Software Requirements Specification

Version 2.3

(7th week)

HYU ⓔ mini Application

4. 18. 2016

//목차

1. **Introduction**

**1.1. Purpose**

The purpose of this document is to present a detailed description of the Mobile web application named 'HYUⓔmini'. This document will explain the features of the application, the interfaces of the application, what the application will do, the constraints under which it operates and how the application will react to external inputs. This document is intended for both the stakeholders and the developers of the application.

**1.2. Scope of Project**

This application will a compact version of currently using Hanyang University application. We will call this as a 'HYUⓔmini'. This application provides only key functions of current Hanyang University application. This application is more intuitive and lighter(uses less memory spaces and fast run time). By reducing functions, user can reduce overuse of time to search for wanted functions and that leads to remove inconvenience of using abundant application.

**1.3. Glossary**

university : Hanyang University at Ansan

Restaurants : Dormitory, Student, Foodcourt, Staff, Business incubator Center restaurants

DB : including student information, shuttle bus schedule information and so on

User : Hanyang University student and manager of 'HYUⓔmini' application

Shuttle bus : bus on circular route whole day

Commute bus : bus for commute to or from school, that only runs on morning and night

operation time : hour of operation of buses

BBS(Bulletin Board System) : Free bulletin board that students can write questions, comments, and so on in the internet

Content: the main text stuff which user writes in BBS

Content page: the page in which content is displayed in BBS

**1.4. Overview of Document**

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.  
 The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes the details of the functionality of the product.  
 Both sections of the document describe the same software product in its entirety, but are intended for different audiences.

1. **Overall Description**

**2.1 System Environment**

**C:\Users\baek\AppData\Local\Microsoft\Windows\INetCache\Content.Word\시스템인바이론먼트.png**

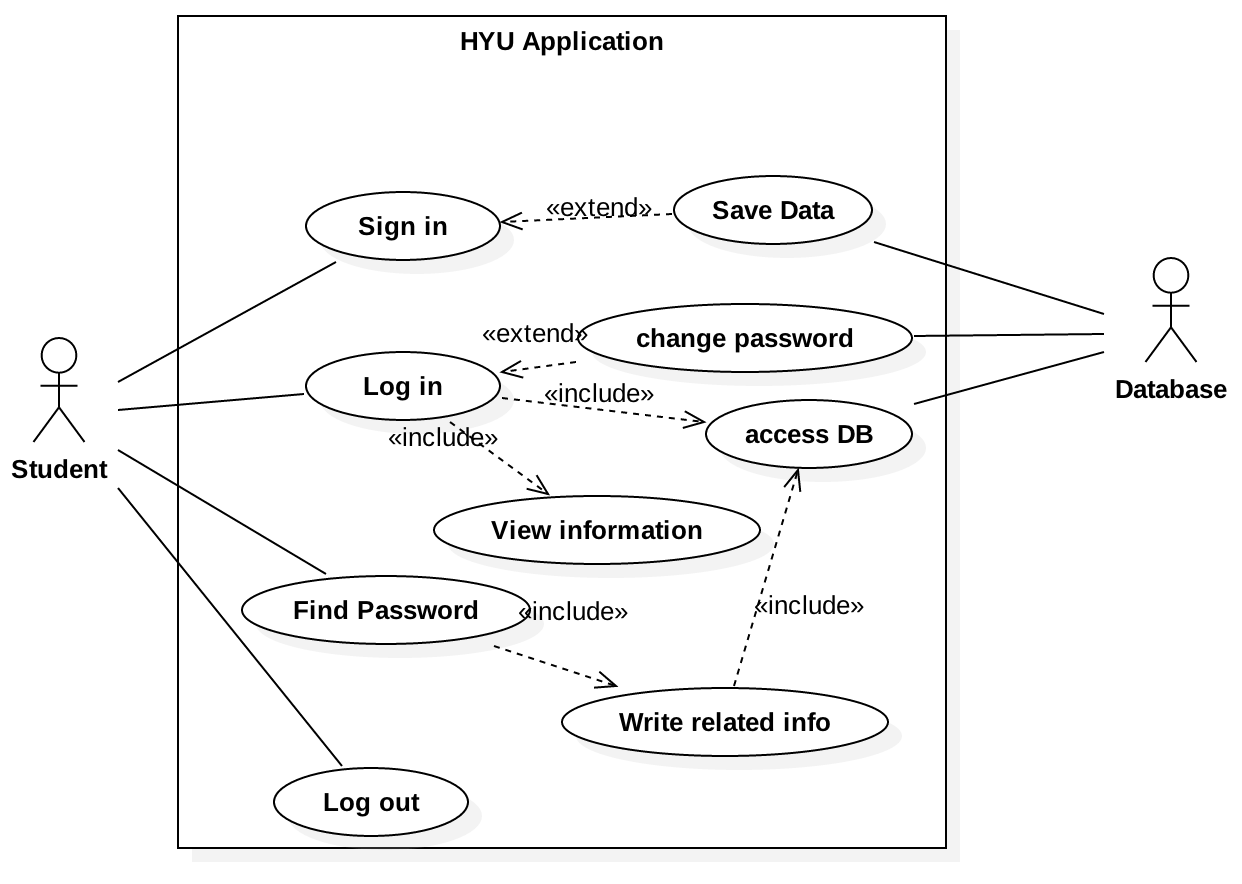
**2.2 Use Cases**

**2.2.1 Use Case 1: Login system**

* **Description**

The student uses the application to check some information what are student of the concerned. Student should input his student ID and password in textbox. When he completed the input, student should click ‘enter’ to sign in. If student forgets the password, he can reset his password by ‘find password’ feature. If student has not change his password for long time, student can use the feature of ‘change password’ to change his password.

* **Use Case Diagram**



**2.2 Use Case 2: Main Page**

* **Description**

A student succeed login to the main page. The application presents main menus, ‘Weekly Menu’, ‘Time Table’, ‘School Bus Schedule’, ‘Free Board’, ‘Mobile Student ID’ and ‘Logout’. The student selects one of the main menus. The application presents the web page of the selected menu.

* **Use Case Diagram**

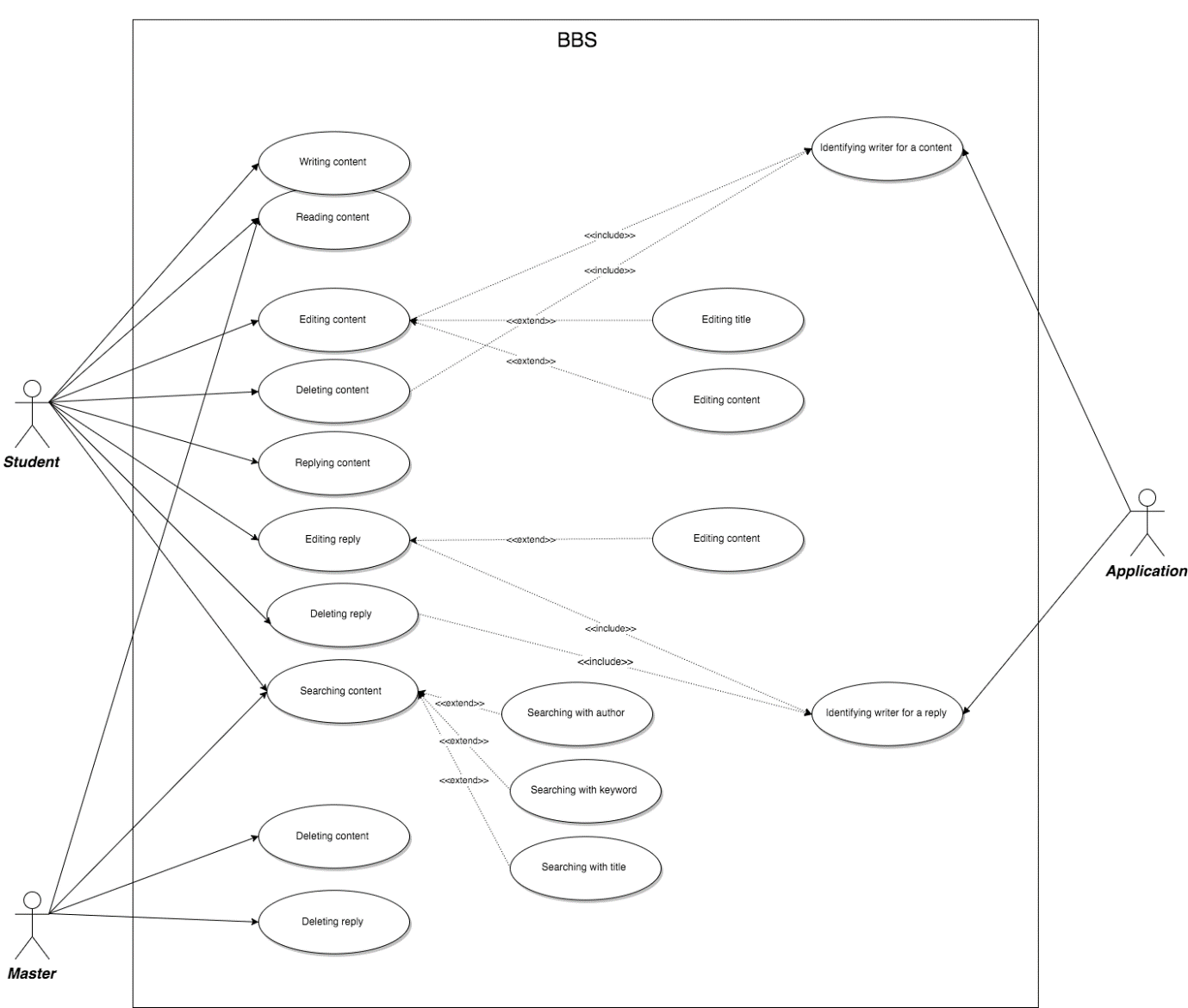
usecase2_mainpage (2)

**2.3 Use Case 3: BBS**

* **Description**

Student can write a content that they want to notify to other student. Also student can read the content in the bulletin board or if they want to reply for the content, they can do it. Additionally, Student can edit and delete a content or reply which student wrote and student can search a content with author, keyword and title.

* **Use Case Diagram**

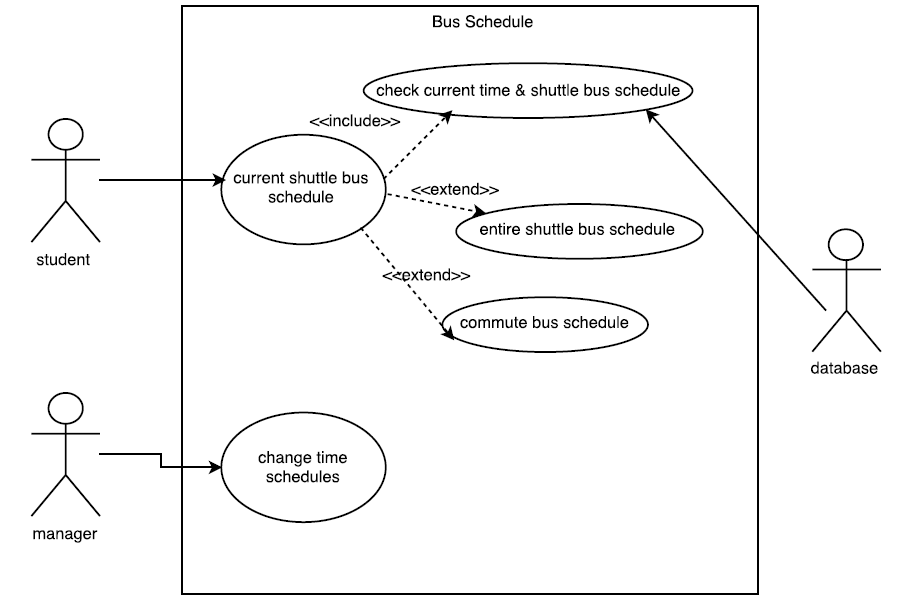


**2.4 Use Case 4 : Bus Schedule**

* **Description**

User can check shuttle bus and commute bus schedule. There are two tabs in bus schedule page. In shuttle bus tab, all shuttle bus schedule is provided in table of current term among semester, session and break. Top of the table of shuttle bus schedule, locations of current operating buses are provided in one visualized compact map. In commute bus tab, all lines of commute buses are showed, only name of line is provided. If user clicks each line, user can show detail of that line.

* **Use Case Diagram**

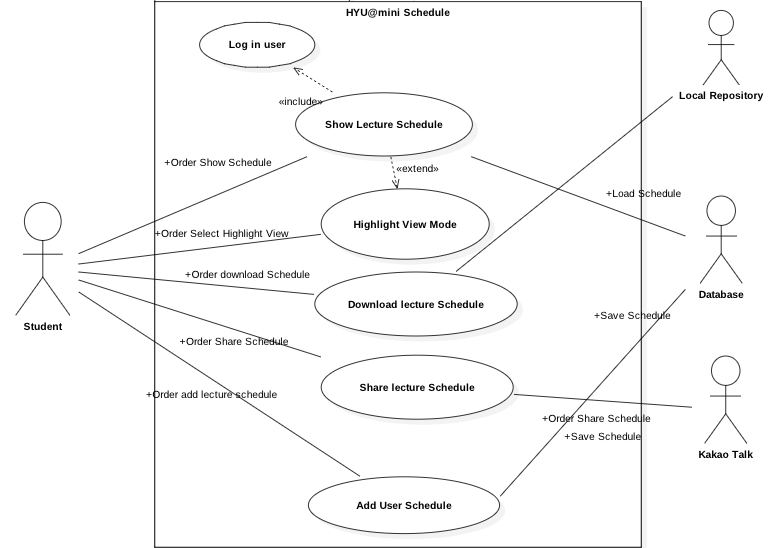


**2.2.5 Use Case 05 : Lecture Schedule**

* **Description**

Lecture Schedule is for student who want to see lecture schedule in 'HYUⓔmini'. In addition to basic function of showing lecture schedules, 'HYUⓔmini' also provides highlight view mode of making highlight to current schedule, adding personal schedule function that user can add their personal schedule and sharing function that user can share their schedule through ‘Kakao Talk’. Also user can save their schedule in their local machine.

* **Use Case Diagram**

****

**2.2.6 Use Case 6: Weekly Menu**

* **Description**

A student selects ‘Weekly Menu’. The application presents meal menus of now opened cafeterias. The student can select another day of week. The student can select another mealtime (‘Breakfast’, ‘Lunch’, ‘Dinner’) on weekdays. The student can select another meal type (‘Korean’, ‘Western’, ‘Snack’) on ‘Food Court’. The student can see only current week’s menu.

* **Use Case Diagram**

weeklymenu

**2.2.7 Use Case 07: Manager page**

* **Description**

Since our project use our own data base, manager(admin) should able to modify all the data bases including board, bus schedule, member information. Because this application crawling weekly menu, manager don’t need to modify this.

* **Use Case Diagram**

****

**2.2.8 Use Case 8: Page Switch**

* **Description**

A student or master can move to other menu in everywhere on the application at one click. For example, when the student or master watches the weekly menu, they can move to the time table as they click the “Time Table” button at the bottom of the application.

* **Use Case Diagram**



**2.3 User Characteristics**

The users are students in the university. The students are expected to be able to use in daily life of university. The main functions of 'HYUⓔmini' will have the bulletin board, restaurant menus, bus timetable and log in for mobile student card and lecture schedule.

1. **Requirements Specification**

**3.1 Use case 1: Login system**

**3.1.1 Functional Requirement 1: Log in**

|  |  |
| --- | --- |
| Use Case Name | Log in |
| Goal in context | Student signs in system for checking other information. |
| Trigger | Student clicks the ‘log in’ button to sign in. |
| Precondition | User didn’t sign in the system. |
| Basic Paths | 1. Students click the ‘Sign in’ button in HYU Mini. 2. Student should input his ID and password in edit box. 3. Student clicks the ‘enter’ button for accessing system. 4. The student is enable to check schedule and BBS after sign in. |
| Alternative Paths | In step 1, if student clicks the ‘Remember’ button, the application should record the user ID and password in cookies.  In step 3, if student forgot the password, the ‘find password’ is able to student reset the password.  In step 3, student clicks the ‘change password’ button, the password can be changed by this function. |
| Post-condition | Show the homepage to user |

**3.1.2 Functional Requirement 2: Log out**

|  |  |
| --- | --- |
| Use Case Name | Log out |
| Goal in context | Student can log out this system after checking information. |
| Trigger | Student clicks the ‘log out’ button to sign out. |
| Precondition | Student already log in this system. |
| Basic Paths | 1. Student already accesses HYU application.  2. Student who wants to log out clicks the ‘log out’ button.  3. HYU application will be broken the connection. |
| Alternative Paths | Remain user information. |
| Post-condition | Clean up the user information and log out |

**3.1.3 Functional Requirement 3: Find password**

|  |  |
| --- | --- |
| Use Case Name | Find password |
| Goal in context | Student can find his password once he forgot it. |
| Trigger | Student clicks the ‘find password’ button . |
| Precondition | Student forgot the password. |
| Basic Paths | 1. Input the answer for question  2. Input new password  3. If answer is correct, the password should be updated |
| Alternative Paths |  |
| Post-condition | The user’s password updated. |

**3.1.4 Functional Requirement 4: Change password**

|  |  |
| --- | --- |
| Use Case Name | Change password |
| Goal in context | Student can change his password by older one. |
| Trigger | Student clicks the ‘change password’ button to sign out. |
| Precondition | Student has not changed password for long time. |
| Basic Paths | 1. Student should know his old password.  2. Input old and new password in textbox.  3. If old password is correct, update the password. |
| Alternative Paths | In step 2, we can verify the correctness between twice new password from input. |
| Post-condition | Update to new password |

**3.2 Use case 2: Main Page**

**3.2.1 Functional Requirement 1: Show main menus**

|  |  |
| --- | --- |
| Function name | Show main menus |
| Goal in context | Student sees the main menu in main page of the application. |
| Trigger | Student logged in successfully or, the ‘Go to main page’ or ‘back’(in Android) button pressed from other pages. |
| Precondition | Student logged in successfully. |
| Basic Path | 1. Student successes login.  2. The application shows 6 main menus (‘Weekly Menu’, ‘Time Table’, ‘School Bus Schedule’, ‘Mobile Student ID’, ‘Free Board’, ‘Logout’) to the student. |
| Alternative Paths |  |
| Postcondition | Student sees 6 main menu buttons. |

**3.2.2 Functional Requirement 2: Select a menu.**

|  |  |
| --- | --- |
| Function name | Select a menu |
| Goal in context | Student enter the menu page which is selected. |
| Trigger | Student clicks a menu button or press ‘back’ button (in Android). |
| Precondition | The six main menu buttons are showing in the screen. |
| Basic Path | 1. Student selects and clicks a menu button.  2. The application presents selected menu page. |
| Alternative Paths | In step 1, even though the student pressed ‘back’ button (in Android), application must keep on presenting main page. |
| Postcondition | The application presents the selected menu page. |

**3.2.3 Non-Functional Requirements**

* **User Interface**

The main menu buttons must be displayed in one screen. The menu buttons have to be expressed by picture, e.g. pictogram. ‘Logout’ button should be sized smaller than the other buttons. The main menu buttons should have some graphical composition (For example, a tree and six apples, the apples would be menu buttons.), but it’s optional requirement.

**3.3 Use case 03 : BBS**

**3.3.1 Functional Requirement 1: Write contents**

|  |  |
| --- | --- |
| Function name | Write contents |
| Goal in context | Student write a content in a bulletin board |
| Trigger | clicks “writing” button in a bulletin board page |
| Precondition | Student logins the app  Student accesses into a bulletin board page |
| Basic Path | 1. The student clicks “writing” button.  2. The application shows the student editing page.  3. The student inputs words of a title and content.  The student chooses if student clicks “save” button or “cancel” button.  4. If student clicks “save” button, the application checks the validation of a title or content.  If the validation is passed, writing action is done. |
| Alternative Paths | In step 4, if the student clicks “cancel” button, the application stops editing function and shows up a list page of a bullentin board.  In step 6, the validation is not passed, the application displays alert for which one student should input again. Return to step 3. |
| Postcondition | The application pops up alert to identify saving for the student.  The application saves the content in a db.  The application shows up a list page of a bulletin board |

**3.3.2 Functional Requirement 2: Read contents**

|  |  |
| --- | --- |
| Function name | Read contents |
| Goal in context | Student or Master reading a content in a bulletin board |
| Trigger | Student or Master clicks one of the titles in a bulletin board page |
| Precondition | Student or Master logins the app  Student or Master accesses into a bulletin board page |
| Basic Path | 1. Student or Master clicks the title of content. |
| Alternative Paths |  |
| Postcondition | The application display the content |

**3.3.3 Functional Requirement 3: Edit content**

|  |  |
| --- | --- |
| Function name | Edit content |
| Goal in context | Student edit the content or the title of which themselves wrote |
| Trigger | Student clicks “editing” button in a content page |
| Precondition | Student logins the app  Student wrote this content before (Student has a ownership for the content)  Student accesses the content page |
| Basic Path | 1. The student clicks “editing” button.  2. The application displays a editing page.  3. The student edits the title or content.  4. The student chooses if student clicks “finish” button or “cancel” button.  5. If student clicks “finish” button, the application checks the validation of a title or content.  6. If the validation is passed, writing action is done. |
| Alternative Paths | In step 4, if the student clicks “cancel” button, the application stops editing function and shows up a list page of a bullentin board.  In step 6, the validation is not passed, the application displays  alert for which one student should input again. Return to step 3 |
| Postcondition | The application pops up alert to identify updating for the student.  The application updates the content in a db.  The application shows up a list page of a bulletin board. |

**3.3.4 Functional Requirement 4: Delete content**

|  |  |
| --- | --- |
| Function name | Delete content |
| Goal in context | Student delete a content of which themselves wrote |
| Trigger | Student clicks “delete” button in a content page |
| Precondition | Student logins the app  Student wrote this content before (Student has a ownership for the content)  Student accesses into the content page |
| Basic Path | 1. The student clicks “delete” button in a content page.  2. The application pops up a alert for student to delete this content once again.  3. If the student clicks “yes” button in the alert, the application deletes a content. |
| Alternative Paths | In step 3, if the student clicks “no” button in the alert, the application stops any actions and displays the content page. |
| Postcondition | The application deletes all about the content in the db.  The application displays the list page after deleting the content. |

**3.3.5 Functional Requirement 5: Reply for a content**

|  |  |
| --- | --- |
| Function name | Reply for a content |
| Goal in context | Student replies for a content |
| Trigger | Student writes text in a reply text box |
| Precondition | Student logins the app  Student accesses the content page |
| Basic Path | 1. Student writes text in a reply text box.  2. Student clicks “reply” button. |
| Alternative Paths |  |
| Postcondition | The application saves the reply in a db.  The application resets for a blank in a reply text box.  The application displays the content page. |

**3.3.6 Functional Requirement 6: Edit reply**

|  |  |
| --- | --- |
| Function name | Edit reply |
| Goal in context | Student edits the reply |
| Trigger | Student clicks “edit” button in a reply place |
| Precondition | Student logins the app  Student accesses the content page  Student wrote this reply before (Student has a ownership for the reply) |
| Basic Path | 1. Student clicks “edit” button in a reply place.  2. Student application displays a editing page.  3. Student student edits the reply.  4. Student student chooses if student clicks “finish” button or “cancel” button.  5. If student clicks “finish” button, the application checks the validation of a content.  6. If the validation is passed, editing action is done. |
| Alternative Paths | In step 4, if the student clicks “cancel” button, the application stops editing function and shows up a content  In step 6, the validation is not passed, the application displays alert for which one student should input again. Return to step 3. |
| Postcondition | The application saves the reply in a db.  The application resets for a blank in a reply text box.  The application displays the content page. |

**3.3.7 Functional Requirement 7: Delete reply**

|  |  |
| --- | --- |
| Function name | Delete reply |
| Goal in context | Student deletes reply |
| Trigger | Student clicks “delete” button in a reply place |
| Precondition | Student logins the app  Student accesses the content page  Student wrote this reply before (Student has a ownership for the reply |
| Basic Path | 1. The student clicks “delete” button in a content page.  2. The application pops up a alert for student to delete this reply once again.  3. If the student clicks “yes” button in the alert, the application deletes the reply. |
| Alternative Paths | In step 3, if the student clicks “no” button in the alert, the application stops any actions and displays the content. |
| Postcondition | The application deletes the reply in a db.  The application resets for a blank in a reply text box.  The application displays the content page. |

**3.3.8 Functional Requirement 8: Search content**

|  |  |
| --- | --- |
| Function name | Search content. |
| Goal in context | Student or Master searches a content which student wants to |
| Trigger | Student or Master accesses the list page of bulletin board. |
| Precondition | Student or Master logins the app.  Student or Master accesses into a bulletin board page |
| Basic Path | 1. The student or Master chooses how to search the content. The choices are by Author, by keyword, and by name of a title .  2. If the search is by Author, the student or Master selects the dropdown menu to author.  3. The student or Master inputs the name of author on the input text box.  4. The student or Master clicks “find” button. |
| Alternative Paths | In step 2, if the student or Master selects to search by keyword, the student or Master selects the dropdown menu to keyword.  3. The student or Master inputs the keyword on the input text box. Return to step 4.  In step 2, if the student or Master selects to search by keyword, the student or Master selects the dropdown menu to a name of title.  3. The student or Master inputs the name of title on the input text box. Return to step 4. |
| Postcondition | The application creates and presents a list of all contents in the db. |

**3.3.9 Functional Requirement 9: Delete content**

|  |  |
| --- | --- |
| Function name | Delete content (by master). |
| Goal in context | Master deletes the content |
| Trigger | Master clicks “delete” button |
| Precondition | Master logins the app.  Master access into the content page |
| Basic Path | 1. The master clicks “delete” button in a content page.  2. The application pops up an alert for master to delete this content once again.  3. If the master clicks “yes” button in the alert, the application deletes a content |
| Alternative Paths | In step 3, if the master clicks “no” button in the alert, the application stops any actions and displays the content page |
| Postcondition | The application deletes the content in a db.  The application displays the bulletin board page. |

**3.3.10 Functional Requirement 10: Delete reply**

|  |  |
| --- | --- |
| Function name | Delete reply (by master). |
| Goal in context | Master deletes the reply |
| Trigger | Master clicks “delete” button |
| Precondition | Master logins the app.  Master into a bulletin board page |
| Basic Path | 1. The master clicks “delete” button in the content page.  2. The application pops up a alert for master to delete this reply once again.  3. If the master clicks “yes” button in the alert, the application deletes the reply. |
| Alternative Paths | In step 3, if the master clicks “no” button in the alert, the application stops any actions and displays the content. |
| Postcondition | The application deletes the reply in a db.  The application displays the content page |

**3.4 Use case 4: Bus Schedule**

**3.4.1 Functional Requirement 1: Show current shuttle bus schedule**

|  |  |
| --- | --- |
| Function name | Show current shuttle bus schedule |
| Goal in context | User checks current shuttle bus schedule |
| Trigger | User clicks the ‘bus schedule’button |
| Precondition | Run 'HYUⓔmini' application |
| Basic Path | 1. User can see two tabs(Shuttle bus and Commute bus). And default is shuttle bus tab.  2. ’Shuttle Bus Schedule’ shows shuttle bus schedule in graphical format.(Bus on operating is showed with tracking the current location of the bus) |
| Alternative Paths | In step 1, It ‘Commute bus schedule’ tab is clicked, User can see the commute bus schedule.  In step 2, If ‘Entire shuttle schedule’ is clicked, User can see whole shuttle bus schedule. |
| Postcondition | Shuttle bus schedule is provided in visualized format |

**3.4.2 Functional Requirement 2: Show entire shuttle bus schedule for semester term**

|  |  |
| --- | --- |
| Use Case Name | Show entire shuttle bus schedule for semester term |
| Goal in context: | User gets entire shuttle bus schedule for semester |
| Trigger | User clicks the ‘Entire shuttle schedule’ button |
| Precondition | User checked shuttle bus schedule |
| Basic Path | 1. User clicks ‘Entire shuttle schedule’ button to check entire shuttle bus schedule. |
| Alternative Paths |  |
| Post Condition | Entire ‘Semester’ shuttle bus schedule is provided in visualized format |

**3.4.3 Functional Requirement 3: Show entire shuttle bus schedule for session term**

|  |  |
| --- | --- |
| Use Case Name | Show entire shuttle bus schedule for session term |
| Goal in context: | User gets entire session shuttle bus schedule |
| Trigger | User clicks the ‘Session’ tab |
| Precondition | User checked semester or break shuttle bus schedule |
| Basic Path | 1. User can see shuttle bus schedule of session term |
| Alternative Paths |  |
| Post Condition | Entire ‘Session’ shuttle bus schedule is provided in visualized format |

**3.4.4 Functional Requirement 4: Show entire shuttle bus schedule for break term**

|  |  |
| --- | --- |
| Use Case Name | Show entire shuttle bus schedule for break term |
| Goal in context: | User gets entire break shuttle bus schedule |
| Trigger | User clicks the ‘Break’ tab |
| Precondition | User checked semester or session shuttle bus schedule |
| Basic Path | 1. User can see shuttle bus schedule of break term |
| Alternative Paths |  |
| Post Condition | Entire ‘Break’ shuttle bus schedule is provided in visualized format |

**3.4.5 Functional Requirement 5: Show commute bus schedule**

|  |  |
| --- | --- |
| Use Case Name | Show commute bus schedule |
| Goal in context | User gets commute bus schedule |
| Trigger | User clicks the ‘commute bus schedule’ tab |
| Precondition | User checked shuttle bus schedule |
| Basic Path | 1. User clicks commute bus schedule tab to check commute bus schedule.  2. In commute bus schedule tab, user can select which route to be displayed among 9 routes.  3. Application displays selected route. |
| Alternative Paths | In step2, If current time is not operation time, the page shows the earliest shuttle bus schedule. |
| Postcondition | Commute bus schedule is provided in visualized format |

**3.5 Use case 5: Lecture Schedule**

**3.5.1 Functional Requirement 1: Show lecture shcedule**

|  |  |
| --- | --- |
| Function name | Show lecture schedule |
| Goal in context | User show the user's schedule |
| Trigger | User clicks the 'Lecture Shcedule' button in the 'HYU@mini' |
| Precondition | User did login on'HYU@mini' |
| Basic Path | 1. User click 'Lecture Schedule' button  2. User see user's Lecture Schedule |
| Alternative Paths | In step 1, If user click another button (not lecture schedule) |
| Postcondition | User sees the converted lecture schedule of the entire week |

**3.5.2 Functional Requirement 2: Download Lecture Schedule**

|  |  |
| --- | --- |
| Function name | Download lecture Schedule |
| Goal in context | User download image file of lecture schedule in user's machine |
| Trigger | User clicks the 'download schedule' button |
| Precondition | User is placed to lecture schedule page in 'HYU@mini' |
| Basic Path | 1. User sees the two button such as 'Save', 'Cancel'  2. User sees the download progress bar and 'cancel' button  3. User sees the complete message  4. User can see image file of lecture schedule in local machine storage |
| Alternative Paths | In step 1, If user clicks the 'cancel' button, the system rollbacks the data and user sees lecture schedule web page of 'Hyu@mini'  In step 2, if user clicks the 'back' button, return to the previous page |
| Postcondition | Lecture schedule image is downloaded in users’ local machine storage |

**3.5.3 Functional Requirement 3: Share Lecture Schedule**

|  |  |
| --- | --- |
| Function name | Share lecture Schedule |
| Goal in context | User shares the lecture schedule with another people for project meeting and making plan in 'KakaoTalk' |
| Trigger | User clicks 'share' button |
| Precondition | User sees Lecture schedule with 'KakaoTalk' friend in the group room |
| Basic Path | 1. User sees to invite friend page in 'KakoTalk'  2. User check up Kakao friend icon and click the 'confirm' button  3. User sees the picture of lecture schedule in group 'KakaoTalk' room picture |
| Alternative Paths | Step 1 if user clicks the 'back' button, return to the previous page  Step 2 if user click the 'back' button, return to the lecture schedule in web page of 'HYU@mini'  Step 3,4 if users click the 'back' button, return to the previous page |
| Postcondition | User sees Lecture schedule with 'KakaoTalk' friend in the group room |

**3.5.4 Functional Requirement 4: Highlight view mode**

|  |  |
| --- | --- |
| Function name | Highlight View Mode |
| Goal in context | User sees the highlighted near time schedule of current time |
| Trigger | User clicks the 'Highlight' button |
| Precondition | User is placed to lecture schedule page in Hyu@mini |
| Basic Path | 1. User sees the highlighted near time schedule of current time |
| Alternative Paths | In step 1, If user clicks the 'back' button, return to the previous page |
| Postcondition | User sees the highlighted near time schedule of current time |

**3.5.5 Functional Requirement 5: Add User Schedule**

|  |  |
| --- | --- |
| Function name | Add User Schedule |
| Goal in context | User adds the personal schedule to current lecture schedule |
| Trigger | User Click 'Add Schedule' Button |
| Precondition | User is placed to lecture schedule page in HYU@mini |
| Basic Path | 1. User sees lecture schedule pop up page  2. User inserts personal schedule in pop up page  3. User sees the two button such as 'Save', 'Cancel'  4. User clicks the 'Save' button  5. User sees is updated lecture schedule page |
| Alternative Paths | In step 1 if user clicks the 'back' button, return to the previous page  In step 2 if user clicks the 'cancel' button, the system rollbacks and user sees lecture schedule web page of 'HYU@mini' |
| Postcondition | User sees the personal schedule to current lecture schedule |

* + 1. **Non-Functional Requirements**

To secure user informationthe system should protect itself and its sensitive data from unauthorized access, malicious, modification . if there is no reaction, the user will be log-out automatically.

To installation 'Kakao Talk' if 'Kakoa Talk' didn’t installed in local machine and mobile phone, user can’t access the Share lecture Schedule function in 'HYU@mini'

**3.6 Use case 6: Weekly Menu**

**3.6.1 Functional Requirement 1: Enter ‘Weekly Menu’.**

|  |  |
| --- | --- |
| Function name | Enter ‘Weekly Menu’ |
| Goal in context | Student see the menus of current week. |
| Trigger | Student clicks ‘Weekly Menu’ button. |
| Precondition |  |
| Basic Path | 1. The student sees day of week buttons, and default selection is current day of week.  2. The student sees mealtime buttons, and default selection is current mealtime. 3. The student sees meal type buttons at ‘Food Court’ panel, and default selection is ‘Korean’.  4. The student sees meal menus of cafeterias which are opened at the selected day of week and mealtime and meal type. |
| Alternative Paths | In step 1, if the selected day of week is weekend, the student doesn’t see mealtime buttons at step 2 and the student doesn’t see meal type buttons at step 3. |
| Postcondition | 1. The student sees meal menus of cafeterias which are opened at current time.  2. The student should be able to select another day of week, mealtime and meal type buttons. |

**3.6.2 Functional Requirement 2: Select day of week.**

|  |  |
| --- | --- |
| Function name | Select day of week |
| Goal in context | Student see menus of cafeterias which opened at selected day of week. |
| Trigger | Student clicks a day of week button. |
| Precondition | ‘Weekly Menu’ page is opened. |
| Basic Path | 1. Student selects a day of week.  2. Student see ‘Breakfast’ menus of opened at the selected day of week.  3. Student see ‘Korean’ menus at ‘Food Court’ panel. |
| Alternative Paths | In step 1, if selected day is weekend, then the student see menus of entire mealtime of ‘Dormitory Cafeteria’ only. |
| Postcondition | The application presents meal menus of cafeterias which are opened at the selected day of week. |

**3.6.3 Functional Requirement 3: Select mealtime.**

|  |  |
| --- | --- |
| Function name | Select mealtime |
| Goal in context | Student see menus of cafeterias which opened at selected mealtime. |
| Trigger | Student clicks a mealtime button. |
| Precondition | 1. ‘Weekly Menu’ page is opened.  2. Weekday selected. |
| Basic Path | 3. Student clicks a day of week button.  4. Student see menus which cafeteria opened at the selected day of week and selected mealtime. |
| Alternative Paths |  |
| Postcondition | The application presents meal menus of cafeterias which are opened at the selected day of week and mealtime. |

**3.6.4 Functional Requirement 4: Change cafeteria preference order.**

|  |  |
| --- | --- |
| Function name | Change cafeteria preference order |
| Goal in context | Student changes the order of cafeteria preference and see the menus order by preference. |
| Trigger | Student starts dragging or long press a panel of cafeteria. |
| Precondition | 1.‘Weekly Menu’ page is opened.  2.A weekday selected. |
| Basic Path | 1. Student starts dragging a panel of cafeteria.  2. End of dragging, the student’s cafeteria preference is changed order by top-down position of panels. |
| Alternative Paths | In step 2, if the position doesn’t changed, then the preference will not change. |
| Postcondition | The database updates the preference of the student and the application presents cafeteria panels order by the updated preference. |

**3.7 Use case 7: Management system**

**3.7.1 Functional Requirement 1: Add student information**

|  |  |
| --- | --- |
| Use Case Name | Add student information |
| Goal in context | Manager is able to add new information for new student. |
| Trigger | Manager clicks the ‘Add student’ button to add information. |
| Precondition | The new student cannot use his ID to access this system. |
| Basic Paths | 1. Manager gets information of new students. 2. Manager login the system. 3. Input the information. 4. Update information to database. |
| Alternative Paths | In step 3, we should verify the unique information in the database. |
| Post-condition | The information should be updated in database. |

**3.7.2 Functional Requirement 2: Delete student information**

|  |  |
| --- | --- |
| Use Case Name | Delete student information |
| Goal in context | The student who graduated should be deleted. |
| Trigger | Manager clicks the ‘delete student information’ button. |
| Precondition | Nobody uses this information which is useless. |
| Basic Paths | 1. The manager needs to check the student who is graduated. 2. The manager clicks the ‘delete’ button 3. The information should be deleted from database. 4. If the information deleted, the schedule of this student should be deleted. |
| Alternative Paths |  |
| Post-condition | Update the database. |

**3.7.3 Functional Requirement 3: Change schedule**

|  |  |
| --- | --- |
| Use Case Name | Change schedule |
| Goal in context | When the lectures time are changed, the schedule should be updated. |
| Trigger | The manager clicks the ‘change schedule’ button. |
| Precondition | The lecture time which is changed needs to update schedule in the system. |
| Basic Paths | 1. Some lectures changed the time. 2. The manager should click the ‘change schedule’ button to jump next page. 3. Changing the time of changed lecture. 4. Save the data to database. |
| Alternative Paths |  |
| Post-condition | The time of changed lectures should be updated. |

**3.7.4 Functional Requirement 4: Delete notes of BBS**

|  |  |
| --- | --- |
| Use Case Name | Delete notes of BBS |
| Goal in context | The manager wants to delete some useless or unhealthy notes from BBS. |
| Trigger | The manager clicks the ‘delete notes’ button to trigger this functional. |
| Precondition | Some useless notes should be deleted. |
| Basic Paths | 1. The manager found some useless notes. 2. Clicking ‘delete notes’ button into Delete Model.   3. The notes should be selected and click ‘delete ’ to delete. |
| Alternative Paths |  |
| Post-condition | Update database. The useless note should be deleted. |

**3.7.5 Functional Requirement 5 : Add notification in BBS**

|  |  |
| --- | --- |
| Use Case Name | Add notification in BBS |
| Goal in context | The manager posts the notification for all students in BBS. |
| Trigger | Clicking the ‘Edit’ button to add a new notification. |
| Precondition | The manager needs to post a news. |
| Basic Paths | 1. The manager clicks the button to add a new notification. 2. Input the title and content. 3. Clicking the ‘POST’ button to publish it in BBS. |
| Alternative Paths | In step 2, we can select category of content for absorbing student.  In step 3, We can add a function for saving unfinished content. |
| Post-condition | The notification will be post on top of BBS. |

**3.7.6 Functional Requirement 6: Update timetable of shuttle bus**

|  |  |
| --- | --- |
| Use Case Name | Update timetable of shuttle bus |
| Goal in context | The latest timetable should be updated. |
| Trigger | Clicking the ‘Update timetable’ button. |
| Precondition | The timetable of shuttle bus was changed. |
| Basic Paths | 1. The manager is able to click the textbox to directly change the time. 2. Clicking the ‘save’ to update the timetable. |
| Alternative Paths | Import the timetable by Excel file. |
| Post-condition | When other users check the timetable , it should be updated. |

**3.8 Use case 8: Page Switch**

**3.8.1 Functional Requirement 1: Click the button(page switching button)**

|  |  |
| --- | --- |
| Function name | Click the button(page switching button) |
| Goal in context | Student or master move page to page at one click |
| Trigger | Student or master click the button |
| Precondition | Student or master logged in successfully. |
| Basic Path | 1. Student or master click the button among 5 buttons (‘Weekly Menu’, ‘Time Table’, ‘School Bus Schedule’, ‘Mobile Student ID’, ‘BBS’)  2. If the student or master click ‘Weekly Menu’, the application shows ‘Weekly Menu’ page to the student. |
| Alternative Paths | In step 2, If the student or master click ‘Time Table’, the application shows ‘Time Table’ page to the student.  In step 2, If the student or master click ‘School Bus Schedule’, the application shows ‘School Bus Schedule’ page to the student.  In step 2, If the student or master click ‘Mobile Student ID’, the application shows ‘Mobile Student ID’ page to the student.  In step 2, If the student or master click ‘BBS’, the application shows ‘BBS’ page to the student. |
| Postcondition | Student or master move to the page that is connected with the button which student clicks. |

**3.8.2 Non-Functional Requirements**

* **User Interface**

The buttons are at the bottom of the application whatever page the application shows.

The buttons are showed in a row at the bottom of the application.