Texas A&M University-Corpus Christi

Department computing Sciences

Using AI to Land Jobs

Amado Lazo
Nabil Ferhat Taleb
Egor Mikhaylov

Elijah Reynolds

March 2, 2026

Contents

1 References

1. OpenAI. (n.d.). OpenAI. Retrieved from https://openai.com/
2. React. (n.d.). React – A JavaScript library for building user interfaces. Retrieved from https://reactjs.org/
3. Williamson, M. (n.d.). Mammoth. GitHub. Retrieved from https://github.com/mwilliamson/mammoth.js/
4. Docxtemplater. (n.d.). Docxtemplater. Retrieved from https://docxtemplater.com/
5. Node.js. (n.d.). Node.js. Retrieved from https://nodejs.org/en/
6. OpenAI. (n.d.). openai-api. npm. Retrieved from https://www.npmjs.com/package/openai

2 SRS

2.1 Introduction to the SRS

2.1.1 Purpose

The purpose of this software requirements specification (SRS) document is to describe the requirements for a website that utilizes the OpenAI API to provide users with a customized resume based on a job description. The website will take in a Word document of the user's current resume and the job description for the position they are applying for. The output will be a new, updated resume in the form of a Word document that matches the job description. The aim of this project is to provide users with a scalable and efficient solution for creating customized resumes for each job application.

2.1.2 Document Conventions

This SRS document follows standard conventions for documenting software requirements. Priorities are assigned to each requirement statement, with higher-level requirements assumed to be inherited by detailed requirements.

2.1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, and other stakeholders involved in the development and implementation of the website. The document is organized into sections that cover the various aspects of the software requirements. Readers should begin with the overview sections and proceed to the sections that are most pertinent to their role in the project.

2.1.4 Product Scope

The software being specified is a website that provides users with a customized resume based on a job description. The website will utilize the OpenAI API to generate a new, updated resume in the form of a Word document that matches the job description. The goal of this software is to help users create customized resumes for each job application, providing a scalable and efficient solution for job seekers.

2.2 Overall Description

2.2.1 Product Perspective

The origin of our product is a result of a message with an individual who specializes in the hiring industry on LinkedIn. This individual has a business that consists of training PhD students on transferring from academia to the business world.

The present focus pertains to a novel software that is yet to be developed. Despite the existence of certain AI-based resume websites in the market, our offering stands apart. Our software will incorporate cutting-edge libraries and state-of-the-art validation sequences, unlike our competitors. As per our testing, some of our competitors utilize fabricated resume data do not present in the job seeker's portfolio. For instance, they may include a fictitious section where the candidate claims to have worked at Starbucks as a barista and efficiently increased departmental flow by 50%. Our product will not indulge in such practices. Instead, it will facilitate user validation to ensure that production increases by 50%, failing which it will provide additional inputs to generate an accurate response. Additionally, our software possesses the unique ability to customize resumes as per specific job descriptions, which our competitors currently lack.

2.2.2 Product Functions

The user's primary task is to enter a job description into the designated field, upload their resume, and click the submit button. In cases where validation is not required, the program will automatically download a job-specific resume template. However, if validation is needed, the user will be prompted to input any missing information.

The critical software function entails a series of data manipulations and server requests to the OpenAI API. Upon uploading their Word Documented resume, the data is first converted to text format and then forwarded to OpenAI for conversion to JSON. The resultant output is subsequently tailored to the job description entered by the user. Finally, the application collates all relevant information and populates a job-specific resume template.

- Accept job description input from the user.
- Validate and parse user-uploaded Word Document resume.
- Process resume text through OpenAI API for conversion to JSON.
- Tailor resume to the job description input.
- Prompt the user for any missing information required for resume validation.
- Populate a job-specific resume template with all relevant information.

2.2.3 User Classes and Characteristics

The software is designed to be user-friendly and straightforward. Users will only require a job description and their resume to access the product. Our target audience comprises individuals with resumes that are not job-specific, especially students who are seeking for employment. No technical expertise or special privileges are necessary to use the software. The introduction of a service charge in the future will not affect the nature of the anticipated user group.

2.2.4 Operating Environment

The software will operate in a serverless environment with a static client. The hardware platform will depend on the specific cloud provider chosen for deployment. The application will be hosted on an Ubuntu operating system, with the latest version of Ubuntu.

2.2.5 Design and Implementation Constraints

One potential limitation is the process for obtaining the API key from OpenAI. Users will need to visit the OpenAI website and create a new API key, which must be kept secure in a designated file to protect their development API key. This process may require additional security measures to ensure the API key is not compromised.

2.2.6 User Documentation

GitHub Readme.md instructs developers how to create and run the application for development purposes.

2.2.7 Assumptions and Dependencies

One potential factor that could negatively impact our product is the availability of the OpenAI service. If the OpenAI service experiences downtime, our product may not be able to function as intended.

2.3 External Interface Requirements

2.3.1 User Interfaces

To achieve a cohesive and visually appealing user interface, we have implemented a consistent theme that utilizes a soft pastel color palette. Our design also features a minimalist font that enhances readability and overall aesthetics. Additionally, we have incorporated interactive animations on all buttons to enhance user engagement and provide an enjoyable browsing experience.

Home page Interface: The Home Page boasts an aesthetically pleasing theme, featuring a prominent "AI Resume Builder" title positioned at the center of the page. An engaging and interactive "Start" button accompanies the title, which initiates the process of tailoring the user's resume to a specific job description. Additionally, a sleek and intuitive navigation bar is integrated into the Home Page, providing users with easy access to the other pages with a simple click.

Resume Interface: The Upload File interface features a visual selection for the current steps the user needs to take. The corresponding step is lit up, with the first step being uploading a resume. After that step is complete, the "Paste or Select" icon is lit up and the user then needs to paste the Job Description into the corresponding box. The Resume interface also includes the same pastel theme to ensure consistency.

2.3.2 Software Interfaces

Our project incorporates a total of eight distinct libraries, each playing a crucial role in the backend development process. These libraries include NodeJS v15.6.0, ReactJS v18.2.0, Docxtemplater v3.34.3, Pizzip v3.1.4, Mammoth v1.5.1, File-saver v2.0.5, OpenAi-api v1.3.1 and React-dotenv v0.2.3. OpenAI api serves as our primary library, driving the functionality of the project as a whole.

To ensure the security of our Private API keys, we leverage React-dotenv to generate a private file that is only accessible on our local devices. With this library, we are able to submit prompts and receive AI-generated text for the manipulation of resume data. The parsed text generated from Mammoth is used to attach our Word Document submission to the API, allowing for seamless integration with OpenAI.

Upon receiving a response from OpenAI, we utilize the Docxtemplater library to input the data and create a word document containing all the relevant resume information. The saving process is seamlessly handled by the file-saver library, ensuring ease-of-use and a seamless user experience.

2.4 System Features

2.4.1 OpenAI Integration

The system feature in question is serving as our primary operating software. While the risk of its failure is relatively low, estimated at around 2 due to the gradual slowing down of AI trends and the promising service of OpenAI, it is still critical that we maintain its functionality at all times. As for the cost of the API, we are currently able to take advantage of $18 worth of free credits, but it is important to note that additional charges may apply beyond this point. As we look to transition the project into a viable business, there will be a startup cost associated with the software, which we must carefully factor into our financial planning.

Upon the user uploading their resume, a function call is triggered to initiate the OpenAI response. This response will comprise the JSON transformation of the Word Document.

Functional Requirements:

- REQ-1: Must have API key integrated in order to use the service.
- REQ-2: Must have paired resume present available for upload to the service.

2.4.2 Mammoth Parser

The feature under consideration is tasked with converting data from a Word Document into regular text, which is compatible with the OpenAI API for further processing. Any malfunctioning of this parser can pose a significant risk. However, with the integration of the Mammoth library, which is not an online service but a simple library, the risk of malfunctioning is significantly minimized. Furthermore, the integration is available at no cost, thus warranting a risk rating of 1.

Upon the user uploading their resume, a function call is triggered to initiate the parsing of data from the Word Document file that has been uploaded.

2.5 Other Nonfunctional Requirements

2.5.1 Performance Requirements

The website must be able to process the user's resume and job description in a timely manner to provide an updated resume quickly. The system should respond to user input within 5 seconds, and the time it takes to generate the updated resume should not exceed 30 seconds. The rationale for this requirement is to provide a quick and efficient solution for users, as delays in generating the updated resume could result in missed job opportunities.

2.5.2 Safety Requirements

The website must ensure the safety and security of user data, particularly in relation to the user's resume and personal information. The website should not allow the user to upload any malicious file that may harm the system or the user's computer. The website should also prevent unauthorized access to user data by implementing appropriate security measures such as encryption, access controls, and auditing. In addition, the website should provide clear instructions and warnings to the user about the potential risks of uploading their resume and personal information to the website.

2.5.3 Security Requirements

The website must ensure the privacy and security of user data, particularly in relation to the user's resume and personal information. User identity authentication is not required for this website. The website must adhere to relevant data protection and privacy regulations, such as the GDPR.

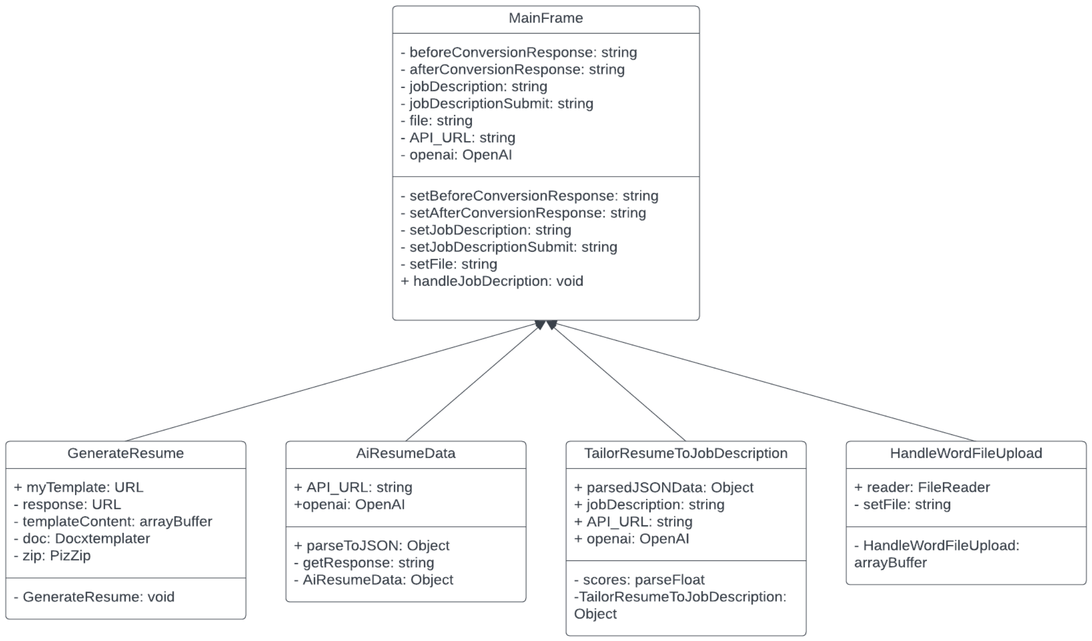
2.5.4 Software Quality Attributes

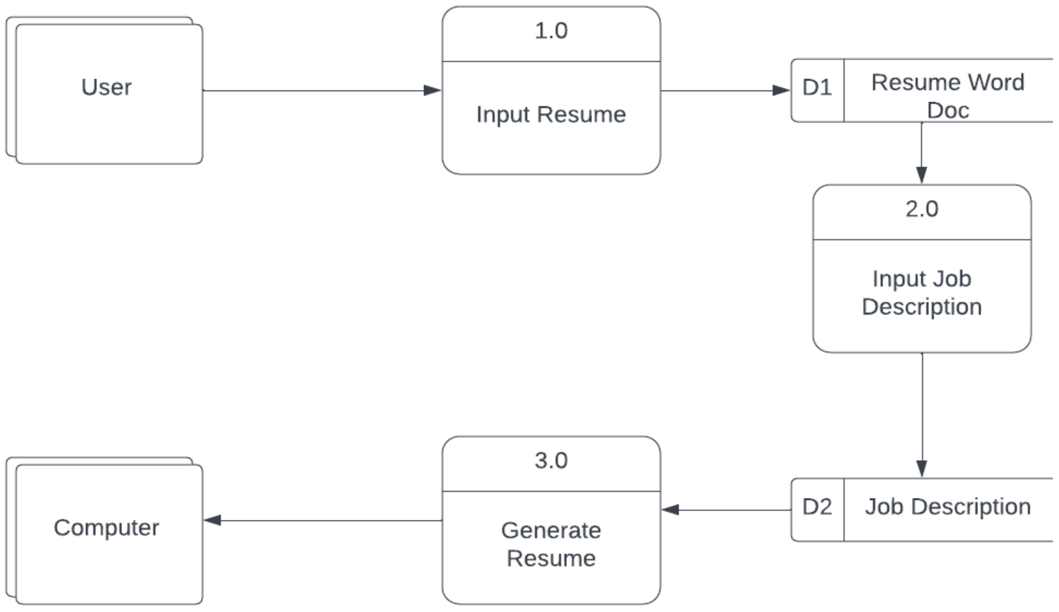
The website must be user-friendly and easy to use, with a simple and intuitive interface. It must be adaptable, flexible, and scalable to meet changing user needs and demands. The website must be reliable, robust, and maintainable to ensure long-term use and effectiveness. Usability is a critical attribute, and the website must be easy to navigate, with clear instructions and feedback for users.

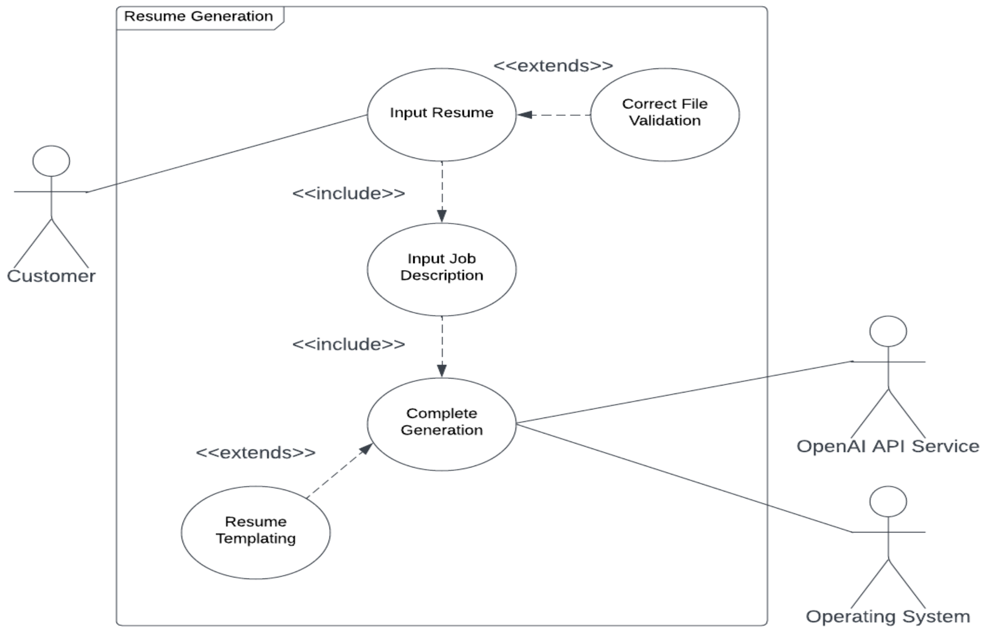
2.5.5 Business Rules

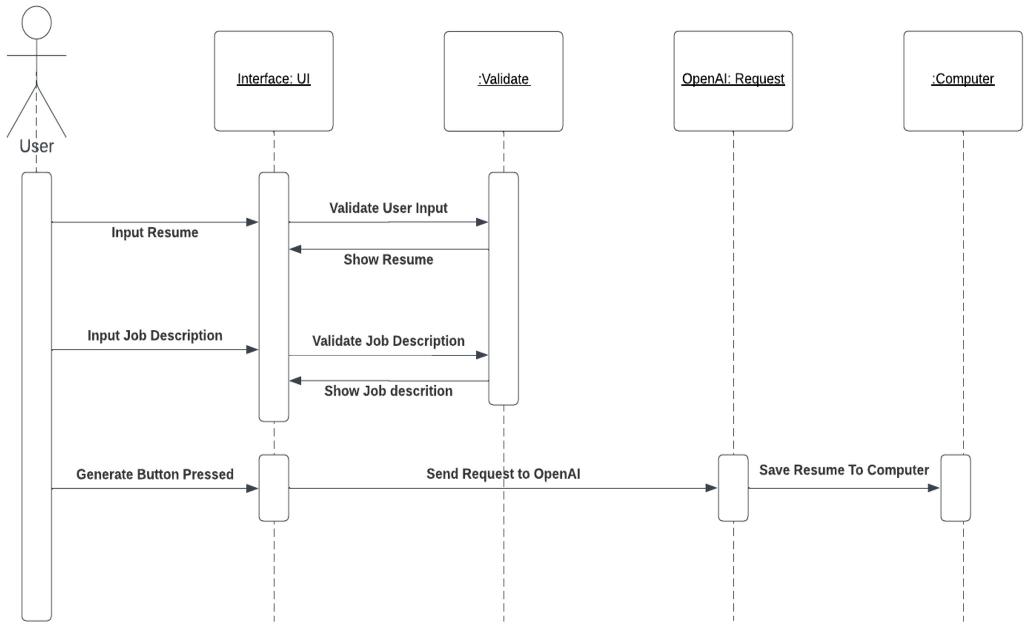
The website will enforce certain business rules, such as restricting access to user data to authorized personnel only. The website will not permit the sharing or distribution of user data to third parties without the user's explicit consent.

2.6 Diagrams


MainFrame
- beforeConversionResponse: string
- afterConversionResponse: string
- jobDescription: string
- jobDescriptionSubmit: string
- file: string
- API_URL: string
- openai: OpenAI
- setBeforeConversionResponse: string
- setAfterConversionResponse: string
- setJobDescription: string
- setJobDescriptionSubmit: string
- setFile: string
+ handleJobDecription: void
GenerateResume
+ myTemplate: URL
- response: URL
- templateContent: arrayBuffer
- doc: Docxtemplater
- zip: PizZip
- GenerateResume: void
AiResumeData
+ API_URL: string
+openai: OpenAI
+ parseToJSON: Object
- getResponse: string
- AiResumeData: Object
TailorResumeToJobDescription
+ parsedJSONData: Object
+ jobDescription: string
+ API_URL: string
+ openai: OpenAI
- scores: parseFloat
-TailorResumeToJobDescription: Object
HandleWordFileUpload
+ reader: FileReader
- setFile: string
- HandleWordFileUpload: arrayBuffer


User
1.0
Input Resume
D1
Resume Word Doc
2.0
Input Job Description
D2
Job Description
3.0
Generate Resume
Computer


Resume Generation
<<extends>>
Input Resume
Correct File Validation
<<include>>
Input Job Description
<<include>>
Complete Generation
<<extends>>
Resume Templating
Customer
OpenAI API Service
Operating System


User
Interface: UI
:Validate
OpenAI: Request
:Computer
Validate User Input
Input Resume
Show Resume
Input Job Description
Validate Job Description
Show Job descrition
Generate Button Pressed
Send Request to OpenAI
Save Resume To Computer

3 Design

The front-end design of the AI Resume Builder application is built using React and CSS. The application consists of multiple components to create a seamless user interface, which include:

3.1 Toolbar

A navigation bar that includes menu items for Features, Resume, Cover Letter, and Pricing, as well as options to log in and try the application for free. The Toolbar component is designed to create a consistent header across the application. The CSS code uses flexbox layouts, font styling, and hover effects to create an intuitive and visually appealing navigation menu. The .header class uses a flex layout to align its content horizontally and applies a box-shadow to create a subtle visual separation from the rest of the page. The .header-left and .header-right classes use flexbox properties and margin adjustments to position their content appropriately. The .menu-item and .sub-menu classes are styled with font sizes, colors, and hover effects to provide a seamless user experience. The .login-box and .create-resume-box classes use linear gradients, rounded corners, and font styling to create a consistent design with the rest of the application.

3.2 Front Page

The landing page for the application, which introduces the AI Resume Builder, its features, and provides a button to start using the application. The FrontPage component is designed to create an attractive and welcoming landing page for the application. The CSS code uses linear gradients, flexbox layouts, and font styling to create a visually appealing interface. The .page-background class uses a linear gradient background with fixed attachment, ensuring that the background remains stationary as the user scrolls down the page. The .front-page-container and .front-page classes use flex layouts to center their content and adjust their position based on the viewport size. The .front-page-text class uses a column-based flex layout to align its content vertically. The .front-page-image class uses absolute positioning to place the image at the top right corner of the component, creating a visually balanced design. The .start-button class is styled with padding, a background color, and a hover effect to enhance user interaction.

3.3 File Upload Page

A component that allows users to upload their existing resume, enter a job description, and generate an optimized resume based on their input. The FileUpload component's design is focused on creating a clean and organized layout that allows users to upload their resumes and job descriptions. The CSS code uses a flexbox layout to create a responsive design that adapts to different screen sizes. The .main-container class uses a column-based flex layout and a height of 100vh, ensuring that the component takes up the entire vertical height of the viewport. The .content class uses a flex layout with a width of 80% to create a responsive design that adjusts to the viewport's width. The .left-section and .right-section classes use flexbox properties such as justify-content and align-items to center their content. The .job-description and .resume-preview classes use absolute positioning and keyframe animations to create smooth transitions when displaying the content. The .button-container class uses absolute positioning to place the buttons at the bottom of the component and applies a linear gradient background to create an eye-catching design.

4 Implementation

4.1 ReactJS

ReactJS is a widely-used JavaScript library for building user interfaces. We chose ReactJS as the front-end framework for our web application, as it provides several benefits, including the ability to create reusable components and a modular architecture. Our ReactJS application consists of several components, including a file upload component, a job description component, and a resume display component.

The file upload component allows users to upload their resumes to the web application, while the job description component allows users to enter the job description. The resume display component displays the tailored resume generated by the OpenAI API.

4.2 Node.js

Node.js is a popular JavaScript runtime that is built on the Chrome V8 JavaScript engine. We used Node.js to create the back end of our web application. The Node.js server receives the user's uploaded resume and job description, processes the Word document using the Mammoth and Docxtemplater libraries, and sends the processed document to the OpenAI API for tailoring.

Using Node.js provided us with several benefits, including the ability to use JavaScript on the server-side, event-driven architecture, and non-blocking I/O operations.

4.3 Mammoth

Mammoth is a JavaScript library for converting Word documents to HTML. We used Mammoth to extract the text and formatting from the user's uploaded resume and job description. The extracted text was then used as input for the OpenAI API.

Using Mammoth allowed us to easily extract the relevant text and formatting from the Word documents, which could then be used as input for the OpenAI API.

4.4 Docxtemplater

Docxtemplater is a JavaScript library for processing Word documents. We used Docxtemplater to replace placeholder text in the processed resume document with the tailored responsibilities generated by the OpenAI API.

Docxtemplater provided us with an easy way to programmatically modify the Word document generated by Mammoth, allowing us to replace the relevant placeholder text with the tailored responsibilities generated by the OpenAI API.

4.5 OpenAI API

The OpenAI API is a machine learning platform that provides access to the GPT-3 language model. We used the OpenAI API to generate tailored resumes based on the job description provided by the user.

To use the OpenAI API, we made an API call from the Node.js server, passing the extracted text from the user's resume and job description as input. The API returned the tailored resume, which was then processed by the Docxtemplater library and returned to the user as a downloadable Word document.

Using the OpenAI API allowed us to generate tailored resumes for job seekers, which could increase their chances of landing a job that is a good match for their skills and experience.

5 User Manual

5.1 GitHub

5.1.1 Main Branch

The heart of the website code lies here, this branch is responsible for the up to date and merged back-end and front-end portions and is the code base used for the live website.

Changes to this branch in the form of push requests need to be verified first in order to not negatively affect the website.

5.1.2 Back-End Branch

This is where the majority of the back-end work happens, including parsing, API calls, etc. This branch can be pushed on demand by the back-end developer without negatively affecting the website.

5.1.3 Front-End Branch

This is where the main parts of the front-end design happen. This branch can also be pushed on demand by the front-end developer without affecting the main branch.

5.1.4 Front-End Dev Branch

This branch is mainly used for brainstorming or modifying the design, and it the most often pushed branch as it is a testing board for new website design ideas without worrying about affecting the functional front-end code in the Front-End branch.

5.2 Home Page

The home page of the website consists of a pleasant pastel gradient with the logo on the top left corner. The functional parts of the home page include the "Start" button at the bottom of the screen which will take you to the Resume Page, a toolbar which will navigate you to other future pages of the site, and the login/signup section in the top right where the user can login to the account.

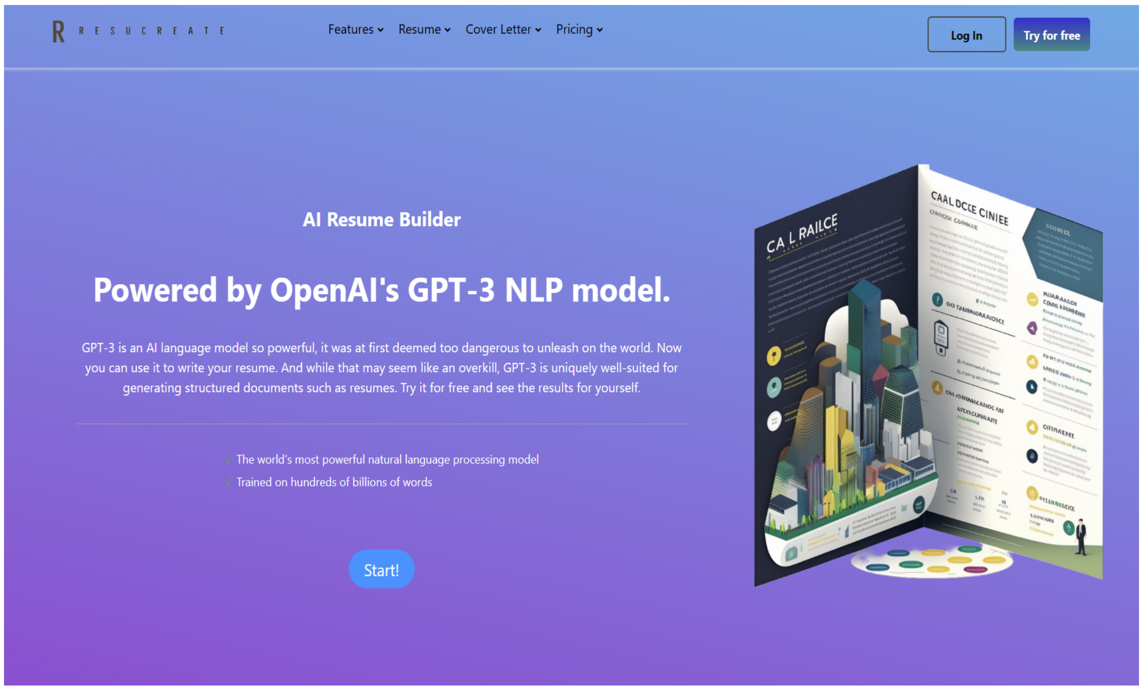
5.3 Resume Page

The resume page also features the pastel gradient as well as visual aid for the process of using our service. First, the "Upload Resume" icon will light up and direct the user to upload a compatible resume, followed by the "Paste or Select" icon lighting up and asking the user to fill out the Job Description page before receiving their modified resume.

6 Demonstration

6.1 Step 1: Open the AI Resume Builder Application

Open the AI Resume Builder application. You will see the Front Page (Fig.1) with an introduction to the application, a brief explanation of its capabilities, and a list of its features.



AI Resume Builder

Powered by OpenAI's GPT-3 NLP model.

GPT-3 is an AI language model so powerful, it was at first deemed too dangerous to unleash on the world. Now you can use it to write your resume. And while that may seem like an overkill, GPT-3 is uniquely well-suited for generating structured documents such as resumes. Try it for free and see the results for yourself.

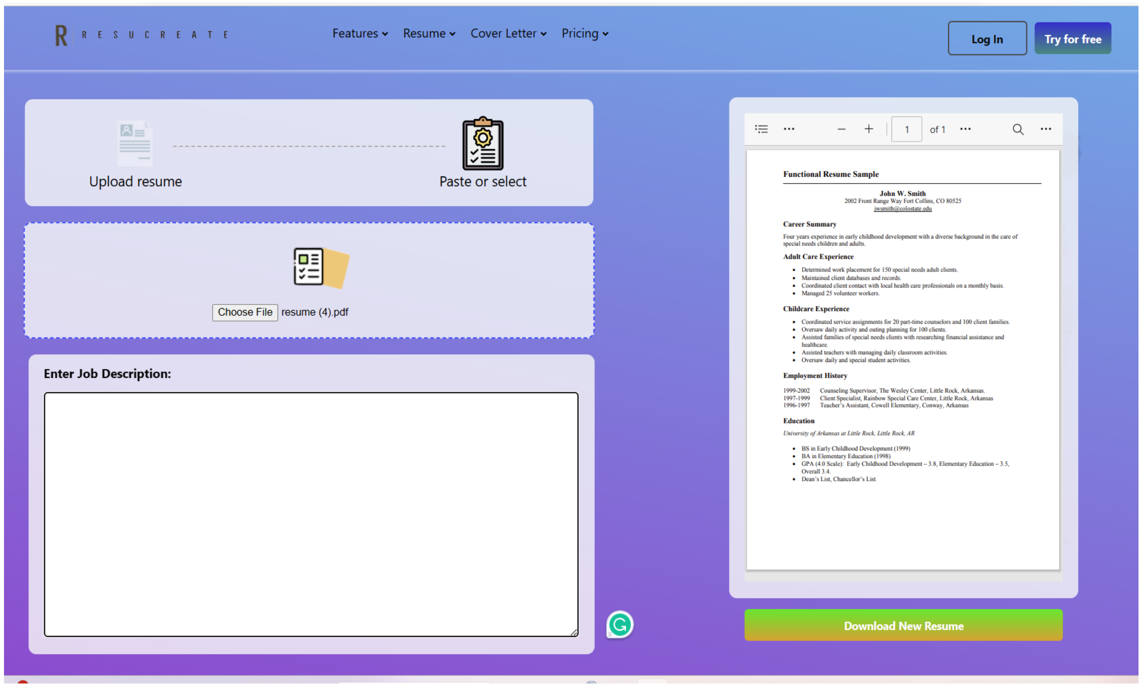
✓ The world's most powerful natural language processing model
✓ Trained on hundreds of billions of words

Fig.1

Front Page

6.2 Step 2: Access the File Upload Page

Click the "Start!" button located in the middle of the Front Page (Fig.1). This will take you to the File Upload page (Fig.2), where you can upload your resume and input the job description.



Functional Resume Sample

John W. Smith
2002 Front Range Way Fort Collins, CO 80525
jwsmith@colostate.edu

Career Summary

Four years experience in early childhood development with a diverse background in the care of special needs children and adults.

Adult Care Experience

- Determined work placement for 150 special needs adult clients.
- Maintained client databases and records.
- Coordinated client contact with local health care professionals on a monthly basis.
- Managed 25 volunteer workers.

Childcare Experience

- Coordinated service assignments for 20 part-time counselors and 100 client families.
- Oversaw daily activity and outing planning for 100 clients.
- Assisted families of special needs clients with researching financial assistance and healthcare.
- Assisted teachers with managing daily classroom activities.
- Oversaw daily and special student activities.

Employment History

1999-2002 Counseling Supervisor, The Wesley Center, Little Rock, Arkansas.
1997-1999 Client Specialist, Rainbow Special Care Center, Little Rock, Arkansas
1996-1997 Teacher's Assistant, Cowell Elementary, Conway, Arkansas

Education

University of Arkansas at Little Rock, Little Rock, AR

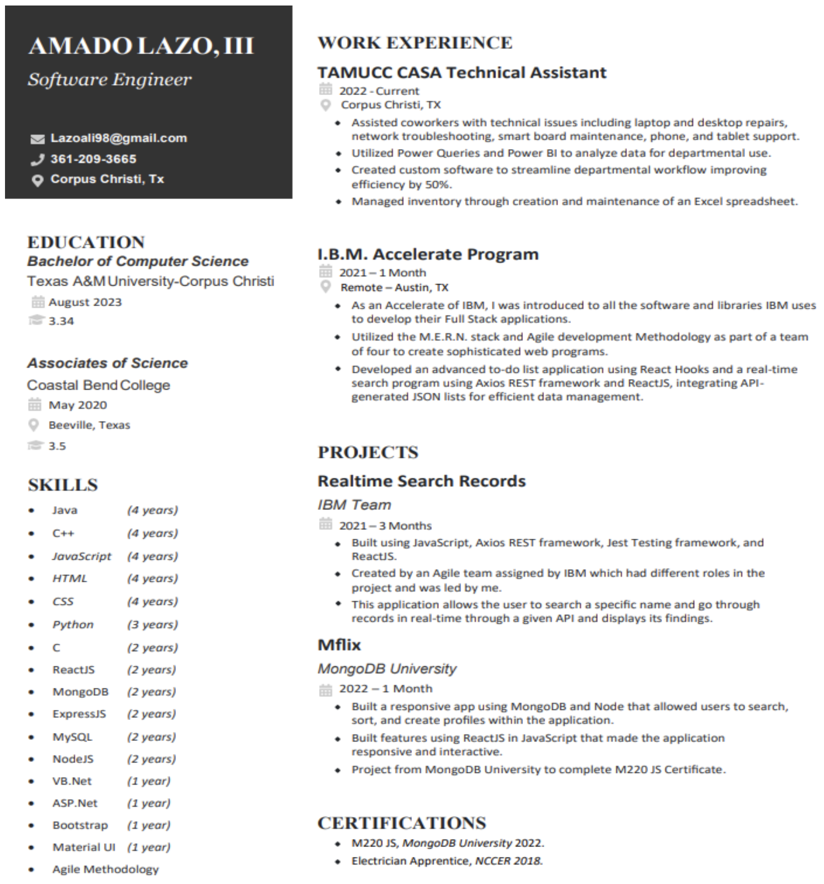
- BS in Early Childhood Development (1999)
- BA in Elementary Education (1998)
- GPA (4.0 Scale): Early Childhood Development – 3.8, Elementary Education – 3.5, Overall 3.4.
- Dean's List, Chancellor's List

Fig.2

File Upload Page

6.3 Step 3: Upload Your Resume

In the File Upload page (Fig.2), find the resume upload form with a Resume Upload Icon. Click on the form to open the file dialog box. Browse your computer and select a resume file in one of the supported formats: PDF, DOC, or DOCX.



AMADO LAZO, III

Software Engineer

Lazoali98@gmail.com
361-209-3665
Corpus Christi, Tx

EDUCATION

Bachelor of Computer Science
Texas A&M University-Corpus Christi
August 2023
3.34

Associates of Science
Coastal Bend College
May 2020
Beeville, Texas
3.5

SKILLS

• Java	*(4 years)*
• C++	*(4 years)*
• *JavaScript*	*(4 years)*
• *HTML*	*(4 years)*
• *CSS*	*(4 years)*
• *Python*	*(3 years)*
• C	*(2 years)*
• ReactJS	*(2 years)*
• MongoDB	*(2 years)*
• ExpressJS	*(2 years)*
• MySQL	*(2 years)*
• NodeJS	*(2 years)*
• VB.Net	*(1 year)*
• ASP.Net	*(1 year)*
• Bootstrap	*(1 year)*
• Material UI	*(1 year)*
• Agile Methodology	

WORK EXPERIENCE

TAMUCC CASA Technical Assistant
2022 - Current
Corpus Christi, TX
- Assisted coworkers with technical issues including laptop and desktop repairs, network troubleshooting, smart board maintenance, phone, and tablet support.
- Utilized Power Queries and Power BI to analyze data for departmental use.
- Created custom software to streamline departmental workflow improving efficiency by 50%.
- Managed inventory through creation and maintenance of an Excel spreadsheet.

I.B.M. Accelerate Program
2021 – 1 Month
Remote – Austin, TX
- As an Accelerate of IBM, I was introduced to all the software and libraries IBM uses to develop their Full Stack applications.
- Utilized the M.E.R.N. stack and Agile development Methodology as part of a team of four to create sophisticated web programs.
- Developed an advanced to-do list application using React Hooks and a real-time search program using Axios REST framework and ReactJS, integrating API-generated JSON lists for efficient data management.

PROJECTS

Realtime Search Records
IBM Team
2021 – 3 Months
- Built using JavaScript, Axios REST framework, Jest Testing framework, and ReactJS.
- Created by an Agile team assigned by IBM which had different roles in the project and was led by me.
- This application allows the user to search a specific name and go through records in real-time through a given API and displays its findings.

Mflix
MongoDB University
2022 – 1 Month
- Built a responsive app using MongoDB and Node that allowed users to search, sort, and create profiles within the application.
- Built features using ReactJS in JavaScript that made the application responsive and interactive.
- Project from MongoDB University to complete M220 JS Certificate.

CERTIFICATIONS
- M220 JS, *MongoDB University* 2022.
- Electrician Apprentice, *NCCER 2018.*

Fig.3 Example of Input Resume

6.4 Step 4: Enter the Job Description

After choosing your resume file, the Job Description section will appear beneath the file upload form. In the provided text area, enter the job description associated with the position you are applying for. Make sure to include relevant details about the job requirements, responsibilities, and desired qualifications.

6.5 Step 5: Observe the Progress

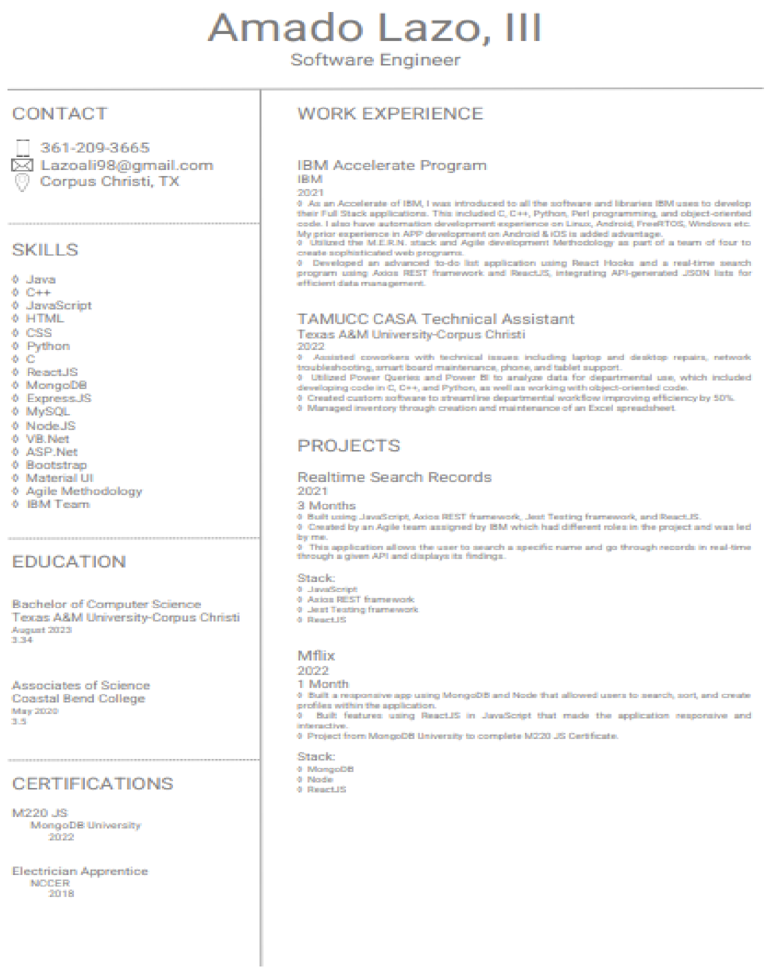
Observe the progress of the process through the visual elements on the File Upload page (Fig.2). As you upload your resume and provide a job description, the Steps Box displays the progress with the "Upload resume" step and the "Paste or select" step, representing the job description entry.

6.6 Step 6: Review Your Resume

Once you have uploaded your resume and entered the job description, a preview of your resume will appear on the right side of the File Upload page (Fig.2). This preview allows you to review your resume before downloading the optimized version.

6.7 Step 7: Download Your AI-Optimized Resume

The AI Resume Builder will analyze and optimize your resume based on the provided job description. When the optimization process is complete, the "Download New Resume" button below the resume preview on the File Upload page (Fig.2) will become active. Click on this button to download your AI-optimized resume (Fig.4).



Amado Lazo, III

Software Engineer

CONTACT

361-209-3665
Lazoali98@gmail.com
Corpus Christi, TX

SKILLS

◊ Java
◊ C++
◊ JavaScript
◊ HTML
◊ CSS
◊ Python
◊ C
◊ ReactJS
◊ MongoDB
◊ Express.JS
◊ MySQL
◊ NodeJS
◊ VB.Net
◊ ASP.Net
◊ Bootstrap
◊ Material UI
◊ Agile Methodology
◊ IBM Team

EDUCATION

Bachelor of Computer Science
Texas A&M University-Corpus Christi
August 2023
3.34

Associates of Science
Coastal Bend College
May 2020
3.5

CERTIFICATIONS

M220 JS
MongoDB University
2022

Electrician Apprentice
NCCER
2018

WORK EXPERIENCE

IBM Accelerate Program
IBM
2021
◊ As an Accelerate of IBM, I was introduced to all the software and libraries IBM uses to develop their Full Stack applications. This included C, C++, Python, Perl programming, and object-oriented code. I also have automation development experience on Linux, Android, FreeRTOS, Windows etc. My prior experience in APP development on Android & IOS is added advantage.
◊ Utilized the M.E.R.N. stack and Agile development Methodology as part of a team of four to create sophisticated web programs.
◊ Developed an advanced to-do list application using React Hooks and a real-time search program using Axios REST framework and ReactJS, integrating API-generated JSON lists for efficient data management.

TAMUCC CASA Technical Assistant
Texas A&M University-Corpus Christi
2022
◊ Assisted coworkers with technical issues including laptop and desktop repairs, network troubleshooting, smart board maintenance, phone, and tablet support.
◊ Utilized Power Queries and Power BI to analyze data for departmental use, which included developing code in C, C++, and Python, as well as working with object-oriented code.
◊ Created custom software to streamline departmental workflow improving efficiency by 50%.
◊ Managed inventory through creation and maintenance of an Excel spreadsheet.

PROJECTS

Realtime Search Records
2021
3 Months
◊ Built using JavaScript, Axios REST framework, Jest Testing framework, and ReactJS.
◊ Created by an Agile team assigned by IBM which had different roles in the project and was led by me.
◊ This application allows the user to search a specific name and go through records in real-time through a given API and displays its findings.

Stack:
◊ JavaScript
◊ Axios REST framework
◊ Jest Testing framework
◊ ReactJS

Mflix
2022
1 Month
◊ Built a responsive app using MongoDB and Node that allowed users to search, sort, and create profiles within the application.
◊ Built features using ReactJS in JavaScript that made the application responsive and interactive.
◊ Project from MongoDB University to complete M220 JS Certificate.

Stack:
◊ MongoDB
◊ Node
◊ ReactJS

Fig.4 Example of Output Resume

Follow these steps to create your AI-optimized resume using the AI Resume Builder application. The application provides a seamless, intuitive, and user-friendly experience, guiding you through each step of the process.

7 Results

7.1 Front-End

For the front-end, we worked hard to create an aesthetically pleasing website and learned how to properly use React, HTML, and CSS along the way. We learned how to overcome challenges during the implementation and learned how to work better as a team and communicate effectively.

7.2 Back-End

For the back-end, we learned how to properly use API calls and how to implement revolutionary technology like OpenAI into potential products. We learned how to troubleshoot JSON parsing, API calls, and other bugs that happened along the way.

7.3 Finished Product

Overall, we believe we did a great job on this project and this has greatly helped us with learning how to work together in groups as software engineers and this experience will be greatly beneficial to us as we transition into the workforce. We also learned a lot of skills together and learned about the challenges that come with group work and how to work together as a group to overcome them.

8 Challenges

8.1 Character Limit/Token Limit of OpenAI API with Davinci Engine

The OpenAI API has a character limit/token limit for each request made to the Davinci engine. This limit can cause issues when processing long resumes, as the text may need to be split into smaller chunks and sent separately to the API for processing. Additionally, if the resume contains complex or lengthy sentences, it may cause the token limit to be exceeded, resulting in incomplete or inaccurate responses from the API.

To work around this issue, we implemented a function to split long resumes into smaller chunks and process them separately using the OpenAI API. We also experimented with different settings for the max tokens parameter in the API request to optimize the performance of the model while staying within the token limit.

8.2 Fluctuation in OpenAI API Performance

During our testing, we observed that the performance of the OpenAI API could fluctuate, resulting in varying levels of accuracy in the tailored resumes generated. This issue could be due to changes in the API's underlying models or resources, as well as fluctuations in network performance.

To mitigate this issue, we implemented error handling in our application to detect when the API is not responding or returning inaccurate responses and provide feedback to the user to try again later.

8.3 Parsing Errors due to AI and Resume Template Formatting

Another challenge we faced was parsing errors due to inconsistencies in resume templates and variations in resume formats. The AI model used by the OpenAI API may not always be able to correctly identify and parse the relevant sections of the resume, leading to inaccurate or incomplete tailored resumes.

To address this issue, we experimented with different resume templates and formats to identify common patterns and structures that the AI model could more accurately parse. Additionally, we provided error handling and feedback to users when parsing errors were encountered, and suggested ways to modify their resume to improve the accuracy of the tailored resume generated.

8.4 OpenAI Davinci Engine Speed

Another challenge we faced was the speed of the OpenAI Davinci engine, which can take several seconds to generate responses to API requests. This issue can cause delays in the processing of resumes, particularly when processing large volumes of resumes.

To optimize the performance of our application, we implemented asynchronous processing of API requests using the Promise.all method in JavaScript. This allows multiple requests to be processed simultaneously, reducing the overall processing time required for generating tailored resumes.

8.5 Integration of multiple technologies

One of the main challenges we faced was integrating multiple technologies and tools into a cohesive system. We had to ensure that the front-end, back-end, and AI components of the system worked seamlessly together, and that the resulting tailored resume was correctly formatted and free of errors. This required careful coordination and communication among team members, as well as a deep understanding of each technology we were using.

8.6 Parsing errors due to AI-generated text

The AI-generated text produced by the OpenAI API was not always perfectly formatted, which caused parsing errors when we tried to process it using other tools and libraries. To overcome this challenge, we implemented several data cleaning and processing steps to ensure that the AI-generated text was correctly formatted and free of errors.

8.7 Inaccurate information and slow response times

The OpenAI API sometimes generated inaccurate information or took a long time to respond, which affected the overall accuracy and performance of our system. To address this issue, we experimented with different prompts and input formats to ensure that the AI model was able to accurately understand the job description and generate relevant and accurately tailor resumes in a reasonable amount of time.

8.8 Long resumes

We also encountered issues with long resumes, which sometimes caused the system to crash or produce incorrect results. This was because the AI model was not always able to accurately identify the most relevant experience sections and tailor them to the job description. To address this issue, we implemented several filtering and ranking mechanisms to identify the most relevant experience sections and ensure that they were given priority during the tailoring process.

8.9 Resume Template not formatted correctly

Finally, we encountered issues with the resume template not being correctly formatted or containing extra spaces, which caused errors and formatting issues when we tried to process it using other tools and libraries. To overcome this challenge, we implemented several pre-processing and data-cleaning steps to ensure that the resume template was correctly formatted and free of errors before processing it using other tools and libraries.

9 Testing

9.1 Job Description Testing

We tested our software by inputting various job descriptions from fields relating to software engineering. We wanted to ensure that our software could generate tailored resumes that accurately matched the job descriptions. We discovered that the scoring system of our software worked effectively, as it provided a score for each experience section of the resume based on how well it matched the job description.

For example, when we added a Plumber job experience and applied for a Software Engineering job, the score was nearly 0 as it had no relevance to the job description. However, when we added relevant programming experience such as working at IBM as a software engineer, the score increased to nearly 90. Similarly, when we tested a job that required Power Query experience and mentioned working as an IT at TAMUCC, the score was around 80 due to the relevant experience.

9.2 Resume Format Testing

We also tested different formats of resumes to ensure that our software could accurately extract and process the data. We found that some resumes were not ATS (Applicant Tracking System) friendly, which led to inaccurate data extraction. Therefore, we recommend using an ATS-friendly resume template to ensure accurate processing of the data.

9.3 Token Limit Testing

One of the challenges we faced during testing was the token limit of the OpenAI API's Davinci engine. The engine has a token limit that restricts the amount of text that can be processed at a time. We had to optimize our code to ensure that the input text did not exceed the token limit, which required multiple API calls and concatenating the results.

10 Known Issues

Despite our best efforts, we encountered several issues during the implementation and testing of our project. In this section, we outline some of the known issues and limitations of our system.

10.1 Long Resumes

Our system currently has a limitation on the length of resumes that can be processed. If the resume is too long, it may take longer to process and may even cause the application to crash. We suggest that users limit their resumes to one or two pages to avoid this issue.

10.2 OpenAI API Online and Offline Fluctuation

The OpenAI API can sometimes experience fluctuations in its online and offline status. This can result in delays or errors in generating tailored resumes. While this is beyond our control, we suggest that users check the OpenAI API status page before using our system to ensure that it is online.

10.3 Slow Performance of OpenAI Davinci Engine

The Davinci engine, the most powerful language model offered by OpenAI, can be slow to generate responses. This can result in longer processing times for tailored resumes. We suggest that users switch to a less powerful language model, such as the Curie engine, if faster processing times are required.

10.4 Inaccurate Information

While the OpenAI API is highly accurate, it is not perfect. There may be instances where the tailored resume generated by our system contains inaccurate information or formatting errors. We recommend that users carefully review their tailored resumes before submitting them to potential employers.

10.5 Parsing Errors due to AI

The process of extracting information from the user's uploaded resume using the Mammoth library and parsing it for input into the OpenAI API can sometimes result in parsing errors. These errors can cause the tailored resume to contain inaccurate information. We suggest that users review their uploaded resumes for accuracy before using our system.

10.6 Resume Template not Formatted Correctly

Our system assumes that the user's uploaded resume is formatted in a standard way. However, some resume templates may not be formatted correctly, causing parsing errors or formatting issues in the tailored resume. We suggest that users carefully review their uploaded resumes to ensure that they are formatted correctly before using our system.

11 What we learned

11.1 The importance of collaboration

This project required collaboration between team members with different backgrounds and skill sets. We learned the importance of effective communication and collaboration, especially when working remotely. We used tools such as Discord, Zoom, and GitHub to stay in touch and coordinate our work. In addition to those tools, we also got each others phone numbers to have direct connection with each other, therefore, we can text and call if we have questions.

11.2 The power of AI

We gained a deeper appreciation for the power of AI and its potential to transform the job search process. The OpenAI API provided a valuable tool for generating tailored resumes, and we were impressed by the quality of the output. However, we also learned that AI is not perfect, and there can be inaccuracies in the generated text. It is important to carefully review and edit the output to ensure accuracy.

11.3 The challenges of working with external APIs

Working with external APIs can be challenging, especially when the API is hosted online and subject to fluctuations in performance. We encountered some issues with the OpenAI API being slow or unresponsive at times, which affected the overall performance of our application. We also had to be mindful of API rate limits and adjust our code accordingly.

11.4 The importance of testing

We learned the importance of testing throughout the development process. We used Network tools that are built into Google Chrome to test our API endpoints and tested our application numerous times with different data sets. Testing helped us identify and fix bugs early in the development process, which saved time and improved the overall quality of our code.

11.5 The value of open source libraries

We used several open-source libraries in our project, including ReactJS, Node.js, Mammoth, and Docxtemplater. These libraries provided valuable functionality and saved us time and effort in development. We also contributed back to the open-source community by reporting issues and submitting pull requests to fix bugs.

11.6 The challenges of working with resume templates

Working with resume templates can be challenging due to the variability in formatting and the potential for parsing errors. We encountered issues with extra spaces in the template causing parsing errors and had to modify our code to handle these edge cases. It is important to thoroughly test the application with a variety of templates to ensure compatibility.

Overall, this project provided a valuable learning experience in software development and the use of AI in the job search process. We gained a deeper understanding of the challenges and opportunities of working with external APIs, open-source libraries, and resume templates, and learned the importance of effective communication, collaboration, and testing throughout the development process.

12 Future Work

12.1 Adding a Third Page for Resume Editing

We will be adding a third page to the application where users will be able to edit their newly generated resume. This page will feature a scoring system where each section of the resume, including various bullet points, will be scored based on their relevance and quality. The scoring system will help users identify areas of improvement and ensure that their resume is as effective as possible in showcasing their skills and experiences.

12.2 AI GPT-3 Suggestions for Bullet Points

Our system will integrate AI GPT-3 technology to provide different suggestions for various bullet points within the resume. This will allow users to have multiple options for enhancing their resume content, ultimately creating a more personalized and tailored document. Users can compare and choose from AI-generated suggestions, ensuring that their resume stands out and effectively showcases their expertise.

12.3 Customized Suggestions for Each Subsection

In addition to providing AI-generated suggestions for bullet points, the system will offer customized suggestions for each subsection of the resume. For example, under the "Experience" section, if a user has a software engineering job, the system could provide a suggested bullet point to add to their resume, such as highlighting a specific project or accomplishment related to the position. This feature will help users create a more comprehensive and relevant resume, increasing their chances of success in the job market.

13 Individual Project Assessment

In this section, we provide an individual assessment of each team member's contributions to the project. This evaluation helps to ensure a fair distribution of credit and responsibility among the group members.

13.1 Amado Lazo

My main contribution to this project was focused on researching and implementing the necessary libraries and tools for the back-end of the AI Resume Builder application. As part of this role, I spent a significant amount of time researching the OpenAI API documentation to understand how we could utilize it to generate tailored resumes for job seekers. This involved learning about the various endpoints and parameters available, as well as experimenting with different input formats to optimize the performance of the API. I also spent time researching and implementing the Mammoth and Docxtemplater libraries to process Word documents, which was a crucial component of our back-end system. This involved understanding how to extract the necessary text and formatting from the uploaded resumes and job descriptions, and then using this data as input for the OpenAI API. I was responsible for the overall structure and organization of the back-end, ensuring that the various components of the system were integrated seamlessly and functioned correctly.

13.2 Nabil Ferhat Taleb

My individual contribution to the AI Resume Builder application involved making modifications to the initial React template provided by Egor and refining the CSS styling to create a visually appealing and user-friendly interface.

For the **Toolbar** component, I updated the React code to enhance the navigation menu's functionality and applied CSS to create an intuitive and visually appealing navigation menu using flexbox layouts, font styling, and hover effects.

In the **Front Page** component, I made changes to the React code to provide a more engaging introduction and implemented CSS using linear gradients, flexbox layouts, and font styling to create a visually appealing interface. I also added a button that directs users to start using the application, incorporating event handlers for seamless navigation.

For the **File Upload Page** component, I made modifications to the React code to create a more user-friendly and interactive file upload experience. I applied CSS to create a clean and organized layout using a Flexbox layout. I implemented features that allow users to upload their existing resumes, enter job descriptions, and download their optimized resumes based on their inputs. I also added event listeners and handlers to manage user interactions and error handling to ensure a smooth user experience.

13.3 Egor Mikhaylov

For the project, I took on several key responsibilities. I was responsible for creating the first React website implementation template and integrating the necessary libraries to ensure it worked correctly. This involved designing an early version of the Toolbar component, Preview component, Front Page, and File Upload Page. My teammate Nabil later helped improve the look and functionality of these pages and components. I also set up the GitHub repository and helped my teammates resolve issues with pushing code. In addition, I helped write the project's documentation, making sure it was easy for everyone on the team to understand and replicate the website's functionality on their own systems. Finally, I worked on the resume template, refining it so that it could automatically generate a new document with the correct formatting when JSON data was entered. I also played a key role in ensuring the website's functionality and design were optimized for both mobile and desktop devices. By doing so, we were able to provide a seamless user experience that enabled clients to access the service from a variety of devices.