**Group 1: Arman Sanentz, Phong Pham, Jin Hwan Lee, Shirui Ye, Carlos Luis Syquia**

**Use Cases for**

**“Find My Professor” Application**

**Version 1.0**

**Prepared by Arman Sanentz, Phong Pham, Jin Hwan Lee, Shirui Ye, Carlos Luis Syquia**

**Boston University CS411**

**March 24, 2016**

1. Guidance for Use Case Template

Document each use case using the template shown in the Appendix. This section provides a description of each section in the use case template.

2. Use Case Identification

## **1.1.** **Use Case ID**

Give each use case a unique integer sequence number identifier. Alternatively, use a hierarchical form: X.Y. Related use cases can be grouped in the hierarchy.

## **1.2.** **Use Case Name**

State a concise, results-oriented name for the use case. These reflect the tasks the user needs to be able to accomplish using the system. Include an action verb and a noun. Some examples:

• View part number information.

• Manually mark hypertext source and establish link to target.

• Place an order for a CD with the updated software version.

## **1.3.** **Use Case History**

### **1.1.1.** **Created By**

Supply the name of the person who initially documented this use case.

### **1.1.2.** **Date Created**

Enter the date on which the use case was initially documented.

### **1.1.3.** **Last Updated By**

Supply the name of the person who performed the most recent update to the use case description.

### **1.1.4.** **Date Last Updated**

Enter the date on which the use case was most recently updated.

3. Use Case Definition

## **1.1.** **Actors**

An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor that will be initiating this use case and any other actors who will participate in completing the use case.

## **1.2.** **Trigger**

Identify the event that initiates the use case. This could be an external business event or system event that causes the use case to begin, or it could be the first step in the normal flow.

## **1.3.** **Description**

Provide a brief description of the reason for and outcome of this use case, or a high-level description of the sequence of actions and the outcome of executing the use case.

## **1.4.** **Preconditions**

List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each precondition. Examples:

1. User’s identity has been authenticated.

2. User’s computer has sufficient free memory available to launch task.

## **1.5.** **Postconditions**

Describe the state of the system at the conclusion of the use case execution. Number each postcondition. Examples:

1. Document contains only valid SGML tags.

2. Price of item in database has been updated with new value.

## **1.6.** **Normal Flow**

Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and description. This description may be written as an answer to the hypothetical question, “How do I <accomplish the task stated in the use case name>?” This is best done as a numbered list of actions performed by the actor, alternating with responses provided by the system. The normal flow is numbered “X.0”, where “X” is the Use Case ID.

## **1.7.** **Alternative Flows**

Document other, legitimate usage scenarios that can take place within this use case separately in this section. State the alternative flow, and describe any differences in the sequence of steps that take place. Number each alternative flow in the form “X.Y”, where “X” is the Use Case ID and Y is a sequence number for the alternative flow. For example, “5.3” would indicate the third alternative flow for use case number 5.

## **1.8.** **Exceptions**

Describe any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions. Also, describe how the system is to respond if the use case execution fails for some unanticipated reason. If the use case results in a durable state change in a database or the outside world, state whether the change is rolled back, completed correctly, partially completed with a known state, or left in an undetermined state as a result of the exception. Number each alternative flow in the form “X.Y.E.Z”, where “X” is the Use Case ID, Y indicates the normal (0) or alternative (>0) flow during which this exception could take place, “E” indicates an exception, and “Z” is a sequence number for the exceptions. For example “5.0.E.2” would indicate the second exception for the normal flow for use case number 5.

## **1.9.** **Includes**

List any other use cases that are included (“called”) by this use case. Common functionality that appears in multiple use cases can be split out into a separate use case that is included by the ones that need that common functionality.

## **1.10.** **Priority**

Indicate the relative priority of implementing the functionality required to allow this use case to be executed. The priority scheme used must be the same as that used in the software requirements specification.

## **1.11.** **Frequency of Use**

Estimate the number of times this use case will be performed by the actors per some appropriate unit of time.

## **1.12.** **Business Rules**

List any business rules that influence this use case.

## **1.13.** **Special Requirements**

Identify any additional requirements, such as nonfunctional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.

## **1.14.** **Assumptions**

List any assumptions that were made in the analysis that led to accepting this use case into the product description and writing the use case description.

## **1.15.** **Notes and Issues**

List any additional comments about this use case or any remaining open issues or TBDs (To Be Determineds) that must be resolved. Identify who will resolve each issue, the due date, and what the resolution ultimately is.

Use Case List

|  |  |  |
| --- | --- | --- |
| **ID** | **Primary Actor** | **Use Case Title** |
| **A.1** | User | Search for Professor by course number |
| **A.2** | User | Add / Remove course from user course list |
| **A.3** | User | Obtain directions to Professor’s office |
| **A.4** | User | Link to the course site which the user searches for |
| **B.1** | Prof/database collector | Add / Remove office hours and classes |
| **B.2** | Prof/database collector | Add / Remove office location |
| **C.1** | Prof / User | Login to account |
| **C.2** | Prof / User | Create account |

Use Case A.1

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | A.1 | | |
| Use Case Name: | Search for Professor using his/her BU course number(s). | | |
| Created By: | Carlos, Arman Phong | Last Updated By: | Shirui, Jin Hwan |
| Date Created: | 02/16/16 | Date Last Updated: | 02/17/16 |

|  |  |
| --- | --- |
| Actors: | User (student) |
| Description: | Allow the student to find a professor using a BU Course number as the search key |
| Trigger: | Inserting the search key in the search bar |
| Preconditions: | 1. Authenticate user identity 2. Verify input information is in database |
| Postconditions: | 1. A Professor’s data is visible to the user. 2. On the screen, a Professor’s “tab/pin” displays. 3. The user has the option of adding this professor/course to his/her course list. 4. The user has the option of obtaining directions to the professor’s office. 5. The user has the option button which links the user to Ratemyprofessor. |
| Normal Flow: | A.1.1 The user has his/her course numbers available/ready to be searched  A.1.2 The user enters a course number from his/her classes into the search bar to search for the specific professor  A.1.3 The user presses search  A.1.4 The system displays course number and professor e.g. CS 101 A1 - Prof. Donham  A.1.4.1 If one corresponding professor, the user is matched to the professor  A.1.4.2 If more than one professors, the user chooses the professor from the display  A.1.4.3 If no search found, the system displays “No match found” and return to A.1.2  A.1.5 The system displays the location of the professor’s office, as well as the professor’s “Tab/Pin” - containing all of the information |
| Alternative Flows: | A.1.3.1 The user doesn’t press the search. Changes his/her mind and clears the search box  A.1.3.1.1 The user either enters a different course number and presses search then proceeding to A.1.4 or the user goes back to viewing content on the current page |
| Exceptions: | A.1.3.E.1 System finds user’s input is invalid (i.e wrong course number, tried to enter a string, etc.). On screen will prompt the user to input a valid course number |
| Includes: | A.2, A.3 |
| Priority: | Lower priority in comparison to other use cases |
| Frequency of Use: | About 5 times a day per user |
| Business Rules: | None |
| Special Requirements: | Strings have requirements (i.e a course # is 5-9 characters long without/with white space ex: “CS 101 A1” = 9 characters), anything else should give an error when search button is pressed |
| Assumptions: | 1. User knows beforehand what course they want to search.(or user knows the name of the Professor) 2. User is a student and not a Professor (in this use case) |
| Notes and Issues: | Priority could change. Searching would be implemented last after we get everything else running. Special Requirements can be changed to be less constricted (i.e let user search by name). Frequency of Use can vary depending on user and time in the semester (i.e more use closer to assignment due dates or exam dates). |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Shirui Ye | 2/17/16 | update B.1,B.2 and add one more use case A.4, one postcondition |  |
| Jin Hwan Lee | 2/17/16 | update precondition 5 and edited normal flow |  |

Use Case A.2

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | A.2 | | |
| Use Case Name: | Add / Remove course from user course list | | |
| Created By: | Arman Sanentz | Last Updated By: | Arman Sanentz |
| Date Created: | 02/16/16 | Date Last Updated: | 03/15/16 |

|  |  |
| --- | --- |
| Actors: | User (student) |
| Description: | Allows the user to store his/her specific enrolled courses in a page/tab. Specifically, the user is able to “add to my courses” or “remove from my courses” by clicking an “ADD” or “REMOVE” option next to the enrolled course in the my courses tab. |
| Trigger: | Click |
| Preconditions: | 1. Authenticate user identity 2. Verify input information is in database |
| Postconditions: | 1. The user has stored an enrolled course in the courses page, readily available for access 2. The user has removed a course from his/her course list, no longer viewing pertinent information to the course |
| Normal Flow: | A.1.1 The user has his/her course numbers available and are found within the database  A.1.2 The user selects the course in which he/she is enrolled  A.1.3 The user presses “Add to My Courses”  A.1.3.1 The user selects “Remove from My Courses”  A.1.4 The system adds the selected course to a My Courses Page  A.1.4.1 The system removes the selected course from the My Courses Page  A.1.5 The My Courses page displays the registered course information  A.1.5.1 The My Courses page no longer displays the registered course |
| Alternative Flows: | A.1.5.1 The user heads directly to the My Courses page to view previously added courses. |
| Exceptions: | A.1.3.E.1 The user tries to add a course to the My Courses page that is already present on the page. |
| Includes: | A.1, C.1 |
| Priority: | Relatively high priority, in terms of purpose of the web app. |
| Frequency of Use: | As many times as courses are registered |
| Business Rules: | None |
| Special Requirements: | The user is familiar with the BU course numbering system. |
| Assumptions: | 1. User knows beforehand what course they want to search.(or user knows the name of the Professor) 2. User is a student and not a Professor (in this use case) 3. The student has already added a course to the My Courses Page (in the case of a removal) |
| Notes and Issues: | Major functional component to the app, meant to make things much easier for the user. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

Use Case A.3

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | A.3 | | |
| Use Case Name: | Obtain directions to Professor’s office | | |
| Created By: | Arman Sanentz | Last Updated By: | Arman Sanentz |
| Date Created: | 02/16/16 | Date Last Updated: | 03/23/16 |

|  |  |
| --- | --- |
| Actors: | User (student) |
| Description: | Allows the user to obtain walking, driving, or public transit directions (using Google Maps) from his or her current location to the building in which a specific professor’s office is. |
| Trigger: | Click |
| Preconditions: | 1. The user’s IP address will allow access to his/her current location. 2. The professor’s office location is stored within our system/pinned on the map. |
| Postconditions: | 1. The user is given a highlighted route from point A to point B, allowing them to determine how to go to the office. |
| Normal Flow: | A.1.1 The user locates a specific professor/course.  A.1.2 The user presses “Guide me”  A.1.3 The system obtains the user’s current location through an IP Address  A.1.4 Interacting with the Google Maps API, the user is presented with a route from the current location to the already stored office location.  A.1.5 The user is able to select a mode of transportation to be used from point A to B to specify the most optimal route |
| Alternative Flows: | A.1.5.1 The user simply views the location of the office as a pin, rather than getting a route to the office. |
| Exceptions: | A.1.3.E.1 The user’s device does not allow access to the user’s current location, thus not allowing routing from a current location. |
| Includes: | A.1, C.1 |
| Priority: | High priority, in terms of purpose of the web app. |
| Frequency of Use: | As many times as courses are registered |
| Business Rules: | None |
| Special Requirements: | The user is familiar with the BU course numbering system. |
| Assumptions: | 1. User knows beforehand what course they want to search.(or user knows the name of the Professor) 2. User is a student and not a Professor (in this use case) 3. The student has already added a course to the My Courses Page (in the case of a removal) |
| Notes and Issues: | Major functional component to the app, meant to make things much easier for the user. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

Use Case A.4

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | A.4 | | |
| Use Case Name: | Link to the course site the user searches for | | |
| Created By: | Carlos | Last Updated By: | Carlos |
| Date Created: | 03/15/16 | Date Last Updated: | 3/16/16 |

|  |  |
| --- | --- |
| Actors: | User (student) |
| Description: | User is linked to the specific course site they searched for |
| Trigger: | User clicks the link provided under the course and professor information they searched for |
| Preconditions: | 1. User has searched for a course 2. User has found the course and professor they searched for 3. Course searched for has a course site |
| Postconditions: | 1. The course site specified is displayed on screen in a new tab or window. 2. Our application is still open in the original tab or window. |
| Normal Flow: | A.4.1 The user has successfully found the course and professor they were looking for.  A.4.2 The system is displaying all the relevant course information, including the link to the course site.  A.4.3 The user clicks the link to the course website.  A.4.3 The user is redirected to the course website. |
| Alternative Flows: | A.4.2.1 There is no specified course |
| Exceptions: | A.4.2.E.1 There is no course website publicly available. There will be no course site link on screen with the course information. |
| Includes: | A.1 |
| Priority: | Lower priority – not all courses have a course site |
| Frequency of Use: | About 2 times a day per user |
| Business Rules: | None |
| Special Requirements: | Course must have a course site stored in database. |
| Assumptions: | 1. Course has a course site |
| Notes and Issues: | Priority could change. Possibly can allow a professor to update the course website stored in the database. Frequency of Use can vary depending on user and time in the semester (i.e more use closer to assignment due dates or exam dates). |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Use Case B.1**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | B.1 | | |
| Use Case Name: | Add / Remove office hours and classes | | |
| Created By: | Jin Hwan Lee | Last Updated By: | Jin Hwan Lee |
| Date Created: | 03/15/16 | Date Last Updated: | 03/15/16 |

|  |  |
| --- | --- |
| Actors: | User(Professors) and administrator |
| Description: | Allow the professor to add and remove his/her office hours and classes from list in the application |
| Trigger: | Click Add / Remove in the his / her page |
| Preconditions: | 1. Authenticate user identity 2. Verify input information is in database |
| Postconditions: | 1. The user can view updated office hours and classes |
| Normal Flow: | B.1.1 The user clicks “Add / Remove classes” icon on his/her page  B.1.2 The system displays choice between Add and Remove  B.1.2.1 If user chooses “Add”, the add format is displayed, continued B.1.3  B.1.2.2 If user choose “Remove”, checkboxes appear next to current schedule, continued B.1.6  B.1.3 The user fills out the given format  B.1.4 The user clicks “add” to add the new office hours and classes to his/her profile  B.1.5 The system display preview of the page with choice of “Post” and “Edit”  B.1.5.1 If user chooses “Post”, the system displays the new office hours and classes on the web for users to view  B1.5.2 If user choose “Edit”, the system repeats from B.1.3  B.1.6 The user clicks check and click “Remove”  B.1.6 The system display pop-up confirming the removal of checked office hours and classes(“Yes” / “No”)  B.1.6.1 If the user clicks “Yes”, the system remove checked office hours  B.1.6.2 If the user clicks “No”, the system returns to B.1.1 |
| Alternative Flows: | None |
| Exceptions: | IB.1.3.E.1 System finds user’s input is invalid (i.e wrong course number, tried to enter a string, etc.). On screen will prompt the user to input a valid course number  B.1.3.E.2 If some parts left blank, system prompt user to fill out the missing part |
| Includes: | B.2 |
| Priority: | High priority compared to other user cases |
| Frequency of Use: | Twice per semester |
| Business Rules: | None |
| Special Requirements: | Has to be a professor or an administrator |
| Assumptions: | 1. User knows beforehand time and place of the office hour and classes. 2. User is a Professor and not a student (in this use case) |
| Notes and Issues: | Basic format adding office hours and classes should be made |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Use Case B.2**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | B.2 | | |
| Use Case Name: | Update office location | | |
| Created By: | Shirui Ye | Last Updated By: | Carlos |
| Date Created: | 03/14/16 | Date Last Updated: | 3/23/16 |

|  |  |
| --- | --- |
| Actors: | User(Professors) and administrator |
| Description: | Allow professors and administrators to add/remove professor’s office locations |
| Trigger: | The user clicks “Add/remove office location” |
| Preconditions: | 1. Authenticate user identity 2. Verify input information is in database 3. User is logged in as a professor or administrator |
| Postconditions: | 1. The location of the professor’s office is updated 2. The location can be seen by other users 3. The professor and administrator can update the location anytime |
| Normal Flow: | B.2.1 The user clicks “Update office location” on his/her page  B.2.2 The user is prompted to enter the address of their new office location.  B.2.3 The user is prompted to enter the room number of their office  B.2.4 The user clicks “Update” at the bottom of the page  B.2.4.1 The user decides not to update their office location and clicks “Cancel”.  B.2.5 The system displays a preview of their updated information on their page and is prompted to confirm the changes.  B.2.6 The user clicks “confirm” and the DB is updated with the new information and the page is reloaded to reflect the new changes. |
| Alternative Flows: | None |
| Exceptions: | Input invalid or illegal input. System fail to update and ask the user to update new location again |
| Includes: | B.1 |
| Priority: | High priority compared to other user cases |
| Frequency of Use: | Twice per day |
| Business Rules: | None |
| Special Requirements: | Has to be a professor or an administrator |
| Assumptions: | 1. The user knows where their new office location is 2. The user is a professor |
| Notes and Issues: | We may need a tutorial to teach professor how to use this application. More specific, we need to teach them how to update their information.  Where should we put this feature in the app? |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Carlos | 3/16/16 | Minor formatting changes |  |
| Carlos | 3/23/16 | Adding more details to preconditions, postconditions and normal flow |  |

Use Case C.1

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | C.1 | | |
| Use Case Name: | Login to account | | |
| Created By: | Phong Pham | Last Updated By: | Phong Pham |
| Date Created: | 03/16/2016 | Date Last Updated: | 03/16/2016 |

|  |  |
| --- | --- |
| Actors: | Professor / User (student) |
| Description: | Allows the professor or the student to login to their account |
| Trigger: | Entering appropriate data (username/email and a corresponding password) followed by pressing the login button |
| Preconditions: | 1. The user is registered with the system |
| Postconditions: | 1. The user is then redirected to the homepage  2. User now has access to his/her course list |
| Normal Flow: | C.1.1 The user inputs their username in the corresponding box  C.1.2 The user inputs their password in the corresponding box  C.1.3 The user then presses the login button  C.1.4 The system authenticate the user to user database  C.1.4.1. If yes, go to home page  C.1.4.2. If no, system display “Wrong ID/PW, please type username and password again” and repeat C.1.1 |
| Alternative Flows: | C.1.3.1 The user does not press the login button and clears the information they typed  C.1.3.1.2 The user enters new information and logs in  C.1.3.2 The user decides to close the browser without logging in |
| Exceptions: | C.1.3.E.1 The user id is not registered within the system. On screen will prompt the user to register an account.  C.1.3.E.2 The user inputs the correct id, but the wrong password. On screen will prompt the user to enter the password again. |
| Includes: | C.2 |
| Priority: | High priority (allows us to link data to specific users) |
| Frequency of Use: | 1 time per session per user |
| Business Rules: | None |
| Special Requirements: | Usernames/ids are unique; Usernames will have a minimum character requirement; passwords will have a minimum character requirement; |
| Assumptions: | 1. Assume unique usernames for every user.  2. Assume user is already registered |
| Notes and Issues: | Have to do check on the username first and a second check on the password if the first check passes. Possibility of guest account for users. Guest account would have restricted access and data would not be saved. Guest account should only be considered if time permits. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

Use Case C.2

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | C.2 | | |
| Use Case Name: | Create Account | | |
| Created By: | Phong Pham | Last Updated By: | Phong Pham |
| Date Created: | 03/18/2016 | Date Last Updated: | 03/18/2016 |

|  |  |
| --- | --- |
| Actors: | Professor / User (student) |
| Description: | Allows the professor or the student to create an account |
| Trigger: | Entering appropriate data (non registered username/email and a corresponding password) followed by pressing the create button |
| Preconditions: | None |
| Postconditions: | 1. The user is then redirected to the login page  2. User can now login |
| Normal Flow: | C.2.1 The user inputs desired username in the corresponding box  C.2.2 The user inputs desired password in the corresponding box  C.2.3 The user then presses the create button  C.2.4 The system checks the username to user database  C.2.4.1. If available then the database enters the username and password into the database  C.2.4.2. If not available, system displays “Username not available, please type in a different username” and start over from C.2.1 |
| Alternative Flows: | C.2.3.1 The user does not press the create button and clears the information they typed  C.2.3.1.2 The user enters new information  C.2.3.2 The user decides to close the browser without creating an account |
| Exceptions: | C.2.3.E.1 The user id is registered within the system. On screen will prompt the user to login. |
| Includes: | None |
| Priority: | High priority |
| Frequency of Use: | 1 time per session per user |
| Business Rules: | None |
| Special Requirements: | Usernames/ids are unique; Usernames will have a minimum character requirement; passwords will have a minimum character requirement; |
| Assumptions: | 1. Assume unique usernames for every user. |
| Notes and Issues: | Have to do check on the desired username. Need to make sure username is not already in database. |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |