistently in different environments.

4.2.2. Fast API

Fast API is a modern and high-performance Python web framework for building APIs with stands Python hints. Our system's API is built on Fast API.

4.2.3. MySQL

MySQL is an open-source relational database management system. Our system's database is MySQL.

4.2.4. RabbitMQ

RabbitMQ is an open-source message broker that implemented advanced message queuing. It gives the common platform to send and receive messages. To maintain the tasks and their messages RabbitMq is used in our system.

4.2.5. Celery

Celery is an open-source asynchronous task queue that is based on distributed message passing. Clery is used in our system to handle tasks asynchronously

4.2.6. Redis

Redis is an in-memory data structure store, used as a distributed, in-memory key-value database, cache, and message broker, with optional durability. Redis is used as an in-memory database in our system

4.3. Logical Design

4.3.1. Class diagram

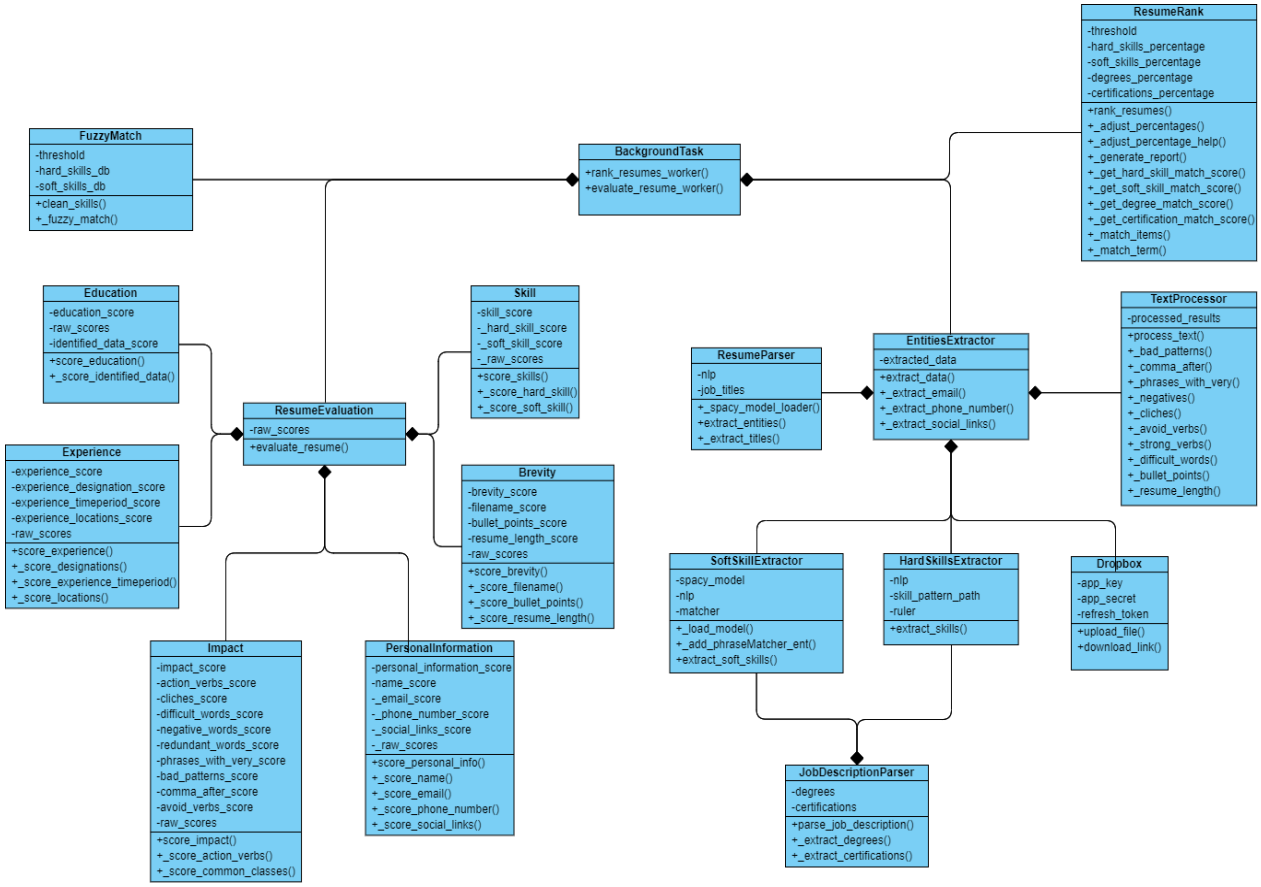

Figure 5 System class diagram

4.4. Dynamic View

4.4.1. Login

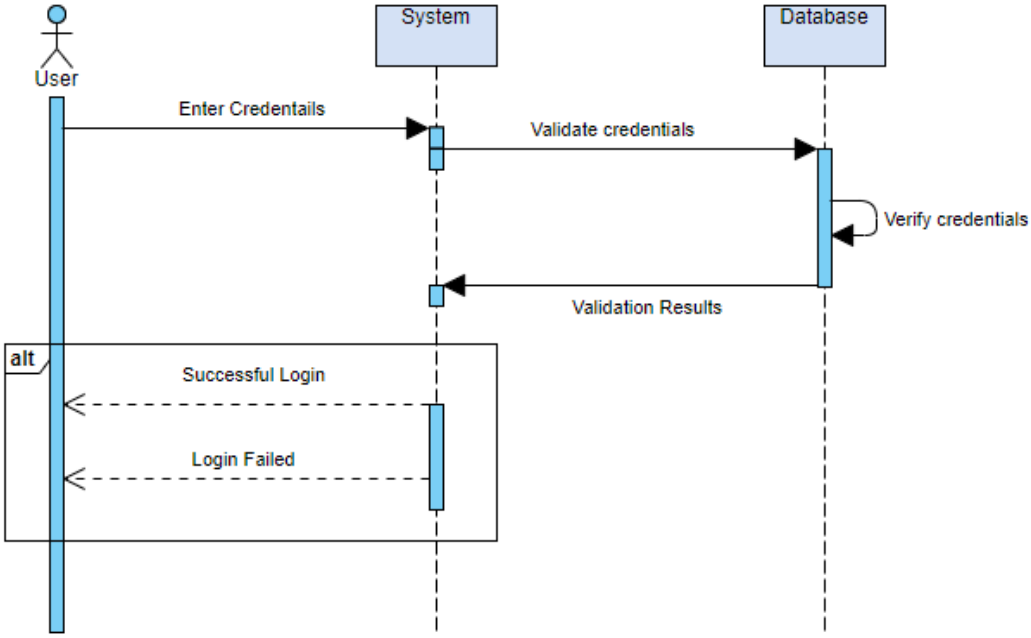

Figure 6 Login sequence diagram

4.4.2. Registration

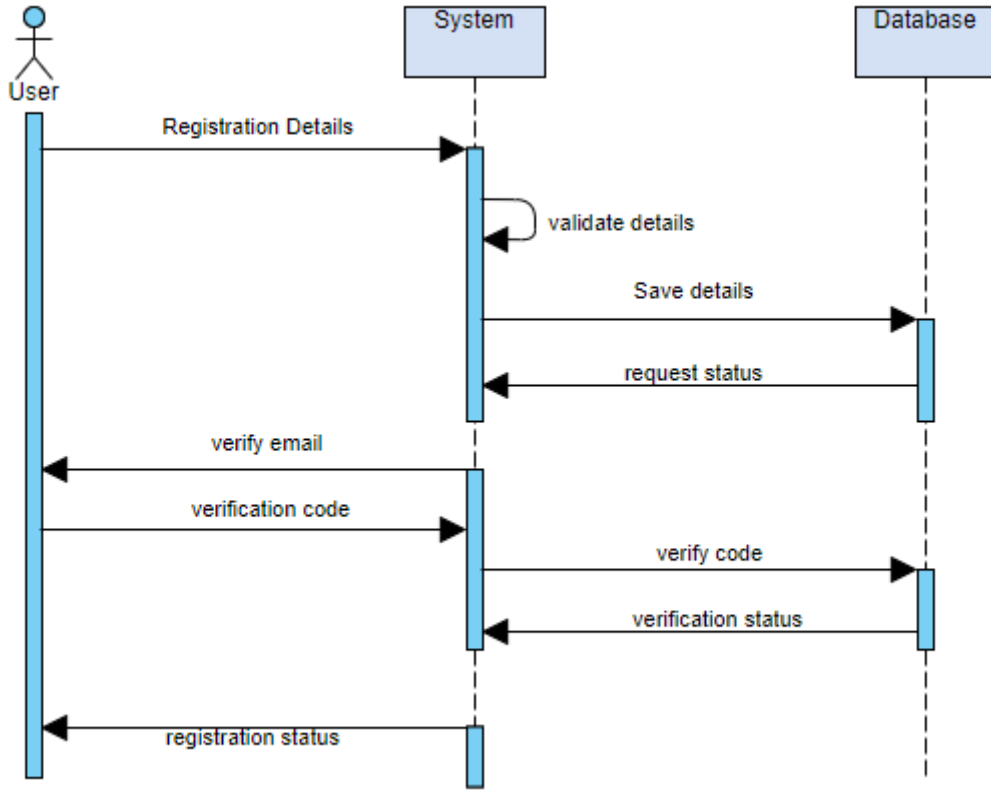

Figure 7 Registration sequence diagram

4.4.3. Resume analysis

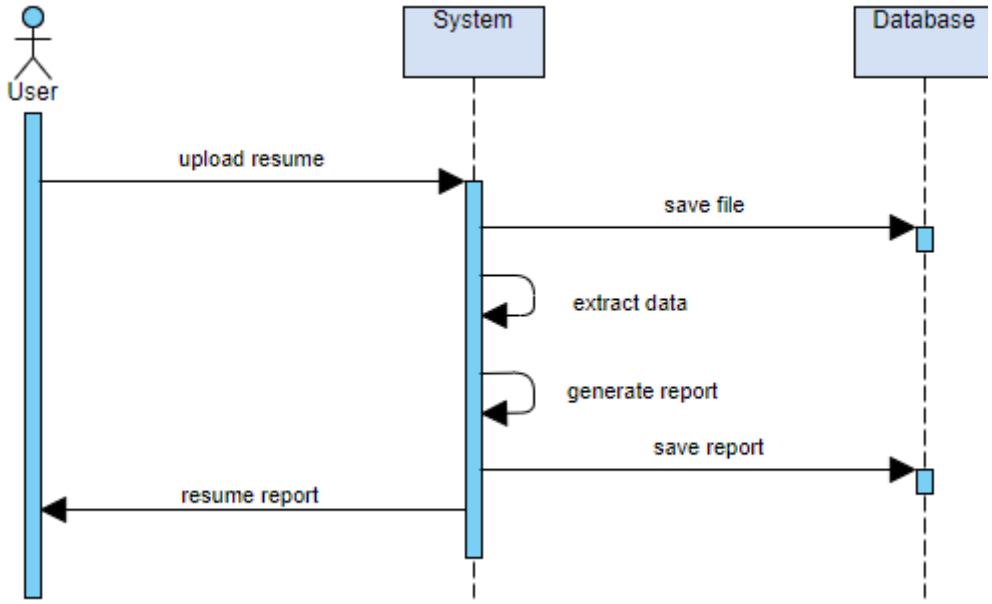

Figure 8 Resume analysis sequence diagram

4.4.4. Sort report

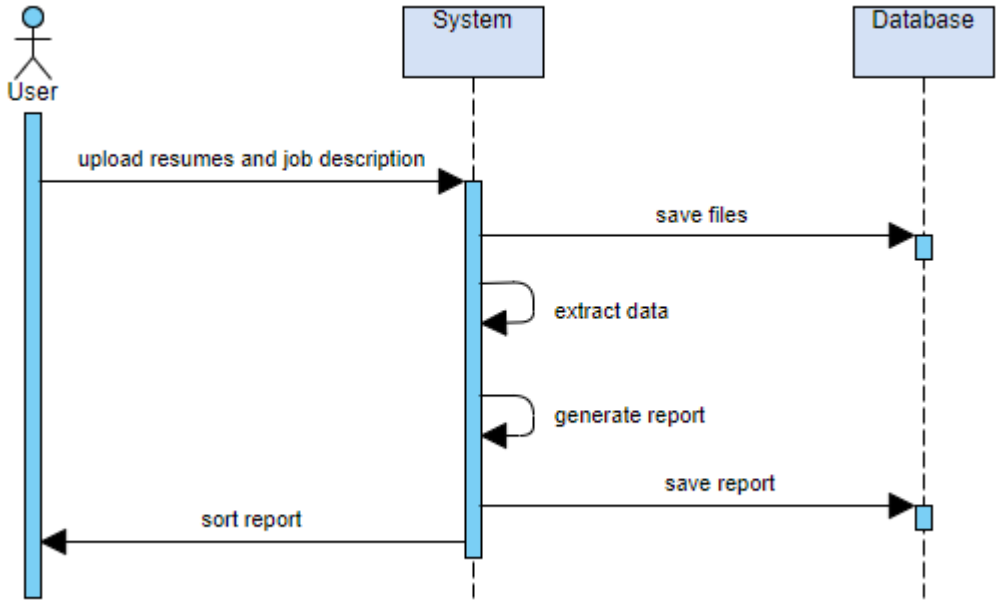

Figure 9 Sort report sequence diagram

4.5. Component Design

4.5.1. Deployment Diagram

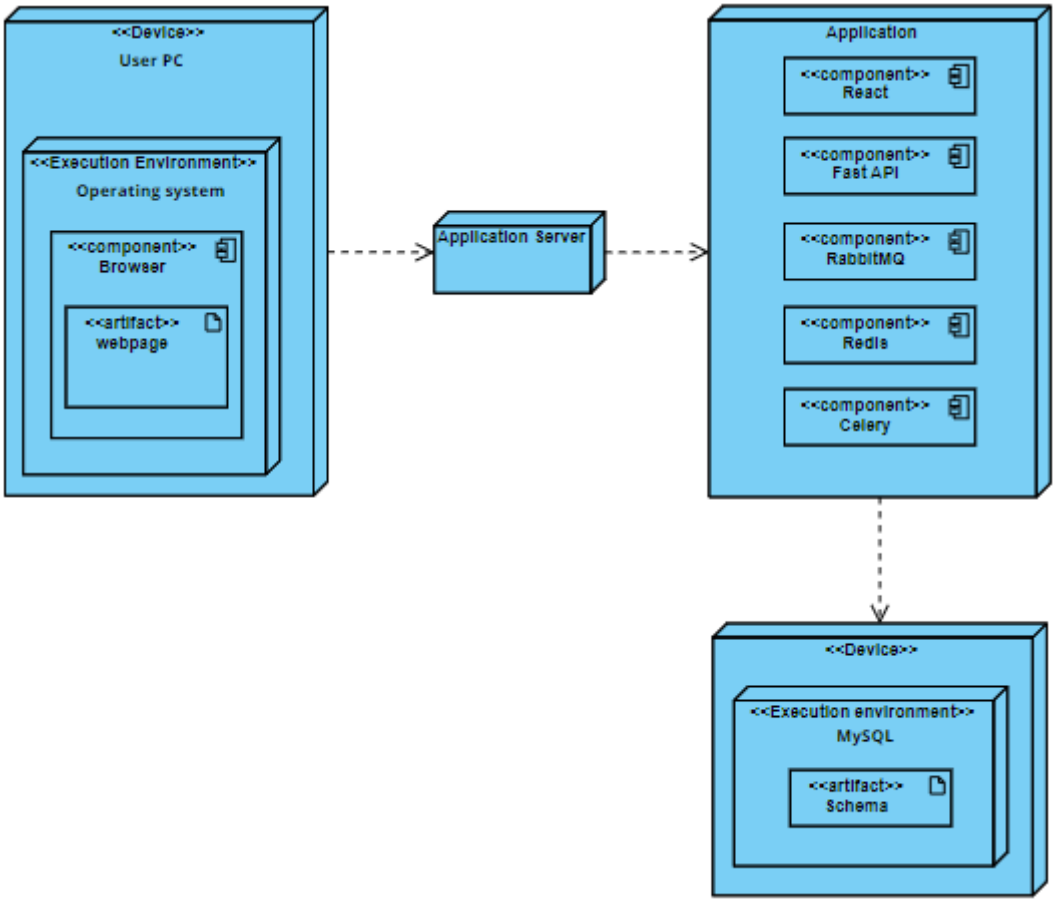

Figure 10 System deployment diagram

4.6. Data Models

4.6.1. Entity relationship diagram

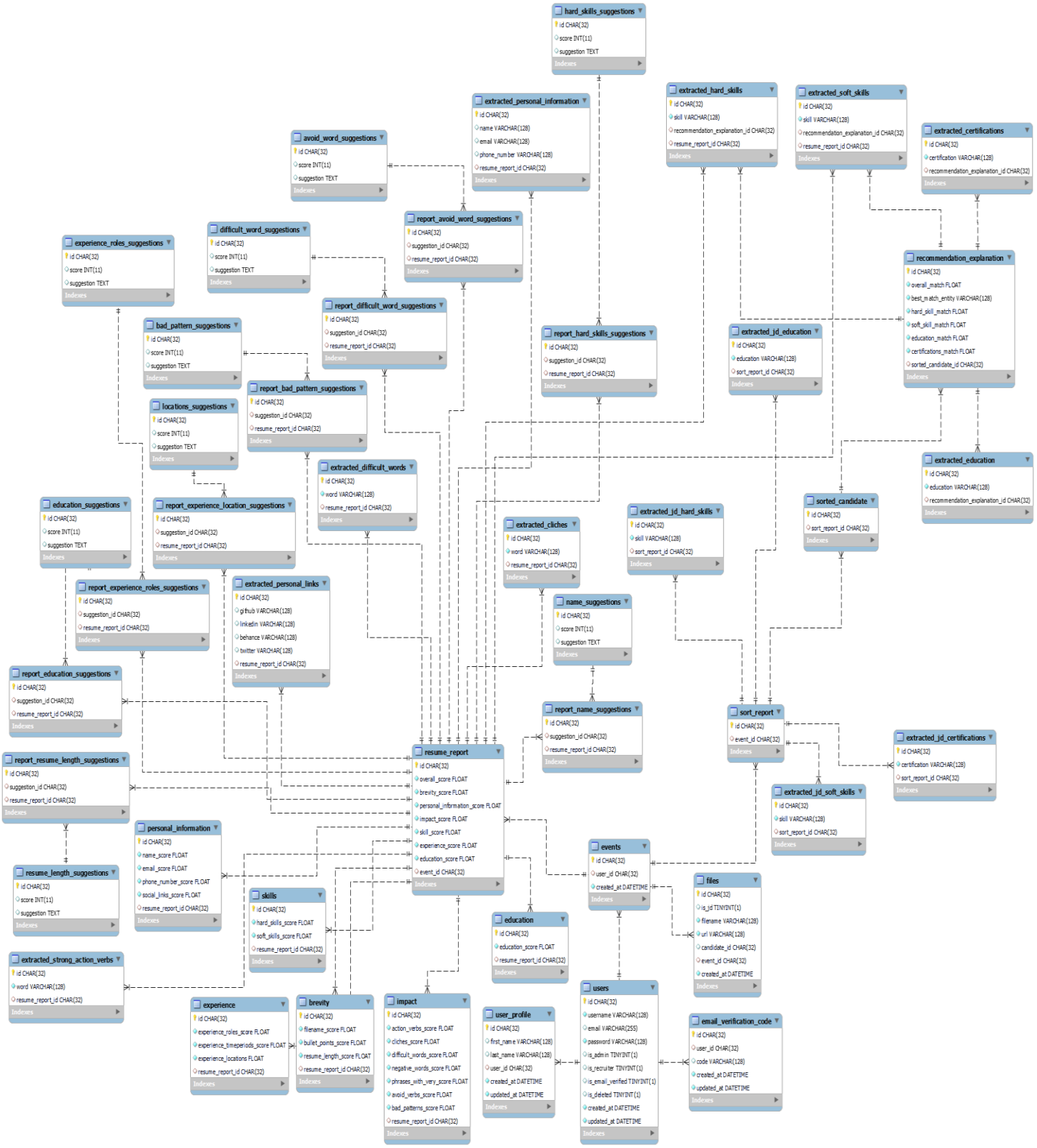

Figure 11 Entity relationship diagram

4.7. System Prototype

The system is built the evolutionary, the stepwise enhancement of prototype leads to the final user interface design.

Some protype design are given below:

4.7.1. Home page

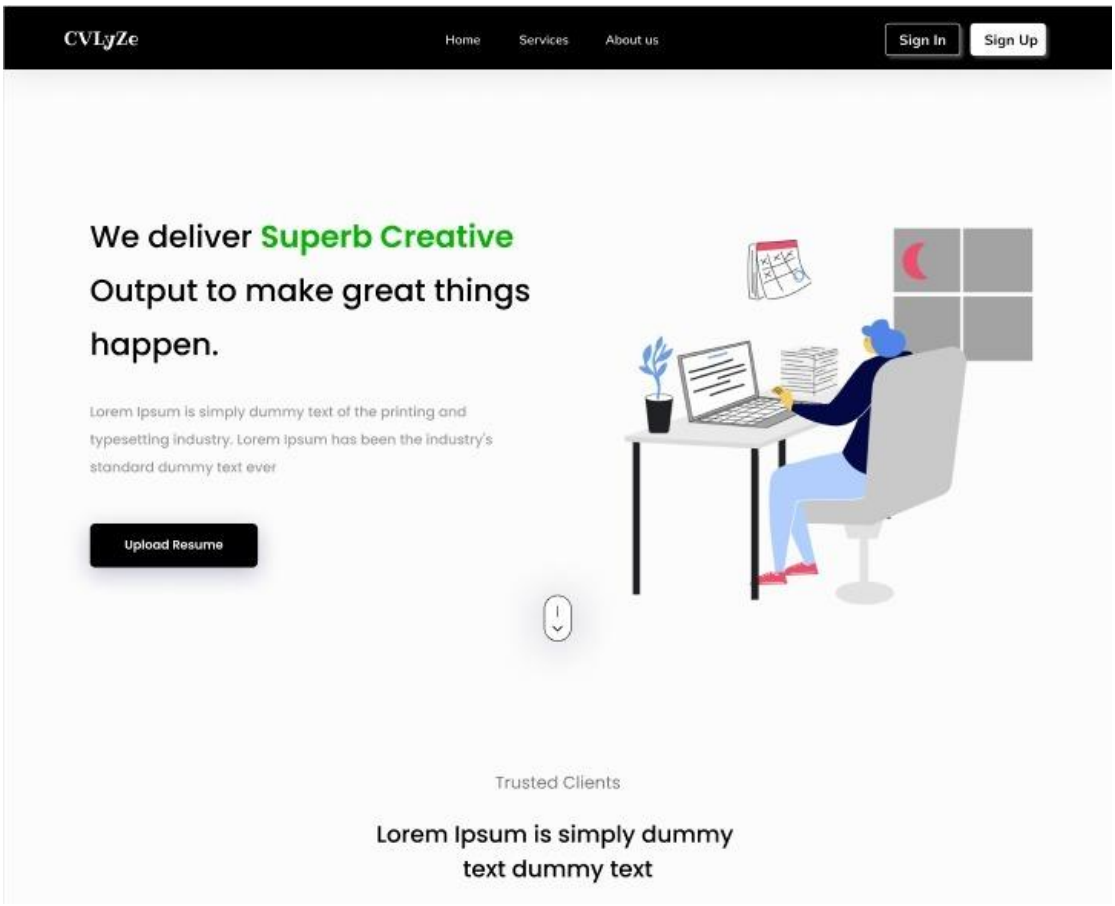

Figure 12 System homepage design (prototype)

4.7.2. Analysis report

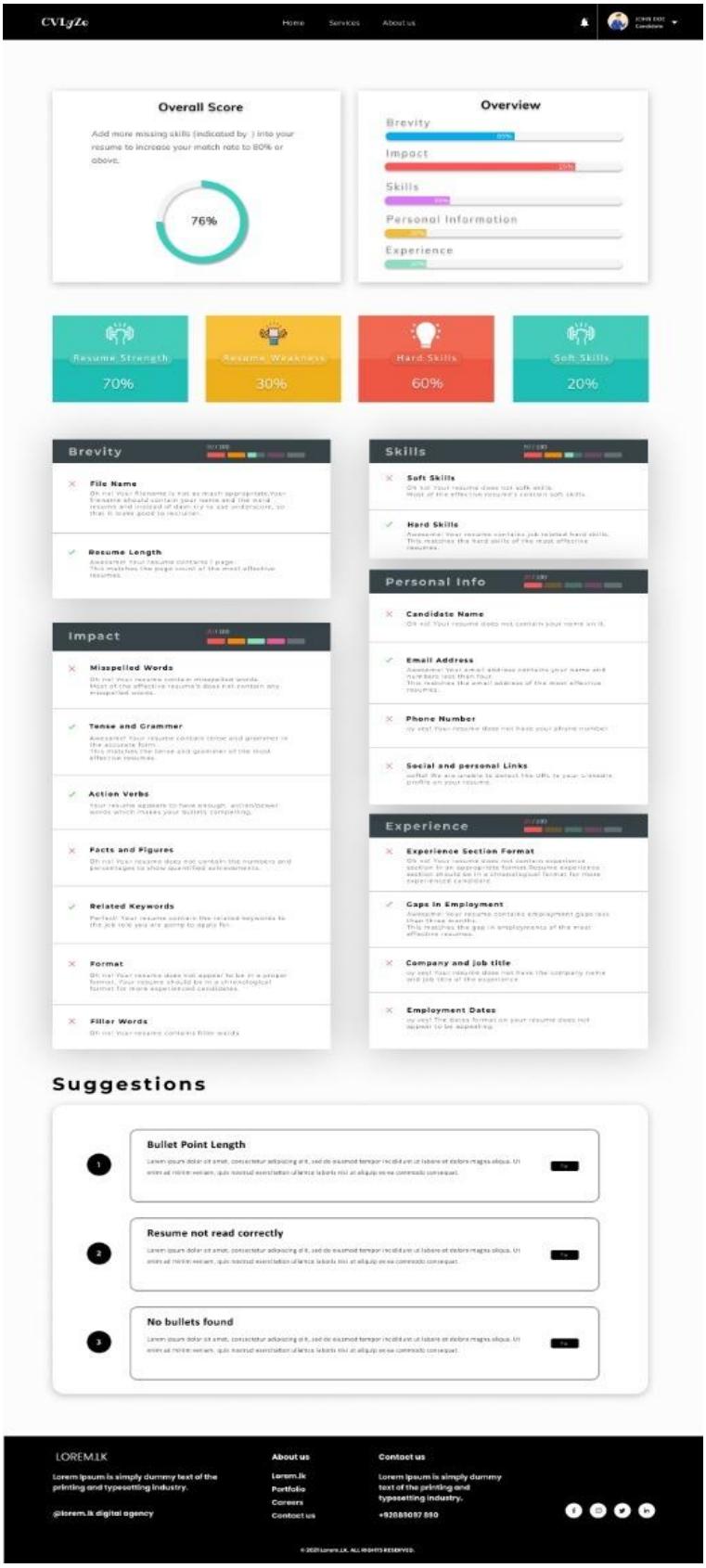

Figure 13 Analysis report UI (prototype)

4.7.3. Recruiter section

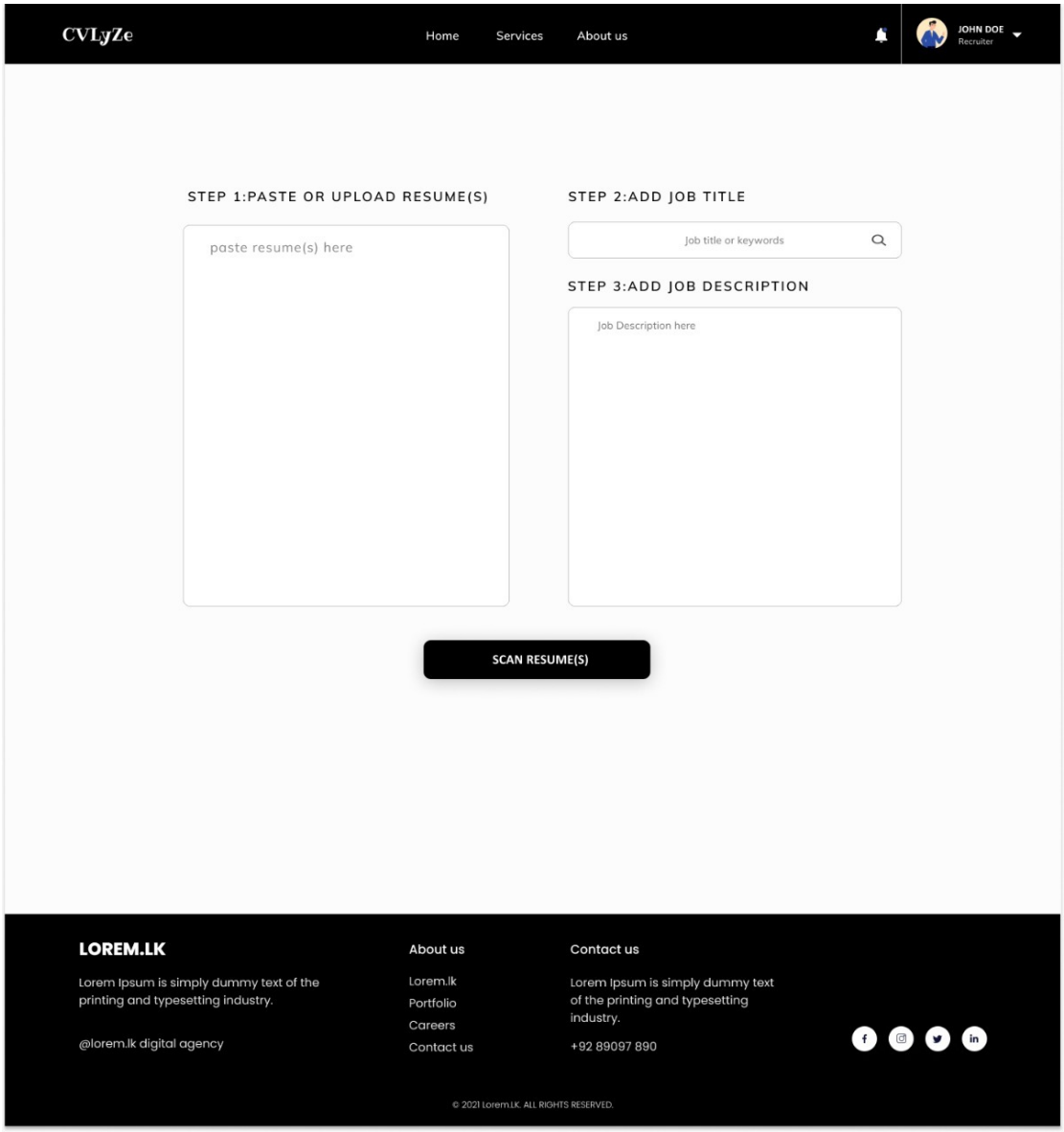

Figure 14 Recruiter section (prototype)

4.7.4. Sort report

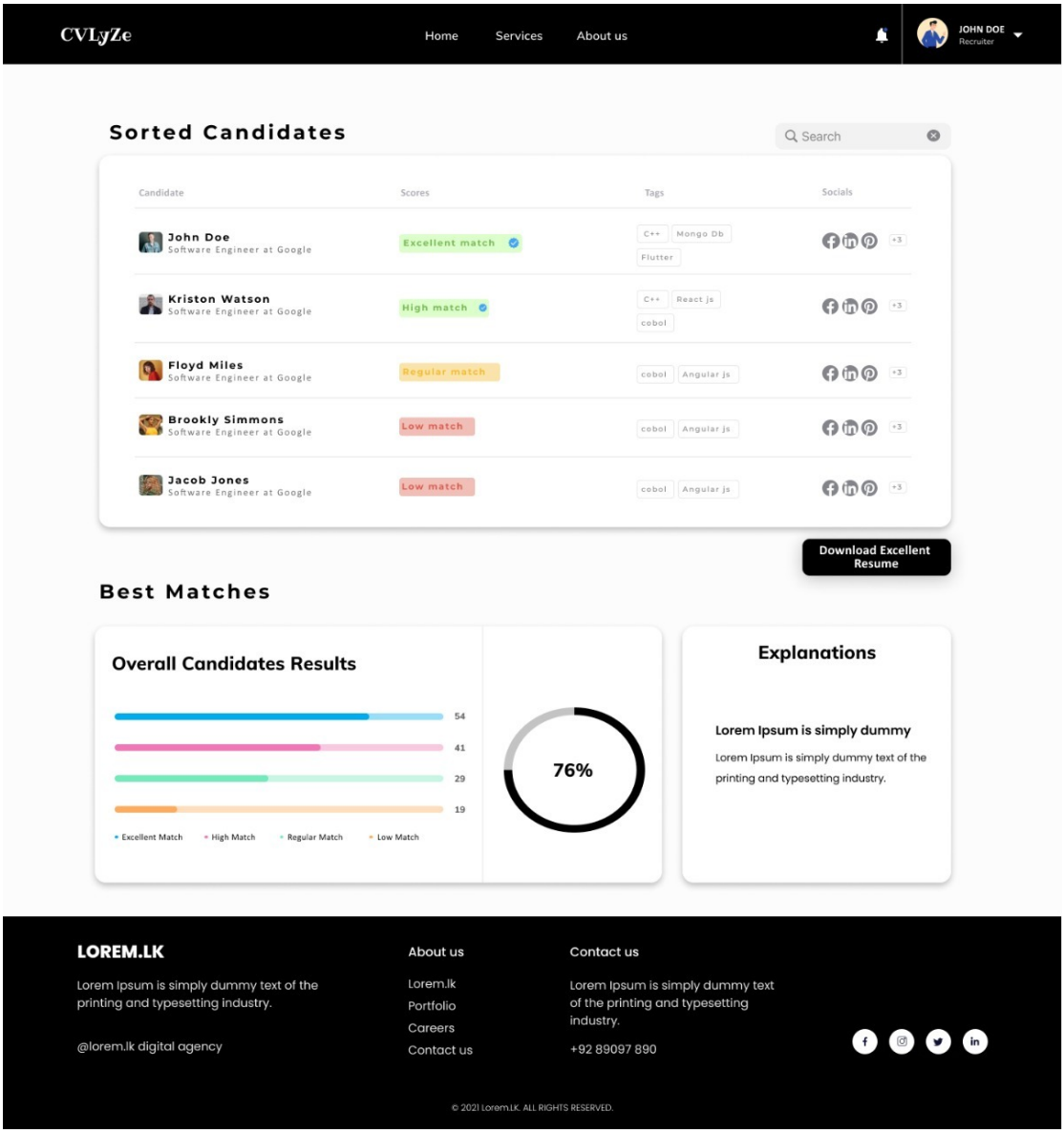

Figure 15 Sort report (prototype)

4.8. User Interface Design

The final user interface is given below after iterative improvements in the protypes

4.8.1. Home page

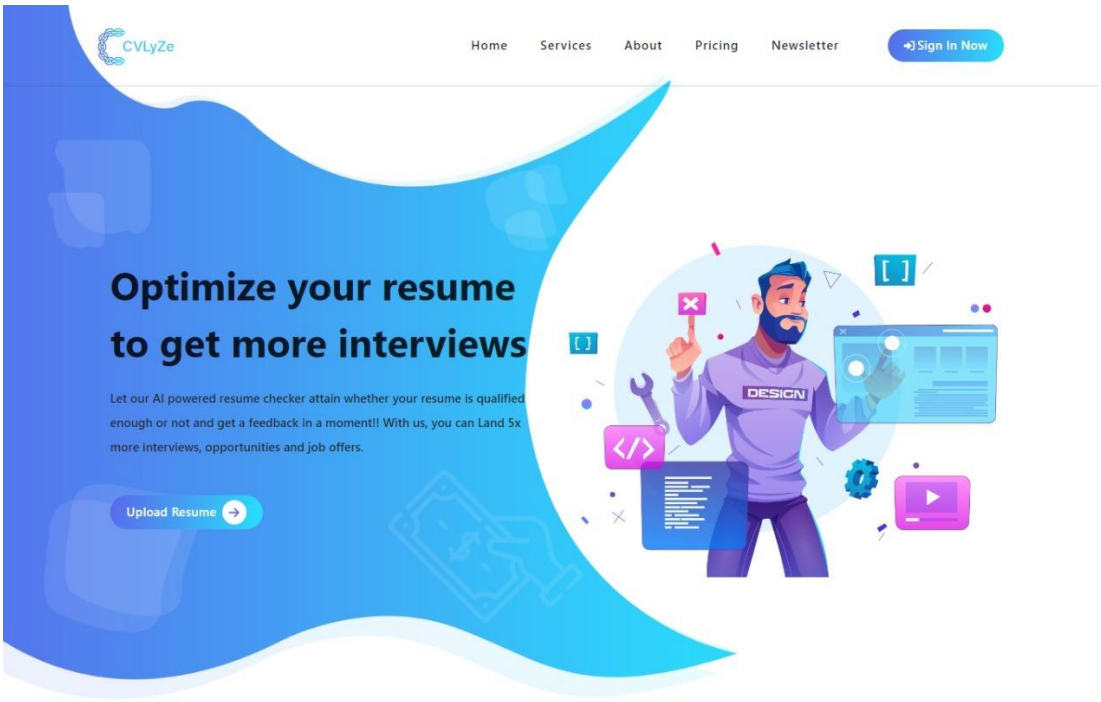

Figure 16 Home page design

4.8.2. Login

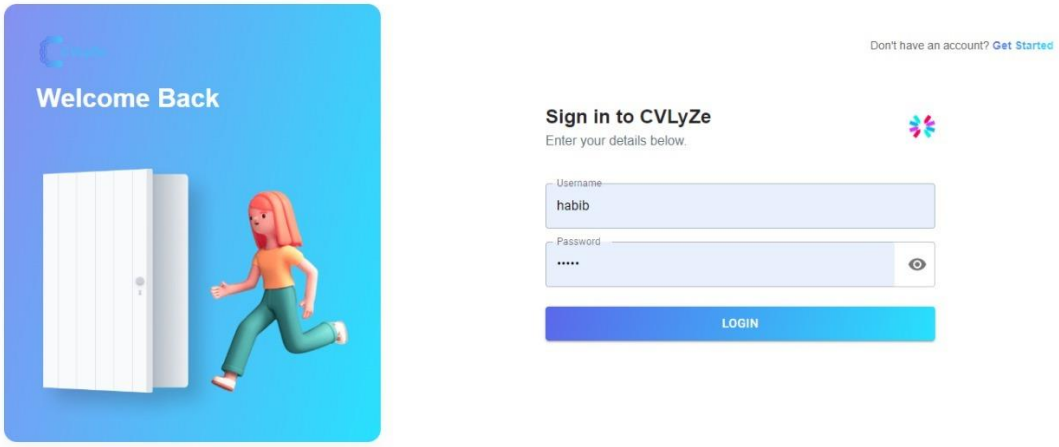

Figure 17 Login page design

4.8.3. Email verification

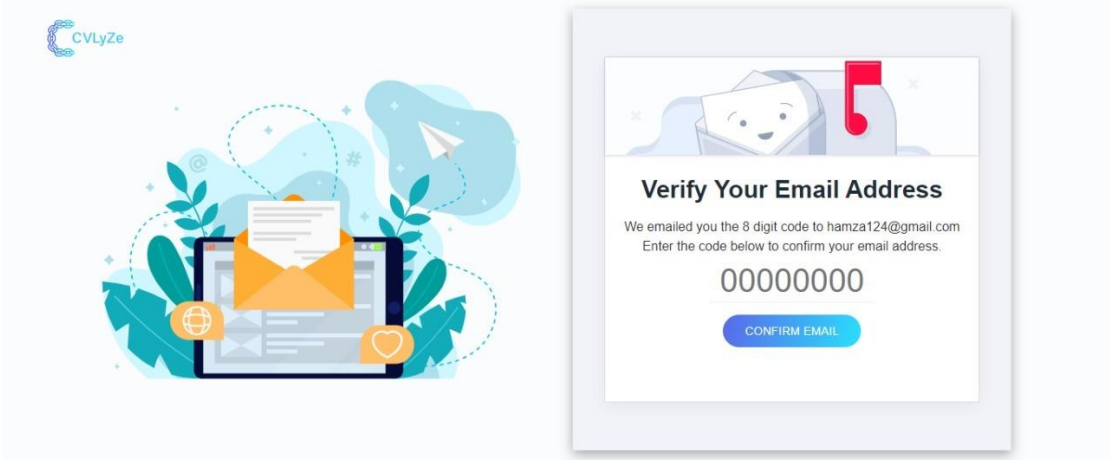

Figure 18 Email verification design

4.8.4. Candidate home page

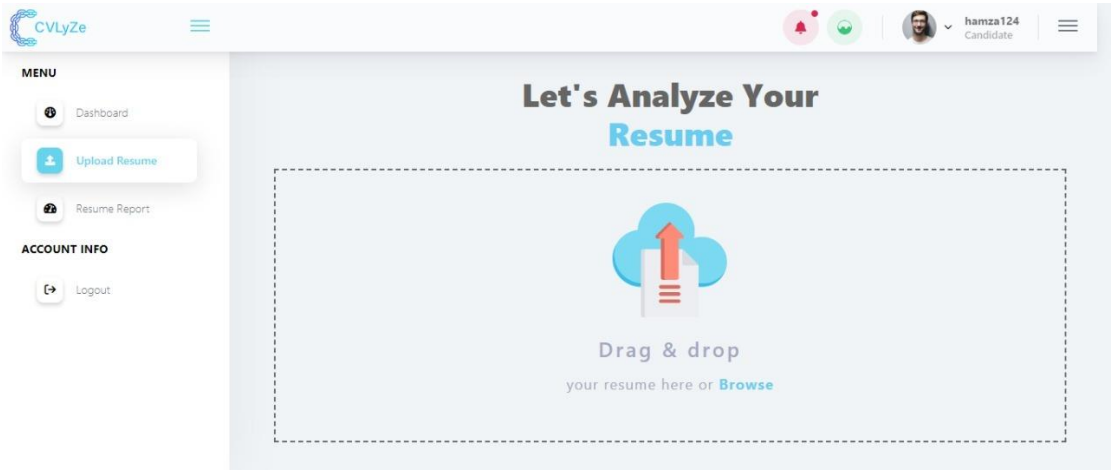

Figure 19 Candidate home page design

4.8.5. Resume report

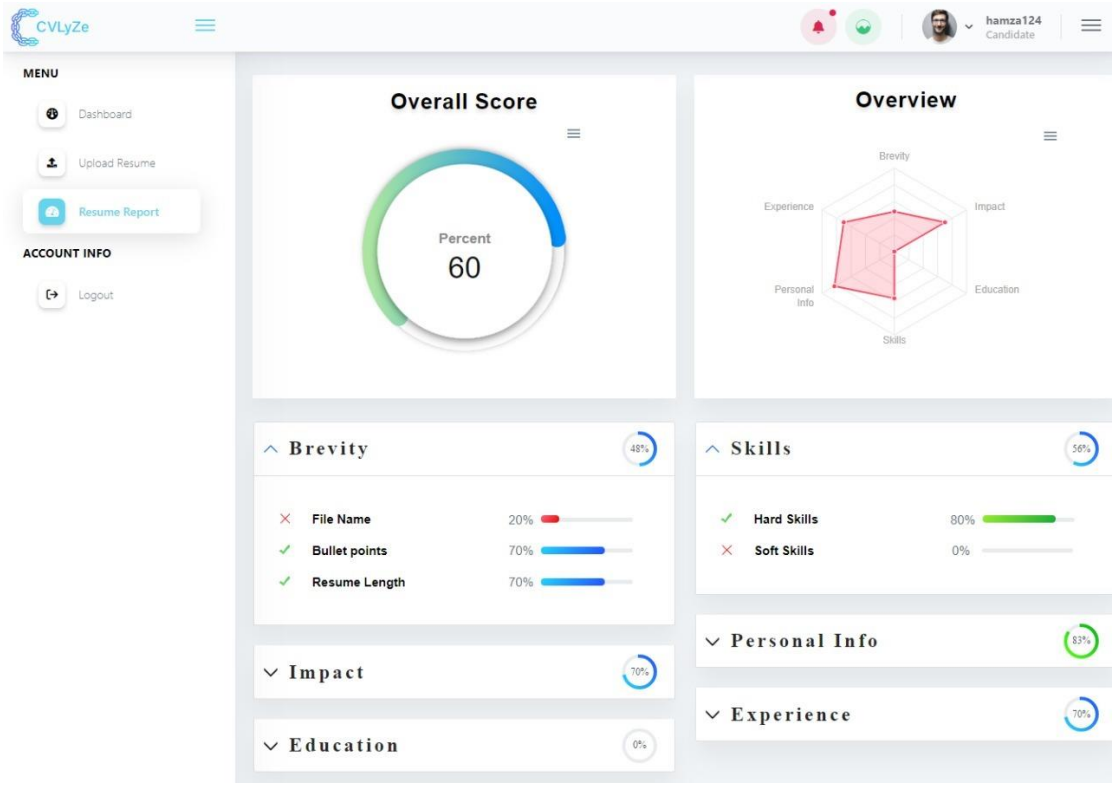

Figure 20 Resume report design

4.8.6. Improvement suggestions

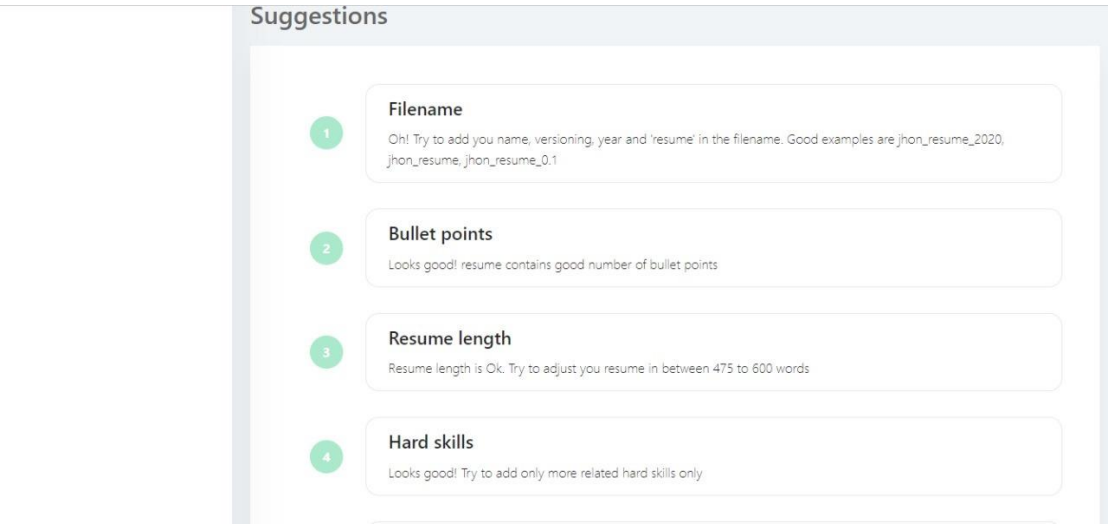

Figure 21 Improvement suggestions design

4.8.7. Candidate previous reports

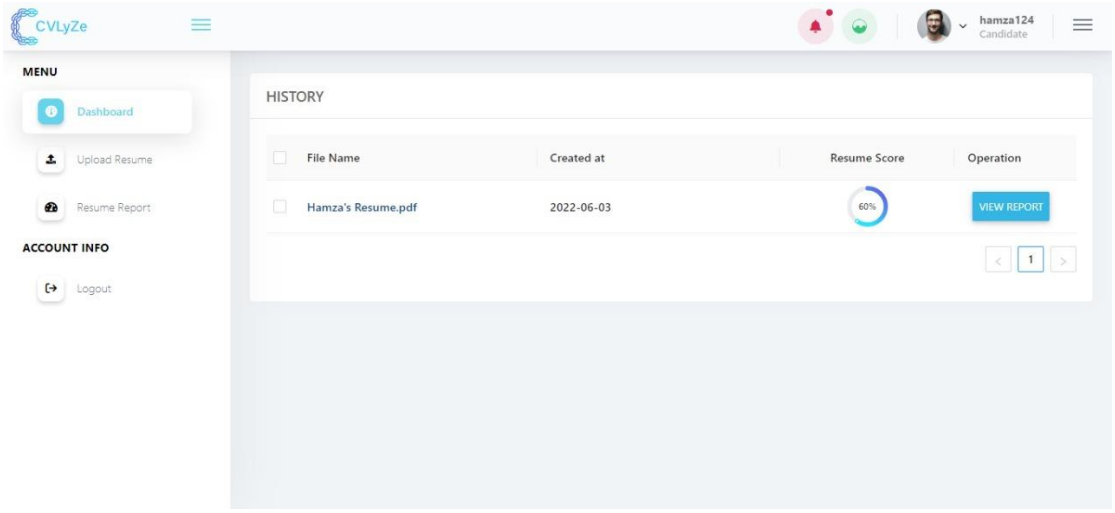

Figure 22 Candidate previous reports

4.8.8. Recruiter home page

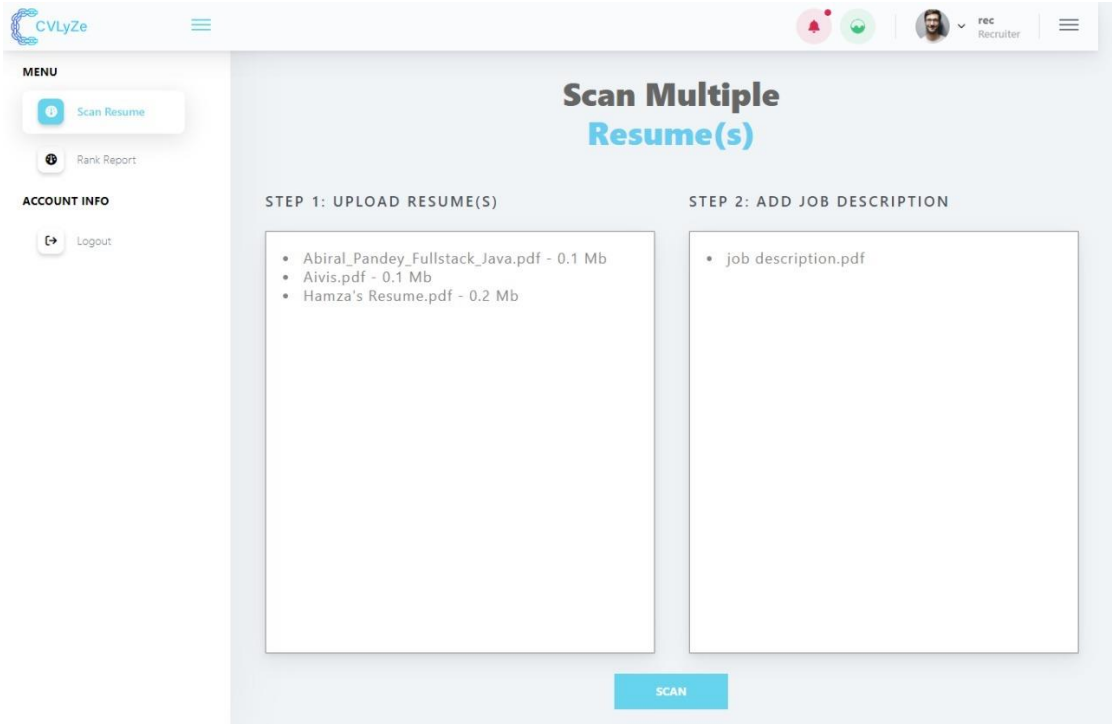

Figure 23 Recruiter home page design

4.8.9. Sort report

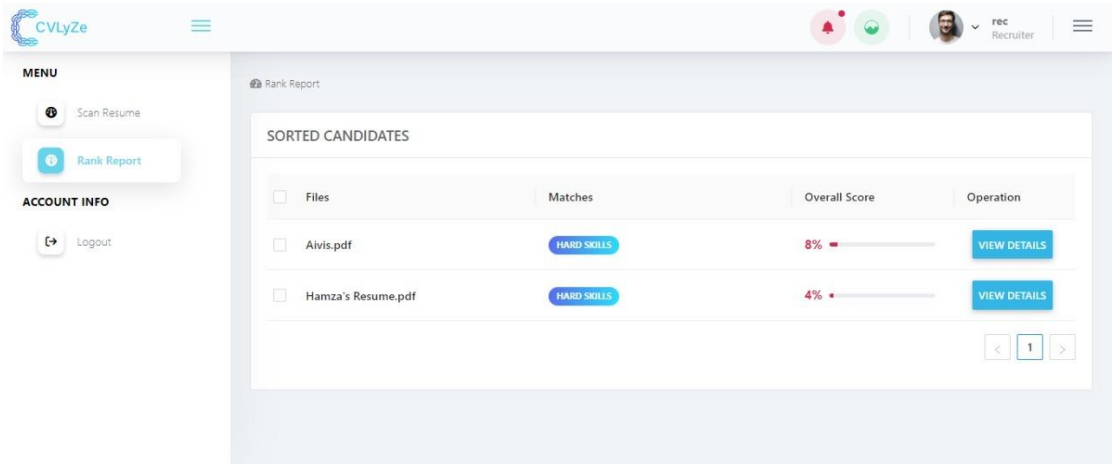

Figure 24 Sort report design

4.8.10. Sort report individual report

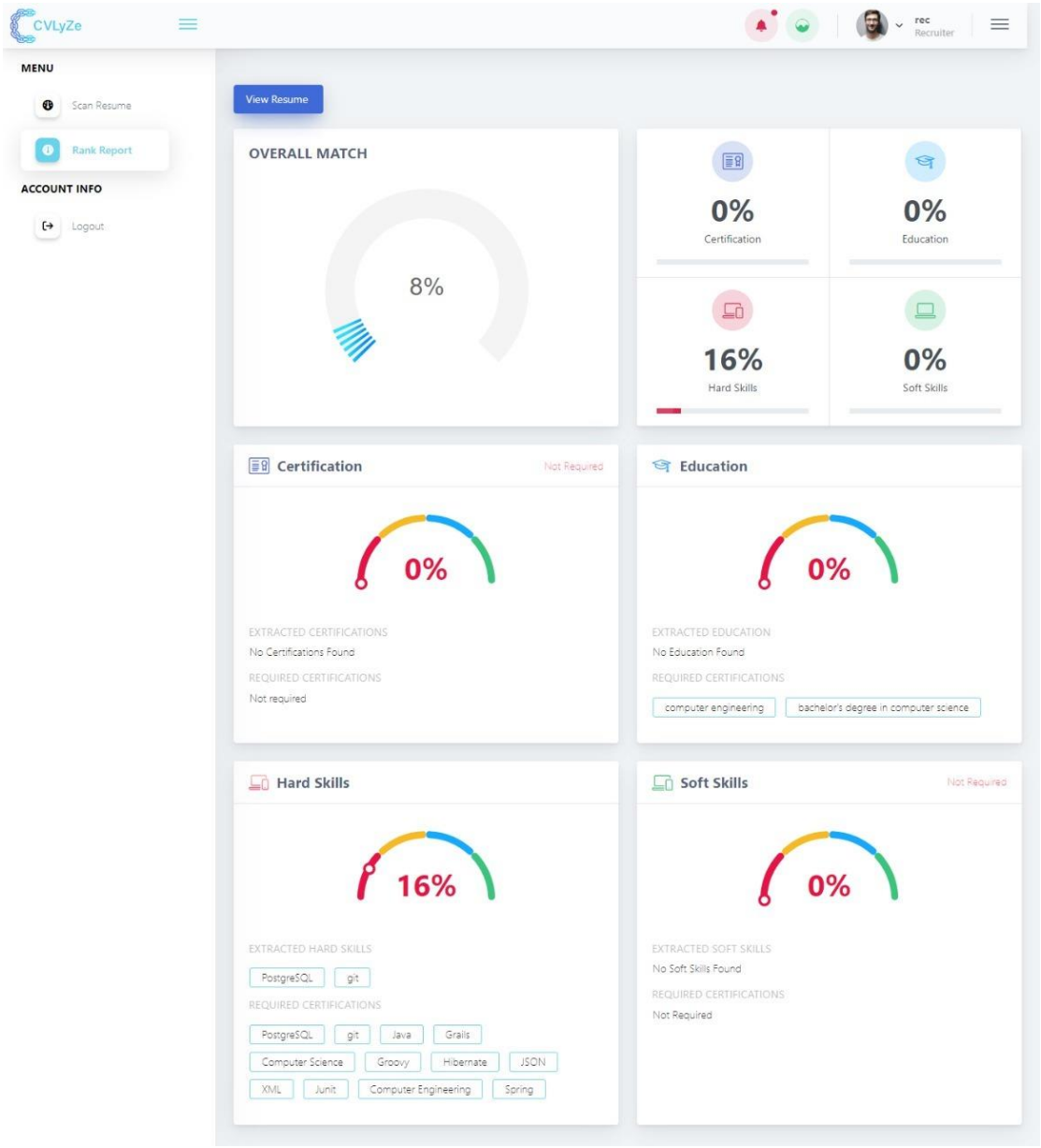

Figure 25 Sort report individual report design

4.8.11. Admin home page

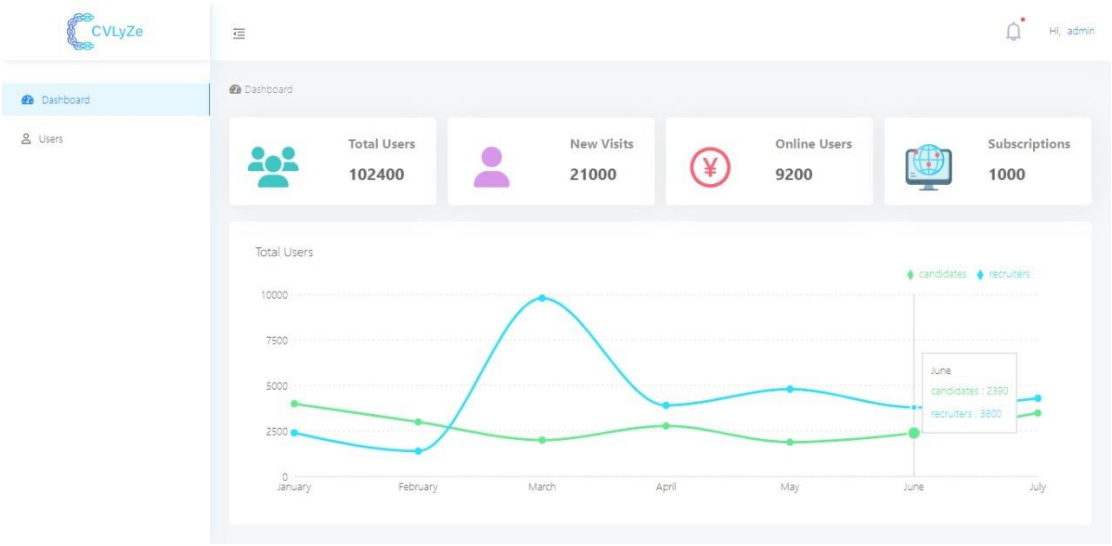

Figure 26 Admin dashboard design

4.8.12. Admin users panel

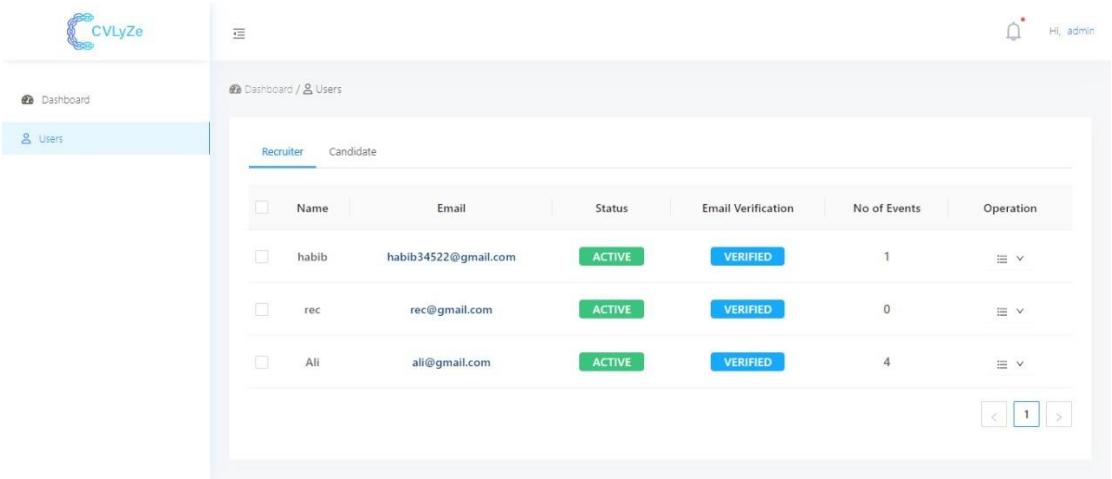

Figure 27 Admin user panel design

4.9. Conclusion

Chapter covers the detailed system design, architecture, dynamic view, component representation and the data models. At the end of this chapter, one has knowledge of system's inner structure and the working of the system

Chapter 5

System Implementation

5.1. Tools and technologies

5.1.1. Backend

Fast API is used to build API around the system. The system used other technologies like Celery, RabbitMQ, and Redis to handle asynchronous tasks.

5.1.2. Frontend

ReactJS is used to build the front-end of the system. Other react supportive libraries like Redux, graphs, etc are used to make the front-end more interactive.

5.1.3. Database

MySQL is used as a database for the system due to its high performance.

5.1.4. Architecture

The system is built using microservices architecture which makes the code more reusable, highly maintained, and the quick response time.

5.1.5. Authentication

JWT (JSON Web Token) is used to authenticate user requests and encrypt the user passwords that expire after some time and the user needs to log in again to get the new token.

5.1.6. Object-relational mapper

SQLAlchemy is used as an ORM in the system which automatically converts the queries to base SQL queries and returns the data in objects.

5.1.7. System libraries

1. amqp==5.1.1
2. anyio==3.5.0
3. asgiref==3.5.0
4. async-timeout==4.0.2
5. billiard==3.6.4.0
6. blis==0.4.1
7. cachetools==5.1.0
8. catalogue==1.0.0
9. celery==5.2.6
10. certifi==2021.10.8
11. cffi==1.15.0
12. charset-normalizer==2.0.12
13. ci-info==0.2.0
14. click==8.1.2
15. click-didyoumean==0.3.0
16. click-plugins==1.1.1
17. click-repl==0.2.0
18. colorama==0.4.4
19. configobj==5.0.6
20. configparser==5.2.0
21. cssselect==1.1.0
22. cssutils==2.4.0
23. cymem==2.0.6
24. Deprecated==1.2.13
25. dropbox==11.30.0
26. ecdsa==0.17.0
27. etelemetry==0.3.0
28. fastapi==0.75.2
29. filelock==3.6.0
30. fitz==0.0.1.dev2
31. future==0.18.2
32. fuzzywuzzy==0.18.0

33. gevent==21.12.0
34. greenlet==1.1.2
35. h11==0.13.0
36. httplib2==0.20.4
37. idna==3.3
38. install==1.3.5
39. isodate==0.6.1
40. kombu==5.2.4
41. lxml==4.8.0
42. mariadb==1.0.11
43. murmurhash==1.0.7
44. mysqlclient==2.1.0
45. networkx==2.8
46. nibabel==3.2.2
47. nipype==1.7.1
48. numpy==1.22.3
49. packaging==21.3
50. pandas==1.4.2
51. passlib==1.7.4
52. pathlib==1.0.1
53. plac==1.1.3
54. ply==3.11
55. premailer==3.10.0
56. preshed==3.0.6
57. prompt-toolkit==3.0.29
58. prov==2.0.0
59. pyasn1==0.4.8
60. pycparser==2.21
61. pydantic==1.9.0
62. pydot==1.4.2
63. PyMuPDF==1.19.6
64. PyMySQL==1.0.2
65. pyparsing==3.0.8
66. python-dateutil==2.8.2

67. python-jose==3.3.0
68. python-Levenshtein-wheels==0.13.2
69. python-multipart==0.0.5
70. pytz==2022.1
71. pyxnat==1.4
72. rdflib==6.1.1
73. redis==4.2.2
74. requests==2.27.1
75. rsa==4.8
76. scipy==1.8.0
77. simplejson==3.17.6
78. six==1.16.0
79. sniffio==1.2.0
80. spacy==2.3.0
81. SQLAlchemy==1.4.36
82. SQLAlchemy-Utils==0.38.2
83. srsly==1.0.5
84. starlette==0.17.1
85. stone==3.3.1
86. thinc==7.4.1
87. tqdm==4.64.0
88. traits==6.3.2
89. typing-extensions==4.2.0
90. urllib3==1.26.9
91. uvicorn==0.17.6
92. vine==5.0.0
93. wasabi==0.9.1
94. wcwidth==0.2.5
95. wrapt==1.14.0
96. yagmail==0.15.277
97. zope.event==4.5.0
98. zope.interface==5.4.0

5.2. Development process

The system is developed using incremental development and prototyping by taking small steps towards the final product. Every small step is implemented and tested separately. After testing every iteration is integrated to the entire system until the whole product is built.

5.3. Key Features

5.3.1. Resume analysis

The user uploads the resume from the front end. After that, the backend system extracts the data from the resume using Natural Language Processing. This extracted data is then passed to the evaluation framework which generates scores for different classes. These scores are then used to generate reports and suggestions.

5.3.1.1. Evaluation framework

The evaluation framework contains the importance of every entity in the resume. The evaluation formwork contains the following entities

- **Brevity**
  - Filename
  - Bullet points
  - Resume length
- **Education**
  - Education-related data
- **Experience**
  - Experience role
  - Experience locations
  - Experience time periods
- **Impact**
  - Action verbs
  - Cliches
  - Difficult words
  - Negative words
  - Very phrase
  - Bad patterns
  - Avoid words
- **Personal information**
  - Candidate name
  - Email address
  - Phone number

    - Personal links
- **Skill**
    - Hard skills
    - Soft skills

5.3.2. Sort Report

The recruiter uploads the resumes and the job description. The system extracts data from each resume and the job description. The system matches the extracted data and generates a report for every candidate with a sorted list. Currently it only extraction education, certification, hard skills, and soft skills.

5.4. Conclusion

The discussion related to system development covers the process of system development. Most of the code is reused which is the main advantage of using Python as a core language. All libraries and the frameworks used are mentioned above.

Chapter 6

System Testing & Evaluation

6.1. Test Strategy

After the development is done, the system testing is needed to be carried out to point out defects and errors that were made during the development phases. System testing is an essential phase because it makes sure of the customers reliability and their satisfaction for the system. In our project, we have done 5 types of testing which are listed below

6.2. Component Testing:

Component testing has been performed to test how individual components of the system are working separately without integrating with other components.

6.3. Unit Testing:

Individual program units and objects have been tested in unit testing. Here, by doing unit testing we have focused on testing the functionality of methods and to confirm that each method gives the expected result.

6.4. Integrated Testing:

Integration testing has been performed to test how different units, modules and components of our system are behaving as a combined entity.

6.5. System Testing:

System testing is the combination of two or more sub-system and then it is tested. Here, we tested the entire system as per the requirements.

6.6. Test Cases:

6.6.1. Login Scenario

Test Case ID: 1			Test Designed by: Hamza Asif		
Test Priority: High			Test Designed date: May 22, 2022		
Module Name: Login			Test Executed by: Hamza Asif		
Test Title: User gets logged in			Test Execution date: May 22, 2022		
Description: Input username and password and check if the test case fails or passes by comparing the actual and expected result.					
Test Case Id	**Input**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
1	Username and Password are blank	firstname=“ ” password=“ ”	Error Message “Username and password cannot be blank” displayed	Message Displayed	Pass
2	Username and Password are incorrect Password	firstname=“hamza” password=“123”	Alert Message “Invalid username or password”	Message displayed	Pass
3	Correct Username and Password	firstname=“habib” password=“habib123”	Based on role, user will be redirected to his dashboard	Based on role, user will be redirected to his dashboard	Pass

Table 9 Login test cases

6.6.2. Sign up Scenario

Test Case ID: 2			Test Designed by: Hamza Asif		
Test Priority: High			Test Designed date: May 22, 2022		
Module Name: Sign up			Test Executed by: Hamza Asif		
Test Title: User gets Signed up			Test Execution date: May 22, 2022		
Description: Input first-name, last-name, username, email, password, confirm password and recruiter checkbox and check if the test case passes or fails by comparing the expected and actual results.					
Test Case Id	**Input**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
1	All input fields are blank	All fields=blank	Error Message “Please fill out the fields” displayed	Message Displayed	Pass
2	firstname is blank, other fields are valid	firstname=“” lastname=“ahmed” firstname=”aliahmed” email=”ali@gmail.com” password=”123” confirmPassword=”123”	Alert Message “please fill out first name field” displayed	Message displayed	Pass

3	lastname is blank, other fields are valid	firstname=“ali” lastname=“” firstname=”aliahmed” email=”ali@gmail.com” password=”123” confirmPassword=”123”	Alert Message “please fill out last name field” displayed	Message displayed	Pass
4	username is blank, other fields are valid	firstname=“ali” lastname=“ahmed” firstname=”” email=”ali@gmail.com” password=”123” confirmPassword=”123”	Alert Message “please fill out username field” displayed	Message displayed	Pass
5	email is blank, other fields are valid	firstname=“ali” lastname=“ahmed” firstname=”aliahmed” email=”” password=”123” confirmPassword=”123”	Alert Message “please fill out email field” displayed	Message displayed	Pass
6	password is blank, other fields are valid	firstname=“ali” lastname=“ahmed” firstname=”aliahmed” email=”ali@gmail.com” password=”” confirmPassword=”123”	Alert Message “please fill out password field ” displayed	Message displayed	Pass

7	Conform password is blank, other fields are valid	firstname="ali" lastname="ahmed" firstname="aliahmed" email="ali@gmail.com" password="123" confirmPassword=""	Alert Message "please fill out confirm password field " displayed	Message displayed	Pass
8	All fields are entered but passwords and confirm password fields are not same	firstname="ali" lastname="ahmed" firstname="aliahmed" email="ali@gmail.com" password="123" confirmPassword="4f3"	Alert Message "password doesn't match " displayed	Message displayed	Pass
9	Username entered that already exists	firstname="ali" lastname="ahmed" firstname="hamza" email="ali@gmail.com" password="123" confirmPassword="4f3"	Alert Message "username already exists " displayed	Message displayed	Pass
10	email entered that already exists	firstname="ali" lastname="ahmed" firstname="aliahmed" email="hamza@gmail.com" password="123" confirmPassword="4f3"	Alert Message "username already exists " displayed	Message displayed	Pass

11	All fields are valid	firstname="ali" lastname="ahmed" firstname="aliahmed" email="ali@gmail.com" password="123" confirmPassword="123"	User redirected to email verification page	User redirected to email verification page	Pass

Table 10 Sign up test cases

6.6.3. Email Verification Scenario

Test Case ID: 3			Test Designed by: Hamza Asif		
Test Priority: High			Test Designed date: May 22, 2022		
Module Name: Email Verification			Test Executed by: Hamza Asif		
Test Title: User gets his email verified			Test Execution date: May 22, 2022		
Description: Input 8-digit code from the email sent on the provided email account of the user and checks if the test case fails or passes by comparing the actual and expected results.					
Test Case Id	**Input**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
1	Blank Email verification code	email-verification=blank	Error Message "Please fill out the field" displayed	Message Displayed	Pass
					Pass

2	Invalid email verification code	email-verification=”12345678”	Alert Message “please enter valid email verification code” displayed	Message displayed	
3	Valid email verification code	email-verification=”21412212”	On success user will be redirected to login page	User redirection to login page	Pass

Table 11 Email verification test cases

6.6.4. Candidate Upload Resume Scenario

Test Case ID: 4			Test Designed by: Hamza Asif		
Test Priority: High			Test Designed date: May 22, 2022		
Module Name: Candidate uploading resume			Test Executed by: Hamza Asif		
Test Title: Uploading resume			Test Execution date: May 22, 2022		
Description: Inputs resume and checks if the test case fails or passes by comparing the actual and expected results.					
Test Case Id	**Input**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
1	Resume uploaded .pdf format uploaded by user	“resume.pdf”	Success Message “Resume uploaded successfully” displayed	Message Displayed	Pass

2	Invalid resume format uploaded by user	“resume.docs”	Alert Message “only pdf format accepted” displayed	Message displayed	Pass
3	Valid resume format and scan button clicked	“resume.pdf”	User will be redirected to candidate results page	User redirection to candidate results page	Pass

Table 12 Upload resume test cases

6.6.5. Recruiter Upload Resume Scenario

Test Case ID: 5			Test Designed by: Hamza Asif		
Test Priority: High			Test Designed date: May 22, 2022		
Module Name: Recruiter uploading resume against Job description			Test Executed by: Hamza Asif		
Test Title: Uploading resume(s)			Test Execution date: May 22, 2022		
Description: Inputs resume(s) and job description and checks if the test case fails or passes by comparing the actual and expected results.					
Test Case Id	**Input**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
1	Resume(s) uploaded .pdf format uploaded by user	Resume=[{“resume.pdf”}] Job description=””	Alert Message “please input job description” displayed	Message Displayed	Pass

	But job description empty				
2	Resume(s) section blank But job description uploaded .pdf format uploaded by user	Resume=[] Job description=”jd.pdf”	Alert Message “only pdf format accepted” displayed	Message displayed	Pass
3	Resume(s) uploaded .pdf format uploaded by user Job description uploaded	Resume=[{“resume.pdf”}] Job description=”jd.pdf”	User will be redirected to recruiter rank results page	User redirection to recruiter rank results page	Pass

Table 13 Multiple resume upload test cases

6.7. Conclusion

In this section, we went through the software testing phase, we understood the functional testing and we have written test cases for our system.

Chapter 7
Conclusion

The project was an attempt to help candidates and recruiters in a positive way. Now, candidates are one click away from getting their resume scores and suggestions.
This report provides a complete documentation of the project including its introduction, brief and technical details of design and implementation.

7.1. Contributions

Equal contributions have been carried out by each group member during each phase such as, the documentation, project analysis, requirements, design, development etc.

7.2. Future work

There is always a space of improvement especially in learning-based systems. We will work on the efficiency of the named entity recognition model of the system. NER is the core of the system because the system is heavily based on the data extracted. The improvement evaluation framework is also important because it defines the rules of evaluation. Every resume is evaluated according to the evaluation framework. In short, we are planned to improve the whole system to meet needs.

REFERENCES

[1] N. G. O, "Named Entity Recognition based Resume Parserand Summarizer," *International Journal of Advanced Research in Science, Communication and Technology,* vol. 2, no. 1, p. 8, 2022.

[2] B. Gaur, "Semi-supervised deep learning based named entity recognition model to parse education section of resumes," p. 14, 2020.

[3] Dr.K.Satheesh, "Resume Ranking based on Job Description using SpaCy NER model," *International Research Journal of Engineering and Technology,* vol. 7, no. 5, p. 4, 2020.

[4] S. Sanyal, "Resume Parser with Natural Language Processing," vol. 7, no. 2, p. 6, 2017.

[5] H. Huang, "A Low-Cost Named Entity Recognition Research Based on," p. 10, 2018.

[6] A. Roy, "Recent Trends in Named Entity Recognition," p. 27, 2021.

[7] P. Sun, "An Overview of Named Entity Recognition," *Minority Languages Branch, National Language Resource and Monitoring Research Center,* p. 6.

[8] "A Two-Step Resume Information Extraction Algorithm," *Hindawi,* vol. 2018, p. 8, 2018.

Turnitin Originality Report

Processed on: 05-Jun-2022 01:03 PKT
ID: 1850402266
Word Count: 8469
Submitted: 1

cvlyze By Habib Rehman

Similarity Index	Similarity by Source
9%	Internet Sources: 6% Publications: 0% Student Papers: 6%

3% match (Internet from 21-Dec-2021) http://etd.aau.edu.et/bitstream/handle/123456789/19174/Endashaw%20Wolde%20%20%202018.pdf?isAllowed=y&sequence=1
1% match (Internet from 09-Mar-2022) http://oric.aiu.edu.pk/wp-content/uploads/2021/07/Annex-C3-Thesis-Format-ARC.docx
1% match (Internet from 09-Sep-2020) https://www.slideshare.net/FatimaQayyum1/srs-software-requirement-specification-document
1% match (student papers from 23-Apr-2018) Submitted to De Montfort University on 2018-04-23
1% match (student papers from 08-Aug-2011) Submitted to Institute of Development Management on 2011-08-08
1% match (student papers from 03-Oct-2017) Submitted to Texas A&M University, College Station on 2017-10-03
1% match (student papers from 27-May-2011) Submitted to Manchester Metropolitan University on 2011-05-27
1% match (Internet from 21-Jul-2020) https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=1509&context=etd&httpsredir=1&referer=