Intelligent Software Agents Project Report

Intelligent Job Hunter

TEAM MEMBERS

A0198535N LI XINLIN

A0198491M XU JIACHEN

A0198519L Zhang Zhilin

A0198447L Nirmalendu Prakash

MASTER OF TECHNOLOGY

Contents

1.0 Executive Summary

Recently, every student in our institute of systems science has been shuttling between various interviews in order to find a good internship. Not only ISS students, almost everyone's life will definitely experience the job hunting, especially near the period of graduation. According to the survey, as of 2016, the number of recent graduates in Singapore has reached 8 million. But it is undeniable that the entire job search process is very cumbersome, the process of scanning through multiple job postings to find relevant ones is time-consuming enough, let alone to write cover letters manually respectively for different job positions.

Although there are many cover letter generators already, but on the one hand, they only focus on the format of the resume and cover letter without content which need be typed in by customers own, on the other hand, they ignore the process of selection jobs before cover letter generation things and the process of job application after generations. Our team, consists of 4 ISS students, want to build an agent which can help us do the whole working flow of finding a job.

The system is the Intelligent Job Hunter (IJH). After the user registers with our system and uploads their personal information and existing resumes, our system will firstly extract the information from the existing resumes and save them into database for the comparation purpose in the next stage. In every wee hours of the morning, our system will automatically collect all of the job positions through email, compare those job positions to the information extracted before and select out those job positions that are suitable for the user and generate cover letters for all suitable positions respectively. In the morning, after these processes are completed, our system will send the selected job positions to user through WeChat, user can not only just select the job they want to apply in WeChat, but also can go to our website to check the cover letter generated and modify them. Then after user confirmed, our system will send emails to the hiring manager automatically.

To build the Intelligent Job Hunter, we designed a hybrid solution combining the web application and chatbot, using the Vue.js, Flask, WeChat Subscription, TagUI and Natural Language Process.

In this project, our team applied the knowledge and techniques acquired from the lectures and workshops into the reality business application scenario, and we would recommend our Intelligent Job Hunter to every student in our institute, hoping it can help them improve the job finding efficiency and quality, help them to find a better job

2.0 Problem Description

According to the survey, as of 2016, the number of recent graduates in Singapore has reached 8 million, four-fifths of them utilized online resources. They receive job opportunity emails from various sources every day, which are considered essential but, most of the time, have unrelative information. To deal with these emails, they are required to click email links, read, filter, write a cover letter, add attachments, and send mail repeatedly.

NUS ISS students will receive emails attached job opportunity files uploaded to Luminus, which include job title, job description, and job requirement. Students need to download files from Outlook, checking the job description and requirement, writing cover letters if they interest, and preparing resumes. Consequently, they need to send emails to companies one by one.

With all of there requirements, it is a quite time-consuming process for a student who has many classes at the same time. Meanwhile, writing cover letter takes much effort, because it should be related to job requirement to impress the manager deeply. Our hope to create an automated tool for the unemployed to write charming cover letters as well as reply to job emails.

2.1 Project Objective

Based on the background, our team aims to build an intelligent, automated assistant system for NUS ISS students, which can simplify the workflow of email processing.

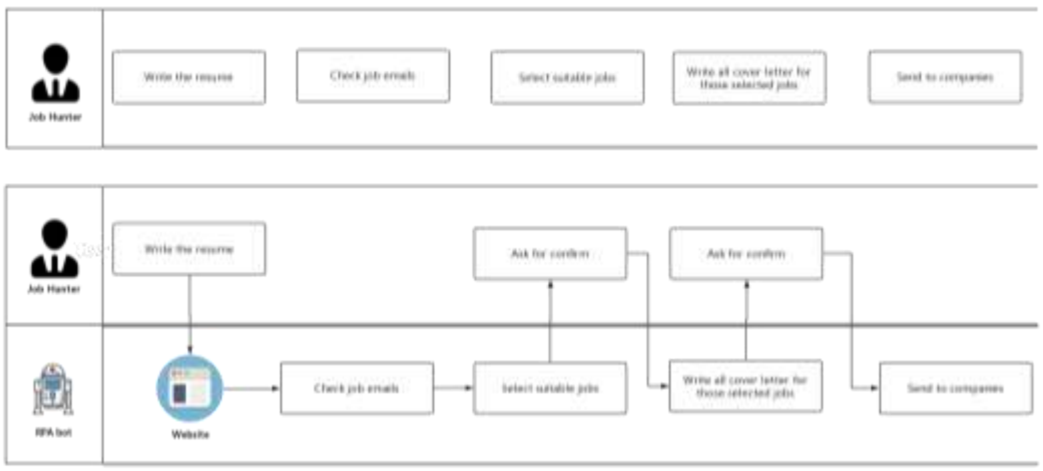


Figure 1. Intelligent Job Hunter – Current process and IJH solution

As the Figure 1 showed above, our system can maximize reduce the manual work when users receive job emails.

From the perspective of system work procedure,

- Firstly, students can open the website and scan QR code by WeChat App; moreover, they will be recommended to subscribe to the WeChat Official Account to login. Meanwhile, they should insert their user ID, password, gender, and age, as well as an email address as a login account.

- After login into the website, users can fill in their emails and passwords, which supposed to be utilized to receive job opportunity emails. At the same time, uploading resumes is essential for users. The website can complete part of the user information automatically, but users still were required to finish the region which is hard to be solved by the website. Once they finish these above parts, the user profile has already accomplished. It means that the system begins to check their emails and deal with aggregate processing.
- Every day, the system starts to log in every user's email account at the beginning of the day. It checks every job opportunity email, downloads files from Luminus, and then writes cover letters that fit the job description.
- At 9 am, generally, target users receive a template message from WeChat Official Account, which shows a brief introduction of job opportunities they receive from emails. Users glance over the webpage and select the job title they interest. After selecting, they get a message with a website link that includes cover letters that they applied. They should confirm the cover letters and, consequently, judge whether sending emails to companies just by reply "yes" or "no" in WeChat.

For the user part, they supposed to log in to upload resumes, fill out their personal information, and inset the email account and its password that should not be ignored. After accomplishing their profile, they can receive messages in their WeChat App on the smartphone every day, which exhibits the work introductions they receive from emails. They can obtain an utterly fit cover letter for the job they interest, and it absolutely can be fixed and optimized by themselves. After finalizing all these works, they can tell the WeChat Official Account whether they want to send emails to companies.

2.2 Project Team

The project members' name and work items are listed below:

Name	Work Items
LI XINLIN	Application Architect RPA Development Backend Development Documentation Video
XU JIACHEN	Application Architect Application Frontend Development Backend Development Documentation
ZHANG ZHILIN	Application Architect Chatbot Development Backend Development Documentation
NRMALENDU PRAKASH	Application Architect NLP Engine Development Documentation

Table 1. Project contribution

Our team met twice a week to discuss project status and updates. Tasks were assigned and areas of responsibilities were clearly outlined for each member.

3.0 Solution

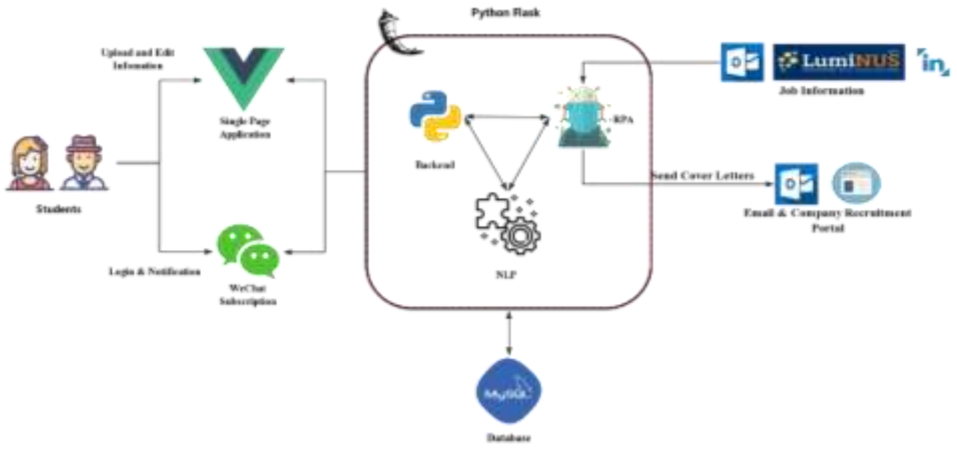


Figure 1. Intelligent Job Hunter -- System Architecture

3.1 System Architecture

The Intelligent Job Hunter System is a hybrid solution using the Vue.js, Flask, WeChat Subscription, TagUI and Natural Language Process.

The system is designed following the Frontend and Backend Separation Architecture, which makes the system highly componentized with high scalability and maintainability. Based on function, our team split our system into four parts, application, WeChat Subscription Chatbot and RPA part and IPA part.

The application part of our system uses the separation architecture of frontend and backend, the front end is a single-page application developed based on vue.js and element.js, and the back end is a web service that provides RESTful API calls based on Flask. For integration with WeChat Subscription Chatbot, the application part used the WeChat QR Code and GoEasy Websocket service.

The WeChat Subscription Chatbot of our system uses the WeChatpy, which is a third party WeChat subscription python library, to make our system can get openids of the subscribers and use those to send notifications, push the daily job position recommendation and communicate with users.

The RPA part of our system uses the tagui to collect the job information from the user email, luminus and job information portals, which is including auto login, auto search and auto download, and send generated cover letters and user's resume to the target companies.

The IPA part of our system uses the Nature Language Process technology to extract data from the user's resume and job position information and use that data and several templates to generate the suitable cover letters for different jobs.

For the communication and interaction of those four parts, we use database and Flask-Apscheduler. The database is based on MySQL and it is to store the data of user, resume, cover letter and job information, like the file location of the job position in the server, the user's email which need to be monitored by system. The Flask-Apscheduler is to automatically trigger the

RPA, IPA and WeChat tasks for job information collection, cover letter generation and sending notification, etc.

3.2 System implementation

3.2.1 Database Design

Considering the requirements, we design a database for our system.

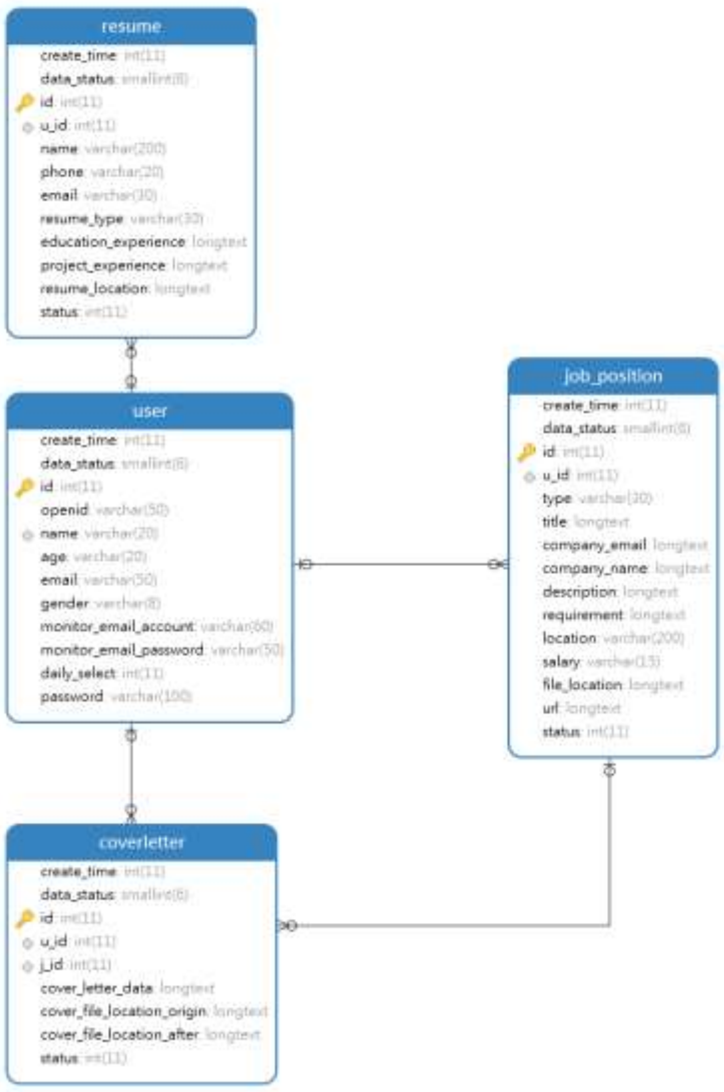


Figure 2. Database ER

User:

User table is designed to store user profile, including id, openid, name, age, email, gender, monitor_email_account, monitor_email_password, daily_select, password.

Job_Position:

Job Position table is designed to store job position profile, including id, u_id, type, title, company_email, company_name, description, requirement, location, salary, file_location, url, status

Resume:

Resume tables is designed to store the information of users resume, including id, u_id, name, phone, email, resume_type, education_experience, project_experience, resume_location, status

Coverletter:

Coverletter table is designed to store users' cover letter info for different job position, including id, u_id, j_id, cover_letter_data, cover_file_location_origin, cover_file_location_after and status

3.2.2 Application Part

To implement the application part of Intelligent Job Hunter System requires three parts of design and programming

- Design and implement the frontend based on the Vue.js and Node.js
- Design and implement the backend based on the Flask
- Design and implement the login and register process based on the WeChat Subscription

Because we follow the Frontend and Backend Separation Architecture, our system's frontend and backend have their own development and deployment environment. The frontend's development environment is Node.js and the deployment environment is using the Nginx. The backend's development and deployment environment are both Python 3.6.

For the security and user permissions, our system has carried out authorization and restriction of permissions based on pages and API requests respectively in the front and back, ensuring the Security of the system.

For the integration with the WeChat Subscription Platform, our system add the using WeChat QR to subscribe the system's WeChat Subscription and login method, let user can login our system just by using the WeChat to scan the Login QR code.

1. Design and implement the frontend based on the Vue.js and Node.js

For the design and implementation of the frontend interface, we use the most popular progressive JavaScript framework, Vue.js, and the Vue-element-admin Frontend Solution.

a) UI Design

We use the Vue-element-admin Frontend Solution for our UI design which provides many useful and components and tools. The UI design is mainly designed by using Element.js, which is a Vue 2.0 based component library.

b) Frontend Architecture and Security

Because of using Vue.js, our frontend Architecture is following the MVVM Architecture, it allows separating the user interface logic from the business logic. And each of our pages is composed of different components, which make the frontend system can be very scalable.

In terms of security, we use the page security authentication based on user's action scope and present different pages through different action scope of each user, which is to ensure that there will be no unauthorized operation. At the same time, we also use Token as the authentication way to interact with the background to ensure that the background Restful API will not be called without permission

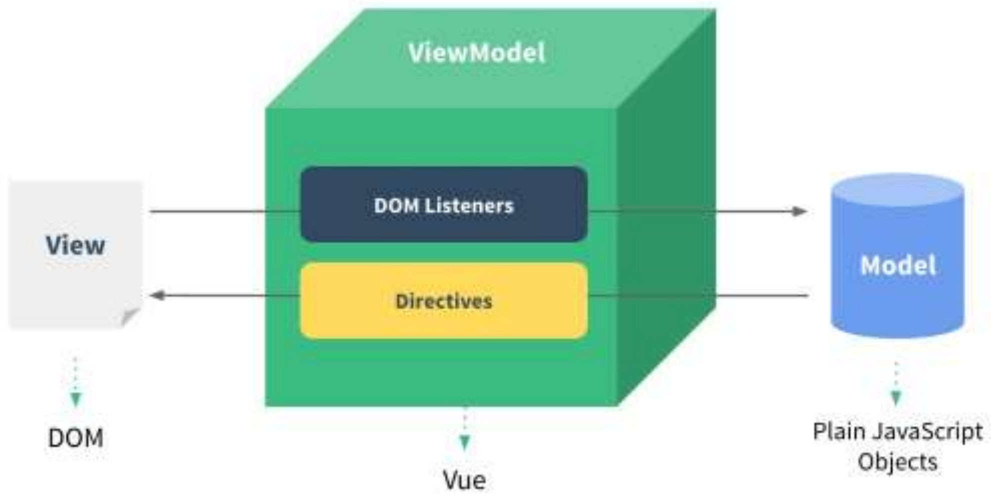


Figure 3. MVVM Framework

2. Design and implement the backend based on the Flask

Our system's backend uses the most popular web application python library, flask to build a Restful API web service, which is very suitable for backend web applications that are separated from the frontend

a) RESTful API and its Security

In the development of backend, we use the combination of the Blueprint and Redprint (modified Blueprint) to build the Restful API and security authentication based on user permissions to ensure the security of Restful API calls

b) Architecture of the backend

In our system, the backend is mainly responsible for providing the RESTful API service for the frontend to get data, communicating with the database to store and select students', teachers' and courses' relevant data, calling the Operation Layer, sending problem and getting the solution.

We use the SqlAlchenmy ORM framework to connect with the database for easily storing and selecting data, the ViewModel to modify the data selected from the database for frontend to display and the multithreading and http request to interact with the Operation Layer making sure this operation will not affect the user experience of the frontend.

3. Login Process Based on WebSocket and WeChat subscription

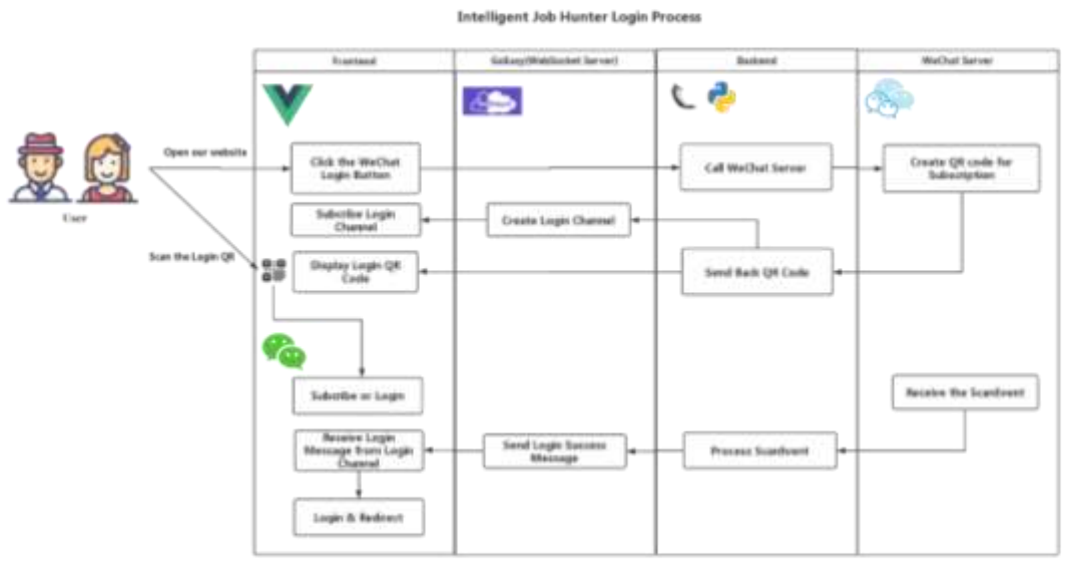


Figure 4. System Login Process

The login process of the application part in the system is using the most popular login method in the software industry, which is using the third-party OAuth login method. Because our team choose the WeChat Subscription as our chatbot platform, we use the WeChat as the third-party OAuth platform.

The login process in our system as following:

1. User opens our website on the browser
2. User click the WeChat Login button
3. User scan the Login QR code
    a. If user did not subscribe our system WeChat Subscription
        i. User click the subscribe button on the WeChat
        ii. Fill in the register form on the website
    b. If user has subscribed and finished the register part
        i. User will login our system

There are five components that interact in the login process:

1. Login page of Vue frontend
2. WeChat and WeChat Subscription
3. GoEasy websocket service
4. Flask backend
5. WeChat Server.

The challenge of using this login method is how to let the frontend listen the user login status in the whole login process, the basic http request and response can't be used for too long timeout and polling request can solve this problem but it will take up a lot of front-end resources and lead to a poor user experience. So our team decided to use websocket tech with GoEasy Platform, which can let our create a login channel for every user login process and the frontend can just subscribe and listen the message sent through the certain channel without polling request. We also use the QR code with parameter generated by WeChat to carry the channel name from the backend to the frontend in order to let the frontend subscribe the channel.

3.2.3 WeChat Subscription Chatbot

To implement the WeChat subscription chatbot, Intelligent Job Hunter requires two parts of design and programming

- Design and implement the Backend based on the Flask and WeChatpy
- Design and implement the Frontend based on the Html5
- Design and implement the chatbot on the NLTK

As the Frontend website generated by Backend, Frontend and Backend of WeChat chatbot have the same development environment, which is Python 3.7.

1. Design and implement the Backend based on the Flask and WeChatpy

Our system's Backend uses Flask framework to build a Restful API web service, which is quite convenient and light, extremely suitable for Backend web applications. Meanwhile, we use WeChatpy SDK efficiently to deal with QR code login and message sending.

In WeChat chatbot, the backend is mainly responsible for building webpages that show the brief introduction of job opportunities, providing the RESTful API service for the Frontend to get and post data, sending and replying message to users, and sending the application website links for apply.

WeChatpy is a third-party Python SDK from WeChat that implements APIs such as WeChat public account, enterprise WeChat, and WeChat payment. It can partly help to receive the XML message from WeChat platform and transport tuple data to XML. We use it to deal with the message we receive include time, id of users, and message they send. At the same time, we add menu on WeChat to show our official website and team introduction.

WTForms is utilized to verify user request data that prevent different user's individual information by WeChat openid. Meanwhile, for those expired website link we add Python timestamp as the keyword to ensure repeat reading.

2. Design and implement the Frontend based on the HTML5

Because the website generated by backend, frontend has no development. We letter down HTML5 code in Backend and post it by Flask framework.

a) UI Design

The Frontend utilizes Bootstrap 3.4 as a CSS framework, which provides many flattening and light components and tools. We chiefly combine two templates of the Html5 webpage to improve the visual experience of our users, making the webpage more visualized and elegant.

b) Frontend Architecture

The central function of our webpage is to show the introduction of jobs, include the job title and job description, as well as company name, salary and position if it contains. We fulfill essential functions like job selecting and submitting by JavaScript, which code can be inserted into the HTML5 file.

For improving user experience, we separate our Frontend with three webpages —— website explained as the job selector, finish website which notifies the users that they have already submitted, and expired website to prevent user receive overdue information.

3. Design and implement the chatbot on the NLTK

In the development of the chatbot, to understand the users' requirement, we use NLTK, which is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language.

The essential ability of WeChat chatbot is to send template messages to target users and depend on their answers to finish the next step. We use NLTK to extract the entities of the sentence, judging whether the emotion of user is positive, and then control the workflow.

3.2.4 Robotic Process Automation

To implement the RPA part of Intelligent Job Hunter requires two parts of design and programming

- Design and implement the job positions collection
- Design and implement the job application

1. Design and implement the job positions collection

Since the main target user of our system is ISS student, the sources of job position opening are mainly from nus talent collect and Luminus system. And both of these two sources will send an email notification when posting a position, this allows us to get all the job positions information by manipulating the user's mailbox.

On this basis, this part can be divided into the following steps:

a) Get links to all the job positions through user's mailbox
b) Download all the job positions through the link

By utilizing TagUI, a python package which can use live mode to send instructions from Python environment and retrieve data back to Python environment, our system can automatically log in to the user's mailbox and filter mail through advanced search, include email sender, keywords and date.

Filter by	Content	Notes
Sender	nustalentconnect@csm.symplicity.com	Two types of mail format
	luminus-do-not-reply@nus.edu.sg	One types of mail format
Keywords	"Job Opportunity"	Can accurately filter out the job positions email

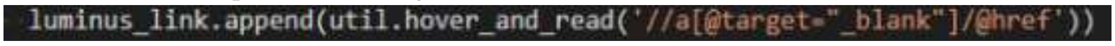
Date	Begin Date End Date	Set to the same date to get emails in that day

Table 2 . Filter set

As described above, there are three types of mail format. Regarding on different mail types, we mainly get all the information through different ways to use techniques including:

- Descendant selectors
- Attribute selectors
- Order selectors
- Class check
- hover function, read function and so on

Here shows a example which is to get the element's href attribute content:



```
luminus_link.append(util.hover_and_read('//a[@target="_blank"]/@href'))
```

Figure 5. Attribute selector example

2. Design and implement the job application

All of the jobs posting on these two platforms require an application via email, which brings up a problem, that is, it is very difficult to use TagUI to send emails with attachments through the web version of Outlook, since the file selection process can only be completed through image recognition mode not the selector, it's very unstable.

Based on what have been discussed above, we choose to call COM component using Python win32com under Windows and use win32com.client.Dispatch() to take out the class object in the com component, "outlook.application", then our system can generate and send emails through those API function, includes To() to specify recipients, Attachments() to attach cover letters and resumes, CC() to copy to someone and Subject to specify the subject of the email and so on. Because the authorization process for mailbox login is cumbersome, this part is simplified to send an email from us and copy it to the user to complete the user's job application task.

After these two processes are done, we get all the information we need about the posting job positions, then we will store relevant information into database by creating new job position objects, including flowing items:

- u_id
- file_location
- url
- title
- status

3.2.5 NLP

1. Information Extraction from Resume and Job opening

Letter generation consists of solving following problems at a high level:

1. Extract following information from applicant resume:
    - Name
    - Phone number,
    - Email
    - Work Experience-duration, description-This information is to be used in identifying applicant desired job type as well as for cover letter generation.

2. Extract following information from job opening:
    - Role description
    - Requirement

    These are required to identify job type as well as for matching against a list of applicant work experiences, to find out which experience matches best and use that for letter generation.

For problems 1 and 2, following approach has been used:

Name-File name of the resume

Phone number, Email-Regular expression match against all words in the resume

Work Ex-

A typical resume is divided into several blocks such as summary, professional experience, education, skills etc. To identify blocks from resume which describes work done by applicant in an educational\profit organization, a keyword based approach doesn't work as applicants can come up with numerous ways to mention work experience. Also, keyword based approach doesn't work for identifying job requirement and role sections, as a recruiter can come up with many different ways to mention them in a job post.

1) NLP approach
    - Work experiences from many resumes of 3 types of Applicants-Software Engineer, Data Scientist and UX designer was gathered.
    - When an applicant uploads a resume with our system, the resume is matched, using tf-idf and cosine similarity against all resumes of each type and an average score is calculated for each of the above 3 types, and the type with highest match is considered the applicant desired job type.
    - Roles and responsibility description, and requirement description from many resumes was gathered to create a database.

- When RPA finds a new job posting, all blocks from the posting are matched against roles and requirement database. Thus blocks containing role and requirement description are identified this way.
- When job type and applicant desired job type matches, job role and job requirement is matched against all work experience descriptions of the applicant, using tf-idf and cosine similarity. The work experience which has highest match is considered for letter generation.
- After identifying relevant work experience, all containing sentences within it are matched against job role and job requirement. Top 2 high scoring sentences are used in letter. These sentences fit a predefined format in the template.

2) Creating a database of Work Experience, Job Role and Job Requirement

Resumes are available in a fixed format on https://www.indeed.com.sg , which allows a keyword search for type of resume.

A python script using tagUI scrapes work experiences from many resumes and saves in a folder, thus creating workex database.

Job descriptions are not available in fixed format, as there are posted by different recruiters.

For Example-Roles and responsibility section can have many different headings:

What you will do

You will be involved in activities such as

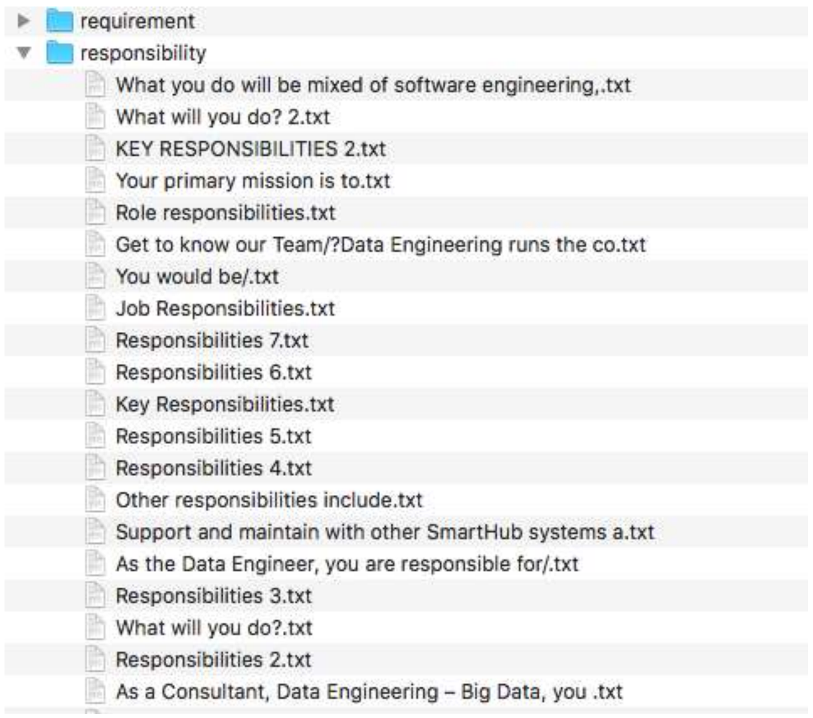
The Position

Your primary mission is to

and many more.

This HTML for this website has a structure for job posting-

The job description text is put under unordered list and nearest ‘p’ element contains heading. The script captures each block of list and heading and saves in a different file, file name being heading. This is done many times for each job type. Following are the files generated:



```
▶ requirement
▼ responsibility
    What you do will be mixed of software engineering,.txt
    What will you do? 2.txt
    KEY RESPONSIBILITIES 2.txt
    Your primary mission is to.txt
    Role responsibilities.txt
    Get to know our Team/?Data Engineering runs the co.txt
    You would be/.txt
    Job Responsibilities.txt
    Responsibilities 7.txt
    Responsibilities 6.txt
    Key Responsibilities.txt
    Responsibilities 5.txt
    Responsibilities 4.txt
    Other responsibilities include.txt
    Support and maintain with other SmartHub systems a.txt
    As the Data Engineer, you are responsible for/.txt
    Responsibilities 3.txt
    What will you do?.txt
    Responsibilities 2.txt
    As a Consultant, Data Engineering – Big Data, you .txt
```

Figure 6. Job description

Without opening the file, we can know which file to be used for role\responsibility and which one for requirement and which file we can ignore.

Thus, the database is created quickly without collecting and classifying data.

2. Letter Generation

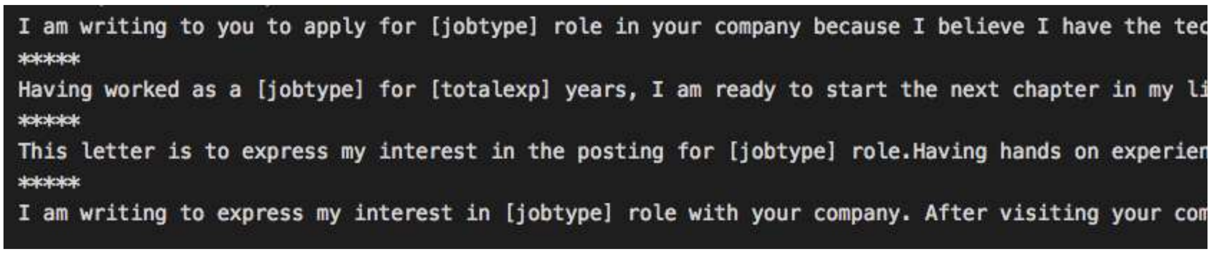
Templates for cover letter

A template txt file is created for each paragraph

A cover contains 4 paragraphs:

1st paragraph:

This paragraph mentions the purpose of the letter as well as a sentence about why the applicant is best fit or what is it about the company that applicant admires\follows.



```
I am writing to you to apply for [jobtype] role in your company because I believe I have the tec
*****
Having worked as a [jobtype] for [totalexp] years, I am ready to start the next chapter in my li
*****
This letter is to express my interest in the posting for [jobtype] role.Having hands on experien
*****
I am writing to express my interest in [jobtype] role with your company. After visiting your com
```

This file is read into a string and split by '*****' and randomly 1 item is selected, to be included in the letter.

This way, more template sentences could be added independently. The fill in positions are marked by '[]' and have specific keywords.

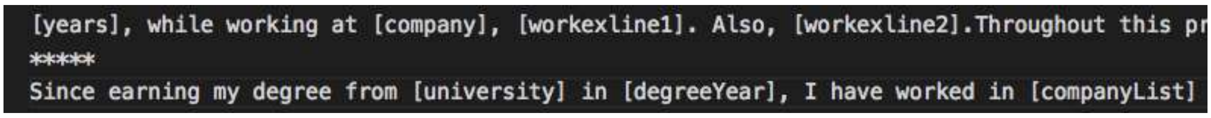
Jobtype-type of the job applied for

Totalexp-total user experience

and many more

2nd paragraph:

This paragraph contains relevant user experience.



```
[years], while working at [company], [workexline1]. Also, [workexline2].Throughout this pr
*****
Since earning my degree from [university] in [degreeYear], I have worked in [companyList]
```

Here actual lines with most match from the work experience part of applicant resume replaces workexline1 and workexline2

3rd paragraph:

This paragraph mentions how the job will help applicant advance in his\her career.

4th paragraph:

This is the concluding paragraph and contains applicant contact information. It may also contain some general strengths of the applicant.

3.2.6 Integration

In our system, the integration is to combine the application part, RPA, IPA and Chatbot. Because in the system architecture we use the Frontend and Backend Separation Architecture and all of the application backend, WeChat chatbot backend, RPA and IPA are implemented by Python, it is easy for our team to integrate all those parts into Flask baskend to be a part of application and use Flask-Apscheduler to set the crontab of those functions for automatically running , which makes them work as an end-to-end solution system.

We use the job position status to control the step of process:

No.	Status
0	Initial
1	Recommended
2	Selected on WeChat
3	Ready to Send
4	Closed

In the RPA and IPA automatic process, the RPA will insert job position information based on user’s email and Luminus and set the initial value to the status of job position. After that the job data extraction method in the IPA part will extract data from the those job position data. Then the IPA part will filter the job position, select the those has high correlationship with the user’s resume data and set status of those job position data to be 1. Finally the cover letter generator of IPA part will generate cover letter for those recommended job position for users.

For the chatbot, when it is triggered daily, it will select all users with recommended job positions, send them the notification of those job positions and start the communication and process of editing and sending cover letters.

Our system also allow user to update the resume and re-generate cover letter based on their edit, because we integrate the IPA part into the Restful API request and response which can further enhance our system’s user experience.

3.3 Process Flow

The general process flow of our system is described below, and the details are written in the user guide.

System Automatic Process

1. Every day 1 AM, System will trigger the Job Information Collection task, which will collect daily job information for every user from user email and LumiNUS, extract the data from job information and generate the cover letters for those job position which has high correlationship with the user's resume.
2. Every day 9 AM, System will trigger the Daily Post Task, which will send the daily job recommendation to the user WeChat through System WeChat Subscription.

User uploading data

1. Open our website
2. Click the WeChat Login Button
3. Use WeChat to Scan the Login QR Code on the website
4. Fill in the register form for the first login
5. Login the system and Check the Personal information
6. Upload the email user need to monitored by system
7. Upload your resume
8. Check and Edit the resume data extracted by system, Then Submit
9. User can upload the resume data anytime, which will take effect next day

User daily job application

1. Open the notification link pushed by system through system WeChat Subscription
2. Select the job positions user want to apply
3. Open the cover letter editing link receive after previous stop on the browser with a laptop
4. Check, Edit, Re-generate the job position information and corresponding cover letter generate by system
5. Submit the selected cover letters
6. Receive the confirmation message from system about whether send the resume and cover letters to companies or not
7. If user types "yes", system will send those cover letters to the corresponding companies
8. If user types "no", system will send those cover letters and job position information to user

3.4 System's Features

Time Saving

The system mainly to solve the time-consuming problem in job emails processing. It helps users to receive emails and write relative cover letters. After using our system, users can deal with hundreds of emails if they want, which ten times efficient than manual. It aims to save precious time of users by only clicking the button then finish.

Easy Using

The system is both convenient and effortless to use. Intelligent Job Hunter is built on a web-based application and WeChat Official Account. After the deployment on a server, the user can quickly login to our system by scanning QRcode and say "yes" if they like to send the email.

Productive Result

The cover letter we provide quite proximately matches its job description, fitting every requirement with users' resumes, which may become the reason why users get the interview opportunities.

3.5 Limitations and Future Enhancements

- Resume should have bold font for headings and normal font for paragraph text.
- Resume headings should be left aligned; resumes with center aligned headings don't work in the current version.
- Headings in the resume and job posting should have 1 blank line followed by description. This format fits most of the types.
- Currently only 3 types of jobs are supported-Software Engineer, Data Scientist, UX Designer.
  In future, we want to make the task of extending the solution to new job types easy without requiring any code change. Following changes to be done:
    - Read job types dynamically from dataset folder to show "resume type" options in the UI.
    - Provide a config file for any config mapping, for the cover letter. For example- "AI" field is mapped to "Data Scientist" profession.

  So, in future, following steps are required to add a new job\resume type:
  1. Add new type in job_type.json (present in dataset folder)
  2. Run resume_job_scrape.py
  3. Add new mapping in jobfield_mapping.xml
- Currently, job postings are downloaded from luminus only.
- Currently, job application is through mail only.
- User can upload resume only once.

- The user details should not include any special characters such as bullets. As the system uses latex for letter generation, special characters break the latex format. After uploading resume, user can see the details in UI. If the user finds any special character, the user can remove them and save.

4.0 Conclusion

In this project, our team built a hybrid Robotic Process Automation (RPA) and Intelligent Process Automation (IPA) system combining the Neuro-linguistic programming techniques, machine learning techniques, and reasoning techniques, which can provide intelligent and semi-automatic solution throughout complex and repetitive job pursuing workflow. At the same time, we accomplish our product by the minimum viable product (MVP) method, which helps us to control the process briefly and controllably.

Through this project, our team established a new understanding of how to make a brainstorm to use RPA and IPA techniques to simplify target users daily works; how to utilize knowledge of NLP to extract keywords for accurate matching and cover letter building; how to develop RPA workflow by tagUI — one of the widely used open-source RPA tool; how to deal with Python and Flask multithreading programming and WeChat API development.

During the whole process, we have encountered many challenges. For the IPA part, it is challenging to use TagUI to send emails with attachments through the web version of Outlook, as we can only use image recognition mode to select the file, which is quite unstable. Thus, we choose to call the COM component and process emails through the API function. We additionally faced difficulties when we design WeChat Login because there had no enough resources that could be referenced. Because Oauth2.0 authentication need official certify, at last, we combine a parameter with the QR code to solve this puzzle.

Moreover, to build a cover letter, we need to divide the resume into several blocks, but a keyword-based approach doesn't work as work experience can be described in different forms by job seekers as well as employers mention requirements in many approaches. Basing on the above discussion, we use IF-IDF and cosine similarity to deal with the obstacle. Last but not least, meeting all user's requirements and trying our best to simplify the workflow are quite complicated, but with the help of the MVP method, we definitely control the whole process.

We overcome the barriers and get a perfect and productive result that meets all the requirements commanded by target users, and all efforts are worthwhile.

5.0 Bibliography

Vue https://vuejs.org/index.html

Vue-element-admin https://panjiachen.github.io/vue-element-admin-site/

Flask https://flask.palletsprojects.com/en/1.1.x/

WeChat Subscription Document

https://developers.weixin.qq.com/doc/offiaccount/en/Getting_Started/Overview.html

WeChatpy

https://WeChatpy.readthedocs.io/zh_CN/master/

TagUI

https://github.com/kelaberetiv/TagUI

https://github.com/tebelorg/TagUI-Python

6.0 APPENDICES

APPENDIX A User Guide

Link: https://github.com/FoFxjc/ISA-IPA-2019-10-07-IS01FT-GRP1-Intelligent-Job-Hunter/blob/master/UserGuide/IJH_User_Guide.pdf

APPENDIX B Individual Reports:

1. Nirmalendu Prakash

Personal contribution

I worked on the NLP engine for the solution, which contains:

- data extraction from resume
- data extraction from job postings
- deriving applicant desired job type and job type from job posting
- Finding out most relevant applicant work experience and extracting most suitable sentences from it to be used in cover letter.
- Creating templates for cover letter
- Generating cover letter

Learning

I explored many different ways to approach this problem, namely:

Keyword based-

All job requirements mention skills required. For a technology job, this can be a technology or programming language such as python, java, r etc. For a teaching job, it can be subjects such as maths, science etc. If the job mentions "programming experience on machine learning tools such as python etc." Here if the applicant has mentioned r or matlab experience, this should still be a match as these technologies are related. An ontology map solves the problem of identifying similar work experiences but when I browsed through a list of softwares online, I realized that a software exists for any English word. So any random word in job requirement can be mistaken as a software.

NLP based approach 1-

After discarding ontology based approach, I considered matching every job posting with user resume and find the most matching sentences and use it to generate letter. This can create noise for the user, as there could be many irrelevant job mails. Also, the solution can evolve to be goal based instead of being reactive i.e. instead of waiting for job mails, the RPA agent can crawl the internet for job posting. This would result in a lot of noise. So a way has to be found for filtering irrelevant jobs. Therefore, each resume and job posting has to be classified.

NLP based approach 2-

A method has to be devised for classifying job types and resume types. For this, a dataset has to be created for each resume type and for each job type. Creating such a dataset manually can be time taking. The automated approach is described in detail in the NLP section. Also, the approach should be extensible without need of a code change.

The approach used only needs addition of new types in job_type.json and executing the script using a simple command:

Also, after creating a dataset and going through few of them, I realized that a job posting may contain information about the company, business unit etc. which is not useful for filtering and can skew the scores. So the next task is to identify which are the relevant blocks of information in the job post-namely the roles\responsibility section and requirement section.

A separate dataset was created for role and requirement section and each block from a new jo post is matched against each job role and requirement in the dataset and using an average score and a threshold, the irrelevant blocks of information from job post is discarded.

A similar approach is adopted for data extraction from resume.

It has been a great learning exploring various ways to achieve the objective. I came to know about the pros and cons of using 1 approach versus the other. I would learn about generative NLP and perhaps get rid of the need of a template for letter generation.

Also, I enhanced my knowledge about web scraping tools and how to effectively use them.

How will you apply this knowledge and skills in other situations?

I will read about generative NLP and knowing the pros and cons of a template based approach will help me. I will also explore statistical models for speech to text conversion and see how I can improve the existing solution.

2. XU JIACHEN

1. Personal contribution to group project.

For the Intelligent Software Agent group project, our team built an Intelligent Job Hunter system to provide the cover letter generation service for users, which can automatically generate suitable cover letters for different job positions based on user's resume

In this project, my contribution is mainly in two aspects.

1. System architecture and integration.
2. Design and Implementation the frontend and backend of the application part and database of system.

2. What you have learnt.

In this group project, I am still a full stack engineer and architect and responsible for the development of full stack applications and integrated WeChat Subscription Login. I learned a lot in this aspect. Although there are many examples and use cases in the industry by scanning the QR code of WeChat Subscription and login system or website, there is no open source solution based on python. So, I could only analyze those methods and design the solution by myself. In my architecture and implementation experience, this is the first time that I spend a lot of time on the user login and registration. Because our system needs a WeChat Subscription as Chat-Bot platform to send the job position information and communicate with users, so our system must be able to get the user's Openid in our system's WeChat Subscription and save into our system, Furthermore, we need to simplify users' actions. If let users login in our system, and then subscribe the WeChat Subscription to receive messages and notifications, it will make users feel that chatbot is just a by-product of the system and has not been truly integrated into our system. Therefore, my method is let users complete both the register of our system and subscribe the WeChat Subscription for receiving notification through the login process. Through a lot of tests and studies, I successfully completed the login solution based on WeChat Subscription, which integrated WebSocket and WeChat platform services, and provided users with smooth and simple login and registration process as far as possible.

On the other hand, while in the group project, I am not responsible for the RPA and IPA part, I still learnt a lot from the advantages and disadvantages of these technologies.In the RPA part, although TagUI is a very good RPA tool, but there are still some compatibility problems and limitations, such as, not like selenium, TagUI can't change and control the browser's to clear the cache and website login data. In our project, there is a case that for multiple users' job position data collection, the previous login user did not exit after the `t.close()`, leads to repeated fetching from same user's email. In the part of IPA, our team adopted NLP + Template method to generate cover letter, because there is no functional Cloud AI or Local AI with such function. Meanwhile, we have investigated many methods, such as Named-Entity recognition, but the effect was not very good. So, we can only use NLP + Rule-based method to generate based on separated templates. Through this process, I feel that I still need learn a lot of knowledge in the NLP field. For example, whether we can integrate multiple models and combine templates to complete the process of NLG, which certainly requires a lot of time for design and training, but if we can achieve this method, our group project will be more intelligent and provide better cover letter generation services.

3. How you can apply the knowledge and skills in other situations.

I think RPA and IPA both are very interesting and powerful tools or method to let computer handle those time-consuming tasks and repeated tasks, but they still have some limitations. For example, They are very strong for the web-based data collection but not very good for the image-based or the code-based data collection. Furthermore, I think RPA and IPA is the transition part of the system which changes from the server separated systems to an End-to-End system. As long as there is not a overall system, RPA and IPA technology is still very useful. I think I will use RPA and IPA technology on the dataset creation and decision making system for the project leaders or department leaders. Because there are many web-based applications in a company or a department, they spend much time on collecting information from the different web applications. RPA and IPA can help them to generate the overall report and collect the data

3. LI XINLIN

1. Your personal contribution to this project.

After the discussion and brainstorming of our four together, we set the topic of this project. With the help of MVP course, it’s clear to us about how the project should be divided, including website development, RPA part, IPA part and chatbot part.

Since the IPA part is the most difficult and important part of our project, we worked together few days at the beginning of our project and designed the implement method together, include the method to extract key information from the resume and the job positions and the method to generate a personalized cover letter. At the same time, we also designed the database and the whole system architecture together.

Then we separate to work at different part, I am mainly responsible for the development of the RPA part, mainly includes the job position collection task and the job application task.

After all of us are almost finished, we work together again to do some integrate and test things until the whole process works well.

And at last, we are responsible for our own parts’ report writing and the remainder part report writing divided equally to everyone. Besides, I also responsible for making the video presentation.

2. What you have learnt from this project.

For this CA, I learned the basic idea of several fundamental techniques in the natural language processing. At the same time, I also have a deeper understand about the process automation, both the robotic process automation and the intelligent process automation, and I improved relevant skills about the process automation including natural language processing, TagUI, win32com and so on. The most important part I learned from this CA is that I get more familiar about the realistic utilization situation of the process automation, not just limited on books and courses.

3. How you can apply the knowledge and skills in other situations

Through this CA, I have accumulated relevant experience in process automation which is widely used in many working or living situations.
For me, I am really tired with dealing the tons of emails received everyday. Since those

emails are mainly in five types, include job posting, news, school events, administrative emails and others. Now I’m quite familiar to let program manage the mailbox for me, I am actually planning to develop some tools to help me get rid of this trifles, maybe combining the machine learning to do the classification and the RPA to do the sort process. For others, there are too many situations can make a use of the process automation process to improve the efficiency, actually, there already have many application especially in the area of administrative works, I think the official automation applications are all based on this same idea. And I also think there still are many tedious process can be replace or improve by these process automation techniques, like fill forms things, the daily check-in events (many applications have these kind of activity with some bonus to improve customer retention rate).

4. ZHANG ZHILIN

1. Your personal contribution to the project in this Graduate Certificate.

I mainly response to WeChat Chatbot development, include backend and submit website, as well as application architect and report.

2. What you have learnt from the project.

I learned the knowledge of NLP technologies and the basic development of chatbot. I also learn how to develop WeChat Official Account and experience full-stack development.

Moreover, it is the first time I use MySQL as a database. It's quite a fantastic experience that we don't need to write SQL sentences anymore; Python is such great language that it's so convenient and fascinating. Meanwhile, It's the second time to touch the product developing, and this time we control the whole development workflow well than before. MVP method helps me a lot to control the project.

3. How you can apply this in future work-related projects.

First of all, I believe this full-stack experience will definitely benefit my further career. Meanwhile, I interest in it and want to learn more about it, including security, web socket, web animation, and other technologies in real productive work. I also learned how to develop WeChat Official Account, which may be becomes more and more popular in China.

On the other hand, I found more I learn more I loss. I am badly required to learn much to improve my abilities and skills.

Last but not least, teamwork is an excellent platform to compare myself with others, to learn more expert skills, and more importantly, how to explain yourself and understand others.