**­­­College of Saint Benedict & Saint John’s University**

**Computer Science Department**

**CMC.com**

Phase: 2

Team Name: The Brogrammers

Team Leader: Logan Woods

Team Members: Cameron Bjork, Michael Murphy, Jaren Lindsey, Charlie Becker, Joe Koller, Logan Woods

Date: 3/29/2021

**[This page is intentionally left blank]**

**Table of Contents**

Use Case Diagram. ……………………………………………………………………… 4

Link to Git Repository ………………………………………………………………… 5

Use Case Descriptions ………………………………………………………………… 6-9

Class Diagram …………………………………………………………………………….. 10

Driver to Use Case Mapping ……………………………………………………….. 11-12

Meeting Minutes: Phase 2 ………………………………………………………….. 13-16

**Link to Git Repository**

[**https://github.com/cameronbjork/CMC\_2021.git**](https://github.com/cameronbjork/CMC_2021.git)

**Use Case Diagram**

**­Diagram

Description automatically generatedUse Case Descriptions**

|  |  |
| --- | --- |
| Name | U1: Login (Extended by U2: View All Users, U3: View All Universities, U4: View Profile, U5: View Saved Schools, U6: Search School, and U7: Log Out). |
| Actors | User (Student or Admin) |
| Triggers | User goes to the website |
| Preconditions | 1. User must have an account created by the administrator |
| Events | 1. User enters their username. 2. User enters their password. 3. User is logged in and directed to a menu which has different options for students and administrators. 4. The menu for students has options to “Manage my Profile”, “Manage My Saved Schools”, and “Search for Schools”. 5. The menu for administrators has options to “Manage Universities” and “Manage Users”. |
| Post-Conditions | 1. User is logged in |
| Alt Scenarios | 1. Username is incorrect Postcondition: User is directed to re-enter username 2. Password is incorrect Postcondition: User is directed to re-enter password |

|  |  |
| --- | --- |
| Name | U4: View Profile (Extended by U13: Edit Profile). |
| Actors | Student |
| Triggers | Student clicks “Manage My Profile”. |
| Preconditions | 1. Student must be logged in to account created by the administrator |
| Events | 1. The student can view their information after clicking the “Manage My Profile” button. 2. The information included is First Name, Last Name, Username, Password, and Type. |
| Post-Conditions | 1. Student is viewing their information. 2. Student has the option to “Edit User” or “Reset”. |
| Alt Scenarios | N/A |

|  |  |
| --- | --- |
| Name | U5: View Saved Schools (Extended by U14: Remove Saved School and U15: View School) |
| Actors | Student |
| Triggers | Student Clicks the “View Saved Schools” button |
| Preconditions | 1. Must be logged in 2. At least one school has been saved 3. Must click on the “View Saved Schools” button |
| Events | 1. Clicks the “View Saved Schools” button |
| Post-Conditions | The users saved schools are displayed |
| Alt Scenarios | 1. User has no saved school   None |

|  |  |
| --- | --- |
| Name | U6: Search School (Includes U16: View Search Results.) |
| Actors | Student |
| Triggers | Student clicks “View Saved Schools” button |
| Preconditions | 1. Must have logged in 2. Must have clicked “Search School” |
| Events | 1. Clicks “Search School” 2. Types in criteria 3. Clicks search |
| Post-Conditions | The results of the search are displayed |
| Alt Scenarios | N/A |

|  |  |
| --- | --- |
| Name | U7: Logout |
| Actors | User (Student or Admin) |
| Triggers | Student or admin clicks the logout button. |
| Preconditions | 1. User must be logged in. |
| Events | 1. Once user is logged out, they will be redirected to login screen. |
| Post-Conditions | 1. User is logged out. |
| Alt Scenarios | N/A |

|  |  |
| --- | --- |
| Name | U13: User Edit User (Extends U4: View Profile) |
| Actors | User (Student or admin) |
| Triggers | When user clicks on edit profile button after clicking on view profile |
| Preconditions | 1. User must successfully log into the system   1. User must click on Edit Profile button |
| Events | 1. User is redirected to the edit profile page 2. Can edit First Name 3. Can edit Last Name 4. Can edit Password with the correct criteria 5. Click save button |
| Post-Conditions | 1. Successfully saved in database (User is back to viewing profile) |
| Alt Scenarios | 1. User exits the page or selects the cancel button   Postcondition: None |

|  |  |
| --- | --- |
| Name | U14: Remove Saved School (Extends U5: View Saved Schools) |
| Actors | Student |
| Triggers | When user clicks on remove button next to school they selected |
| Preconditions | Student must be logged in and viewing a specific school |
| Events | 1. Student clicks on remove school button next to school 2. School gets removed from saved schools |
| Post-Conditions | 1. Successfully remove intended school from saved schools |
| Alt Scenarios | 1. Student exits the page   Postcondition: None |

|  |  |
| --- | --- |
| Name | U15: View School (Extended by U18: View Recommended Schools, U19: Compare. Universities, U20: Open School. Page, and U21: Save School) |
| Actors | Student |
| Triggers | Student clicks the view button of a particular school. |
| Preconditions | 1. Student is logged on. 2. Student is viewing search results. |
| Events | 1. A school can now be viewed individually along with its information. |
| Post-Conditions | Student is viewing the school they selected. |
| Alt Scenarios | N/A |

|  |  |
| --- | --- |
| Name | U16: View Search Results (included by Search School, extends View School, Sort School and Save School.) |
| Actors | Student |
| Triggers | Student hits search button |
| Preconditions | 1. Student is logged in 2. Student has filled out at least one query to search by |
| Events | 1. A list of schools that match the given search definitions are displayed |
| Post-Conditions | Student is viewing the list of schools |
| Alt Scenarios | Search query is empty  Post conditions: error message displays telling the user to add at least one category to search for |

|  |  |
| --- | --- |
| Name | U17: View Recent Search (Extended by Search School) |
| Actors | Student |
| Triggers | Clicking the Recent Search Icon |
| Preconditions | Student having searched something before |
| Events | 1. Click the view recent search button 2. It will fill in the last search parameters 3. Will show you the results?(or just fill in) |
| Post-Conditions | 1. It will fill in your most recent search. |
| Alt Scenarios | 1. They did not have a recent search   Post Condition: The button does not do anything. |

|  |  |
| --- | --- |
| Name | U18: View Recommended School (extended by View School) |
| Actors | Student |
| Triggers | Student clicks view recommended school when in View School |
| Preconditions | 1. Student is logged in 2. Student is viewing a school |
| Events | 1. A school that is very similar to the currently viewed school will appear as a suggested school. |
| Post-Conditions | Recommended school is now in view |
| Alt Scenarios | N/A |

|  |  |
| --- | --- |
| Name | U21: Save School (Extends View School and View Search Results) |
| Actors | Student |
| Triggers | Student Clicks the “Save School” Button |
| Preconditions | 1. Must be logged in to an account created by the admin. 2. Must be viewing schools or already viewing schools. |
| Events | 1. Save from the viewing school to database. 2. Will show in ‘Saved Schools’ List. |
| Post-Conditions | 1. The School is moved to the ‘Saved Schools’ List |
| Alt Scenarios | 1. Already saved that school.   Post Condition: It will not do anything.  (Where to go after most likely – search results page) |

**Class Diagram**

Diagram

Description automatically generated

**Driver to Use Case Mapping**

U1: Login -> runLogin() :

* Input: Username so user can be logged on
* Tests the main scenario when the username and password is correct which returns true.
* Tests alternate scenarios where password is wrong, username is wrong, and null data is implemented and returns false.

U4: View Profile -> runViewProfile()

* Input: Username so user can view their profile
* Checks to see if a User is returned to the ui, if so returns true.
* Output: User object

U5: View Saved Schools -> runDisplaySavedUniversities()

* Input: Username to check user’s saved schools
* Returns a list of saved universities to the Driver and displays them within the system.
* Returns true if this list is returned to the UI and the Driver
* Output: ArrayList of type University, the saved schools of the user

U13: User Edit User -> runEditProfile()

* Input: Username to get User object, First name to change, Last name to change, password to change
* First test case edits all of the users data and returns true with what is updated.
* Second test case makes sure that if 1 data point is null, the others will still update.

U6: Search School -> runSearchUniversity()

* Input: School name, state, location, control, minNumStudents, maxNumStudents, minPercentFemale, maxPercentFemale, minSATVerbal, maxSATVerbal, minSATMath, maxSATMath, minAnnualExpenses, maxAnnualExpenses, minPercentFinancialAid, maxPercentFinancialAid, minNumApplicants, maxNumApplicants, minPercentAdmit, maxPercentAdmit, minPercentEnrolled, maxPercentEnrolled, minAcademicScale, maxAcademicScale, minSocialScale, maxSocialScale, minQOLScale, maxQOLScale, emphasis1, emphasis2, emphasis3, emphasis4, emphasis5.
* Tests when the user inputs all data for all fields and returns true if search was executed.
* Tests when the user inputs all null data and returns false.
* Output: ArrayList of type university with similar results to what was searched.

U16: View Search Results

* Tests to see if the ArrayList is displayed to the user
* Output: ArrayList of type University with similar results to search above.

U18: View Recommended Schools -> runTopRecommendedUniversities()

* Input: University that is being viewed
* Checks to see if list of recommended schools is returned, outputs them to the user and then returns true.
* Output: ArrayList of recommended schools

U7: Log Out -> runLogOut()

* The main test scenario checks to see if a user is logged out and their account set to inactive, if correct the program will return true
* There is a username parameter that is passed through to the account controller where the user is found and the logged in status is set to false

U14: Remove Saved School -> runRemoveSavedUniversity()

* Input: University to be removed
* Tests to see if university is removed from ArrayList stored in User, then returns true

U21: Save School -> runSaveUniversity()

* Input: University to be added to User’s saved university.
* Checks if University is added to ArrayList stored in user, then returns true.

U15: Display School -> runDisplayUniversity()

* Input: School name to be displayed
* Tests when the user wants to display a university and returns the data of the university
* Output: University object for the requested school

U17: View Recent University -> runRecentUniversity()

* Input: User name and the university object of the most recent search by the user
* Tests to see if the most recent search by the user is correctly returned
* Output: The most recently searched University object

**Meeting Minutes Phase 2: 03/29/21 2:00pm – 3:30pm**

Michael worked on 100% of the use diagram with contributions from the group based on previously created communication diagrams and the use case diagram.

Joe made 100% contributions towards the Driver class based on the class diagram.

Each group member doing 2 communication diagrams before next meeting:

* Cam: U1, U4
* Joe: U5, U6
* Michael: U17, U21
* Jaren: U7, U15
* Charlie: U13, U14
* Logan: U16, U18

Detailed Task Description:

* For Phase 2 Progress meeting, we completed 22.5% (25% is re-do phase 1, and we have not completed the communication diagrams yet so 12.5% & Joe completed 10% of the driver).
* For Phase 2, we have completed 10% of what is required with the beginning of the driver.

**Meeting Minutes: Phase 2 – 03/30/21**

2:00 – 4:00:

* Each person Individually worked on 100% of their communication diagrams (75% done in total).
* Cameron pushed the project to git and began setting up the class structure.
* Joe worked 100% on the Driver class.

Next meeting scheduled for 7pm the same night.

Detailed Task Description:

* For Phase 2 Progress meeting, we completed about 50% of the way done(80% done with re-doing phase 1, Joe completed 70% of the driver class, Implementation started with classes 10% done).
* For Phase 2 , we have completed about 25% of what is needed (Driver, Implementation, GitHub).

7:00 – 10:30:

* Logan, Cameron, Michael, and Joe worked on about 50% of the Driver and Use Case Implementation. (Charlie is still having issues with git, and Jaren didn’t attend the meeting).
* Cameron created the github repository 100% and everyone else attempted to access it. group member had issues pushing/pulling but for the most part everything worked during this meeting.

Detailed Task Description:

* For Phase 2 Progress meeting, we are about 75% completed (re-do Phase 1 is done, Driver class and Implementation about 50% done).
* For Phase 2, we have completed about 40% of what is needed (Driver, Implementation, and GitHub).

**Meeting Minutes: Phase 2 – 03/31/21**

7:00pm – 11:00pm:

* Each group member focused 100% on their individual use cases but when someone needed help, we adjusted to based on wheat was needed. (Logan, Joe, Michael, and Cameron; Charlie still having issues with Git and Jaren didn’t attend the meeting).
* Together we finished 100% of the driver class, and implementation of use cases.

Detailed Task Description:

* For Phase 2 Progress, we are 95% done (Driver done, Implementation done, still need to revisit Class Diagram before meeting).
* For Phase 2, we are still probably 40%-45% of the way done (Driver, Implementation, and GitHub).

**Meeting Minutes: Phase 2 – 04/1/21**

4:00pm – 7:00pm:

* Joe and Cam: Use Case work and implementation (15% each)
* Michael: Admin Use case diagrams (30%)

**Meeting Minutes: Phase 2 – 04/2/21**

9:00am – 10:00am:

* Joe: Java docs and class clean up (30%)
* Michael: Admin Use case diagrams (30%)
* Cam: Use case work and implementation (25%)

11:00-12:00 pm:

* Logan: Use case work and java docs edits (15%)

**Meeting Minutes: Phase 2 – 04/4/21**

7:30pm – 11:00pm:

* Joe and Cam: Use Case clean up and completion (Cam: 20%, Joe:10%)
* Joe, Cam, Michael, Logan: Java docs completion (25% each)
* Cam: Java docs comment and class creation(100%)
* Michael, Logan, and Joe: Class diagram edits and clean-up (33% each)
* Everyone: Final submission and documentation clean up (25% each)

**Task Decomposition of Overall Work**

Cam: 35%

Joe, Logan, Michael: 20%

Charlie: <5% (Could not work git)

Jalen: 0% (Did not show up to any meetings or did not turn on mic at all)