**MINISTRY OF EDUCATION AND TRAINING**

|  |
| --- |
| **FPT UNIVERSITY** |
| **Capstone Project Document** |
| **Course-Scheduling** |
|  |
| |  |  | | --- | --- | | **Course Scheduling Team** | | | **Group Member** | Nguyễn Sơn Hà – SE02618  Hoàng Minh Đức – SE02551  Lưu Thành Công – SE02543  Nguyễn Văn Hoài – SE90191 | | **Supervisor** | Nguyễn Tất Trung  Phạm Quang Dũng | | **Capstone Project code** | CourseScheduling |   Hanoi, 24/5/2015 Acknowledgements |

# Definitions and Acronyms

# Contents

[Acknowledgements 2](#_Toc421134246)

[Definitions and Acronyms 2](#_Toc421134247)

[Contents 3](#_Toc421134248)

[1 CHAPTER 1: INTRODUCTION 4](#_Toc421134249)

[1.1 Purpose 4](#_Toc421134250)

[1.2 Project Information 4](#_Toc421134251)

[1.3 The People 4](#_Toc421134252)

[1.3.1 Supervisor 4](#_Toc421134253)

[1.3.2 Team members 5](#_Toc421134254)

[1.4 Background 6](#_Toc421134255)

[1.5 Literature Review 6](#_Toc421134256)

[1.6 Ideas 6](#_Toc421134257)

[1.7 Proposal of System 6](#_Toc421134258)

[1.7.1 Our proposal system 6](#_Toc421134259)

[1.7.2 Introduction about our proposal system 6](#_Toc421134260)

[2 Chapter 2. Project Management Plan 7](#_Toc421134261)

[2.1 Purpose 7](#_Toc421134262)

[2.2 Project Organization 7](#_Toc421134263)

[2.2.1 Our Proposal System 7](#_Toc421134264)

[2.2.2 Roles and Responsibility 7](#_Toc421134265)

[2.3 Project Management Plan 9](#_Toc421134266)

[2.3.1 Task 9](#_Toc421134267)

[2.3.2 Meeting Minutes 9](#_Toc421134268)

[2.3.3 Coding Conventions 9](#_Toc421134269)

[2.3.4 Risk Management Plan 10](#_Toc421134270)

[2.3.5 Risk probability and impact 11](#_Toc421134271)

[2.3.6 Communication Plan 11](#_Toc421134272)

[3 CHAPTER 3. SOFTWARE REQUIREMENT SPECIFICATION 12](#_Toc421134273)

[3.1 Purpose 12](#_Toc421134274)

[3.2 Functional Requirements 12](#_Toc421134275)

[3.2.1 Use Cases Diagram 12](#_Toc421134276)

[3.2.2 Use Cases List 12](#_Toc421134277)

[3.2.3 Business Rules 14](#_Toc421134278)

[3.2.4 Use Cases 15](#_Toc421134279)

[3.3 Non-functional Requirements 27](#_Toc421134280)

[3.3.1 Security 27](#_Toc421134281)

[3.3.2 Maintainability & Extensibility 27](#_Toc421134282)

[3.3.3 Performance 27](#_Toc421134283)

[3.4 Entity Relationship Diagram 28](#_Toc421134284)

# CHAPTER 1: INTRODUCTION

## Purpose

The introduction chapter provides main information of project about team’s members and project’s objectives.

## Project Information

* Project name: **Course-Scheduling**
* Project code: **Course-Scheduling**
* Project group name **:**
* Product type: **Website Application**
* Timeline: **From 10th May 2015 to 30thAug 2015**

## The People

### Supervisor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Full name** | **Phone** | **E-Mail** | **Title** |
| Supervisor | Nguyễn Tất Trung | 0904399139 | [*trungnt@fpt.edu.vn*](mailto:trungnt@fpt.edu.vn) | Lecturer |
| Supervisor | Phạm Quang Dũng | 0947732280 | [*dungpq1@fsoft.com.vn*](mailto:dungpq1@fsoft.com.vn) |  |

*Table 1.1: Supervisor information*

### Team members

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Full name** | **Student code** | **Phone** | **E-mail** | **Role in Group** |
| 1 | **Nguyễn Sơn Hà** | SE02618 | 0945488693 | [*hansse02618@fpt.edu.vn*](mailto:hansse02618@fpt.edu.vn) | Leader |
| 2 | **Hoàng Minh Đức** | SE02551 | 0904634667 | [*duchmse02551@fpt.edu.vn*](mailto:duchmse02551@fpt.edu.vn) | Member |
| 3 | **Lưu Thành Công** | SE02543 | 01656083093 | [*congltse02543@fpt.edu.vn*](mailto:congltse02543@fpt.edu.vn) | Member |
| 4 | **Nguyễn Văn Hoài** | SE90191 | 01659029841 | [*hoainvse90901@fpt.edu.vn*](mailto:hoainvse90901@fpt.edu.vn) | Member |

*Table 1-2: Team member’s information*

## Background

## Literature Review

## Ideas

Scheduling Courses is always one of the hardest work for FPT University because of conflicts about time of teachers and the change of timetable in every semester. Our project will apply algorithms to do this work automatically by computer and also provide a tool to help them create timetable easer in the website.

## Proposal of System

### Our proposal system

### Introduction about our proposal system

# Chapter 2. Project Management Plan

## Purpose

This chapter describes about project team’s structure and plan of the project. All members will use this chapter as a guideline to work in this project.

## Project Organization

### Our Proposal System

This figure above describes the information and products flow lifecycle process model. Course-Scheduling project uses the Iterative and Incremental Software Process Model.

The Iterative and Incremental Software Process Model is most use when the scope of the project is big, the major requirements were defined clearly, some more detail will be added in time, and for the newbie group in software development. By using this software process model, we break down the developing system task into series of smaller tasks which be completed separately, evaluated, and subsequently re-worked until the system’s performance adequately. In addition, the iterative model is easier than other models when the issues are discovered. They are fed back to the team, and solutions found while the project is still in development.

### Roles and Responsibility

#### Organization Structure

|  |  |
| --- | --- |
| **Role** | **Responsibility** |
| Project Manager | Planning and defining scope, developing schedules, allocating resources, coordinating communication, generally responsible for keeping the team’s focus on main goal, and tries to keep the project team focused on the right goal at a time. |
| Technical Leader | Responsible for the underlying architecture for the software program, assigning tasks, mentoring people, reporting and updating regular associates RAG status. Also technical leader is a reference book for other software programmers. |
| Business Analyst | Eliciting requirement, analyzing and documenting requirements, communicate requirements, identifying solution and verifying solution meets the requirements, leads and coordinates use-case modeling. Outline the system’s functionality. |
| Test Leader | Responsible for test execution, including test set-up and test run, evaluation of test run and error recovery, defect logging and test results recording. |
| Developer | Involve to code product |
| Designer | Involve to design product |
| Tester | Involve to test product |

#### Project Team Members

|  |  |
| --- | --- |
| **Team Member** | **Role** |
| HaNS | Project Manager, Developer, Tester, Designer |
| DucHM | Technical Leader, Test Leader, Developer, Tester, Designer |
| CongLT | Business Analyst, Developer, Tester |
| HoaiNV | Business Analyst, Developer, Tester |

#### Tools and Techniques

* **Programming languages:** Java, HTML5, CSS3, JQuery.
* **Framework:** MVC Spring
* **Software Architecture:** Web Service (REST architecture).
* **Process Model:** Iterative and Incremental Software Process Model.
* **Version Control:** Github.
* **IDEs**: Spring Tool Suite
* **DBMS**: MySQL v5.5
* **UML tools:** Enterprise Architect v12, Astah Professional v6.9.0
* **Manage tools:**  Microsoft Project Professional 2013.
* **Web server:** Apache2.0.

## Project Management Plan

### Task

Refer to file Course-Scheduling.mpp

### Meeting Minutes

All meeting minutes will be written follow this template:

| ***Meeting/Project Name:*** | *CourseScheduling* | | | | |
| --- | --- | --- | --- | --- | --- |
| ***Date of Meeting:*** | *01/06/2015* | | ***Time: (Type)*** | *3 hours (Skype)* | |
| ***Meeting Called By:*** | *HaNS* | | ***Location:*** | *B109* | |
| ***Note Taker:*** | *DucHM* | | ***Time Keeper:*** | *CongLT* | |
| 1. Meeting Objective: | | | | | |
| Brainstorming all functions of systems | | | | | |
| 2. Attendance | | | | | |
| ***Name*** | ***Roles*** | ***E-mail*** | | | ***Phone*** |
| Nguyen Son Ha | Project Manager | [hansse02618@fpt.edu.vn](mailto:hansse02618@fpt.edu.vn) | | | 0945488693 |
| Hoang Minh Duc | Technical Leader, Test Leader | [duchmse02551@fpt.edu.vn](mailto:duchmse02551@fpt.edu.vn) | | | 0904634667 |
| Luu Thanh Cong | Business Analyst | [congltse02543@fpt.edu.vn](mailto:congltse02543@fpt.edu.vn) | | | 01656083093 |
| Nguyen Van Hoai | Business Analyst | [hoainvse90191@fpt.edu.vn](mailto:hoainvse90191@fpt.edu.vn) | | | 01659029841 |
| 3. Content: | | | | | |
|  | | | | | |

### Coding Conventions

Follow Java coding convention. See the reference for detail:

<http://www.oracle.com/technetwork/java/codeconvtoc-136057.html>

### Risk Management Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Name** | **Root Cause** | **Avoidance plan** | **Contingency** |
| R1 | **Member**  **un-cooperating** | Conflict between team members. | Team building to connect member with each others. | Finding root conflict (social relationship, private conflict) |
| R2 | **Team member do not have enough appropriate skill, experience** | All member are students. | When developing project plan, estimating more time for study new technique. | Trying to contact with expertise to learn from their experience and get their advices. |
| R3 | **Changing in the way of executing** | Creative members. |  | Meeting for evaluating, and changing the way to execute the project if it's really better. |
| R4 | **Over deadline** | Low responsibility in working of members | Estimate time to plan task appropriate with ability of team members and hold daily meeting to keep track on project status day by day. | Other team members will support to help the task can complete in time. |
| R5 | **Low quality infrastructure of the internet** | Incident |  | Use other internet like 3G. |
| R6 | **Lack of real data** | Can not contact with people who can provide team the real data. |  | Create dummy data based on the knowledge about this business of team members. |

### Risk probability and impact

High

|  |  |  |
| --- | --- | --- |
|  | R5 |  |
|  | R1 | R4 |
| R3 | R2 | R6 |

Probability

Medium

Low

High

Medium

Low

Impact

*Figure 2.2. Risk probability and impact*

### Communication Plan

*Weekly meeting schedule:* By using Iterative and Incremental Process Model, Footprints Project system will be divided into a series of small tasks, each task will be assigned to team members by Technical Leader and depend on difficulty, and Technical Leader will assign deadlines for each task. We will have meeting every Tuesday and Thursday to keep track about our works in a week then discuss about the tasks we have to done in next week.

*Unscheduled meeting*: If any unplanned or important situation happened, we have to hold a meeting to find solution or discuss about what we need to do in this situation.

*Communication channel*: Our main communication channels are face-to-face meeting, e-mail and Facebook.

# CHAPTER 3. SOFTWARE REQUIREMENT SPECIFICATION

## Purpose

## Functional Requirements

### Use Cases Diagram

Admin, Staff, Teacher, Student

### Use Cases List

|  |  |
| --- | --- |
| Primary Actor | Use Cases |
| Staff, Teacher, Student | UC-01. Login |
| Staff, Teacher, Student | UC-02. Logout |
| Staff | UC-03. View All Staffs |
| Staff | UC-04. Add Staff |
| Staff | UC-05. Remove Staff |
| Staff | UC-06. Search Staffs |
| Staff | UC-08. View All Rooms |
| Staff | UC-09. Add Room |
| Staff | UC-10. Add List of Rooms |
| Staff | UC-11. Update Room |
| Staff | UC-12. Remove Room |
| Staff | UC-13. Search Rooms |
| Staff | UC-15. View All Semesters |
| Staff | UC-16. Add Semester |
| Staff | UC-17. Update Semester |
| Staff | UC-18. Search Semester |
| Staff | UC-20. View All Teachers |
| Staff | UC-21. Add Teacher |
| Staff | UC-22. Add List of Teachers |
| Staff | UC-23. Update Teacher |
| Staff | UC-24. Remove Teacher |
| Staff | UC-26. Search Teachers |
| Staff | UC-28. View All Students |
| Staff | UC-29. Add Student |
| Staff | UC-30. Add List of Students |
| Staff | UC-31. Update Student |
| Staff | UC-32. Remove Student |
| Staff | UC-34. Search Students |
| Staff | UC-36. View All Courses |
| Staff | UC-37. Add Course |
| Staff | UC-38. Add List of Courses |
| Staff | UC-39. Update Course |
| Staff | UC-40. Remove Course |
| Staff | UC-41. Search Courses |
| Staff | UC-43. View All Classes |
| Staff | UC-44. Add Class |
| Staff | UC-45. Add List of Classes |
| Staff | UC-46. Update Class |
| Staff | UC-47. Remove Class |
| Staff | UC-48. Search Classes |
| Staff | UC-51. Change Student’s Semester |
| Staff | UC-52. Update Student’s Learning Progress |
| Staff | UC-53. Student into Classes |
| Staff | UC-55. Create Timetable |
| Staff, Teacher | UC-56. Choose Teachers for Classes |
| Staff | UC-57. View All Timetable |
| Teacher, Student | UC-58. View Personal Timetable |
| Teacher | UC-59. View A Class Timetable |

### Business Rules

|  |  |
| --- | --- |
| **No** | **Description** |
| **B1** | The account must not be empty |
| **B2** | The account must be the account of mail |
| **B3** | The e-mail must not be empty |
| **B4** | The e-mail must be fpt university’s mail (fpt.edu.vn) |
| **B5** | Room’s code |
| **B6** | Room’s type |
| **B7** | Room’s capacity |
| **B8** | Semester’s Code |
| **B9** | Semester’s Name |
| **B10** | Semester’s Finish Date |

### Use Cases

#### Admin

##### UC-01. Login

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-01. Login | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff, Teacher, Student** | Secondary Actors: | N/A |
| Trigger: | Actor wants to login to Course Scheduling. | | |
| Description: | Actor logins to website to use its function. | | |
| Pre-conditions: | Actor has a fpt’s mail (…@fpt.edu.vn) and this account was registered in server. | | |
| Post-conditions: | Actor logins successfully. | | |
| Normal Flow: | **UC – 01. Login.**   1. Actor clicks “Login” button. 2. A dialog from google will be shown. 3. Actor choose an account they will use to login. 4. System indicates this account. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | * 1. **Account Not Found Exception**  1. Course-Scheduling will show a dialog to inform the account does not exist. | | |
| Priority: | Low | | |
| Frequency of Use: | Medium | | |
| Business Rules: | **N/A** | | |
| Other Information: | N/A | | |
| Assumptions: | N/A | | |

##### UC-02. Logout

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-01. Logout | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff, Teacher, Student** | Secondary Actors: | N/A |
| Trigger: | Actor wants to logout to Course Scheduling. | | |
| Description: | Actor logouts to website after using its function. | | |
| Pre-conditions: | 1. Actor logged in successfully. | | |
| Post-conditions: | Actor logouts successful. | | |
| Normal Flow: | **UC – 02. Logout.**   1. In main page, Actor clicks “Logout” button. 2. Actor logout. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | N/A | | |
| Priority: | Low | | |
| Frequency of Use: | Low | | |
| Business Rules: | **N/A** | | |
| Other Information: | N/A | | |
| Assumptions: | N/A | | |

##### UC-03. View All Staffs

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-03. View All Staffs | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the staffs. | | |
| Description: | Actor uses this function to see all the staffs. | | |
| Pre-conditions: | **PRE – 03.01.** Actor is logged to Course-Scheduling Website.  **PRE – 03.02.** Actor is set as the admin of the Website. | | |
| Post-conditions: | **POST – 03.01.** Course-Scheduling shows list of all staffs. | | |
| Normal Flow: | **UC – 03. View All Staffs**   1. Actor clicks to menu “*Staffs Management*”. 2. Course-Scheduling shows list of all Staffs. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-04. Add Staff

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-04. Add Staff | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new staff to use the web. | | |
| Description: | Actor uses this function to add new staff to use the web. | | |
| Pre-conditions: | **PRE – 04.01.** Actor is logged to Course-Scheduling Website.  **PRE – 04.02.** Actor is set as the staff of the Website.  **PRE – 04.03.** Actor accessed to “*Staffs Management*” page. | | |
| Post-conditions: | **POST – 04.01.** The staff was added will be shown in list of all staffs of Staff page. | | |
| Normal Flow: | **UC – 04. Add Staff**   1. Actor clicks to button "Add Staff”. 2. The actor enters information about new staff: “Account”, “Name”, “E-mail”. 3. Actor click “Save” button. 4. Course-Scheduling will validate all information, if it’s correct, the staff will be added to database. 5. Course-Scheduling redirects to Staff page with message “*Add Staff Successful.*”. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B1, B2, B3, B4** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-05. Remove Staff

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-05. Remove Staff | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a staff. | | |
| Description: | Actor uses this function to remove a staff. | | |
| Pre-conditions: | **PRE – 05.01.** Actor is logged to Course-Scheduling Website.  **PRE – 05.02.** Actor is set as the staff of the Website.  **PRE – 05.03.** Actor accessed to “*Staffs Management*” page. | | |
| Post-conditions: | **POST – 05.01.** The staff was removed must not be shown in list of all staffs of Staff page. | | |
| Normal Flow: | **UC – 05. Remove Staff**   1. Actor clicks “Delete” of a staff in list of staffs. 2. Course-Scheduling asks for confirmation to remove this staff. 3. Actor chooses “Accept” 4. Course-Scheduling removes this staff in database. 5. Course-Scheduling redirects to Staff page with message “*Remove Staff Successful.*”. | | |
| Alternative Flows: | **AF – 05.01. At step 5, Actor chooses “Decline”**   1. Actor chooses “Decline” in the dialog. 2. Course-Scheduling redirects to Staff page. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-06. Search Staffs

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-06. Search Staff | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific staff. | | |
| Description: | Actor uses this function to search for a specific staff in the website. | | |
| Pre-conditions: | **PRE – 06.01.** Actor is logged to Course-Scheduling Website.  **PRE – 06.02.** Actor is set as the staff of the Website.  **PRE – 06.03.** Actor accessed to “*Staffs Management*” page. | | |
| Post-conditions: | **POST – 06.01.** The staffs have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 06. Search** **Staff**   1. Actor enters information about the staff that they want to find in search box. 2. Course-Scheduling shows list of staffs that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-07. View All Buildings

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-07. View All Buildings | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all buildings. | | |
| Description: | Actor uses this function to see all buildings. Actor can access to “*Buildings Management*” page in many ways:  . In “*Main*” page, click to menu “*Buildings Management*”.  . In “*General Management*” pages (*“Staffs Management”, “Rooms Management”, “Specialized Management”, “Semesters Management”, “Departments Management”, “Programs Management”, “Buildings Management”)* click to tab “Building” in sidebar. | | |
| Pre-conditions: | **PRE – 07.01.** Actor is logged to Course-Scheduling Website.  **PRE – 07.02.** Actor is set as the staff of the Website. | | |
| Post-conditions: | **POST – 07.01.** Course-Scheduling shows list of all buildings. | | |
| Normal Flow: | **UC – 07. View All Staffs**   1. Actor clicks to menu “*Buildings Management*”. 2. Course-Scheduling shows list of all *Buildings*. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-08. Add Building

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-08. Add Room | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new room. | | |
| Description: | Actor uses this function to add new room. | | |
| Pre-conditions: | **PRE – 08.01.** Actor is logged to Course-Scheduling Website.  **PRE – 08.02.** Actor is set as the staff of the Website.  **PRE – 08.03.** Actor accessed to “*Buildings Management*” page. | | |
| Post-conditions: | **POST – 08.01.** The room was added will be shown in list of all rooms in “*Buildings Management*” page. | | |
| Normal Flow: | **UC – 08. Add Building**   1. In Building page, Actor clicks to button "Add Building”. 2. The actor enters information about new Building: “Code. 3. Actor clicks save “button”. 4. Course-Scheduling will validate all information, if it’s correct, the Room will be added to database. 5. Course-Scheduling redirects to “*Buildings Management*” with message “*Add* Building *Successful.*”. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **08.01 Building Existed Exception (Duplicate code).**  . Course-Scheduling redirects to “*Buildings Management”* with error message *“This Building’s code existed! Please Try Again!”.*  **08.02 Code input is empty.**  . Course-Scheduling show message “*Your Code can not be empty!!!*” | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-09. Remove Building

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-09. Remove Building | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a building. | | |
| Description: | Actor uses this function to remove a building. | | |
| Pre-conditions: | **PRE – 09.01.** Actor is logged to Course-Scheduling Website.  **PRE – 09.02.** Actor is set as the staff of the Website.  **PRE – 09.03.** Actor accessed to “*Buildings Management*” page. | | |
| Post-conditions: | **POST – 09.01.** The building was removed must not be shown in list of all buildings of “*Buildings Management*” page. | | |
| Normal Flow: | **UC – 09. Remove Building**   1. Actor clicks to “Delete” in the row of a building in list of buildings. 2. Course-Scheduling asks for confirmation to remove this building. 3. Actor chooses “Accept” 4. Course-Scheduling removes this building in database. 5. Redirect to “*Buildings Management*” page with message: “*Remove Building successfully*”. | | |
| Alternative Flows: | **AF – 09.01. At step 3, Actor chooses “Decline”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-10. Search Buildings

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-10. Search Buildings | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific building. | | |
| Description: | Actor uses this function to search for a specific building. | | |
| Pre-conditions: | **PRE – 10.01.** Actor is logged to Course-Scheduling Website.  **PRE – 10.02.** Actor is set as the admin or staff of the Website.  **PRE – 10.03** Actor accessed to “*Buildings Management*” page. | | |
| Post-conditions: | **POST – 10.01.** The buildings have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 10. Search** **Buildings**   1. Actor enters information about the building that they want to find in the search box. 2. Course-Scheduling shows list of buildings that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-11. View All Rooms

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-11. View All Rooms | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the rooms. | | |
| Description: | Actor uses this function to see all the rooms. Actor can access to “*Rooms Management*” page in many ways:  . In “*Main*” page, click to menu “*Rooms Management*”.  . In “*General Management*” pages (*“Staffs Management”, “Rooms Management”, “Specialized Management”, “Semesters Management”, “Departments Management”, “Programs Management”, “Buildings Management”)* click to tab “*Rooms*” in sidebar. | | |
| Pre-conditions: | **PRE – 11.01.** Actor is logged to Course-Scheduling Website.  **PRE – 11.02.** Actor is set as the staff of the Website. | | |
| Post-conditions: | **POST – 11.01.** Course-Scheduling shows list of all rooms. | | |
| Normal Flow: | **UC – 11. View All Rooms**   1. Actor clicks to menu “*Rooms Management*”. 2. Course-Scheduling shows list of all Rooms. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-12. Add Room

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-12. Add Room | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new room. | | |
| Description: | Actor uses this function to add new room. | | |
| Pre-conditions: | **PRE – 12.01.** Actor is logged to Course-Scheduling Website.  **PRE – 12.02.** Actor is set as the staff of the Website.  **PRE – 12.03.** Actor accessed to “*Rooms Management*” page. | | |
| Post-conditions: | **POST – 12.01.** The room was added will be shown in list of all rooms in “*Rooms Management*” page. | | |
| Normal Flow: | **UC – 12. Add Room**   1. In Room page, Actor clicks to button "Add Room”. 2. The actor enters information about new room: “Building”, “Code”, “Capacity” and some specific “course” will be learnt in this room. 3. Actor clicks save “button”. 4. Course-Scheduling will validate all information, if it’s correct, the Room will be added to database. 5. Course-Scheduling redirects to Room page with message “*Add Room Successful.*”. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **12.01 Room Existed Exception (Duplicate code)**  . Course-Scheduling redirects to “*Rooms Management*” page with error message “*This Room’s code existed! Please try again!*”.  **12.02 Room’s code input is empty.**  . Course-Scheduling shows message *“Your Code can not be empty!!!”.*  **12.03 Room’s capacity input is empty.**  . Course-Scheduling shows message *“Your Capacity can not be empty!!!”.* | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B5, B6, B7** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-13. Add List of Rooms

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-13. Add List of Rooms | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add List of new room. | | |
| Description: | Actor uses this function to add List of new room. | | |
| Pre-conditions: | **PRE – 13.01.** Actor is logged to Course-Scheduling Website.  **PRE – 13.02.** Actor is set as the staff of the Website.  **PRE – 13.03.** Actor accessed to “*Rooms Management*” page. | | |
| Post-conditions: | **POST – 13.01.** The rooms were added will be shown in list of all rooms of “*Rooms Management*” page. | | |
| Normal Flow: | **UC – 13. Add List of Rooms**   1. In Room page, Actor clicks to button "Add List of Rooms”. 2. Course-Scheduling shows a dialog with an input to enter the link of excel file contains all information about rooms. 3. Actor enters the link of file. 4. Actor choose “Add”. 5. Course-Scheduling will validate all information, if it’s correct, the Rooms be added to database. 6. Course-Scheduling redirects to Room page with a message “*Import List of Rooms successful.*“. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **13.01. Building Not Found Exception.**  Course-Scheduling redirects to “*Rooms Management*” page with error message *“Building … are not found! Please Insert It First!”* | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B5, B6, B7** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-14. Edit Room

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-14. Edit Room | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to update information of a room. | | |
| Description: | Actor uses this function to update information of a room. | | |
| Pre-conditions: | **PRE – 14.01.** Actor is logged to Course-Scheduling Website.  **PRE – 14.02.** Actor is set as the admin or staff of the Website.  **PRE – 14.03.** Actor accessed to “*Rooms Management*” page. | | |
| Post-conditions: | **POST – 14.01.** The information was edited will be saved to database. | | |
| Normal Flow: | **UC – 14. Edit Room**   1. Actor clicks to “Edit” of a staff in list of rooms. 2. Course-Scheduling shows a dialog with information of this room for actor to edit. “Capacity” and “Course”. 3. Actor edit information of this room. 4. Actor click “Save”. 5. Course-Scheduling closes dialog and saves information. | | |
| Alternative Flows: | **AF – 14.01. At step 4, Actor chooses “cancel”.**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-16. Remove Room

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-15. Remove Room | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a room. | | |
| Description: | Actor uses this function to remove a room. | | |
| Pre-conditions: | **PRE – 15.01.** Actor is logged to Course-Scheduling Website.  **PRE – 15.02.** Actor is set as the staff of the Website.  **PRE – 15.03.** Actor accessed to “*Rooms Management*” page. | | |
| Post-conditions: | **POST – 15.01.** The room was removed must not be shown in list of all rooms of “*Rooms Management*” page. | | |
| Normal Flow: | **UC – 15. Remove Room**   1. Actor clicks to “Delete” in the row of a room in list of rooms. 2. Course-Scheduling asks for confirmation to remove this room. 3. Actor chooses “Accept” 4. Course-Scheduling removes this room in database. 5. Redirect to Room page with message: “*Remove Room successful!*”. | | |
| Alternative Flows: | **AF – 15.01. At step 3, Actor chooses “Decline”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-16. Search Rooms

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-16. Search Room | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific room. | | |
| Description: | Actor uses this function to search for a specific room. | | |
| Pre-conditions: | **PRE – 16.01.** Actor is logged to Course-Scheduling Website.  **PRE – 16.02.** Actor is set as the admin or staff of the Website.  **PRE – 16.03** Actor accessed to “*Rooms Management*” page. | | |
| Post-conditions: | **POST – 16.01.** The rooms have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 16. Search** **Room**   1. Actor enters information about the room that they want to find in the search box. 2. Course-Scheduling shows list of rooms that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-17. View All Semesters

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-17. View All Semesters | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the Semesters. Actor can access to “*Semesters Management*” page in many ways:  . In “*Main*” page, click to menu “*Semesters Management*”.  . In “*General Management*” pages (*“Staffs Management”, “Rooms Management”, “Specialized Management”, “Semesters Management”, “Departments Management”, “Programs Management”, “Buildings Management”)* click to tab “*Semesters*” in sidebar. | | |
| Description: | Actor uses this function to see all the Semesters. | | |
| Pre-conditions: | **PRE – 17.01.** Actor is logged to Course-Scheduling Website.  **PRE – 17.02.** Actor is set as the admin of the Website. | | |
| Post-conditions: | **POST – 17.01.** Course-Scheduling shows list of all Semesters. | | |
| Normal Flow: | **UC – 17. View All Semesters**   1. Actor clicks to menu “*Semesters Management*”. 2. Course-Scheduling shows list of all Semesters. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-18. Add Semester

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-18. Add Semester | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new Semester. | | |
| Description: | Actor uses this function to add new Semester. | | |
| Pre-conditions: | **PRE – 18.01.** Actor is logged to Course-Scheduling Website.  **PRE – 18.02.** Actor is set as the staff of the Website.  **PRE – 18.03.** Actor accessed to “*Semesters Management*” page. | | |
| Post-conditions: | **POST – 18.01.** The staff was added will be shown in list of all Semesters of “*Semesters Management*” page. | | |
| Normal Flow: | **UC – 18. Add Semester**   1. In “*Semesters Management*” page, Actor clicks to button "Add Semester”. 2. The actor enters information about new Semester: “Code”, “Name”, “Start Date”, “End Date”. 3. Actor clicks “Save” button. 4. Course-Scheduling will validate all information, if it’s correct, the Semester will be added to database. 5. Course-Scheduling redirects to “*Semesters Management*” page with message “*Add Semester Successful*!” | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **18.01 Semester Existed Exception (Duplicate code).**  **.** Course-Scheduling redirects to “*Semesters Management”* page with error message “*This Semester’s code existed! Please Try Again!*”.  **18.02 Semester’s code input is empty.**  . Course-Scheduling shows message “*Your Code can not be empty!!!*”.  **18.03 Semester’s name input is empty.**  **.** Course-Scheduling shows message “*Your Name can not be empty!!!*”.  **18.04 Semester’s start date input is empty.**  . Course-Scheduling shows message “*Your Start Date can not be empty!!!*”.  **18.05 Semester’s end date input is empty.**  . Course-Scheduling shows message “*Your End Date can not be empty!!!*”.  **18.06 Semester’s end date is a previous date of Semester’s start date.**  . Course-Scheduling shows message “*Your End Date must be greater than Start Date!!!*”. | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B8, B9, B10** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-19. Remove Semester

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-16. Remove Semester | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a semester. | | |
| Description: | Actor uses this function to remove a semester. | | |
| Pre-conditions: | **PRE – 19.01.** Actor is logged to Course-Scheduling Website.  **PRE – 19.02.** Actor is set as the staff of the Website.  **PRE – 19.03.** Actor accessed to “*Semesters Management*” page. | | |
| Post-conditions: | **POST – 19.01.** The semester was removed must not be shown in list of all semesters of “*Semesters Management*” page. | | |
| Normal Flow: | **UC – 19. Remove Semester**   1. Actor clicks to “Delete” in the row of a semester in list of Semesters. 2. Course-Scheduling asks for confirmation to remove this Semester. 3. Actor chooses “Accept” 4. Course-Scheduling removes this Semester in database. 5. Redirect to “*Semesters Management*” page with message: “*Remove Semester successful!*”. | | |
| Alternative Flows: | **AF – 19.01. At step 3, Actor chooses “Decline”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-20. Search Semester

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-20. Search Semester | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific Semester. | | |
| Description: | Actor uses this function to search for a specific Semester in the website. | | |
| Pre-conditions: | **PRE – 20.01.** Actor is logged to Course-Scheduling Website.  **PRE – 20.02.** Actor is set as the admin or staff of the Website.  **PRE – 20.03.** Actor accessed to “*Semesters Management*” page. | | |
| Post-conditions: | **POST – 20.01.** The Semesters have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 20. Search** **Semester**   1. Actor enters information about the Semester that they want to find in search box. 2. Course-Scheduling shows list of Semesters that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-18. View All Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-18. View All Teachers | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the Teachers. | | |
| Description: | Actor uses this function to see all the Teachers. | | |
| Pre-conditions: | **PRE – 18.01.** Actor is logged to Course-Scheduling Website.  **PRE – 18.02.** Actor is set as the admin or staff of the Website.  **PRE – 18.03.** Actor accessed to “*Teachers Management*” page. | | |
| Post-conditions: | **POST – 18.01.** Course-Scheduling shows list of all Teachers. | | |
| Normal Flow: | **UC – 18. View All Teachers**   1. Actor choose Semester in left panel. 2. Course-Scheduling shows list of all Teachers in the semester was chosen. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-19. Add Teacher

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-19. Add Teacher | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new teacher. | | |
| Description: | Actor uses this function to add new room. | | |
| Pre-conditions: | **PRE – 19.01.** Actor is logged to Course-Scheduling Website.  **PRE – 19.02.** Actor is set as the staff of the Website.  **PRE – 19.03.** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 19.01.** The teacher was added will be shown in list of all teachers of Teacher page. | | |
| Normal Flow: | **UC – 19. Add Teacher**   1. In Teacher page, Actor clicks to button "Add Teacher”. 2. The actor enters information about new room: “Code”, “Capacity” and some specific “course” will be learnt in this room. 3. Actor click “Save” button. 4. Course-Scheduling will validate all information, if it’s correct, the Room will be added to database. 5. Course-Scheduling redirects to Room page with message “*Add Teacher Successful.*”. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B1, B2, B3, B4** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-20. Add List of Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-20. Add List of Teachers | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add List of new teachers. | | |
| Description: | Actor uses this function to add List of new teachers. | | |
| Pre-conditions: | **PRE – 20.01.** Actor is logged to Course-Scheduling Website.  **PRE – 20.02.** Actor is set as the staff of the Website.  **PRE – 20.03.** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 20.01.** The teachers were added will be shown in list of all staffs of Staff page. | | |
| Normal Flow: | **UC – 20. Add List of Rooms**   1. In “*Teacher Management*” page, Actor clicks to button "Add List of Teachers”. 2. Course-Scheduling shows a dialog with an input to enter the link of excel file contains all information about rooms. 3. Actor selects “Add Teachers” option. 4. Actor enters the link of file. 5. Actor choose “Add”. 6. Course-Scheduling will validate all information, if it’s correct, the Rooms be added to database. 7. Course-Scheduling redirects to Teacher page with a message “*Import List of Teachers successful*!“. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B5, B6, B7** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-21. Add Courses For Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-21. Add Courses For Teachers | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add List of new teachers. | | |
| Description: | Actor uses this function to add List of new teachers. | | |
| Pre-conditions: | **PRE – 21.01.** Actor is logged to Course-Scheduling Website.  **PRE – 21.02.** Actor is set as the staff of the Website.  **PRE – 21.03.** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 21.01.** The teachers were added will be shown in list of all staffs of Staff page. | | |
| Normal Flow: | **UC – 21. Add Courses For Teachers**   1. In “*Teacher Management*” page, Actor clicks to button "Add List of Teachers”. 2. Course-Scheduling shows a dialog with an input to enter the link of excel file contains all information about rooms. 3. Actor selects “Add Courses For Teachers” option. 4. Actor enters the link of file. 5. Actor choose “Add”. 6. Course-Scheduling will validate all information, if it’s correct, the Rooms be added to database. 7. Course-Scheduling redirects to Teacher page with a message “*Import Courses For Teachers successful*!“. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **21.01 Course Not Found Exception.**  Course-Scheduling redirects to Room Management with message *“Course … not found. Please Insert It First.”* | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-22. Edit Teacher

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-22. Edit Teacher | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to update information of a teacher. | | |
| Description: | Actor uses this function to update information of a teacher. | | |
| Pre-conditions: | **PRE – 22.01.** Actor is logged to Course-Scheduling Website.  **PRE – 22.02.** Actor is set as the staff of the Website.  **PRE – 22.03.** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 22.01.** The information was edited will be saved to database. | | |
| Normal Flow: | **UC – 22. Edit Teacher**   1. Actor clicks to “Edit” of a staff in list of teachers. 2. Course-Scheduling shows a dialog with information of this teacher for actor to edit. (Edit courses the teacher will teach). 3. Actor edit information of this teacher. 4. Actor click “Save”. 5. Course-Scheduling closes dialog and saves information. 6. Course-Scheduling redirects to “*Teacher Management*” with message “*Edit Teacher’s Information Successful!*”. | | |
| Alternative Flows: | **AF – 22.01. At step 4, Actor chooses “cancel”.**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-23. Remove Teacher

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-23. Remove Teacher | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a teacher. | | |
| Description: | Actor uses this function to remove a teacher. | | |
| Pre-conditions: | **PRE – 23.01.** Actor is logged to Course-Scheduling Website.  **PRE – 23.02.** Actor is set as the staff of the Website.  **PRE – 23.03.** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 23.01.** The teacher was removed must not be shown in list of all teachers of Teacher page. | | |
| Normal Flow: | **UC – 23. Remove Room**   1. Actor clicks to “Delete” in the row of a teacher in list of teachers. 2. Course-Scheduling asks for confirmation to remove this teacher. 3. Actor chooses “Accept”. 4. Course-Scheduling removes this teacher in database. 5. Redirect to “*Teacher Management*” page with message: “*Remove Teacher successfully*”. | | |
| Alternative Flows: | **AF – 23.01. At step 3, Actor chooses “Cancel”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-24. Search Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-24. Search Teachers | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific teacher. | | |
| Description: | Actor uses this function to search for a specific teacher. | | |
| Pre-conditions: | **PRE – 25.01.** Actor is logged to Course-Scheduling Website.  **PRE – 25.02.** Actor is set as the staff of the Website.  **PRE – 25.03** Actor accessed to “*Teacher Management*” page. | | |
| Post-conditions: | **POST – 25.01.** The teachers have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 25. Search** **Room**   1. Actor enters information about the teacher that they want to find in the search box. 2. Course-Scheduling shows list of teachers that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-25. View All Students

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-25. View All Students | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the students. | | |
| Description: | Actor uses this function to see all the students. | | |
| Pre-conditions: | **PRE – 25.01.** Actor is logged to Course-Scheduling Website.  **PRE – 25.02.** Actor is set as the staff of the Website. | | |
| Post-conditions: | **POST – 25.01.** Course-Scheduling shows list of all students. | | |
| Normal Flow: | **UC – 25. View All Staffs**   1. Actor clicks to menu “*Students Management*”. 2. Course-Scheduling shows list of all Studentss. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-26. Add Student

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-26. Add Student | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new student. | | |
| Description: | Actor uses this function to add new student. | | |
| Pre-conditions: | **PRE – 26.01.** Actor is logged to Course-Scheduling Website.  **PRE – 26.02.** Actor is set as the staff of the Website.  **PRE – 26.03.** Actor accessed to “*Students Management*” page. | | |
| Post-conditions: | **POST – 26.01.** The student was added will be shown in list of all students of Student page. | | |
| Normal Flow: | **UC – 26. Add Student**   1. In “*Students Management*” page, Actor clicks to button "Add Student”. 2. The actor enters information about new student: “Name”, “Specialized”, “Batch”, “Semester”. 3. Actor click “Save” button. 4. Course-Scheduling will validate all information, if it’s correct, the Room will be added to database. 5. Course-Scheduling redirects to “*Students Management*” page with message: “*Add Student Successful*.”. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-27. Add List of Students

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-27. Add List of Students | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add List of new student. | | |
| Description: | Actor uses this function to add List of new student. | | |
| Pre-conditions: | **PRE – 27.01.** Actor is logged to Course-Scheduling Website.  **PRE – 27.02.** Actor is set as the staff of the Website.  **PRE – 27.03.** Actor accessed to “*Students Management*” page. | | |
| Post-conditions: | **POST – 27.01.** The students were added will be shown in list of all students of Student page. | | |
| Normal Flow: | **UC – 27. Add List of Students**   1. In “*Students Management*” page, Actor clicks to button "Add List of Students”. 2. Course-Scheduling shows a dialog with an input to enter the link of excel file contains all information about students. 3. Actor enters the link of file. 4. Actor choose “Add”. 5. Course-Scheduling will validate all information, if it’s correct, Students will be added to database. 6. Course-Scheduling redirects to “*Students Management*” page with a message “*Import List of Students successful!*“. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **27.01. Specialized not found Exception**  Course-Scheduling redirects to “*Students Management*” with message *“Specialized … not found. Please Insert It First.”* | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-28. Edit Student

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-28. Edit Student | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to update information of a student. | | |
| Description: | Actor uses this function to update information of a student. | | |
| Pre-conditions: | **PRE – 28.01.** Actor is logged to Course-Scheduling Website.  **PRE – 28.02.** Actor is set as the admin or staff of the Website.  **PRE – 28.03.** Actor accessed to “*Students Management*” page. | | |
| Post-conditions: | **POST – 28.01.** The information was edited will be saved to database. | | |
| Normal Flow: | **UC – 28. Edit Student**   1. Actor clicks to “Edit” of a staff in list of students. 2. Course-Scheduling shows a dialog with information of this student for actor to edit. 3. Actor edit information of this student. 4. Actor click “Save”. 5. Course-Scheduling closes dialog and saves information. | | |
| Alternative Flows: | **AF – 28.01. At step 4, Actor chooses “cancel”.**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-29. Remove Student

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-29. Remove Student | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a student. | | |
| Description: | Actor uses this function to remove a student. | | |
| Pre-conditions: | **PRE – 29.01.** Actor is logged to Course-Scheduling Website.  **PRE – 29.02.** Actor is set as the staff of the Website.  **PRE – 29.03.** Actor accessed to “*Students Management*” page. | | |
| Post-conditions: | **POST – 29.01.** The student was removed must not be shown in list of all students of Student page. | | |
| Normal Flow: | **UC – 29. Remove Student**   1. Actor clicks to “Delete” in the row of a room in list of students. 2. Course-Scheduling asks for confirmation to remove this students. 3. Actor chooses “Accept” 4. Course-Scheduling removes this student in database. 5. Redirect to “*Students Management*” page with message: “*Remove Student successful!”.* | | |
| Alternative Flows: | **AF – 29.01. At step 3, Actor chooses “Cancel”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-30. Search Students

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-30. Search Students | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific student. | | |
| Description: | Actor uses this function to search for a specific student. | | |
| Pre-conditions: | **PRE – 30.01.** Actor is logged to Course-Scheduling Website.  **PRE – 30.02.** Actor is set as the admin or staff of the Website.  **PRE – 30.03** Actor accessed to “*Students Management*” page. | | |
| Post-conditions: | **POST – 30.01.** The students have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 30. Search** **Student**   1. Actor enters information about the room that they want to find in the search box. 2. Course-Scheduling shows list of students that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-31. View All Courses

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-31. View All Courses | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the courses. | | |
| Description: | Actor uses this function to see all the courses. | | |
| Pre-conditions: | **PRE – 31.01.** Actor is logged to Course-Scheduling Website.  **PRE – 31.02.** Actor is set as the staff of the Website.  **PRE – 31.03.** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 31.01.** Course-Scheduling shows list of all courses. | | |
| Normal Flow: | **UC – 31. View All Staffs**   1. Actor clicks to menu “*Courses Management*”. 2. Actor choose Semester in left panel. 3. Course-Scheduling shows list of all Courses. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-32. Add Course

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-32. Add Course | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new course. | | |
| Description: | Actor uses this function to add new course. | | |
| Pre-conditions: | **PRE – 32.01.** Actor is logged to Course-Scheduling Website.  **PRE – 32.02.** Actor is set as the admin, staff of the Website.  **PRE – 32.03.** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 32.01.** The course was added will be shown in list of all courses of Course page. | | |
| Normal Flow: | **UC – 32. Add Course**   1. In “*Courses Management*” page, Actor clicks to button "Add Course”. 2. The actor enters information about new room: “Code”, “Name”, “Slots”, “Semester”, “Department” and “Course Condition” if needed. 3. Actor clicks “Save” button. 4. Course-Scheduling will validate all information, if it’s correct, the Course will be added to database. 5. Course-Scheduling redirects to Room page with message “Add Course Successful.”. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **32.01. Can not connect to server.**  Browser shows error message. | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: |  | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-33. Add List of Courses

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-33 Add List of Courses | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add List of new courses. | | |
| Description: | Actor uses this function to add List of new courses. | | |
| Pre-conditions: | **PRE – 33.01.** Actor is logged to Course-Scheduling Website.  **PRE – 33.02.** Actor is set as the staff of the Website.  **PRE – 33.03.** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 33.01.** The courses were added will be shown in list of all courses of Course page. | | |
| Normal Flow: | **UC – 33. Add List of Courses**   1. In Courses page, Actor clicks to button "Add List of Courses”. 2. Course-Scheduling shows a dialog with an input to enter the link of excel file contains all information about courses. 3. Actor enters the link of file. 4. Actor choose “Add”. 5. Course-Scheduling will validate all information, if it’s correct, the Courses be added to database. 6. Course-Scheduling redirects to “*Course Management*” page with a message “*Import List of Courses successful!*“. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **B10** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-34. Edit Course

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-34. Edit Course | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to update information of a course. | | |
| Description: | Actor uses this function to update information of a course. | | |
| Pre-conditions: | **PRE – 34.01.** Actor is logged to Course-Scheduling Website.  **PRE – 34.02.** Actor is set as the admin or staff of the Website.  **PRE – 34.03.** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 34.01.** The information was edited will be saved to database. | | |
| Normal Flow: | **UC – 34. Edit Course**   1. Actor clicks to “Edit” of a staff in list of courses. 2. Course-Scheduling shows a dialog with information of this courses for actor to edit. 3. Actor edit information of this course. 4. Actor click “Save”. 5. Course-Scheduling closes dialog and saves information. 6. Course-Scheduling redirects to “*Courses Management*” Page with message “*Edit Course’s information successful!*” | | |
| Alternative Flows: | **AF – 34.01. At step 4, Actor chooses “cancel”.**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-35. Remove Course

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-35. Remove Course | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a course. | | |
| Description: | Actor uses this function to remove a course. | | |
| Pre-conditions: | **PRE – 35.01.** Actor is logged to Course-Scheduling Website.  **PRE – 35.02.** Actor is set as the staff of the Website.  **PRE – 35.03.** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 35.01.** The room was removed must not be shown in list of all courses of Course page. | | |
| Normal Flow: | **UC – 35. Remove Room**   1. Actor clicks to “Delete” in the row of a room in list of rooms. 2. Course-Scheduling asks for confirmation to remove this course. 3. Actor chooses “Accept” 4. Course-Scheduling removes this course in database. 5. Redirect to “*Courses Management*” page with message: “*Remove Course successfully*”. | | |
| Alternative Flows: | **AF – 35.01. At step 3, Actor chooses “Cancel”**   1. Dialog is closed. 2. Nothing will be saved. | | |
| Exceptions: | **N/A** | | |
| Priority: | **High** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-36. Search Courses

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-36. Search Courses | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to search for a specific course. | | |
| Description: | Actor uses this function to search for a specific course. | | |
| Pre-conditions: | **PRE – 36.01.** Actor is logged to Course-Scheduling Website.  **PRE – 36.02.** Actor is set as the admin or staff of the Website.  **PRE – 36.03** Actor accessed to “*Courses Management*” page. | | |
| Post-conditions: | **POST – 36.01.** The courses have information similar with the information that actor entered will be shown. | | |
| Normal Flow: | **UC – 36. Search** **Courses**   1. Actor enters information about the course that they want to find in the search box. 2. Course-Scheduling shows list of courses that have this information. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-37. View All Classes

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-37. View All Classes | | |
| Created By: | **HaNS** | Date Created: | **2015-06-01** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to see all the classes. | | |
| Description: | Actor uses this function to see all the classes. | | |
| Pre-conditions: | **PRE – 37.01.** Actor is logged to Course-Scheduling Website.  **PRE – 37.02.** Actor is set as the staff of the Website.  **PRE – 37.03.** Actor accessed to “*Classes Management*” page. | | |
| Post-conditions: | **POST – 37.01.** Course-Scheduling shows list of all classes. | | |
| Normal Flow: | **UC – 37. View All Staffs**   1. Actor clicks to menu “*Classes Management*”. 2. Actor choose Semester in left panel. 3. Course-Scheduling shows list of all Classes in the semester selected. | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **N/A** | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-38. Add Class

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-38. Add Class | | |
| Created By: | **DucHM** | Date Created: | **2015-06-18** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to add new Class. | | |
| Description: | Actor uses this function to add new Class. | | |
| Pre-conditions: | **PRE – 38.01.** Actor is logged to Course-Scheduling Website.  **PRE – 38.02.** Actor is set as the staff of the Website.  **PRE – 38.03.** Actor accessed to “*Classes Management*” page. | | |
| Post-conditions: | **POST – 38.01.** The class was added will be shown in list of all Classes of Class page. | | |
| Normal Flow: | **UC – 38. Add Class**   1. In “*Classes Management*” page, Actor clicks to button "Add Class”. 2. The actor enters information about new Class: “Type”, “Batch”, “Specialized/Detail Specialized or Course”. 3. The actor clicks to button “Save”. 4. Course-Scheduling will validate all information and generate class name, if it’s correct, the Class will be added to database. 5. Course-Scheduling redirect to “*Classes Management*” page with message: “*Add Class Successful!*” | | |
| Alternative Flows: | **N/A** | | |
| Exceptions: | **38.01. Unexpected exception.**  **1.** Course-Scheduling shows error message.  **2.** Rollback transaction. | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-39. Add List of Classes

##### UC-40. Update Class

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-40. Update Class | | |
| Created By: | **DucHM** | Date Created: | **2015-06-18** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to update a Class. | | |
| Description: | Actor uses this function to update a Class. | | |
| Pre-conditions: | **PRE – 40.01.** Actor is logged to Course-Scheduling Website.  **PRE – 40.02.** Actor is set as the staff of the Website.  **PRE – 40.03.** Actor accessed to “*Classes Management*” page. | | |
| Post-conditions: | **POST – 40.01.** The class was updated will be shown in list of all Classes of Class page. | | |
| Normal Flow: | **UC – 40. Update Class**   1. The actor clicks to button “Edit” corresponding to a class 2. The actor edits information about this Class: “Type”, “Batch”, “Specialized/Detail Specialized or Course”. 3. The actor clicks to button “Save”. 4. Course-Scheduling will validate all information and generate class name, if it’s correct, the Class will be updated to database. 5. Course-Scheduling redirects to “Classes Management” page with | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **40.01. Unexpected exception.**  **1.** Course-Scheduling shows error message.  **2.** Rollback transaction. | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-41. Remove Class

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-41. Remove Class | | |
| Created By: | **DucHM** | Date Created: | **2015-06-18** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to remove a Class. | | |
| Description: | Actor uses this function to remove a Class. | | |
| Pre-conditions: | **PRE – 41.01.** Actor is logged to Course-Scheduling Website.  **PRE – 41.02.** Actor is set as the staff of the Website.  **PRE – 41.03.** Actor accessed to Manage Classes page. | | |
| Post-conditions: | **POST – 41.01.** The class was removed will not be shown in list of all Classes of Class page. | | |
| Normal Flow: | **UC – 41. Remove Class**   1. The actor clicks to button “Delete” corresponding to a class 2. The actor clicks to button “Yes” in confirm popup. 3. Course-Scheduling will validate all information, if it’s correct, the Class will be remove from database. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **41.01. Unexpected exception.**  **1.** Course-Scheduling shows error message.  **2.** Rollback transaction. | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **Low** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-42. List Classes

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-48. List Class | | |
| Created By: | **DucHM** | Date Created: | **2015-06-18** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to list all Classes. | | |
| Description: | Actor uses this function to list all Classes. | | |
| Pre-conditions: | **PRE – 48.01.** Actor is logged to Course-Scheduling Website.  **PRE – 48.02.** Actor is set as the staff of the Website.  **PRE – 48.03.** Actor accessed to Manage Classes page. | | |
| Post-conditions: | **POST – 48.01.** The classes will be shown in list of all Classes of Class page. | | |
| Normal Flow: | **UC – 48. List Class**   1. Course-Scheduling will get list of classes from database and display when actor accessed to Manage Classes page. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **48.01. Unexpected exception.**  **1.** Course-Scheduling shows error message.  **2.** Rollback transaction. | | |
| Priority: | **Medium** | | |
| Frequency of Use: | **High** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-44. Change Student’s Semester

##### UC-45. Update Student’s Learning Progress

##### UC-46. Auto put students into class(es)

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-46. Auto put students into class(es) | | |
| Created By: | **DucHM** | Date Created: | **2015-07-16** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to automatically put students into classes. | | |
| Description: | Actor uses this function to automatically put students into classes. | | |
| Pre-conditions: | **PRE – 46.01.** Actor is logged to Course-Scheduling Website.  **PRE – 46.02.** Actor is set as the staff of the Website. | | |
| Post-conditions: | **POST – 46.01.** Students were put into classes. | | |
| Normal Flow: | **UC – 46. Auto put students into class(es)**   1. Actor clicks to “Manage ClassSemesters” link. 2. Actor clicks to “Auto put students into class” or “Auto put all students into classes” 3. Course-Scheduling auto puts all suitable students into class(es). | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **46.01. Unexpected exception.**  **1.** Course-Scheduling shows error message.  **2.** Rollback transaction. | | |
| Priority: | **High** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-47. Manual put student into class

|  |  |  |  |
| --- | --- | --- | --- |
| UC ID and Name: | UC-47. Manual put student into class | | |
| Created By: | **DucHM** | Date Created: | **2015-07-16** |
| Primary Actor: | **Staff** | Secondary Actors: | N/A |
| Trigger: | The actor wants to put a student into a class. | | |
| Description: | Actor uses this function to put a student into a class. | | |
| Pre-conditions: | **PRE – 47.01.** Actor is logged to Course-Scheduling Website.  **PRE – 47.02.** Actor is set as the staff of the Website. | | |
| Post-conditions: | **POST – 47.01.** Student was put into class. | | |
| Normal Flow: | **UC – 47. Manual put student into class**   1. Actor clicks to “Manage ClassCourseStudentSemesters” link. 2. Actor Select semester, class & course(s) 3. Actor select a student from suitable students list to put into class. | | |
| Alternative Flows: | N/A | | |
| Exceptions: | **47.01. Max Student Number Exception.**  **1.** Course-Scheduling shows error message: “Student number of this class exceed 30”.  **2.** Rollback transaction. | | |
| Priority: | **High** | | |
| Frequency of Use: | **Medium** | | |
| Business Rules: | **N/A** | | |
| Other Information: | **N/A** | | |
| Assumptions: | **N/A** | | |

##### UC-48. Create Timetable

##### UC-49. Choose Teachers for Classes

##### UC-50. View All Timetable

##### UC-51. View Personal Timetable

##### UC-52. View Timetable of A Class

## Non-functional Requirements

### Security

### Maintainability & Extensibility

### Performance

## Entity Relationship Diagram

# CHAPTER 4: SOFTWARE DESIGN

## Purpose

## Architecture Overview

### System Architecture

### System Architecture Explanation

## Design of CourseScheduling system

### Architecture Layers Design

### Database Design

#### Database Diagram

### Common Design

### Detail Design