

STUDENT REGISTRATION SYSTEM

PART 2: REQUIREMENTS

COURSE: SOFTWARE ENGINEERING

Fall 2019

CLASS: MONDAY MORNING (LABORATORY)

SATURDAY MORNING (THEORY)

INSTRUCTOR: Ms. N.T.T.LOAN

**Member:**

Trịnh Minh Đăng ITITIU16022

Nguyễn Đức Phi Hồng ITITIU17022

Võ Huy Thành ITITIU17026

|  |
| --- |
| **MEMBER CONTRIBUTION** |

|  |  |  |
| --- | --- | --- |
| **NAME** | **ID** | **CONTRIBUTION** |
| **Nguyễn Đức Phi Hồng** | **ITITIU17022** | I. INTRODUCTION  II. GLOSSARY  III. USECASE AND USERSTORY  3.1. Register account  3.2. Login/Logout  3.3. Add student  IV. FUNCTIONAL REQUIREMENTS  V. NON-FUNCTIONAL REQUIREMENTS |
| **Trịnh Minh Đăng** | **ITITIU16022** | III. USECASE AND USERSTORY  3.4. Add/edit course  3.5. Add course (Student)  3.6. View student |
| **Võ Huy Thành** | **ITITIU17026** | III. USECASE AND USERSTORY  3.7. View enrolled course  3.8. View enrolled course (Admin)  3.9. Change Password |

**OUTLINE**

|  |
| --- |
|  |

I. INTRODUCTION

II. GLOSSARY

III. USECASE AND USERSTORY

IV. FUNCTIONAL REQUIREMENTS

V. NON-FUNCTIONAL REQUIREMENTS

**I. INTRODUCTION**

|  |
| --- |
|  |

The Requirement Specification focuses on the introduction of the project including the purpose, scope as well as the details of the system requirements.

Based on this requirements specification, implementation of each function including all the conditions as well as functional and non-functional requirements, supplied by the customers, will be made. This document helps the reader to understand the Online Retail Web Application by providing details into the product features.

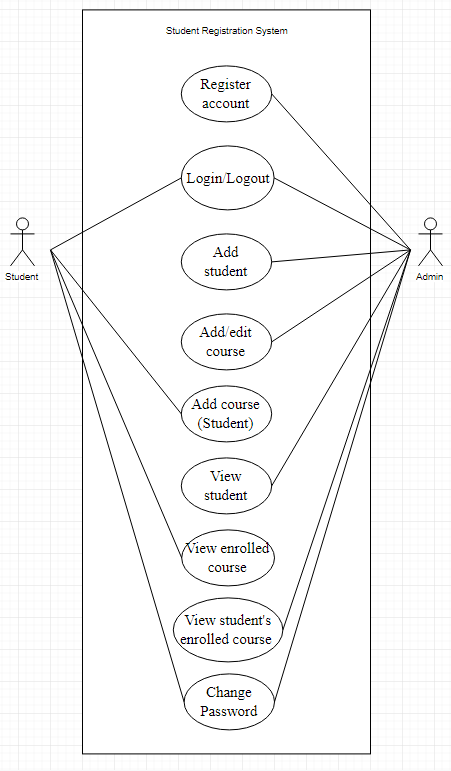
**II. GLOSSARY**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| **Use Case** | Identify the actors involved in an interaction and names the type of interaction. The additional information may be a textual description or one or more graphical models such as UML sequence or state charts. |
| **Functional Requirements** | Description of the system features to show how the software operates and processes the data from the user’s input. |
| **Non-functional Requirements** | Standards and constraints on the functionalities of the system to meet customer’s expectation. |

**III. USECASE AND USER STORIES**

**3.1 Use Case Diagram**



**3.2 Use Case Description (9 user stories)**

**Use case description for user story 1:** As an admin, I want to register a student so that I can have all student’s accessibilities.

- Name: Account Registration

- Identifier: UC1

- Input:

+Password

+StudentRegNo

-Output:

+If success: The successful message.

+If fail: The registration page with error message

-Pre-conditions:

+User must be in Home Page

-Basic Course:

|  |  |
| --- | --- |
| User: Admin | System |
| 1.Open Home Page | 1.1. Display Home Page |
| 2.Click on “Registration” button in Home Page | 2.1. Direct and display Registration Page |
| 3.Enter user’s information (include StudentRegNo, password) |  |
| 4. Click on “Sign Up” button | 4.1 Check the user’s input  4.1.1. Check empty  4.1.2. Check user’s name availability  4.2. Display the successful message |

-Alternative Path:

+Condition 1: StudentRegNo and password are invalid

\*In step 4, if StudentRegNo and password of a customer are invalid, the system will display an error message “Invalid input” and prompt admin to choose a different StudentRegNo / password or fill the empty section.

-Post-condition:

+New student’s account will be added into database.

**Use case description for user story 2**: As a student and admin, I want to login/logout my account.

**\*Use case description 2.1: Login**

-Name: Log in

-Identifier: UC2.1

-Input:

+StudentRegNo for Student or Username for Admin

+Password

-Output:

+Home page after successfully logged in.

-Pre-condition:

+User must have an account already.

-Basic course:

|  |  |
| --- | --- |
| User: Admin and student | System |
| 1.Open Home Page | 1.1. Display Home Page |
| 2. Click on “Sign in” button | 2.1 Direct and display Login Page |
| 3. Enter StudentRegNo for Student or Username for Admin and password |  |
| 4. Click on “Login” button | 4.1. Validate the input  4.2. Retrieve matching user from database  4.3. Record the result user object to session  4.4. Redirect user to the Home Page |

-Alternative Path:

+Condition 1: Users enter invalid StudentRegNo for Student or Username for Admin or password.

\*In step 4, after validating user’s input, system displays a warning message “Invalid Username or password” and ask users to enter again.

-Post-condition:

+Users successfully log in.

**\*Use case description 2.2: Log out**

-Name: Log out

-Identifier: UC2.2

-Input: None

-Output:

+Return home page after successfully logged out.

-Pre-condition:

+User has already logged in.

-Basic Course:

|  |  |
| --- | --- |
| User: Admin and student | System |
| 1.Click on “Log out” button in “account” bar | 1.1. Retrieve user object form session  1.2. Erase current user object from session  1.3. Redirect user to the home page |

-Alternative Path: None

-Post-condition:

+Users successfully log out.

**Use case description for user story 3**: As an admin, I want to add student.

-Name: Add student

-Identifier: UC3

-Input:

+Student‘s information

-Output:

+If success: The successful message.

+If fail: The error message

-Pre-condition:

+ User must log in as a admin

+ User must be in “Manage Student ” page

-Basic Course:

|  |  |
| --- | --- |
| User: Admin | System |
| 1.Click on “Manage Student ” button |  |
| 2.Input student‘s information |  |
| 3.Press “Enter” key | 3.1.Check valid input  3.2. Check empty input  3.3. Display the successful message 3.4. Save the project info in database |

-Alternative Path:

+Condition 1: User input invalid or empty input

\*In step 3, display “Student Info” with error message “Invalid Input” or “Empty input”.

-Post-condition:

+Save the project info in database

**Use case description for user story 4**: As an admin, I want to Add/edit courses to course list.

-Name: Add/Edit course

-Identifier: UC4

-Input:

+Course’s name

+Course’s id

+Course’s number of seat

+Course’s fee

+Course’s time

-Output:

+ Return “Course” Page with “Save product succefful” message.

-Pre-condition:

+ User must log in as a admin

+ User must be in “Course” page

-Basic Course:

|  |  |
| --- | --- |
| User: Admin | System |
| 1.Click “Course” button | 1.1.Display “Course” Page |
| 2.Enter course’s information (name, id, number of seat , fee, time) |  |
| 3.Click “Save” button | 3.1.Check valid input  3.2. Check empty input  3.3. Return “Course” Page with “Save product successful” message.  3.4. Save the project info in database |

-Alternative Path:

+Condition 1: User input invalid or empty input

\*In step 3, display “Course Info” with error message “Invalid Input” or “Empty input”.

-Post-condition:

+Save the project info in database

**Use case description for user story 5**: As a student, I want to add courses and remove course in my registration list.

**\*Use case description 5.1**: Add course (Student)

-Name: Add course (Student)

-Identifier: UC5

-Input: None

-Output:

+A pop-up that displays all course in the enrolled list with their information.

-Pre-condition:

+User must be in the “Enroll for course” Page.

+User must log in already

-Basic Course:

|  |  |
| --- | --- |
| User: Student | System |
| 1. Select course. |  |
| 2. Press “Enroll” button. | 2.1 Save the course in student’s enroll history. |

-Alternative Path:

+Condition 1: The course is out of seat

\*In step 2, if the course is not available, display error message”Out of Seat”.

-Post-condition:

+ Save the course in students enroll history

**\*Use case description 5.2**: Remove course (Student)

-Name: Remove course (Student)

-Identifier: UC5.2

-Input: None

-Output:

+ Display new enroll history in “Enroll history” Page.

-Pre-condition:

+User must be in the “Enroll history” Page.

+ User must log in already

-Basic Course:

|  |  |
| --- | --- |
| User: Student | System |
| 1.Press “Enroll history” button | 1.1. Display “Enroll history” Page |
| 2.Press “Delete” Button | 2.1. Remove an item from “Enroll history” Page  2.2. Update enrolled list |

-Alternative Path: None

-Post-condition:

+ Update enrolled list.

**Use case description for user story 6**: As an admin, I want to see details of student.

-Name: View student

-Identifier: UC6

-Input: None

-Output:

+All student information.

-Pre-condition:

+ User must log in as a admin

+ User must be in “Manage Student” page

-Basic Course:

|  |  |
| --- | --- |
| User: Admin | System |
| 1.Click on “Manage Student” button | 1.1 Display all student |
| 2.Click on “Edit” button next to a specific student | 2.1. Display all information of that student |

-Alternative Path: None

-Post-condition:

+All student’s information are listed out.

**Use case description for user story 7**: As a student, I want to see my enrolled course

-Name: View enrolled course (Student)

-Identifier: UC7

-Input: None

-Output:

+All enrolled course information.

-Pre-condition:

+ User must log in as a student

+ User must be in “Enroll history” page

-Basic Course:

|  |  |
| --- | --- |
| User: Student | System |
| 1.Click on “Enroll history” button | 1.1 Display enrolled course. |

-Alternative Path: None

-Post-condition:

+All enrolled course of that student are listed out.

**Use case description for user story 8**: As an admin, I want to see student’s enrolled course.

-Name: View enrolled course (Admin)

-Identifier: UC8

-Input: None

-Output:

+All enrolled course information of all student.

-Pre-condition:

+ User must log in as a admin

+ User must be in “Enroll history” page

-Basic Course:

|  |  |
| --- | --- |
| User: Admin | System |
| 1.Click on “Enroll history” button | 1.1 Display enrolled course of all student. |

-Alternative Path: None

-Post-condition:

+All student’s enrolled course are listed out.

**Use case description for user story 9**: As a student and admin, I want to change my password.

-Name: Change password

-Identifier: UC9

-Input:

+Old password

+New password

-Output:

+Home page after successfully logged in.

-Pre-condition:

+User must have an account already.

-Basic course:

|  |  |
| --- | --- |
| User: Admin and student | System |
| 1.Open Home Page | 1.1. Display Home Page |
| 2. Click on “Change password” button | 2.1 Direct and display “Change password” Page |
| 3. Enter old password , new password and confirm password |  |
| 4. Click on “Submit” button | 4.1 Check the user’s input  4.1.1. Check empty  4.2. Display the successful message |

-Alternative Path:

+Condition 1: Users enter invalid old password.

\*In step 4, after validating user’s input, system displays a warning message “Invalid password” and ask users to enter again.

-Post-condition:

+ New user’s password will be added into database.

**IV. FUNCTIONAL REQUIREMENTS**

|  |
| --- |
|  |

**UC1: Account Registration**

- Shall display a page that require the user to information such as name, year, and department.

- Shall check the name, StudentRegNo and password.

- Shall save the user’s registration information in database if successfully sign up.

- Shall check for the existed account in the database

**UC2: Login/Logout**

- Shall display Username and password bar for the user to sign in.

- Shall check the user’s input with the database.

- Shall notify the user if login successfully or failed.

- Shall redirect the user to the home page if login successfully.

- Shall display the logout icon in the user’s expansion menu if the user has logged in.

- Shall close all the user’s accessibilities after the user has logged out.

- Shall redirect user back to the homepage after the user has logged out successfully.

**UC3: Add student**

- Shall display the “Manage Student” button in the homepage.

- Shall allow the user to type in the information bars.

- Shall save the student information in database.

**UC4: Add/Edit course**

- Shall allow admin to modify course

- Shall save all changes made by admin to the database

**UC5: Add course (Student)**

- Shall display the “Course” button in the homepage.

- Shall check if the course is still have enough seat in the database, else it’s unable to add.

- Shall save the course the user added to the enroll history and in the database.

- Shall display the “Delete” button for the user to remove undesired course.

**UC6: View student**

- Shall display the list of student.

- Shall display the student’s information in the list of students.

- Shall display the “Delete” button for the user to remove undesired student.

**UC7: View enrolled course (Student)**

- Shall display the list of enrolled course.

- Shall display the course’s information the list of enrolled course.

- Shall display the “Delete” button for the user to remove undesired course.

**UC8: View enrolled course (Admin)**

- Shall display the list of enrolled course of all student.

- Shall display the course’s information in the list of enrolled course.

**UC9: Change password**

- Shall display the “Change password” button.

- Shall allow customer to enter their new password.

- Shall check for the user old password to allow the “submit” button to be clicked.

- Shall save the user’s new registration information in database if successfully change password.

|  |
| --- |
| **V. NON-FUNCTIONAL REQUIREMENTS** |

|  |  |
| --- | --- |
| **#1** | **Environment** |
| **Summary** | The system is built upon a reliable environment which must both be stable and upgradable. |
| **Rationale** | Developing a website requires a stable environment for securing both the user and the developer. Lack of stability may affect customer’s trust thus reduce the profit from the website. The environment also needs to be frequently updated for future improvement. |
| **Requirements** | The system is developed based on Xampp.  Database: using MySQL database. |

|  |  |
| --- | --- |
| **#2** | **Security** |
| **Summary** | All external communications between the system’s data server and clients must be encrypted |
| **Rationale** | Only registered user can use the system. Any attack from the outsiders will be prevented to ensure security. |
| **Requirements** | Security provided by Xampp. |

|  |  |
| --- | --- |
| **#3** | **Understandability** |
| **Summary** | Website UI needs to be simple, easy to use. |
| **Rationale** | As there are more customer register, the system must be simple for different types of user. |
| **Requirements** | Minimalist design: The UI appears easy to look at without tons of information, while still being understandable by the user. |

|  |  |
| --- | --- |
| **#4** | **Usability** |
| **Summary** | An interface should be easy to learn how to use and easy to remember how to use |
| **Rationale** | Information must be presented in a clear way that users should not be required to consult a manual for items. |
| **Requirements** | Efficiency of use: goals are easy to accomplish quickly and with few or no user errors.  Intuitiveness: the interface is easy to learn and navigate.  Low perceived workload: the interface appears easy to use, rather than intimidating, demanding and frustrating. |

|  |  |
| --- | --- |
| **#5** | **Reliability** |
| **Summary** | The product’s data must be full and must not return any inaccurate result. |
| **Rationale** | To provide the user with reliable and detail information about the product. Inaccurate data might lead to misunderstanding in product’s quality. |
| **Requirements** | Contact information visible on website  Accuracy of the search query result must be ensured. |

|  |  |
| --- | --- |
| **#6** | **Stability** |
| **Summary** | The system must be stable in order to deal with user’s growth. |
| **Rationale** | When the website becomes popular, there will be more people register. Thus, the website must be stable and maintain the loading speed to ensure the user experience. |
| **Requirements** | Database must be simple, sufficient and upgradable. |

|  |  |
| --- | --- |
| **#7** | **Supportability** |
| **Summary** | Developer must reply any questions from the user as soon as possible. |
| **Rationale** | When the user wants to contact the administrator or the administrator want to contact the development team, the website must provide a method that is quick and reliable for better communication. |
| **Requirements** | A contact page for the user to ask for support. |