

JobMate: A Web Application for Resume and Job Application Improvement

Progress Report

Supervisor: Dr. Lynn Marshall

Team Members:

Areeb Haq 101115337

Ali Alvi 101114940

Ahmad Abuoudeh 10107236

Chapter 1: Abstract

This document outlines the information required for the 4th year project progress report. This report is structured to provide an update on the progress, difficulties, and alterations on requirements regarding the JobMate project from Team 29. Also found in this report are the chosen Architecture and Design Patterns, Full-Stack Progress, Objectives and Deadlines, as well as associated Difficulties and Mitigation Strategies. The rest of this report will showcase how Team 29 has taken steps towards completing the objective of the project.

With the introduction of applicant tracking systems known as ATS [1], the job search process for candidates worldwide has changed drastically. ATS is utilized by employers to swiftly automate skim through resumes and only display resumes that match their job postings and criteria for the jobs. This technology comes with pros and cons, however mainly hurting the employee as a computer software is able to reject their application solely based on the configurations set in the ATS system.

Due to recent international events such as the COVID-19 pandemic, the job search process has become increasingly competitive due to the huge shift towards remote work and an overall shift towards different and more prosperous careers.

The purpose of this project is to develop a web application that automates parts of the job search process and helps users improve their resumes. The application will allow users to input information about their job search, including the websites they want to use, their field of employment, location, and level of seniority. The application will then use this information to retrieve relevant job postings and provide feedback on the user's resume.

The application will also provide a list of jobs that the user's resume could have qualified for but fell short of the set percent match threshold. This will give users an idea of what skills and experiences they can add to their resumes to improve their chances of being hired. Additionally, the application will perform grammar checks and provide a rating of the user's resume, along with guidelines for improving it. This feature helps combat the strictness of modern ATS systems.

Overall, the goal of this project is to create a one-stop-shop for users looking to improve their resumes and find new job opportunities. By automating parts of the job search process and providing personalized feedback on resumes, the application will make it easier and more effective for users to find the right job. By providing personalized feedback and guidance on resumes, the project will improve the job application process and help job seekers to stand out from the competition. This will ultimately benefit both job seekers and employers by making it easier to find the right candidate for the job.

Table of Contents

Chapter 2: Standard Information

- 2.1: Health and Safety
- 2.2: Engineering Professionalism
- 2.3: Project Management
- 2.4: Justification of Suitability for Degree Program
- 2.5: Individual Contributions
 - 2.5.1: Project Contributions
 - 2.5.2: Report Contributions

Chapter 3: Architecture and Design Patterns

Chapter 4: Progress on Current Full-Stack Solution

- 4.1: Progress on Current Full-Stack Solution
- 4.2: Front-End Progress
- 4.3: Database Progress

Chapter 5: Reflections

- 5.1: Mitigation Strategy Application and Objective Alterations
- 5.2: Upcoming Work and Possible Risks/Issues
- 5.3: Conclusion on Progress

Chapter 7: References

Proposal

Chapter 2: Standard Information

2.1: Health and Safety

In the development of the systems in this project, all general health and safety principles as well as safety procedures were implemented. Due to the negligent chemical, or hazardous waste risks associated with this project, the majority of the Health and Safety risks that were adhered to were physical concerns. Fatigue, neck and shoulder pains, headaches, eyestrains, are all risks that are associated with bad posture while working on the computer and sitting for prolonged periods of time [2]. Due to the high complexity of this project requiring all members to spend large amounts of hours at a computer setup, regular breaks and limiting of screen usage were adhered to by all members.

2.2: Engineering Professionalism

To maintain a high level of engineering professionalism during collaboration and development of this project and associated documentation, practices learned during the ECOR 4995 Professional Practice course are implemented. All 3 members of the team have either taken or are currently taking the course and have followed the rules set in the proposal in moments of conflict or problems. These rules are stated below:

- Define the problem
- Determine the effect on the overall progress of our project
- Brainstorm ideas individually and as a group
- Select the most reliable solution for both the short and long term
- Determine an action plan to implement the agreed upon solution

In engineering projects, it is of utmost importance that communication between the engineers is taken very seriously. To communicate effectively, we aim for clarity with context, avoiding unnecessary information during problem solving sessions, and ensuring all team members are treated fairly. Finally, all members are expected and have been working in accordance with the Professional Engineer Code of Ethics [3]. The code states:

- “Fairness and loyalty to the practitioner's associates, employers, clients, subordinates, and employees; fidelity to public needs.
- devotion to high ideals of personal honour and professional integrity.
- knowledge of developments in the area of professional engineering relevant to any services that are undertaken; and
- competence in the performance of any professional engineering services that are undertaken.” [3]

These points are implemented in our workflow and are adhered to in order to uphold our professionalism.

2.3: Project Management

Our project management approach has continued to be the Agile methodology [4]. We have stuck to true Agile to perform iterative development as milestones were set in the proposal. The team has successfully split up requirements fairly and in conjunction with relative skills, preferences, and new technologies to learn for all individuals of the team.

To visualize our work, we have utilized a Kanban board [5] made in a simple word file to track current progress and roadblock of all work performed. The project is majority programming based requiring a use of a Git versioning system [6]. The team has a GitHub project setup linked in the references section of the report [7] where each team member has contributed to their own personal branches. These branches are then submitted as a PR to the main development branch where it is reviewed individually by the other team members. Upon approval, the branch is merged in. At the end of each iteration, the team collectively reviews all commits made in the development branch which is reviewed as a team and a final merge is done into the master branch. The master branch contains our product and is expected to be functioning correctly for any user to pull and start using according to the documentation provided.

2.4: Justification of Suitability for Degree Program

The team members are all in the Bachelor of Engineering: Software program at Carleton University and at the same year level. This education level coupled with various co-op terms completed by all members links very closely to the objectives and use of technologies in this project. The justification for suitability for this degree program can be broken into a few different components:

Software Requirements: For the successful development of our project, we have targeted a problem and correctly identified a solution that can be implemented. The solution is broken into the many aspects learned in SYSC 3120: Software Engineering Requirements and has allowed our team to implement the methodologies learned throughout this course.

Software Development and Design: Since the majority of our programs is focused on development and design, our team has thoroughly gone over the concepts and skills learned in courses such as SYSC 2004 focusing on object-oriented development, SYSC 2100 focused on algorithms and data structures, SYSC 3110 focused on object-oriented design patterns, SYSC 4504 focusing on the fundamentals of web development and finally SYSC 4101 focusing on software testing and validation. The combination of these classes has given us the confidence to implement this solution from start to end.

Databases: For our database we are utilizing the skills learnt through COMP 3005 which is focused on relational databases. All of us have taken this class and are confident in working with databases and integrating them into our solutions.

Communication and Presentation: Engineer Professionalism and communication skills are taught in CCDP2100 and ECOR4995 which both provided necessary information on how to communicate and work efficiently as a team whilst following many engineering principles.

In conclusion, our team is readily equipped with knowledge pertaining to the development and finalization of this project and we believe with our combined skills we are on track.

2.5: Individual Contributions

A project of this size requires fairly equal individual contributions from all team members. Our team has successfully followed the plan set out in our proposal and each team member has accordingly helped to fulfill the requirements. These contributions are further discussed in section 2.5.1 and 2.5.2.

2.5.1: Project Contributions

Areeb has contributed to back-end and front-end portions of the project. From the proposal, Areeb has completed all iterations assigned to him on time and with approval of all team members. Areeb developed fully functioning job posting web scrapers that are required for our back-end logic. This development is discussed in the progress section of the report. Areeb has also developed the resume upload page which is a page that is crucial to the flow of the application. Areeb's contributions have been very important in the current progress made thus far.

Ali has contributed to the back end and front-end portions of the project as well as setting up the database connections. Ali setup the Django project [18], researched and integrated a Resume Parser API [17] which is a major portion of our back-end logic service, and has implemented the user account creation process integrated with our database. Ali's contributions have been very important in the current progress made thus far.

Ahmad has contributed mainly to the front-end portion of our project. Ahmad has successfully designed and created the landing page for our front end for login and sign-up pages. These pages are very important as they are the starting point for the users of the web application. Ahmad's contributions have been very important in the current progress made thus far.

As a team there has been numerous pair programming sessions where all team members have contributed evenly to support and debug any problems that a team member may be facing. Due to this our development flow has been great and we are on track to complete our milestones.

2.5.2: Report Contributions

Areeb is the author of sections 1, 2.1, 2.2, 2.3, 4.2 of the report.

Ali is the author of sections 3, 4.3, 5.1, 5.2 of the report.

Ahmad is the author of sections 2.4, 2.5, 4.1, 5.3 of the report.

References are contributed to by all team members according to references given in sections.

Chapter 3: Architecture and Design Patterns

For a smooth and continuous delivery of project requirements, Team 29 has successfully developed a software architecture that matches the demanding requirements of a full-stack web application solution. The use of design patterns to optimize and simplify our code base is also discussed in this section.

Our web application is developed using the Django framework in the Python language [20]. The Django framework is based on the Model-View-Template architecture otherwise known as MVT. MVT is a design pattern for developing web applications with 3 main components. In the Django file structure, Python code can be placed within the project for logic and can be referenced wherever needed as Django is not strict with the files and structure. The model acts as the interface to the data, the view is the user interface, and the template consists of the static code of the web application such as HTML, and CSS [8]. Figure 1 below clearly describes the architecture of the MVT design pattern [9].

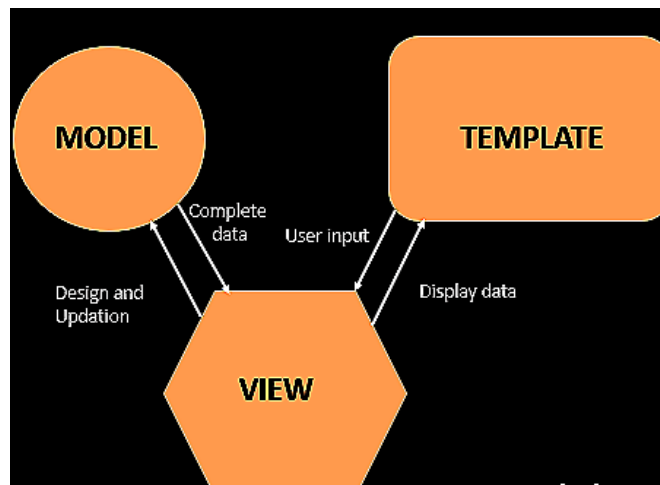


Figure 1: MVT Architecture [2]

The back end of our program and logic is based on a Monolith Architecture [10]. Monolith architecture allows for “ease of code management, cognitive overhead, and deployment” [11]. These benefits are achieved due to the application being built on one centralized code base. All the individual components that are required in the web application are stored in one repository and the small microservices such as the job application parsers, resume-parsing, and other services are all small components of the entire project. Due to the nature of this project being a 4th-year project, it makes sense to start our development using a monolithic structure that is independent of the use of technologies such as Docker.

The use of these design patterns has enabled our approach to the development of the program to be continuous and aligns with the technology stack that our team members are capable of implementing.

Chapter 4: Progress on Current Full-Stack Solution

This component of the report will outline the progress that Team 29 has accomplished to this date. The progress is split up into Sections 3.1: Front-End Progress, Section 3.2 Database Progress, and Section 3.3: Back-End Progress. In our Proposal, we discussed that Milestones 1-5 would be completed by November 18th, and we have achieved our target of hitting our milestones. The milestones are outlined below:

Milestone 1 (Ahmad): Creation of landing page and front-end UI: October 7, 2022

Milestone 2 (Ali): User account creation and database connection: November 18th, 2022

Milestone 3 (Areeb): Resume box upload page with database connection: November 18th, 2022

Milestone 4 (Ali): Resume Parser API: November 18th, 2022

Milestone 5 (Areeb): Scrapers for LinkedIn, Indeed (cut as not needed), Workopolis (changed to Glassdoor) November 18th, 2022

4.1: Front-End Progress

The JobMate web application has multiple web pages that are to be visual to the end user. These web pages include our main login page shown in Figure 2 and a sign-up page shown in Figure 3.

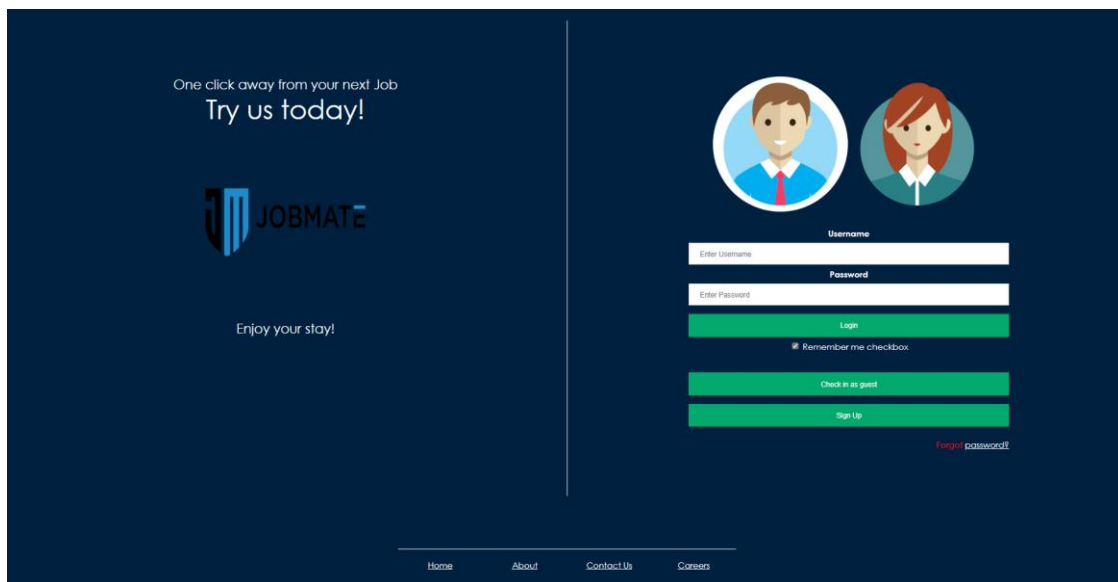


Figure 2: Login Page

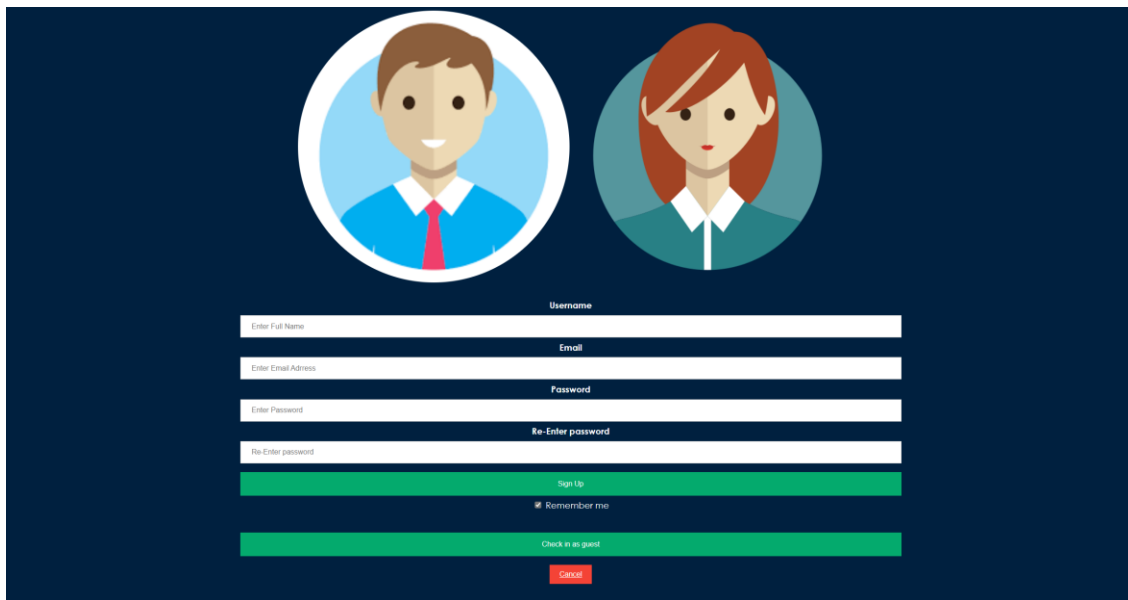


Figure 3: Sign-Up Page

The static HTML pages are added in the Django project: JobMate/my site/myapp/templates. The pages are then further separated into the login page and signup folders which maintain the actual HTML code files. These files maintain the HTML tags needed to set up the structure of our landing pages. As seen in Figures 2 and 3, there are images linked into the HTML which are stored in the static folder under myapp in the images folder. The associated CSS file for the webpages is seen under the same static folder named as styles.css. This CSS file maintains all of the styling choices made for our HTML templates.

For a user to sign up, the JobMate web application requests vital information such as Username, Email, and Password. This information is to be stored in our Postgres Database [19] which is further discussed in Section 3.2. The HTML template ensures that basic checks such as an email requiring the correct format {{username@domain_name.domain}} shown in Figure 4, and that passwords match in the Password and Re-Enter Password field box's shown in Figure 5.

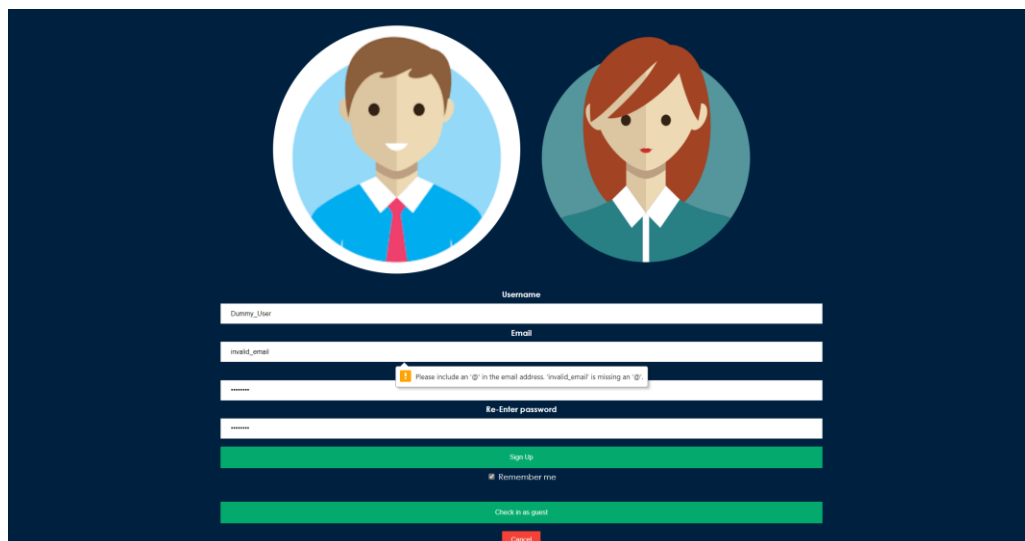


Figure 4: Invalid Email Address Error

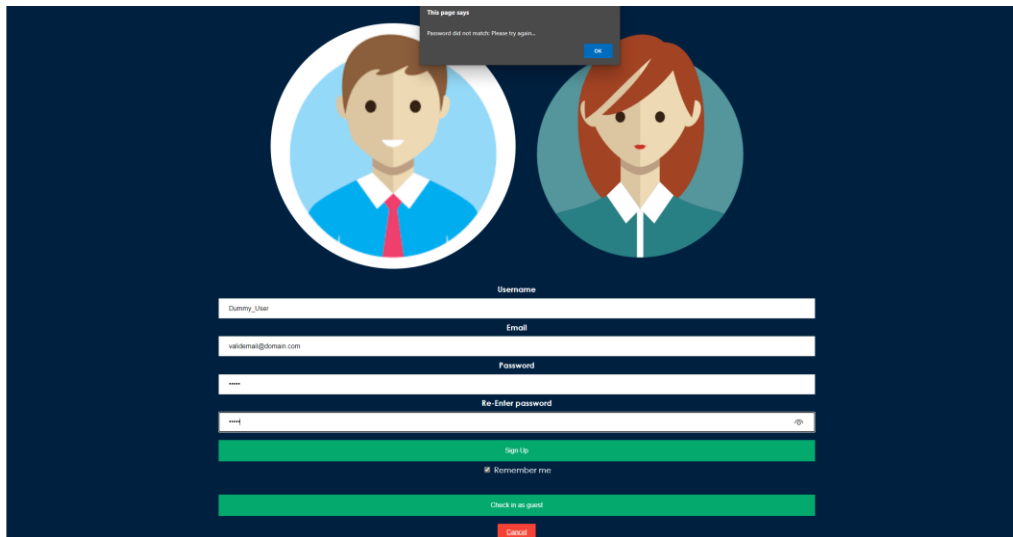


Figure 5: Non-Matching Password Error

In the HTML sign-up page, the logic for checking if the email is of the correct format in the input field is set as type email. This enables the error to be outputted. For the password error, as shown in Figure 5, we reference the HTML page to a JavaScript file which maintains the logic for checking if the passwords are equivalent. The JavaScript file is called logic-funcs.js which is stored under JobMate/mysite/myapp/static. The checkPassword function is passed a form as a parameter. This form from the HTML has psw and repsw id values which are the initial password and the re-entered passwords respectively. These passwords are stored in the password1 and password2 fields. The logic for checking that these two passwords are equivalent returns true and return false otherwise and an alert is given to the user.

Once the user has signed into the application they are redirected to the resume-upload page. Here they are given a clean UI to upload their resume by either dragging and dropping or browsing using the browse option. This is shown below in Figure 6. In Figure 7 shown below is the thumbnail/preview of the file the user has chosen to ensure that the file they want to upload is correct.

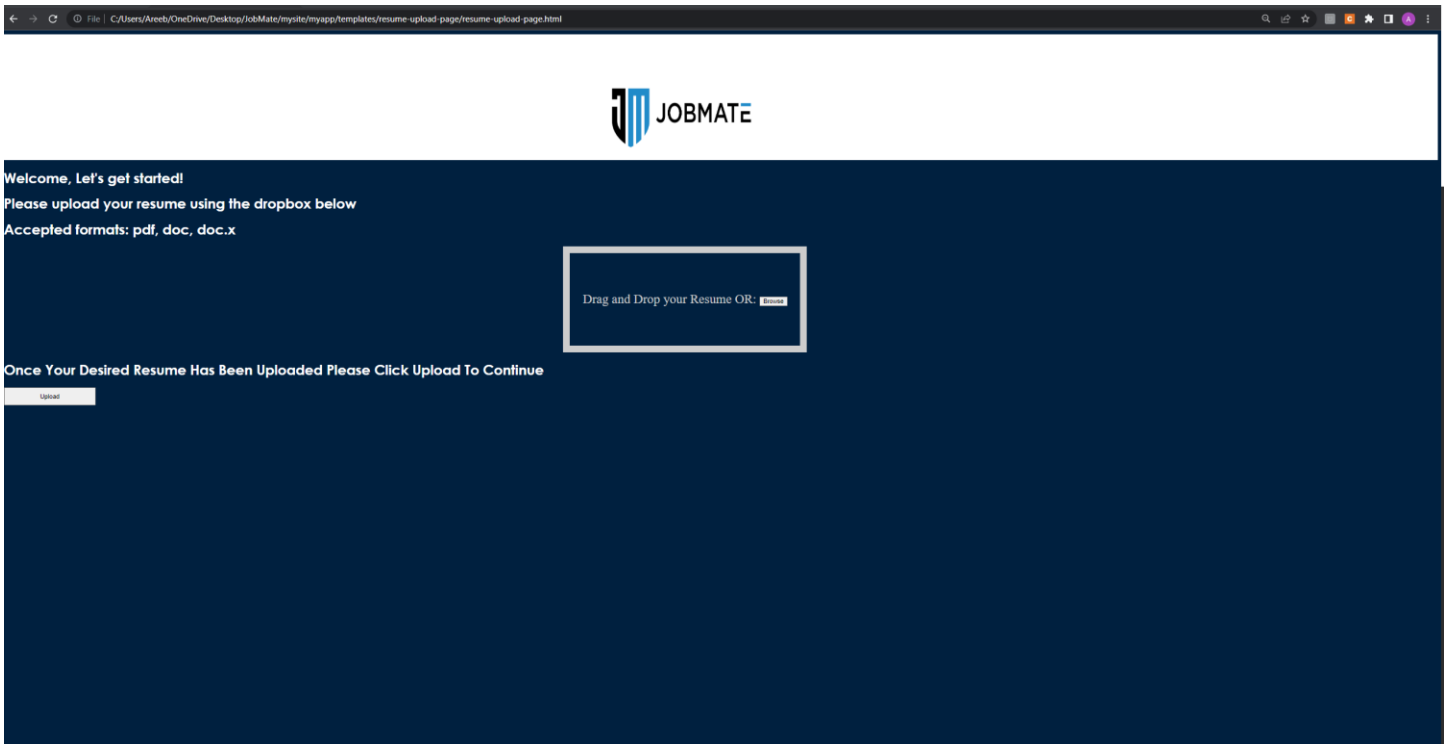


Figure 6: Resume Upload Page

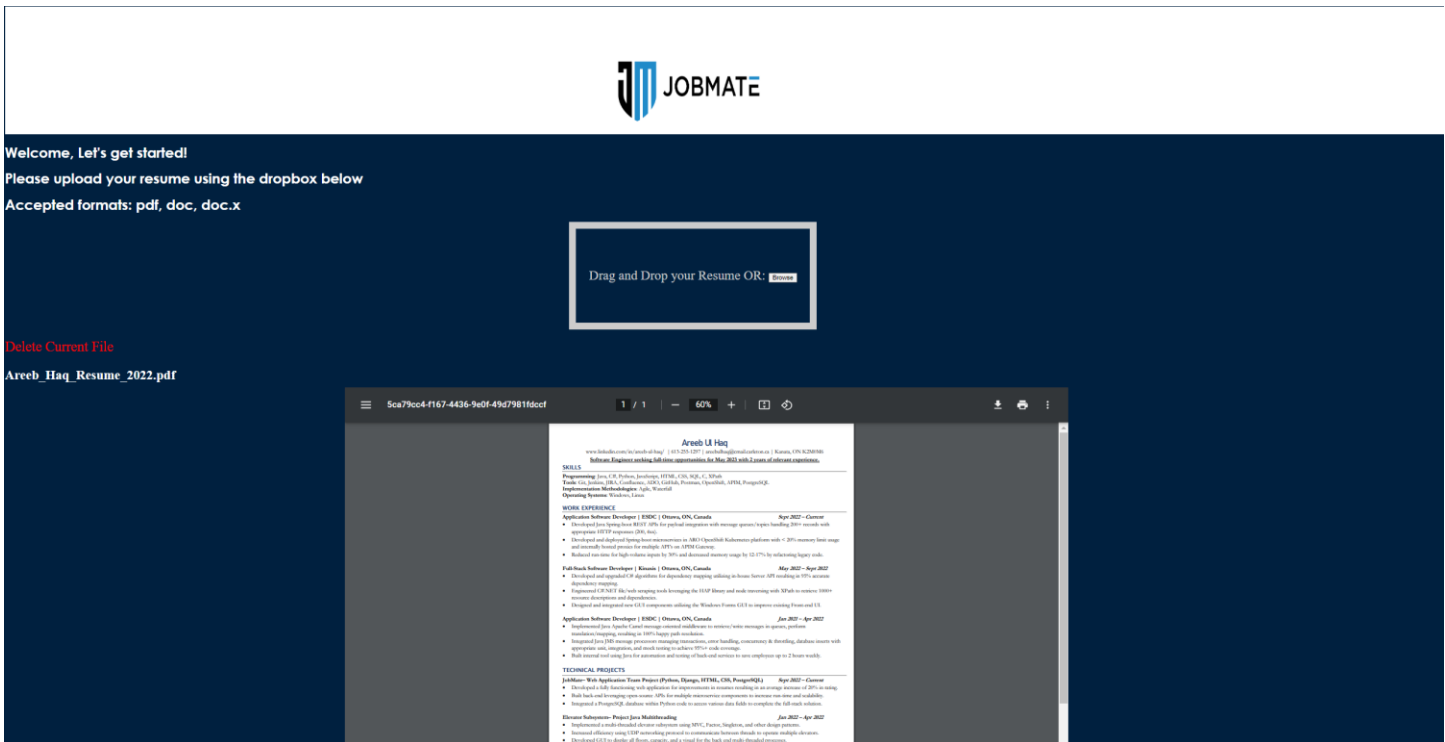


Figure 7: Resume upload page after file chosen

The code for this page can be found:

<https://github.com/alialvi00/JobMate/blob/master/mysite/myapp/templates/resume-upload-page/resume-upload-page.html>

4.2: Database Progress

The PostgreSQL database for our web application is set up and functioning as expected. Once a user account is created the account is stored in the database and during the login stage, the application checks with the database if the user is authenticated. An example of how our database functions for the user account creation process is shown below:

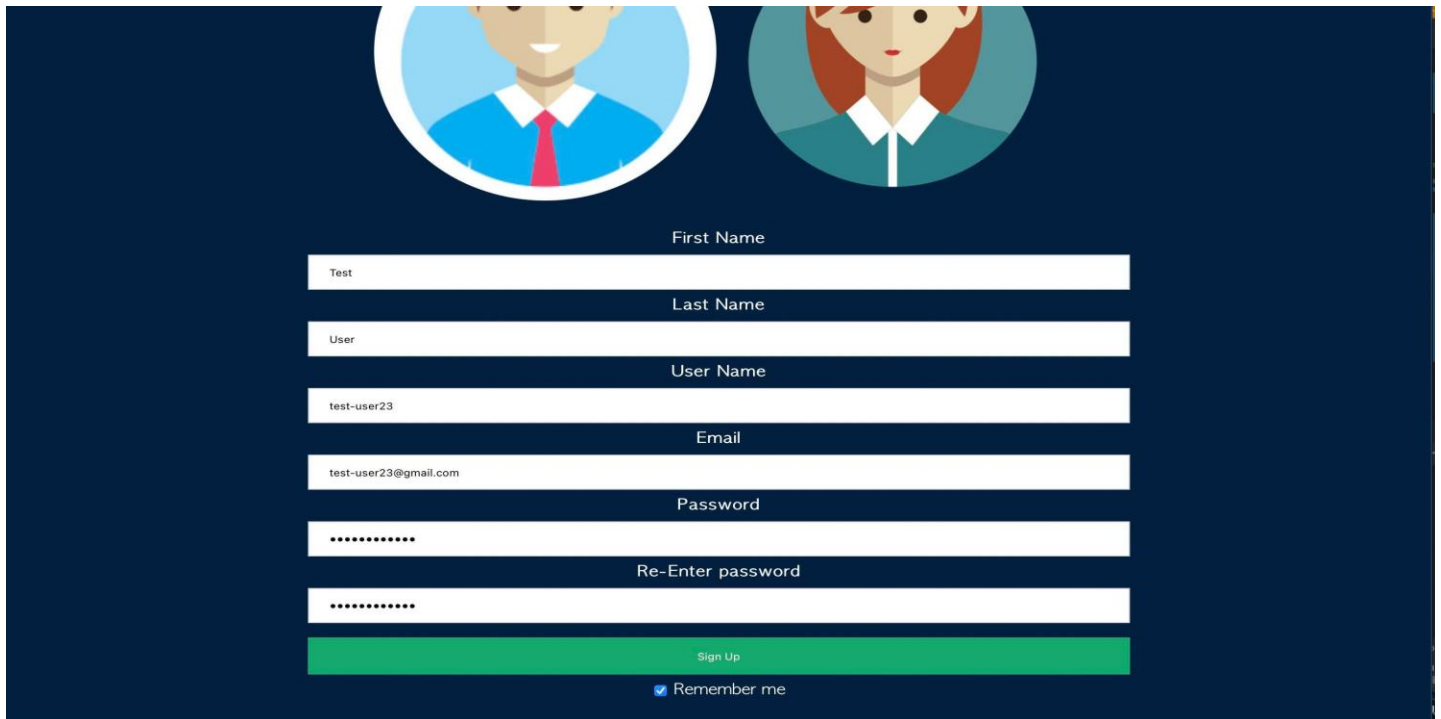
A user registration form on a dark blue background. At the top, there are two circular profile icons: a man in a blue shirt and red tie, and a woman with red hair in a teal top. Below the icons are several input fields: 'First Name' with the value 'Test', 'Last Name' with the value 'User', 'User Name' with the value 'test-user23', 'Email' with the value 'test-user23@gmail.com', 'Password' with masked characters, and 'Re-Enter password' with masked characters. At the bottom, there is a green 'Sign Up' button and a checkbox labeled 'Remember me' which is checked.

Figure 8: Creating a test user to showcase database

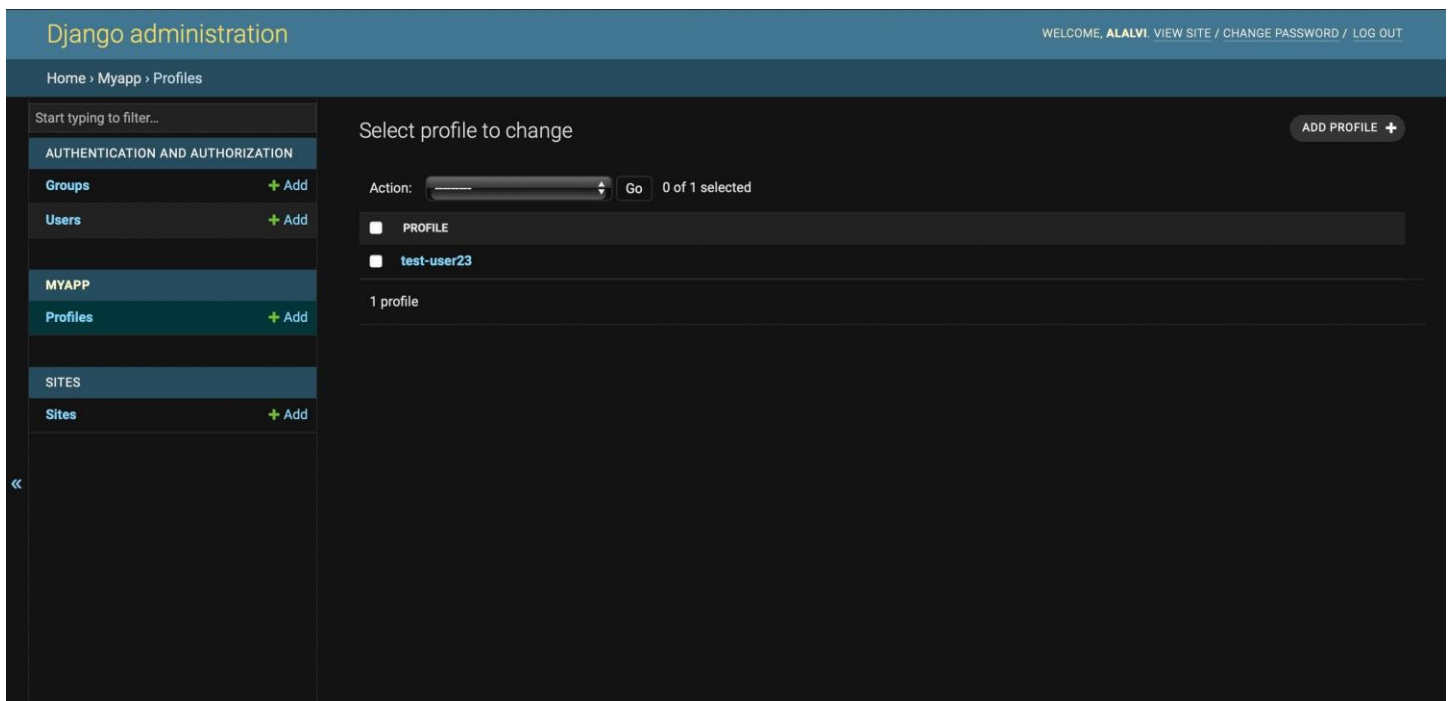
A screenshot of the Django administration interface. The top header shows 'Django administration' and a welcome message for 'ALALVI'. The left sidebar contains a navigation menu with categories like 'AUTHENTICATION AND AUTHORIZATION' (Groups, Users), 'MYAPP' (Profiles), and 'SITES' (Sites). The main content area is titled 'Select profile to change' and shows a list of profiles with a table containing one entry: 'test-user23'. There are buttons for 'ADD PROFILE' and 'Go'.

Figure 9: Django admin page showing profile created

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays a tree view of database objects under the 'public' schema. The 'myapp_profile' table is selected. The main pane shows a SQL query: `SELECT * FROM public.myapp_profile ORDER BY id ASC`. Below the query, the 'Data Output' tab displays the results of the query in a table format.

	id [PK] bigint	first_name text	last_name text	user_id integer	email text	password1 text	password2 text
1	3	Test	User	15	test-user23@gmail.com		

At the bottom of the interface, a status bar indicates 'Total rows: 1 of 1', 'Query complete 00:00:00.247', and 'Ln 1, Col 1'.

Figure 10: Database table myapp_profile

	id [PK] bigint	first_name text	last_name text	user_id integer	email text	password1 text	password2 text	date_created timestamp with time zone
1	3	Test	User	15	test-user23@gmail.com			2022-12-08 22:57:28.160456-05

Figure 11: Database entry for test user

```

class Profile(models.Model):
    first_name = models.TextField()
    last_name = models.TextField()
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    email = models.TextField()
    password1 = models.TextField()
    password2 = models.TextField()
    date_created = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.user.username

    @receiver(post_save, sender=User)
    def update_profile_signal(sender, instance, created, **kwargs):
        if created:
            Profile.objects.create(user=instance)
        instance.profile.save()

```

Figure 12: Model for the profile Link: <https://github.com/alialvi00/JobMate/blob/master/mysite/myapp/models.py>

Figure 8 showcases a user inputting the required fields for a user account creation. Figure 9 shows the Django admin page where we can see the user is registered as a profile is created. To delve deeper and see the information of the user we open the database table called myapp_profile. This database table stores each user in a new row as shown in Figure 10 and 11.

Our profile model has a one-to-one relationship with user model which is a default Django model provided to store users in Django admin page.

This database connection is a vital part of our flow and will be used in future iterations to store user preferences, their resumes, job searches, and saved jobs. There will be more tables added where they will be related with a unique id assigned to each user and with that we can perform relational queries to retrieve the information we need.

The use of databases overall provides us with a full-stack solution that is safe, reliable, and can handle large amounts of users.

4.3: Back-End Progress

The back end of our web application has multiple services each with its responsibilities. In our progress, we have successfully developed two fully functioning job posting website parsers. These parsers are located in the JobMate/mysite/mysite/scrapers folder. The two websites being scraped are well known and are known for hosting most of the job postings that are readily available on the internet. The glassdoor_scraper.py and linkedin_scraper.py maintain the logic for the scraping of Glassdoor and LinkedIn respectively. The Glassdoor scraper code is shown in Figure 13 below and the LinkedIn scraper code is shown in Figure 17.

```
#!/usr/bin/env python3
# linkedin_scraper.py

import requests
from bs4 import BeautifulSoup as soup
import csv
import random

# Create a file called jobs.csv, create writer object to write to file, and set initial headers for csv file
file = open("jobs.csv", "a", newline="", encoding="utf-8")
writer = csv.writer(file)
initial_headers = ("Employer Name", "Job Location", "Job Details", "Job URL")
writer.writerow(initial_headers)

# Setting headers required for http requests
user_agents = [
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",
    "Mozilla/5.0 (Macintosh; Intel Mac OS x 10_15_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36",
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36",
    "Mozilla/5.0 (iPhone; CPU iPhone OS 13_2 like Mac OS X) AppleWebKit/537.51.1 (KHTML, like Gecko) Mobile/15E142",
    "Mozilla/5.0 (Linux; Android 10; SM-G975N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Mobile Safari/537.36"
]

# Randomizing the choice of header to prevent glassdoor from blocking requests
user_agent = random.choice(user_agents)
headers = {"user-agent": user_agent}

# URL below is for now, eventually update with user info in url request
url = "https://www.glassdoor.com/jobposting-autosave-devops-jobs.htm?to=0_a_10228668_487_73.htm?requestCount=6&requestChosenFile=linkedSource=source&linkType=work&software=3264444444"

html = requests.get(url, headers=headers)

# Create beautiful soup object for parsing html returned from requests
bsobj = soup(html.content, "lxml")

# Retrieve all links in glass beautiful soup object to parse
job_links = bsobj.find_all("a", {"class": "job-title"})

# Parsing all the links for information
for i in job_links:
    for a in i.find_all("a", {"href": True}):
        joburl = ("https://www.glassdoor.com/jobposting-autosave-devops-jobs.htm?to=0_a_10228668_487_73.htm?requestCount=6&requestChosenFile=linkedSource=source&linkType=work&software=3264444444" + a["href"])
        result = requests.get(joburl, headers=headers)
        job_bsobj = soup(result.content, "lxml")
        employer_name = job_bsobj.find_all("div", {"class": "css-lm0w0e-align-right-1110-text"})[0].text
        job_location = job_bsobj.find("div", {"class": "css-lv0l0g-align-right-11-text"})[0].text
        job_details = job_bsobj.find("div", {"class": "text css-0h0p0n-align-right-11-text"})[0].text
        file = open("jobs.csv", "a", newline="")
        job = (employer_name, job_location, job_details, joburl)
        # Writing all scraped information to csv
        writer.writerow(job)
```

Figure 13:Glassdoor Scraper Code (Link: https://github.com/aliavi00/JobMate/blob/master/mysite/mysite/scrapers/glassdoor_scraper.py)

This scraper utilizes the BeautifulSoup [12] library from bs4. The library is meant for retrieving data from HTML files which in our case are the source HTML files of the pages that we are requesting to scrape. To retrieve the HTML files the requests library [13] is used. Requests is an HTTP library that allows you to send HTTP requests without the use of query strings or PUT and POSTS.

In lines 6-10 a file is created called jobs.csv in which some information is set, shown in the Figures below. To perform the HTTP requests the websites, require user agents. User agents are headers that are “characteristic strings that lets servers and network peers identify the application, operating system, vendor, and/or version of the requesting user agent.” [14]. Since we want to be able to submit requests without the risk of being blocked from the websites, the randomization of header choice is used to be submitted into the requests.get method. The returned Html is then stored in the Html variable in line 27.

Due to the current progress, the system does not yet accept user resumes/input on desired jobs, so the current scrapers have the URL hardcoded into the code. In line 26 the URL is seen and the changing of searches is displayed in Figures 7 and 8 with the associated changes in the links. The link in Figure 14 and 15 is found in the references [15], [16].

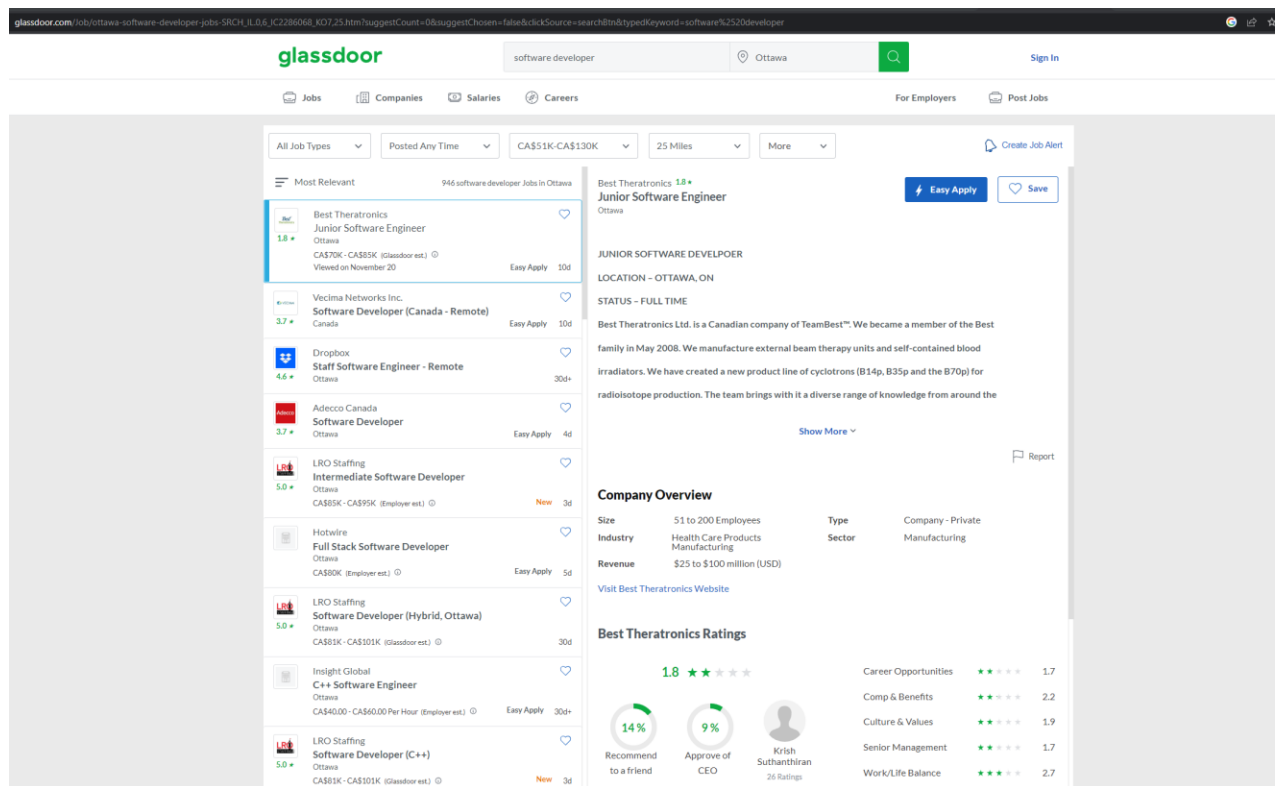


Figure 14: Software Developer Job Search

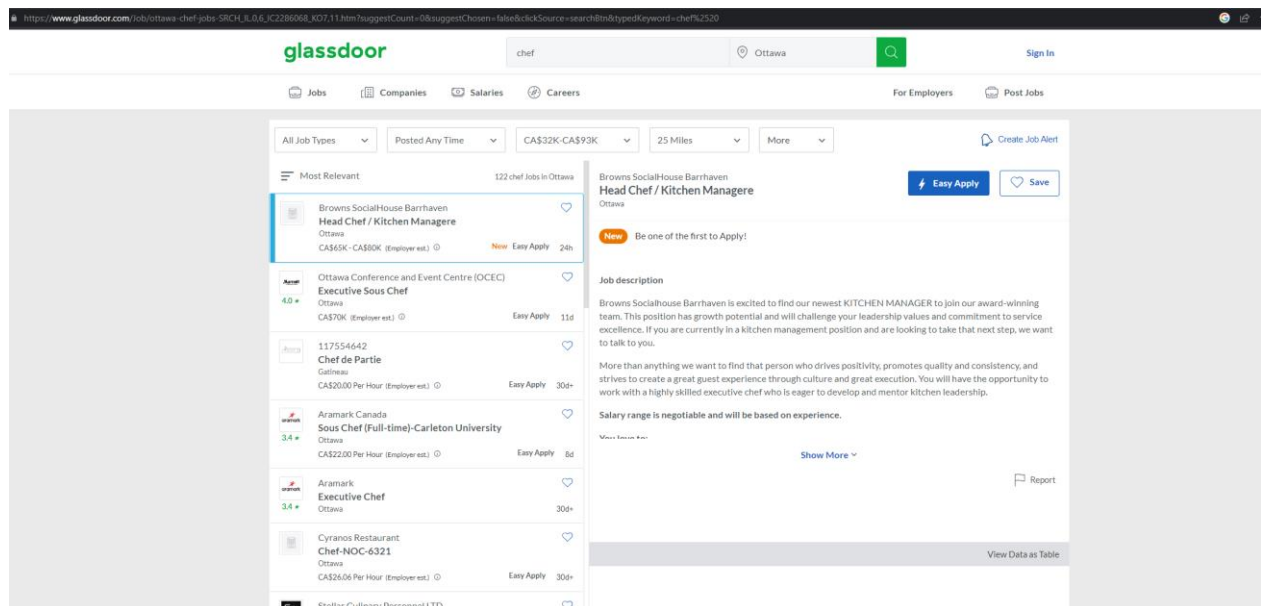


Figure 15: Chef Glassdoor Job Search

This URL request is the format that we will follow, as seen in the software-developer-jobs and the keyword component can be changed to the desired job title that the user wants in further iterations. In figure 8 it is changed to a job search for Chef, and we receive the list of Chef jobs in the Ottawa Region. We will simply change the URL requests that our program will generate for us based on user input or scanned from the resume. This choice will also be given to the user.

Figure 18 shows the output CSV from the LinkedIn Scraper.

	Job Title	Employer Name
1	Class	What You'll DoOur software engineers are the gurus behind the scenes, ensuring our programs are easy to use and bug-free. Using a keen eye, you'll develop software and tools in support of many of our high-misc technology platforms such as operating systems, networks, databases and more. While we're growing in the software business, you'll need to see the big picture and watch for hardware compatibility with even potentially influencing design choices where to
2	SWIFT Energy	Company DescriptionSWIFT Energy is proud to be a Mariner company.We're part of the Mariner group of companies, representing 3 energy owned divisions and a portfolio of 30 start-ups. Mariner is 100% employee owned, which means we're driven by our longer-term vision - not just quarterly financial results. We're about to enter our 20th year of operation.SWIFT is focused on reducing the carbon footprint of large buildings - the invisible driver behind 25% of the world's c
3	Ray Video	Software Developer - Video ID Job Req 4022-360Why Work at Ray Video? We have a great group of people working together to create and deliver cutting edge products that look amazing and are easy to use. We go all in so that our customers can have the best possible experience and achieve quality results. With a product focus, continual learning, results driven processes, and creative thinking, we constantly drive to improve our solutions and to deliver results. If you
4	Redwire Technology LLC	Job DescriptionWe're a Software Developer - Ottawa, ONCompany DescriptionRedwire Technology Canada is an application software consulting and systems integration company in British Columbia, Canada. We deliver software solutions that improve the lives of people in need throughout Canada, the United States, and the Caribbean by means of building new applications, implementing cloud and packaged systems, and/or modernizing legacy systems. We are focused
5	Hard Motor Company	Job DescriptionWe're a Software Developer - Ottawa, ONCompany DescriptionHard Motor is a leading provider of custom software solutions for the manufacturing industry. We are currently seeking a Software Developer to join our team. The successful candidate will be responsible for the design of the automation framework and architecture. The candidate will be involved in building and maintaining the software solutions used to execute functional, integration and regression test suites. Also, the candidate will be interacting with the development team to ensure that the products meet set quality standards throughout the software development lifecycle. The scope of
6	Thoughtline	Thoughtline's smart building and smart hospital software solutions empower clients in commercial, real estate and healthcare to optimize building operations and improve account experience by orchestrating data from people, processes, and the connected building in a smart digital asset. The Built Environment Digital TeamThoughtline's innovative approaches that we designed in the cloud and on-premise, are powered by Thoughtline's Built Environment Digital T
7	Hard Motor Company	Job DescriptionWe're a Software Developer - Ottawa, ONCompany DescriptionHard Motor is a leading provider of custom software solutions for the manufacturing industry. We are currently seeking a Software Developer to join our team. The successful candidate will be responsible for the design of the automation framework and architecture. The candidate will be involved in building and maintaining the software solutions used to execute functional, integration and regression test suites. Also, the candidate will be interacting with the development team to ensure that the products meet set quality standards throughout the software development lifecycle. The scope of
8	US Tech Solutions	US Tech Solutions is seeking a Software Developer for one of our biggest Financial Services clients in Canada.Job Title Software DeveloperLocation Ottawa Office, Ontario - Fully RemoteLength of Assignment: 12 Months with 24 month extension possibilityJob Type Full Time ContractClearance Required: Secret Security ClearanceDescriptionWe are looking for a highly qualified, resourceful, and talented Software Developer for one of our well established clients. This p
9	Class	SWIFT YOU'D DO Class is funding a small, agile, product development team to innovate and build a new product offering in the SDN controller space. The team will operate in an agile development mode to rapidly bring a new product offering to market, reacting very quickly to initial customer feedback and market needs. Over time as customer adoption increases, the platform will be further utilized as the basis for many network applications.Who You'll Work WithClass is a
10	Hard Motor Company	Job DescriptionThe architecture and Software Platform group is responsible for the software platform that enables connectivity and data exchange between various vehicles within our generation fleet vehicles. The team is designing and building the real generation software platform for the PaaS service architecture structured around high-performance compute nodes and data aggregation flowing as a member of the Real Automation team. The successful candidate will be involve
11	Strut	Career Growth, Flexibility and CollaborationStrut is dedicated to securing a world in motion by enabling trusted identities, payments, and data protection around the globe. Headquartered in Toronto, we offer our colleagues the ability to work globally, in a flexible and collaborative environment. Our team makes an impact!The Company Strut is an on-curve, dedicated and innovative individuals whom anticipate the future and provide solutions for a more connected
12	Navi CANADA	Working as part of the BBS A&M Billing and Fight Group you will be one of the key developers of a new web portal solution at NAV CANADA. In addition, you will be heavily involved in the life cycle support of some of our agency systems doing root cause analysis and bug fixing using Microsoft Azure Cloud platform day AccountabilityDevice, infra, and test applications, hardware interfaces, systems interconnection software, identify, define and correct maintenance problem
13	Bentley Systems	Bentley Systemization SummaryBentley Systems, Inc. is looking for an IT team, experienced Software Engineer for the Commercial Program Portfolio Team. As a member of our team, you'll have the opportunity to learn and work with cutting edge cloud platform with Microsoft Azure and the latest AI technologies. You will be instrumental in building robust and scalable fleet components and backend API for our fleets. In this exciting and challenging role, you will work
14	Class	What You'll DoClass is funding a small, agile, product development team to innovate and build a new product offering in the SDN controller space. The team will operate in an agile development mode to rapidly bring a new product offering to market, reacting very quickly to initial customer feedback and market needs. Over time as customer adoption increases, the platform will be further utilized as the basis for many network applications.Who You'll Work WithClass is a
15	Nokia	Come create the technology that helps the world act together! Nokia is committed to innovation and technology leadership across mobile, fixed and cloud networks. Your career here will have a positive impact on people's lives and will help us build the capabilities needed for a more productive, sustainable, and inclusive world!We challenge ourselves to create an inclusive way of working where we are open to new ideas, empowered to take risks and fearless to bring our autho
16	Amadeus	Software Engineering Canada - Ottawa/Jessica Day: Negotiate are well! you're a team player and then you are part of an ever more connected and digital world. At Amadeus, we are leading the digital revolution into the future. From virtual team foundations, releasing Big Data and Market of Things to mobile finance services, bring and operations support systems, we are continually growing our business to help you achieve new successes. We make sure
17	Diverse Lynx	Job DescriptionTitle QA Engineering- RMOThe QA engineer will be responsible for the design of the automation framework and architecture. The candidate will be involved in building and maintaining the software solutions used to execute functional, integration and regression test suites. Also, the candidate will be interacting with the development team to ensure that the products meet set quality standards throughout the software development lifecycle. The scope of

Figure 18:CSV Output from LinkedIn Scraper

The next component of our back end is the User inputted resume parser service. This service is very crucial as it extracts the key data from our user's resume such as experience, education, etc. For this service, we are utilizing an open-source API [17] that we will call from our back end submitting the user's resume as input. This parser has custom-trained models to extract that are capable of recognizing certain fields and extracting them. This API fits the needs of our resume parsing and is perfect for implementing into our solution.

Our Django project simply sends a POST request to 127.0.0.1:2000/sendResume and passes a base64 of the resume as a parameter. The ResumeParser receives it using the receiveResume function and converts back to pdf. The Django app then receives a success message from the API to confirm the resume was received by the parser. Django then sends a GET request to 127.0.0.1:2000/parseResume which invokes the parseResume function in resumeParser and returns the parsed resume as a JSON object with all parsed fields. The API is set up to run the resume through the trained machine learning algorithms and processes the resume. The code for the receiveResume and parseResume functions are shown below in Figure 19.

```

1 from flask import Flask, request
2 from pyresparser import ResumeParser
3 from base64 import b64decode
4
5 app = Flask(__name__)
6
7
8 @app.route('/sendResume', methods=['POST'])
9 def receiveResume():
10     resume_in_b64 = request.get_data()
11     resume_in_bytes = b64decode(resume_in_b64, validate=True)
12
13     # Validate it is pdf file
14     if resume_in_bytes[0:4] != b'%PDF':
15         raise ValueError("Missing the PDF file signature")
16
17     resume = open('resumeReceived.pdf', 'wb')
18     resume.write(resume_in_bytes)
19     resume.close()
20     return "SUCCESS"
21
22
23 @app.route('/receiveParseData', methods=['GET'])
24 def parseResume():
25     parsed_resume = ResumeParser('..resumeReceived.pdf').get_extracted_data()
26     return parsed_resume
27
28
29 if __name__ == '__main__':
30     app.run(debug=True, port=2000)

```

Figure 19: receiveResume and parseResume functions from the pyresparser API (Link [17])

The POST request with the URL is shown below where we pass the base64 code of a sample pdf resume.

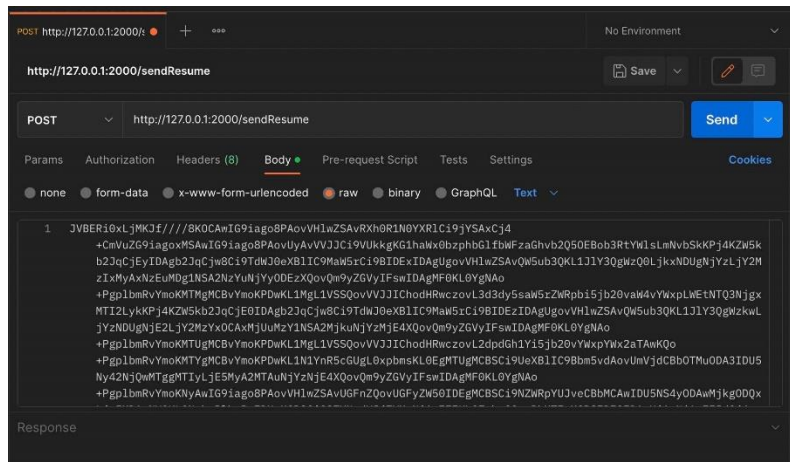


Figure 20: POST request on Postman (Link not available as postman request check out [17])

The success message shown in Figure 21 is the confirmation that the resume was received by the parser. The PDF is then created, and we send a GET request and receive the parsed resume as a JSON object shown in Figure 22.

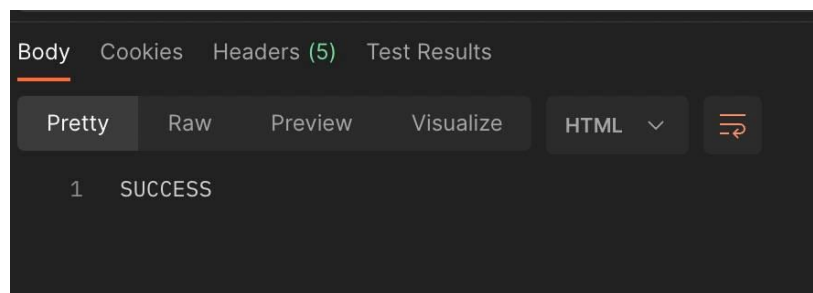


Figure 21: Confirmation of Success from POST (Link not available as postman request check out [17])

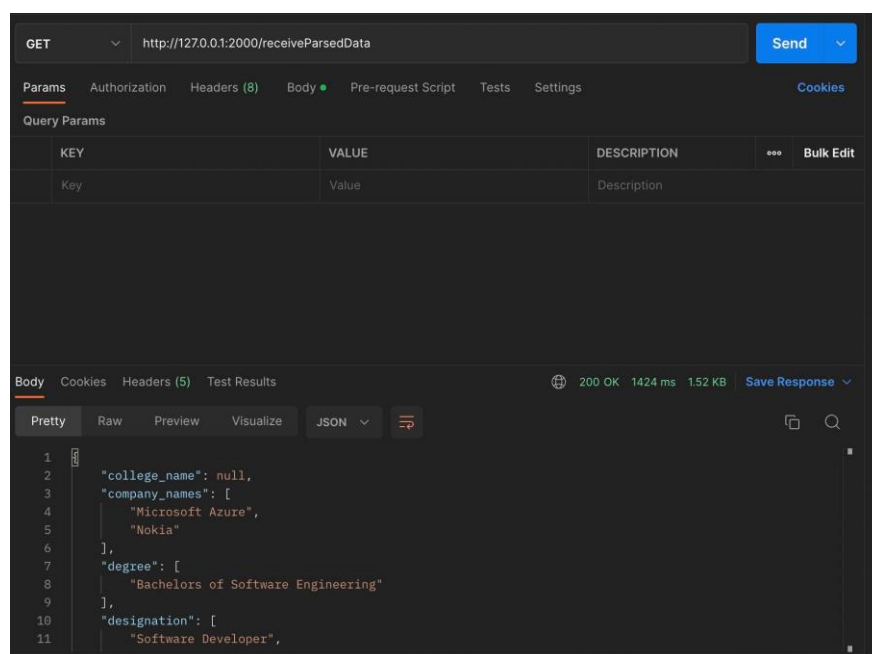


Figure 22: Returned JSON Object (Link not available as postman request check out [17])

Chapter 5: Reflections

As with any project with multiple components and team members of varying experiences, obstacles and difficulties arise. In our current progress, we have not yet hit any major roadblocks. The only disruption we experienced was during a reading week where we expected to do a lot of the major milestones, but we were faced with issues such as illness, family problems, and timing. Our communication thus far has been great, and all team members can complete their tasks on time.

5.1: Mitigation Strategy Application and Objective Alterations

As explained in our Proposal for this project, we have certain risk and mitigation strategies. These are outlined below:

A) Risk: Communication between teammates

A) Mitigation strategy: Discord, messaging apps, slotted meeting times every week, milestones, and constant communication throughout development and testing phases to avoid issues and complete milestones on time. Section 1.2 has details in terms of mitigation strategies for miscommunication and facing problems as a team.

B) Risk: 3rd Party API Maintenance

B) Mitigation Strategy: We will be utilizing well-known website APIs that have a track record of staying online and being reliable

C) Risk: Coding Skill/Competencies

C) Mitigation Strategy: Discuss the use of technologies that are already in the skillset of all team members or learn new skillsets required for each milestone and implement them together.

D) Risk: Sickness

D) Mitigation Strategy: Rescheduling deadlines and dropping less important features such as grammar checks.

E) Risk: Technical Difficulty

E) Mitigation Strategy: Rescheduling deadlines, dropping less important features such as grammar checks.

As we faced the sickness risk, we performed our mitigation strategy of rescheduling deadlines. However, we were still able to complete our milestones in time.

5.2: Upcoming Work and Possible Risks/Issues

The Milestones left in our work are listed below:

Milestone 7(Ali, Areeb, Ahmad): Integrating components into one application: December 14th, 2022

Milestone 8(Ali, Areeb, Ahmad): Final touches and Bug fixes: January 10th, 2022

Milestone 9(Ali, Areeb, Ahmad): Final product launch: TBD

Much of the work will be connecting all of our little components and services to work efficiently and effectively. In terms of possible issues, we may face during the execution of the programs all simultaneously running together. For this, we will be looking into how Django projects manage multiple services in the back end and possibly monitor these services using a monitoring system such as Grafana or Prometheus.

Right now, we are in discussions as a team to change some of the upcoming dates as we have been very busy with examinations and final labs, assignments, and projects due for our courses. This will be reflected in our presentations in January.

5.3: Conclusion on Progress

To conclude, our team is moving in the right direction as we are completing our Milestones on time and have the plan to tackle the rest of the milestones. This means we are confident our planning and organization of requirements was as close to accurate as possible. If we were to start the project again the only thing we would change is make more time in between iterations as the expected workload of this semester was much higher than we thought. We have a solid back end and front-end setup and are awaiting our final few components to join it all together into a fully functioning proof of concept. Once our web app is deployed and working as expected we will monitor the service times and memory usage and refactor our code accordingly. We expect to keep moving at this pace as we are going into the second semester and believe we will accomplish our targets on time.

References

- [1] "8 Things You Need to Know About Applicant Tracking Systems." Jobscan.co, www.jobscan.co/blog/8-things-you-need-to-know-about-applicant-tracking-systems/.
- [2] "Computer-related Injuries." Better Health Victoria, www.betterhealth.vic.gov.au/health/healthyliving/computer-related-injuries#:~:text=Summary&text=Working%20at%20a%20computer%20can,posture%20and%20good%20working%20habits.
- [3] "Code of Ethics." PEO.on.ca, www.peo.on.ca/licence-holders/code-ethics
- [4] "What is Agile?" Atlassian, www.atlassian.com/agile
- [5] "Kanban Boards." Atlassian, www.atlassian.com/agile/kanban/boards
- [6] "About Version Control." <https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control>
- [7] JobMate." GitHub, <https://github.com/alialvi00/JobMate>
- [8] "Django Project MVT Structure." GeeksforGeeks, www.geeksforgeeks.org/django-project-mvt-structure/
- [9] "MVT Structure Image." GeeksforGeeks, <https://media.geeksforgeeks.org/wp-content/uploads/20210606092225/image.png>
- [10] "Monolithic Architecture Definition." TechTarget, www.techtarget.com/whatis/definition/monolithic-architecture
- [11] "Microservices vs Monolith." Atlassian, www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith.
- [12] "Beautiful Soup Documentation." Beautiful Soup 4, <https://beautiful-soup-4.readthedocs.io/en/latest/>
- [13] "Python Package Index: Requests." PyPI, <https://pypi.org/project/requests/>
- [14] "User-Agent." Mozilla Developer Network, <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent>
- [15]"Software Developer Jobs in Ottawa, ON." Glassdoor, www.glassdoor.com/Job/ottawa-software-developer-jobs-SRCH_IL.0,6_IC2286068_KO7,25.htm?suggestCount=0&suggestChosen=false&clickSource=searchBtn&typedKeyword=software%2520developer.
- [16]"Chef Jobs in Ottawa, ON." Glassdoor, www.glassdoor.com/Job/ottawa-chef-jobs-SRCH_IL.0,6_IC2286068_KO7,11.htm?suggestCount=0&suggestChosen=false&clickSource=searchBtn&typedKeyword=chef%2520
- [17]"pyresparser." GitHub, www.github.com/OmkarPathak/pyresparser?fbclid=IwAR0kVXgj6mp27EcaBsSW0JDIYhpaf4X3O_BS297EIJo9JmminG-tcwwwWE
- [18] Django Project." DjangoProject, <https://www.djangoproject.com/>
- [19] "PostgreSQL." PostgreSQL, www.postgresql.org/.
- [20] "Python Programming Language." Python, www.python.org/.

JobMate: A Web Application for Resume and Job Application Improvement

Proposal

Supervisor: Dr. Lynn Marshall

Team Members:

Areeb Haq 101115337

Ali Alvi 101114940

Ahmad Abuoudeh 10107236

Table of Contents

1.0: Introduction
1.1: Statement of Objectives
1.2: Plan to Progress Towards Objectives
2.0: Background
2.1: Current State-Of-The-Art Model
3.0: Description
4.0: Group Skills and Education, Relation of Project to Degree Program and Collective Skills
5.0: Methods and Relation of Methodology to Degree Program
6.0: Proposed Timetable and Milestones
7.0: Risks and Mitigation Strategies
8.0: Special Components and/or Facilities
9.0: References

1.0: Introduction

This document outlines the information required for 4th year project proposal. This proposal is structured to provide an in-depth insight into the JobMate project that Team 23 is working on this year.

1.1 Statement of Objectives

The aim of this project is to develop a web application that will act as a tool automate parts of the job search processes whilst being an excellent aid to users for improving their respective resumes in both structure and content. The application will accept user resume/cover letter and retrieve input from the user about certain information regarding the application processes. This information includes which of the many available job application websites the user would like to use, the field of employment they wish to search for, location, and level of seniority of the position. These choices will be made through radio buttons and will be easy to navigate through. Once the personalized options have been chosen for the user, the application will then perform back-end logic to retrieve jobs from multiple websites that match the criteria given by the user. With these jobs there will be an information section which will give details about the percent match of the resume and what skills were relevant enough for the match to occur.

For further improvements to the resume, the application will provide a list of certain jobs that could have been met by the users resume but fell short of the percent match threshold that was set. This way the user will be given an information section that explains what fields/skills can be added to the resume to improve their resumes to match more available jobs. Grammar checks will also be done through the application using backend external api's and the system will give a rating of the resume to the user with certain rules that resume should follow. We want this application to be the one stop shop for any user with the desire of improving their resume or currently seeking new employment opportunities.

1.2: Plan to progress towards objectives

Our team has opened a GitHub repository [4] which we will be using as our versioning and collaborating system. We have broken up the major features of the project into parts that have been assigned to each member correlating with the strengths/weaknesses of said team member. We have weekly meetings as a group with our supervisor in which we will be discussing our current progress, roadblocks, and any questions that may arise. Aside from this our team has also opened chat channels in Discord [5] and other messaging apps to have a constant flow of communication between team members. This way if any issues come up it is up to the team member to communicate it through our various modes of communication to find a solution. If a problem is not able to be resolved on our own we can then bring this issue to Dr. Lynn for any further feedback or suggestions. Each team member is responsible for developing readable and maintainable code with comments to allow for further improvements by other team members.

In terms of development, as a team we will begin a proof of concept to prove that the objectives are both obtainable and maintainable in a live web application environment. The POC will be a simple website that will fulfill the acceptance criteria. This is a reduced scope of work however it must still include all the major components such as resume parsing, job search according to fields, and resume improvement recommendations which will make this application unique and useful. The POC will prove the use of the proposed API's or headless chrome selenium scripts, back-end integration of databases, and other complex software components. To perform this POC we will use dummy resume's and job applications and prove out the concept.

For this project we will be following an Agile approach, in which we are going to be doing weekly meetings to discuss our progress. This is the approach we followed in our previous university projects and professional work environments.

In the case of one of the team members facing a issue that will affect the progress, will be offering to help with that issue to progress further in the project so no deadline is missed.

If there are any impediments holding the whole team back from progressing and no solution can be found online, refer back to our lectures and notes from previous classes and also consult Dr. Lynn Marshall for direct help.

The steps we are going to take when facing a problem are

- Define the problem
- Determine the effect on the overall progress of our project
- Brainstorm ideas individually and as a group
- Select the most reliable solution for now and the future
- Determine an action plan to implement the agreed upon solution

These steps have been agreed upon by all team members and are an important part of problem solving while doing a project of this size and complexity. We have been involved in team projects together as a team before and this has helped us to dissect big problems into smaller more manageable issues. In return this process will allow things to flow smoothly throughout the term whilst also saving time.

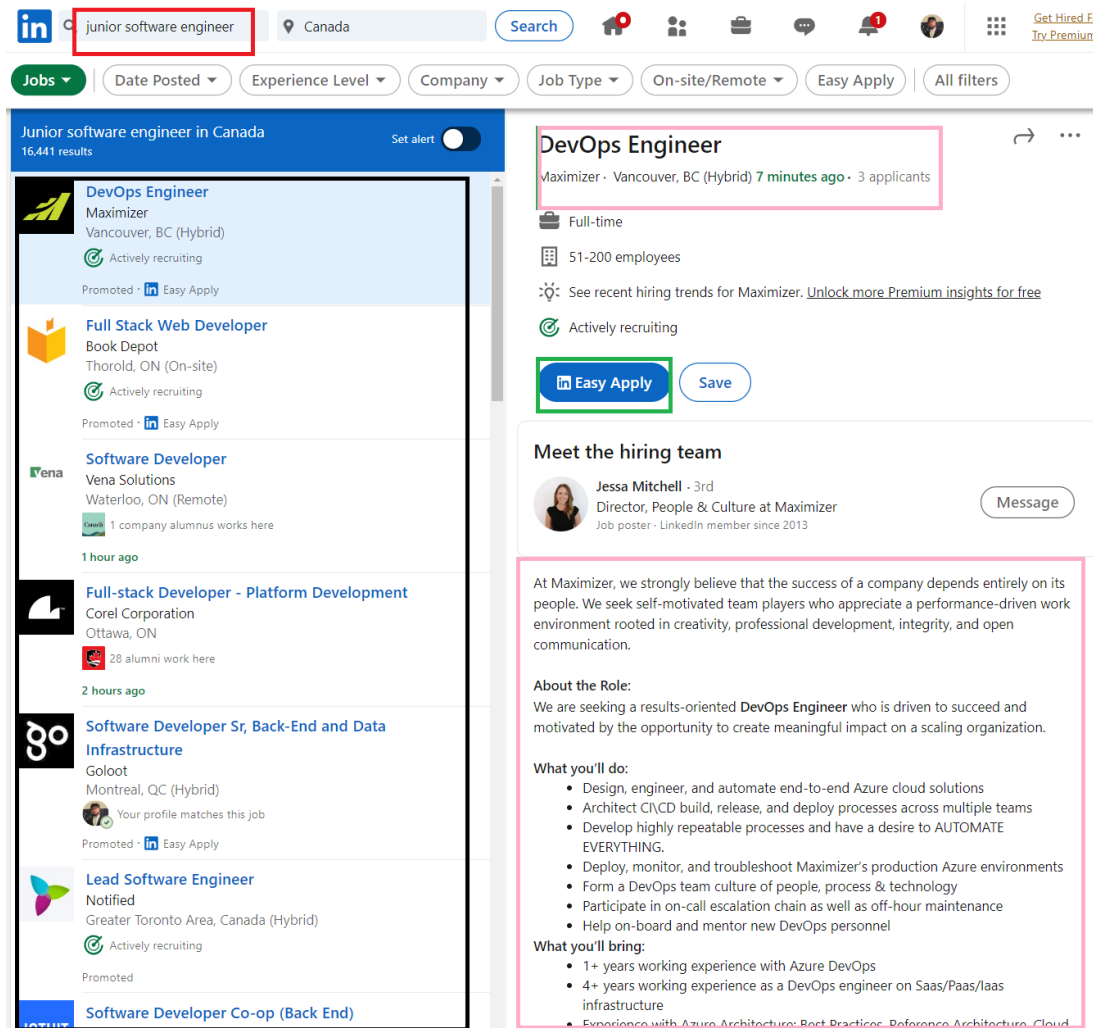
2.0: Background

The job search is a process that millions of people deal with every year. There are a wide range of applications and websites that people utilize in order to acquire a new position. In recent events of the Covid-19 pandemic a wide range of opportunities have opened up with new jobs that are accessible to people nation wide. The remote access of jobs has opened up the job market to people and to apply to these jobs the majority of applications are done through online job postings. Websites like Indeed [6], LinkedIn [7], Workopolis [8], etc. have seen a huge increase in traffic. These websites either directly accept user job application materials and send it to the companies or link the user to the company's internal job posting pages.

This process lacks the feedback that users need to improve their current resumes with keywords and experiences. A statistic from the popular company rating website. Glassdoor states that "On average, each corporate job offer attracts 250 resumes. The typical employer will then interview 4–6 candidates for the job, and only one will be successful." [1]. This means that it is crucial for applicants to thoroughly review their resume for any improvements in grammar, structure, and content. Many job application websites utilize a system called ATS short for Applicant Tracking System [9]. A Capterra statistic shows that "75% of recruiters and hiring professionals use a recruiting or applicant tracking system." [2]. This system is responsible for removing applications that don't match certain requirements in key words in documentation, format, or structure.

2.1: Current State-Of-The-Art Model

The current model as it stands allows users to create an account on the website i.e., LinkedIn, upload your resume, search, and apply for jobs. In the example below the process for applying to a job in LinkedIn is shown:

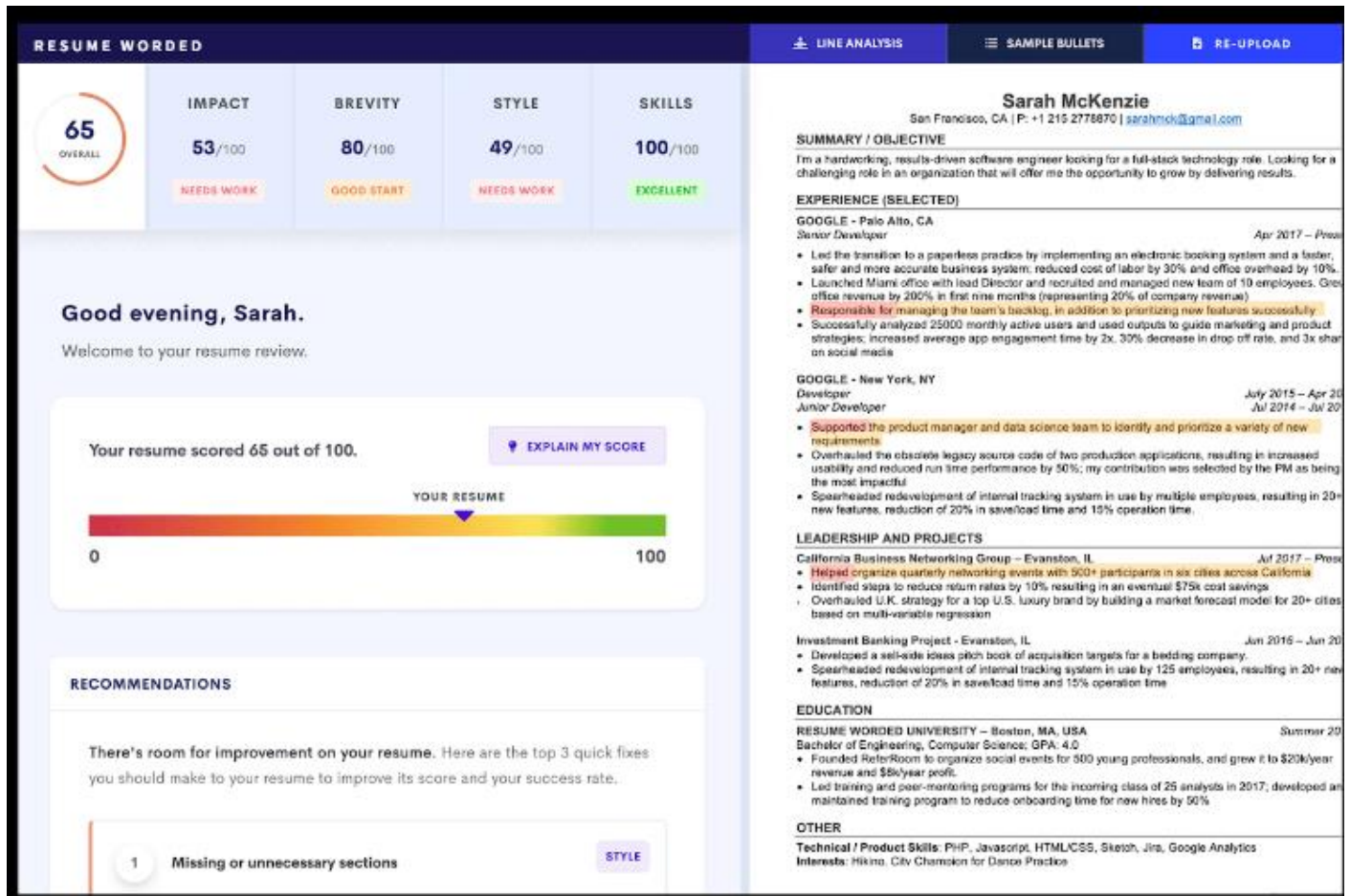


In the image above is a direct screenshot of what a user will see while applying for jobs on an online platform, in this case LinkedIn. In the red box, is a query made for junior software engineer postings with the location of the job right beside it. In the black box is the jobs that show up based on the search criteria given. In the Pink boxes is the current job posting that the user is on and the information about it such as job title, location, time of posting, # of applicants, and description. Finally, in the green box is the current job the user can apply to by hitting Easy Apply. This is already a very seamless process and has very little room for improvement thus making sense to utilize APIs offered by these companies instead of developing our own job posting webpage. If API's are not available we can develop our own python scripts or selenium web implementations

What this model lacks are the ability to offer any feedback and suggestions to the user on how well their resume has matched this job description, if these jobs are even relevant to the experience the user currently has,

what experience/skills the user would need to gain and add to the resume to meet the standards set by the job description, any grammar or structural flaws that may be present in the resumes.

There are also applications on the internet that rate resume's and provide feedback on the structure, formatting, and grammar of any inputted resume. An example of this would be resumeworded.com [10] as shown below.



Websites like these are great resources that are current state of the art for users to input their resumes and receive feedback on how to improve the document in grammar, format, and structure.

With all these given applications readily available both free and paid, there is a lack of cohesion of these services in to a one stop application that does all this. LinkedIn does not provide meaningful feedback on the resume, while websites like resumeworded.com do not provide feedback based on the actual jobs that are available on the market at the given moment. The feedback on the resume is solely based on the back-end rules set up by the website which means it does not improve the persons resume based on actual job postings.

3.0: Description

We are going to develop a fully functioning web application from start to finish. The back-end coding of this web application will be done using Python [11] and the Django framework [12]. Python is our programming language of choice as it has many characteristics that are advantageous to our development. Some of these advantages include being scalable and adaptable, easy to prototype with, ease of data collection through web-scraping, plug-ins and APIs, and database connectivity [3]. These are all features we require to complete the objective of this project.

Django is a high-level Python web framework that allows for development of web applications with security, speed, and scalability in mind. Our users will definitely require a secure system as we are handling critical information about them through their resumes. Speed is also very important as we want to retrieve jobs and rate resumes quickly.

Firstly, our website HTML [13] and CSS [14] formatting will be based on current online HTML templates that are available for free and we will customize them to our liking. These customizations will offer us a clean and usable landing page all the while having our own requirements fulfilled. The landing page will have a secure and fast user account creation system in which basic user information will be taken. A user account will be necessary in order to maintain the resume and past results of job searches. However, the application will also allow for a user to access the application as a guest which will offer a one-time job search and will not store the resume or any results that come from a given job search.

Once a user has access to the application through an account or a guest access, they will be asked to upload their resume using a file explorer system or a drop box. If a user account has been created that account and the information regarding the account + resume will be stored in a PostgreSQL database [15]. For guest accounts, the resume will be stored in a temporary database for usage and will be discarded of after the user has left the website. PostgreSQL is fully supported by the Django plugin and connectivity to the database is very easy. PostgreSQL is also known for its Open-Source community, security, and scalability. These are all valuable characteristics to the development of the JobMate application.

The user is then redirected to a page where they will be given a few options to chose through about the type of job they are looking for and which websites that the application should scan that match those characteristics. The application will then return a list of jobs and provide information as to how well your resume that is stored in the database matches the given job descriptions as a whole. It will also provide a suggestions box which will give the user some input on what skills the resume is lacking and how much more it would increase the resume score by.

If a user has an account, we can maintain these scores and information on hand and the user can perform multiple searches to see how well their resume has increased over time and what additions are left to be made.

Finally, the JobMate website is going to be a one stop application for students, new grads, or people in their careers who are wishing to find new employment opportunities. This application will provide a place for the user to look for jobs through multiple major job posting websites, apply through the application or be provided a link to the internal job postings, and receive constructive feedback on the effectiveness of the resume and possible further improvements.

4.0: Group skills and education

For this project, we are going to be using various methods that we learned through different classes during our time at university.

One of the first methods we are going to be using is objected-oriented programming (SYSC2100) which was taught by Dr. Marshall. The course covered the different abstract types and their usage in Java classes, some examples of those abstracts are Stacks, queues, linked lists, and trees; those abstract types are going to help in this project by giving us various options which could be useful later while creating methods and how we going to be able to store accounts.

Another class we are going to use methods for this project is the Software Development Project (SYSC 3110) which was taught by Babak Esfandiari. Some of the methods that will help us are how we going to be developing and design pattern we are going to be using for our project such designs are the Model View Controller (MVC), Composite, etc. For this project, we are going to follow the MVC pattern since it is the most reliable method, due to its versatility and if there are any changes in the future you wouldn't need to change the entire model.

Another class in which we are going to use its methods for the front end is "Fundamentals of Web Development (SYSC 4504)" which was taught by Thomas Kunz. The course helped us by understanding the basics of HTML, CSS, JavaScript [16] and how those three languages can be combined, and how we can make them work together, this course method will help us drastically since most of our front end would be done with those three languages.

Some of the other classes in which its methods are going to be helpful are "Project Management (SYSC 4106)". This course would guide us in which kind of way we are going to keep track of our project work and make sure everything is going according to the deadlines. We are going to follow the agile methodology which will be doing weekly meetings in which we are going to discuss our progress and any impediments we can be facing and where are we in terms of project completion.

One final course we are going to be using its methods for this project is COMP3005 Database Management Systems. This course emphasizes database management system and how we can save data in the database and use it later in our project. The course also taught us the programming language SQL [17] which is the programming language mainly used for databases.

The final course we going to be talking about is SYSC 1005. The course was introductory to the programming language Python, which is going to be our main language of programming for this course. The course also introduced us to dictionaries [18] which is an important feature in our project since will keep track of the account using dictionaries [18].

5.0: Methods and relation of methodology to degree program

The objective of developing a web application that helps students and professionals apply for jobs tailored for their skills requires a specialized skillset present in students that are in the field of software engineering. Our group of three students with each student studying and succeeding in the software engineering program gives us the leverage to create such an inspiring web application that can help shape the future of students and professionals.

First student in our group, Ali Alvi, has the experience of working on the frontend of various web applications due to his coop experience. Ali has gained certain skills related to the front-end development that is only achievable through professional experience. Ali's expertise in front end development will give our group a boost in creating a user-friendly interface and improve user experience (UX).

Our second member, Areeb-UI Haq, has the experience of developing back-end systems for web applications as well as REST API's [19]. With this sort of experience coupled with UI design experience at Kinaxis, he is able to provide aid in the full-stack solution required for this project. Areeb also has real world experience helping students build their resumes which will prove to be valuable when designing the logic behind the web application. Areeb's experience will also give us the correct direction in making the web application so that it fits the needs and requirements of our potential clients.

Our third member, Ahmed Abuoudeh, has great experience in handling the backend and framework of web applications due to his extensive coop experience at Motorola Solutions. Ahmed's ability to create and implement astound logic in web applications and to construct a bridge of communication between backend and frontend will prove to be valuable when we will need to make our web application interactive.

We believe that as a group, our experience and understanding of complex software concepts will help us succeed in this objective of deploying a platform for users to apply for jobs and improve their resume. Our group has a great amount of professional experience which will prove to be helpful in setting up regular meetings and meeting deadlines in timely manner. Our group has the common skills of understanding the syntax of the programming language we will use, the concept of the web application that is being developed and the understanding of how Django projects work.

Communication in the group is also very important to produce a valuable project deliverable. Our team has split up tasks depending on strengths and weaknesses as well as interest in certain aspects of the project. For example, in this report, the sections were split up evenly amongst the three group members. Moving forward, tasks will also be split up to ensure a smooth flow of the development. The splitting of work is further discussed in Section 6.0.

These experiences combined are also heavily tied to the degree program of all three team members which is Software Engineering. As discussed in section 4 and 5 of this proposal, it is evident that the wide variety of courses and experiences are easily linked to each other and will certainly aid in the eventual development of this system.

6.0: Proposed timetable and milestones

Milestone 1 (Ahmad): Creation of landing page and front-end UI: October 7, 2022

Milestone 2 (Ahmad): User account creation and database connection: November 18th, 2022

Milestone 3 (Areeb): Resume box upload page with database connection: November 18th, 2022

Milestone 4 (Ali): Resume Parser API: November 18th, 2022

Milestone 5 (Areeb): Application using LinkedIn, Indeed, Workopolis API's or Selenium Scripts: November 18th, 2022

Milestone 6 (Ali, Areeb, Ahmad): Resume Rating Service (grammar, structure, format) November 30th, 2022

Milestone 7 (Ali, Areeb, Ahmad): Integrating components together into one application: December 14th, 2022

Milestone 7(Ali, Areeb, Ahmad): Final touches and Bug fixes: January 10th, 2022

Milestone 8(Ali, Areeb, Ahmad): Final product launch: TBD

** These dates are estimated and may be changed according to assessment of requirements, and risks.

** The milestones are written in order of priority and Milestone 2-5 are key components which should be developed before adding new features.

7.0: Risks and Mitigation strategies

A) Risk: Communication between teammates

A) Mitigation strategy: Discord, messaging apps, slotted meeting times every week, milestones, and constant communication throughout development and testing phases to avoid issues and completing milestones on time. Section 1.2 has details in terms of mitigation strategies for miscommunication and facing problems as a team.

B) Risk: 3rd Party API Maintenance

B) Mitigation Strategy: We will be utilizing well known website API's that have a track record of staying online and being reliable

C) Risk: Coding Skill/Competencies

C) Mitigation Strategy: Discuss the use of technologies that are already in the skillset of all team members or learn new skillsets required for each milestone and implement together.

D) Risk: Sickness

D) Mitigation Strategy: Rescheduling of deadlines, dropping less important features such as grammar checks.

E) Risk: Technical Difficulty

E) Mitigation Strategy: Rescheduling of deadlines, dropping less important features such as grammar checks.

8.0: Special components and/or facilities

This project will require multiple special components. These components are mainly third-party API's that should be available to access. Some websites have paid APIs, but our \$500 allowance should cover any API or database costs [15]. To name a few, LinkedIn [7], Grammarly [20] are examples of API's that we will be using for job applications and grammar checks.

9.0: References

- [1] "2022 HR Statistics: Job Search, Hiring, Recruiting & Interviews." *Zety*, 22 Apr. 2022, zety.com/blog/hr-statistics#recruiting-statistics.
- [2] "10 Applicant Tracking Statistics That Prove You're Hiring like an 'Atshole'." *Capterra*, <https://www.capterra.com/resources/10-applicant-tracking-statistics-that-prove-youre-hiring-like-an-atshole/>.
- [3] Deery, Matthew. "7 Advantages of Learning Python for Web Development." *Career Foundry*, careerfoundry.com/en/blog/web-development/reasons-to-learn-python/.
- [4] *GitHub*, github.com/.
- [5] *Discord*, discord.com/.
- [6] *Indeed*, ca.indeed.com/.
- [7] *LinkedIn*, www.linkedin.com/.
- [8] "Job Search Engine | Workopolis." www.workopolis.com/en.
- [9] "8 Things You Need To Know About Applicant Tracking Systems." *Jobscan*, 14 July 2021, www.jobscan.co/blog/8-things-you-need-to-know-about-applicant-tracking-systems/.
- [10] *Resume Worded - Free Instant Feedback on Your Resume and LinkedIn Profile*, resumeworded.com/.
- [11] *Python.org*, www.python.org/.
- [12] *Django*, www.djangoproject.com/.
- [13] "HTML Tutorial." *W3Schools Online Web Tutorials*, www.w3schools.com/html/.
- [14] "CSS Tutorial." *W3Schools Online Web Tutorials*, www.w3schools.com/css/.
- [15] *PostgreSQL*, 19 July 2022, www.postgresql.org/.
- [16] "JavaScript Tutorial." *W3Schools Online Web Tutorials*, www.w3schools.com/js/.
- [17] "SQL Tutorial." *W3Schools Online Web Tutorials*, www.w3schools.com/sql/.
- [18] "Python Dictionaries." *W3Schools Online Web Tutorials*, www.w3schools.com/python/python_dictionaries.asp.
- [19] "What is a REST API?" *Red Hat - We Make Open-Source Technologies for the Enterprise*, www.redhat.com/en/topics/api/what-is-a-rest-api.
- [20] "Write Your Best with Grammarly." *Grammarly: Free Online Writing Assistant*, www.grammarly.com/?q=brand&utm_source=google&utm_medium=cpc&utm_campaign=brand_core_row&utm_content=brandcorerow&utm_term=grammarly&matchtype=e&placement=&network=g&gclid=Cj0KCQjwhsmaBhCvARIsAibEbH6dwrkB2ozILz9qba3lAVEPAsl7XD-hw_ICziUp7Ya6CG27jBrn8OcaAgVBEALw_wcB&gclid=aw.ds.