**University Catalog Management System Version 2.0**

Mentor:

Tim Downey

Team Members:

Jose Astudillo

Christopher Sutton

Senior Project

CIS 4911 – U01

Masoud Sadjadi

September 4th, 2014

Copyright © by Florida International University – Senior Project – UCMS Team

All rights reserved. No part of the University Catalog Management System Project or its documentation may be reproduced or transmitted in any form or by any means without prior written consent of the UCMS Team.

ABSTRACT

The Catalog Management System (CMS) project is designed to provide a way to manage electronically the content of the university catalogs. The system is sophisticated and highly user friendly in this aspect. On the second version of CMS, functionalities as create, edit, propose, accept, and reject prospective catalogs, which content may be part of the university catalog once they are accepted by an Administrator. In addition, flowchart generation has been added in order to provide a better visualization of a degree path in a catalog.

This document contains information about the design of the entire project including the content added in this version (v 2.0) of the project. The main chapters on this document are the Introduction, Current System, Project Plan, and Proposed System Requirements. The introduction presents the problem definition, scope of the system, and the terminology to be used in this document. The Project Plan shows the how the project is organized, how the work is broken down, as well as the cost estimate for the project. On the Proposed System Requirement chapter, the expected functionalities to be implemented on the second version of the Catalog Management System are proposed. This chapter includes different scenarios for the functionalities, the use case model, static model, and dynamic model for the system. In addition, this document also contains glossary, appendix, and references, if any.

Contents

[1. Introduction 4](#_Toc405117922)

[1.1. Problem Definition 4](#_Toc405117923)

[1.2. Scope of the System 4](#_Toc405117924)

[1.3. Terminology 4](#_Toc405117925)

[1.3.1. Definitions 4](#_Toc405117926)

[1.3.2. Acronyms 5](#_Toc405117927)

[1.3.3. Abbreviations 5](#_Toc405117928)

[1.4. Overview of the System 5](#_Toc405117929)

[2. Current System (Limitation and Problems) 6](#_Toc405117930)

[3. Project Plan 7](#_Toc405117931)

[3.1. Project Organization 7](#_Toc405117932)

[3.2. Work Breakdown 7](#_Toc405117933)

[3.3. Cost Estimate 8](#_Toc405117934)

[4. Proposed System Requirements 9](#_Toc405117935)

[4.1. Functional Requirements 9](#_Toc405117936)

[4.2. Analysis of the System Requirements 13](#_Toc405117937)

[4.2.1. Scenarios 13](#_Toc405117938)

[4.2.2. Use Case Model 14](#_Toc405117939)

[4.2.3. Static Model 14](#_Toc405117940)

[4.2.4. Dynamic Model 14](#_Toc405117941)

[5. Glossary 15](#_Toc405117942)

[5.1. Definitions 15](#_Toc405117943)

[5.2. Acronyms 15](#_Toc405117944)

[5.3. Abbreviations 15](#_Toc405117945)

[6. Appendix 16](#_Toc405117946)

[6.1. Appendix A – Complete Use Cases 16](#_Toc405117947)

[6.2. Appendix B – Use Case Diagram Using UML 31](#_Toc405117948)

[6.3. Appendix C – Static UML Diagram 33](#_Toc405117949)

[6.4. Appendix D – Dynamic UML Diagram 35](#_Toc405117950)

[6.5. Appendix E – User Interface Design 50](#_Toc405117951)

[6.6. Appendix F – Diary of Meeting and Tasks 56](#_Toc405117952)

[7. References 64](#_Toc405117953)

## Introduction

This chapter introduces the University Catalog Management System version 2.0 including what its purpose is, and what can be solved using it. In addition, this chapter defines the scope for the system, and the different terms (acronyms and abbreviations) that will be used throughout the different document that will be done for the project. Finally, a brief overview for the system is provided.

## Problem Definition

Currently, having hard copies of university catalog makes it difficult for advisors to find out differences between programs of study, tracks of program, changes made in the requirements for a degree, etc. For this reason, the University Catalog Management System tries to simplify the work for advisors, so that they can help out students faster.

On the other hand, there is no way to control the input of new data into the system. This data refers to new majors, minors, certificates, groups, set, and courses in the catalog system. Currently, the only one with the capability to input the data is the administrator, which is biased since the administrator can enter new information, but this information is not reviewed by anyone else. For this reason, it is necessary to have a process which can control the input of data.

Finally, all departments within the university are required to have some sort of flowchart in order to facilitate the understanding of a program. But these charts are sometimes created manually in Microsoft Visio, which is time consuming. This is the main reason that an automated creation of these flowchart is required.

## Scope of the System

Currently, this system is expected to work for the department of Computer and Information Sciences from FIU. However, there exists a high probability that it can be used throughout the entire university by the different departments. In addition, this system is concerned to work with all programs offered by the CIS department.

## Terminology

## Definitions

* Catalog : Complete list of items ordered systematically.
* User : Any person that will have access to the system, either to just view information, or to edit data in the system.
* Administrator : User that has control over all other users within their DGU, such as advisors and students.
* Super Admin : User that will manage the entire system including administrator.

## Acronyms

* UCMS : University Catalog Management System
* V2.0 : Version 2.0
* SQL : Structured Query Language
* PHP : Hypertext Pre-Processor
* JS : JavaScript
* GUI : Graphical User Interface
* HTML : Hyper Text Markup Language
* CSS : Cascading Style Sheet
* Yii : Yes It Is! Framework.
* CIS : Computer and Information Sciences
* FIU : Florida International University.
* MVC : Model View Controller
* DGU : Degree Granting Unit

## Abbreviations

* Db : Database
* Admin : Administration

## Overview of the System

In the following chapters, the system will be explained in a more detailed manner. Chapter 2 provides information about the limitations and problems of the current system. Furthermore, in chapter 3, the roles for the team members, a cost estimate for the development of the system, and the schedule for the different tasks will be shown so that the project plan for this project can be better understood. Chapter 4 will give us a better understanding of what the system will do. The different functionalities that will be added in the system will be explained in terms of use cases. Moreover, in this chapter, uses cases will be analyzed using scenarios, use case model, and diagrams. Chapter 5 provides a glossary of terms used throughout the document. Finally, Chapter 6 is the Appendix for the document, which contains complete use cases, diagrams modeled in UML, and the diary of meetings.

## Current System (Limitation and Problems)

Currently, advisors and students (or prospective students) are having a hard time when looking for catalogs at FIU. For advisors, there is no way other than to look through all the different books that contain information about the different programs of study to find out an older version for the plans of these programs. Furthermore, FIU’s website, which contains information about the programs of study, is hard to follow. In addition, flowcharts for the different degree have to be done manually, and posted in every department’s website. For this reason, the University Catalog Management System is expected to ease all these tedious searches.

The University Catalog Management System v1.0 has already been implemented. However, it still lacks some key functionalities. This version of the program only allows users to look for catalogs of the programs offered by the CIS Department, as well as the different degrees offered by this department. Moreover, advisors and administrators do not have their key functionalities that they are supposed to have in the system, which basically makes them regular users.

Moreover, UCMS v1.0 lacks a way to communicate ideas and propose changes to the catalogs. Currently, the system only allows entering information that is currently active such as majors, minors, courses, sets, and groups. The proposal of prospective changes in a catalog must be a task that advisors and administrators within a degree granting unit must complete. Following this process, a administrator must be able to either accept or reject the changes that have been proposed by advisors or admins.

The UCMS v2.0 is expected to resolve all these issues by implementing the most crucial functionalities for this program. In addition, this version is projected to be the most complete aiding on managing the catalogs within university.

## Project Plan

This chapter includes information about that how the team working in the project is organized. It also contains the work breakdown of task and activities to be completed during the development of UCMS v2.0. Furthermore, a cost estimate for the projects is provided.

## Project Organization

The team working on this project consists of two members:

* Jose Astudillo : Manager, Developer, Tester, System Designer, GUI Designer
* Christopher Sutton : Manager, Developer, Tester, Database Manager

Manager, Developer, Tester, System Designer, GUI Designer

Jose Astudillo

Manager, Developer, Tester, Database Manager,

Christopher Sutton

**Diagram 3.1.1** Team Project Organization

Diagram 3.1.1 helps us understand better how the team is organized. Each team member will be manager of the other. This means that every work done by a single team member will be revised by its manager. In addition, both members will work on the code, and testing. Each member also has different roles; for example, Jose Astudillo is the System Designer, and Christopher is the Database Manager. However, work will be revised by each other.

## Work Breakdown

The table below shows the different milestones required for this project.

|  |  |  |
| --- | --- | --- |
| Milestone | Description | Due Date |
| Milestone 1 | Feasibility Analysis |  |
| Milestone 2 | Requirement Document |  |
| Milestone 3 | Design Document |  |
| Milestone 4 | Final Document |  |
| Milestone 5 | Poster |  |
| Milestone 6 | Project Presentation |  |
| Milestone 7 | Project Showcase |  |

**Table 3.2.1** Work Breakdown

## Cost Estimate

The following table shows the cost estimate for developing project.

|  |  |  |
| --- | --- | --- |
| Item | Description | Cost |
| Human Resources | Team member working on the project during the entire development process. | $0.00 |
| Hardware Tools | Laptops | $0.00 |
| FIU Computers | $0.00 |
| Software Tools | Yii Framework | $0.00 |
| Netbeans | $0.00 |
| PHPAdmin | $0.00 |
| Total cost | | $0.00 |

**Table 3.3.1** Cost Estimate

## Proposed System Requirements

This chapter defines the functional requirement to be implemented in version 2.0 of the University Catalog Management System project. These requirements are described in terms of functional requirements and their respective nonfunctional requirements.

## Functional Requirements

Below are the functional requirements for the v2.0 of the Catalog Management System.

The system shall allow…

**Allow advisors to create prospective catalogs**

* **Usability**: The form to be filled must be easy to follow when filling up.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Data must be save within 3s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisors to propose prospective catalogs**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Data must be save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to accept prospective catalogs**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Data must be save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to reject prospective catalogs**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Data must be save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to view prospective catalogs**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Catalog must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin and students to view track flowchart**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin and students to view group flowchart**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin and students to view set flowchart**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to alter flowchart track layout**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to alter flowchart group layout**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow admin to alter flowchart set layout**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Flowchart must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective course**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective course form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective set**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective set form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective group**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective group form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective minor**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective minor form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective track**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective track form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective certificate**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective certificate form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to create prospective major**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: new prospective major form must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective course**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective course form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective set**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective set form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective group**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective group form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective minor**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective minor form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective track**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective track form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective certificate**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective certificate form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

**Allow advisor to edit prospective major**

* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: editable prospective major form loaded with current content must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

## Analysis of the System Requirements

This section contains scenarios for some system requirements as well as the different models related to the requirements such as use case model, static model, and dynamic model.

## Scenarios

This section contains the main scenario for the second version of the Catalog Management System.

**Proposing of a Prospective Catalog**

The advisor user has the capability to propose changes in an active catalog as well as to add new information such as majors, minors, certificates, tracks, groups, sets, and courses. To make this possible, it is crucial that the user creates a prospective catalog he/she will work on (UCMSv2 - 0001). After the creation of the catalog the user can access his/her catalog if and only if the catalog has not been proposed yet. The user can make use of the addition of new content (UCMSv2 – 0012 through UCSMv2 – 0018) and edit current information (UCMSv2 – 0019 through UCMSv2 – 0025) which will be information in the prospective catalog the user is creating. Once the user feels that necessary prospective changes to the catalogs have been made he/she can propose it (UCMSv2 – 0002) so that administrators can further review it.

**Accepting and Rejecting Prospective Catalogs**

Once a prospective catalog has been proposed, the administrator has the capability to view the changes (UCMSv2 – 0005). The main purpose of viewing changes is that the administrator can either accept (UCMSv2 – 0004) or reject (UCMSv2 – 0003) the prospective catalog. If catalog has been accepted, the catalogs will become active; otherwise the advisor that proposed the catalog will have to keep working on the mentioned catalog.

**Flowchart Generation and View**

The visualization of a program is an important feature for the CMS. Administrators and advisers have the capability to make changes to a track, group, or set chart. Changed can be made by dynamically making changes by dragging course elements (UCMSv2 – 0009 through UCMSv2 – 0011). All changes made are reloadable and used in related chart visualizations. Once all changes to the flowcharts have been made, all users have the capability to view the program flowchart (UCMSv2 – 0006 through UCMSv2 – UCMSv2 – 0008). If a flowchart does not exist in a database a default flowchart will be displayed with courses ordered by the number of pre-requisites they each have. Groups and sets will be ordered by their corresponding identification numbers in the system.

## Use Case Model

Appendix B contains the Use Case diagrams with all use cases pertaining to the system. The use cases that have been included in the diagram are the ones that were implemented in the version 1.0 on the system as well as the use cases related to the functionalities that are implemented on the v2.0.

This diagram clearly shows the different actors that will be interacting with the system. These are super-admin, admin, advisor, student, and the guest of the system. Each of these actors is linked with his/her respective capabilities within the system.

## Static Model

Appendix C shows some of the different class diagrams for the Catalog Management System. These diagrams show the relationship between classes. The class diagrams provided in Appendix C are minimal due to space. On top of each class diagram, the name of diagram can be seen.

## Dynamic Model

Appendix D shows the different sequence diagrams for the Catalog Management System. These diagrams show the interaction among the objects as they are planned to occur when the system is in execution. On top of each sequence diagrams, its use case id is displayed as well as its name.

## Glossary

## Definitions

* Catalog : Complete list of items ordered systematically.
* User : Any person that will have access to the system, either to just view information, or to edit data in the system.
* Administrator : User that has control over all other users in the system, such as advisors and students.
* Super Admin : User that will manage the entire system including administrator.

## Acronyms

* UCMS : University Catalog Management System
* V2.0 : Version 2.0
* SQL : Structured Query Language
* PHP : Hypertext Pre-Processor
* JS : JavaScript
* GUI : Graphical User Interface
* HTML : Hyper Text Markup Language
* CSS : Cascading Style Sheet
* Yii : Yes It Is! Framework.
* CIS : Computer and Information Sciences
* FIU : Florida International University.
* MVC : Model View Controller

## Abbreviations

* Db : Database
* Admin : Administration

## Appendix

## Appendix A – Complete Use Cases

**Use Case ID :** UCMSv2 – 0001

**Name :** Create DGU Proposed Catalog

**Details :** This use case allows Admin and advisor to create prospective new catalogs for their corresponding DGU. This includes prospective courses, major, minor, certificates, tracks, groups and set.

**Actors :** Admin, and advisor.

**Pre-condition(s):**

1. Advisor must be logged in into the system

**Description :**

1. Use case begins when Advisor clicks on the “Prospective Catalog” button.
2. Use select option “Create Prospective Catalog” from the prospective catalog page.
3. User must select his/her DGU.
4. User must enter catalog name.
5. User must enter description for the catalog.
6. User must enter term in which catalog will be activated.
7. User must enter the year catalog will be activated.
8. Use Case ends when user clicks in “Create New Catalog” button.

**Post condition(s):**

1. Prospective catalog must be saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0002

**Name :** Propose DGU Catalog.

**Details :** This use case allows Advisors to post a prospective catalog for their corresponding DGU.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been already created.

**Description :**

1. Use case begins when Admin or Advisor clicks on the “Prospective Catalog” button.
2. Use select option “View Prospective Catalog” from the prospective catalog page.
3. System must display the prospective catalog.
4. Use case ends when user clicks on “Propose” button.

**Post condition(s):**

1. DGU Admin must be able to see the proposed prospective catalog.

=====================================================================

**Use Case ID :** UCMSv2 – 0003

**Name :** Accept Prospective Catalog.

**Details :** This use case allows Admin to accept prospective catalogs that have been proposed.

**Actors :** Admin

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been proposed already.

**Description :**

1. Use case begins when Admin or Advisor clicks on the “Prospective Catalog” button.
2. Use select option “View Prospective Catalog” from the prospective catalog page.
3. System must display the list of proposed prospective catalogs.
4. Admin must select the prospective catalog that he/she wants to review.
5. System must display the prospective catalog
6. Admin must click the button “Accept.”

**Post condition(s):**

1. Status of the prospective catalog must change in the database.
2. The prospective catalog that was rejected must appear in the list with a status of accepted.

=====================================================================

**Use Case ID :** UCMSv2 – 0004

**Name :** Reject Prospective Catalog

**Details :** This use case allows Admin to reject prospective catalogs that have been proposed.

**Actors :** Admin

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been proposed already.

**Description :**

1. Use case begins when Admin or Advisor clicks on the “Prospective Catalog” button.
2. Use select option “View Prospective Catalog” from the prospective catalog page.
3. System must display the list of proposed prospective catalogs.
4. Admin must select the prospective catalog that he/she wants to review.
5. System must display the prospective catalog
6. Admin must click the button “Reject.”

**Post condition(s):**

1. Status of the prospective catalog must change in the database.
2. The prospective catalog that was rejected must appear in the list with a status of rejected.

=====================================================================

**Use Case ID :** UCMSv2 – 0005

**Name :** View Prospective Catalog.

**Details :** This use case allows Admin to see the prospective catalogs that have been proposed by either advisors or admin itself

**Actors :** Admin

**Pre-condition(s):**

1. Admin must be logged in into the system.

**Description :**

1. Use case begins when Admin or Advisor clicks on the “Prospective Catalog” button.
2. Admin selects option “View Prospective Catalogs” from the prospective catalog page.
3. System must display the list of proposed prospective catalogs.
4. Admin must select the prospective catalog that he/she wants to review.
5. Use case end when system displays the prospective catalog that Admin has selected.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0006

**Name :** View Track Flowchart.

**Details :** This use case allows users to view the track flowchart for a particular degree.

**Actors :** Users, Admins, Super Admins

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. In order to view the flow chart a user would have to click the “View Flowchart” button.
3. The flowchart would then be loaded divided by:
   1. Major track
      1. Groups
      2. Sets
      3. Courses
4. The flowchart components will be positioned in the same view that it was last saved in.
5. Clicking on the group will display the flowchart for that particular group. See UCMSv2- 0007.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0007

**Name :** View Group Flowchart.

**Details :** This use case allows users to view the group flowchart for a particular degree.

**Actors :** Users, Admins, Super Admins

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. In order to view the flow chart a user would have to click the “View Flowchart” button.
3. In order to navigate to the group level click on any group inside the track flowchart.
4. The flowchart would then be loaded divided by:
   1. Major group:
      1. Sets
      2. Courses
5. The flowchart components will be positioned in the same view that it was last saved in.
6. Clicking on the set will display the flowchart for that particular set. See UCMSv2-0008.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0008

**Name :** View Set Flowchart.

**Details :** This use case allows users to view the set flowchart for a particular degree.

**Actors :** Users, Admins, Super Admins

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. In order to view the flow chart a user would have to click the “View Flowchart” button.
3. In order to navigate to the group level click on any group inside the track flowchart, and then click on any set.
4. The flowchart would then be loaded divided by:
   1. Major set:
      1. Courses
5. The flowchart components will be positioned in the same view that it was last saved in.
6. Clicking on the course will display detailed course information for that particular course.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0009

**Name :** Alter Flowchart Track Layout.

**Details :** This use case allows users to alter layout of the flowchart for a particular track.

**Actors :** Admin

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. By navigating to the track layout for the flowchart a user will see a visualization organized by groups.
3. Users can move these groups to different positions if they wish to alter their order.
4. Once the user has finished making the position changes, clicking submit will write these changes to the database.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0010

**Name :** Alter Flowchart Group Layout.

**Details :** This use case allows users to alter layout of the flowchart for a particular group.

**Actors :** Admin

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. By navigating to the group layout for the flowchart a user will see a visualization organized by sets.
3. Users can move these sets to different positions if they wish to alter their order.
4. Once the user has finished making the position changes, clicking submit will write these changes to the database.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0011

**Name :** Alter Flowchart Set Layout.

**Details :** This use case allows users to alter layout of the flowchart for a particular set.

**Actors :** Admin

**Pre-condition(s):**

1. User must select some degree.

**Description :**

1. Once the users have selected a degree track they will then be able to see the courses associated with that major.
2. By navigating to the set layout for the flowchart a user will see a visualization organized by courses.
3. Users can move these courses to different positions if they wish to alter their order.
4. Once the user has finished making the position changes, clicking submit will write these changes to the database.

**Post condition(s):** None

=====================================================================

**Use Case ID :** UCMSv2 – 0012

**Name :** Create Prospective Course

**Details :** This use case allows Advisor to create prospective course for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add courses section.
3. User must enter name for the new course.
4. User must click in the “Add” button.
5. System must display the “create prospective course” form.
6. User must enter the prefix, code, number of credits, description, and notes for the course being added.
7. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective course must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0013

**Name :** Create Prospective Set

**Details :** This use case allows Advisor to create prospective set for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add set section.
3. User must enter name for the new set.
4. User must click in the “Add” button.
5. System must display the “create prospective set” form.
6. User must enter the description, minimum number of credits, and the maximum number of credits for the set being added.
7. User must select the course to be added in the new set.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective set must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0014

**Name :** Create Prospective Group

**Details :** This use case allows Advisor to create prospective group for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add group section.
3. User must enter name for the new group.
4. User must click in the “Add” button.
5. System must display the “create prospective group” form.
6. User must enter the description, minimum number of credits, and the maximum number of credits for the group being added.
7. User must select the sets to be added in the new group.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective group must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0015

**Name :** Create Prospective Minor

**Details :** This use case allows Advisor to create prospective minor for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add minor section.
3. User must enter name for the new minor.
4. User must click in the “Add” button.
5. System must display the “create prospective minor” form.
6. User must enter the description, and the minimum number of credits for the minor being added.
7. User must select the groups to be added in the new minor.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective minor must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0016

**Name :** Create Prospective Track

**Details :** This use case allows Advisor to create prospective track for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add track section.
3. User must enter name for the new track.
4. User must click in the “Add” button.
5. System must display the “create prospective track” form.
6. User must enter the description, and the minimum number of credits for the track being added.
7. User must select the groups to be added in the new track.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective track must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0017

**Name :** Create Prospective Certificate

**Details :** This use case allows Advisor to create prospective certificate for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add certificate section.
3. User must enter name for the new certificate.
4. User must click in the “Add” button.
5. System must display the “create prospective certificate” form.
6. User must enter the description, and the minimum number of credits for the certificate being added.
7. User must select the groups to be added in the new certificate.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective certificate must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0018

**Name :** Create Prospective Major

**Details :** This use case allows Advisor to create prospective major for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+” button in the Add major section.
3. User must enter name for the new major.
4. User must click in the “Add” button.
5. System must display the “create prospective major” form.
6. User must enter the description for the major set being added.
7. User must select the tracks to be added in the new major.
8. Use case ends when user click on “Save” button.

**Post condition(s):**

1. A new prospective major must have been created.

=====================================================================

**Use Case ID :** UCMSv2 – 0019

**Name :** Edit Prospective Course

**Details :** This use case allows Advisor to make prospective change to a course for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A course must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add courses section.
3. User must select course to be edited.
4. System must display the “prospective course” form with all field filled out with the current information.
5. User must enter changes to the prefix, code, number of credits, description, and notes for the course being edited.
6. Use case ends when user click on “Save” button.

**Post condition(s):**

1. Changes to the course must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0020

**Name :** Edit Prospective Set

**Details :** This use case allows Advisor to make prospective change to a set for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A set must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add set section.
3. User must select set to be edited.
4. System must display the “prospective set” form with all field filled out with the current information.
5. User must enter changes to the minimum number of credits, description, and the maximum number of credits for the set being edited.
6. User must click on “Save” button.
7. User has the option to either add course or remove course to the set he/she is editing by clicking either on the “Add Course” or “Remove Course” button.

**Post condition(s):**

1. Changes to the set must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0021

**Name :** Edit Prospective Group

**Details :** This use case allows Advisor to make prospective change to a group for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A group must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add group section.
3. User must select group to be edited.
4. System must display the “prospective group” form with all field filled out with the current information.
5. User must enter changes to the minimum number of credits, description, and the maximum number of credits for the group being edited.
6. User must click on “Save” button.
7. User has the option to either add set or remove set to the group he/she is editing by clicking either on the “Add Set” or “Remove Set” button.

**Post condition(s):**

1. Changes to the group must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0022

**Name :** Edit Prospective Minor

**Details :** This use case allows Advisor to make prospective change to a minor for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A minor must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add minor section.
3. User must select minor to be edited.
4. System must display the “prospective minor” form with all field filled out with the current information.
5. User must enter changes to the minimum number of credits, and description for the minor being edited.
6. User must click on “Save” button.
7. User has the option to either add group or remove group to the minor he/she is editing by clicking either on the “Add Group” or “Remove Group” button.

**Post condition(s):**

1. Changes to the minor must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0023

**Name :** Edit Prospective Track

**Details :** This use case allows Advisor to make prospective change to a track for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A track must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add track section.
3. User must select track to be edited.
4. System must display the “prospective track” form with all field filled out with the current information.
5. User must enter changes to the minimum number of credits, and description for the track being edited.
6. User must click on “Save” button.
7. User has the option to either add group or remove group to the track he/she is editing by clicking either on the “Add Group” or “Remove Group” button.

**Post condition(s):**

1. Changes to the track must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0024

**Name :** Edit Prospective Certificate

**Details :** This use case allows Advisor to make prospective change to a certificate for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A certificate must exist.

**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add certificate section.
3. User must select certificate to be edited.
4. System must display the “prospective certificate” form with all field filled out with the current information.
5. User must enter changes to the minimum number of credits, and description for the certificate being edited.
6. User must click on “Save” button.
7. User has the option to either add group or remove group to the certificate he/she is editing by clicking either on the “Add Group” or “Remove Group” button.

**Post condition(s):**

1. Changes to the certificate must been saved in the database.

=====================================================================

**Use Case ID :** UCMSv2 – 0025

**Name :** Edit Prospective Major

**Details :** This use case allows Advisor to make prospective change to a major for the prospective catalog he/she is working on.

**Actors :** Advisor

**Pre-condition(s):**

1. Advisor must be logged in into the system.
2. A DGU Prospective Catalog must have been created already.
3. A major must exist.

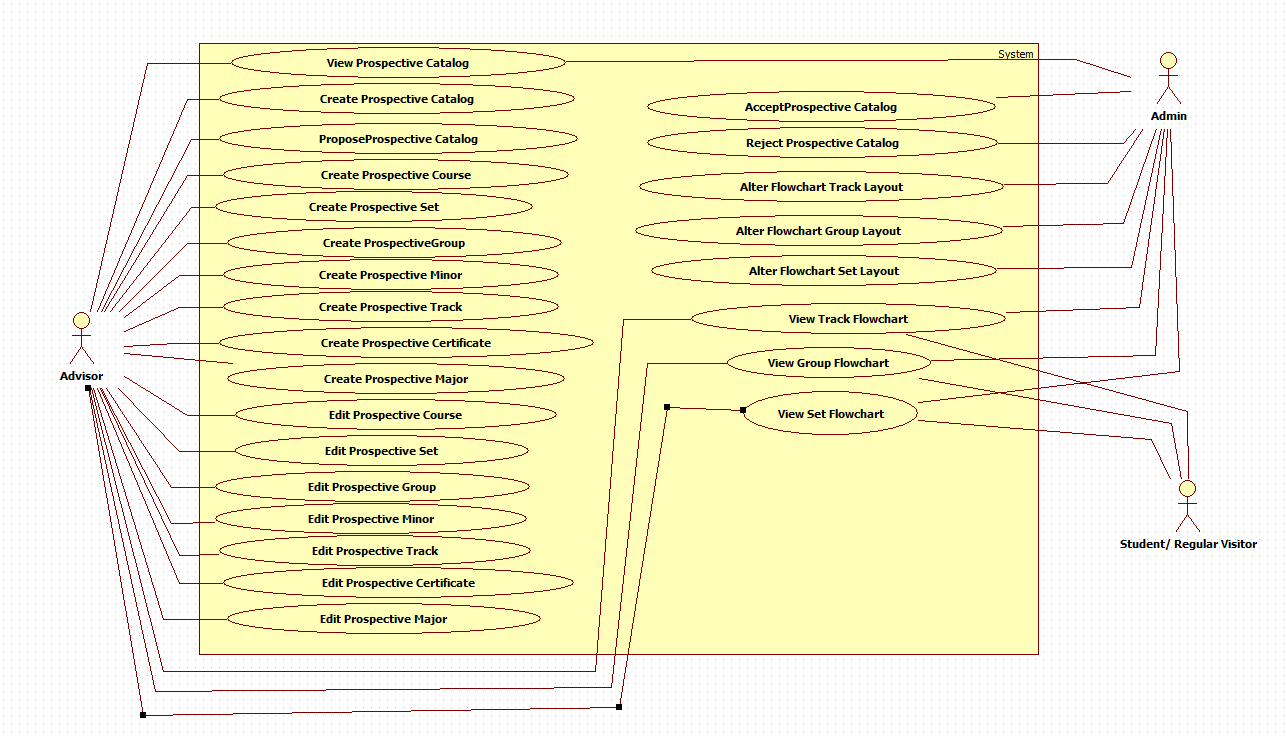
**Description :**

1. Use case begins when Advisor click on “Create Prospective Catalog.”
2. User clicks the “+edit” button in the Add major section.
3. User must select major to be edited.
4. System must display the “prospective major” form with all field filled out with the current information.
5. User must enter changes to the description for the certificate being edited.
6. User must click on “Save” button.
7. User has the option to either add track or remove track to the major he/she is editing by clicking either on the “Add Track” or “Remove Track” button.

**Post condition(s):**

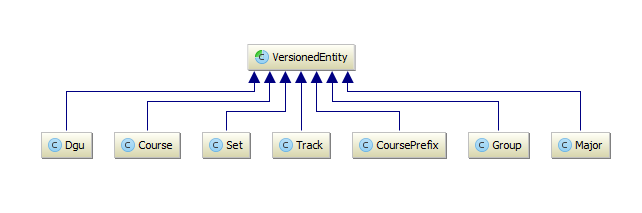
1. Changes to the major must been saved in the database.

## Appendix B – Use Case Diagram Using UML

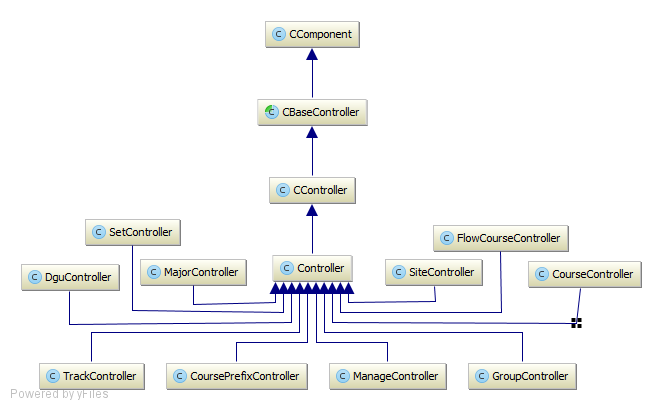


## Appendix C – Static UML Diagram

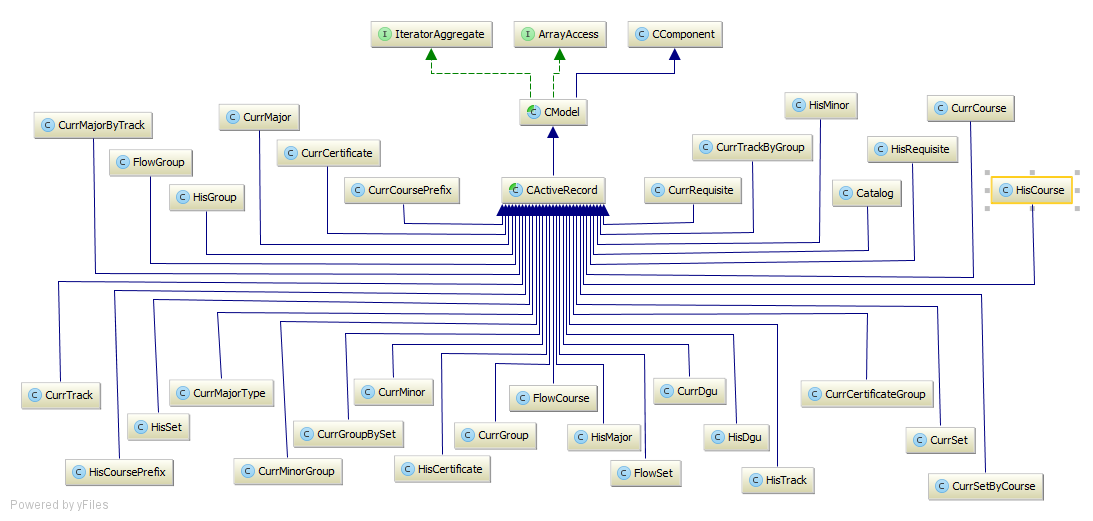
Version Entity Minimal Class Diagram



CMS\_Model Minimal Class Diagram

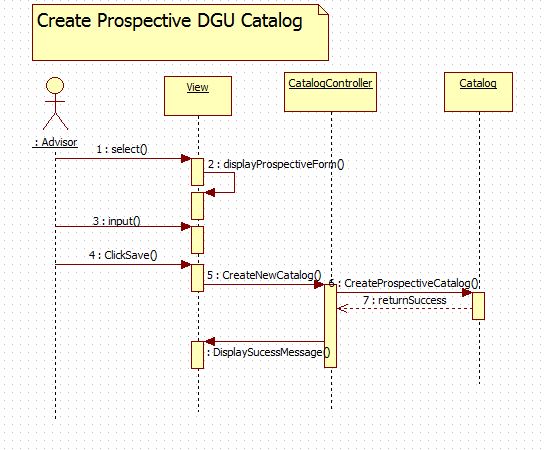


CMS\_Controller Minimal Class Diagram

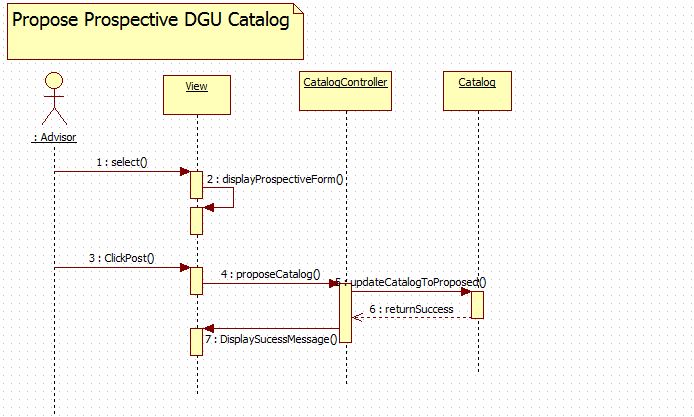


## Appendix D – Dynamic UML Diagram

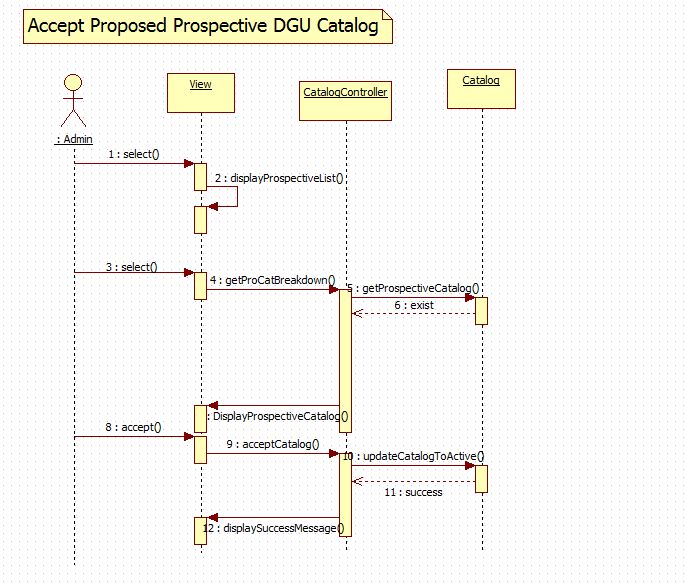
UCMSv2-0001: Create Prospective DGU Catalog



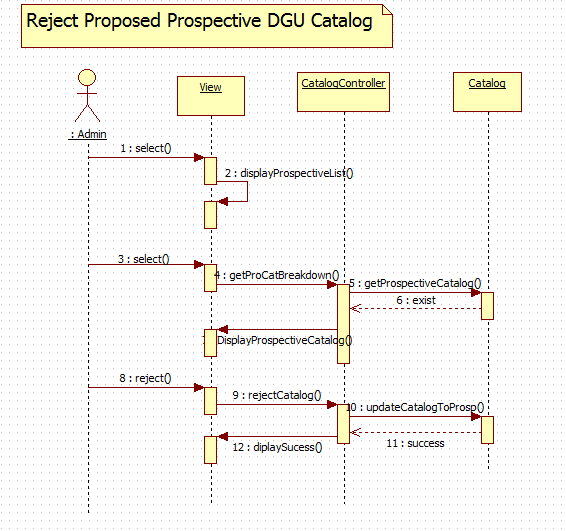
UCMSv2-0002: Propose Prospective DGU Catalog



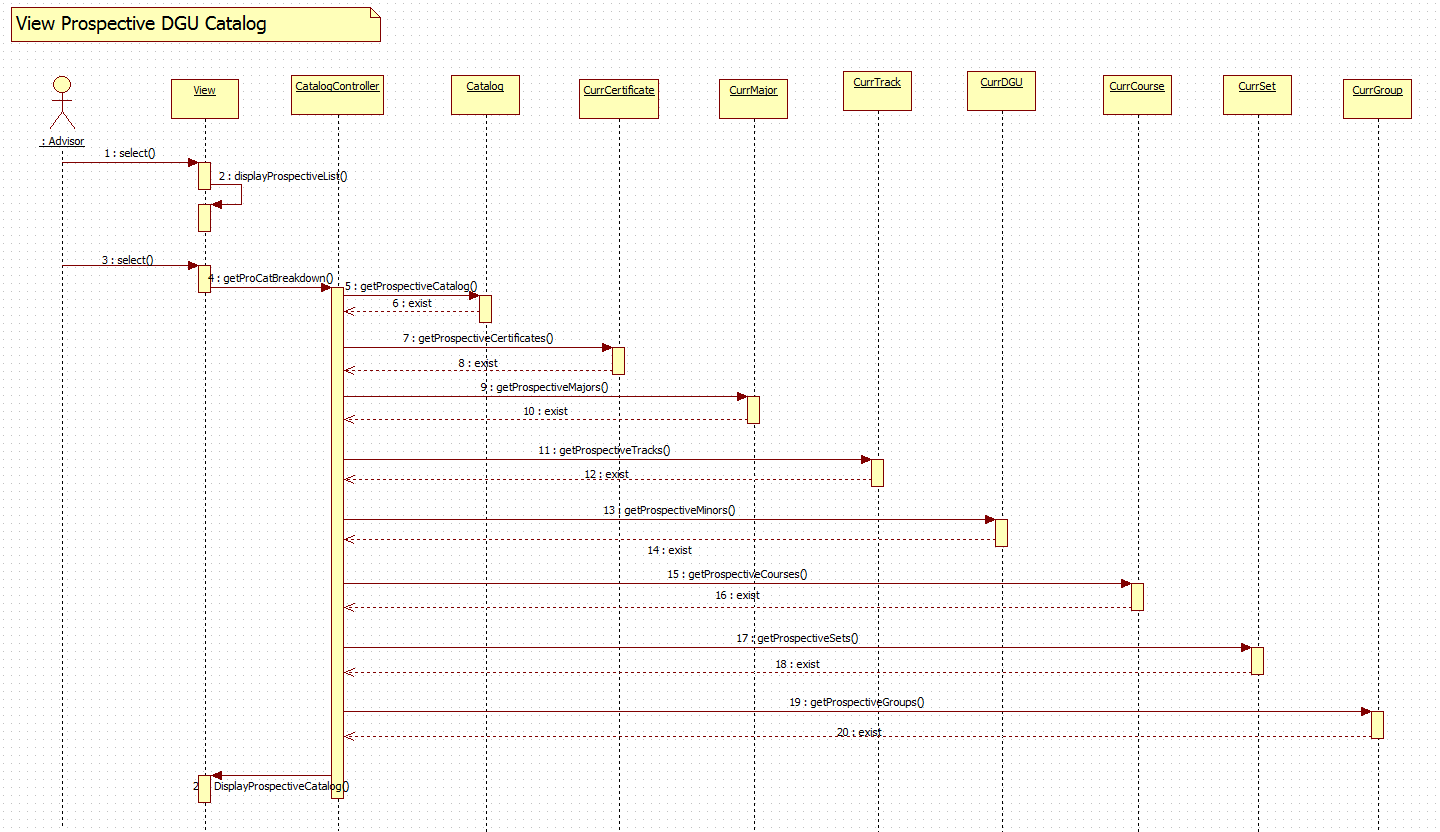
UCMSv2-0003 : Accept Prospective Catalog



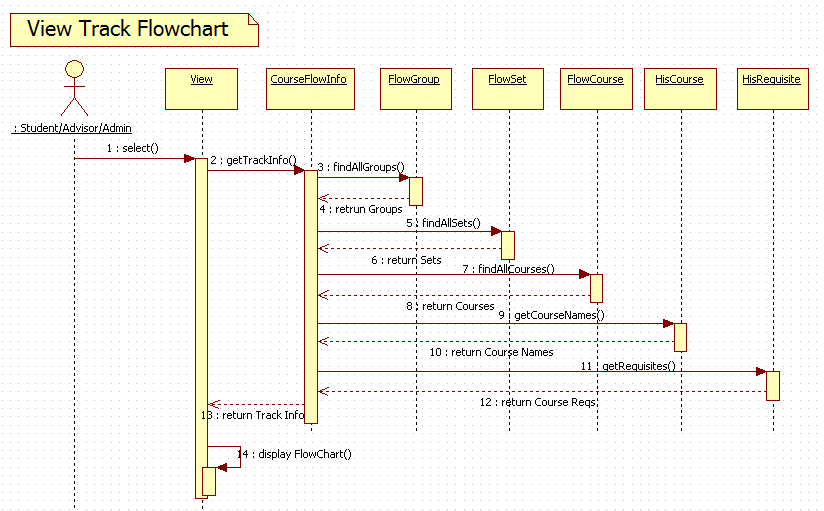
UCMSv2-0004 : Reject Prospective Catalog



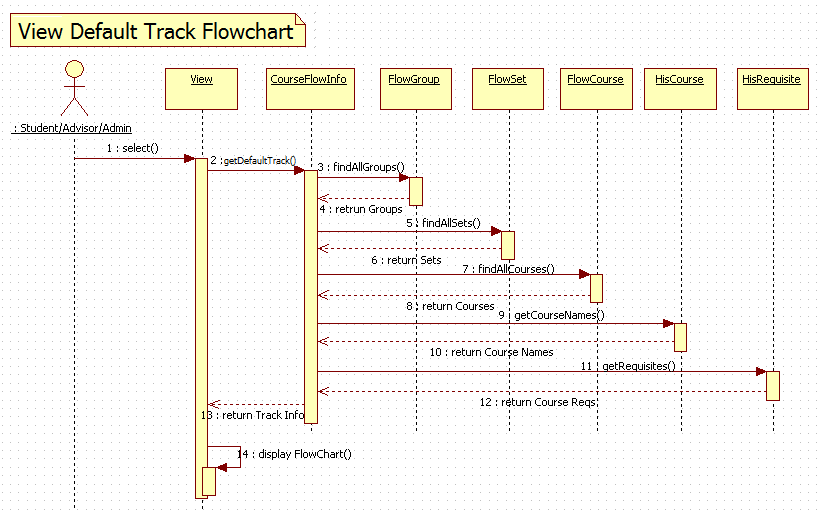
UCMSv2-0005 : View Prospective Catalog



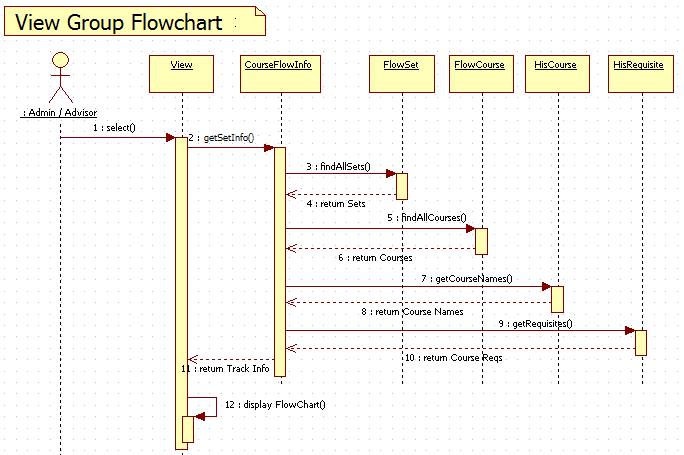
UCMSv2-0006 : View Track Flowchart (a)



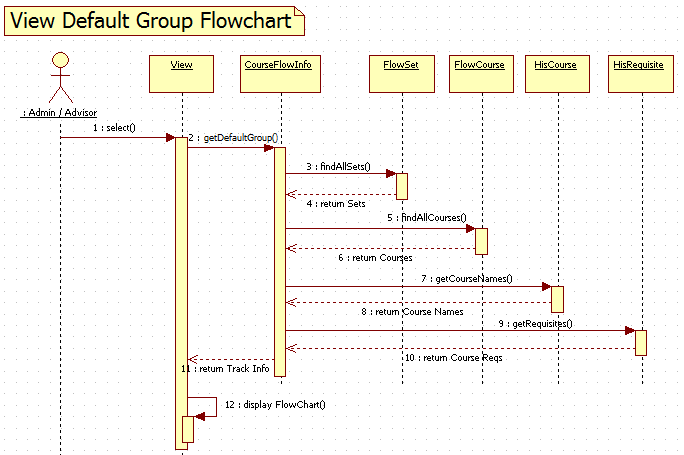
UCMSv2-0006 : View Track Flowchart (b)



UCMSv2-0007 : View Group Flowchart (a)



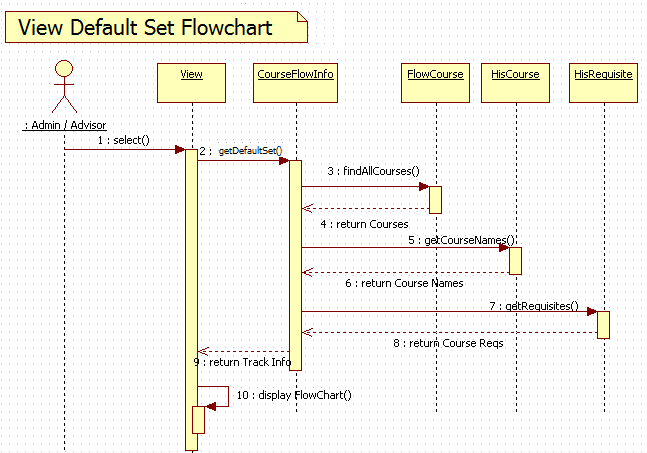
UCMSv2-0007 : View Group Flowchart (b)



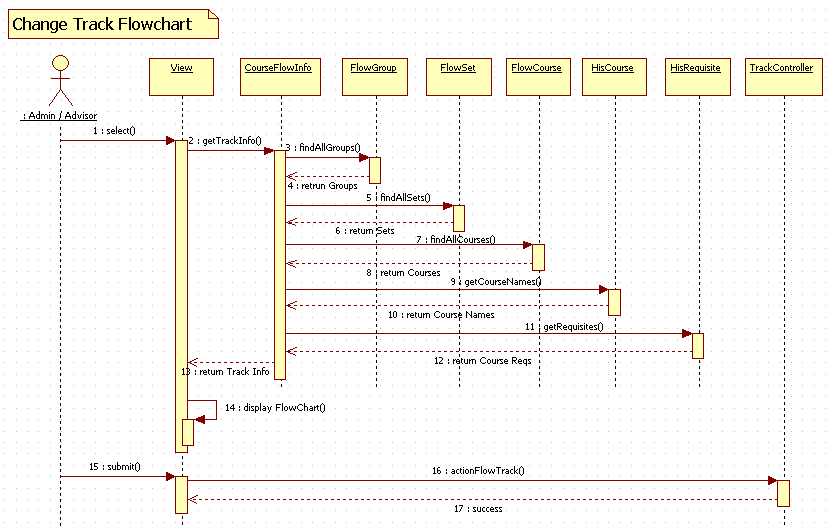
UCMSv2-0008 : View Set Flowchart (a)



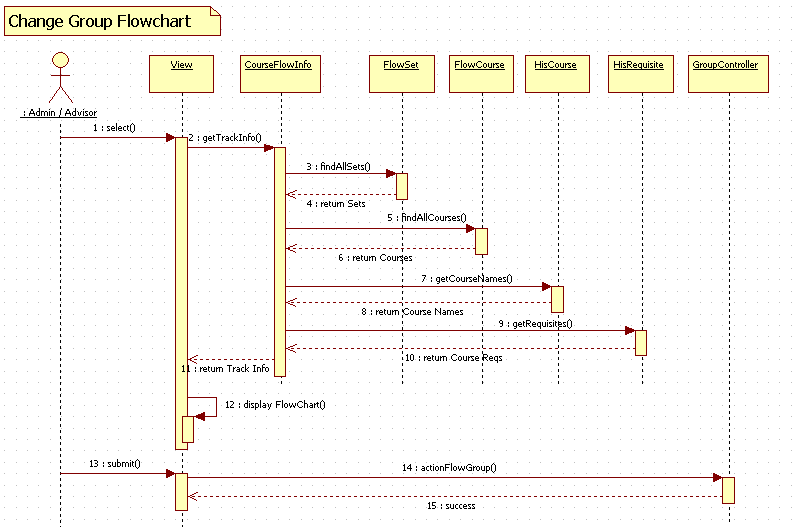
UCMSv2-0008 : View Set Flowchart (b)



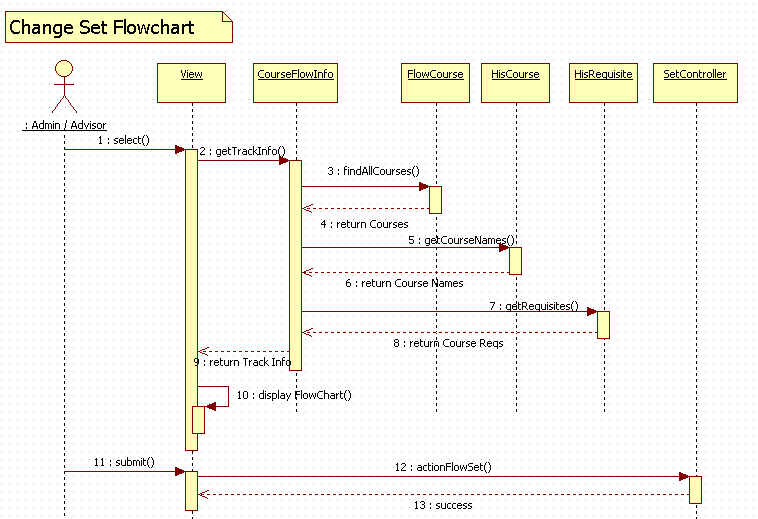
UCMSv2-0009 : Change Track Flowchart



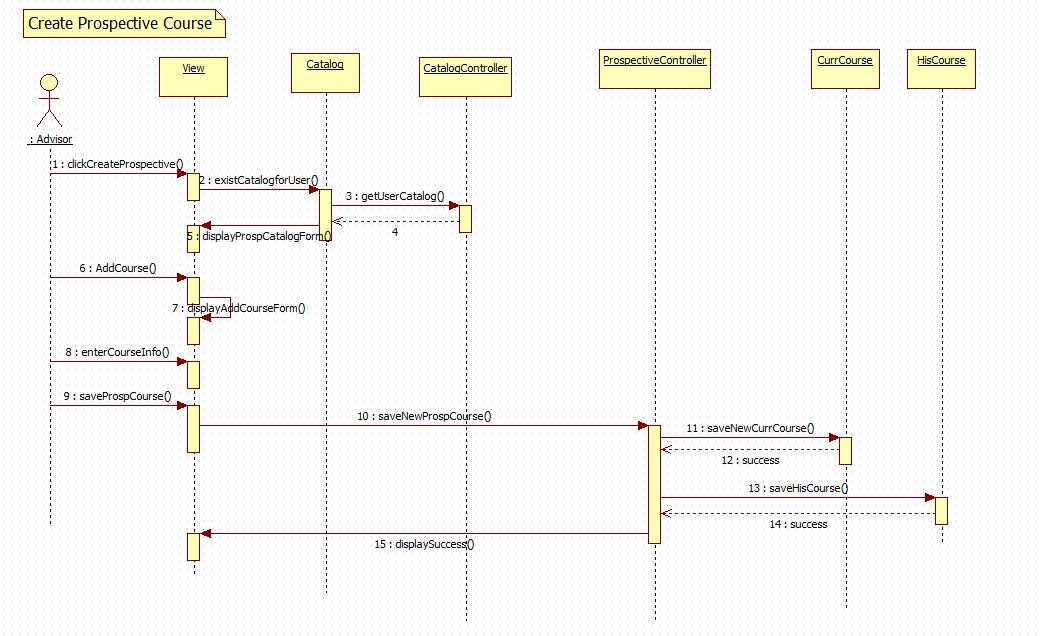
UCMSv2-0010 : Change Group Flowchart



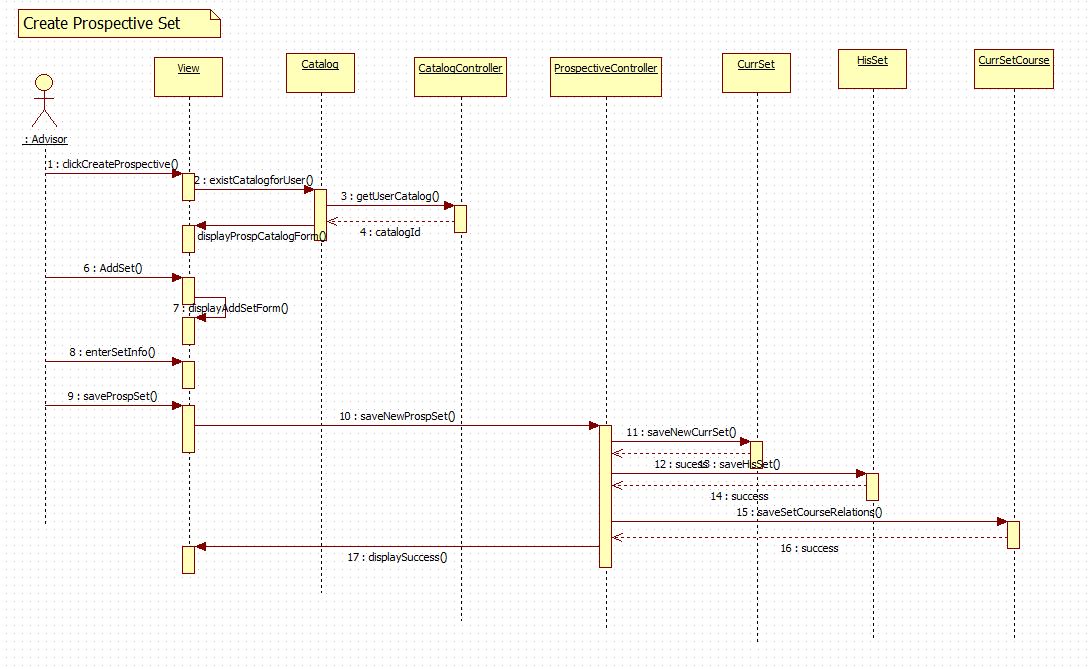
UCMSv2-0011 : Change Set Flowchart



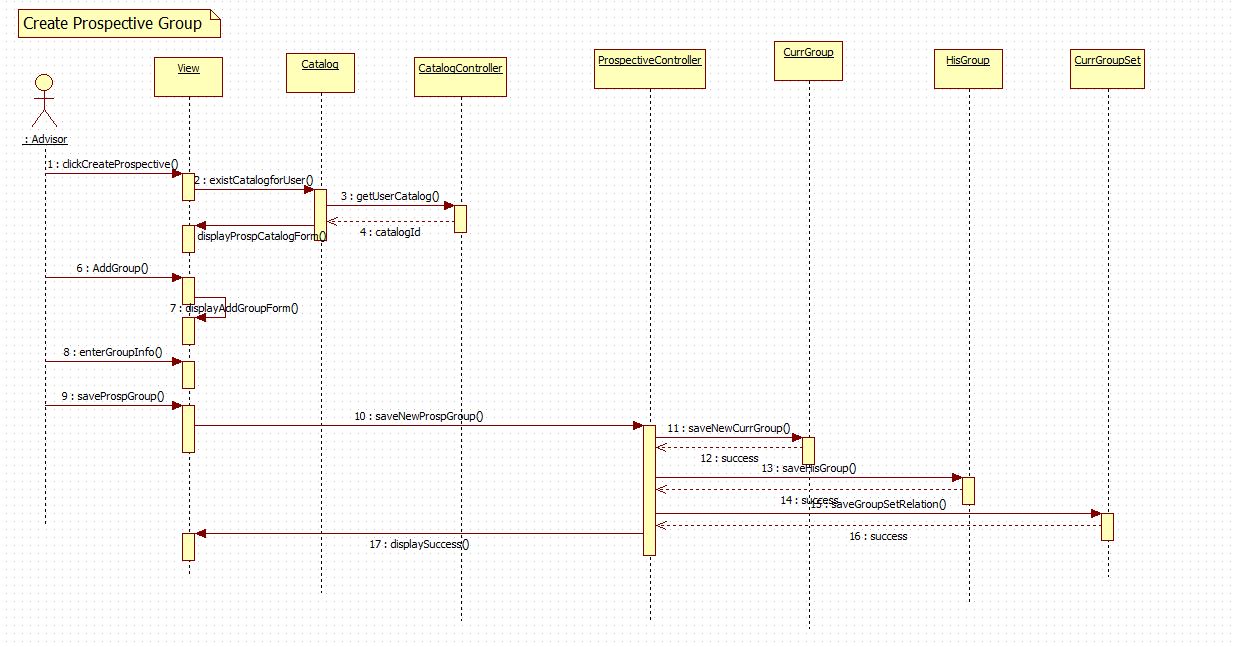
UCMSv2-0012 : Create Prospective Course



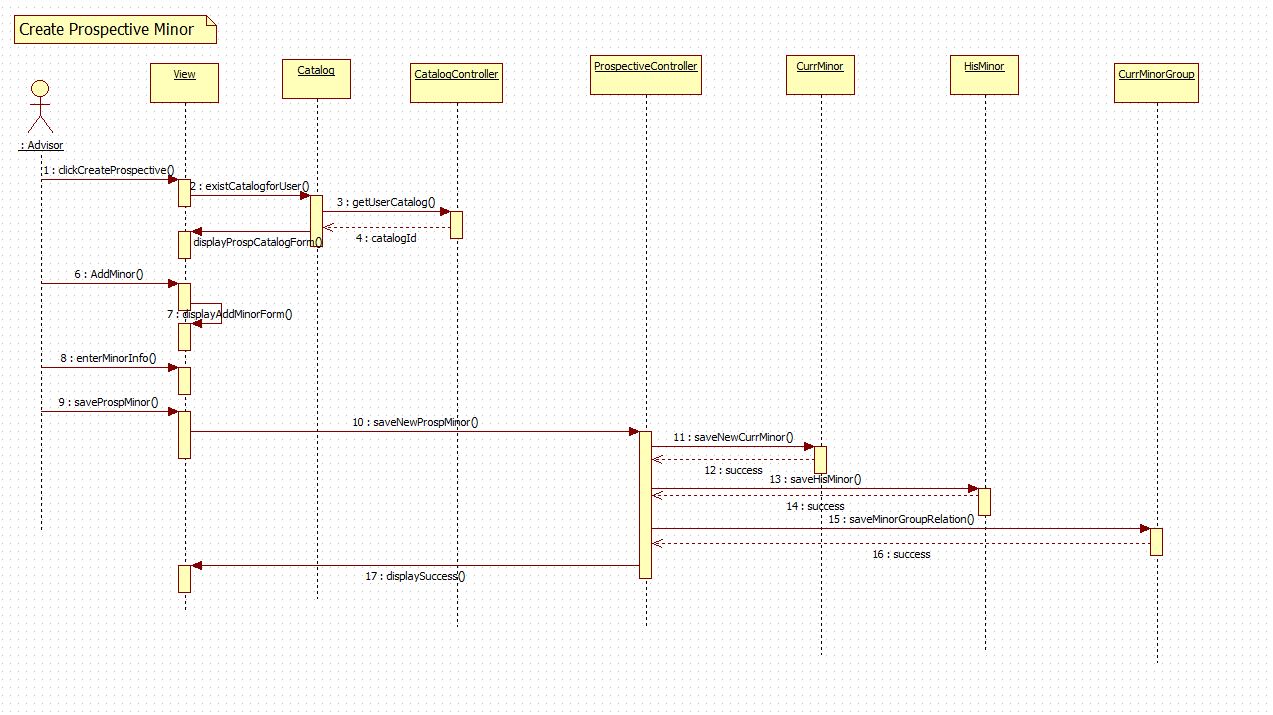
UCMSv2-0013 : Create Prospective Set



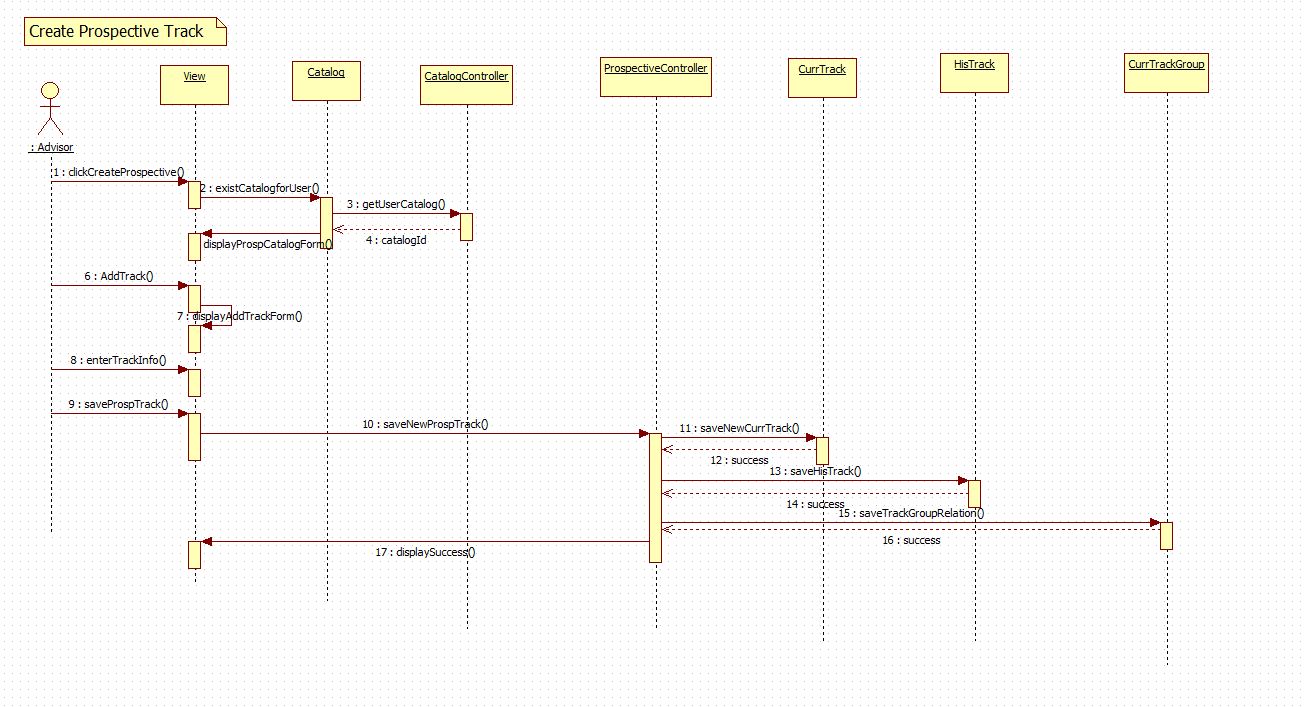
UCMSv2-0014 : Create Prospective Group



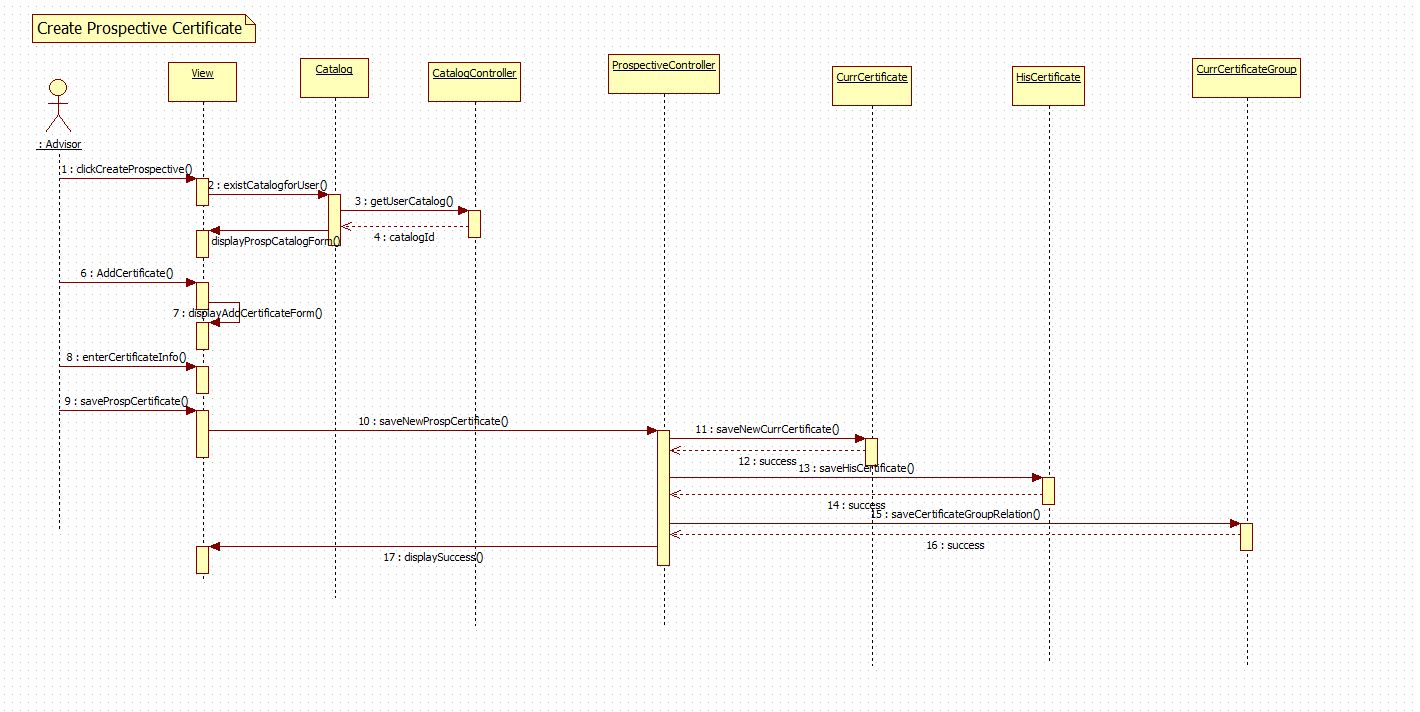
UCMSv2-0015 : Create Prospective Minor



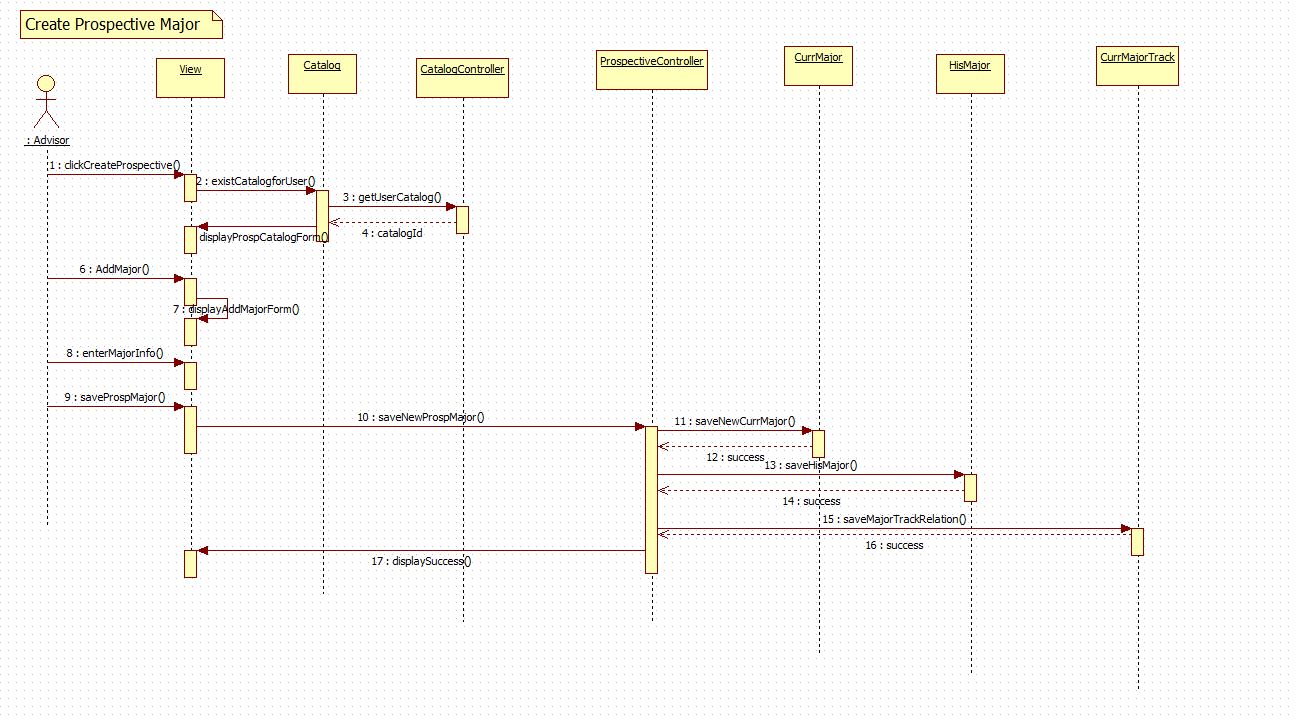
UCMSv2-0016 : Create Prospective Track



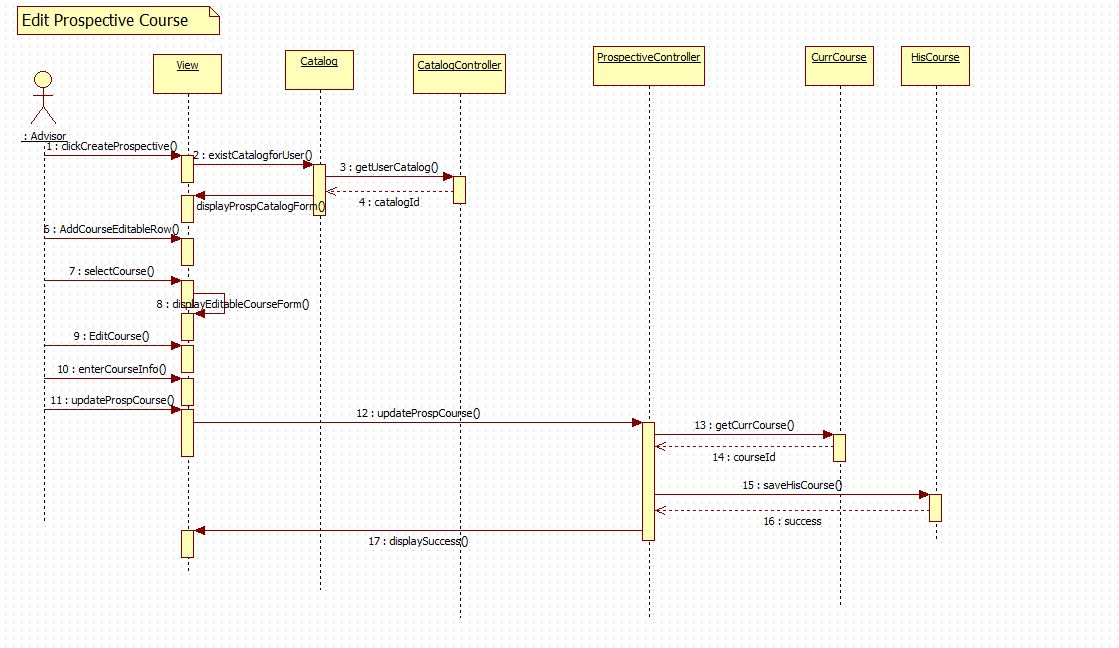
UCMSv2-0017: Create Prospective Certificate



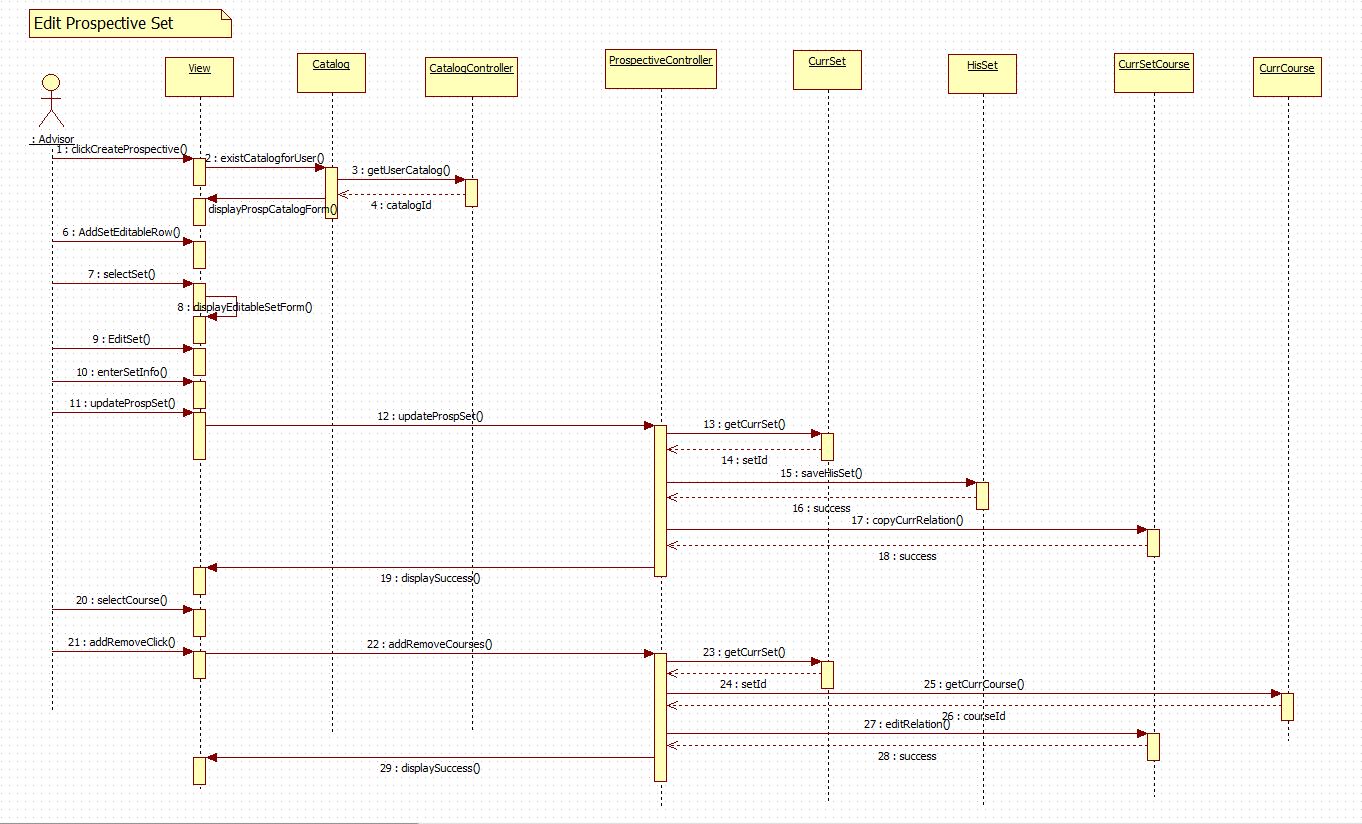
UCMSv2-0018 : Create Prospective Major



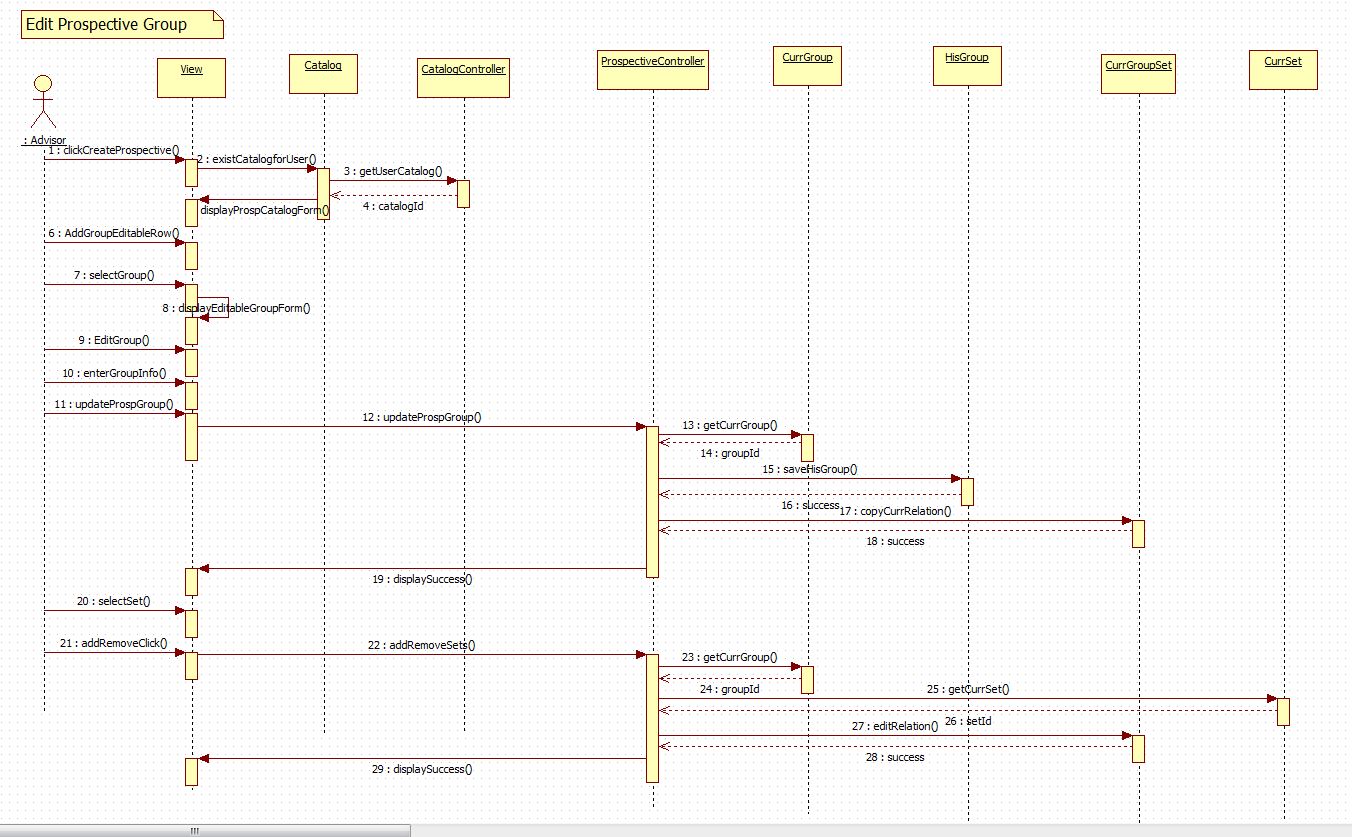
UCMSv2-0019 : Edit Prospective Course



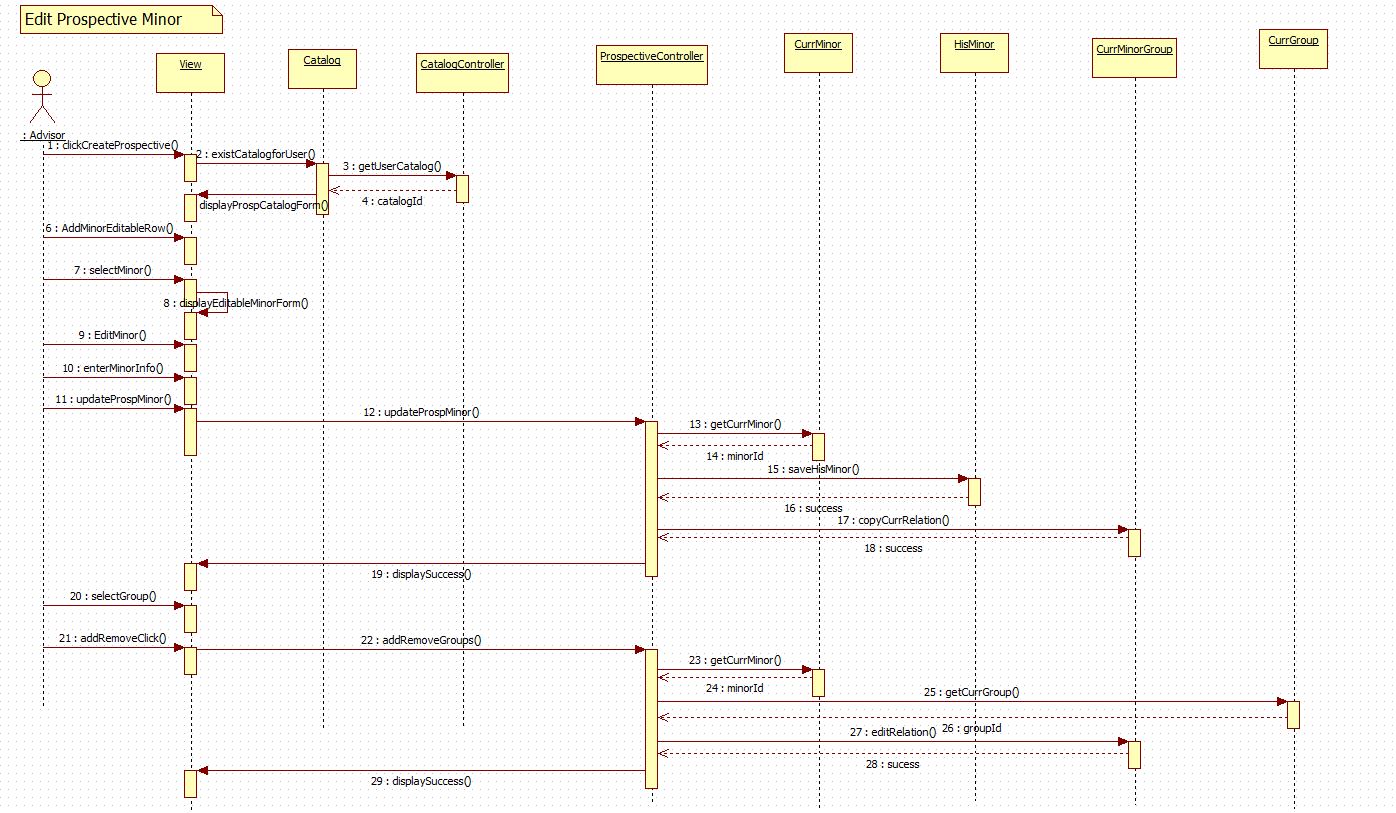
UCMSv2-0020 : Edit Prospective Set



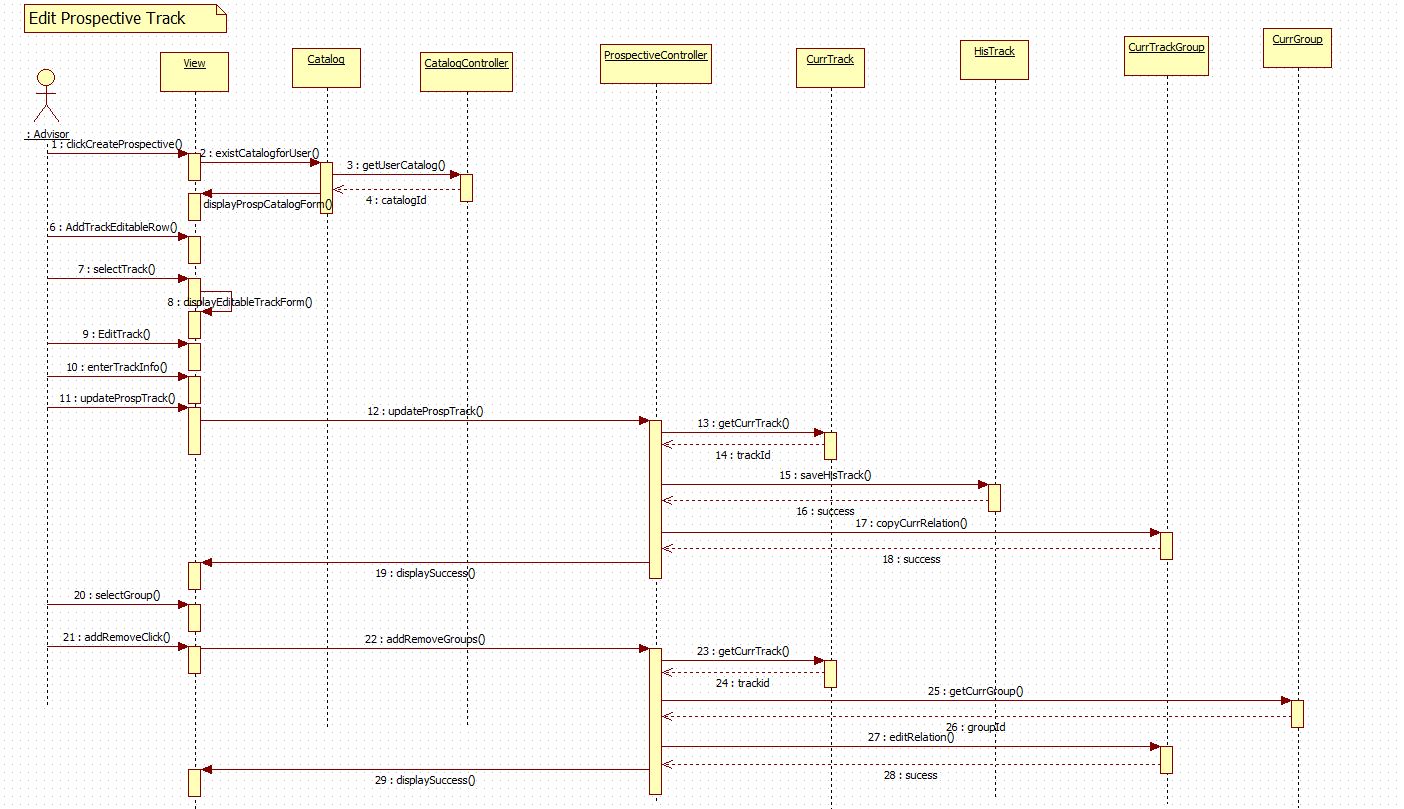
UCMSv2-0021 : Edit Prospective Group



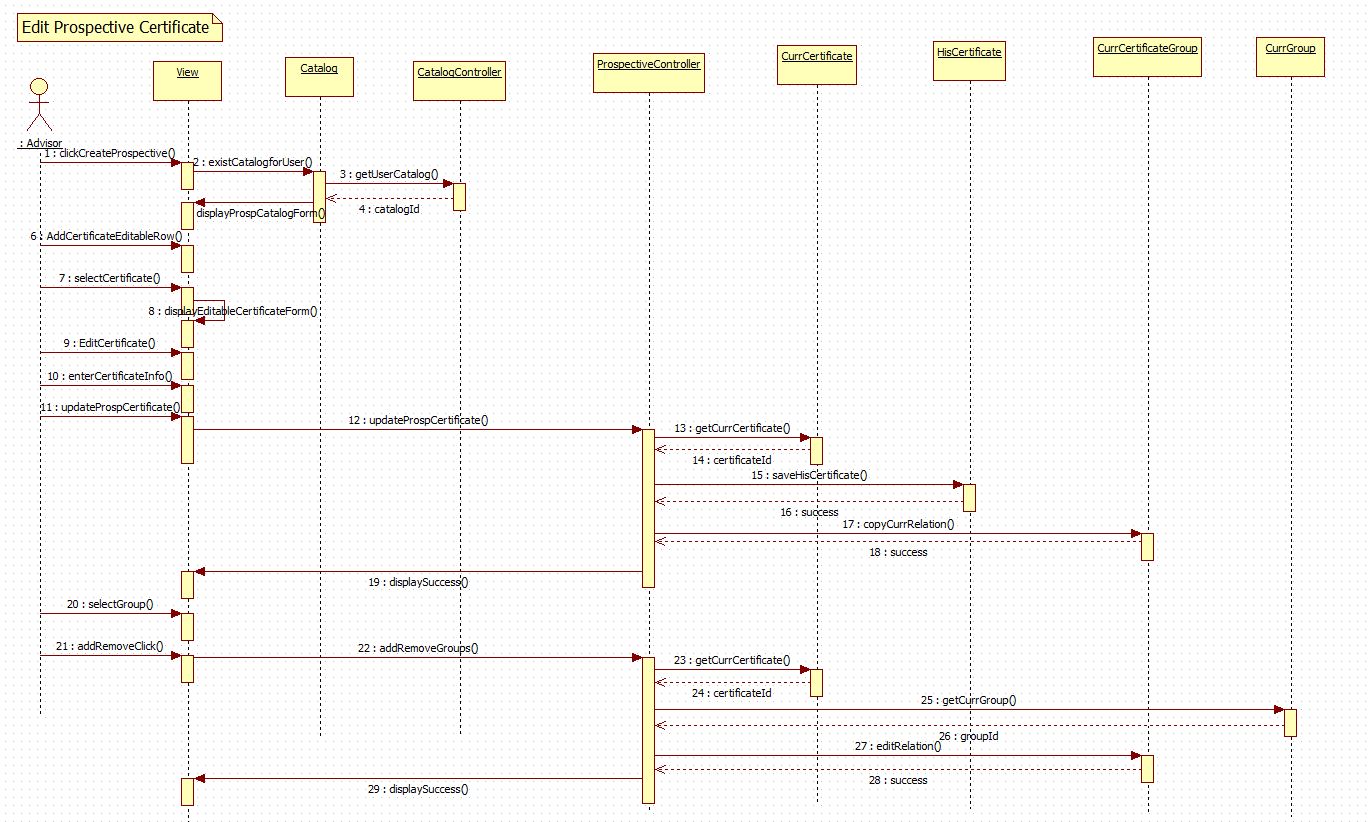
UCMSv2-0022 : Edit Prospective Minor



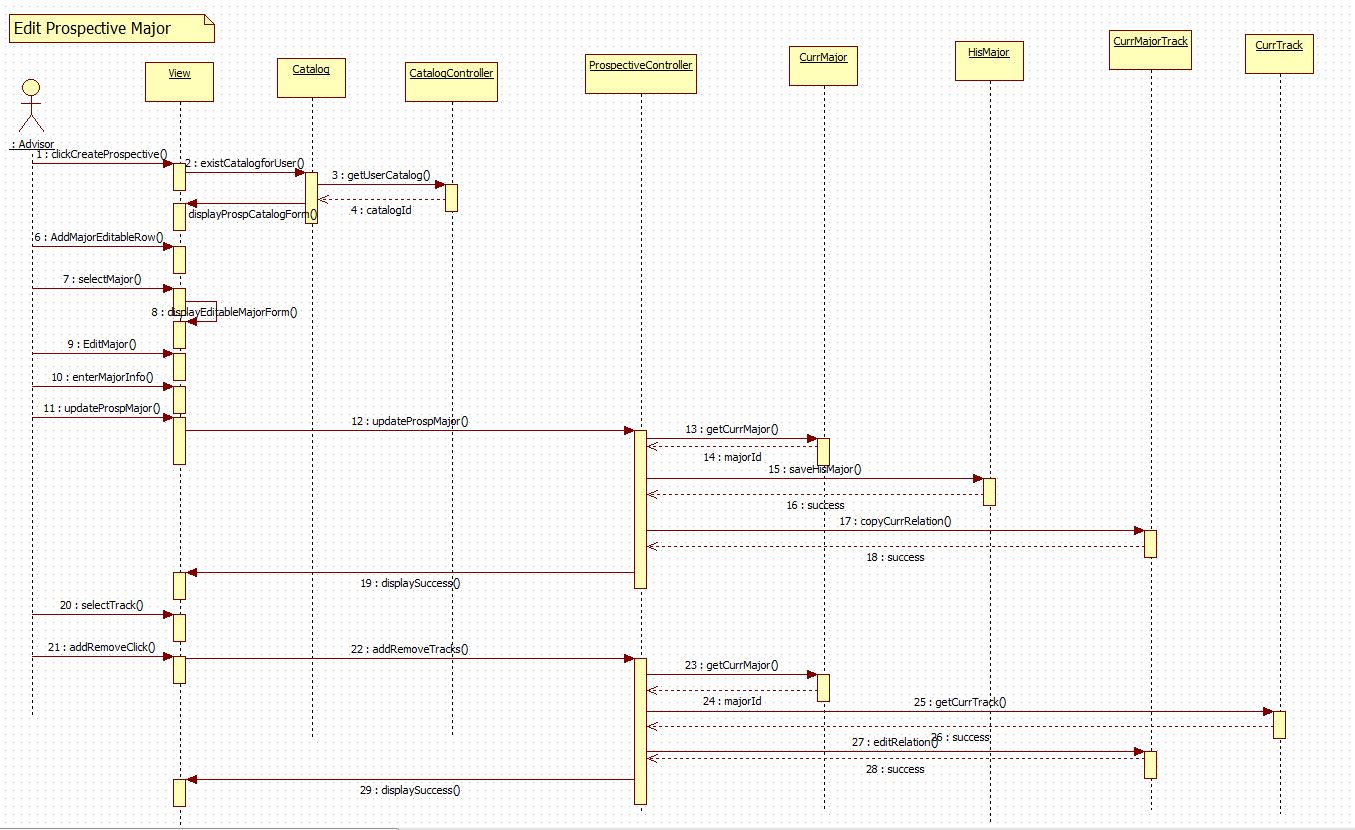
UCMSv2-0023 : Edit Prospective Track



UCMSv2-0024 : Edit Prospective Certificate

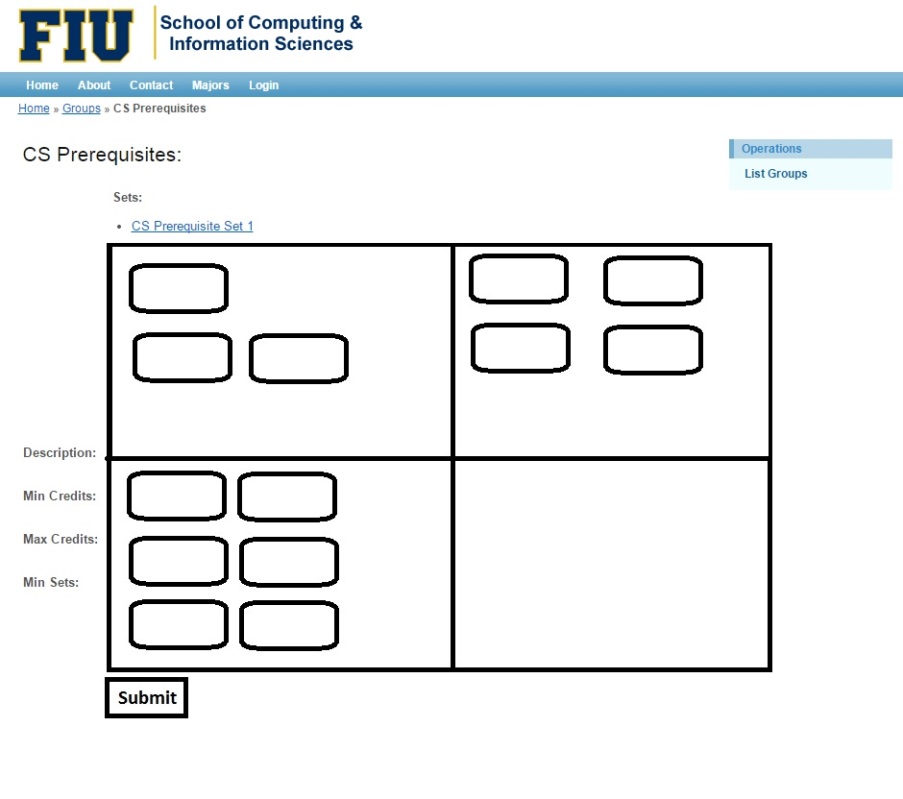


UCMSv2-0025 : Edit Prospective Major

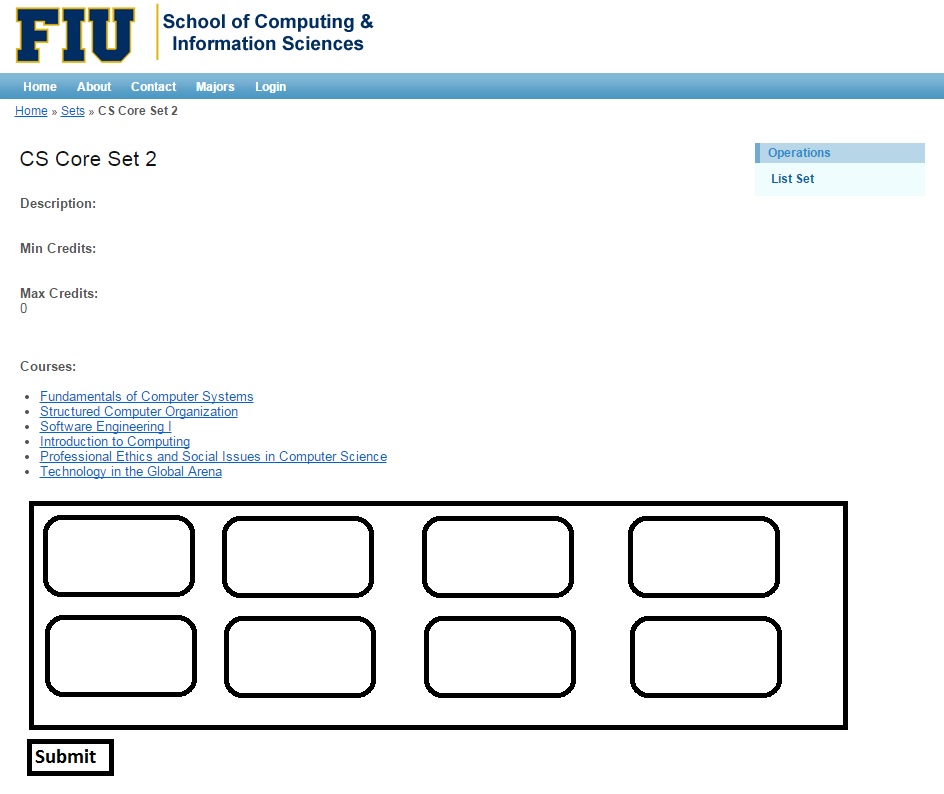


## Appendix E – User Interface Design

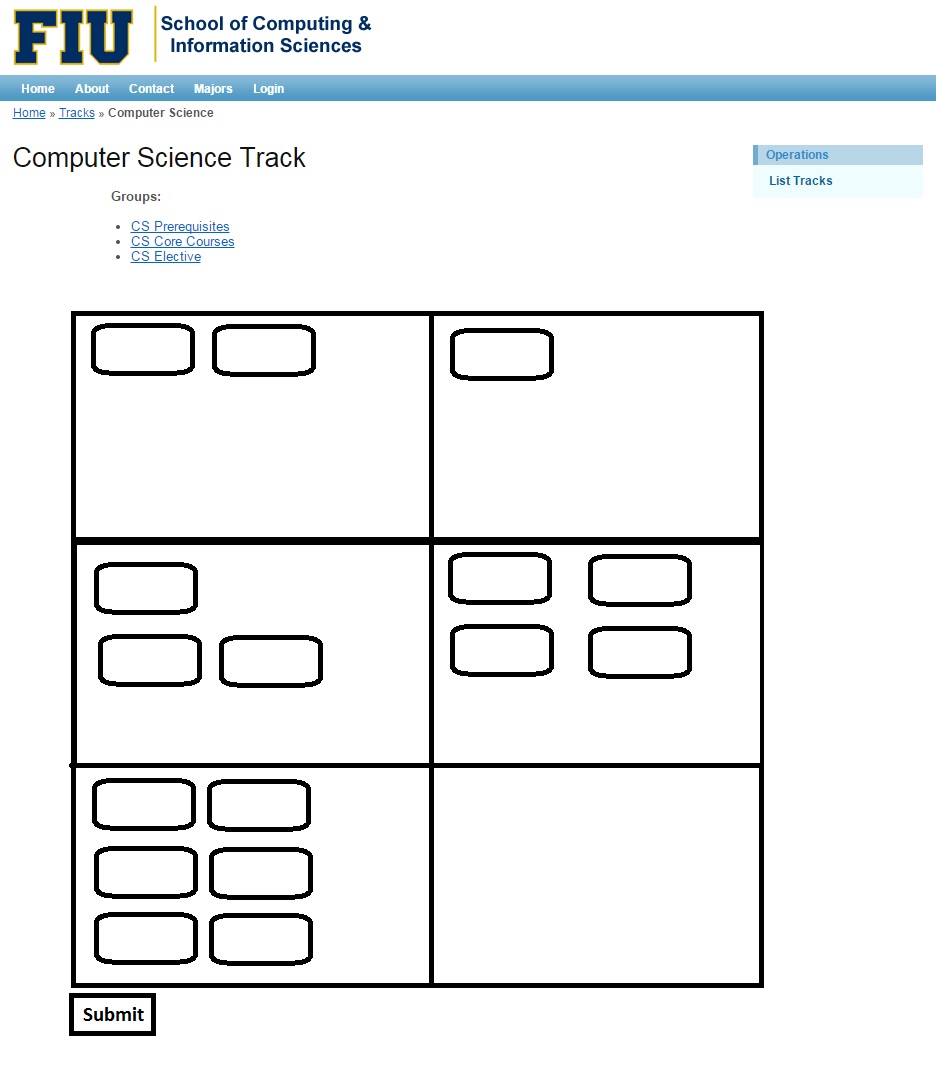
Group Flow



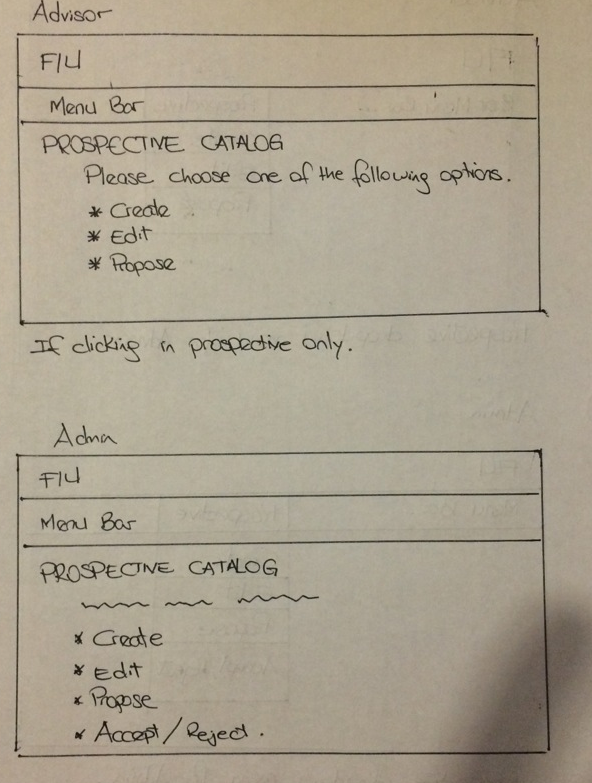
Set Flow



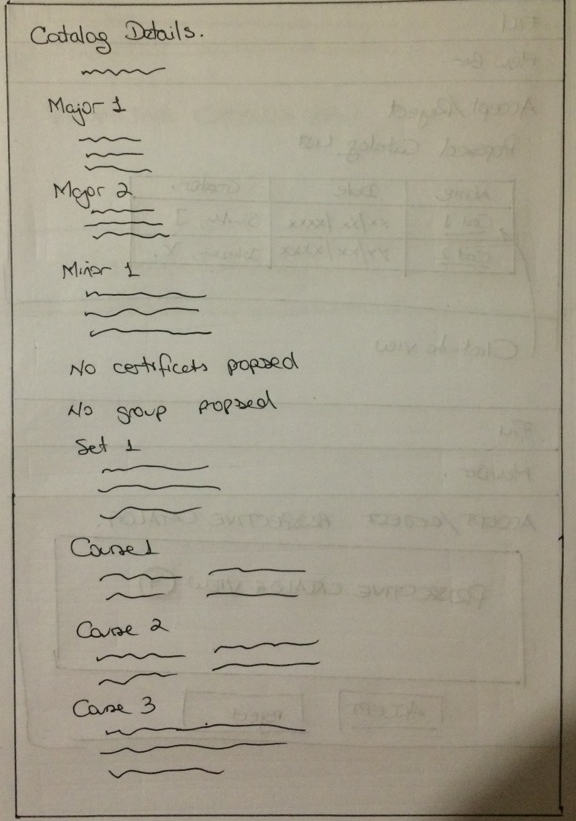
Track Flow



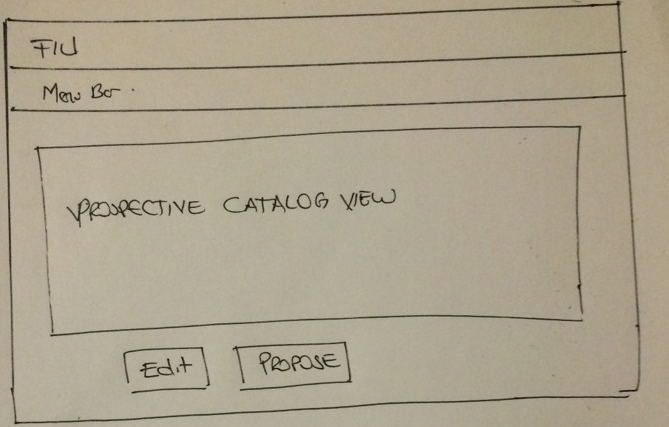
Prospective Catalog Menu



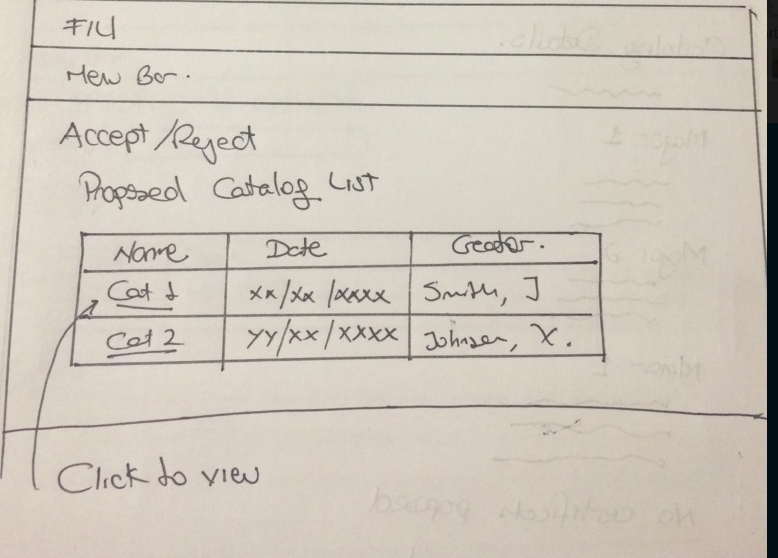
View Prospective Catalog



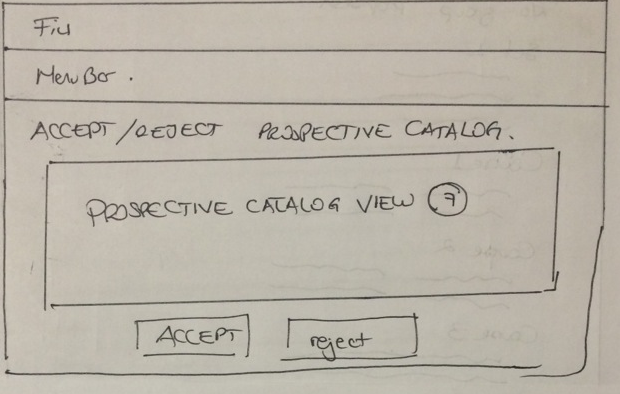
Propose Prospective Catalog



Accept and Reject Prospective Catalog Selection



Accept and Reject Prospective Catalog



## Appendix F – Diary of Meeting and Tasks

|  |  |
| --- | --- |
| Diary Entry 1 | |
| Date | Wednesday, September 3rd, 2014 |
| Location | ECS 341 |
| Start | 7:00 PM |
| End | 8:00 PM |
| Attendees | * Tim Downey * Jose Astudillo * Christopher Sutton |
| Agenda | * Review the existing system * Get requirements for the project |
| Summary | * Defined tools to be used for the development * Explanation of the current system * Brief definition of the functionalities to be implement for this version of the system |
| Assigned Tasks | For both team members:   * Explore current system * Get familiar with the tools to be used. |

|  |  |
| --- | --- |
| Diary Entry 2 | |
| Date | Sunday, September 7th, 2014 |
| Location | JCCL Lab |
| Start | 4:00 PM |
| End | 7:00 PM |
| Attendees | * Jose Astudillo * Christopher Sutton |
| Agenda | * Start working on project plan * Revise Feasibility Analysis document * Revise SRD * Work on Trello |
| Summary | * Trello was set up. * Worked on Feasibility document * Worked on SRD. * Brief work on project plan. |
| Assigned Tasks | Jose: work on SRD.  Christopher: work on feasibility document. |

|  |  |
| --- | --- |
| Diary Entry 3 | |
| Date | Monday, September 8th, 2014 |
| Location | ECS 341 |
| Start | 7:00 PM |
| End | 8:00 PM |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Run v1.0 locally * Discuss functionalities to be implemented. |
| Summary | * Tried to run v1.0 locally. * Discussed about adding new user. Now, there will be 4 types of users: admin, student, advisors, and the regular visitor. * Christopher was assigned to work on administrator modules; additionally, he might do some work on regular user. * Jose was assigned to work on advisor module mainly; however, he might work also in the student module. |
| Assigned Tasks | * Keep trying to get v1.0 to run locally. * Start writing use cases, and create sequence diagrams. |

|  |  |
| --- | --- |
| Diary Entry 4 | |
| Date | Wednesday, September 10th, 2014 |
| Location | ECS 341 |
| Start | 7:00 PM |
| End | 8:00 PM |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Run v1.0 of CMS locally |
| Summary | * Got v1.0 runnning |
| Assigned Tasks | * Keep trying to get v1.0 to run locally. * Start writing use cases, and create sequence diagrams. * Familiarize with Yii framework * Get use to the database |

|  |  |
| --- | --- |
| Diary Entry 5 | |
| Date | Monday, September 15th, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 7:30 PM |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Discuss what next to be done |
| Summary | * Discussed what needs to be done |
| Assigned Tasks | * Complete documents for use cases. * Make sequence diagrams from the use cases. * Create UI for the use cases to be implemented. |

|  |  |
| --- | --- |
| Diary Entry 6 | |
| Date | Monday, September 29th, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 7:30 PM |
| Attendees | * Jose Astudillo * Tim Downey |
| Agenda | * Talk about create, edit, and propose prospective catalogs. |
| Summary | * Discussed about the flow for creating, editing, and proposing prospective catalogs. |
| Assigned Tasks | * Keep writing use cases, and creating diagrams |

|  |  |
| --- | --- |
| Diary Entry 7 | |
| Date | Wednesday, October 1st, 2014 |
| Location | ECS 341 |
| Start |  |
| End |  |
| Attendees | * Tim Downey * Christopher Sutton |
| Agenda | * Discuss about Yii framework |
| Summary | * Talked about the controller and view set ups for degree unit |
| Assigned Tasks |  |

|  |  |
| --- | --- |
| Diary Entry 8 | |
| Date | Monday, October 6th, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 7:30 PM |
| Attendees | * Jose Astudillo * Tim Downey * Christopher Sutton |
| Agenda | * Discuss changes to the database for implementing prospective catalogs. * Discuss changes for implementing new type of users. * Discuss algorithm for flow chart implementation. |
| Summary | * Discussed changes to the database for implementing prospective catalogs. * Discussed changes for implementing new type of users. * Permission granted by Tim Downey to work in flow chart. |
| Assigned Tasks | Jose:   * Start creating view for the system. * Make changes to the database so that it can accept new users. * Make changes to the catalog table so that it can work with prospective catalogs.   Chris:   * Work on flow chart algorithm |

|  |  |
| --- | --- |
| Diary Entry 9 | |
| Date | Wednesday, October 8th, 2014 |
| Location | ECS 341 |
| Start |  |
| End |  |
| Attendees | * Tim Downey * Christopher Sutton |
| Agenda | * Show graphical API to use on the flow chart implementation |
| Summary | * Two API were show: Tree Map, and Org Chart. |
| Assigned Tasks | * Put required information into charts which can then be linked at a later date. |

|  |  |
| --- | --- |
| Diary Entry 10 | |
| Date | Friday, October 10th, 2014 |
| Location | ECS 341 |
| Start |  |
| End |  |
| Attendees | * Tim Downey * Christopher Sutton |
| Agenda | * Discussed table structures |
| Summary | * Learned how to use tables created as a base point for linking. * Explored tables currently being used by panther soft. |
| Assigned Tasks |  |

|  |  |
| --- | --- |
| Diary Entry 11 | |
| Date | Monday, October 13th, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 8:30 |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Discuss concern about the presentation * Talk about topic suggested to automate students schedule for the next semester. * Discuss pop up implementation. |
| Summary | * The automation need was outside the project according to Tim Downey. * Decided to implement the flowchart using DOM. |
| Assigned Tasks | Jose:   * Work on pop up.   Christopher:   * Start working on flowchart DOM. |

|  |  |
| --- | --- |
| Diary Entry 12 | |
| Date | Wednesday, October 15th, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 7:30 PM |
| Attendees | * Jose Astudillo * Christopher Sutton |
| Agenda | * Discuss Yii framework. * Discuss process to use DOM to create flow chart. |
| Summary | * Discussed problems using Yii active forms. * Discovered a large data gap; no pre/co requisites were listed in the databases. * Chris proposed a schema to implement which was accepted by Tim Downey. |
| Assigned Tasks | Jose:   * Keep working on pop up forms.   Christopher:   * Keep working on flow chart implementation. |

|  |  |
| --- | --- |
| Diary Entry 13 | |
| Date | Friday, October 17th, 2014 |
| Location | ECS 341 |
| Start |  |
| End |  |
| Attendees | * Tim Downey. * Christopher Sutton |
| Agenda | * Show Downey new tables. |
| Summary | * Chris was provided with an example of DOM programs. |
| Assigned Tasks | * Ensure courses can be moved dynamically around the page. |

|  |  |
| --- | --- |
| Diary Entry 14 | |
| Date | Monday, October 20th , 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 8:30 PM |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Discuss about implementation of pop up from using Yii active form, or using JQuery. * Show current flexibility and layout of the dynamic flow chart form. |
| Summary | * Discussed methods that would allow saving the layout and reloaded on refresh. * Discussed about using JQuery. |
| Assigned Tasks | Jose:   * Keep working on forms.   Christopher:   * Look at methods for saving layout. |

|  |  |
| --- | --- |
| Diary Entry 15 | |
| Date | Wednesday, October 22nd, 2014 |
| Location | ECS 341 |
| Start | 6:30 PM |
| End | 8:30 PM |
| Attendees | * Jose Astudillo * Christopher Sutton * Tim Downey |
| Agenda | * Discuss problems in flowchart. |
| Summary | * Decided to use JQuery to pop up form for the different inputs of the prospective. * Fixed minor bug issues. * Showed current dynamic functionality. |
| Assigned Tasks | Jose:   * Keep working on pop up forms.   Christopher:   * Working on saving layout. |

|  |  |
| --- | --- |
| Diary Entry 16 | |
| Date | Friday, October 24th, 2014 |
| Location | ECS 341 |
| Start |  |
| End |  |
| Attendees | * Tim Downey * Christopher Sutton |
| Agenda | * Discuss program. |
| Summary | * Errors fixed in JS. * Left columns fixed. * Discussed on saving object location to database. |
| Assigned Tasks | Work on table that can link a degree track to a flowchart id.  Christopher: |

|  |  |
| --- | --- |
| Diary Entry 17 | |
| Date |  |
| Location |  |
| Start |  |
| End |  |
| Attendees | * Jose Astudillo * Christopher Sutton |
| Agenda |  |
| Summary |  |
| Assigned Tasks | Jose:  Christopher: |

|  |  |
| --- | --- |
| Diary Entry 18 | |
| Date |  |
| Location |  |
| Start |  |
| End |  |
| Attendees | * Jose Astudillo * Christopher Sutton |
| Agenda |  |
| Summary |  |
| Assigned Tasks | Jose:  Christopher: |

## References