Project Progress Report

A reference materials recommendation system in personal projects

—— improving the productivity in software engineering

Team Name: Two Engineers
Jiayi Li, jl4924, jl4924@columbia.edu
Chi Zhang, cz2465, cz2465@columbia.edu

1. Introduction

1.1 Motivation

According to the survey of our midterm papers, we find that for most developers, when they begin to work on new projects, they may care more about libraries, toolboxes as well as useful packages of their projects rather than programming languages' execution speeds, memory usage, etc., even though these programming languages' features are also very important in improving one's productivity. We also find that the developer community is another key factor that developers may consider.

But for the most time, it is always not easy to find these reference resources that developers care more, as we have mentioned above, from the Internet in a short time. As is known to all, there is an ever-growing variety of open-sourced demos, useful packages & libraries, efficient tools, academic papers, etc. on the Internet. As a result, given such a large amount of possibly useful online resources, it is really difficult for developers to find the most appropriate ones for their projects, which may make them feel really confused. And the process of searching appropriate resources online may also be time-consuming. Thus, we want to develop a web application to help developers shorten their time spent on searching resources.

1.2 What our system does

1.2.1 General idea

We plan to build a recommendation system to help developers or graduate students to find useful reference materials of their projects in an easier way in order to improve their developing productivity and work efficiency. **(what value our system provides to our target community)** We conducted a survey among graduate students at Columbia University to find the most popular project topics (research fields) among them and the most useful reference materials that will help them most in their projects. Based on the results we get from our survey, we build our recommendation system to meet their demands. We also put more emphasis on the most popular research fields as well as the most useful reference materials. We extract many useful reference materials from the internet including related course list, the course description, GitHub demos, programming language tutorials, and developer communities, which covers 11 different research fields and 9 different programming languages. We build a web application to display all these reference materials in one web page.

1.2.2 System structure

Figure 1.1 shows the architecture of the web application we have built. We use flask [1] to establish the back-end of our recommendation system, coding in Python. It can send requests through the GitHub API v3 [2] and receive responses from it. On the other side, the back-end communicates with the front-end using Ajax [3] in jQuery [4], which can refresh certain sections in the web page, rather than the whole page. The dynamic database (project demos) is built in the back-end (Python). The front-end of our

recommendation system is built based on a toolkit called Bootstrap [5], coding in JavaScript and HTML. And the static database of our recommendation system is built in the front-end (JavaScript), which mainly consists of three parts: the course information, programming language tutorials and developer communities.

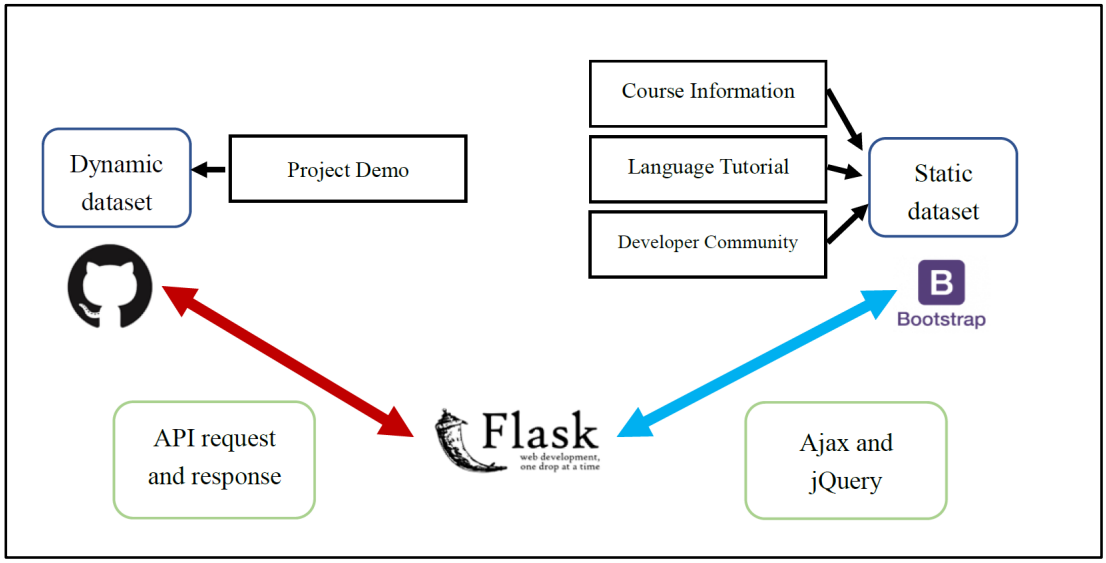

Figure 1.1 Architectural organization diagram

1.3 Innovation point

(what is novel?)

We have not found any similar recommendation system yet. In this case, our project is really meaningful and the idea of building such a reference materials recommendation system should be innovative. We think our recommendation system meet demands of most graduate students at Columbia University. Our recommendation system will mainly focus on research fields that are related to courses offered by Columbia University. As a result, our recommendation is more specific and more helpful for Columbia graduate students. Reference materials provided by our recommendation system are highly related to courses offered by Columbia University, which could help graduate students at Columbia University to find reference materials they might need for their course projects in an efficient way.

1.4 Target user community

As it is mentioned before, the target community of our recommendation system is graduate students at Columbia University (especially for SEAS), who always have trouble in finding reference materials for their course projects. The reason we choose graduate students rather than undergraduates students as the group that we will focus on is that graduate students may focus on some more specific research fields which would be easier to classify. And we are also more familiar with graduate study mode and truly understand what kinds of resources graduate student need and prefer in their graduate-level course projects. On the contrary, the undergraduate study may be more general compared to the graduate study, which will be a great challenge for us to classify and recommend reference materials. And to be honest, we are also not so familiar with their demands in their projects.

1.5 The data we used

The data we used in our recommendation system mainly consists of two parts. One is the dynamic data and the other is static data. For the dynamic data, we get dynamic repositories information from Github API in the order of their popularity based on the search field and programming language we input. For the static data, we extracted the related course list, the course description, programming language tutorials, and developer communities from the internet, which covers 11 different research fields and 9 different programming languages.

1.6 How to evaluate our system

We use the WLAN to launch our recommendation system and users can get access to our system by visiting the IP address and port number we set from the browser of their own devices. After they experiencing our recommendation system, they are asked to make comments in the feedback input boxes before they existed our system.

2. Preliminaries and background

2.1 GitHub API v3

"The GitHub search API is optimized to help the user to find the specific item, for example, a specific user, a specific file in a repository, etc. Think of it the way you think of performing a search on Google. It is designed to help people find the one result or maybe the few results we may want to find. Just like searching on Google, sometimes we want to see a few pages of search results so that we can find the item that best meets our needs. To satisfy that need, the GitHub Search API provides up to 1,000 results for each search" [2].

We implement the official GitHub API, coding in Python with library urllib2 [6]. The urllib2 library defines some functions and classes "which help in opening URLs (mostly HTTP) in a complex world — basic and digest authentication, redirections, cookies and more." [6] We define a Request function and then save all responses to the local database as a .csv file. Detail description and samples will be shown later.

2.2 Flask

"Flask is a microframework for Python based on Werkzeug, Jinja 2 and good intentions" [1]. It is one of the most important tools we have used to build our web application. By saving the reference materials (the dynamic and static ones) in the local database, we are able to show all of them in our web application with the use of Flask. The implementation of Flask will be described later.

2.3 Bootstrap

The toolkit we used for our front-end is Bootstrap, "one of the world's most popular front-end component library. It is a powerful open-source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery." [5]

3. Market Investigation

3.1 Questionnaire design

To figure out which fields and which reference materials that our project should focus on, we conducted a survey using the wjx survey service [7]. This survey aims at all graduate students at Columbia University. To restrict the respondents of our survey, the survey was mainly distributed to graduate students majoring in Electrical Engineering and Computer Science. And they were also asked to distribute this survey to their friends at Columbia University if possible. Here are questions of our questionnaire:

Q1: What is your major?
(Options: CS, EE, CE, Others).

By asking this question, we try to make sure our respondents all come from target majors, such as CS, EE, CE. Since students from these three majors may be more familiar with research fields we have selected for the Q2 & Q3. As a result, their responses may be more valuable.

Q2: Which of the following research fields are you interested in?
(Options: Data Analytics, Machine Learning, Deep Learning, Reinforcement Learning, Speech/Text Processing and recognition(NLP), Computer Vision/ Image Processing, Distributed Systems, Internet of things, Computer Networks, Neuroscience, Security, Others.)

By asking this question, we try to figure out which research field should be the most popular one among our respondents. According to the result we got from this question and the one get from the Q3, we will build our recommendation system mainly on the Top 3 research fields (or all of them if time permits).

Q3: Which of the following research fields have you gotten into before?
(Options: Data Analytics, Machine Learning, Deep Learning, Reinforcement Learning, Speech/Text Processing and recognition(NLP), Computer Vision/Image Processing, Distributed Systems, Internet of things, Computer Networks, Neuroscience, Security, Others.)

Similar to the Q2, we also try to figure out which research field should be the most popular one among our respondents. But in this question, we evaluate the popularity in research fields that respondents have gotten into before instead of research fields that they are interested in.

Q4: Do you think it is a troublesome and time-consuming thing to search for reference materials during your projects?
(Options: Yes, No)

By asking this question, we want to make sure that there is a need to build such a reference materials recommendation system and our system can sharply reduce their troubles in finding useful reference materials.

Q5: How long will it take you to find reference materials you need during your project (in average)?
(Options: Less than 10% of your development cycle, 10%~20% of your development cycle, 20%~30% of your development cycle, 30%~50% of your development cycle, More than 50% of your development cycle.)

Similar to the Q3, by asking this question, we want to make sure that there is a need to build such a reference materials recommendation system. But in this question, given respondents' average time costed in finding reference materials, we are able to evaluate how much our recommendation system can save users' time on finding reference materials. And we can also use results gained from this question to compute the efficiency improvement rate of our recommendation system in future.

Q6: Which of the following reference materials do you think help you most in your projects?

(Options: similar project demos, powerful packages and libraries, related papers or tutorials, related courses, others.)

By asking this question, we try to figure out what kind of reference material may mostly improve users' developing efficiency. We will build our recommendation system based on results we gained from this question. And we will mainly focus on reference materials that can improve users' productivity significantly.

3.2 Questionnaire response

The survey was mainly distributed to some networks of graduate students that we know. And respondents were also asked to distribute this survey to their friends at Columbia University if possible. Through this method, we totally received 50 responses within 2 days.

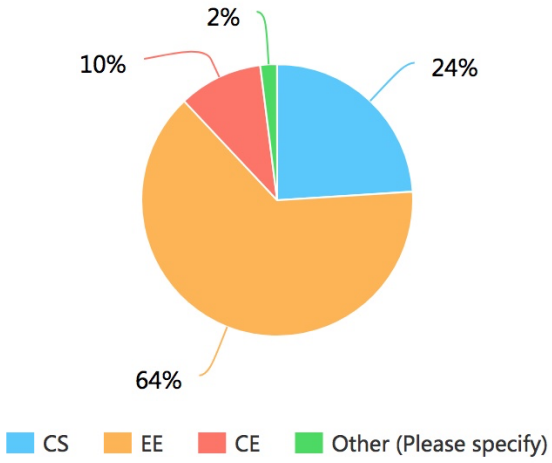

Figure 3.1 Respondents' major distribution

Most of our respondents are majoring in Electrical Engineering, accounting for 64% of the whole respondents. And 24% of our respondents are majoring in Computer Science while 10% of our respondents are majoring in Computer Engineering. Only one of our respondent does not come from our target majors (CS, EE, CE).

3.3 Results analysis

Q2: Which of the following research fields are you interested in?

In this question, respondents were asked to select one or more research field that they are interested in. Figure 3.2 shows responses of this question.

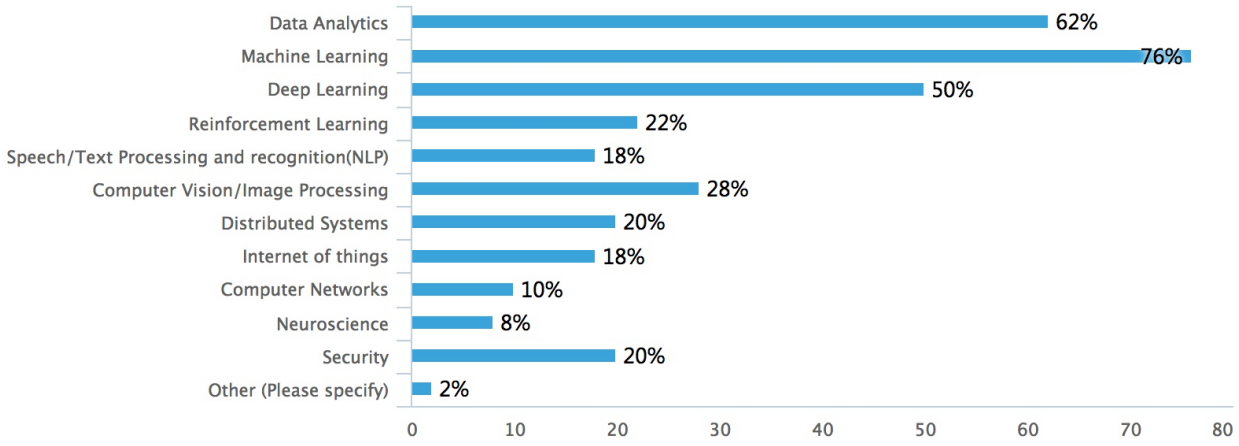

Figure 3.2 Most interested research fields among our respondents

From responses, a surprisingly 76% of respondents said that they are interested in Machine Learning while 62% of respondents said that they are interested in Data Analytic. And Deep Learning are also sharing 50% of respondents. It is obvious that Machine Learning, Data Analytics, Deep Learning are pretty popular among our respondents. Most of them show great interests in these three research fields, which are all related to AI, one of the hottest topics in academy as well as industry. On the contrary, it seems that not so much respondents are interested in Computer Networks (10%) and NeuroScience (8%). And the rest of the research field we selected for this question have quite similar respondent share (about 20%).

Q3: Which of the following research fields have you gotten into before?

In this question, respondents were asked to select one or more research field that they have gotten into before. Figure 3.3 show responses of this question.

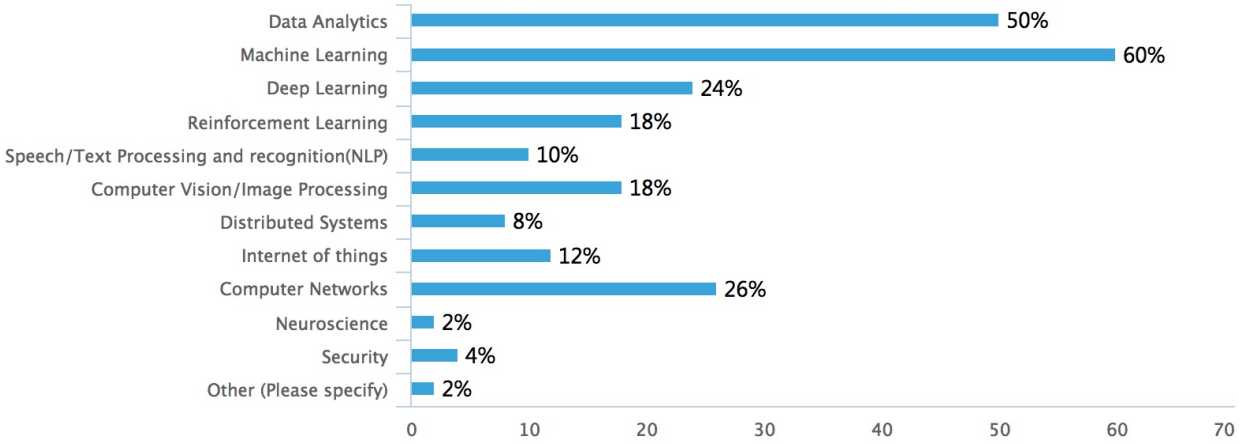

Figure 3.3 Research fields our respondents have gotten into before

Judging from the raw data we gained from this question, there does not share many differences in responses compared to the previous one (Q2), which is pretty reasonable. Machine Learning(60%) and Data Analytics(50%) are still the two research fields that most of our respondents have stepped into before. But on the contrary, Deep Learning(24%) and Security(4%) do not share as much respondent share as they do in Q2. A possible explanation of this phenomenon is that Security field is a research field that is relatively harder than other research fields to step in, for it may require many pre-knowledge in this

field. Although many respondents are interested in this field, it may be hard for them to dig into Security field because of the lack of related pre-knowledge. Comparing to Machine Learning and Data Analytics, Deep Learning is more difficult to learn. Since most of our respondents are first year graduate students, they may not select Deep Learning in their first year. And the phenomenon that Computer Networks has much more respondent shares than it does in Q2 can also be explained by the fact that most of our respondents will select Computer Network courses in their first year since it is not so hard to learn.

Q4. Do you think it is a troublesome and time-consuming thing to search for the reference materials during your projects?

In this question, respondents were asked whether they think searching for reference materials is a troublesome and time-consuming thing. Figure 3.4 show responses of this question.

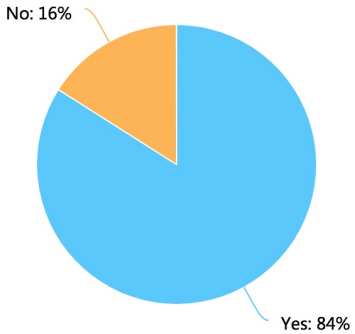

Figure 3.4 whether searching for reference materials is troublesome and time-consuming

From responses, 84% of our respondents find it is really time-consuming to search for related reference materials. Only 16% of our respondents think it is not so troublesome. It is obvious that there do exist a need to build such a reference materials recommendation system.

Q5: How long will it take you to find reference materials you need during your project (in average)?

In this question, respondents were asked about the average they spend on finding reference materials during their project. Figure 3.5 show responses of this question.

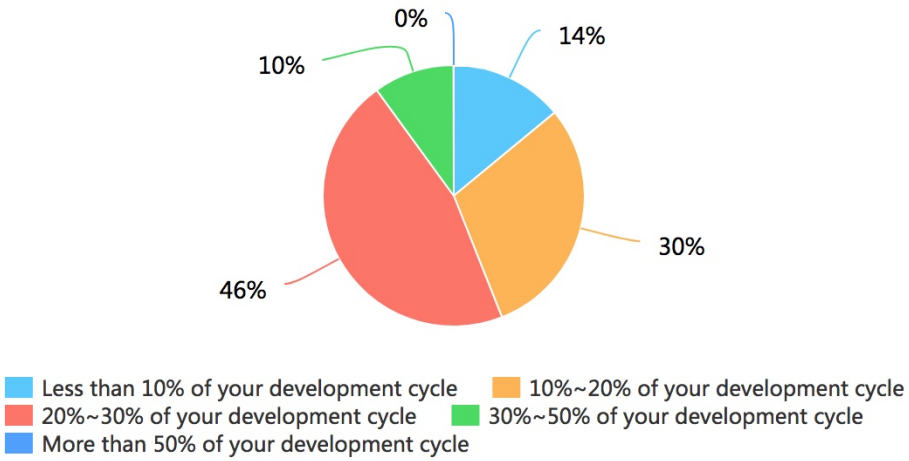

Figure 3.5 Average time spent on searching reference materials

From our responses, most of our respondents (76%) will spend 10%~30% of the whole development cycle on searching for reference materials, which is pretty long. We believe our recommendation will sharply decrease the time spent on searching reference materials.

Q6: Which of the following reference materials do you think help you most in your projects?
In this question, respondents were asked to point out the reference material that may help them most in their projects. Figure 3.6 show responses of this question.

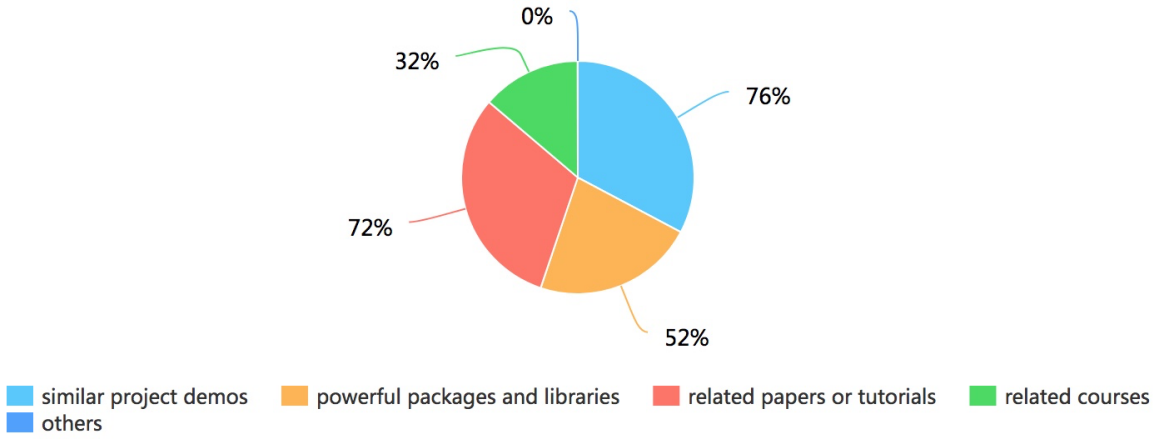

Figure 3.6 The reference material that may help respondents most in their projects

From responses, related papers or tutorials(72%) and similar project demos(76%) are pretty welcomed by our respondents. And also 52% of our respondents think powerful packages and libraries may help them most in their projects. 32% of our respondents like to be recommended for some related courses for their projects.

3.4 Conclusion

According to responses of our survey, we can find that there is a need to build such a reference materials recommendation system, since for the most of our respondents, they may spend 10% ~ 30% of their development circle on searching for related reference materials, which is really time-consuming. And the top 3 research fields that our respondents are most interested in are Machine Learning, Data Analytics & Deep Learning. Also, they said that related tutorials and similar project demos will do a great help to their project. As a result, our recommendation system will emphasize on these aspects.

4. Data Collection

(how we obtained a reasonable range of sample data)

4.1 Dynamic Data

The dynamic data of our project mainly refer to the public project demos that requested from the official GitHub API using the *Search* function. When users input their research field and programming language in the front-end of our recommendation system, then the input will be sent our back-end. After that, the search request will be executed in the back-end. Specifically, the inputs of the user contain two required components and an optional one. The required components are the research field their projects are related to and the programming language they will use, and each of them have several options to be selected by users. The optional component is a text input box, where users can add some extra information to specify the searching result of the given research field, or users can just use it as a single input by keeping the research field input empty. More detail description is in the 5.2 Front-end part.

4.1.1 Parameters

The parameters in the *Search* method contains three components. *q*, *sort*, and *order*:

- *q* is the required parameter, it defines the search keywords, as well as any qualifiers.
- *sort* and *order* are optional parameters, they define the sort field and the sort order of the requested data.

By modifying the *q* parameter, we can specify repositories we request. We add extra parameters *topic* to filter repositories based on the specified research field and add *language* to search repositories based on the programming language they are written in. In this case, the sample request looks like:

*https://api.github.com/search/repositories?q=topic:**ML**+language:**Python**&page=1&per_page=10*

With the above request, we are able to request the repositories relevant to ML (Machine learning) and written in Python, we hope the returned 10 samples are all contained in one page. We have not set parameters of *sort* and *order*, so these two values will be kept in default, which means the returned demos will be sorted in descent order with the best match. The response is stored in a JSON file which contains all relevant information about the GitHub repositories we have searched for. After filtering all useless variables in the JSON file, we are able to save the two variables *name* and *link* in a CSV file, which looks like:

1	ujjwalkarn/Machine-Learning-Tutorials	https://github.com/ujjwalkarn/Machine-Learning-Tutorials
2	uber/horovod	https://github.com/uber/horovod
3	allmachinelearning/MachineLearning	https://github.com/allmachinelearning/MachineLearning
4	lutzroeder/Netron	https://github.com/lutzroeder/Netron
5	ssusnic/Machine-Learning-Flappy-Bird	https://github.com/ssusnic/Machine-Learning-Flappy-Bird
6	wxyyxc1992/AIDL-Series	https://github.com/wxyyxc1992/AIDL-Series
7	MentatInnovations/datastream.io	https://github.com/MentatInnovations/datastream.io
8	rtavenar/tslearn	https://github.com/rtavenar/tslearn
9	atgambardella/pytorch-es	https://github.com/atgambardella/pytorch-es
10	dasguptar/treelstm.pytorch	https://github.com/dasguptar/treelstm.pytorch

Figure 4.1: Topic: ML + Language: Python

From the figure 4.1, we can find that the returned file contains 10 items with each has two variables, the demo provider's ID/the demo's name (*name*) and the link to that demo (*link*). For example: for index 1, this repository is the most popular one in this search. The name of the provider of this repository is "ujjwalkarn" and the name of this project is "Machine-Learning-Tutorial", which is highly relevant to the research field we have inputted in the request. And the second column provides a URL linking to this repository in GitHub.

4.2 Static Data

The static data of our project mainly consists of three parts: course name, course description, and programming language tutorials & developer community.

4.2.1 Course Name

For this part, we create a course list that contains all courses (related to the research field we have selected) offered by Computer Science department and Electrical Engineering department these two semesters (2018 Spring & 2018 Fall). According to the result we get from our market investigation, we summarize the offered courses in the following 11 fields: Machine Learning, Data Analytics, Deep Learning,

Computer Vision, Reinforcement Learning, Security, Distributed Systems, NLP, Internet of things, Computer Networks, Neuroscience (Ranked in popularity rank gained from figure 3.2). Each of the fields may contain 2~8 courses separately, which can be found in Appendix A at the end of the paper. And all these course offering information is extracted from Vergil [8], the CS department website [9], and the EE department website [10].

4.2.2 Course Description

For each course contained in the course list as is mentioned in 4.2.1, we extract the brief course description from its course website or Vergil. For example:

ELEN4903: Topics in Electrical and Computer Engineering. TPC: Machine Learning

The course description: "This course provides an introduction to supervised and unsupervised techniques for machine learning. We will cover both probabilistic and non-probabilistic approaches to machine learning. Focus will be on classification and regression models, clustering methods, matrix factorization and sequential models. Methods covered in class include linear and logistic regression, support vector machines, boosting, K- means clustering, mixture models, expectation-maximization algorithm, hidden Markov models, among others. We will cover algorithmic techniques for optimization, such as gradient and coordinate descent methods, as the need arises. This class is part of the Topics in Electrical & Computer Engineering series." [11]

Since the whole course description document is too long to be shown in this report, we have uploaded it to our GitHub.

4.2.3 Tutorials and Developer Community

For the programming languages part, we did a survey in our midterm paper, which showed a programming language popularity rank:

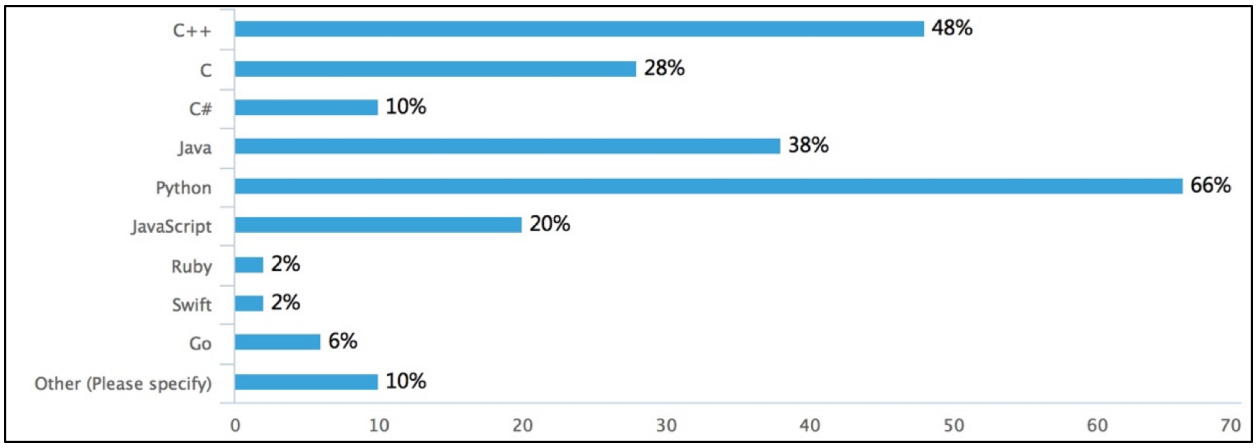

Figure 4.2 A rough programming language popularity rank

You can easily find in figure 4.2 that Python is the most popular language. C++ and Java also share a large proportion of the respondent with 48% and 38% separately. We build our recommendation system based on this popularity rank. We extract the tutorial of each programming language from each official website. And we also find some popular developer communities for each programming language, which can be found in Appendix B.

5. System design

5.1 Back-end

5.1.1 Flask and jQuery

For the back-end part, we implement Flask using a Python library called flask, which is a light-weight back-end architecture. It can implement different functionalities when it receives different feedbacks from the front-end of our recommendation system. And for the front-end part, we use Ajax in jQuery to refresh certain section of the web page in each query process rather than refresh the whole page. By implementing this architecture, each search request from our web application can execute and get responses immediately, without any delays.

5.1.2 API implement

We implement the API request using a Python library called urllib2, which could send HTTP request and communicate with the GitHub API. The API request function is embedded in the back-end and will only be executed when the user clicks on the search button in the front-end of our recommendation system. During this process, the research field, the programming language and extra information provided by users will be sent to the back-end after users' click action. After that, the back-end will execute the request function and then get responses from the GitHub API. After filtering the response, a JSON type vector, as is described in the 4.1 Dynamic Data, will be returned to the front-end. All the returned data will be displayed in the Demo section of the web page, which will be described in details in the 5.2 Front-end part.

5.2 Front-end

The front-end of our recommendation system is built based on a toolkit called Bootstrap [5], coding in JavaScript and HTML. And the static database of our recommendation system is built in the front-end (JavaScript), which mainly consists of three parts: the course information, programming language tutorials and developer communities.

5.2.1 UI design

The UI design of our web application is shown in figure 5.1.

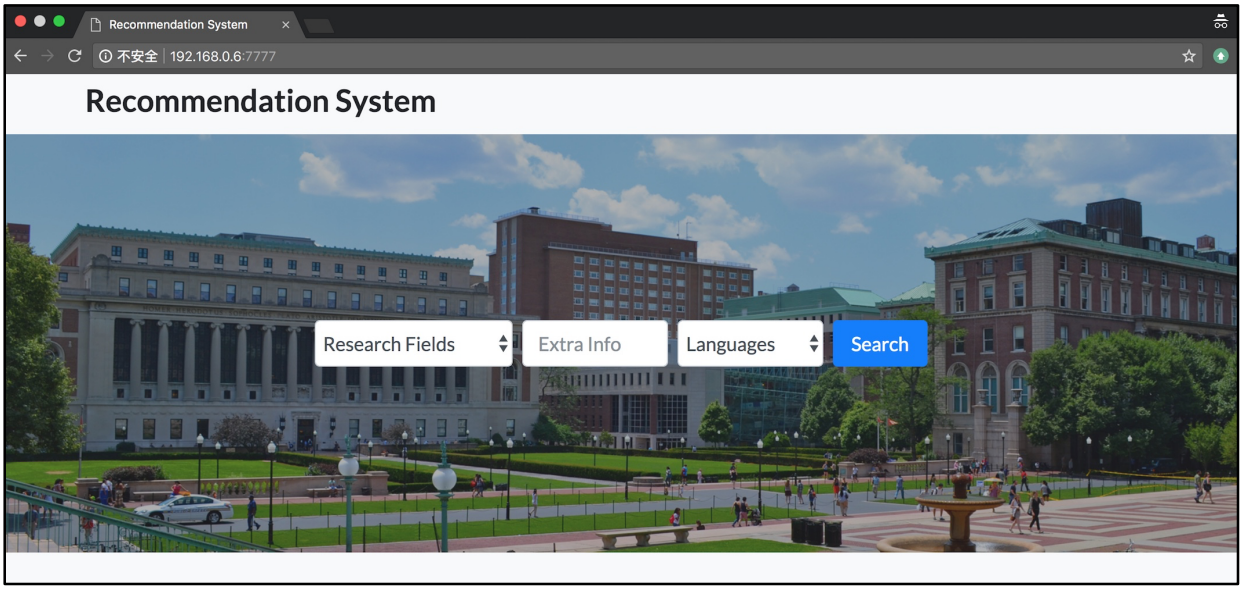

Figure 5.1 UI of our web application

The left select box is where users can select the research field of their projects. The research field options that user can select is shown in figure 5.2.

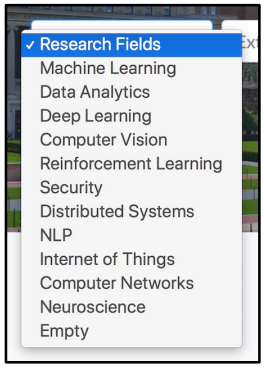

Figure 5.2 Research field options

We rank the research field options in our system based on the research field popularity rank we get from our survey. The most popular research field will be placed at the top of the option list.

The input box in the middle is where users can input some detailed information that they want to search for their projects. For example, users can search in the following format shown in figure 5.3. By doing this, our system will search for demos that related to the research field: “machine learning” as well as the keyword: “data”. Users can also only input the detailed information without selecting the research field if they want to search for a specific research field that is not included in research field options.

Figure 5.3 Input sample of the input box

The left select box is where users can select the programming language of their projects. The programming language options that user can select is shown in figure 5.4.

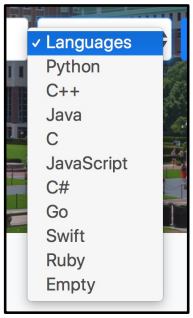

Figure 5.4 Programming Language options

We rank the research field options in our system based on the programming language popularity rank we get from the survey in our midterm paper. The most popular programming language will be placed at the top of the option list.

The blue search button on the right will trigger the search function of our recommendation system. By clicking the search button, all reference materials that are related to the given research field, the given extra information, and the given programming language will be shown on the web page. Taking the research field: Machine Learning and the programming language: Python as an example, the search result is shown in figure 5.5.

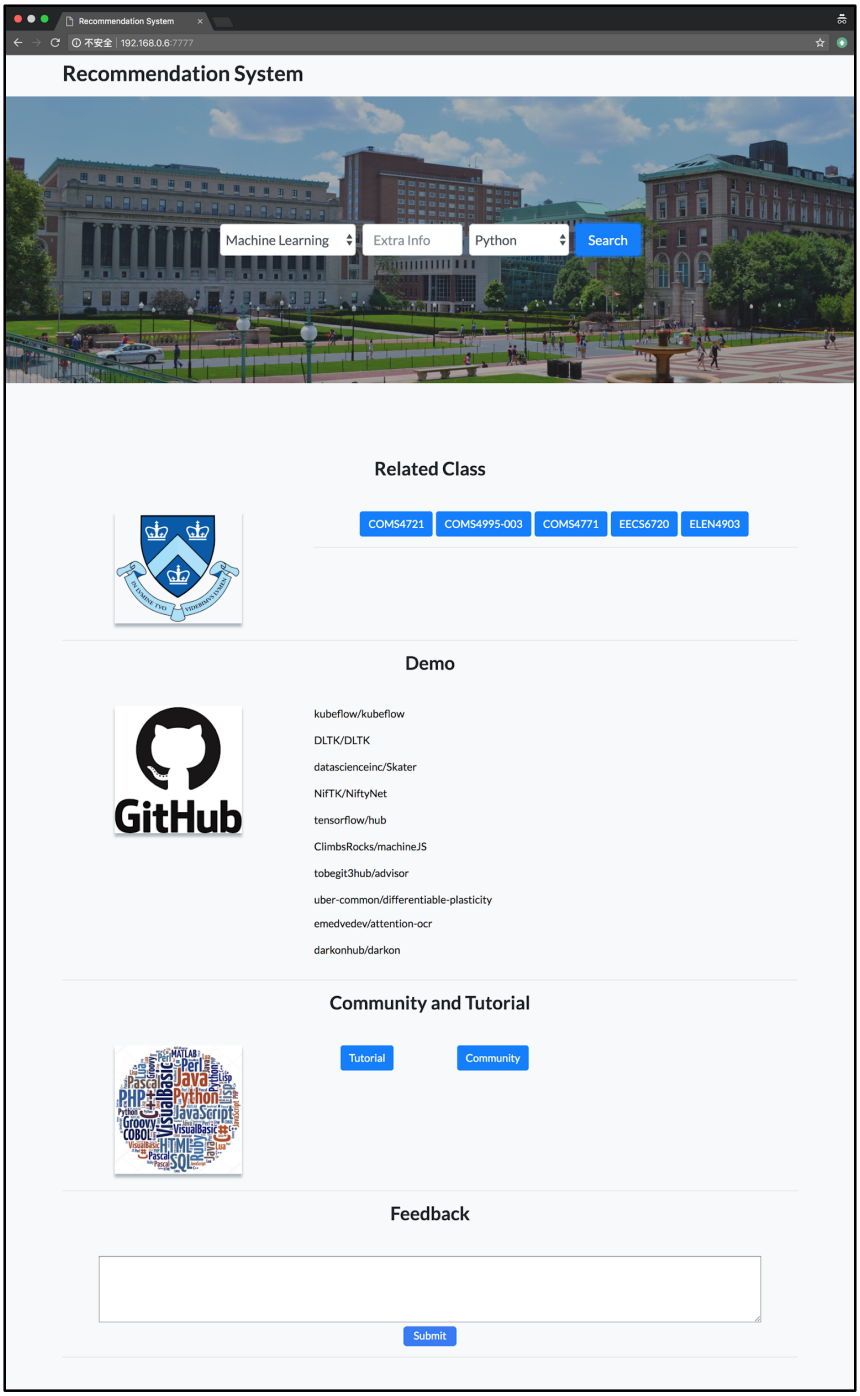

Figure 5.5 the search result of Machine Learning + Python

5.2.2 Related course information display

The related course information shown in this part is only sensitive to the research field users select. It will show all the related courses offered by CS/EE department these two semesters (2018 Spring & 2018 Fall) which are related to the given research field.

Again, take the research field: Machine Learning as an example:

As is shown in figure 5.6, the blue buttons stand for all machine learning related courses offered by CS/EE department these two semesters: COMS4721, COMS4995-003, COMS4771, EECS6720, ELEN4903. By clicking each button, the course name and the course description of the selected button (course) will be shown below these buttons. As is shown in figure 5.6, the course ELEN4903 is selected and its course name and description are shown below. All these information can be found in Appendix A.

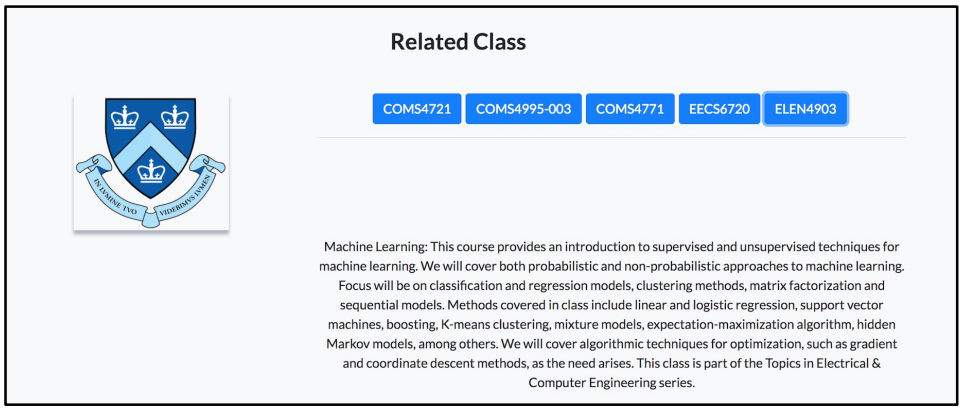

Figure 5.6 the related course information display sample

5.2.3 Demo display

The demos shown in this part are sensitive to the research field and the programming language users select as well as the extra information inputted by users. It will show the top 10 most popular open-source demos in GitHub that are related to the given research field, programming language as well as the extra keywords. If the extracted demos are less than 10, which means there are no more than 10 related demos in GitHub, our system will show "No more demos" there and link it to the GitHub official website, shown in figure 5.7. Each demo shown on the web page is linked to its GitHub repository page, which can help users find the demo easily.

Figure 5.7 No more demo sample

Again, take the research field: Machine Learning and the programming language: Python as an example: As is shown in figure 5.8, the top 10 popular related demos are shown in the middle of the page, which is shown in the format of GitHub ID/Demo Name. By clicking each demo shown on the web page, you will get access to the GitHub repository page of the selected demo.

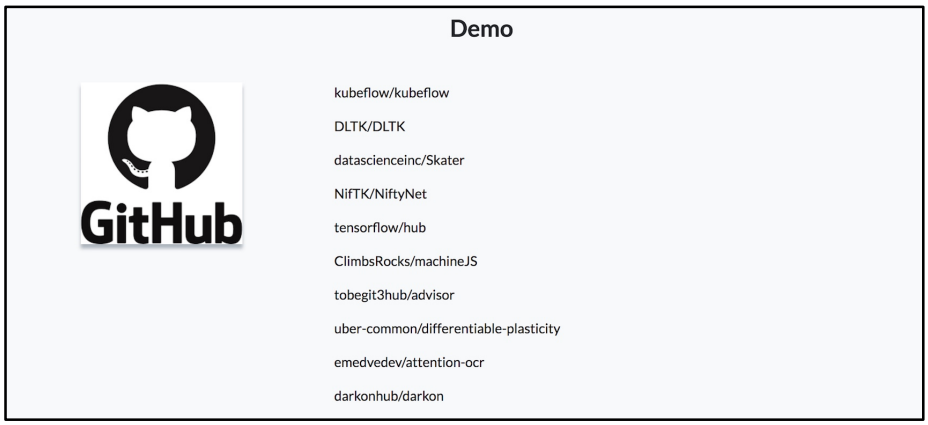

Figure 5.8 the demo display sample

5.2.4 Community and tutorial display

The community and the tutorial shown in this part are only sensitive to the programming language users select. It will show the most commonly-used tutorial and the major developer community of the selected programming language.

Again, take the programming language: Python as an example:
As is shown in figure 5.9, the left button "Tutorial" is linked to the URL of the Python tutorial: https://docs.python.org/3/tutorial/ while the right button "Community" is linked to the URL of the Python developer community: https://www.python.org/community/. All these information can be found in Appendix B.

Figure 5.9 the community and tutorial display sample

5.2.5 Feedback

In order to improve the service of our recommendation system to meet users' demand, we set a feedback input box in our web page, shown in figure 5.10, with which users can submit their comments or suggestions on our recommendation system. And all these comments or suggestions will be stored in the local database (a text file), with which we can update or add our static data as well as improve our algorithms in extracting dynamic data.

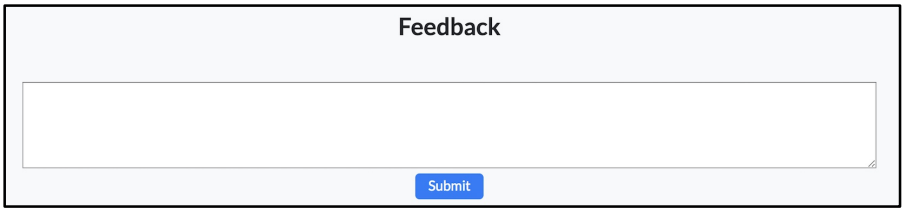

Figure 5.10 Feedback input box

5.3 Update Database

(how does our system improve over the state of the art or the state of the practice)

For the dynamic data (GitHub demos), there is no need to manually update it. Every time users click the search button in the web page, our recommendation will use GitHub API to extract the current most popular related demos from GitHub rather than use the cache data or the local static data. As a result, the GitHub demos shown in our recommendation system is update-to-date. And as long as GitHub is still in operation, our recommendation system will always extract the latest popular demos from GitHub automatically. There is no need to worry about the demos would be out of date one day.

For the static data, we need to update them manually. We will mainly update these data based on the feedback from our users by using the feedback function of our recommendation system as is mentioned in 5.2.6. If our users are willing to share their course materials such as slides, exercises, course video, final demos, etc., we can easily add these reference materials to our recommendation by updating the local database. The process of updating local database is really convenient, we only need to replace the old URL to a new one or build some new plug-in components to show the newly added reference materials.

And also, we want to answer the two questions that Prof. Kaiser asked during our demo presentation in this part.

Q1: How can we update the course information part after we graduate?
All the course information (the course name & the course description) is mainly extracted from public resources (CS/EE department Official Website). In other words, we can get access to these public resources without logining in our UNIs. As a result, even when we graduate from Columbia University, we can still update these course information at any time. And we can also update these course information based on the feedback we collect from our users as is mentioned in 5.2.6.

Q2: How to deal with the issue that one course is only offered in one semester but not offered in the next one?
The reason that we set the related course section is that we want to provide our users with a possible way to find course materials from previous courses that are related to the users' projects, with which users can search on Google with the keyword "Course Name + Columbia University" and get access to the course materials provided in its course website. As a result, it does not really matter whether the course is offered or not at Columbia University now, we only need to let users know that there exist some related courses before and there may have some useful course materials on the course websites that may be helpful to their projects. For the future work, maybe we can link each course (the blue button on the web page) to its course website. By doing this, it will be more convenient for users to find these course website and get useful course materials.

6. System evaluation

6.1 User Distribution

To evaluate the performance of our recommendation system, we invited some students majoring in Electrical Engineering or Computer Science at Columbia University to use our recommendation system and makes some comments or suggestions on it. The invitation was distributed to some graduate students we know and they were also asked to distribute this survey to their friends at Columbia University is possible. But unfortunately, we did not collect too much feedback since most of them are busy with their finals.

6.2 How to access the web application

(how members of our target community will find out about our system and access/utilize it)
We distribute our web application to our friend and classmates and collect their feedbacks to do the evaluation of our recommendation system. In the following paragraphs, we will describe how our users can get access to our web application and use the reference materials recommendation system.

6.2.1 Local Access

We can launch our server (the back-end) on our laptop in the terminal by simply running the Flask instance app without adding any extra parameters. After doing this, we can access the web application locally by typing the default IP address: 127.0.0.1, and default port number: 5000. In this case, users can use the web application from our laptop and make comment in the feedback section of our recommendation system.

6.2.2 WLAN

By adding extra parameters in the command app.run(), we can modify the IP address of the host and the certain port number. We can launch our server by using the IP address of our laptop and specifying a port number. People can get access to our recommendation system from their personal laptops or mobile devices as long as they connect to the same WIFI network as us do and shut down the firewall of their devices.

6.2.3 Cloud Service (in future)

For the future work, we also consider to launch our server in some commercial cloud services and make our recommendation system be accessible to all Columbia graduate students, if the we could get really good and positive feedbacks from users during the trial operation. There should be no technology difficulty in launching our recommendation system in cloud, given so many mature cloud services nowadays.

6.3 Evaluation

(How we test our system)

6.3.1 How to evaluate

(the forums we used to recruit your participants)

We mainly use the WLAN to launch our recommendation system. Users can get access to our system by visiting the IP address and port number we have set from the browser of their own devices. And we also taught them some tips to help them be familiar with our system in a short time. Our system is not hard for users to start with and most users can find the reference materials they want easily. After they experiencing our recommendation system, they are asked to make comments in the feedback input boxes before they existed our system.

6.3.2 Challenge

(issues that arose during or as a consequence of their participation)

One challenge of the evaluation is how to let users get access to the recommendation system we have launched in the WLAN. When we try to use the WLAN to launch our recommendation system, we found users cannot get access to our recommendation system with their own devices and they can only login in the recommendation system from our laptop (the host), which makes us really confused. We tried to change the IP address and the port number but it did not work at all. For a long time debugging and searching, we found it may have something to do with the firewall of our host PC. After shutting down the firework it worked well.

Another challenge we met is how to widely spread our recommendation system. Since at present we only launch it in the WLAN, which means only a small amounts of user can get access to our recommendation system. To solve this problem, for the future work we plan to launch our recommendation system in some commercial cloud services as is mentioned in 6.2.3.

6.3.3 Result

The participants all major in Electrical Engineering or Computer Science at Columbia University and they have taken at least one class of all the research fields we provided in our recommendation system in the previous semesters at Columbia University. **(what participants demographics we achieved)** In this case, their feedbacks are valuable for the update of our system.

We hope our users could evaluate the system from two aspects, one is to evaluate the user experience of our recommendation system, the other one is to give us some suggestions in the possible improvements of our recommendation system, especially in the aspect of the reference materials we have provided in the system. Each participant is asked to evaluate in at least one of these two aspects. **(what we tasked our participants to do)** Here are some feedback we get from our users:

- Student A: "Amazing recommendation system! Simple and convenient to find related class, demos and tutorials for specific research field. It is common that many related class will be distributed in different department which is difficult to search, this systems help us a lot to find classes quickly. It will be better if we can have more detailed information about the class. Also, maybe you can try to offer your recommendations about class arrangement in one specific field, such as if I want to study in the field of machine learning, in what order should I take the classes you recommended."

- Student B: "The website is really good looking and useful. I can easily find the information that I want to know. But sometimes the found information is limited, hope there can be further improvement."

- Student C: "I found the research aggregation tool extremely useful since it also brings together sources from GitHub specifically geared towards classes being taken at Columbia. This could be used in the future by students across CS and EE in Columbia to make the research process easier."

- Student D: "This webpage is wonderful!! I can easily have access to various courses, GitHub links and tutorials with this tool. It will definitely benefit me a lot in improving my data analysis and python programming skills. I will recommend this system to my classmates!!"

According to the feedback from our users, our recommendation system should be a useful tool that could be used in graduate students' course projects and it seems that our recommendation system can meet their demands on searching for the reference materials for their projects. However, there are still some possible improvements of our system, such as the class arrangement section mentioned by Student A, the more detailed material information mentioned by Student B, etc. We will update and improve our recommendation system based on all these creative and positive feedbacks and suggestions collected from our users in future.

7. Conclusion

7.1 Work Distribution

Chi Zhang, cz2465:

- Mainly responsible for the back-end part of the recommendation system.

- Help in building the front-end part of the recommendation system..
- Collect the dynamic data of GitHub.

Jiayi Li, jl4924:

- Mainly responsible for conducting the market investigation (the survey).
- Collect and clean the static data: the related course list, the course description, programming language tutorials, and developer communities.
- Mainly responsible for the front-end part of the recommendation system.

7.2 Achievement

(What we learned?)

Chi Zhang(cz2465):

In this project, I have much more understands of the software engineering, which should cover a lot of aspects rather than only consider programming. I also practice my programming skills in implementing back-end and front-end architectures, as I have no experience working with flask, Ajax and jQuery before. I have spent a lot of time on learning these tools by searching tutorials, domes and official documents on the Internet, which is really a tough time. I have to admitted that the demo is the most useful public resource in searching public materials for our projects, that is also why we want to build such a system and set demos in the first section of our web. By collecting public resources from the Internet, we really find a lot of useful resources that we have not considered in our other course projects before. As a result, I get improvement in both programming skills and knowledge about public resources in this course project.

Jiayi Li(jl4924)

Firstly of all, of course, I form a comprehensive understanding of how to build a web application base on Flask and Bootstrap. To be honest, I do not have much experience in building a web application and build an iOS application should be much easier for me. But after reading the official document of Flask and Bootstrap, I begin to have some idea on how to build them. And also, building the front-end using JavaScript and HTML is also a great challenge for me, since they are totally new to me. But fortunately, I can easily find a tutorial on JavaScript from the programming language tutorials I collected for this project, which truly improve my productivity. I truly understand how reference materials may improve one's developing productivity. And after building this web application, I form a basic programming skill in JavaScript and HTML, which I believe will help me a lot in my future career.

During the design of the front-end UI, I also learned how to increase the user experience of a web application. By ranking the research field/ programming language options in our system based on the popularity rank get from our survey, users can usually find the option they want to select more quickly without the need to traverse the whole option list. The concise UI design will also help users to start with our application in a short time without any previous study. These experience will definitely help me a lot when I build other applications in future.

And also, based on the experience in conducting the survey in my midterm paper, I am able to quickly design the questionnaire this time and also overcome some drawbacks of my previous questionnaire in the midterm. I form a solid foundation in how to conduct a survey from this project. I really appreciate it that this course gives me such a good opportunity to practice all these skills.

At the end of this paper, we want to say thanks again to those who have helped us in our final projects!

8. Reference & Appendix

Reference:

[1] Flask official document. Official Website. Retrieved from: http://flask.pocoo.org/

[2] GitHub API v3. Official Website. Retrieved from: https://developer.github.com/v3/search/

[3] Ajax wikipedia: Retrieved from: https://en.wikipedia.org/wiki/Ajax_(programming)

[4] jQuery official tutorial: Retrieved from: http://jquery.com/

[5] Bootstrap. Official Website. Retrieved from: https://getbootstrap.com/

[6] Python library urllib2. Official Website. Retrieved from: https://docs.python.org/2/library/urllib2.html

[7] wjx survey creator. Official Website. 2018. Retrieved from: https://www.wjx.cn/

[8] Vergil. Official Website. Retrieved from: https://vergil.registrar.columbia.edu/

[9] CS department website. Official Website. Retrieved from: https://www.cs.columbia.edu/education/courses/

[10] EE department website. Official Website. Retrieved from: http://www.ee.columbia.edu/course-offerings

[11] ELEN4903: Topics in Electrical and Computer Engineering. TPC: Machine Learning. Course Website. Retrieved from: http://www.columbia.edu/~jwp2128/Teaching/E4903/E4903Spring2016.html

Appendix A: the class list of the research fields

(All these course offering information is extracted from Vergil [8], the CS department website [9], and the EE department website [10].)

A.1 Data Analytics

EECS6895: *Topics in Information Processing. TPC: Advanced Big Data Analytics*

EECS6690: *Topics in Data-driven Analysis and Computation. TPC: Statistical Learning in Biological & Information Systems*

EECS6893: *Topics in Information Processing: Big Data Analytics*

COMS4721: *Machine Learning for Data Science*

COMS4995-004: *Causal Inference for Data Science*

COMS6998-010: *Cloud Computing and Big Data*

COMS4121: *Computer System for Data Science*

A.2 Machine Learning

COMS4721: *Machine Learning for Data Science*

COMS4995-003: *Applied Machine Learning*

COMS4771: *Machine Learning*

EECS6720: *Bayesian models for machine learning*

ELEN4903: *Topics in Electrical and Computer Engineering. TPC: Machine Learning*

A.3 Deep Learning

ECBM6040: *Neural Networks & Deep Learning*

ECBM6070: *Topics in Neuroscience and Deep Learning. TPC: Fruit Fly Brain as Neuro Information Processor*

EECS6894: *Topics in Information Processing. TPC: Deep Learning for Computer Vision, Speech and Language*

COMS6998-003: *Advanced Topics in Deep Learning*

COMS4995-002: *Deep Learning*

A.4 Reinforcement Learning

COMS6998-006: *Bandits Reinforcement Learning*

ELEN6885: *Topics in Signal Processing. TPC: Reinforcement Learning*

A.5 Speech/Text Processing and Recognition, (NLP)

EECS6894: *Topics in Information Processing. TPC: Deep Learning for Computer Vision, Speech and Language*

COMS6998-007: *Fund Speech Recognition*

COMS6998-012: *NLP in Context Computational M*

A.6 Computer Vision/ Image Processing

EECS6894: *Topics in Information Processing. TPC: Deep Learning for Computer Vision, Speech and Language*

ELEN4830: *Digital Image Processing*

COMS4731: *Computer Vision*

A.7 Distributed Systems

COMS6998-003: *Advanced Distributed System*

COMS4113-001: *Fundamentals of Large-Scale Distributed Systems*

A.8 Internet of Things

EECS4764: *Internet of Things - Intelligent and Connected Systems*

EECS6765: *Internet of Things: Systems and Physical Data Analytics*

A.9 Computer Networks

CSEE4119: *Computer Networks*

COMS4180: *Network Security*

CSEE4140: *Networking Lab*

ELEN6771: *Topics in Networking. TPC: 5G Programmable Networks*

ELEN6772: *Topics in Networking. TPC: Network Algorithms and Learning*

ELEN6770: *Topics in Networking. TPC: Next Generation Networks*

ELEN6775: *Topics in Networking. TPC: Advanced Networks: A Systems Approach*

ELEN6776: *Topics in Networking. TPC: Content Distribution Networks*

A.10 Neuroscience

EEBM6091: *Topics in Computational Neuroscience and Neuro engineering. TPC: Devices and Analysis for Neural Circuits*

BMEB4020: *Computational Neuroscience: Circuits in the Brain*

BMEE4030: *Neural Control Engineering*

ECBM4040: *Neural Networks and Deep Learning*

ECBM4060: *Introduction Genomic Information Science & Technology*

ECBM4090: *Brain Computer Interfaces Lab*

COMS6998-007: *Introduction Brain Computer Interact*

A.11 security

COMS6998-009: *Security and Robustness of ML Systems*

COMS 4180: *Network Security*

COMS 4187: *Security Architecture and Engineering*

Appendix B: Tutorials and Developer Community for each programming language

(All these information is extracted from public sources.)

B.1 C++

Tutorials: http://www.cplusplus.com/doc/tutorial/

Developer Community: http://www.cplusplus.com/forum/

B.2 Java

Tutorials：https://docs.oracle.com/javase/tutorial/

Developer Community: http://www.oracle.com/technetwork/java/community/index.html

B.3 Python

Tutorials: https://docs.python.org/3/tutorial/

Developer Community: https://www.python.org/community/

B.4 C

Tutorials: https://www.cprogramming.com/tutorial/c-tutorial.html

Developer Community: http://www.dreamincode.net/forums/forum/15-c-and-c/

B.5 Go

Tutorials: https://golang.org/doc/

Developer Community: https://forum.golangbridge.org/

B.6 Swift

Tutorials: https://swift.org/documentation/#the-swift-programming-language

Developer Community: https://swift.org/community/#forums

B.7 Ruby
Tutorials: https://www.ruby-lang.org/en/documentation/
Developer Community: https://www.ruby-lang.org/en/community/
B.8 JavaScript
Tutorials: https://www.w3schools.com/js/default.asp
Developer Community: https://www.codingforums.com/javascript-programming/
B.9 C#
Tutorials: https://docs.microsoft.com/en-us/dotnet/csharp/
Developer Community: http://www.csharpforums.net/

Appendix C: Course Description

https://github.com/Debug1995/6156_FinalProject/blob/master/Course%20Description.pdf

Appendix D: The raw data of my survey

https://www.wjx.cn/report/22439093.aspx

(How to access to our deliverables?)

Appendix E: GitHub link of our project

https://github.com/Debug1995/6156_FinalProject