Deployment Instructions

1.1 Create a Dynamic Web Project in Eclipse

1.2 Open folder "/Full-Integrated" after unzipping

1.3 Replace your newly created Dynamic Web Project files

-Replace the default "/src" and "/WebContent" folder from your newly created project with the "/src" and /WebContent" folders from "/Fully-Integrated"

1.4 In the "/WebContent" folder run LogIn.html on Tomcat Server

-Right click LogIn.html and click "Run as" -> "Run on Server"

1.5 Finished!

Tara Conti, Ritika Dendi, Tyler Fong, Aaron Ly, Lauren Tsai, and Steven Vorona

Professor Adamchik

CSCI 201

16 October 2020

Final Project Proposal

Concept of Idea:

Our team is planning to create a web application that allows USC students to see reviews of past courses (by integration with websites like RateMyProfessor API), see past students who have taken it and written reviews, and message them about their experiences. Students will also be able to input their class schedules on to a calendar within the application.

Login Functionality:

Guests will be able to view the RateMyProfessor rating for the specific class and professor (as opposed to the general score) and see how many students have taken that course.

Registered users will have the abilities guests have and will be able to save their preferred classes, register their own previous classes, and be able to "friend" other users see shared classes. Registered users can also access contact information of other users to consult about current or future classes.

Profile Section

Users will also have a profile section that their friends can view. This section will contain information such as their name, graduation date, major, past and current classes (including the professors they took the class with).

Search Section

The core functionality of this web application is that users will be able to search for classes and see the other students that have taken the course. They will also be able to filter for specific professors for the course. If the user has a friend that has taken the course before, it will be indicated. Users will be able to see the contact information of the other students so they can reach out to get more information about the course or professor.

Potential Tools and Outside Courses Used

Currently, we plan to use React and the Java suite (HTML/CSS/JS) for our front-end. We plan to use Java along with our knowledge of data structures (CSCI 104) and algorithms (CSCI 170/270) to develop the back-end. For other features like integration with RateMyProfessor, Google, and Apple Calendar, we will be using the respective APIs. For login functionality, we will use SQL and JDBC to store chat messages, class information, pre/post-requisites. “Cheerio” Web scraping using javascript to scrape USC sites for relevant information. Or, alternatively “beautiful soup” for web scraping in Python.

Software Requirements Specification

for

SCearch

University of Southern California
Fall 2020

Team Members:
Tara Conti
tconti@usc.edu
Ritika Dendi
dendi@usc.edu
Tyler Fong
tylerf@usc.edu
Aaron Ly
aaronly@usc.edu
Lauren Tsai
lstsai@usc.edu
Steven Vorona
vorona@usc.edu

Weekly Meetings: Fridays at 4pm PST

TABLE OF CONTENTS

1. Introduction

1.1 Purpose

This SRS for SCearch release 0.1.0 outlines the functional and nonfunctional requirements for the system being built. This document is to be used as a reference for the team members building and testing the system.

1.2 Project Scope

The SCearch system allows students to make a more informed decision about the classes and professors they want to take for the upcoming academic term by having the ability to search for classes, view information about professors teaching those classes, write reviews about their experiences in a class, and contact past students who have taken the class from a professor.

2. SRS

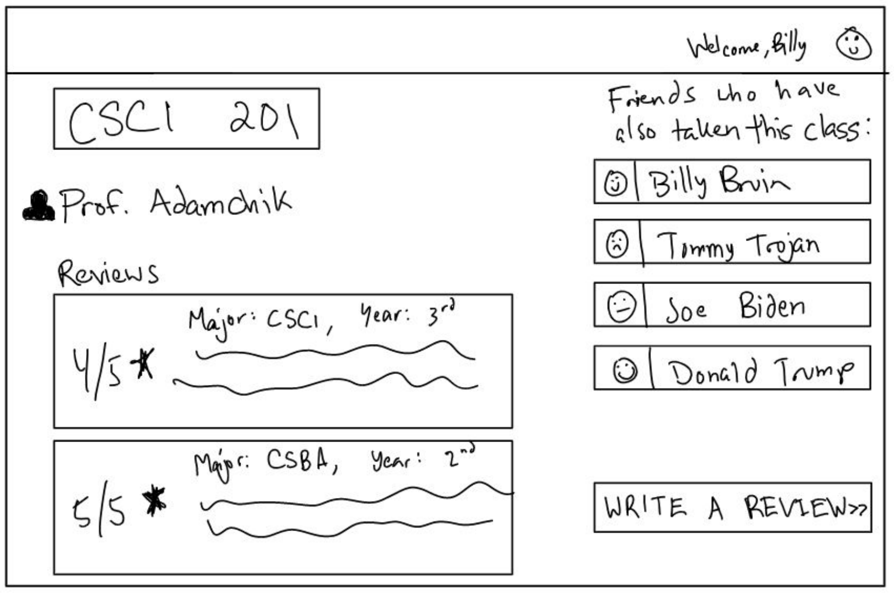
2.1 Product Perspective

This product will allow students to view class and professor information. Students can search for classes from the web-app and view a list of professors teaching that class. This product will also allow students to view information about the professor, such as overall rating and “tags” from popular review website RateMyProfessor using an API. Students will also be able to contact past students who have taken the class and ask them about their experiences through email attached to reviews. Additionally, students will also be able to post their own reviews about a particular course/professor they took and have their email attached to increase credibility. Overall, the system will take the best features from RateMyProfessor and enhance them with a more streamlined approach to finding and comparing professors, as well as increase credibility by verifying reviews to students.

2.2 Classes Data

2.2.0 Basic Interface

View Class Information



2.2.1 Description

Students will be able to search and look at information about classes and professors. This includes the list of classes available for the upcoming semester, as well as the professors teaching it and student reviews/ratings, on those professors.

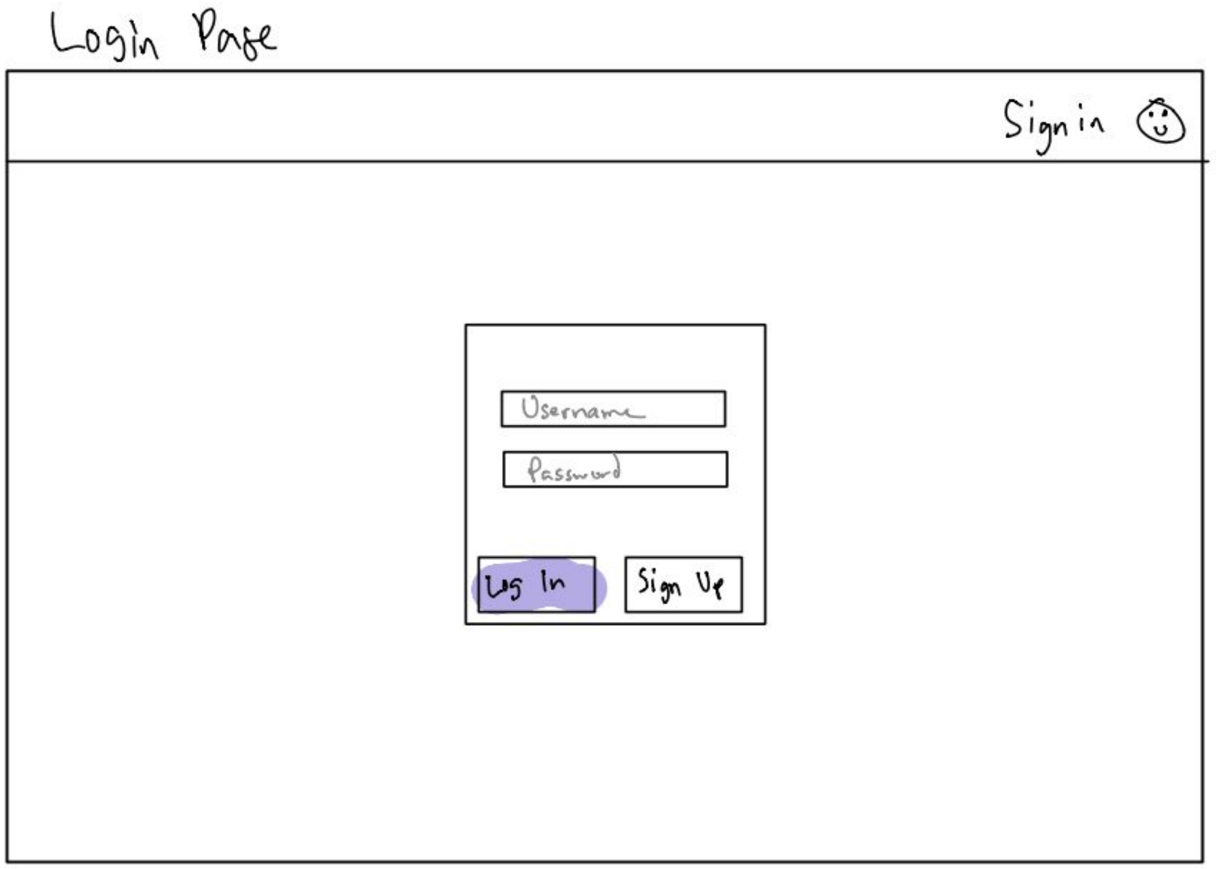
2.2.2 Functional Requirements

- ***Logged-In-Specific Functional Requirements***
    - Users can search for classes and add them to their schedule
        - Each class will also list
            - Professor
            - Course title
            - Course number
            - Section number
        - Once the student adds a class, their information will also be added to the list of students who are taking/have taken the class (email will be visible to other logged in students)
    - Each class will also have professor information
        - Overall rating on rate my professor
        - Rate my professor "tags"
    - Users will be able to view the list of other student who have taken the course
        - Emails will be listed so that the users can contact other students
    - Users will be able to view reviews that other students have posted about the class

        - Reviews will be professor specific
            - Some reviews may be for other classes the professor has taught
    - Users will be able to write reviews for specific professors
- ***Guest-Specific Functional Requirements***
    - Guests can view classes and reviews of professors teaching those classe
        - Reviews will be from past students
        - Classes can be searched for in the search bar using department code
        - Classes will include all classes from the current/upcoming semester
    - Professor information from RateMyProfessor using the respective API will include
        - Overall Rating
        - "Tags"
        - Link to reviews page of the professor
    - Class and professor information will be pulled from classes.usc.edu
        - Using an R scrapper

2.3 Login Functionality

2.3.0 Basic Interface



2.3.1 Description

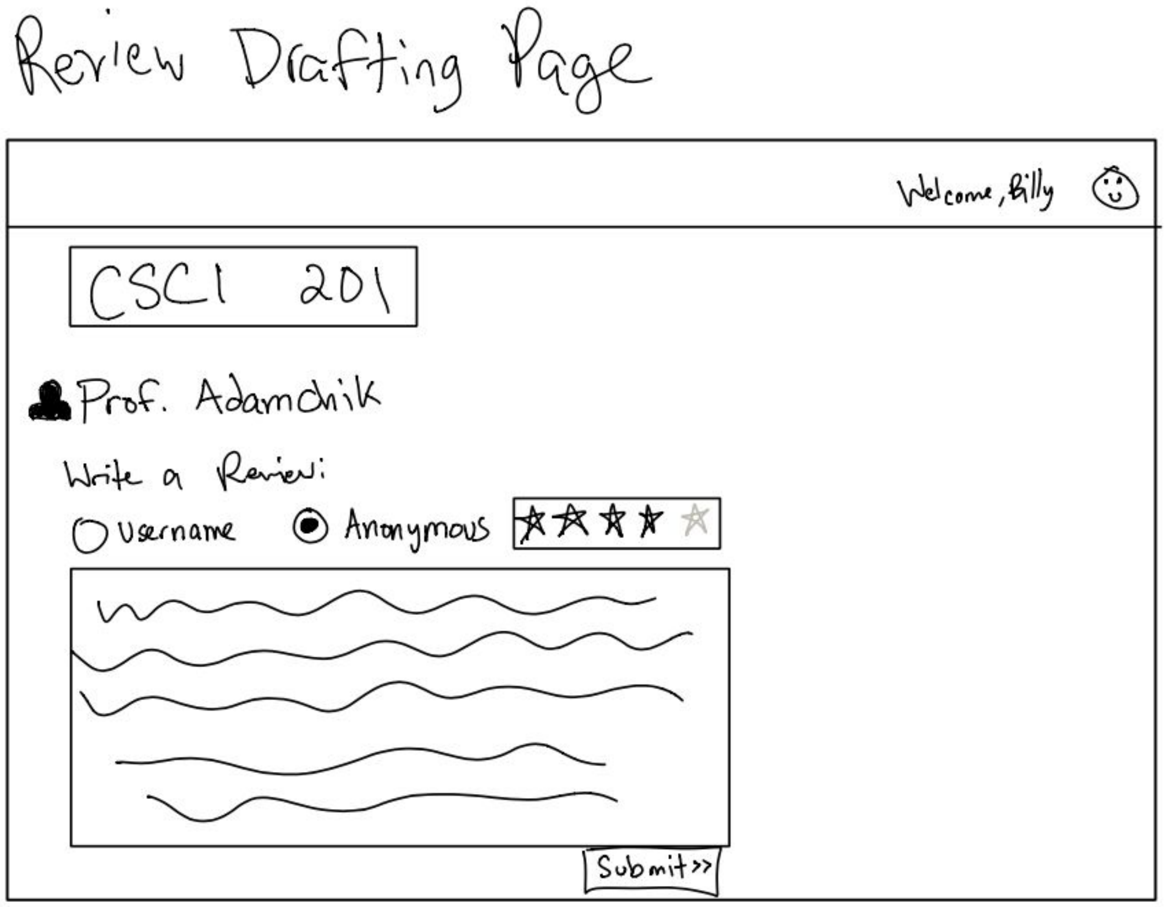
Students will be able to login and have their information stored. Their profile will hold their username, their first and last name, major, year, email, and classes taken. There will also be a list of each user's friends and a list of every class and who has taken it.

2.3.2 Functional Requirements

- Maintain a database of each user's Username and Password for login functionality
- Attach each user to a unique ID for easy reference to log into the web application
- Be able to see which users a particular user is friends with
  - This will be used so that users can see which of their friends have taken a class so that they can reach out to them
- Holds profile of users including their username, name, major, email, and classes
  - This information will be used on a profile page and to look up a user when someone wants to add them as a friend
- Get a list of students (by id) who have taken a given course
  - This will be used when a user looks up a class to see which students have taken that class already

2.4 Review Writing Functionality

2.4.0 Basic Interface



2.4.1 Description

User will be able to write reviews on an input and save these reviews to their specified professor or class

2.4.2 Functional Requirements

Review Attributes on display:

- Time Stamp
- Review itself
- Anon/User
- What class and professor review is for

3. Technical Specifications

3.1 Data Storage

3.1.1 Description

The database will hold four tables: Login Table, Friends Table, Profile Table, and Classes Table. The tables will be queried behind the scenes to allow login and save user and class information. This will use the software My SQL Workbench to create SQL queries and Eclipse for Java.

3.1.2 Database Tables

- **Login Table**

id	user	pass
int	varchar	varchar

  - Will be queried when a user logs in or creates a new account

- **Profile Table**

id	user	firstname	lastname	major	year	email
int	varchar	varchar	varchar	varchar	int	varchar

  - Will be queried when a User is searching for a friend
  - Will be queried to display User's information on their Profile

- **ClassInfo Table**

course num	course title	units	type	time	days	instructor	classrating	profrating
varchar	varchar	double	varchar	varchar	varchar	varchar	float	float

  - Will be queried when the user searches for a class
  - Will return the professor, time, days, and rating that corresponds to the class that was searched for
  - *For clarity:*
    - coursenum = course number in the USC registrar (i.e. CSCI-201)
    - coursetitle = course name in the USC registrar (i.e. Principles of Software Development)
    - type = lab, discussion, lecture, etc.

    - time = time in the day when section occurs.
    - days = days when the section is being taught.

- **Review table**

id	user	class	prof	major	year	review
int	varchar	varchar	varchar	varchar	int	varchar

  - Will be queried when the user searches for reviews of a class
  - Will return review text, basic class information, and (optionally) corresponding user information (queried from profile table by “user” string)

3.1.3 Referenced CSV Lists

- **Certain elements are more conveniently stored in CSV lists than an SQL database**
  - List of a user’s previous courses
  - List of students previously enrolled in a course
  - List of a user’s friends
- **Lists**
  - friends/[user].csv
    - This file contains a single column list of the usernames of [user]’s friends
  - classlist/[user].csv
    - This file contains a single column list of the classes [user] has indicated they have previously taken
  - prevstudents/[classname].csv
    - This file contains a single column list of the usernames of students who have indicated that they were previously enrolled in [classname]

3.2 Search

3.2.1 Description

The data will be stored in a MySQL database. It will use JDBC to connect the database with the Java programs in order to query, insert, and update the database. There will be multiple tables in the database with each pertaining to a different project feature.

Some data is more conveniently stored in CSV format rather than database format. Upon registration, each user will have two CSV lists generated with their usernames: a friends list and a class list, storing the user’s friends and previous/current classes respectively. Additionally, each class will have a corresponding named list of students who previously took that course.

3.2.2 Class and Professor Data

The data for the USC professor ratings was obtained by scraping the "Rate My Professor" website using the python API RateMyProfesorPyAPI. This data was then saved in a csv file and used to prepopulate the database. An R web-scraper was used on webreg to gather information about fall 2020 classes. This information included course number, title, professor, time, days, units, etc. The ClassInfo table was generated by using a SQL script to combine the classes and professor data. This allowed the class, professor, and professor rating to be displayed in one cohesive table.

3.2.3 Search Functionality

3.2.3.1 Class Professor Search

When a user searches for a course, the application will prompt for a course number. This course number will be used to query the database. The function will search for all the entries in the ClassInfo table that contain the search term in the coursenum field. The return table contains professor name, overall rating, time, and days for the corresponding session.

3.2.3.2 Professor Review Search

After a user searches for a course, a list of professors who are teaching those courses will be displayed. A user will then be able to choose a professor from that list to see their reviews. This will be done by using the professor name to query the reviews table in the database. The application will then return all the entries that correspond to the selected professor. Each review will also have a foreign key that references a user (who is the author of the review) in the user table. This allows the user to optionally display their contact information alongside reviews, similar to how Piazza allows users the option to create either named or Anonymous posts. This inclusion of contact information will allow interested students to interact with the reviewer (if the reviewer chooses not to remain anonymous).

4. Testing Plan

4.1 White-Box Testing

4.1.1 Unit Testing

For unit testing, we're going to test each method individually to ensure that it is outputting as expected.

4.1.1.1 Search Functionality

For the search functionally, the tests will consist of the following functions:

- printClassSearchRes()
  - Input: Class name
    - Ex1: CSCI-201
    - Ex2: BUAD-312
    - Ex3: CSCI-104
  - Output: Array list of professors teaching that class
    - Ex1: Array.size() = 1 (Victor Adamchik)
    - Ex2: Array.size() = 1 (Jacob Bien)
    - Ex3: Array.size() = 1 (Sandra Batsita)
- selectProf()
  - Input: Array list of professors size
    - Ex1: 1
    - Ex2: 10
    - Ex3: 31
  - Output: Number of the professor in the data table
    - Ex1: 0
    - Ex2: 9
    - Ex3: 30
- printReviews()
  - Input: Result set of the professors reviews
    - Ex1: Victor Adamchik
    - Ex2: Sandra Batista
    - Ex3: Shaddin Dughmi
  - Output: String of all reviews received by a professor (on our platform)
    - Ex1: 4 reviews
    - Ex2: 7 reviews
    - Ex3: 3 reviews
- getCourseToSearch()
  - Input: Class through user input
    - Ex1: CSCI-201
    - Ex2: BUAD-304
    - Ex3: MATH-126
  - Output: String of class that was typed in
    - Ex1: CSCI-201
    - Ex2: BUAD-304
    - Ex3: MATH-126

4.1.1.2 Login Functionality

For login functionality, the tests will consist of the following functions:

- CreateUser()
    - Input: username and password
        - Ex1: admin, abcd1234
        - Ex2: ritika, qwerty
    - Output: bool if the user was created and then found
        - Ex1: true
        - Ex2: true
- LogIn()
    - Input: username and password
        - Ex1: admin, abcd1234
        - Ex2: ritika, qwerty
    - Output: bool if login succeeds
        - Ex1: true
        - Ex2: true
- UpdateProfile()
    - Input:user id and attribute to update, attribute information
        - Ex1: 1, firstname, ritika
        - Ex1: 1, firstname, admin
    - Output: entire profile
        - Ex1: 1, ritika, ritika, dendi, cs, 3, dendi@usc.edu
        - Ex2: 1, ritika, admin, dendi, cs, 3, dendi@usc.edu
- AddFriend()
    - Input: user id and friend id
        - Ex1: 1, 3
        - Ex2: 2, 4
    - Output: bool if the friend added to the user's friend attribute and then found
        - Ex1: true
        - Ex2: true
- FindClassFriends()
    - Input: class
        - Ex1: csci201
        - Ex2: csci 356
    - Output: friend id's of who took the class
        - Ex1: ritika dendi, aaron ly, tara conti, steven vorona, lauren tsai, tyler fong
        - Ex2: ritika dendi

4.1.1.3 Review Functionality

For Review functionality, the tests will consist of the following functions:

- uploadReview()
    - Input: review description
    - Output: review is saved in database
- editReview()
    - Input: specified user that write a review

    - Output: newly refreshed review stored in database
- deleteReview()
    - Input: specified user that wrote a review
    - Output: review is deleted from database

4.2 Black-box Testing

4.2.1 Regression Testing

For regression testing, we're going to test the program as if we were users, to ensure that changes made to the code down the line doesn't "break" anything.

4.2.1.1 Search Functionality

For the search functionality, we're going to test that inputting a class followed by a professor returns the respective reviews and ratings of a particular professor.

Input:

- Ex1: CSCI-201, 1
- Ex2: BUAD-312, 1
- Ex3: CSCI-104, 1

Output:

- Ex1: Victor Adamchik, 4 reviews
- Ex2: Jacob Bien, 2 reviews
- Ex3: Sandra Batista, 7 reviews

4.2.1.2 Login Functionality

For the login functionality, we would create a new user, update their profile and see if the resulting changes appear on the home page. We would also add friends and then search for a class and see if the appropriate friends come up.

Input:

- Ex1: ritikadendi, password, firstname, ritika, 3, csci201
- Ex2: taraconti, pass2, last name, conti, 4, csci356

Output:

- Ex1: user created
  profile: 4, ritikadendi, ritika, [void], [void], [void], [void]
  friend added
  csci201 friends: tara conti
- Ex2: user created
  profile: 3, taraconti, [void], conti, [void], [void], [void], [void]
  friend added

csci356 friends: ritika dendi

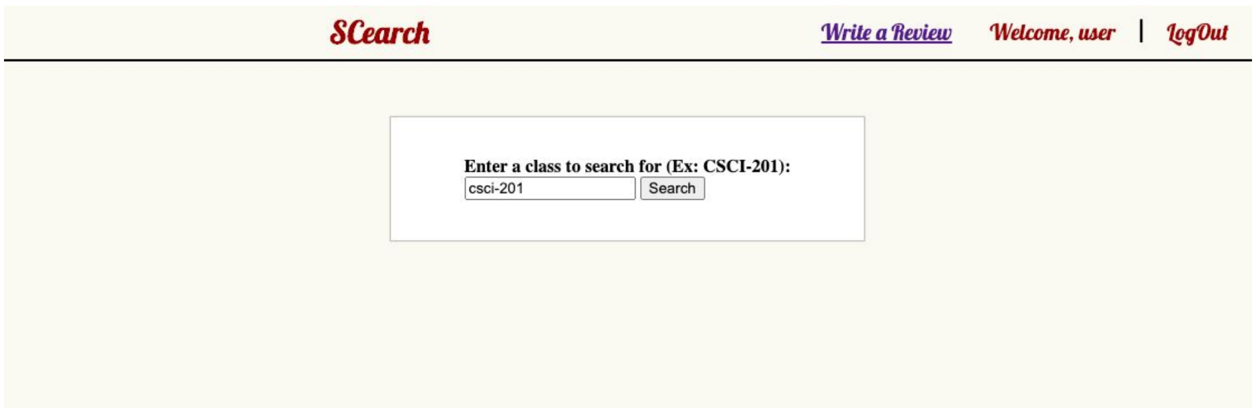
4.2.1.3 Review Functionality

For the review functionality, we would create a new user and upload reviews. We would then see whether we can edit or delete those reviews.

5. User Interface

5.1 Search Functionality

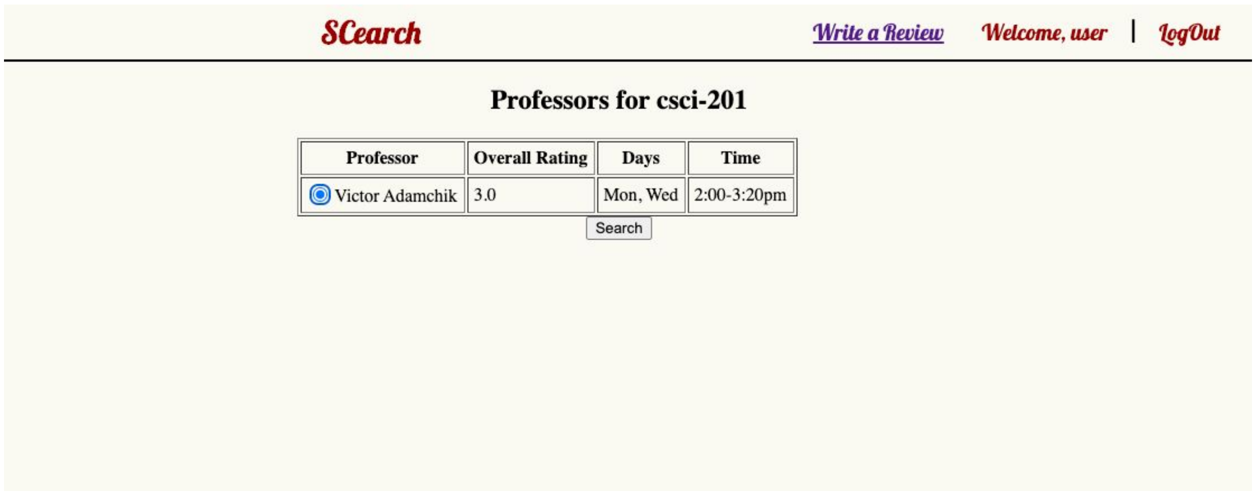
5.1.1 Searching Professors from Course



Enter a class to search for (Ex: CSCI-201):

csci-201 Search

5.1.2 Searching for Reviews from Professor

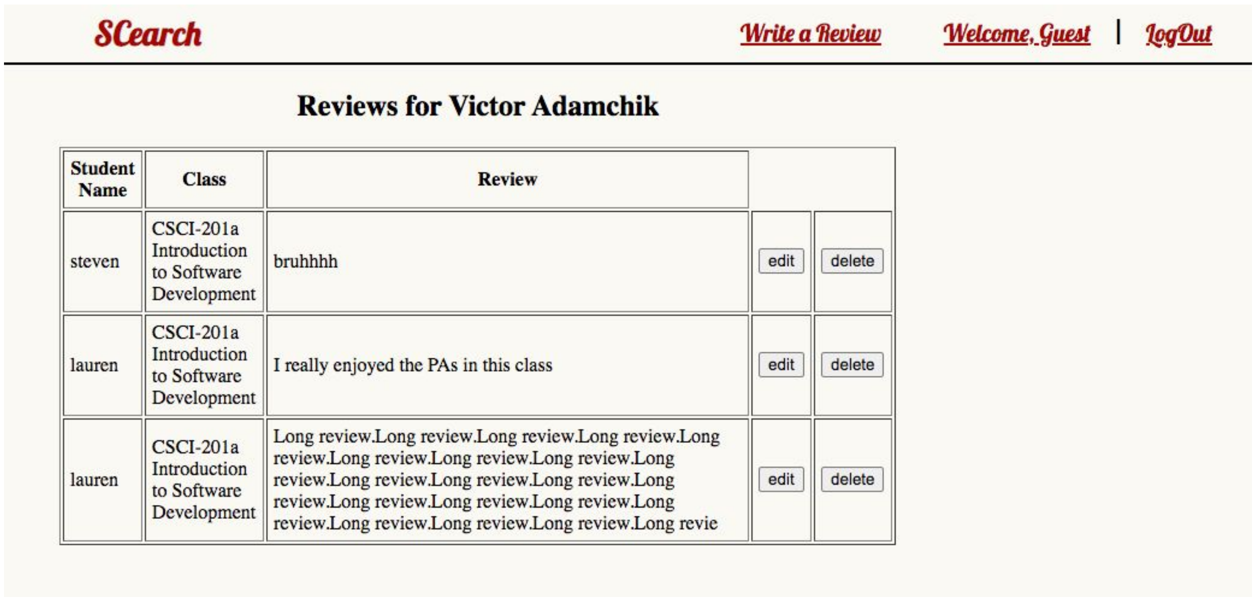


Professors for csci-201

Professor	Overall Rating	Days	Time
● Victor Adamchik	3.0	Mon, Wed	2:00-3:20pm

Search

5.1.3 Display Reviews for a Professor



SCearch

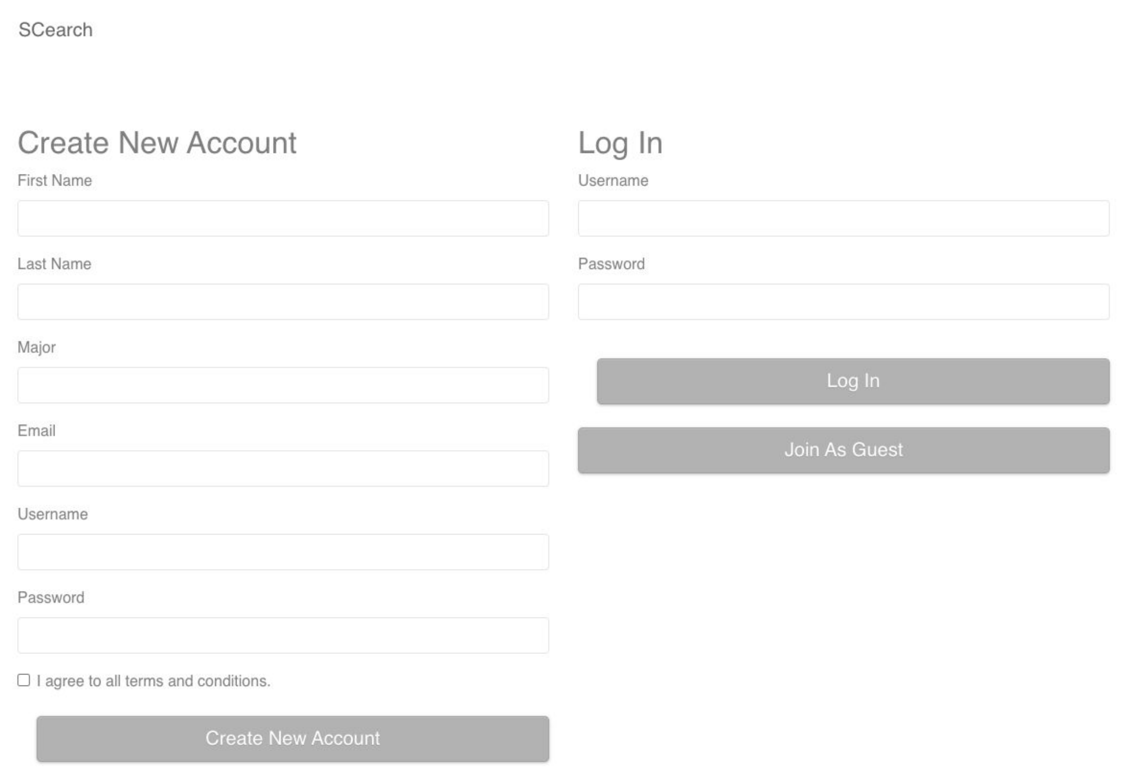
Write a Review | Welcome, Guest | LogOut

Reviews for Victor Adamchik

Student Name	Class	Review		
steven	CSCI-201a Introduction to Software Development	bruhhhh	edit	delete
lauren	CSCI-201a Introduction to Software Development	I really enjoyed the PAs in this class	edit	delete
lauren	CSCI-201a Introduction to Software Development	Long review.Long revie	edit	delete

5.2 Login Functionality

5.2.1 Login Page



SCearch

Create New Account

First Name

Last Name

Major

Email

Username

Password

☐ I agree to all terms and conditions.

Create New Account

Log In

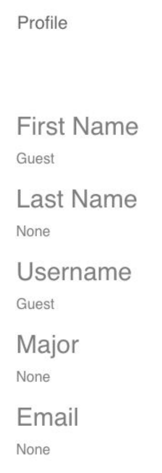
Username

Password

Log In

Join As Guest

5.2.1 Profile Page(Signed in as Guest)



Profile

First Name
Guest

Last Name
None

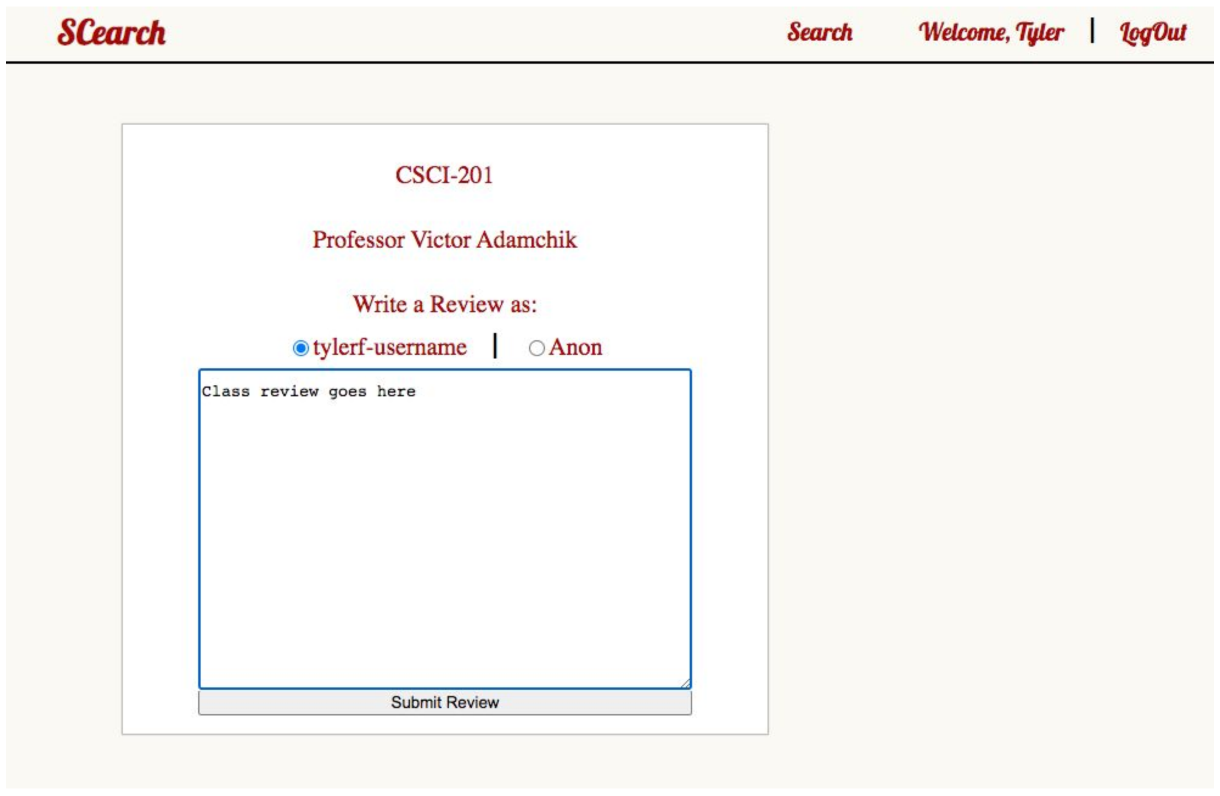
Username
Guest

Major
None

Email
None

5.3 Review Functionality

5.3.1 Uploading a Review



CSCI-201

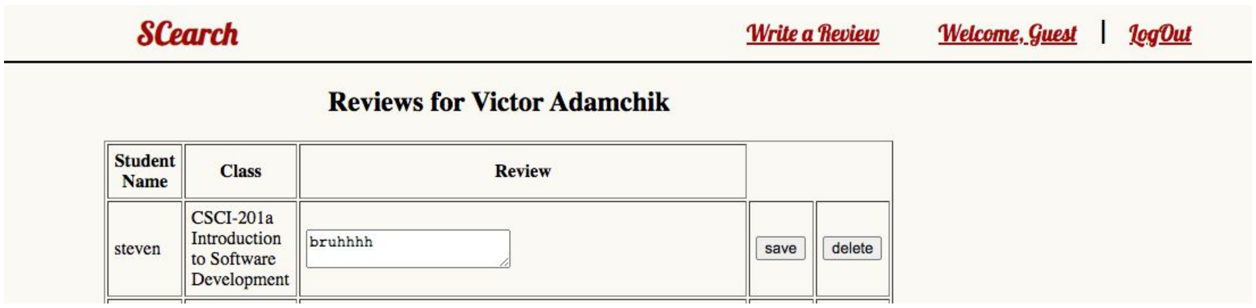
Professor Victor Adamchik

Write a Review as:

[✓ tylerf-username] | Anon

```
Class review goes here
```

5.3.2 Editing a Review/ Press delete Button



Reviews for Victor Adamchik

Student Name	Class	Review		
steven	CSCI-201a Introduction to Software Development	bruhhhh	save	delete