ConGrad

Group members: Jieyu Ren, Qi Cheng, Zijun Liu, Dengling Zhang, Hao Yang, Tao Huang

1. Background

ConGrad is a web application dedicated to tracking and providing advice surrounding the graduate school application process. It has two major functions: 1. It allows users to search admission information from previous years and use our sort and time filters to refine the results. 2. It generates a recommended program list based on users' application information. In the normal process of applying for graduate schools, there are too many programs/schools that are listed on various websites. Students would need a lot of time for the information gathering process. With our web app, students can find all the information they need in our system and get recommendations for the school/programs that they are probably interested in and have a larger chance of getting accepted given their academic status.

2. Technical Specifications

2.1 Login and Register Page

The login and register page is where users can enter their account number and password to access what they have stored in the database, such as the schools or programs that they would like to view later. After a user entered the account number with the password and clicked the login button below, he/she should enter the front page (2.2 section below). If the password and account number do not match in our database, there should be a pop-up window showing users the information they entered is not correct.

The login and register page should also have a register button for users to register as a member of our system with their desired account number and password. When users click on the register button, a pop-up window should show up for users to enter the account number and password. If the account number already exists, the page should ask users to pick another account number.

See Figure 1.1 for the proposed scratch for the login page.

2.2 Front Page

The front page should have three components, a recommendation section (see section 2.3 below) to recommend programs, a user information-showing section (see section 2.4 below) to show what the user had inputted into our database or edited in the *user info edition section* button, and a search box(section 2.5 below) to manually search for programs. There should be two navigation buttons for users to enter their own user page (2.7 section below), and to edit his personal information such as GPA or interested major, etc.

See Figure 1.2 for the proposed scratch for the front page.

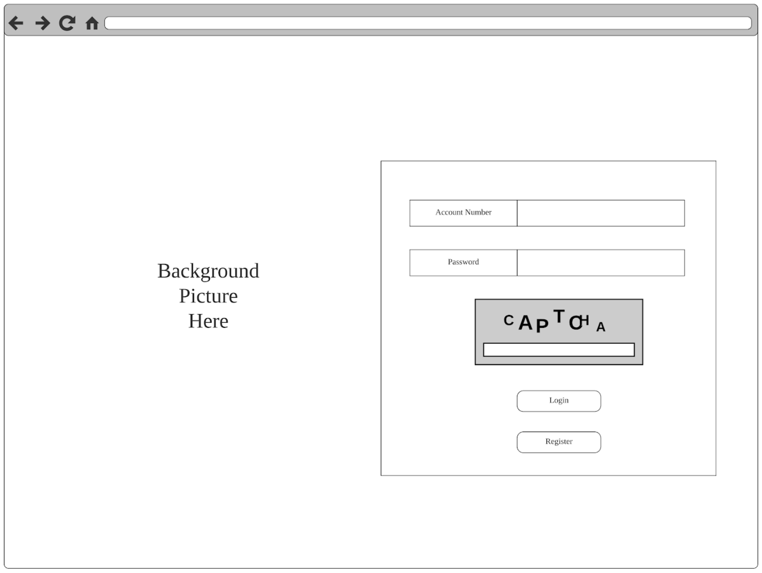


Figure 1.1

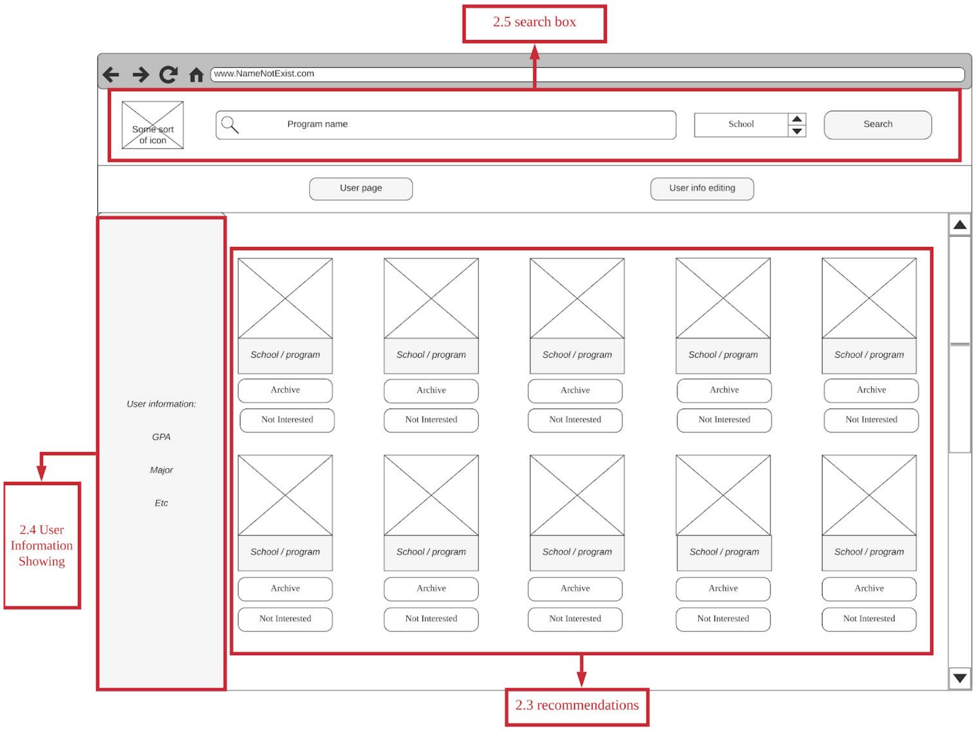


Figure 1.2

2.3 Recommendation

The recommendation section is one of our core functionality for the application. The recommendation section should recommend programs for users according to their GPA, the rank of their current college, GRE score, current major in college, desired major for graduate school and desired location of graduate school to recommend master programs. The level of recommendation for a program should be based on the possibility of acceptance, the major ranking of a program and user preference.

Ten results of the above calculation, including the school name, program name, and link to the program website, should be shown on the front page in the recommendation section. When users see a recommended program, they can choose to archive the program information shown by clicking on the *archive* button below each recommended program or to say they are not interested in the program by clicking on the *not interested* button. If the user has not entered his information, the recommendation section should show Case Western Reserve University and other nine top schools such as MIT, Stanford, Cal Tech, etc.

2.4 User Information Presenting

This section should show all the information that a user inputted in our application so that the user can easily decide whether they want to change his/her information and preference for the programs. That information should be shown on the left of the front page (section 2.2 above).

2.5 Search Box and Search Result Page

There should be a search box at the top of the front page (section 2.2 above) for users to search for specific programs or schools manually. The search box should be composed of two elements, a type box for users to enter the name of programs, such as MS computer science or professional degree computer engineering, and a select box to let users choose a school that exists in our database. When users finished filling in the two boxes and clicked on the search button on the right, a new web page should show up to show the search results.

The search result page is the same as the front page except the recommendation section replaced with a list of search results showing the basic information about the program searched by the users. If no results are found, the list of results should show other similar programs.

Below is the proposed scratch for the search result page

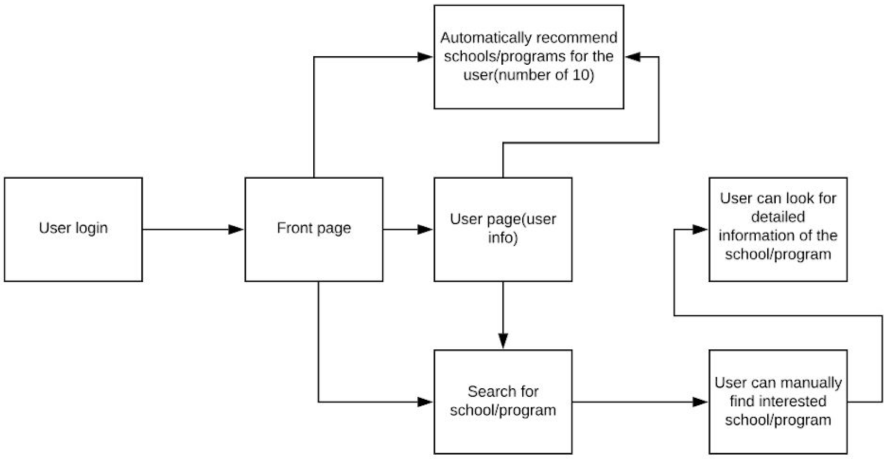
2.6 User Page

The user page should consist of four parts, the current application status, the status of recommendation letters, a section showing all the schools and program the user had archived before, and a section with the user information.The user page can be navigated to by clicking on the user page button on the top left of the front page.

2.7 Administrator Page

The administrator page should have four main functions, create, read, update and delete for administrator to access user data and school data. The administrator can login to the administrator page with specific administrator account and password. The admin page should show a table of current registered users and school programs and provide buttons for administrators to change the user information.

3. Proposed Solutions


Automatically recommend schools/programs for the user(number of 10)
User login
Front page
User page(user info)
User can look for detailed information of the school/program
Search for school/program
User can manually find interested school/program

- Backend
  - Java on Intellij
- SQL server
  - PostgreSQL
- Frontend
  - JSP
  - HTML5
  - CSS
- Test
  - JUnit

4. Team background

Jieyu has a demonstrated background in Java development and database management. He is most comfortable with backend development and database connection. Some similar projects he has worked on include a schedule generating system written in Java, several machine learning algorithms written in Python and an E-Sports web application written in Python and SQL.

Zijun has worked on a project involving database connection and generating optimal plans as a backend developer. She is familiar with coding in IntelliJ and has taken a data science course, which allows her to handle database tasks. Zijun has also created simple web pages using Django before, so it will be easier for her to make use of other web application frameworks.

Tao has experience in C# and Java development and web page development. He is up on frontend development using JSP, HTML, and CSS and also backend management using Java or C#. He has worked on several projects including two game projects using C# and developing the frontend for a schedule generating system using JSP, HTML, and CSS.

Qi has experience in Java, Python, and SQL for software engineering and database management. He has worked on projects like phone price comparison APP on android, machine learning algorithm, and P2P network project. He is capable of doing backend development for our application as well as planning the overall logic for the application.

Hao has experience in Java, Python, and C#, has worked on projects like machine learning algorithm and company's income comparison APP. Also has worked on 2D,3D game project using C# as a programmer.

Denglin is a computer science senior with extensive experience in Java, C, and Python programming. He has worked on large projects such as a house stocker cloud service that assists users in keeping a house fully stocked with necessities. He is familiar with front-end building for Android applications.

5.Related work

Gradcafe: https://www.thegradcafe.com/
Offerduoduo: https://offer.1point3acres.com/

6. Work To Be Done and Management Plan

Our team will meet at least once a week to discuss progress and to decide what specific tasks to do next. Our team will try to meet the deadlines specified in the table below, but an extension of one week is acceptable.

Task	Members in charge	Deadline
Login and Registration page ready	Tao, Zijun,Hao	February 7th
Information collection	Jieyu, Qi, Hao, Denglin	February 7th
Pseudocode ready	Everyone	February 14th
Framework of main (front) page ready	Tao, Zijun, Hao	February 21st
Information entered into the database	Everyone	February 21st
Code ready	Everyone	February 28th
Recommendation page and User page's framework ready	Tao, Zijun, Hao	February 28th
Progress report 1 submission	Everyone	March 6th

Code tests	Everyone	March 13rd
Database and backend connection	Qi, Jieyu	March 16th
Backend and frontend connection	Tao, Qi, Jieyu, Zijun,Hao,Denglin	March 19nd
Functionality test	Everyone	March 25th
Progress report 2 submission	Everyone	March 27th
Web view beautification	Everyone	April 3rd
Final report draft	Everyone	April 10th
Poster Due	Everyone	April 15th
Final report ready	Everyone	April 19th
Final project submission	Everyone	April 20th