SMART RESUME
BI TOOL TO SELECT THE CVs APPLIED FOR A JOB VACANCY

Shimira Prabhath Senarath

(IT15099778)

Bachelor of Science Special (Honors) Degree in Information Technology

Department of Software Engineer

Sri Lanka Institute Of Information Technology

Sri Lanka

August 2018

SMART RESUME
BI TOOL TO SELECT THE CVs APPLIED FOR A JOB VACANCY

Shimira Prabhath Senarath

IT15099778

Final Report

The dissertation was submitted in partial fulfillment of the requirements for the B.Sc. Special (Honors) Degree in Interactive Media.

Department of software engineering

Sri Lanka Institute Of Information Technology

Sri Lanka

August 2018

DECLARATION

"I declare that this is my own work and this dissertation1 does not incorporate without acknowledgment any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text. Also, I hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or another medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The supervisor/s should certify the dissertation with the following declaration.

The above candidate has carried out research for the B.Sc Dissertation under my supervision.

Signature of the supervisor: Date

ABSTRACT

Now-a-day in Sri Lanka, most industries follow up one traditional process of hiring new employees. The normal process in the industry, advertising the vacancy and calling Curriculum Vitals (CV). Then shortlisting them by referring the CVs and interviewing the shortlisted candidates.

Having the right set of CVs, CV is the representation of the qualifications of an applicant. Also, when it comes to an emergency project, the employer should be able to hire the best employee set within a minimum time period.

In this case, the employer can get the help of a third party CV storage which already has a collection of related CVs, and has the ability to generate the list of most qualified applicants among them, would be helpful. The submitted CVs should be read properly and check several attributes such as skills, experiences and some personal information in order to select the best.

It is much time consuming for a human to read and draw a mind image of the applicant. There is a shortage of tools that support in selecting the best-qualified set of employees to an employer.

Smart Resume is a business intelligence tool for the IT sector, which analyses and classify operational data with classification algorithms to present complex and competitive information to decision makers, in order to dynamically fulfill the business needs. It is built to satisfy the task of generating the list of most suitable candidates.

Smart Resume is a business intelligence tool for the IT sector, the component has “BOT Creation”. This component has four major subcomponents called Download the CVs in the website automatically, to read the downloaded CV, Classify the data into relevant columns and Save the classified data in CSV format. Smart resume consists with a desktop and web application.

ACKNOWLEDGEMENT

I wish to express my sincere thanks to our supervisor, Mr. Lakmal Rupasinghe, whose constant guidance, direction, as well as encouragement, helped us to complete this research.

I am grateful to Mr. Jayantha Amararachchi for providing final years students with the knowledge that is necessary for writing and presenting effective project progress reports.

I take this opportunity to extend my sincerest thanks and appreciation to my parents, for their support and encouragement in completing this research.

Finally, I place on record, my sincere gratitude and appreciation for the contribution and support from all the individuals that have not been mentioned by name, but without whom the completion of this research would not have been possible.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Acronyms	Definition
BOT	A bot (short for "robot") is an automated program that runs over the Internet
HTTPS	Hypertext Transfer Protocol Secure
CV	Curriculum vitae
CSV	CSV file format
JS	JavaScript
NLP	Natural language processing
AI	Artificial Intelligence

Table 1- Definitions, Acronyms, and Abbreviations

1. INTRODUCTION

1.1. Problem to be addressed

The world is a chain of businesses. As the businesses get bigger day by day, the complexity and the competition are highly increasing. New trends are being incorporated with business ecosystems. So day by day small to large all companies have to update themselves in terms of resources, manpower, and infrastructures in order to maintain a competent and business system.

When company hiring new employees, the company will have to spend much time, effort and cost on finding suitable candidates among thousands of CV's in educated and qualified ones. Nowadays, the company recruiting process has to spend a huge cost and time on selecting the perfect ones for the vacant position.

1. Advertise the vacancy
2. Call Curriculum Vitae of the interested candidates.
3. Shortlisting the applied candidate list by referring their Curriculum Vitae.
4. Interview the shortlisted candidates and recruit the most suitable ones for the position.

But practically, it takes a lot of time and effort for a human to judge an employee's skill and talent just by reading their Curriculum Vitae. Normally, a CV should contain 2 to 3 pages and all the relevant qualifications should be listed there. Because, according to normal policy, the time dedicated to reading one CV is 6 to 7 seconds. The reader should be able to grab the relevant information within that time period.

But, practically, there may be well qualified, talented candidates, who have a large skill set and a CV extended from 7 to 8 pages since it has to hold each and every qualification they achieved. Sometimes, the required qualifications for the specific position they applied, would be included in the last pages of the CV. In this kind of scenario, the reader would miss the important skills or points because they cannot waste much time on one CV. It is much time consuming for a human reader to read one CV end to end. And also, the most qualified candidates would not be called to the

interview just because their CV is too long or not well formatted. It is a huge disadvantage not only to the candidate but also to the company, since the company may lose the best employee to their vacancy.

On the other hand, there may be hundreds of applicants for a vacancy in a large IT industry. In such a scenario, it is very hard and time consuming to download each and every CV and read them one by one in order to shortlist in human hands.

1.2. Background context

One of the best way to download the CV's to use automated web site. There are technology cold Noad.js. Node.js is a platform built on Chrome's JS. It runtime building easily and fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices. So, in that case, can create a website to download the candidate's CV in automatically.

There is a python library call pdfminer. Pdfminer converts the pdf file into text. Natural language processing (NLP) is a branch of Artificial Intelligence (AI). Its ability of a computer program to understand, interpret and manipulate human language. NLP has an open-source library SpaCy.

Pdfminer use to convert pdf file to text and split the word. Then use the pattern and NLP to identify the words. SpaCy library support for 31+ languages. So SpaCy library use to identify the candidates' name and regular expression used to identify the pattern of age, address, e-mail etc…

Pandas is an open source, a BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. This function creates a Pandas DF with one row for every input resume and columns including the resumes file path and raw text

1.3. Research Gap

Even though there are existing proposed products in the market area, they do not address most of the problems that the proposed system is going to address.

Most of the available tools haven't a way to download CV's in automatically and read the downloaded CV's. There is no way to identify the details and write into a CSV format in a proper way.

1.4. Research Objective

1.4.1 Specific Objectives

- Download the CVs in the automatically
- To read the downloaded CVs
- Classify the data into relevant columns
- Save the classified data in CSV format

1.4.2 General Objective

The outcome of this research project will produce a hybrid application to download the candidates CV's, read the CV's and convert it into a csv file within the proper format.

1.5. Research question

Below research problems are being addressed by this system.

- Increasing the efficiency and fast in the website
- Increasing the correctness of identifying the candidate's name.
- The correctness of the identifying the pattern.
- Increasing the efficiency of reading and writing the CV's

2. METHODOLOGY

2.1. Addressing the Literature

2.1.1 Web Page Downloading and Classification

According to the research paper, the system downloads the Web pages using basically using Microsoft's Windows Internet API Tool (Winlnet). Also to go through the links, PDFs, to identify texts, generate the successors of the downloading Web pages, Breadth-First search algorithm, and the Constraint Satisfaction method is used. There are two processes happening in the system; Downloading and Classification. Winlnet is there to connect to web servers when downloading data in different formats; HTML, images, and PDF. Several tasks are done by Winlnet such as requests to the web server for downloading the pages, determining a transfer mode (ASCII or binary) based on the relevant Web page's header. For controlling data flow and to track the downloading progress Breadth-First search algorithm is used. Furthermore to make sure that the downloading moves smoothly without any error, the same Web page is not downloaded twice, and to revisit the unsuccessfully downloaded Web pages again. Breadth-First search algorithm is used because the links among Web pages are similar to a tree structure.

The Breadth-First search is implemented by using two lists. There is an open list and closed list. It's to keep track of the progress through the state space. The open list is maintained as a queue. Queue means first-in-first-out (FIFO). It contains all states that have been generated but whose children haven't been examined. The order in which states are removed from the open list and recorded in the closed list determines the order of the search. When the search is finished, the closed list contains the path of states that have been examined through the search process.

Each list consists of a series of nodes, which contain the uniform resource locator (UIU) addresses of the Web pages. The lists are defined as follows:

- Open list: stores the addresses of pages waiting to be downloaded.
- Closed list: stores the addresses of pages that are successfully downloaded.
- Revisited list: stores the addresses of pages that failed during the downloading process, and are to be revisited later.

```
//Start algorithm
begin
Open := [So 3; // set flag(1) to So
Closed := [];
Revisited := [];
while Open != []
begin
 pick the head node in Open list, call it X;
  if download X succeeds
  generate children of X with flag( I);
  add those children to the tail of Open;
  remove X from Open;
  put X on Closed;
else //download X fails
  remove X from Open;
  if node has flag( 1)
     add X to the head of Revisited;
  else //node has flag(0)
    add X to the tail of Revisited;
  end if
end if
if Open = [] &&Revisited != [] .
  if the head node of Revisited has flag( I )
    remove the node from Revisited;
    set flag(0) to the node;
```

```
    put it on Open;
  end if
 end if
end while
end
//End of the algorithm
```

Under the classification, contents of the hyperlinks will be categorized into texts, abstract PDF files and etc. For this purpose, the Constraint Satisfaction method is used. As the Classification is happening when downloading, only the necessary pages will be downloaded and placed in the relevant directories according to their formats. [1]

2.1.2 Text recognition from image using ANN and Genetic Algorithm

According to the paper, an artificial neural network and genetic algorithm are used to solve effective text recognition problem. Artificial neural networks (ANNs) are a family of statistical learning models inspired by biological neural networks (the central nervous systems of animals, in particular, the brain). ANNs are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. Artificial neural networks are generally presented as systems of interconnected "neurons" which send messages to each other. The connections have numeric weights that can be tuned based on experience, making neural nets adaptive to inputs and capable of learning.

In order to do that a hetero-associative neural network is used to train the system for deciphering digits from pdf or jpeg images which are not readable.

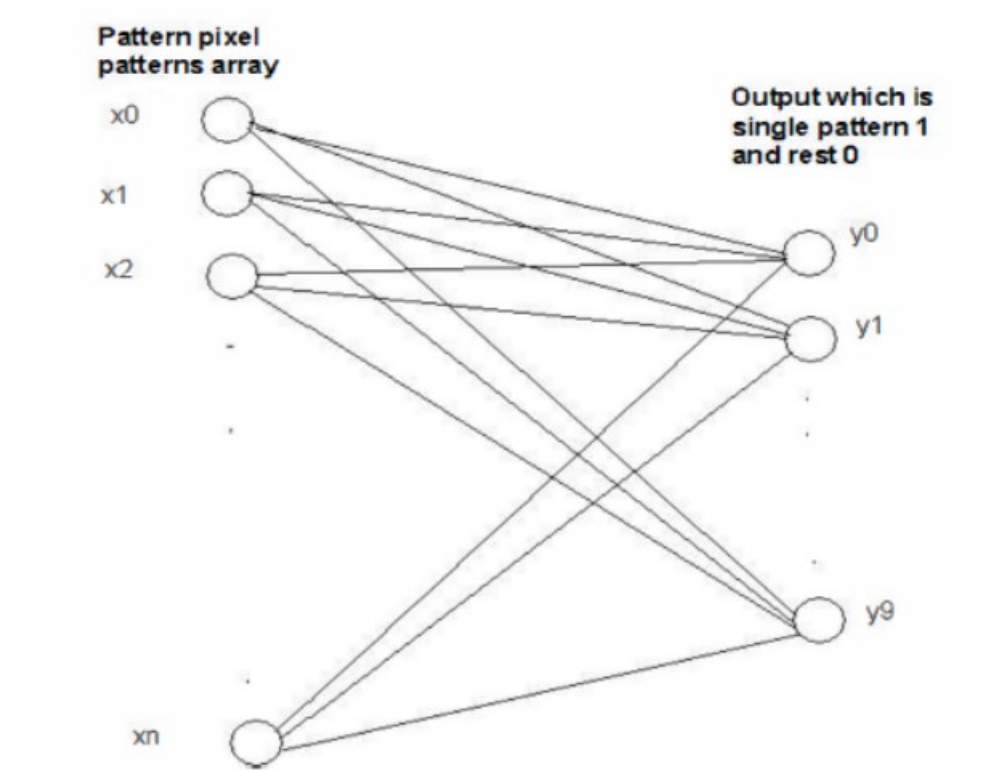

Figure 1- Neural network proposed for digits

For the purpose of analyzing texts from handwritten or text file a crossover based genetic algorithm used. The algorithm solves the problem of deciphering digits and characters from the image. It's done by parsing image and converting it to a pixel array. The algorithm selects digits and characters and performs crossover with trained patterns with variable heights. [2]

2.2. Methodology

This section indicates detailed descriptions about the techniques and mechanisms used to make this project a success. And also how our project is carried out, what are the materials and data needed, and how they will be collected. Apart from that, the research areas that we have identified are explained logically.

The main goal of this research part is to develop an intelligent boot. A bot (short for "robot") is an automated program that runs over the Internet will be implemented to do the following tasks.

- Download the CVs in automatically
- To read the downloaded CV's.
- Classify the data into relevant columns.
- Save the classified data in CSV format.

2.2.1 Download the CVs in automatically

The candidates will upload the CV in the company website, it will automatically download and save the company local space.

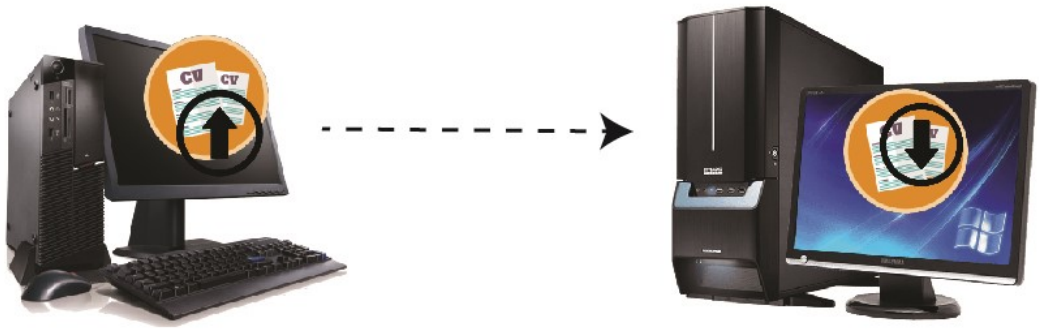

Figure 2- *Download CVs in automatically*

Node.js is using to implement the website. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

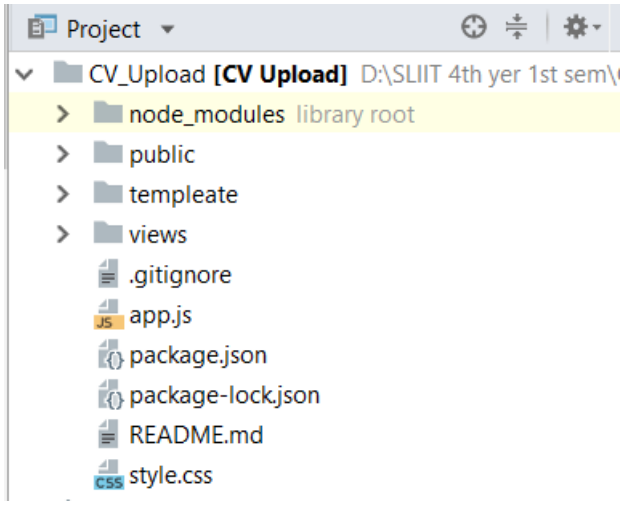

Figure 3- Node modules

2.2.2read the downloaded CV's

Using python library pdfminer convert the pdf file into text.

```
from pdfminer.converter import TextConverter
from pdfminer.layout import LAParams
from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter
from pdfminer.pdfpage import PDFPage
from cStringIO import StringIO
```

Figure 4- Import pdfminer

```
def convert_pdf_to_txt(input_pdf_path):

    try:
        logging.debug('Converting pdf to txt: ' + str(input_pdf_path))
        # Setup pdf reader
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        codec = 'utf-8'
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, codec=codec, laparams=laparams)
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos = set()

        # Iterate through pages
        path_open = file(input_pdf_path, 'rb')
        for page in PDFPage.get_pages(path_open, pagenos, maxpages=maxpages, password=password,
                                      caching=caching, check_extractable=True):
            interpreter.process_page(page)
        path_open.close()
        device.close()

        # Get full string from PDF
        full_string = retstr.getvalue()
        retstr.close()

        # Normalize a bit, removing line breaks
        full_string = full_string.replace("\r", "\n")
        full_string = full_string.replace("\n", " ")

        # Remove awkward LaTeX bullet characters
        full_string = re.sub(r"\(cid:\d{0,2}\)", " ", full_string)

        return full_string.encode('ascii', errors='ignore')
```

Figure 5- convert pdf

2.2.3Classify the data into relevant columns.

Identify the candidate's name using SpaCy NLP library. SpaCy is the best way to prepare text for deep learning. SpaCy can

- Non-destructive **tokenization**
- **Named entity** recognition
- Support for **31+ languages**
- **13 statistical models** for 8 languages
- Pre-trained **word vectors**
- Easy **deep learning** integration
- Part-of-speech tagging
- Labeled dependency parsing
- Syntax-driven sentence segmentation

```
def check_name(string_to_name):

    try:
        string_to_name = unicode(string_to_name)

        nlp = spacy.load('en_core_web_sm')

        doc = nlp(string_to_name)

        doc_entities = doc.ents

        # doc_2 = nlp(my_text)
        for ent in doc.ents:
            if ent.label_ == "PERSON":
                print('{}'.format(ent))
                a = ent
                break
        result = a

        return result
    except Exception, exception_instance:
        logging.error('Issue parsing name: ' + string_to_name + str(exception_instance))
        return None
```

Figure 6- Identify Names

The regular expression used to identify the pattern of age, address, e-mail etc…

```
def check_email(string_to_search):

    try:
        regular_expression = re.compile(r"[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}", re.IGNORECASE)
        result = re.search(regular_expression, string_to_search)
        if result:
            result = result.group()
        return result
    except Exception, exception_instance:
        logging.error('Issue parsing email number: ' + string_to_search + str(exception_instance))
        return None

def check_address(string_to_search):

    try:
        regular_expression = re.compile(r"[0-9]+ [a-z0-9,\.# ]+\bSL\b", re.IGNORECASE)
        result = re.search(regular_expression, string_to_search)
        if result:
            result = result.group()

        return result
    except Exception, exception_instance:
        logging.error('Issue parsing email number: ' + string_to_search + str(exception_instance))

        return None
```

Figure 7- Pattern identify

2.2.4 Save the classified data in CSV format.

After the identify the data, it's writing an XL sheet in a proper format and save the XL file in local space in csv format.

```
def create_resume_df(data_path):

    # Create a list of documents to scan
    logging.info('Searching path: ' + str(data_path))

    # Find all files in the data_path which end in `.pdf`. These will all be treated as resumes
    path_glob = os.path.join(data_path, '*.pdf')

    # Create list of files
    file_list = glob.glob(path_glob)

    logging.info('Iterating through file_list: ' + str(file_list))
    resume_summary_df = pd.DataFrame()

    # Store metadata, raw text, and word count
    resume_summary_df["file_path"] = file_list
    resume_summary_df["raw_text"] = resume_summary_df["file_path"].apply(convert_pdf_to_txt)
    resume_summary_df["num_words"] = resume_summary_df["raw_text"].apply(lambda x: len(x.split()))

    # Scrape contact information
    resume_summary_df["name"] = resume_summary_df["raw_text"].apply(check_name)
    resume_summary_df["gender"] = resume_summary_df["raw_text"].apply(check_gender)
    resume_summary_df["age"] = resume_summary_df["raw_text"].apply(check_age)
    resume_summary_df["languages"] = resume_summary_df["raw_text"].apply(check_languages)
    resume_summary_df["phone_number"] = resume_summary_df["raw_text"].apply(check_phone_number)
    #resume_summary_df["area_code"] = resume_summary_df["phone_number"].apply(functools.partial(term_match, term=r"\d{3}"))
    resume_summary_df["email"] = resume_summary_df["raw_text"].apply(check_email)
    #resume_summary_df["email_domain"] = resume_summary_df["email"].apply(functools.partial(term_match, term=r"@(.+)"))
    resume_summary_df["address"] = resume_summary_df["raw_text"].apply(check_address)
    resume_summary_df["linkedin"] = resume_summary_df["raw_text"].apply(functools.partial(term_count, term=r"linkedin"))
    resume_summary_df["github"] = resume_summary_df["raw_text"].apply(functools.partial(term_count, term=r"github"))

    # Scrape education information
    resume_summary_df["phd"] = resume_summary_df["raw_text"].apply(functools.partial(term_count, term=r"ph.?d.?"))
```

Figure 8- Save data in a csv file

2.3. Research Findings

The purpose of this research was to exploit the flaws of currently available resume filter applications and come up with a unique solution which caters the needs of the company, who expects more and more facilities from web solutions applications, with the ever-evolving nature of modern technology.

****

2.4. Testing

Software testing is an investigation conducted to provide software stakeholders with information about the quality of the product or service under test. It checks whether the actual result matches the expected result. Software testing can also provide the software to allow the business to appreciate and understand the risks of software implementation. Testing is executing a system or application in order to find software bugs, defects or errors.

It involves the execution of a software component or system to evaluate one or more properties of interest.

Unit Testing

Unit testing is a part of a software development process, which is the smallest testable parts of an application.

Integration Testing

Integration testing is a part of a software development process, which program units are combined and tested as groups in multiple ways.

System Testing

System testing, or end-to-end testing, testing of a complete and fully integrated software product. We intend to perform a system testing to ensure that we have achieved all the objectives of our research up to the level of performance expected.

In the testing phase of the BOT, I have followed below steps in order to maintain the product quality.

- Unit Testing – Each interface of the BOT was tested by me produced a defects free unit of coding. Following are the test cases I have followed in my Unit Testing phase.

Test case ID	TC1
Test case Description	Validate CV is uploaded the web page
Pre-Condition	CV uploaded bar is loaded
Test Procedure	Press upload button without select pdf file upload fields.
Test Input	Select File : <Blank>

Expected Output	The error should be displayed saying files doesn't select.
Actual Output	Display an error message saying “File not selected”

Table 2: Test Case 1

Test case ID	TC2
Test case Description	CV is uploaded the web page
Pre-Condition	CV uploaded bar is loaded
Test Procedure	Press the upload button after the select pdf file upload fields.
Test Input	Select File: filename.pdf
Expected Output	Display the message “File uploaded”.
Actual Output	Display the message “File uploaded”.

Table 3: Test Case 2

Test case ID	TC3
Test case Description	Check the new cv is downloaded
Pre-Condition	Get a count of cv’s
Test Procedure	Check the old and new cv count is not equal
Test Input	

Expected Output	Updated the csv file.
Actual Output	Update the csv file.

Table 4: Test Case 3

- Integration Testing – In this testing level I integrated each interface of the BOT and tested.
- System Testing – In this testing level we have integrated the BOT, prediction tool, and dashboard and tested as a whole system.

3. Results & Discussion

3.1. Evidence

The evidence is an important section in any software product in the software industry. The end user can get a clear idea about the project and the final product. It is the associated verification and validation process. The evidence is used to support understand the concept behind the project development.

This is the starting main interface of the application. It contains two-part, cv upload one by one or upload set of cv's.

CV Upload

Can get a basic idea about data upload process from below interfaces.

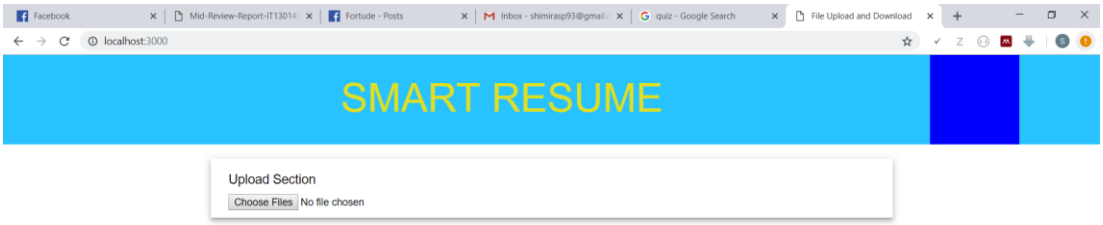

Figure 9: candidate's interface of smart resume

Then can upload the cv one by one.

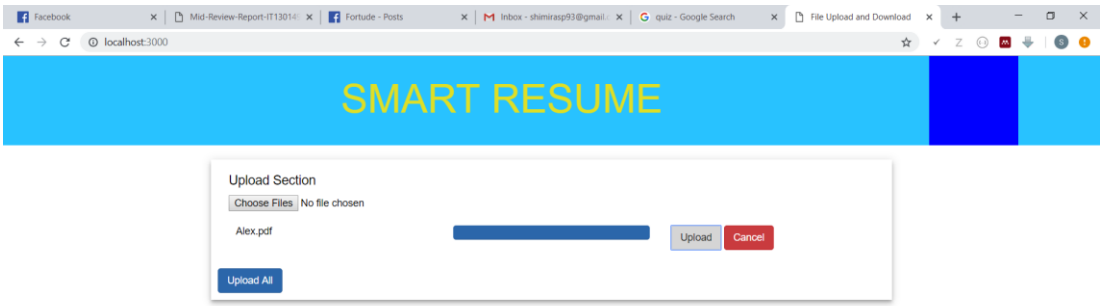

Figure 10: one by one cv upload interface

Also can upload set of cv's

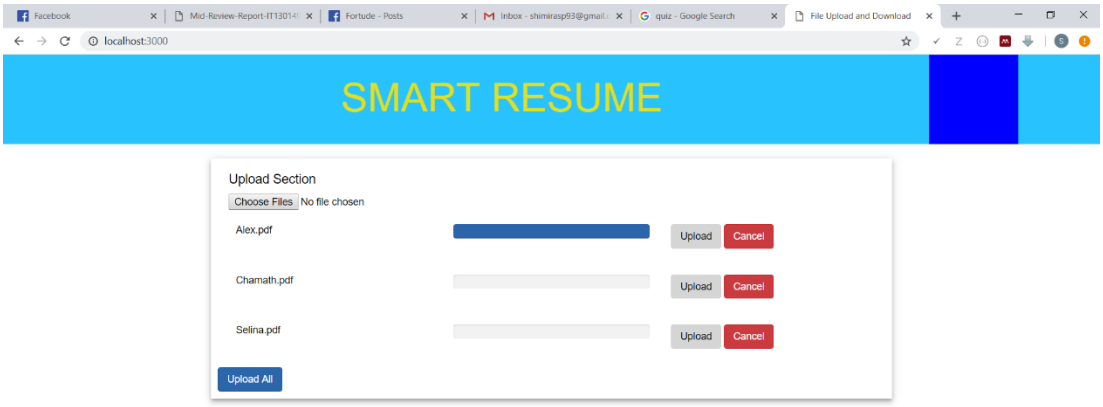

Figure 11: Set of cv's upload interface

Then automatically download the cv's in local space.

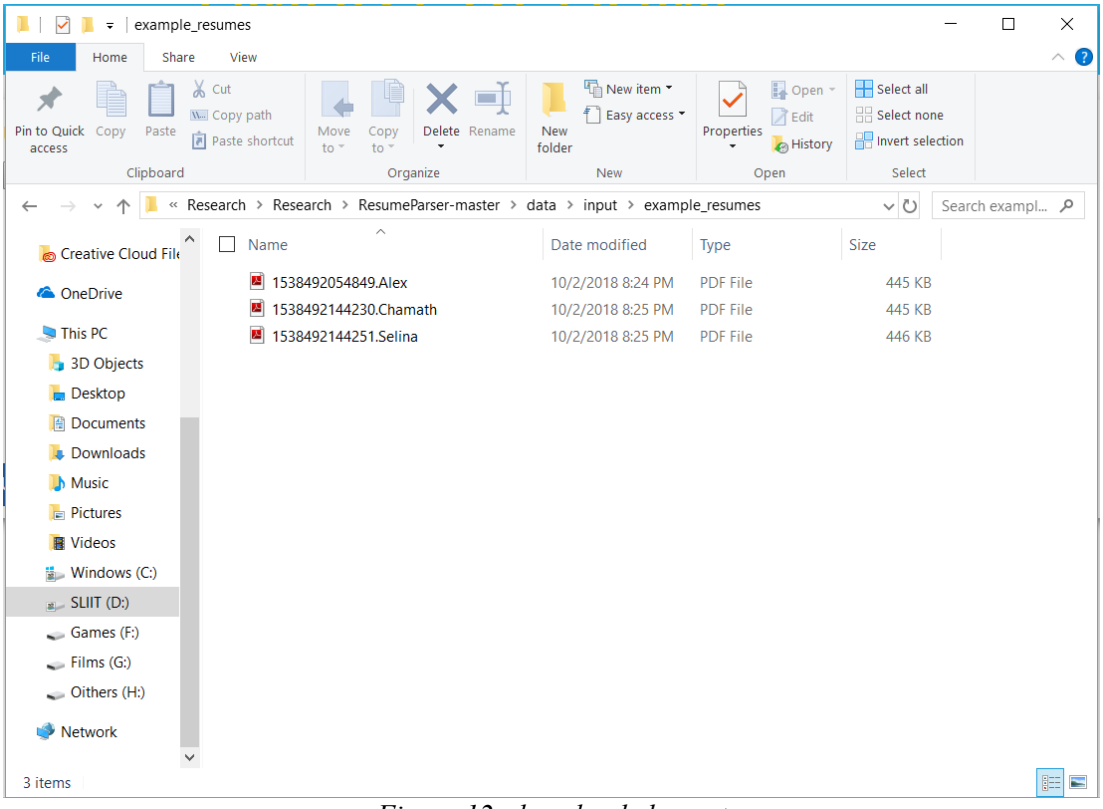

Figure 12: downloaded cv set

Then automatically create the csv files.

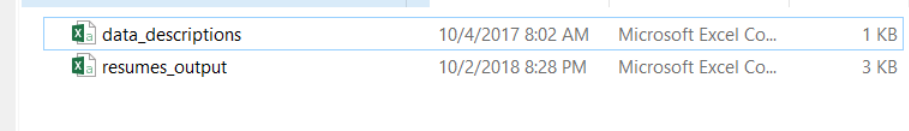

Figure 13: CSV file

CSV files contain the candidate's details.

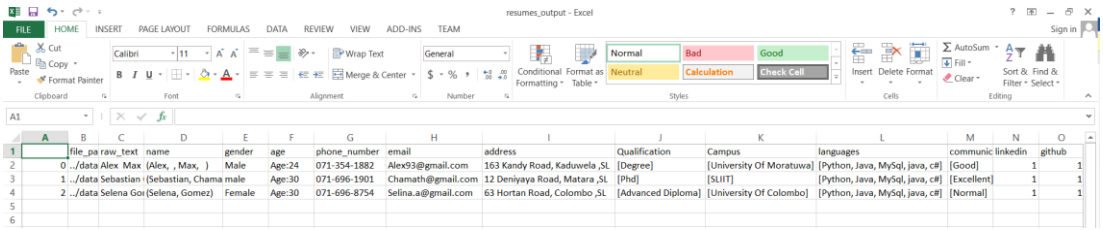

Figure 14: Candidate's data

3.2. Discussion

With hard working through the year, we have completed the Smart Resume BI tool for selecting CVs for the IT Industry. The research was successful because of the help of our supervisor Mr. Lakmal Rupasinghe and my team members who supported to achieve this goal.

Candidates CVs data is the most important data set in this system. To gather these data it was very hardest part of the project as these Candidates data are not given to outsiders as they are very important to the company. The system gets more efficient using NLP libraries.

User-friendliness is there as the interfaces are simpler and more understandable by anyone. Develop the front-end consider about speed & performance. So we use to develop front-end Angular JS. Node.JS used to download cv's in local space.

Read the cv and create the csv part developed by using python. I have to use python 2.7 because python 3.7 doesn't work some libraries. Python library pdfminer used to convert pdf to text. Python NLP library spacy used to identify the candidate's name.

4. Conclusion

Today for most companies like IT, receive a huge number of CVs for a vacancy as there are a lot of graduates coming out from a university within a year. The quality of the company depends on the capabilities of the recruiters. Therefore getting the best-qualified people from incoming applications is very difficult for them. Currently, the selections of CVs is done manually. As the huge number of CVs are summarized by manually, qualified CVs can be skipped by CV errors and human errors. That is where Smart Resume come in to play by reducing all the difficult tasks of selecting an optimal set of candidates in place of referring CVs.

Smart Resume contains a web BOT to download cv ’s, read cv’s and create a CSV data file.

5. References

6. Glossary

7. Appendices

REFERENCES

[1] L. Q. Tran, C. W. Moon, D. X. Le, and G. R. Thoma, “Web Page Downloading and Classification,” *Proc. IEEE Symp. Comput. Med. Syst.*, pp. 321–326, 2001.

[2] M. Agarwal, “Text recognition from image using Artificial Neural Network and Genetic Algorithm,” pp. 1610–1617, 2015.