Northeastern University
Department of Electrical and Computer Engineering

EECE 4520 Software Engineering

Software Requirements Specification:
University Classifier

Authors: Felix Chan, Marissa D'Alonzo, Sumer Malhotra, Jensen Rosemund, Stockton Sheehan

February 11, 2019

1. INTRODUCTION

This section gives a scope description and overview of everything included in this SRS document. Also the purpose of this document is described and a list of abbreviations and definitions is provided.

1.1 Purpose of this Document

The purpose of this document is to give a detailed description of the requirements for the “University Classifier” software. It will illustrate the purpose and complete declaration for the development of the system. It will also explain system constraints, interface and interactions with other external applications. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

1.2 Scope of the Development Project

“University Classifier” is a project that leverages Kaggle datasets to help potential graduate students determine which college is best for them. The student can enter the university they are interested in, and the application will tell them what scores and supplementary materials they need to boost their chances of getting in. Alternatively, the student can enter information about their current profile, including GRE scores, research experience, and GPA, and the application will suggest a list of schools that they have a chance of getting in. If we were to develop this further, we would also filter out schools based off the desired major of the student.

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
User	Someone who interacts with the application
Admin/Administrator	System administrator who is given specific permission for managing and controlling the system
Stakeholder	Any person who has interaction with the system who is not a developer
Developer	Person responsible for writing the software
University	A school with graduate admissions information recorded in the database

1.4 References

[1] M. S. Acharya, “Graduate Admissions,” *RSNA Pneumonia Detection Challenge | Kaggle*, 28-Dec-2018. [Online]. Available: https://www.kaggle.com/mohansacharya/graduate-admissions. [Accessed: 12-Feb-2019].

[2] M. O'Neill, “World University Rankings,” *RSNA Pneumonia Detection Challenge | Kaggle*, 27-Sep-2016. [Online]. Available: https://www.kaggle.com/mylesoneill/world-university-rankings. [Accessed: 12-Feb-2019].

1.5 Overview of Document

The remainder of this document includes three chapters and appendices. The second one provides a general description of the system functionality and system interaction with other systems. This chapter also introduces different types of stakeholders and their interaction with the system. Furthermore, the chapter also mentions the system constraints and assumptions about the project.

The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces. Different specification techniques are used in order to specify the requirements more precisely for different audiences.

The fourth chapter deals with the any other/additional requirements that may be deemed necessary in order for the completion and smooth execution of the project.

The Appendices in the end of the document include all the results of the requirements including use cases, class diagrams and extra wire frames.

2. GENERAL DESCRIPTION

2.1 User Characteristics

The main purpose of this system is to help prospective graduate students search for and select graduate schools that are determined to fit the needs of the student. Finding the perfect graduate program can be a daunting task for someone who’s finishing up their undergraduate studies, or for someone who’s looking to go back to school while dealing with their daily lives. Therefore, this system needs to be intuitive, easy to use, and most importantly, effective. If the system does a satisfactory job of supplying users with programs that are the perfect fit, then this system has completed its purpose. The following human characteristics will be essential for the system to meet the users’ needs.

- Intuitive - The interface must be simple enough to understand without a user manual

- Efficient - Generating a report shouldn't take an excessive amount of time
- Attractive - The UI must be appealing and engaging in order to keep the users returning.
- Responsive - Navigating through the system should be smooth to prevent user irritation.

2.2 Product Perspective

This system in of itself, is a stand-alone product, however, it will be built based upon data gathered from external sources. This data will consist of values used to represent the likelihood of the user being accepted to a particular graduate school using the data about the user and the university in question. The user data will include GPA, GRE score, TOEFL score, and other parameters. The university data will be gathered once upon the system's initial release, with possible periodic updates to the data.

While this product is a stand-alone product, it can be used in tandem with other graduate program search tools. No tool is perfect, so a user may wish to use this product alongside others that account for factors not considered in our application before making a final decision. Also, this product has no external interfaces or hardware requirements. The server that will hold all data is considered to be apart of the software system, so there's no external interface requirement. And since the system can be ran from a computer, there is no hardware requirement besides the single computer.

2.3 Overview of Functional Requirements

The functional requirements of the system are as follows:

1. The user must be able to create an account.
2. The user must be able to log into their account after its creation.
3. The user must be able to update their profile.
4. The user must be able to input their information and be given a list of schools with their chance of admission as a percentage.
5. The user must be able to search for a specific school and be given a list of the scores needed to maximize their chance of admission.
6. The user must be able to cancel any action
7. The application must display an error page if the information given is incorrect

2.4 Overview of Data Requirements

This system is entirely data-driven. The user profile will contain the following information:

1. Name
2. Password
3. GRE score
4. TOEFL score
5. Research conducted
6. Letter of Recommendation
7. Statement of Purpose

8. GPA

The university profile will contain the university name, as well as data points 3 through 8 in the user profile, represented as the range of scores achieved by admitted students. This data will be collected from pre-existing Kaggle datasets.

After the data has been collected from the user, it will be fed into an algorithm that generates the report. If the user is searching for a specific school, the report will output a list of all the parameters accompanied by the range of scores needed to maximize the chance of admission. If the user is inputting their own information, it will return a list of schools with the chance of admission to each.

2.5 General Constraints, Assumptions, Dependencies, Guidelines

1. This product can only be used on a PC. However, it can be used on a Mac with 3rd party software called MonoDevelop.
2. The computer must have access to the web in order to pull data from the server.
3. The app will not hold any user data on disc.
4. All user inputs will be acknowledged within 3 seconds.
5. A system crash should not result in data loss.

2.6 User View of Product Use

The following figures show a rough outline of the look of each page of the application.

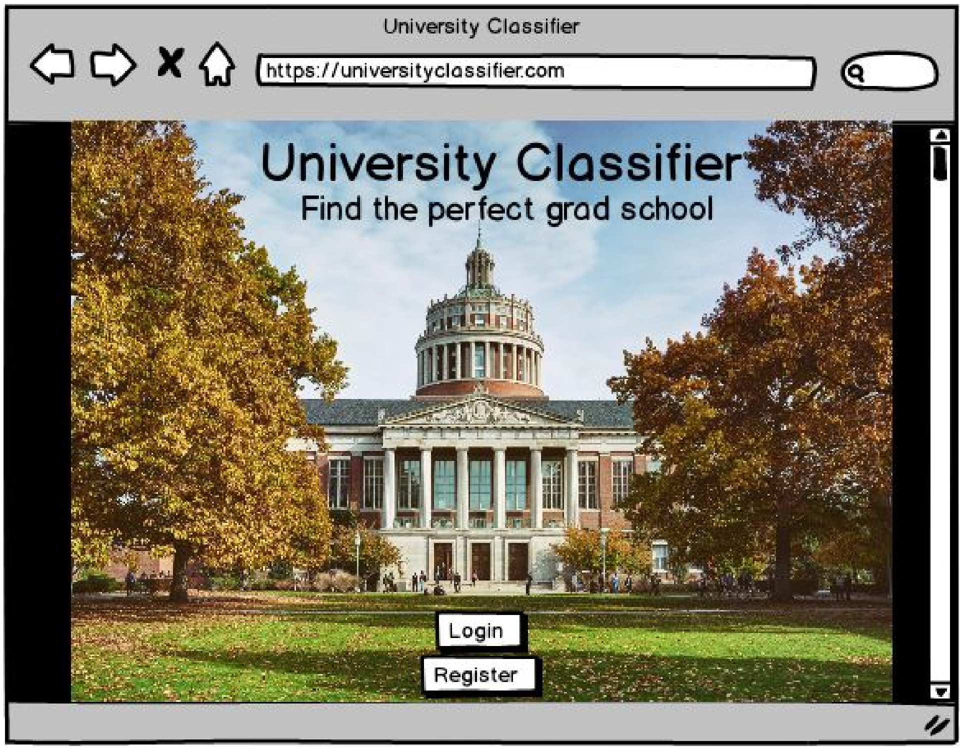

University Classifier
https://universityclassifier.com
University Classifier
Find the perfect grad school
Login
Register

Figure 1 - The home screen of the application

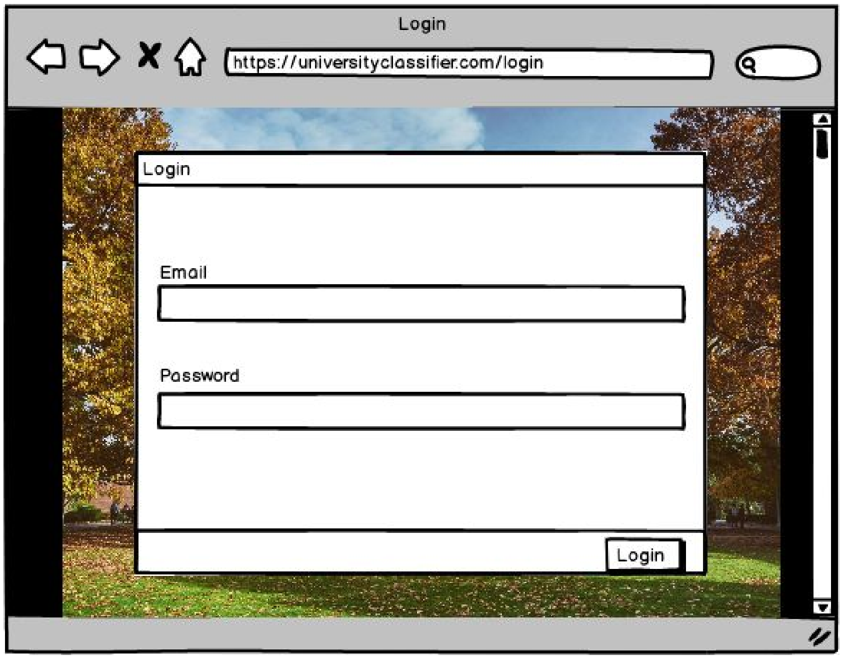


Figure 2 - The Login page

The Register and Edit Profile options look very similar to Figure 2, and can be found in Appendix 5.3, along with an example error message.

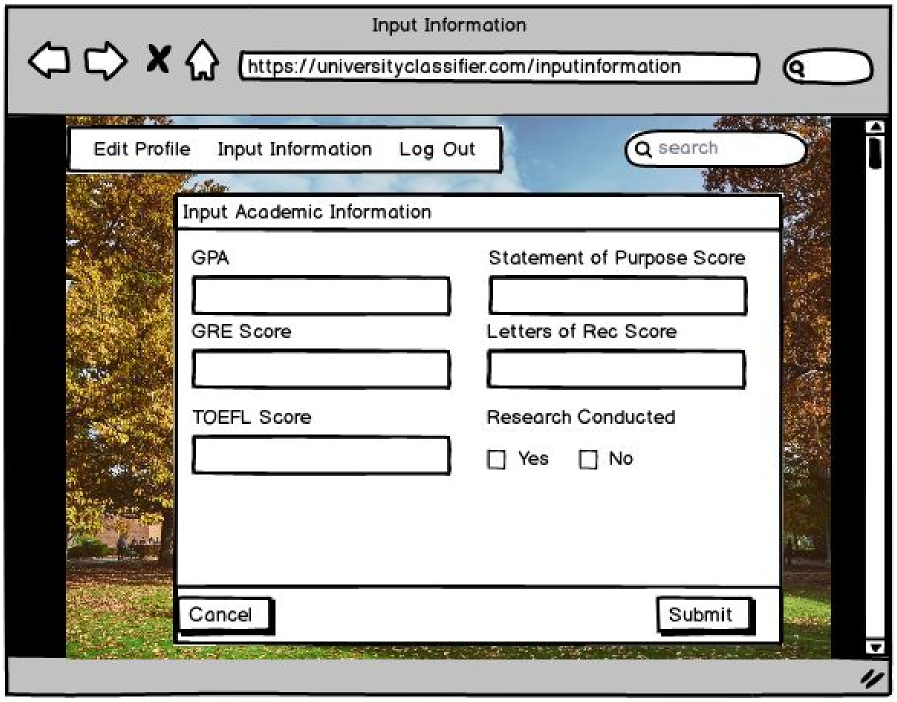
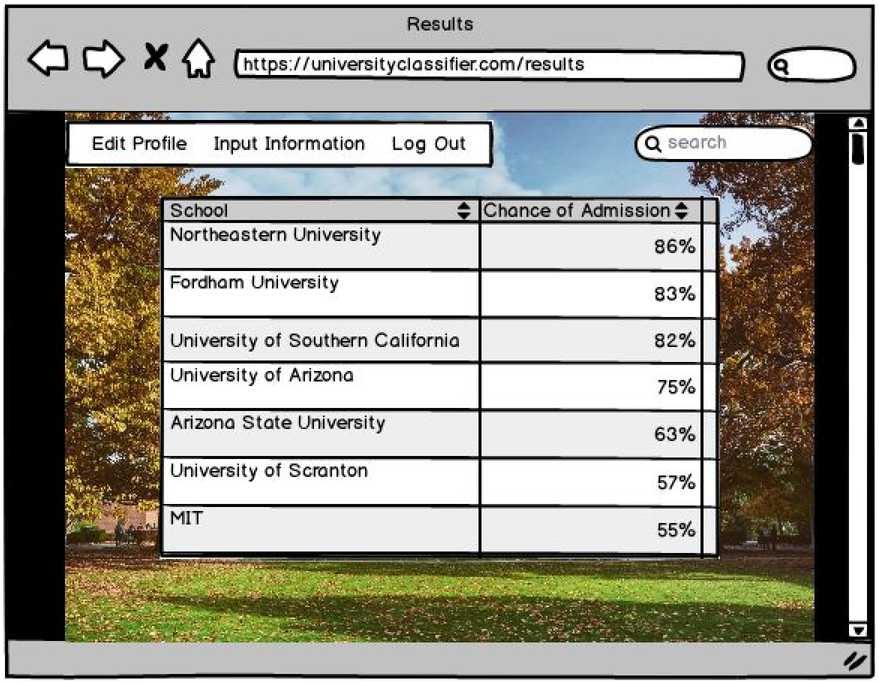
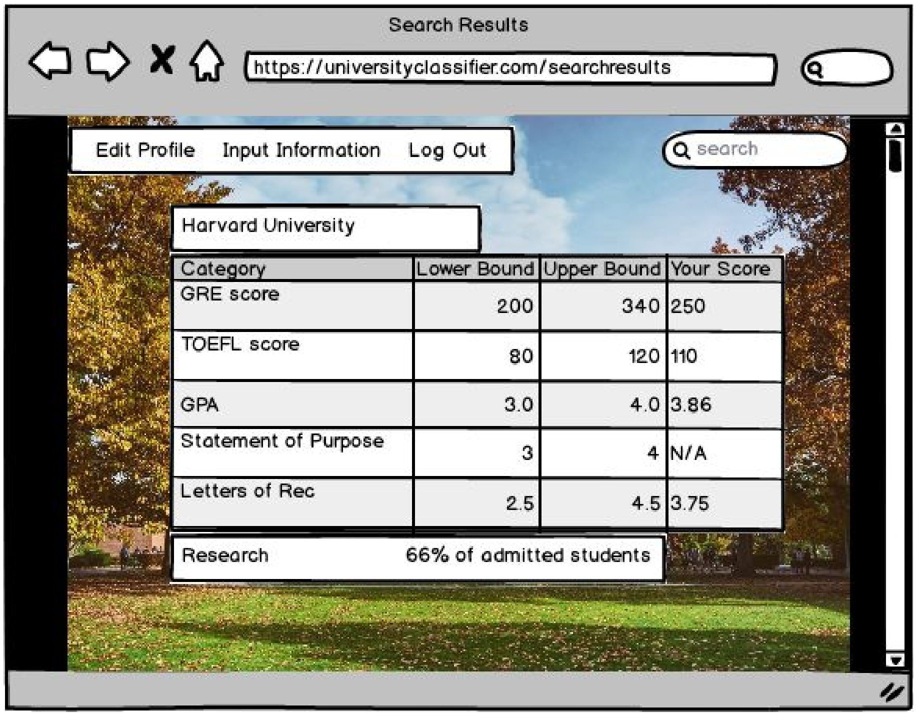


Figure 3 - This is the page where the user enters their academic information



School	Chance of Admission
Northeastern University	86%
Fordham University	83%
University of Southern California	82%
University of Arizona	75%
Arizona State University	63%
University of Scranton	57%
MIT	55%

Figure 4 - An example results page generated after the user presses “Submit”



Harvard University

Category	Lower Bound	Upper Bound	Your Score
GRE score	200	340	250
TOEFL score	80	120	110
GPA	3.0	4.0	3.86
Statement of Purpose	3	4	N/A
Letters of Rec	2.5	4.5	3.75

Research 66% of admitted students

Figure 5 - The results of a university search

3. SPECIFIC REQUIREMENTS

3.1 External Interface Requirements

3.1.1 User Interface

3.1.2 Hardware Interface

N/A

3.1.3 Software Interfaces

When the application is opened, the user will be shown a page with the following two buttons:

1. Register - Leads to a page where the user will be prompted to enter information such as desired username and password.
2. Login - Lead to a page prompting the user to enter username and password. The interface will display if a password is incorrect and prompt the user to retry entering the details.

The user will reach the main application interface after successfully logging in. This page contains the following options:

1. Edit Profile - Allows the user to enter or modify information to be displayed on his or her account.
2. Enter Information - Prompts the user to enter academic information.
3. Search Schools - Prompts the user to select a school from the list of options.
4. Logout - Lead the user back to the initial page with ‘Register’ and ‘Log in’

Both ‘Enter Information’ and ‘Search Schools’ include a ‘Generate Results’ button which will display the results if all of the necessary information is entered; otherwise, the user will be prompted to fill out the missing information. Clicking on that button will lead the user to the results page where they will be able to see the results generated from the algorithm, which would include either a list of schools that the user can likely get into or a percent chance of admission to the chosen school. This page has a ‘return’ button in order to return to the main screen.

Our system does not require an interface between hardware and software since no hardware components are used. The system will include an interface between the application and the algorithm. The application will send the user results to the algorithm, the algorithm will compute results based on the user inputs, and the generated results will be sent back to the application to be displayed. The application will also need an interface between itself and the dataset of schools in order for the user to select an option. An interface between the system and the dataset for scores and admission percent chances will not need to be included since the algorithm will already be trained using this data in order to generate its results.

3.2 Detailed Description of Functional Requirements

3.2.X Template for describing functional requirements

The template used to describe functional requirements is as follows:

- *Purpose:* What is accomplished by this functional requirement
- *Inputs to the Component:* What the user must input to the computer

- *Processing:* What the computer does to meet the requirement
- *Outputs:* What is returned by the system to the user

3.2.1 Create an account:

- *Purpose:* This function allows the user to access all other features of the application. Without an account, a user will not be able to successfully log in, so they cannot reach the main page of the application.
- *Inputs to the Component:* User will type in the following inputs: Desired username, password, and email address
- *Processing:* The system will check that the username is not already taken and that the password meets the standard requirements
- *Outputs:* With valid inputs, the user's account information will be stored so they will be able to login; with invalid inputs, the user will be prompted to reenter information

3.2.2 Log In:

- *Purpose:* This function allows a user to view their specific results generated by the application without having to input their information every time they access it.
- *Inputs to the Component:* User will type in the following inputs: Chosen username, and password
- *Processing:* The system will check that the username and password are both registered within the system, as well as verifying that they match each other.
- *Outputs:* With valid inputs, the user may access their account information and utilize the application's otherwise functions; with invalid inputs, the user will be given an error notification.

3.2.3 Update Profile:

- *Purpose:* This function allows a user to add extra information to their profile, allowing for more accurate searches and results when using the application.
- *Inputs to the Component:* User will input data, such as GPA, names of classes, desired field of study, and extracurricular activities.
- *Processing:* The system will take all inputs, adjusting the data if it conflicts with previous inputs. This will happen in cases such as updating the GPA. If there is no conflict, it will add the data to the profile.
- *Outputs:* The user's profile will include the new data inputted.

3.2.4 Input Information:

- *Purpose:* This function allows the user to input their information the first time, with all of their credentials and merits. After doing so, this allows the system to provide a list of schools to show

- *Inputs to the Component:* User will input data, which includes GPA, extracurricular activities, and desired field of study.
- *Processing:* The system will take the inputs and save them under their relevant fields. For example, as GPA is a common input, it will be saved under the GPA label.
- *Outputs:* The user will be presented with a list of schools, along with an approximation of their chance to be admitted.

3.2.5 Search for Schools:

- *Purpose:* This function is to allow for users to see if they are in good standings to enter the school of their choice, and if not, they can see where they may be lacking.
- *Inputs to the Component:* The user will input the name of the school.
- *Processing:* The system will run through the database for a school name that matches the input. If no school matches the input, the output will reflect this.
- *Outputs:* Either a single school, or a list of schools if the input is not specific enough. it will show all schools with similar names., with the scores of tests and grades in order to enter. If there is no school in the database with the inputted name, it will show that there are no results.

3.2.6 Cancel:

- *Purpose:* This function will allow the user to stop what they are doing in the case that there is an error, or they wish to stop using the function.
- *Inputs to the Component:* The user will click on a cancel button that will be present in all functions.
- *Processing:* The system will act once the cancel button is pressed at any function.
- *Outputs:* The user will be presented the page prior to when they cancelled.

3.2.7 Error Message:

- *Purpose:* This function ensures that all inputs will comply with what the application is ready to handle.
- *Inputs to the Component:* The user can input anything into any function.
- *Processing:* The system checks if the input from the user does not match the specified requirements of the function they are using.
- *Outputs:* The system will output an error message under the invalid input on the page that they are currently viewing.

3.3 Performance Requirements

- Shorten processing time of the application
  - The time between generating and displaying results should be within 20 seconds
  - This will most likely be done with a python script

- Be able to handle requests from multiple users at once
    - Not a huge priority, with the only expected drawback being extended loading times
- User switching between pages should be within 5 seconds

3.4 Quality Attributes

- Security is a main factor
    - Users should never have their data stolen by third parties from the system
    - 3 attempts allowed per account before a security email is sent
- Application should be available almost all the time
    - System should be going down only for maintenance and updates

4 Other requirements: N/A

5 Appendix

5.1 Use Cases

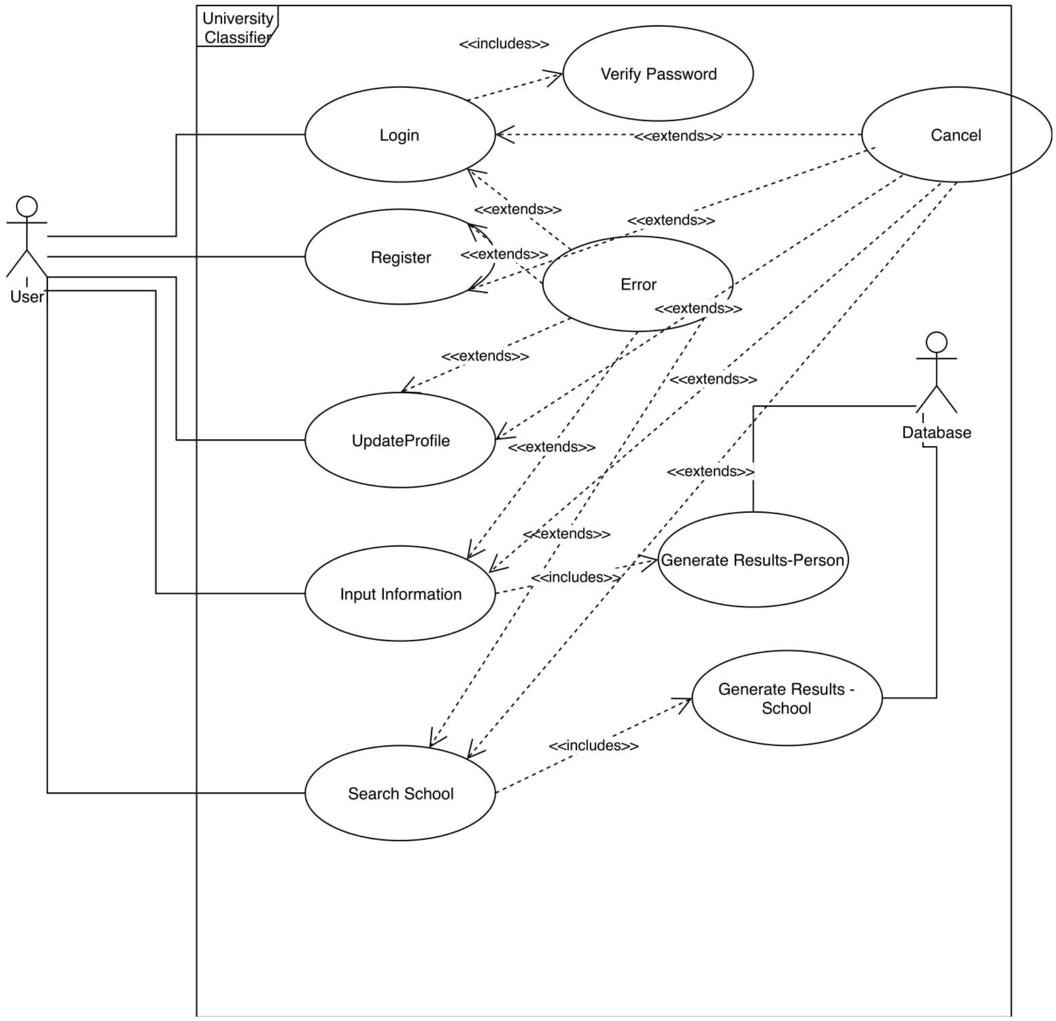


Figure 6 - Use case diagram

USE CASE SPECIFICATIONS TEMPLATE

Use Case Identification and History

Use Case ID:	Classifier.Register.1	
Use Case Name:	CreateProfile	**Version No:**
End Objective:	Create a profile that a user can sign into and modify	
Created by:	University Classifier	**On (date):**
Last Update by:		**On (date):**
Approved by:		**On (date):**
User/Actor:	User who accesses application for the first time	
Business Owner Name	University Classifier	**Contact Details**
Trigger:	User triggers use case by clicking ‘Register’ button	
Frequency of Use:	Once per user	

Preconditions

User does not already have a profile

Basic Flow

Step	**User Actions**	**System Actions**
1	User selects "Register"	System displays Create Profile page
2	User fills out profile fields and clicks "Submit"	System validates input and creates account
3		Displays Successful Creation page

Alternate Flow

Step	**User Actions**	**System Actions**
1	User selects "Register"	System displays Create Profile page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow

Step	**User Actions**	**System Actions**
1	User selects "Register"	System displays Create Profile page
2	User fills out profile fields and clicks "Submit"	System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions

Application displays Successful Creation or error page

Includes or Extension Points

Includes:
a. Verify Information
Extends:
a. Error
b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case creates a user profile

Use Case ID:	Classifier.Login.1		
Use Case Name:	Login	**Version No:**	1
End Objective:	Sign into profile that user has created		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User who has created an account		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User triggers use case by clicking ‘Log In’ button		
Frequency of Use:	Every time application is opened		

Preconditions
User has an existing account

Basic Flow

Step	**User Actions**	**System Actions**
1	User selects "Log In"	System displays Log In page
2	Users enters username and password and clicks "Submit"	System validates input logs user in
3		Displays profile

Alternate Flow

Step	**User Actions**	**System Actions**
1	User selects "Log In"	System displays Log In page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow

Step	**User Actions**	**System Actions**
1	User selects "Log In"	System displays Log In page
2	User fills out username and password and clicks "Submit"	System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions
Application displays profile or error page

Includes or Extension Points

Includes:
a. Verify Information
Extends:
a. Error
b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case logs a user into their profile

Use Case ID:	Classifier.UpdateProfile.1		
Use Case Name:	UpdateProfile	**Version No:**	1
End Objective:	User modifies the information on his/her profile		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User who is logged into account		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User triggers use case by clicking ‘Edit profile’ button		
Frequency of Use:	Not often unless a user makes frequent changes to his/her profile		

Preconditions
User has an existing account and is logged into their profile

Basic Flow		
Step	**User Actions**	**System Actions**
1	User selects "Edit Profile"	System displays Edit Profile page
2	Users edits desired information and clicks "Submit"	System validates, updates and saves information
3		Displays profile

Alternate Flow		
Step	**User Actions**	**System Actions**
1	User selects "Edit Profile"	System displays Edit Profile page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow		
Step	**User Actions**	**System Actions**
1	User selects "Edit Profile"	System displays Edit Profile page
2	User fills out username and password and clicks "Submit"	System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions

Application displays updatedprofile or error page

Includes or Extension Points

Includes:
a. Verify Information
Extends:
a. Error
b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case updates a user profile

Use Case ID:	Classifier.InputInfo.1		
Use Case Name:	InputInfo	**Version No:**	1
End Objective:	User inputs their scores and college information		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User who is logged into account		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User triggers use case by clicking ‘Enter Academic Information’ button		
Frequency of Use:	One time per user, but users can update their information if their scores change		

Preconditions
User must have an existing account and be logged in. Must be on the update page.

Basic Flow		
Step	**User Actions**	**System Actions**
1	After the user first creates their profile, they will be brought to a blank profile page.	System displays blank profile page
2	Users edits desired information and clicks "Save"	System validates, updates and saves information
3		Displays profile

Alternate Flow		
Step	**User Actions**	**System Actions**
1	After the user first creates their profile, they will be brought to a blank profile page.	System displays blank profile page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow		
Step	**User Actions**	**System Actions**
1	After the user first creates their profile, they will be brought to a blank profile page.	System displays blank profile page
2	Users edits desired information and clicks "Save"	System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions

Application displays profile or error page

Includes or Extension Points

Includes:
- a. Verify Information

Extends:
- a. Error
- b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case allows a user to input information into their profile

Use Case ID:	Classifier.GenerateResults.1		
Use Case Name:	GenerateResults	**Version No:**	1
End Objective:	Take user information and produces a list of percent chance of admission to a variety		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User who has logged into their account		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User enters information and presses "Generate Results"		
Frequency of Use:	Very often since users this is the main purpose of the application		

Preconditions
User must be logged in and have valid profile information.

Basic Flow		
Step	**User Actions**	**System Actions**
1	User hits "Generate Results"	System validates profile information and searches database
2		Displays list of schools and percent change of admission

Alternate Flow		
Step	**User Actions**	**System Actions**
1	User clicks on "Generate Results" button.	System displays search page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow		
Step	**User Actions**	**System Actions**
1	User clicks on "Generate Results" button.	System validates input and finds an error
2		Displays error message and prompts user to try again

Post Conditions

Application displays school information or error page

Includes or Extension Points

Includes:
a. Verify Information
Extends:
a. Error
b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case allows a user to search for schools

Use Case ID:	Classifier.SchoolSearch.1		
Use Case Name:	SchoolSearch	**Version No:**	1
End Objective:	User searches for graduate school they are interested in		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User who has logged into their account		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User triggers use case by clicking ‘Search For Schools’ button		
Frequency of Use:	Very often since users will likely check for chance of admission at multiple schools		

Preconditions
User must be logged in and have profile information.

Basic Flow		
Step	**User Actions**	**System Actions**
1	User clicks on "Search for Schools" button.	System displays search page
2	User enters a school and presses "Submit"	System searches database for information
3		Displays school information

Alternate Flow		
Step	**User Actions**	**System Actions**
1	User clicks on "Search for Schools" button.	System displays search page
2	User presses "Cancel" button	System returns to home page and does not save data

Error Flow		
Step	**User Actions**	**System Actions**
1	User clicks on "Search for Schools" button.	System displays search page
2	User enters a school and presses "Submit"	System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions

Application displays school information or error page

Includes or Extension Points

Extends:

a. Error
b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case allows a user to search for schools

Use Case ID:	Classifier.VerifyInfo.1		
Use Case Name:	VerifyInfo	**Version No:**	1
End Objective:	Ensure the user has entered valid info		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User triggers use case by clicking ‘Search For Schools’ button		
Frequency of Use:	Very often since it is included in all other use cases		

Preconditions
N/A

Basic Flow		
Step	**User Actions**	**System Actions**
1	User inputs information and presses "Submit"	System searches database for information
2		Displays requested information

Alternate Flow N/A		
Step	**User Actions**	**System Actions**
1		
2		

Error Flow		
Step	**User Actions**	**System Actions**
1	User inputs information and presses "Submit"	System searches database for information
2		System validates input and finds an error
3		Displays error message and prompts user to try again

Post Conditions

Application displays information or error page

Includes or Extension Points

Extends:

a. Error

b. Cancel

Special Requirements

N/A

Buisness Rules

N/A

Other Notes

This use case validates information

Use Case ID:	Classifier.Cancel.1		
Use Case Name:	Cancel	**Version No:**	1
End Objective:	User is returned to home menu		
Created by:	University Classifier	**On (date):**	
Last Update by:		**On (date):**	
Approved by:		**On (date):**	
User/Actor:	User		
Business Owner Name	University Classifier	**Contact Details**	
Trigger:	User is returned to home menu		
Frequency of Use:	Not often; whenever a user decides to not follow through which what they clicked on		

Preconditions
User is in a use case with cancel functionality

Basic Flow		
Step	**User Actions**	**System Actions**
1	User inputs information and changes their mind, so presses "Cancel"	System validates, updates and saves information
2		System returns to home page and does not save data

Alternate Flow N/A		
Step	**User Actions**	**System Actions**
1		
2		

Error Flow N/A		
Step	**User Actions**	**System Actions**
1		
2		
3		

Post Conditions
Application returns to home screen

Includes or Extension Points
N/A

Special Requirements
N/A

Buisness Rules
N/A

Other Notes
This use case validates information

5.2 Class Diagram

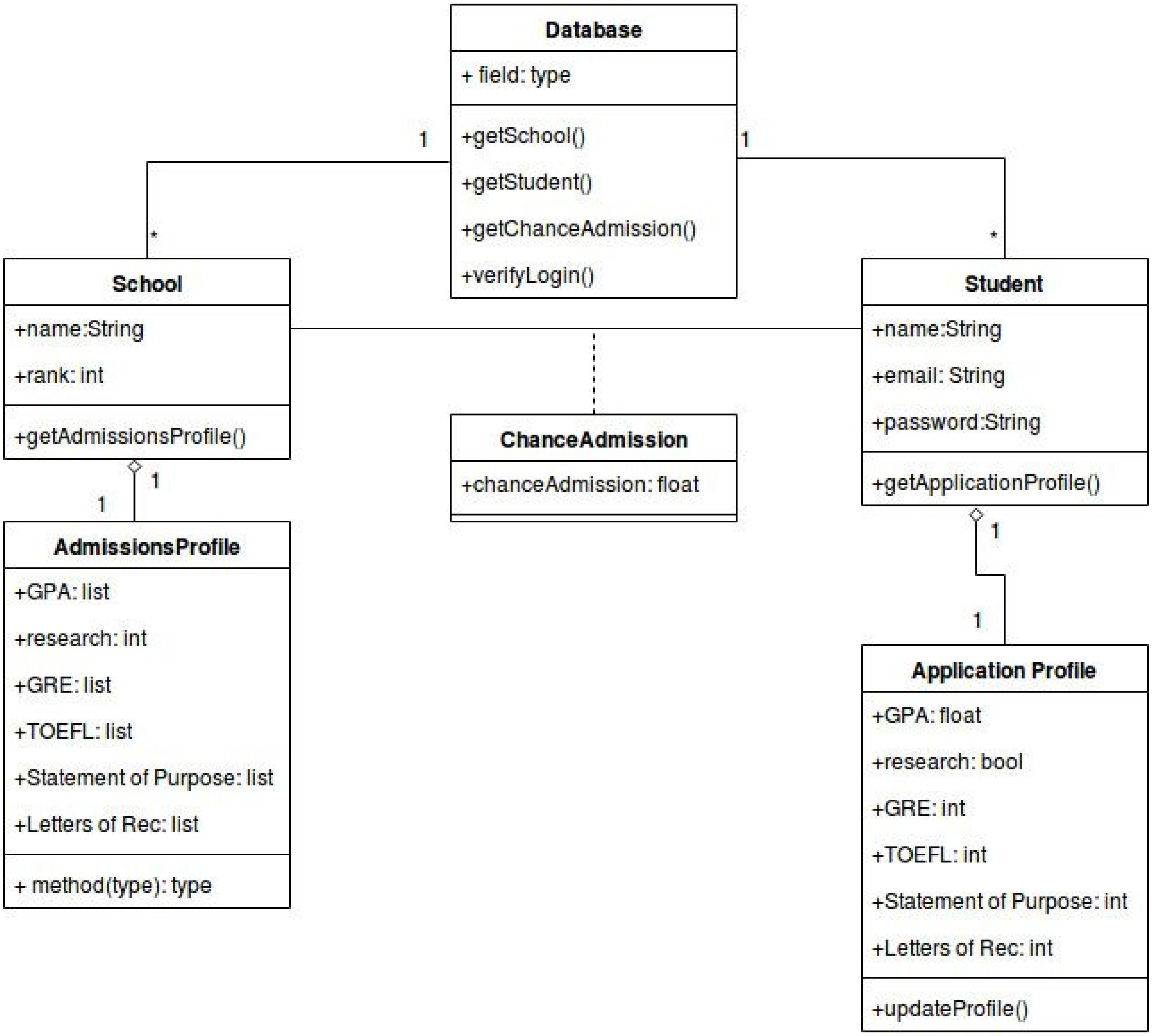


Figure 7 - The class diagram

5.3 Extra Wire Frames

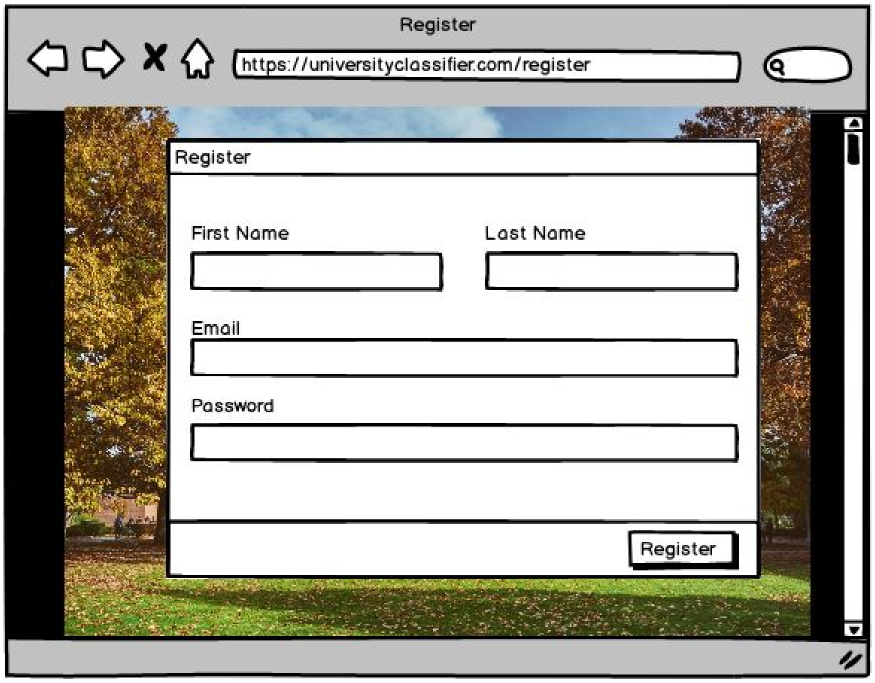
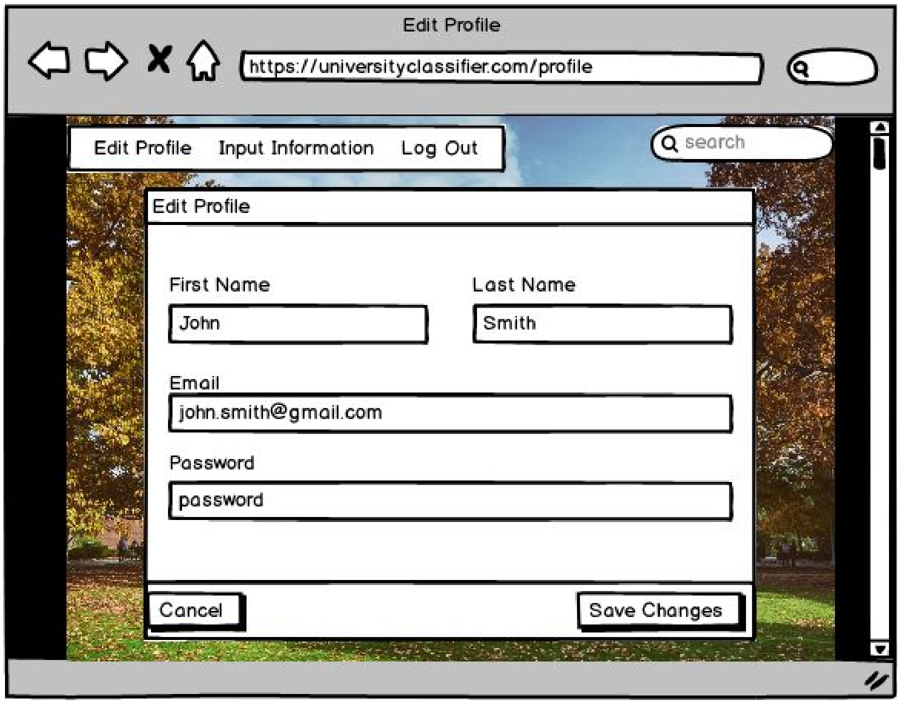


Figure 8 - The register page



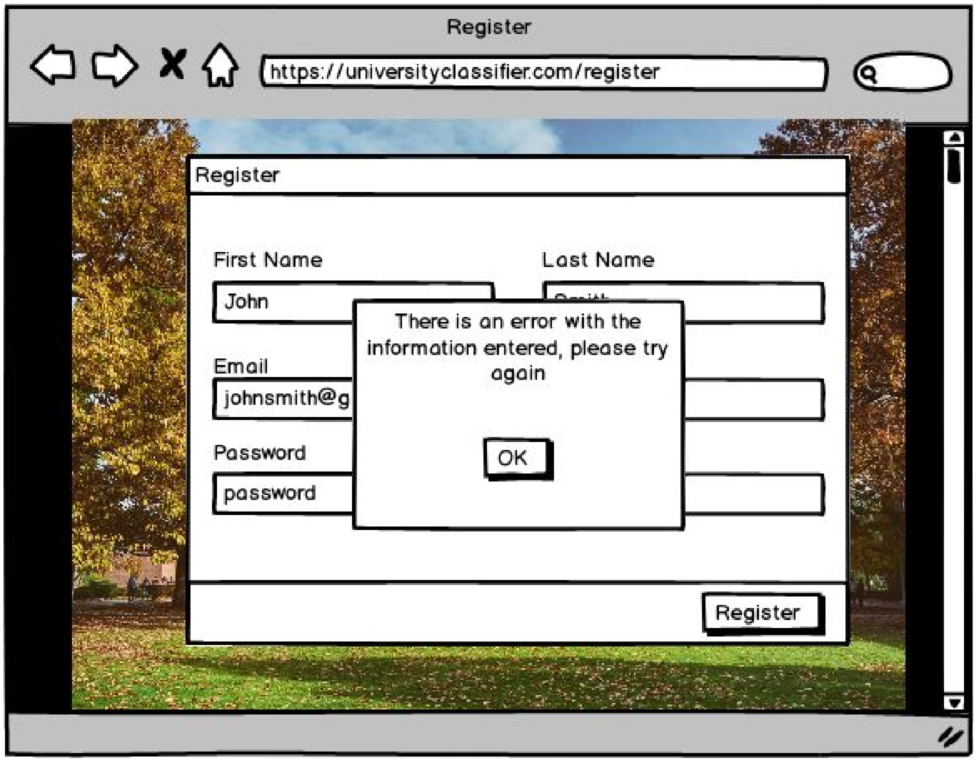
First Name
John

Last Name
Smith

Email
john.smith@gmail.com

Password
password

Figure 9 - The edit profile page



There is an error with the information entered, please try again

Figure 10 - The error message