The City College of New York

Grove School of Engineering

CSC 22100 – Software Design

Fall 2023

FIRST PUSH

Professor Albi Arapi

Tenzin Choezom & Emir Dincer

Objective

The objective of this project is to ease the process of job application by developing an application that will organize any relevant information into a professional resume format, and enhance it using generative AI.

Introduction

Students often feel hesitant to start their application process for jobs/internships due to the lack of a resume. This application will solve this problem. With the use of technologies such as VSCode, ChatGPT, Git, and Python, we have created an application that takes user input through a GUI that the user can fill out and presents the user with a PDF of their resume based on their guided inputs. The front end of the application creates a GUI that displays a form to the user, asking to fill out any necessary information about them to create the resume. This can include topics such as personal information, professional experience, and skills. On the other hand, the backend works to take that information and add it to a generated PDF in a professional format with the required styling.

Our code is written in Python, and we are particularly using two libraries, FPDF and Tkinter. FPDF is a known python library used to generate PDF documents, also known as “Free PDF.” In our application, this library is used in the backend part of the project to generate a PDF for the resume and add chapters onto the PDF for every vital category. The other library we are using is Tkinter, which is a toolkit used to create graphical user interface (GUI). This is used for the front end so that the user is presented with a simple and clear GUI that can be filled out by the user without confusion.

Main

The main python file is what combines our frontend and backend together. We first import our frontend file, gui, and import our backend file, pdf. From the two files, we are only importing two functions, collect_data and generate_pdf. The two are used in our main function with the call "**collect_data(generate_pdf)**". This will run the **collect_data** function from "**gui**" with a callback function **generate_pdf** as the parameter. Once the **collect_data** has completed collecting the data, the **collect_data** will invoke the function **generate_pdf**, which will then generate a resume in the form of a PDF based on user inputs. There are established entry points, checking whether the script is being run directly as the main script. If it is, the "**main**" function is called, initiating the entire process.

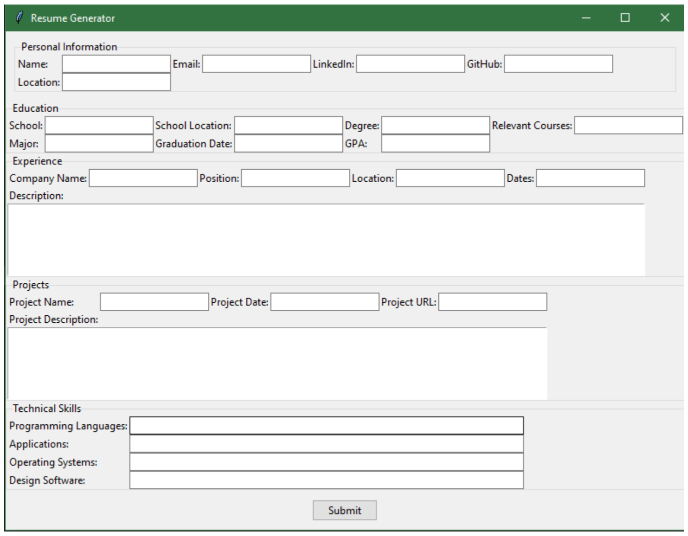
Front-end

As previously stated, the front end consists of a visually appealing and user-friendly graphical user interface to enhance user experience. To do this, we incorporated the Python library Tkinter, a standard GUI toolkit. Some of the initial steps are to import the necessary library, Tkinter, for GUI components. Specifically, we are using ttk, a module from Tkinter, for its themed widgets, allowing for a more organized, pleasing, and functional user interface. Following the import of the required libraries, define the function that will collect data from the user through a GUI that the backend will use.

The focal point of this application's front end is the '**collect_data**' function that takes a callback function as an argument. A callback function is a function that receives input from another function and is invoked after the initial function completes its task. In this case, the initial nested function is called '**submit_data**', which creates a dictionary called 'data' to systematically store user inputs into various subcategories as defined on the UI (user interface).

After this data collection process, the provided callback function is invoked with the collected data.

To further expand on when the '**submit_data**' is used, you must first understand the structure of the UI. The user interface is in the form of a Tkinter window titled "Resume Generator" is then instantiated and underneath the instantiation code, we define all the necessary sections that need to be presented on that window. In our case, this includes sections such as "Personal Information", "Education", etc. and each of those sections may require different formatting. For example, in our personal information section, we incorporated smaller entry widgets titled "Name:", "Email:", "LinkedIn:", and other sections for vital information. The image on the right shows the GUI presented when the application is run.



Finally, to trigger the execution of the 'submit_data' function, you can click the ttk button named submit included at the bottom of the window. This is possible due to the '**command**' parameter that specifies the function that should be called when the button is clicked, which here is '**submit_data**'. The user-entered information is then organized into the dictionary 'data' mentioned earlier. To ensure continuous presentation of the UI, the command '**root.mainloop()**' is used. Therefore, the Tkinter window will only close when the user decides to close the window. In addition, there are established entry points, checking whether the script is being run directly as the main script. If it is, the '**collect_data**' function is called, initiating the entire process of data collection and submission.

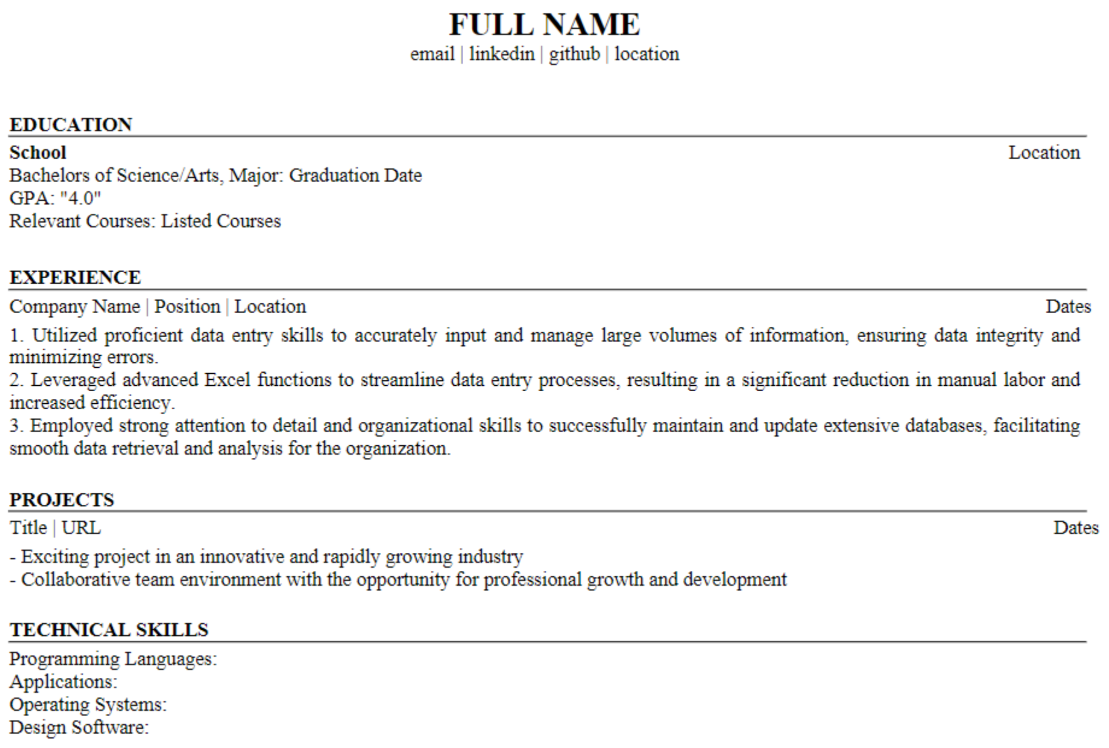
This concludes the front end of the application, and the backend work begins. As mentioned before, the ‘**submit_data**’ function has finished collecting the data using the GUI and the callback function is then invoked. When the callback function is called, the collected data is used to generate a professional resume for the user.

Back-end

The back-end of our application plays a crucial role in processing user input and generating the final resume PDF. It is composed of two main components: the utility functions for enhancing text descriptions using the OpenAI API, and the PDF generation module. Our utility functions, located in ‘**utils.py**’, leverage the OpenAI API to enhance descriptions provided by the user. These functions, ‘**enhance_internship_description**’ and ‘**enhance_project_description**’, utilize the GPT-3.5 model to transform plain descriptions into professional, recruiter-appealing bullet points. This integration not only adds a layer of sophistication to the resume but also assists users in presenting their experiences more effectively.

The ‘**pdf.py**’ module is responsible for creating the resume PDF. It uses the FPDF library to format and structure the resume. Key functionalities include formatting personal information, adding headers, and organizing sections like education, internships, projects, and skills. Each section is carefully designed to ensure a professional and readable layout.

The integration of these components is seamless. The GUI, developed in gui.py, collects data and passes it to the back-end. The utility functions enhance specific text inputs, and the PDF generation module compiles everything into a well-structured document. Error handling and user feedback mechanisms are implemented to ensure reliability and ease of use. The following image is an example of a resume PDF produced by the application.



FULL NAME
email | linkedin | github | location

EDUCATION

School Location
Bachelors of Science/Arts, Major: Graduation Date
GPA: "4.0"
Relevant Courses: Listed Courses

EXPERIENCE

Company Name | Position | Location Dates
1. Utilized proficient data entry skills to accurately input and manage large volumes of information, ensuring data integrity and minimizing errors.
2. Leveraged advanced Excel functions to streamline data entry processes, resulting in a significant reduction in manual labor and increased efficiency.
3. Employed strong attention to detail and organizational skills to successfully maintain and update extensive databases, facilitating smooth data retrieval and analysis for the organization.

PROJECTS

Title | URL Dates
- Exciting project in an innovative and rapidly growing industry
- Collaborative team environment with the opportunity for professional growth and development

TECHNICAL SKILLS

Programming Languages:
Applications:
Operating Systems:
Design Software:

Conclusion

This project successfully developed an application that simplifies the process of resume creation, particularly for students embarking on their job or internship applications. By utilizing technologies like Python, Tkinter, FPDF, and the OpenAI API, we created an easy-to-use GUI that collects user data and processes it into a professionally formatted resume.

Throughout this project, we deepened our understanding of Python programming, GUI development with Tkinter, PDF generation, and API integration. One of the main challenges was ensuring the enhanced descriptions were contextually relevant and professional. As this was our first time using OpenAI’s API, overcoming this involved fine-tuning the interaction with the and testing with various input scenarios was a very important first step not only for the project, but for the future of our careers.

In the future, the application can be expanded with features like resume customization options (different templates, color schemes), integration with LinkedIn for automatic data retrieval, and language translation capabilities for broader accessibility. This can be taken a step further, and create custom cover letters, using experience listed on the user’s resume, and tailored to a job posting requirements.

The application stands as a testament to the effective combination of user interface design and AI-powered content enhancement. It significantly eases the resume creation process for many students, potentially increasing the confidence and preparedness of students entering the job market.