Project Report

PROJECT 2 – The ‘skills backpack’: e-portfolios for learning

Group: Team 488
Team member:

Yang Yang z5098181 z5098181@ad.unsw.edu.au Scrum Master

Zhiqin Zhang z5105196 z5105196@ad.unsw.edu.au Developer

Han Zhang z5043158 han.zhang6@student.unsw.edu.au Developer

Submission date: 21/10/2018

Table of Contents

1. Overview ... *3*

1.1 Goals of system ... **3**

1.2 Brief functionalities for different types of users ... **3**

1.2.1 Brief functionalities for candidates ... 3

1.2.2 Brief functionalities for employers ... 3

1.2.3 Brief functionalities for instructors ... 3

2. Descriptions of the functionalities ... *4*

2.1 Main functionalities ... **4**

2.1.1 E-portfolio feature ... 4

2.1.2 Recommending feature ... 4

2.1.3 Searching feature ... 4

2.1.4 Scheduling interview features ... 4

2.2 Missing functionalities ... **5**

2.2.1 Like and comment feature ... 5

2.2.2 Recommending course feature ... 5

2.2.3 Tailored portfolio feature ... 5

2.2.4 Privacy feature ... 5

2.2.5 Share portfolio feature ... 6

2.2.6 Share portfolio feature ... 6

3. Third-party functionalities ... *7*

3.1 Flask ... **7**

3.2 Flask-cors ... **7**

3.3 Flask-HTTPAuth ... **7**

3.4 Flask-JWT ... **7**

3.5 Flask-RESTful ... **7**

3.6 spaCy ... **8**

3.7 passlib ... **8**

3.8 psycopg2-binary ... **8**

4. Implementation Challenges ... *9*

4.1 Lack of data ... **9**

4.2 Lack of teammates ... **9**

5. User manual ... *10*

5.1 The requirement of this system ... **10**

5.1.1 Database ... 10

5.1.2 Server ... 10

5.1.3 NLP Process ... 10

5.2 Setup ... **10**

5.2.1 Setup Server part ... 10

5.2.2 Setup database part ... 11

5.2.3 Setup NLP library part ... 11

5.3 Start application ... **12**

5.4 User guide 12
5.4.1 Candidate 12
5.4.2 Employer 13
5.4.3 Instructor 14
Bibliography 16

1. Overview

1.1 Goals of system

The system is mainly focus on resolving two problems, one is helping Computer Science and Engineering students in UNSW demonstrating the skills they acquire from courses as well as general employability skills and soft skills to prospective employers, the other one is helping employers to evaluate potential candidates based on their skills rather than their qualifications. Thus, to comprehensively meet these two goals of the system, this system is designed as a system that has a user-friendly and intuitive dashboard interface with different functionalities for 3 different roles of users, which is candidates, employers and instructors.

1.2 Brief functionalities for different types of users

In our design, different types of users can use different functionalities trailed for their own as well as share some common functionalities.

1.2.1 Brief functionalities for candidates

After signed up as a candidate of user types, you would access to your own dashboard so that you can edit your e-portfolio, search and save different jobs as well as get recommended jobs from instructors and system based on your technic, employability and soft skills.

1.2.2 Brief functionalities for employers

After signed up as an employer of user types, you would access to your own dashboard so that you can post a job, search and save different candidates as well as get recommend candidates from instructors and system based on how much jobs match with the skills of candidates. Also, you would be able to schedule 6 phases of interview for a candidate you like.

1.2.3 Brief functionalities for instructors

After signed up as an instructor of user types, you would access to your own dashboard so that you can see your saved jobs and saved candidates after searching and saving them. You would be able to make a connection between a job you saved and a candidate you saved so that they can get your recommendation as a reference.

2. Descriptions of the functionalities

2.1 Main functionalities

In order to help us resolving the above two problems mentioned above in overview, we built four main functionalities for different types of users which are e-portfolio feature, recommending feature, searching feature and scheduling interview feature.

2.1.1 E-portfolio feature

Providing e-portfolio for candidates is the most important core of this project. Candidates will have an automatically generated e-portfolio to show to both instructors and employers once they finished fill in their information in their dashboard. This progress is dynamically happened in our system so that you don’t need to manage database for it. Other than that, this e-portfolio will be update every time in the system for any single change of students’ personal details so that any e-portfolio saw by anyone is always the newest version.

2.1.2 Recommending feature

Recommending feature is to recommend suitable jobs to candidates depends on their skills by system. We believe that technic skills, soft skills and employability skills will all effect the performance of a person working on a specific job. Therefore, our recommending features will recommend jobs to candidates with considering these three different types of skills. What has to be emphasized here is the technic skills we use to analyse in our system is not based on the technic skills candidates fill in by themselves, instead, skills that you gain from courses you have taken is the skills we actually extract to analyse in the algorithm. Thus, recommending result will be more reliable and trustable.
Also, recommending features will recommend suitable candidates to employer with measuring how much their skills match with job requirement which is based on the same idea of recommending jobs to candidates.
Last but not least is recommending features can manually use by an instructor to recommend a candidate and a job to each other which we called making connection. Connection is very important for candidates to be attached to industry when they still in their education. Rather than applying jobs by themselves, making connections for candidates will provide them high opportunity to be considered to enter an interview.

2.1.3 Searching feature

Searching feature is for all roles of users to searching either jobs or candidates in our system database. We made a few categories and skill tags to help get more precise results rather than matching with keyword only. Besides these benefits, searching results can also be saved by each user to either view or do other operations later based on user role.

2.1.4 Scheduling interview features

Scheduling interview feature is to help employer manage interview in a simple and visible way. We split a whole process of interview into 6 phases so that employers will be able to see amounts of candidates they have in each phase clearly, which means even different staffs who share one employer account will be able to know the whole process conveniently. Once employers are satisfied with the performance of one candidate in current phase, they can easily move this candidate to next phase by just one click.

2.2 Missing functionalities

We finished most of essential functionalities from our project proposal, but there are still some missing functionalities due to different reasons. We will describe the missing features and discuss about reasons in the following sections.

2.2.1 Like and comment feature

Description: As a student, I want to know if my peers and teachers appreciate my works using likes and comments features.

Reasons: We thought it will be a good feature if recommend feature can put amount of likes of a portfolio as an element to measure quality of a portfolio. Since algorithm logic of recommending features is nothing relative to this feature, we decided to give up on implementing this feature.

2.2.2 Recommending course feature

Description: As a student, I want to know which courses I should choose in my following semesters.

Reasons: This feature is not very helpful to resolve the problems mentioned above in overview. Besides, it needs to change whole structure of our system to blend itself in our system. Instead, the relationship of skills and courses are been used by recommending feature. Therefore, due to time limitation for this project, we decided to give up on implementing this feature before deadline.

2.2.3 Tailored portfolio feature

Description: As a student, I want to be able to present my portfolios tailored to different job applications.

Reasons: This feature is too complicated to implement. Instead, we generate e-portfolio in the same format and use recommend system to help with job applications.

2.2.4 Privacy feature

Description: As a student, I want to share my portfolio with selected people.

Reasons: This feature is not very helpful to resolve the problems mentioned above in overview. Besides, it needs to change whole structure of our system to blend itself in our system. Therefore, due to time limitation for this project, we decided to give up on implementing this feature before deadline.

2.2.5 Share portfolio feature

Description: As an employer, I want to share portfolios of different candidates with my recruitment team.

Reasons: There is no such a function that one account can communicate with another account. It will take much time to implement communication in our system. Instead, we designed a very intuitive interface for scheduling interview feature so that once they share the same account, it will be intuitive to know whole progress of interview for different jobs.

2.2.6 Share portfolio feature

Description: As an employer, I want to be able to message users directly through the platform.

Reasons: There is no such a function that one account can communicate with another account. It will take much time to implement communication in our system. Therefore, due to time limitation for this project, we decided to give up on implementing this feature before deadline.

3. Third-party functionalities

3.1 Flask

Flask: The reason we use this microframework is that its very light to build a web application and suitable to be an API provider.
Website: http://flask.pocoo.org/ (Ronacher, n.d.)
License: https://github.com/pallets/flask/blob/master/LICENSE (davidism, 2010)

3.2 Flask-cors

Flask-cors: This is a separation of frontend and backend application; therefore, the front-end and back-end communication involve cross-domain issues. And this library is a Flask extension which could make cross-origin AJAX request possible.
Website: https://flask-cors.readthedocs.io/en/latest/ (Flask-CORS, 2013)
License: https://github.com/corydolphin/flask-cors/blob/master/LICENSE (corydolphin, 2016)

3.3 Flask-HTTPAuth

Flask-HTTPAuth: The backend only provide Restful API, therefore cannot save communication state. In another words, can't rely on cookies and sessions to save user information. Due to this reason, it's impossible to use Flask-Login extension to implement user authentication. and this library could solve this problem.
Website: https://flask-httpauth.readthedocs.io/en/latest/ (flask-HTTPAuth, n.d.)
License: https://github.com/miguelgrinberg/Flask-HTTPAuth/blob/master/LICENSE (Grinberg, 2013)

3.4 Flask-JWT

Flask-JWT: JSON Web Tokens (JWT) is a new technology for identifying users which is very suitable for application which separation of frontend and backend. And This library provides a python implementation of JWT.
Website: https://pythonhosted.org/Flask-JWT/ (Flask-JWT, Flask-JWT, n.d.)
License: https://github.com/mattupstate/flask-jwt/blob/master/LICENSE (Wright, 2014)

3.5 Flask-RESTful

Flask-RESTful: This is a Flask extension that adds support for quickly building REST APIs. So, we use this library to implement Resourceful Routing and provide API for frontend.
Website: https://flask-restful.readthedocs.io/en/latest/ (flask-restful, n.d.)
License: https://github.com/flask-restful/flask-restful/blob/master/LICENSE (Twilio, n.d.)

3.6 spaCy

spaCy: This is a well-known Natural Language Processing (NLP) library which provide some model to do processing more easily. We use this library to do NLP for candidate about me part and job summary part.
Website: https://spacy.io/ (spaCy, n.d.)
License: https://github.com/explosion/spaCy/blob/master/LICENSE (GmbH, 2016)

3.7 passlib

passlib: Passlib is a password hashing library for Python 2 & 3, which provides cross-platform implementations of over 30 password hashing algorithms, as well as a framework for managing existing password hashes. So, we use this library to do Password encryption and verification.
Website: https://passlib.readthedocs.io/en/stable/ (Passlib, n.d.)
License:
https://bitbucket.org/ecollins/passlib/src/849ab1e6b5d4ace4c727a63d4adec928d6d72c13/LICENSE?at=default&fileviewer=file-view-default (Collins, n.d.)

3.8 psycopg2-binary

psycopg2-binary: This library is used to connect PostgreSQL database from python.
Website: (psycopg, psycopg, n.d.)
License: http://initd.org/psycopg/license/ (psycopg, psycopg, n.d.)

4. Implementation Challenges

4.1 Lack of data

Main feature of this system is recommendation system, which rely on big data to find the relation between courses and skills as well as skills between job titles. For relation between courses and skills, unfortunately the course outline on UNSW handbook are brief and summary. Since we don't have enough data to find relation between them. As for relation between skills and job titles, because the open related dataset cannot be found, the analysis of this relationship is very rough. In addition, different companies have different requirements for the same job title. So, it's hard to analyse a common skill set. Due to the above two reasons, the effect of the recommendation system did not meet expectations.

4.2 Lack of teammates

Since we only have 3 people in our team, we have to give up some extensional features before deadline. Also, this affect our working progress of sprint 3 because front end and backend are working well respectively without connection of API.

5. User manual

5.1 The requirement of this system

5.1.1 Database

Using Postgresql which run on localhost port 5432 with a database named "comp9900". For detail of setup database, see 5.2.2.

5.1.2 Server

Python3. For detail, see 5.2.1.

5.1.3 NLP Process

Spacy library and statistical model "en_core_web_sm". For detail, see 5.2.3.

5.2 Setup

5.2.1 Setup Server part

This application use Python3 as running environment and Flask as framework. To setup server, run the following command.

On your own machine:

```
# install require library
$ pip3 install -r requirements.txt
```

On your own machine, in virtualenv:

```
# create a sandbox for the backend
$ python3 virtualenv -p /usr/bin/python3 env
# enter sandbox
$ source env/bin/activate
# set up sandbox
$ pip install -r requirements.txt
```

On CSE machine: (Not recommended, you may face cannot find gcc compiler error)

```
# create a sandbox for the backend
$ virtualenv -p /usr/bin/python3 env
# enter sandbox
$ source env/bin/activate
# set up sandbox
$ pip install -r requirements.txt
```

If you could not compile locally on CSE machine, we have prepared the pre-compiled library. You can download the virtualenv which include compiled librarys using below link:

WARNING:

Due to CSE machine disk space limitation, there is no guarantee that the precompiled library will work normally.

For x86-64: https://drive.google.com/open?id=1Zr6b3AWMivTRnsHLyyb2C4WIqj6BqzdV

For i686: https://drive.google.com/open?id=1KiNfVO5MJEqY_u_sAtyv0uQNxGZtfJs5

After download virtualenv, decompression file and copy all folders and files EXCEPT bin folder to virtualenv folder you created on CSE machine.

5.2.2 Setup database part

This application use Postgresql as database to maintain data.

Make sure that you have a Postgresql server running on localhost:5432.

Make sure that createdb, dropdb, psql commands could be used.

On your own machine:

```
$ createdb comp9900
$ psql comp9900
comp9900=# \i init.sql
comp9900=# \q
```

Once you finish using this application, run the following command to drop whole database.

```
$ dropdb comp9900
```

On CSE machine:

```
$ pg_virtualenv
$ createdb comp9900
$ psql comp9900
comp9900=# \i init.sql
comp9900=# \q
```

Once you finish using this application, run the following command to exit pg virtualenv.

```
$ exit
```

5.2.3 Setup NLP library part

This application uses Spacy as the NLP library to implement the recommendation job feature. Once you have setup server, Spacy has already been installed in your computer. Run the following command to finish setup NLP library.

On your own machine:

```
$ python3 -m spacy download en_core_web_sm
```

On your own machine, in virtualenv:

```
$ python -m spacy download en_core_web_sm
```

On CSE machine, in virtualenv:

```
$ python -m spacy download en_core_web_sm
```

5.3 Start application

Once you finish setup requirement for this application, you can use following command to start this application.

This application will run on localhost:5000, so make sure that no other service or application running on port 5000.

On your own machine:

```
$ python3 backend.py
```

One your own machine, in virtualenv:

```
$ python backend.py
```

On CSE machine, in virtualenv:

```
$ python backend.py
```

The application should be running on localhost:5000 now.

5.4 User guide

5.4.1 Candidate

1. Sign up a new account with selecting user type Candidate.

2. After successfully sign up, login with this candidate account and you will access to a candidate dashboard. You can see an empty resume page with an icon on the upper right corner and several tabs under title "Portfolio" as well as a "Search" button on the upper left corner.

3. You will access to a new page to update your profile by selecting "My profile" after clicking on the icon on the upper right corner. You will be able to update your full name, your birthday, your gender, your email address and also your password for this account in different fields by clicking on "UPDATE", filling on your information and clicking on "CONFIRM". You will actually update your details in the system by clicking "Save" button.

4. You can choose to upload your photo by clicking on the second icon on the left side of your profile. Click on "Upload photo" and click on "Upload" to save to the system.

5. After you finish uploading photo and updating profile, you can close this profile page and details that you just changed will update in your dashboard.

6. After clicking on "Skill Set" tab, you can add your technic skills and the courses you have taken by filling corresponding fields and clicking on "Add" button.

7. After clicking on "Projects" tab, you will be able to add your projects by clicking on "+" button. After clicking on pencil icon, you will be able to fill in corresponding fields with your information with

clicking on pencil icon again to finish editing. You can edit your existing project details by clicking on pencil icon and delete them by clicking on rubbish bin icon.
8. After clicking on “Contact” tabs, you can modify your contact detail by clicking on pencil icon, filling the information and clicking on save icon.

9. After you finish all the above operations. You can click on “Home” tab and then modify About me, Work Experience, Education and Personal skills by clicking on pencil icon, filling the information and clicking on pencil icon again to finish editing and you can delete them by clicking on rubbish bin icon. After you finish all the operations you want, clicking on “Update” to update all your information on Porfolio.

10. By clicking “Search” button on the upper left corner, you will open a new search page. On search page, you could search job by typing job title and location and clicking on “Search” button. You can also refine your result by choosing other categories after clicking on “+” button. (By using 'any' as a keyword, you could view all exists jobs)

11.Once you are on search results page, you can click on each job and view details of this job and save this job by clicking on “Save this job” button.

12. Back to your own dashboard, you can see your saved jobs and some recommended jobs by instructors and system by clicking on “Jobs” tab. (Might need to refresh the page). You can delete your saved jobs easily by clicking on rubbish icon next to corresponding job. You can also add your demand to “I’m looking for a job that:” by filling in the field and clicking on “Add” button.

13. By clicking job title or recommended job title under “Job” tab, you can view the details of job and apply for this job by clicking on “Apply”. Your application will show up on employer’s dashboard dynamically.

5.4.2 Employer

1. Sign up a new account with selecting user type Employer.

2. After successfully sign up, login with this candidate account and you will access to an employer dashboard. You can see 3 tabs which is “Opening jobs”, “Candidates” and “Interview” with an icon on the upper right corner and a “Search” button on the upper left corner.

3. You will access to a new page to update your profile by selecting “My profile” after clicking on the icon on the upper right corner. You will be able to update your full name, your birthday, your gender, your email address and also your password for this account in different fields by clicking on “UPDATE”, filling on your information and clicking on “CONFIRM”. You will actually update your details in the system by clicking “Save” button.

4. You can choose to upload your photo by clicking on the second icon on the left side of your profile. Click on “Upload photo” and click on “Upload” to save to the system.

5. After you finish uploading photo and updating profile, you can close this profile page.

6. You can create a new job by clicking on “+” button under “Opening Jobs” tab. On the popup interface, filling work-related information as requested. By Clicking on “Create a job” button, this job will be saved to the system and published. You can find the job you just posted under “Opening

Jobs” tab after the above operations and you can click on “More Info” button to view more information of this job. You can also see the numbers of candidates in different phase of interview.

7. By clicking “Search” button on the upper left corner in your dashboard page, you will open a new search page. On search page, you can search candidates by typing keyword about candidate, location and clicking on “Search” button after clicking on “Finding Candidates?” button. You can also refine your result by choosing other categories after clicking on “+” button. (By using 'any' as a keyword, you could view all exists candidates)

8. In search results page, you can click on each candidate and view his/her e-portfolio and save this candidate to your candidate list.

9. Back to your own dashboard, you can find the candidates that you have saved and some recommended candidates by clicking on “Candidates” tab.

10. By clicking on saved candidate or recommended candidate under “Candidates” tab, you will be able to view e-portfolio of this candidate and schedule an interview for him/her.

11. By clicking on “Interview” tab, you can find all the candidates with different status in interview phase.

12. You can click on “More Info” to view e-portfolio of different candidate and schedule the next phase interview to him/her. After this operation, you can see a number add to the next phase which means you have scheduled an interview of next phase to one candidate.

5.4.3 Instructor

1. Sign up a new account with selecting user type Instructor.

2. After successfully sign up, login with this candidate account and you will access to an employer dashboard. You can see 3 tabs which is “Saved jobs”, “Candidates” and “Connections” with an icon on the upper right corner and a “Search” button on the upper left corner.

3. You will access to a new page to update your profile by selecting “My profile” after clicking on the icon on the upper right corner. You will be able to update your full name, your birthday, your gender, your email address and also your password for this account in different fields by clicking on “UPDATE”, filling on your information and clicking on “CONFIRM”. You will actually update your details in the system by clicking “Save” button.

4. By clicking “Search” button on the upper left corner, you will open a new search page. On search page, you could search job by typing job title and location and clicking on “Search” button. You can also refine your result by choosing other categories after clicking on “+” button. (By using 'any' as a keyword, you could view all exists jobs)

5. Once you are on search results page, you can click on each job and view details of this job and save this job by clicking on “Save this job” button.

6. By clicking “Search” button on the upper left corner in your dashboard page, you will open a new search page. On search page, you can search candidates by typing keyword about candidate, location and clicking on “Search” button after clicking on “Finding Candidates?” button. You can also

refine your result by choosing other categories after clicking on “+” button. (By using 'any' as a keyword, you could view all exists candidates)

7. In search results page, you can click on each candidate and view his/her e-portfolio and save this candidate to your candidate list.
8. Back to your own dashboard, you can find your saved jobs under “Saved Jobs” tab. By clicking on “More Info” button of a saved job, you can see more information about this job.

10. By clicking on “Candidates” tab, you can find your saved candidates under this tab. By clicking on “More Info” button of a saved candidate, you can see more information about this candidate.

11. By clicking on “Connections” tab, you can find the connection you made between a candidate and an employer. It will be recommended You can create a connection between saved job and saved candidate by clicking on “+” button and then drag one of saved jobs to “Target Job” field and one of saved candidates to “Target Candidate” field. After that, you will be able to click on “Recommend” button to create a connection. Both candidate and employer will receive recommendation on their dashboards. By clicking on rubbish bin icon, this connection will be deleted.

Bibliography

Collins, E. (n.d.). *LICENSE*. Retrieved from bitbucket: https://bitbucket.org/ecollins/passlib/src/849ab1e6b5d4ace4c727a63d4adec928d6d72c13/LICENSE?at=default&fileviewer=file-view-default

corydolphin. (2016). *flask-cors LICENSE*. Retrieved from github: https://github.com/corydolphin/flask-cors/blob/master/LICENSE

davidism. (2010). *flask License*. Retrieved from github: https://github.com/pallets/flask/blob/master/LICENSE

ExplosionAI UG (haftungsbeschränkt), 2. s. (n.d.).

Flask-CORS. (2013). *Flask-CORS*. Retrieved from Flask-CORS: https://flask-cors.readthedocs.io/en/latest/

flask-HTTPAuth. (n.d.). *flask-HTTPAuth*. Retrieved from flask-HTTPAuth: https://flask-httpauth.readthedocs.io/en/latest/

Flask-JWT. (n.d.). *Flask-JWT*. Retrieved from Flask-JWT: https://pythonhosted.org/Flask-JWT/

Flask-JWT. (n.d.). *Flask-JWT*. Retrieved from Flask-JWT: https://pythonhosted.org/Flask-JWT/

flask-restful. (n.d.). *flask-restful*. Retrieved from flask-restful: https://flask-restful.readthedocs.io/en/latest/

GmbH, s. (2016). *spaCy LICENSE*. Retrieved from github: https://github.com/explosion/spaCy/blob/master/LICENSE

Grinberg, M. (2013). *Flask-HTTPAuth License*. Retrieved from github: https://github.com/miguelgrinberg/Flask-HTTPAuth/blob/master/LICENSE

Passlib. (n.d.). *Passlib 1.7.1 documentation*. Retrieved from Passlib: https://passlib.readthedocs.io/en/stable/

psycopg. (n.d.). *psycopg*. Retrieved from psycopg: http://initd.org/psycopg/

psycopg. (n.d.). *psycopg*. Retrieved from psycopg: http://initd.org/psycopg/license/

psycopg. (n.d.). *psycopg*. Retrieved from psycopg: http://initd.org/psycopg/license/

Ronacher, A. (n.d.). *Flask*. Retrieved from Flask: http://flask.pocoo.org/

spaCy. (n.d.). *Industrial-Strength Natural Language Processing IN PYTHON*. Retrieved from spaCy: https://spacy.io/

Twilio. (n.d.). *flask-restful LICENSE*. Retrieved from github: https://github.com/flask-restful/flask-restful/blob/master/LICENSE

Wright, M. (2014). *flask-jwt LICENSE*. Retrieved from github: https://github.com/mattupstate/flask-jwt/blob/master/LICENSE