

Release 2 Documentation

COSC 4P02

Version 1.0

Due: Saturday April 3rd

R8Scholar

[Github](#)

Team Leader (Scrum Master):

Twino Puthiakunnel / mp13ko@brocku.ca / 5564182 / mp13ko

Product Owner:

Seth Shickluna / ss16wn@brocku.ca / 6217558 / SethShickluna

Erikas Klimusinas / ek15kg@brocku.ca / 5903547 / ErikasK

Grant Nike / gn17az@brocku.ca / 6349302 / GrantNike

James Sargent / js17sy@brocku.ca / 6380356 / shorinbonsai

Logan Bell / lb16tp@brocku.ca / 6047211 / loganbe11

Luciano Ugalde / lu16xx@brocku.ca / 6102545 / lu16xx

Munashe Masango / mm16rh@brocku.ca / 6204911 / munashemasango

Table of Contents

Table of Contents	1
Introduction:	4
Project Objective:	4
System Expectations	4
Team Expectations	4
Release:	5
Release 2 goals:	5
New Features in Release 2:	5
Iteratives:	6
User stories / Progress report / Backlog:	6
Signup	6
Email Confirmation	6
Log In	7
Sign out	7
Change password	7
Change nickname	8
Delete account	8
Password Reset	9
Create Review	9
Edit Review	10
Report Review	10
Delete Review	10
Create comments	11
Edit Comment	11
Delete Comment	11
View Instructors	12
View Departments	12
View Courses	12
Search	13
View Ratings	13
Thumbs Up / Down	14
Admin: Edit Users	14
Admin: Add Reviews, Courses, Instructors and Departments	15
Admin: Edit Reviews, Courses, Instructors and Departments	15
Admin: Delete Reviews, Courses, Instructors and Departments	15

Testing:	16
Informal Function Testing:	16
Admin Functions	16
Top bar	16
Home page	16
Sign up	16
Sign In	17
Reviews	17
Searches	17
Formal UI/Web Testing:	18
Sign Up	18
Log in	19
Log out	20
Edit Nickname	21
Change Password	21
Delete Profile	22
Courses Page:	23
Going to first	23
Going to Last	23
Opening Course	24
Opening Department	24
Sorting System	25
Instructors Page:	25
Going to first	25
Going to Last	26
Opening Instructor	26
Opening Department	27
Sorting System	27
Departments Page:	28
Going to first	28
Going to last	28
Opening Top Instructor	29
Opening Top Course	29
Opening Department	30
Sorting System:	30
Individual Course / Instructors / Departments Page	31
Create Review	32
Edit Review	33
Report Review	33
Search	34

Formal Unittest Testing:	35
Models Test Cases:	35
Test str	35
Email User	35
Has Permission	36
Has Module Permissions	36
Is Staff	36
Update Department Rating	36
Update Instructor Rating within Department	37
Update Course Rating within Department	37
Update Course Rating	37
Update Instructor Rating	37
Generators Test Case	38
Validators Test Case:	38
Validate Brock Email	38
Profanity Validator	38
Password Validator	39
Rating Validator	39
Managers Test Case:	39
Create User	39
Create Superuser	39
Database Test Case	40
Sprints / Teamwork policy:	41
Meeting 16 - Sprint Planning 5 - March 8th 2021	41
Meeting 17 - Sprint Refinement 5 - March 11th 2021	42
Meeting 18 - Sprint Planning 6 (Review 5) - March 16th 2021	43
Meeting 19 - Sprint Planning 7 (Review 6) - March 22th 2021	43
Meeting 20 - Sprint Refinement 7 - March 25th 2021	44
Meeting 21 - Sprint Planning 8 (Review 7) - March 29th 2021	44
Meeting 22 - Sprint Refinement 8 (Release 2) - April 1st 2021	45

Introduction:

COSC 4P02 R8Scholar Release 2 Documentation covering software development from March 7th 2021 to April 3rd 2021. Includes release goals/features, user stories/progress reports/backlog, formal/informal testing, sprints and teamwork policy.

Project Objective:

The objective of this project is to deliver a comprehensive experience for Brock students to be able to communicate about their experiences and to give advice/information on potential interactions they might have at the school.

System Expectations

- Online platform to rate scholarly aspects of Brock University
- Anyone should be able to view all ratings on the site
- Verified brock students should be able to leave ratings on the site
 - Signup / Login
 - Verify Brock Email
- Users must be able to edit or remove their reviews
- Users must be able to delete their account
 - All of users reviews will also be removed
- Needs to display a page for each department, course, and instructor
 - Departments can list staff and courses
 - All of these are dynamically generated, based off database entries
- Search feature on front page
 - Able to search for courses, instructors or departments
- Users must be able to comment on reviews

Team Expectations

The team will be looking to take the project on using the SCRUM methodology, the reason being that SCRUM offers an agile development and the freedom to work in concentrated teams but with a common goal. The interleaved nature of updated requirements and incrementing of software features seems just right for this type of project.

The team will attend all weekly meetings, and collaborate to develop R8Scholar.

The timing for releases/iterations of the project will be followed according.

Release:

Release 2 is focused on adding new features as well as making existing features more secure and stable.

Release 2 goals:

- Get user to verify their accounts on account creation
 - Add terms and conditions to sign up
- Increase filtering options
 - by name, by department, by rating (high or low)
- Sending out confirmation emails
- Token authentication
- Allow user to delete their account, cascade delete their reviews, and their comments
- Allow user to delete reviews, cascade delete all comments on review
- List users' reviews on their profile page
- It should be possible to edit and delete reviews
- It should be possible to view, add, edit, and report comments
- Department page should displays all the courses in the department
- API page should not be accessible publicly
- Have a profanity filter to keep site professional

New Features in Release 2:

- Limited public access to pages. e.g. api pages
- Attribute Reviews to the user (display on profile)
- Attribute Comments to the user
- Ability to edit, report, and delete Reviews
- Token-Based Authentication
- User prompts on verify (junk folder) and signup (don't use brock pass)
- Having professor to rate students (Final Release)
- Ability to search by course name i.e. "Advanced Programming"
- Form validation
- Updated password requirements
- Ability to view all user reviews in profile
- Profanity filter

Iteratives:

User stories / Progress report / Backlog:

Signup

Story: As a user, I would like to be able to sign up for a R8Scholar account.

Acceptance criteria:

1. Check to determine if the user has a valid Brock email account
 - a. If there the user provided a valid email address then check to see if the password is valid
 - i. If the password is valid allow user to create account
 - ii. If the password is invalid inform user that password is invalid
 - b. If the email is invalid the user will be informed to provide a Brock email address
2. User will have to agree to terms and conditions
3. The user will have to input a confirmation code which will be sent to their email address
4. User will be brought to login page after entering confirmation code

Progress report: Users are able to sign up for a R8Scholar account.

1. Checks if the user has provided a valid main as well as password
2. Makes user agree to terms and conditions
3. User is able to input a confirmation code to verify their account

Email Confirmation

Story: As a user, I would like to have a confirmation email sent to my email address when I'm creating my account.

Acceptance criteria:

1. Check to determine if the user has a valid Brock email account
 - a. If there the user provided a valid email address then a confirmation email will be sent to that address
 - b. If the email is invalid the user will be informed to provide a Brock email address
2. The email will contain a hyperlink which allows the user to jump instantly to the R8Scholar site

Progress report: When users create an account an confirmation email is sent with a code.

1. If the user used a valid Brock email account they will receive an email
2. The email contains a hyperlink directing to the R8Scholar site
3. The email has a code which is used for account verification

Log In

Story: As a user, I would like to be able to log in to the site using my email address and password.

Acceptance criteria:

1. Check to determine if the email address is valid
 - a. If the email address is valid check to see if the password is valid
 - i. If the password is valid allow user to log in
 - ii. If the password is invalid inform user that password is incorrect
 - b. If the email address is invalid inform user that email is invalid
2. After logging in, the user will be brought to the main homepage

Progress report: Able to log into site using Brock University email address, and their password.

1. Their email address is checked when inputted
 - a. Informs them if email address is invalid
2. After logging in they are brought to their home page

Sign out

Story: As a user, I would like to log out from the site from any page.

Acceptance criteria:

1. Display logout button on top right of site
2. Allow user to click on log out button and log out immediately
3. User should be brought to home page

Progress report:

1. Users are able to see a logout button at the top right of the site while logged in
2. Users are able to click the logout button and sign out immediately
3. User is brought to home page

Change password

Story: As a user, I would like to be able to go to my profile and change my password

Acceptance criteria:

1. Check to see if the user is logged in
 - a. If not, prompt log in (see log in story)
2. Enter current password, and enter new password
 - a. If current password is correct, the new password is accepted
3. User will be informed if password was changed or any error occurred

Progress report:

1. Profile page is only accessible when user is logged in
2. Profile page has a tab to change password
 - a. User will enter their current password
 - b. They will type in their new password twice
 - c. Password will be changed if their current password is correct and their new password matches and meets all the criteria listed
3. User is not informed if password was changed or any error occurred *backlog*

Change nickname

Story: As a user, I would like to be able to go to my profile and change my nickname

Acceptance criteria:

1. Check to see if the user is logged in
 - a. If not, prompt log in (see log in story)
2. Enter new nickname
 - a. If a nickname has profanity it will not be allowed
3. Enter current password
4. User will be informed if nickname was changed or any error occurred

Progress report:

1. Profile page is only accessible when user is logged in
2. Profile page has a tab to change nickname
 - a. User will enter their new nickname
 - b. User will type in their current password
 - c. Nickname will be changed if their current password is correct
 - d. Site does not check if new nickname contains no profanity *backlog*
3. User is not informed if nickname was changed or any error occurred *backlog*

Delete account

Story: As a user, I would like to be able to delete my account from the profile page.

Acceptance criteria:

1. Allow access to the profile page if the user is logged in
2. User should have to type in password
3. User has to agree to terms of deletion
4. Account deletion should remove all reviews and comments created by that account
5. User should be brought to home page

Progress report:

1. Profile page is only accessible when user is logged in
2. Profile page has a tab to delete profile
 - a. User will enter their current password
 - b. User must agree to terms and conditions of deletion
3. User is informed if they have not agreed to terms and conditions
4. User is not informed if password is incorrect *backlog*
5. User is brought to home page

Password Reset

Story: As a user, I would like to receive an email with a temporary password if I request a password reset.

Acceptance criteria:

1. User should be able to click a password reset button on the sign in page
2. User should be able to type in a email address and submit password reset request
 - a. If there the user provided a valid user email address then a temporary password email will be sent to that address
 - a. If the email is invalid the user will be informed to provide a Brock email address
3. The email will contain a temporary password which complies with all the password requirements for the site

Progress report:

- Password reset not implemented yet *backlog*

Create Review

Story: As a user, I would like to create a review using an input form for a specific course, instructor or department.

Acceptance criteria:

1. Only enable user review form if they are a valid user with a verified email address
2. Only allow review to be submitted if all the necessary parts of the form are filled

Progress report:

1. If user is not verified then create review tab states that account must be verified
2. User must fill in all necessary information in form for review to be submitted

Edit Review

Story: As a user, I would like to be able to edit any reviews I've created.

Acceptance criteria:

1. There should be an edit button available for any reviews created by the account
2. Clicking on the edit button should allow user to edit their review
3. There should be a "submit" or "republish" button to post the edited review
4. The user should be returned to the page they started and be able to see the updated review

Progress report:

1. User must be viewing one of their own reviews in order for the edit button to be visible
2. Only allow review to be re-submitted if all parts of the form still meet the criteria for creating a review

Report Review

Story: As a user, I would like to report reviews that may be inappropriate.

Acceptance criteria:

1. There should be an report button next to any reviews
2. Clicking on the report button should allow user to type in reason why they are reporting
3. Report should not allowed to be submitted if it contains profanity
4. Submitting the report should inform user that the report was submitted

Progress report:

1. There is a red report button to the bottom right of any review
2. Clicking on the report button allows the user to select what is wrong with the review
3. Submitting the report does not inform user that the report was submitted *backlog*

Delete Review

Story: As a user, I would like to be able to delete my own reviews.

Acceptance criteria:

1. There should be a delete button next to any of the users reviews
2. Clicking the delete button should ask user if they are sure
3. Deleting the review should remove the review and reload the page

Progress report:

- Deleting a review is not implemented yet *backlog*

Create comments

Story: As a user, I would like to create a comment under any reviews.

Acceptance criteria:

1. There should be a comment button under any review
2. Comments should not be allowed to be submitted if they contain profanity
3. Submitting should display the comment posted

Progress report:

1. Users are able to comment after clicking the view comments button
2. They are able to write their comment then click submit
3. Comments are displayed in the view comments window

Edit Comment

Story: As a user, I would like to be able to edit any comments I've posted.

Acceptance criteria:

1. There should be an edit button available for any comments by the user
2. Clicking on the edit button should allow user to edit their comment
3. There should be a "submit" button to post the edited comment
4. The user should be returned to the page they started and be able to see the updated comment
5. Edited comments should have a indicator that shows that they have been edited

Progress report:

- Edit comments not implemented yet *backlog*

Delete Comment

Story: As a user, I would like to be able to delete my own comments.

Acceptance criteria:

1. There should be a delete button next to any of the users comments
2. Clicking the delete button should ask user if they are sure
3. Deleting the comment should remove the comment and reload the page

Progress report:

- Delete comment not implemented yet *backlog*

View Instructors

Story: As a user, I would like to be able to view a list of instructors by clicking on the “instructors” tab on the top menu bar.

Acceptance criteria: As a user, when I select the instructor tab, it will take me to the instructor page, which will then have a list of instructors that can be further filtered. It will also have instructor rating rank, rating, department the instructor belongs to and the departments rating.

Progress report: Users are able to select the instructors tab and view a list of all instructors. Users are able to view instructors rating rank, ratings as well as their department and department rating rank.

View Departments

Story: As a user, I would like to be able to view a list of departments that the university offers.

Acceptance criteria: As a user, when I select the department tab, it will take me to the department page, which will then have a list of departments that can be further filtered. It will also display department rating rank, ratings, the top course, and the top instructor.

Progress report: Users are able to select the departments tab and view a list of all departments. Users are able to view department rating rank, ratings as well as top course and top instructor.

View Courses

Story: As a user, I would like to be able to view a list of available courses by clicking on the “Courses” tab on the top menu bar.

Acceptance criteria: As a user, when I select the courses tab, it will take me to the courses page, which will then have a list of courses. It will have the course rating rank, course code, course name, rating, and department of the course.

Progress report: Users are able to select the courses tab and see a list of all courses. Users are able to see the course rating rank, course code, course name, rating and the department of the course.

Search

Story: As a user, I would like to be able to use a search feature to specifically find the information that I want in an efficient manner.

Acceptance criteria: As a user, I will be able to input what I am searching for into the search bar, and the system will return all matching results.

Progress report: Users are able to input what they want to search into the search bar. The system returns all matching results. The results can be departments, instructors or courses.

View Ratings

Story: As a user, I would like to view ratings of courses, departments and instructors.

Acceptance criteria:

1. Users must be able to top rated courses, instructors and departments on the homepage
2. Users must be able to see ratings in the all courses, all instructors and all departments lists
 - a. Users must be able to filter ratings by alphabetical or high to low and vice versa
3. Users must be able to view ratings on individual course, instructor, and department pages

Progress report:

1. Users are able to view the top 5 rated courses, instructors and departments on their homepage
2. Users are able to see ratings in course, instructors, and departments lists
 - a. Users are able to filter ratings by alphabetical or high to low and vice versa
3. Users can view ratings on individual course, instructor, and department pages

Thumbs Up / Down

Story: As a user, I would like to be able to thumbs up good reviews and comments.

Acceptance criteria:

1. There must be a thumbs up and down button next to reviews
2. There must be a number next to the thumbs up and down button indicating the number of thumbs up / down a review has received
3. User must be able to click thumbs up button
 - a. Increase the thumbs number by 1
 - b. Highlight the thumbs up button to indicate that thumbs up has been applied
 - c. If a thumbs down had been previously applied, it will be deactivated and thumbs number will increase by 1
4. User must be able to remove their thumbs up by clicking on an highlighted one
 - a. Decrease the thumbs number by 1
 - b. Remove highlight on thumbs up button to indicate that thumbs up has been removed
5. User must be able to click thumbs down button
 - a. Decrease the thumbs number by 1
 - b. Highlight the thumbs down button to indicate that thumbs down has been applied
 - c. If a thumbs up had been previously applied, it will be deactivated and thumbs number will decrease by 1
6. User must be able to remove their thumbs down by clicking on an highlighted one
 - a. Increase the thumbs number by 1
 - b. Remove highlight on thumbs down button to indicate that thumbs down has been removed

Progress report:

- Thumbs up / down has not been implemented yet *backlog*

Admin: Edit Users

Story: As an admin, I would like to be able to change user email addresses, change user nicknames, and give users administrative privileges.

Acceptance criteria:

1. There must be a tab to edit users
2. There must be a form to change email, nickname or permissions
3. There must be a means to save changes

Progress report:

1. Admins can use the custom user tab in Django administration
2. Admins are input emails, nicknames or permissions
3. Admins are able to save changes

Admin: Add Reviews, Courses, Instructors and Departments

Story: As an admin, I would like to be able to add review, course, instructor and department information.

Acceptance criteria:

1. There must be tabs for users, courses, instructors and departments
2. There must be a button to add users, courses, instructors and departments

Progress report:

1. Admins can use the users, courses, instructors and departments tabs in Django administration
2. Admins can choose any tab: user, courses, instructors and departments and add

Admin: Edit Reviews, Courses, Instructors and Departments

Story: As an admin, I would like to be able to change review, course, instructor and department information.

Acceptance criteria:

1. There must be a tab to edit reviews, courses, instructors and departments
2. There must be a form to edit their respective information
3. There must be a means to save changes

Progress report:

1. Admins can use the reviews, courses, instructors and departments tabs in Django administration
2. Admins can input any changes they wish to make
3. Admins are able to save changes

Admin: Delete Reviews, Courses, Instructors and Departments

Story: As an admin, I would like to be able to delete review, course, instructor and department information.

Acceptance criteria:

1. There must be tabs for reviews, courses, instructors and departments
2. There must be a checkbox to select and an action to delete

Progress report:

1. Admins can use the reviews, courses, instructors and departments tabs in Django administration
2. Admins can choose any of the reviews, courses, instructors and departments and delete them

Testing:

Informal Function Testing:

Admin Functions

Issue:

- Unexpected password error prevents saving. Users cannot be modified due to this. Field length seems to be a recurring issue. Also occurs with departments and courses, while instructors seem to be fine

Expected fix: Correct maximum field length sizes

Issue:

- Some instructors appear as “merged” with other instructors, such as 2 names being treated as one instructor.

Expected fix: Properly separate instructors with a line break

Top bar

Issue:

- The bar flickers before displaying properly
- At half size, top bar is shifted to the right side of the screen

Expected fixes:

- Burger menu opens on the left side listing out courses, instructors, department, signup, and login.

Issue: When clicking the R8 SCHOLAR button from the main page it opens a new tab.

Expected fix: R8 SCHOLAR button press results in refresh of page

Minor issue: YouTube icon/button on top bar leads to twitter account

Home page

Issue:

- Stars are easily mis-aligned, from large course/department/instructor names or window size.

Expected fix: Modify the scaling

Sign up

Issue:

- When creating an account, it can be very unclear what the user is doing wrong in the event of an error.

Expected fix:

- There should be a way to let the user know whether they have an email error, a username error, a password error, and what the error is, whether by a help button or tooltips, or more descriptive/specific error messages.

Sign In

Minor issue:

- After entering in the email and password, pressing the enter key doesn't do anything.

Expected fix:

- Pressing enter should have the same effect as pressing the Sign in button.

Reviews

Minor issue:

- Users should only be able to edit their review for the course.

Expected fix:

- “Create Review” should change to “Edit Review”

Issue:

- When you change your nickname it doesn't update the old reviews with the new nickname.

Expected fix:

- Old reviews should be having your current nickname.

Issue:

- Clicking “VIEW FULL REVIEW” does nothing

Expected fix:

- Add functionality

Searches

Minor issue:

- Queries sometimes glitch out when typed in too fast, and the display doesn't update in time

Issue:

- When clicking on the highlighted area when selecting, selection does not occur and the area stays dark even after the cursor is not on the search results.

Expected fix:

- Clicking on the highlighted area of the search result should be the same as clicking on the result word itself.

Issue:

- Clicking on the area directly around the search bar results in a visible outline (not the same as the outline you get when clicking on the actual search bar.



Expected fix:

- It should only be possible to click inside the search bar.

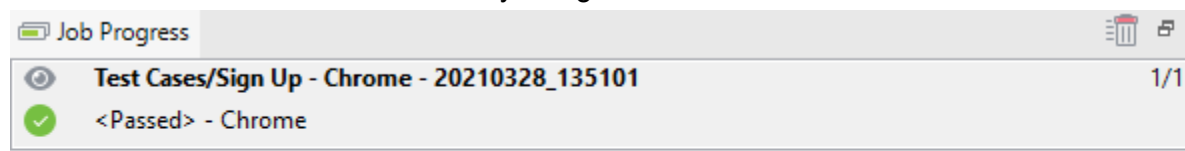
Formal UI/Web Testing:

Formal UI/Web testing done using Katalon Studio, an all-in-one test automation solution. Tested for web, using Google Chrome web browser as testbed with automated tests generated using Katalon Studio. Functionality testing was used for link testing, form validation, cookie testing, and database connection checkup. Performance testing was done by simulating multiple users using functions at the same time via automation and multiple Katalon runtimes. Microsoft Edge was also used in conjunction with Google Chrome for compatibility testing.

Sign Up

Use Case Name	Sign Up / Create Profile
Trigger	The user clicks the “sign up” button on the website
Precondition	The user has not created a profile for their email address (not in use)
Basic Path	<ol style="list-style-type: none"> 1. The system opens a form to enter information 2. The user will enter the correct format of data into the form 3. The system confirms the information with the database and the user 4. The profile is created and the user redirected to the profile page
Alternative Path	None noted
Postcondition	User is able to log in with the specified email and password
Exception Paths	If the user has already created a Profile using that given email address then the use case abandons
Other (Notes)	This can only be used for each email address once

The first test case was run successfully using a test email:



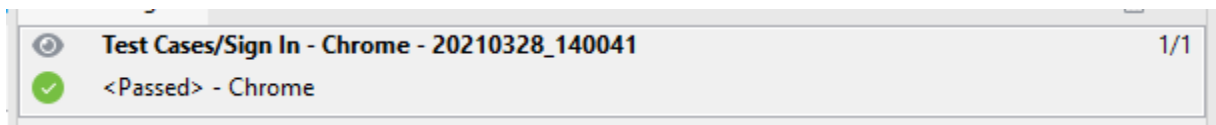
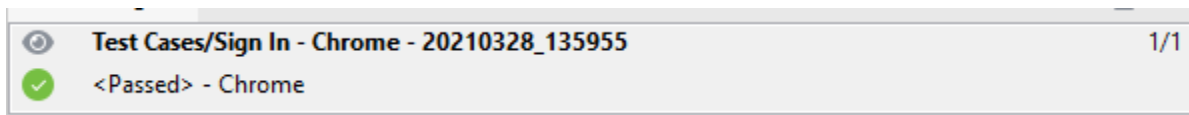
The second test case was also successful using the same test email used previously, and followed the exception path of abandoning the sign up, as a profile was already created with the email.



Log in

Use Case Name	Log in to profile
Trigger	The user selects the “log in” on the website
Precondition	The user has created a profile previously using a defined email address
Basic Path	<ol style="list-style-type: none"> 1. The system opens a log in dialogue 2. The user inputs the correct format of data (text) 3. The database authenticates whether that user is allowed access 4. The user is redirected to the
Alternative Path	There is the possibility of being directed to the login via a direct link
Postcondition	The user is redirected and is logged in
Exception Paths	If the user inputs invalid information there will be an error and it will allow you to re-enter the text.

Test case was run twice - one with a valid email and once without. Both cases were handled correctly, and the user was able to log in on the first one, and the user was prompted with an “invalid email” response on the second one.



Log out





Use Case Name	Log Out
Trigger	The user selects the log out option on the website
Precondition	The user was already logged in and accepted into the database.
Basic Path	<ol style="list-style-type: none"> 1. The system prompts the user to make sure they want to log out 2. The system will redirect the user to the homepage of the website
Alternative Path	None noted so far
Postcondition	The user is no longer logged in and able to write reviews
Exception Paths	The user decides against logging out of the website and thus the use case is abandoned.

While testing, found an exception where when not fullscreened, the sign out button would not be visible. Temporary fix was solved by maximizing the window in line 30, but will need to be corrected.

```

20 WebUI.openBrowser('')
21
22 WebUI.navigateToUrl('http://localhost:8000/login')
23
24 WebUI.setText(findTestObject('Object Repository/Page_BrockU R8Scholar/input_Email address_formGroupEmail'), 'Password
25
26 WebUI.setEncryptedText(findTestObject('Object Repository/Page_BrockU R8Scholar/input_Password_formGroupPassword'), 'p
27
28 WebUI.click(findTestObject('Object Repository/Page_BrockU R8Scholar/button_Sign In'))
29
30 WebUI.maximizeWindow()
31
32 WebUI.click(findTestObject('Object Repository/Page_BrockU R8Scholar/a_Sign Out'))
33
34 WebUI.closeBrowser()


```

	Test Cases/Log Out - Chrome - 20210328_140819	1/1
	<Passed> - Chrome	
	Test Cases/Log Out - Chrome - 20210328_140717	1/1
	<Failed> - Chrome	

Edit Nickname

Use Case Name	Edit Nickname
Trigger	The user selects the Edit feature within the profile.
Precondition	The user must be authorised, be viewing the profile page they created.
Basic Path	<ol style="list-style-type: none"> 1. The profile information form is presented in an editable view 2. The user edits the desired fields in order make changes 3. The system confirms before making any changes to the profile.
Alternative Path	None noted
Postcondition	The nickname changes successfully
Exception Paths	If the user updates the information with invalid content it will prompt them to fix it. If the user cancels the changes, there are no new changes and the use case is abandoned.


Username changes were tested successfully.

Test Cases/Edit Profile - Chrome - 20210328_141704		1/1
	<Passed> - Chrome	

Change Password

Use Case Name	Edit Password
Trigger	The user selects the Edit feature within the profile.
Precondition	The user must be authorised, be viewing the profile page they created.
Basic Path	<ol style="list-style-type: none"> 1. The profile information form is presented in an editable view 2. The user edits the desired fields in order make changes 3. The system confirms before making any changes to the profile.
Alternative Path	None noted
Postcondition	The user is able to use the new password to log in
Exception Paths	If the user updates the information with invalid content it will prompt them to fix it. If the user cancels the changes, there are no new changes and the use case is abandoned.


Password changing was tested successfully, with both an incorrect and correct password.

Test Cases/Edit Profile - Chrome - 20210328_141928		1/1
	<Passed> - Chrome	


Delete Profile

Use Case Name	Delete Profile
Trigger	The user clicks delete profile button after accepting terms and conditions and entering their current password
Precondition	The user must be logged in.
Basic Path	<ol style="list-style-type: none"> 1. User clicks on "PROFILE" button on top bar 2. User clicks on "Delete Profile" tab 3. User enters their current password 4. User accepts terms and conditions
Alternative Path	None noted
Postcondition	Account and all associated reviews and comments will be deleted
Exception Paths	If user inputs invalid password or does not accept terms and conditions then the profile will not be deleted, and user will be informed


The case where user inputs correct password and accepts terms and conditions:

 Test Cases/Delete Profile - Chrome - 20210402_204807 <Passed> - Chrome	1/1
--	-----

The case where user inputs invalid password:

 Test Cases/Delete Profile/Invalid Password - Chrome - 20210402_205202 <Passed> - Chrome	1/1
---	-----

The case where user does not accept terms and conditions:

 Test Cases/Delete Profile/Unacceptable Conditons - Chrome - 20210402_205606 <Passed> - Chrome	1/1
---	-----

Courses Page:

Going to first

Use Case Name	Going to first
Trigger	The user clicks on the first button on the Courses page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if first button keeps you on first page when on first page • Checks if first button brings you on first page when on second page • Checks if first button brings you to first page when on a page far away from first page • Checks that clicking on 1 when not on first page brings you to the same page as first page
Postcondition	The user is brought to the first page
Other (Notes)	Checks all ways of getting to first page

Test Cases/courses/going to first - Chrome - 20210402_201452

1/1



<Passed> - Chrome

Going to Last

Use Case Name	Going to last
Trigger	The user clicks on the last button on the Courses page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if last button brings you on last page when on first page • Checks if last button brings you on last page when on second last page • Checks if last button keeps you to last page when on the last page • Checks that clicking on the last listed number when on the second last page brings you to the same page as the last page
Postcondition	The user is brought to the last page
Other (Notes)	Checks all ways of getting to last page



Test Cases/courses/going to last - Chrome - 20210402_202725

1/1



<Passed> - Chrome

Opening Course

Use Case Name	Opening Course
Trigger	The user clicks on a course listed one the Courses page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on courses 3. User clicks on course name 4. User brought to individual course page which is the same as the course name clicked on
Alternative Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on courses 3. User clicks on course code 4. User brought to individual course page which is the same as the course code clicked on
Postcondition	The user is brought to the last page
Other (Notes)	Checks all ways of getting to last page

Test Cases/courses/opening course - Chrome - 20210402_201514

1/1



<Passed> - Chrome

Opening Department

Use Case Name	Opening Department
Trigger	The user clicks on a Department listed one the Courses page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on courses 3. User clicks on department for a course 4. User brought to individual department page which matches department clicked on
Postcondition	The user is brought to individual department page

**Test Cases/courses/opening department - Chrome - 20210402_203334**



1/1



<Passed> - Chrome

Sorting System


Use Case Name	Sorting System
Trigger	The user clicks on the sorting options for Courses
Precondition	Sorting is currently on A-Z
Basic Path	<ul style="list-style-type: none"> • Checks if Sorting by Z-A causes a difference • Clicks on highest to lowest rating to sorts by highest rating • Sorts by lowest to highest rating to sort by lowest rating first
Postcondition	The ratings are sorted based on the selected sorting option

 **Test Cases/courses/sorting system - Chrome - 20210402_203032** 1/1
 <Passed> - Chrome

Instructors Page:



Going to first

Use Case Name	Going to first
Trigger	The user clicks on the first button on the Instructors page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if first button keeps you on first page when on first page • Checks if first button brings you on first page when on second page • Checks if first button brings you to first page when on a page far away from first page • Checks that clicking on 1 when not on first page brings you to the same page as first page
Postcondition	The user is brought to the first page
Other (Notes)	Checks all ways of getting to first page

Test Cases/instructors/Going to first - Chrome - 20210402_203540 1/1
 <Passed> - Chrome


Going to Last

Use Case Name	Going to last
Trigger	The user clicks on the last button on the Instructors page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if last button brings you on last page when on first page • Checks if last button brings you on last page when on second last page • Checks if last button keeps you to last page when on the last page • Checks that clicking on the last listed number when on the second last page brings you to the same page as the last page
Postcondition	The user is brought to the last page
Other (Notes)	Checks all ways of getting to last page

 **Test Cases/instructors/Going to last - Chrome - 20210402_204047** 1/1
 <Passed> - Chrome

Opening Instructor

Use Case Name	Opening Instructor
Trigger	The user clicks on a instructor listed one the Instructors page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on instructors 3. User clicks on instructor's name 4. User brought to individual instructor page which is the same as the instructor's name clicked on
Alternative Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on instructors 3. User changes instructors page 4. User clicks on instructor's name 5. User brought to individual instructor's page which is the same as the instructor's name clicked on
Postcondition	The user is brought to the individual instructor's page

Test Cases/instructors/opening instructor - Chrome - 20210402_203557 1/1
 <Passed> - Chrome

Opening Department

Use Case Name	Opening Department
Trigger	The user clicks on a department listed one the Instructors page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on instructors 3. User clicks on instructor's department 4. User brought to individual department page which is the same as the department name clicked on
Alternative Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on instructors 3. User changes instructors page 4. User clicks on instructor's department 5. User brought to individual department page which is the same as the instructor's department clicked on
Postcondition	The user is brought to the individual department page



Test Cases/instructors/opening departments - Chrome - 20210402_204153

1/1



<Passed> - Chrome

Sorting System

Use Case Name	Sorting System
Trigger	The user clicks on the sorting options for instructors
Precondition	Sorting is currently on A-Z
Basic Path	<ul style="list-style-type: none"> • Checks if Sorting by Z-A causes a difference • Clicks on highest to lowest rating to sorts by highest rating • Sorts by lowest to highest rating to sort by lowest rating first
Postcondition	The ratings are sorted based on the selected sorting option



Test Cases/instructors/Sorting systems - Chrome - 20210402_204324

1/1



<Passed> - Chrome

Departments Page:

Going to first

Use Case Name	Going to first
Trigger	The user clicks on the first button on the departments page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if first button keeps you on first page when on first page • Checks if first button brings you on first page when on second page • Checks if first button brings you to first page when on a page far away from first page • Checks that clicking on 1 when not on first page brings you to the same page as first page
Postcondition	The user is brought to the first page
Other (Notes)	Checks all ways of getting to first page

Test Cases/departments/Going to first - Chrome - 20210402_204430

1/1



<Passed> - Chrome

Going to last

Use Case Name	Going to last
Trigger	The user clicks on the last button on the departments page
Precondition	None
Basic Path	<ul style="list-style-type: none"> • Checks if last button brings you on last page when on first page • Checks if last button brings you on last page when on second last page • Checks if last button keeps you to last page when on the last page • Checks that clicking on the last listed number when on the second last page brings you to the same page as the last page
Postcondition	The user is brought to the last page
Other (Notes)	Checks all ways of getting to last page

Test Cases/departments/going to last - Chrome - 20210402_204436

1/1



<Passed> - Chrome

Opening Top Instructor

Use Case Name	Opening top Instructor
Trigger	The user clicks on a instructor listed one the Instructors page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on departments 3. User clicks on top instructor for department 4. User brought to individual instructor page which is the same as the instructor's name clicked on
Alternative Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on departments 3. User clicks on department that doesn't have any instructors 4. User clicks on No Data 5. User brought to page that can return user to instructors page
Postcondition	The user is brought to the individual instructor's page if it exists
Other (Notes)	A department with no instructors will have a top instructor of "No Data"

Test Cases/departments/opening top instructor - Chrome - 20210402_204459

1/1



<Passed> - Chrome

Opening Top Course

Use Case Name	Opening top Course
Trigger	The user clicks on a top Course listed one the Departments page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on departments 3. User clicks on top courses name 4. User brought to individual course page which is the same as the courses name clicked on
Postcondition	The user is brought to the individual instructor's page



Test Cases/departments/opening top course - Chrome - 20210402_205138

1/1



<Passed> - Chrome

Opening Department

Use Case Name	Opening Department
Trigger	The user clicks on a department listed one the departments page
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User goes to home page 2. User clicks on departments 3. User clicks on department name 4. User brought to individual department page which is the same as the department name clicked on
Postcondition	The user is brought to the individual department page

Test Cases/departments/opening departments - Chrome - 20210402_204443

1/1



<Passed> - Chrome

Sorting System:

Use Case Name	Sorting System
Trigger	The user clicks on the sorting options for departments
Precondition	Sorting is currently on A-Z
Basic Path	<ul style="list-style-type: none"> • Checks if Sorting by Z-A causes a difference • Clicks on highest to lowest rating to sorts by highest rating • Sorts by lowest to highest rating to sort by lowest rating first
Postcondition	The ratings are sorted based on the selected sorting option



Test Cases/departments/sorting system - Chrome - 20210402_205306

1/1

<Passed> - Chrome

Individual Course / Instructors / Departments Page

Individual Course Page:

Use Case Name	Individual Course testing
Trigger	The user clicks to go to an individual Course Page
Precondition	Course Exists
Basic Path	<ul style="list-style-type: none"> • Checks if name matches course name listed on courses • Checks if course code matches course code listed on courses • Checks if department listed brings you expected individual department
Postcondition	All links work as intended


Test Cases/Individual course/individual course testing - Chrome - 20210402_205423
1/1


 <Passed> - Chrome

Individual Instructors Page:

Use Case Name	Individual Instructor testing
Trigger	The user clicks to go to an individual Instructors Page
Precondition	Instructor Exists
Basic Path	<ul style="list-style-type: none"> • Checks if instructors name matches instructors name listed on instructors • Checks if popular instructors brings you to expected individual instructors pages • Checks if popular courses brings you to expected individual courses pages • Checks if department listed brings you expected individual department
Postcondition	All links work as intended


Test Cases/individual instructor/individual instructor testing - Chrome - 20210402_205617
1/1


 <Passed> - Chrome

Individual Departments Page:

Use Case Name	Individual department testing
Trigger	The user clicks to go to an individual Departments Page
Precondition	Department Exists
Basic Path	<ul style="list-style-type: none"> • Checks if department name matches department name listed • Checks if popular instructors brings you to expected individual instructors pages • If there are no instructors for the department, checks that no instructors are listed • Checks if popular courses goes to expected individual course
Postcondition	All links work as intended
Other (Notes)	Departments are known to have no instructors if there is no top instructor listed on department page (will have "No Data" in its place)

Test Cases/individual department/individual department testing - Chrome - 20210402_205437

1/1



<Passed> - Chrome

Create Review

Use Case Name	Create Review
Trigger	Goes to create review tab
Precondition	Signed in user on specific course, instructor or department page
Basic Path	<ol style="list-style-type: none"> 1. User selects "Create Review" tab 2. User inputs a Review Title 3. User selects a rating for each category listed 4. User writes down what they think about the respective course, instructor or department 5. User clicks on "SUBMIT" button to submit review
Alternative Path	None noted
Postcondition	Review will be published and publicly available
Exception Paths	If title and detail are not included their review will not be submitted





Test Cases/Review/Create Review - Chrome - 20210403_002613

1/1

<Passed> - Chrome



Edit Review

Use Case Name	Edit Review
Trigger	User wishes to edit their review
Precondition	The user must be logged in and on a specific course, instructor, department page, or their profile
Basic Path	<ol style="list-style-type: none"> 1. User selects "Edit" button on their review 2. User edits their Review Title if they want to 3. User changes a rating for each category listed if they want to 4. User edits what they think about the respective course, instructor or department if they want to 5. User clicks on "SUBMIT" button to submit edited review
Alternative Path	None noted
Postcondition	Edited Review will be published and publicly available
Exception Paths	If review has no title and detail, their review will not be submitted

	Test Cases/Review/Edit Review - Chrome - 20210403_003606	1/1
	<Passed> - Chrome	

Report Review



Use Case Name	Report Review
Trigger	User wishes to report a review
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User clicks on "REPORT" button on review 2. User selects what's wrong with review 3. User enters explanation if they choose to 4. User clicks "REPORT" button to submit
Alternative Path	None noted
Postcondition	User will return to page viewing review
Exception Paths	If user makes no selection clicking submit will return user to review

	Test Cases/Review/Report Review - Chrome - 20210403_004643	1/1
	<Passed> - Chrome	



Search

Use Case Name	Search
Trigger	User starts a search
Precondition	None
Basic Path	<ol style="list-style-type: none"> 1. User clicks on R8Scholar search bar 2. User types desired course, instructor or department name or code 3. User presses enter 4. User clicks on their chosen course, instructor or department 5. User is brought to their chosen course, instructor or department page
Alternative Path	None noted
Postcondition	User will be on their chosen course, instructor or department page
Exception Paths	If user inputs name or code incorrectly they will not be able to reach their desired course, instructor or department page



Tests searching course names and course codes:

	Test Cases/Search/Search Courses - Chrome - 20210403_022618	1/1
	<Passed> - Chrome	

Tests searching instructor names:

	Test Cases/Search/Search Instructors - Chrome - 20210403_022700	1/1
	<Passed> - Chrome	

Tests searching department names:

	Test Cases/Search/Search Departments - Chrome - 20210403_022740	1/1
	<Passed> - Chrome	

Formal Unittest Testing:

Formal Unit testing for python/Django framework of R8Scholar was accomplished using unittest unit testing framework. Unittest was used for test automation by creating a test suite with test cases covering a range of functions in the python/Django framework. A test runner is used to orchestrate the tests and to provide a text interface indicating the results from executing the tests.

Models Test Cases:

Imports:

```
from django.test import TestCase
from api.models import CustomUser
from api.models import Department
from api.models import Instructor
from api.models import Course
from api.models import Review
```

Setup:

```
def setUp(self):
    CustomUser.objects.create(email="lb16tp@brocku.ca",nickname="logan",password="G*Zfb2UcV6l0")
    CustomUser.objects.create(email="lb16tp2@brocku.ca",nickname="logan2",password="z^!Zej$d%8hO",is_admin=True)
    Department.objects.create(name="departmenttest",courses_rating=3.555,instructors_rating=2.444,rating=3.00)
    Dept=Department.objects.get(name="departmenttest")
    Cuser=CustomUser.objects.get(email="lb16tp@brocku.ca")
    Instructor.objects.create(name="instructortest",department=Dept,rating=3.44,)
    Inst=Instructor.objects.get(name="instructortest")
    Course.objects.create(name="coursetest",department=Dept,rating=4.22,course_full_name="fullnametest")
    Cor=Course.objects.get(name="coursetest")

    Review.objects.create(reviewer=Cuser,nickname="testnick",subject="tester",title="test",rating=2.5,department_name=Dept,instructor_name=Inst,course_name=Cor)
```

Test str

Test code:

```
def test_str(self):
    CUser=CustomUser.objects.get(email="lb16tp@brocku.ca")
    self.assertEqual(CUser._str_(), 'lb16tp@brocku.ca')
```

Results:

```
TestCase_test_str (__main__.ModelsTestCase) ... ok
```

Email User

Test code:

```
def test_email_user(self):
    CUser=CustomUser.objects.get(email="lb16tp@brocku.ca")
```

Results:

```
TestCase_test_email_user (__main__.ModelsTestCase) ... ok
```

Has Permission

Test code:

```
def test_has_perm(self):
    CUser=CustomUser.objects.get(email="lb16tp@brocku.ca")
    self.assertTrue(CUser.has_perm(CUser.is_active))
```

Results:

```
TestCase_test_has_perm (__main__.ModelsTestCase) ... ok
```

Has Module Permissions

Test code:

```
def test_has_module_perms(self):
    CUser=CustomUser.objects.get(email="lb16tp@brocku.ca")
    self.assertTrue(CUser.has_module_perms(CUser.is_active))
```

Results:

```
TestCase_test_has_module_perms (__main__.ModelsTestCase) ... ok
```

Is Staff

Test code:

```
def test_is_staff(self):
    CUser=CustomUser.objects.get(email="lb16tp@brocku.ca")
    CUser2=CustomUser.objects.get(email="lb16tp2@brocku.ca")
    self.assertFalse(CUser.is_staff)
    self.assertTrue(CUser2.is_staff)
```

Results:

```
TestCase_test_is_staff (__main__.ModelsTestCase) ... ok
```

Update Department Rating

Test code:

```
def test_dept_update_rating(self):
    Dept=Department.objects.get(name="departmenttest")
    Dept.update_rating()
    self.assertEqual(Dept.rating,2.5)
```

Results:

```
TestCase_test_dept-update (__main__.ModelsTestCase) ... ok
```

Update Instructor Rating within Department

Test code:

```
def test_update_instructor_rating(self):
    Dept=Department.objects.get(name="departmenttest")
    Dept.update_instructor_rating()
    self.assertEqual(Dept.instructors_rating,2.5)
```

Results:

```
TestCase_test_update_instructor_rating (__main__.ModelsTestCase) ... ok
```

Update Course Rating within Department

Test code:

```
def test_update_course_rating(self):
    Dept=Department.objects.get(name="departmenttest")
    Dept.update_course_rating()
    self.assertEqual(Dept.courses_rating,2.5)
```

Results:

```
TestCase_test_update_course_rating (__main__.ModelsTestCase) ... ok
```

Update Course Rating

Test code:

```
def test_course_update_rating(self):
    Cor=Course.objects.get(name="coursetest")
    Cor.update_rating()
    self.assertEqual(Cor.rating,2.5)
```

Results:

```
TestCase_test_course_update_rating (__main__.ModelsTestCase) ... ok
```

Update Instructor Rating

Test code:

```
def test_instructor_update_rating(self):
    Inst=Instructor.objects.get(name="instructortest")
    Inst.update_rating()
    self.assertEqual(Inst.rating,2.5)
```

Results:

```
TestCase_test_instructor_update_rating (__main__.ModelsTestCase) ... ok
```

Generators Test Case

Imports:

```
from django.test import TestCase
from api.generators import generate_validation_code
```

Test code:

```
class generatorsTestCase(TestCase):

    def test_generate_validation_code(self):
        self.assertEqual(len(generate_validation_code(6)),6)
        self.assertEqual(len(generate_validation_code(8)),8)
        self.assertEqual(len(generate_validation_code()),6)
```

Results:

```
TestCase.test_generate_validation_code (__main__.GeneratorsTestCase) ... ok
```

Validators Test Case:

Imports:

```
from django.test import TestCase
from api.validators import validate_brock_mail
from api.validators import profanity_validator
from api.validators import rating_validator
from django.core.exceptions import ValidationError
```

Validate Brock Email

Test code:

```
def test_validate_brock_mail(self):
    mail="lb16tp@brocku.ca"
    fakemail="somemail@notgood.ca"
    self.assertEqual(validate_brock_mail(mail), 'lb16tp@brocku.ca')
    with self.assertRaises(ValidationError):
        validate_brock_mail(fakemail)
```

Results:

```
TestCase.test_validate_brock_mail (__main__.ValidatorsTestCase) ... ok
```

Profanity Validator

Test code:

```
def test_profanity_validator(self):
    goodword="good"
    badword="nasty"
    self.assertEqual(profanity_validator(goodword), "good")
    with self.assertRaises(ValidationError):
        profanity_validator(badword)
```

Results:

```
TestCase.test_profanity_validator (__main__.ValidatorsTestCase) ... ok
```

Password Validator

Test code:

```
def password_validator(self):
    strongpassword="This#isAstrongpassword!132"
    weakpassword="thisisnt"
    weakpassword2="neitheristhispassword"
    weakpassword3="Neitheristhispassword2"
    self.assertEqual(password_validator(strongpassword),"This#isAstrongpassword!132")
    self.assertNotEqual(password_validator(weakpassword),"thisisnt")
    self.assertNotEqual(password_validator(weakpassword1),"neitheristhispassword")
    self.assertNotEqual(password_validator(weakpassword2),"Neitheristhispassword2")
```

Results:

```
TestCase_test_password_validator (__main__.ValidatorsTestCase) ... ok
```

Rating Validator

Test code:

```
def rating_validator(self):
    rating=3.5
    rating2=5.5
    self.assertEqual(rating_validator(rating),3.5)
    self.assertNotEqual(rating_validator(rating2),5.5)
```

Results:

```
TestCase_test_rating_validator (__main__.ValidatorsTestCase) ... ok
```

Managers Test Case:

Create User

Test code:

```
def test_create_user(self):
    Cuser = CustomUser.objects.create_user("somemail@brocku.ca","nickname23","Asecurepassword##234")
    self.assertEqual(Cuser.nickname, 'nickname23')
    self.assertEqual(Cuser.email,'somemail@brocku.ca')
    self.assertFalse(Cuser.is_admin)
```

Results:

```
TestCase_test_create_user (__main__.ValidatorsTestCase) ... ok
```

Create Superuser

Test code:

```
def test_create_superuser(self):
    Cuser = CustomUser.objects.create_superuser("someothermail@brocku.ca","nickname2345","Asecurepassword##2345")
    self.assertEqual(Cuser.nickname, 'nickname2345')
    self.assertEqual(Cuser.email,'someothermail@brocku.ca')
    self.assertTrue(Cuser.is_admin)
```

Results:

```
TestCase_test_create_superuser (__main__.ValidatorsTestCase) ... ok
```


Database Test Case

Imports:

```
from django.test import TestCase
from api.models import Instructor, Course, Department
from api.db_entries import ModelGenerator
```

Test code:

```
class db_entriesTestCase(TestCase):
    def test_Department_db_entries(self):
        ModelGenerator('./api/data/departments.txt', "Department")
        self.assertTrue(True)
        ModelGenerator('./api/data/instructors.txt', "Instructor")
        self.assertTrue(True)
        ModelGenerator('./api/data/courses.txt', "Course")
        self.assertTrue(True)
```

Results:

```
TestCase.test_Department_db_entries (__main__.ModelGeneratorTestCase) ... ok
```

```
-----
Ran 16 tests in 20.527s

OK
Destroying test database for alias 'default'...
```

Sprints / Teamwork policy:

Meeting 16 - Sprint Planning 5 - March 8th 2021

Objectives/Backlog:

- Get user to verify their account as soon as they create
- Add terms and conditions to sign up
- Fix text that's hard to read on some pages
- Change Edit Profile into Change Nickname and Change Password
- Implement searching using api calls
- filtering implementation (user supplied)
 - by name
 - department
 - rating (high or low)
- Finish scraping list of instructors
- (later) Email confirmation

Edit profile page:

1. Change username and change password should be separate.
2. Shouldn't need to ever change email.
3. Terms and conditions should be agreed to when the user signs up.
4. Github and youtube links not fixed

Create Review page(Course,Department,Instructor):

1. Wording is redundant.
2. Shouldn't be asking to rate instructors in course reviews and courses in instructor reviews.

Sign Up Page:

1. Doesn't prompt users to validate their account after signing up.
2. Doesn't login after they create an account.
3. Terms and conditions should be agreed to when the user signs up.

Misc:

- Some fonts can't be seen well(example:the home page)
- Nav bar icons could be larger

Progress:

- Grant: Fixed top instructors, worked on review and email confirmation
- James: Worked on search implementation and demonstrated it during sprint refinement
- Seth: Tested out filtering results in the backend and shared his conclusions
- Logan: Worked on setting up the necessary data of which instructors are in which courses

Things different compared to release 1

- API pages should not be accessible openly

Meeting 17 - Sprint Refinement 5 - March 11th 2021

- Everyone attended the meeting
- Grant
 - Documenting testing, (issues) on Release 2 document
 - Encouraged more communication
- James
 - review, comment linked with users (after the demo)
 - displaying user's reviews
 - Issue with current api being exposed (after the demo)
- Logan
 - Has to finish up instructors db
 - Bug: Z-A, on top of A-Z
- Twino demonstrated inconsistency with top bar
 - Switching pages had issues
 - Nickname changed, but page needs to be refreshed
 - "First" button causes page to be blank after changing multiple pages
 - Mobile (optional)
- Seth
 - Authentication token (after the demo)
 - Report and auditing feature (after the demo)
 - Heroku
- Distributed tasks
- Front end:
 - Login/Signup Page: Make return button an actual "Button Component" (Nash)
 - Login Page: List requirements on creating an account (Nash)
 - Courses/Instructors/Departments: Page Filtering working and displaying. (Seth)
 - Profile Page: Redirect if Not logged in (Erikas)
 - Nav Bar: Change Spacing, rearrange features (Seth).
- Backend:
 - Issue with reviews being associated with user (James)
 - Authentication token (Grant & Seth)
 - Exposed API page (Grant)
- Testing:
 - Twino(Mathew) - Top Bar, Bottom Bar, Reviews, Searches
 - Luciano - Front page, Top 5, Sign up, Login page
 - Logan - Course, Instructor and Department lists and individual pages

Meeting 18 - Sprint Planning 6 (Review 5) - March 16th 2021

- Everyone attended the meeting

Release 2 work:

- Remove Forums
- Attribute Reviews to the user (display on profile)
- Attribute Comments to the user
- Edit Reviews
- Report Reviews
- Delete Reviews
- Token-Based Authentication
- User prompts on verify (junk folder) and signup (don't use brock pass)
- Having professor to rate students (Final Release)

Full Stack/Test:

- work on formal testing for UI
- use Katalon Studio

Meeting 19 - Sprint Planning 7 (Review 6) - March 22th 2021

- Everyone except Erikas attended (was given explanation later)
- James
 - Connected reviews to user
 - Demonstrated
 - Added searches for course names
 - Had to add an extra dummy column
 - Removed forums model
- Consider making a way to generate reviews
- Seth and Grant working on Token-Based Authentication
 - Profile Page: Redirect if Not logged in
 - Grant and Seth decided to meet for token auth on Tuesday 23rd
- James will work on hiding API page
- Grant will work on edit reviews, report review, and delete review views
- Erikas will work on help functions for create account, login
- Discussed Heroku
 - Security flaws must be addressed before deployment
 - Seth will handle deploying on Heroku
- FullStack will work on UI testing automation
 - Should have most major use cases ready by March 29th

Meeting 20 - Sprint Refinement 7 - March 25th 2021

- James was unavailable for meeting, had informed earlier
- Demonstrate any changes
 - Login/sign up ui changes
 - Delete profile
 - Added verify account requirement on all pages needing it
 - Optimized hrefs
 - Backend: getuserreviews gets all the reviews for a specific user
- Testing progress
 - Showed documentation
 - need to correct some post conditions
 - Showed scripting and explained testing functionality
- Discussed how to document backlog and concluded that user stories with progress reports are what make up backlog
- Discussed possible features to implement
 - Add all reviews to profile page
 - Add tags to review e.g. tough marker, approachable

Meeting 21 - Sprint Planning 8 (Review 7) - March 29th 2021

- Everyone except Luciano attended meeting, Luciano informed about absence beforehand
- Seth demonstrated updates to profile pages
- Twino(Mathew) and Logan demonstrated formal UI testing
- Created list of all functions that need user stories and progress reports as a group
- Grant listed all the features that can worked on for final release
 - Having two ratings: Quality and difficulty
 - Tags e.g. lost of homework, tough marker, approachable
 - Thumbs up/down for reviews/comments
 - Ask users if they'd recommend a course
 - Show number of ratings
- Split tasks
 - Grant will be working on testing future functions
 - James has a heavy workload in other courses this week
 - Seth will work on bug fixes
 - Twino(Mathew), Logan, and Luciano will work on formal testing and documentation
 - Erikas will be given tasks by Seth

Meeting 22 - Sprint Refinement 8 (Release 2) - April 1st 2021

- Erikas, Luciano, and Logan were not in the meeting
 - Luciano and Logan had informed ahead of time
- Seth demonstrated changes
 - Changes with account creation page
 - New notification when changing password, changing nickname
 - Demonstrated viewing and making comments
 - Demonstrated how courses are now visible in department page
- Discussed testing
 - Finalized formal UI testing
 - Researched python testing documentation format
- Discussed teamwork policy and instructor intervention
- Grant demonstrated new functions he made in final release testing branch
 - tags
 - thumbs up/down
 - profanity filter
 - increased password security
 - reset password email
- Twino(Mathew) discussed bugs and issues
- Decided to not have a professor demonstration 2