**University Catalog Management System Version 2.0**

Mentor:

Tim Downey

Team Members:

Jose Astudillo

Christopher Sutton

Senior Project

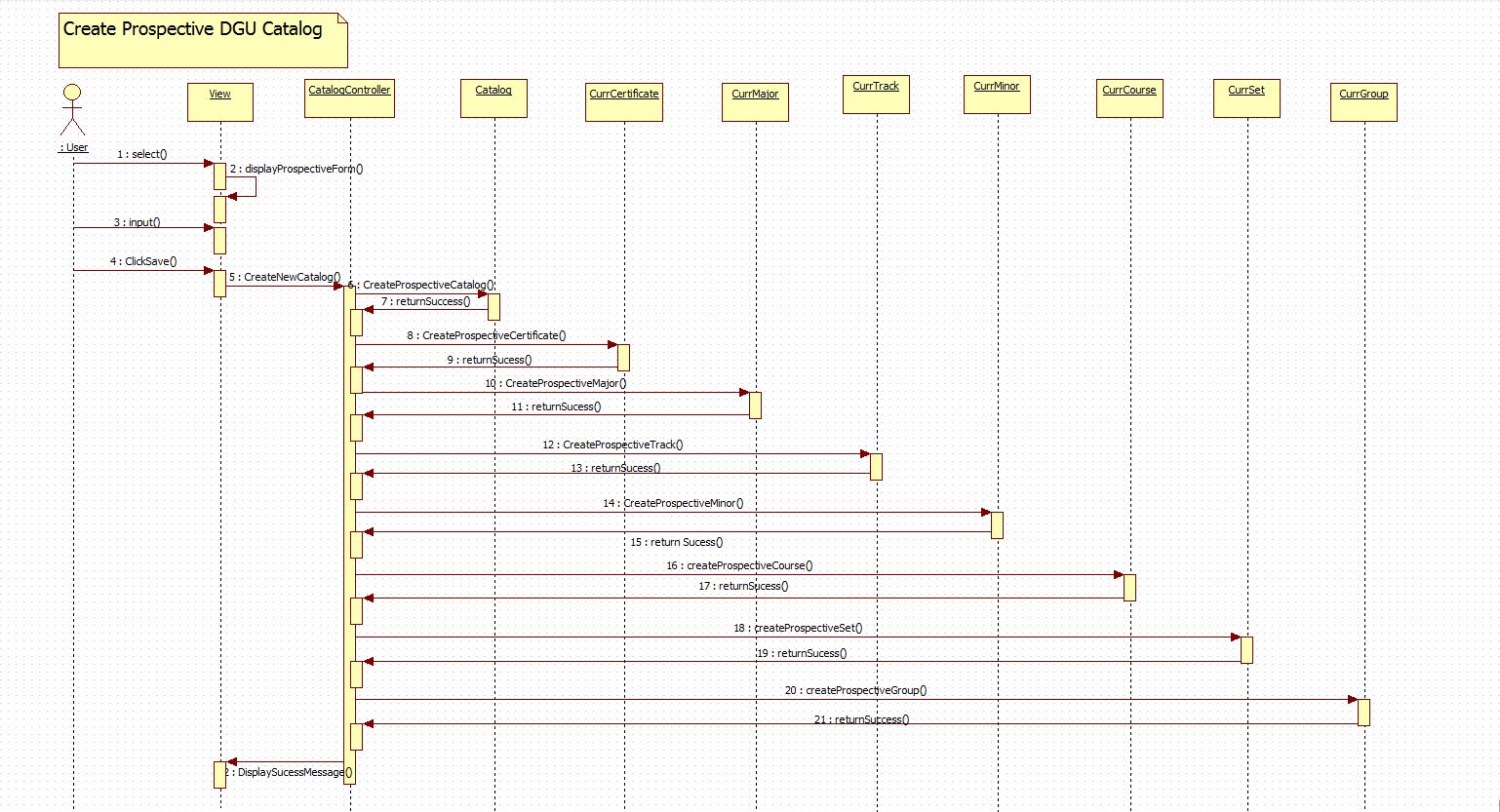
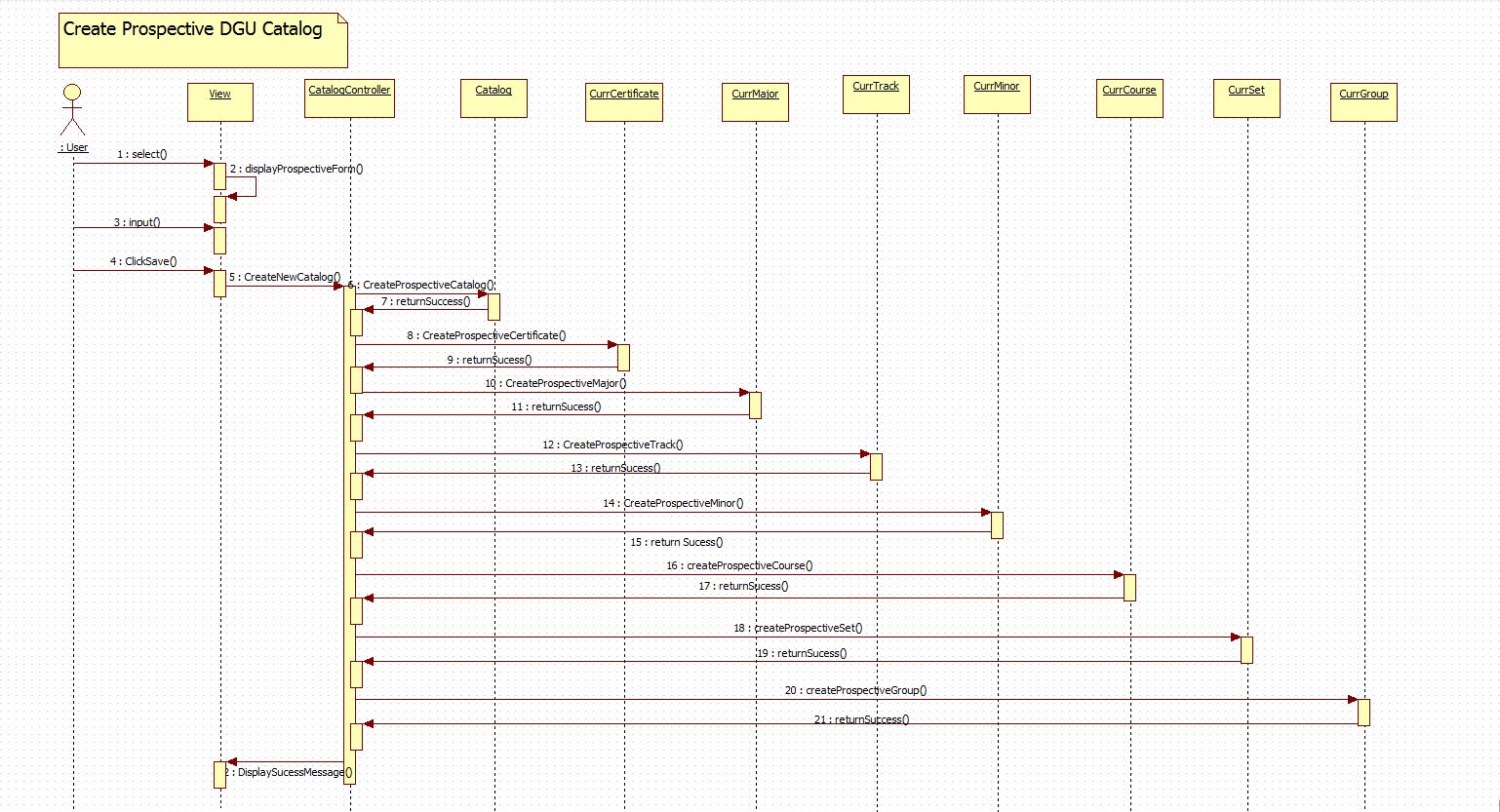
CIS 4911 – U01

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.

Contents

[1. Introduction 3](#_Toc402118752)

[1.1. Problem Definition 3](#_Toc402118753)

[1.2. Scope of the System 3](#_Toc402118754)

[1.3. Terminology 3](#_Toc402118755)

[1.3.1. Definitions 3](#_Toc402118756)

[1.3.2. Acronyms 3](#_Toc402118757)

[1.3.3. Abbreviations 4](#_Toc402118758)

[1.4. Overview of the System 4](#_Toc402118759)

[2. Current System (Limitation and Problems) 5](#_Toc402118760)

[3. Project Plan 6](#_Toc402118761)

[3.1. Project Organization 6](#_Toc402118762)

[3.2. Work Breakdown 6](#_Toc402118763)

[3.3. Cost Estimate 7](#_Toc402118764)

[4. Proposed System Requirements 8](#_Toc402118765)

[4.1. Functional Requirements 8](#_Toc402118766)

[4.2. Analysis of the System Requirements 9](#_Toc402118767)

[4.2.1. Scenarios 9](#_Toc402118768)

[4.2.2. Use Case Model 9](#_Toc402118769)

[4.2.3. Static Model 9](#_Toc402118770)

[4.2.4. Dynamic Model 9](#_Toc402118771)

[5. Glossary 10](#_Toc402118772)

[5.1. Definitions 10](#_Toc402118773)

[5.2. Acronyms 10](#_Toc402118774)

[5.3. Abbreviations 10](#_Toc402118775)

[6. Appendix 11](#_Toc402118776)

[6.1. Appendix A – Complete Use Cases 11](#_Toc402118777)

[6.2. Appendix B – Use Case Diagram Using UML 27](#_Toc402118778)

[6.3. Appendix C – Static UML Diagram 28](#_Toc402118779)

[6.4. Appendix D – Dynamic UML Diagram 29](#_Toc402118780)

[6.5. Appendix E – User Interface Design 30](#_Toc402118781)

[6.6. Appendix F – Diary of Meeting and Tasks 31](#_Toc402118782)

[7. References 33](#_Toc402118783)

## 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.

## 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 to be used in 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 look throughout all the different books that contain information about the different programs of study to find out 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 of some key functionality. 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 of 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 super 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 requirement and their respective nonfunctional requirements.

## Functional Requirements

The system has three different types of users. Below there is list of what will be implemented for each user. In addition, a list of common functionalities among the users is provided.

**Student Module Functional Requirements**

There are no functional requirements proposed for the user at this point.

**Administrator Module Functional Requirements**

The system shall…

* Allow administrator to activate proposed catalog.
* Allow administrator to reject proposed catalogs.
* Allow administrator to make major changes in currently active catalogs.
* Allow administrator to provide authorizations to different user in the system.
* Allow administrator to enable users.
* Allow administrator to disable users.

**Advisor Module Functional Requirements**

The system shall…

* Allow advisor to propose catalogs.
* Allow advisor to view two different catalogs at the same time.
* Allow advisor to comment old catalogs.

**General Functional Requirements**

The system shall…

* Be able to export .xml files.
* Be able to generate catalogs automatically.
* Be able to generate flowchart for the different programs in an automated manner (possibly).

## Analysis of the System Requirements

## Scenarios

## 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

## 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. Admin or advisor must be logged in into the system.

**Description :**

1. Use case begins when Admin or Advisor clicks on the “Prospective Catalog” button.
2. Use select option “Create Prospective Catalog” from the prospective catalog page.
3. User can add prospective course(s) with its/their descriptions, if any to be added.
4. User can add prospective major(s) with its/their descriptions, if any to be added.
5. User can add prospective minor(s) with its/their descriptions, if any to be added.
6. User can add prospective track(s) with its/their descriptions, if any to be added.
7. User can add prospective group(s) with its/their descriptions, if any to be added.
8. User can add prospective set(s) with its/their descriptions, if any to be added.
9. User can add prospective certificate(s) with its/their descriptions, if any to be added.
10. Use case end when user clicks on “Save” button.

**Post condition(s):**

1. Prospective catalog must be saved in the database.

**Alternative Course of Action:** None.

**Related Use Cases:** None

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0002

**Name :** Update DGU Proposed Catalog

**Details :** This use case allows Admin and advisor edit proposed catalogs for their corresponding DGU. This includes changes to the prospective courses, major, minor, certificates, and tracks, groups, and set previously proposed.

**Actors :** Admin, and advisor.

**Pre-condition(s):**

1. Admin or 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 “Edit Prospective Catalog” from the prospective catalog page.
3. User can edit prospective course(s) with its/their descriptions, if any to be added.
4. User can edit prospective major(s) with its/their descriptions, if any to be added.
5. User can edit prospective minor(s) with its/their descriptions, if any to be added.
6. User can edit prospective track(s) with its/their descriptions, if any to be added.
7. User can edit prospective group(s) with its/their descriptions, if any to be added.
8. User can edit prospective set(s) with its/their descriptions, if any to be added.
9. User can edit prospective certificate(s) with its/their descriptions, if any to be added.
10. Use case end when user clicks on “Save” button.

**Post condition(s):**

1. Changes to the prospective catalog must be saved in the database.

**Alternative Course of Action:** Before posting a prospective catalog, advisors are given the choice to edit the prospective catalog.

**Related Use Cases:** UCMSv2 – 0001, UCMSv2 - 0003

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0003

**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 “Post Prospective Catalog” from the prospective catalog page.
3. System must display the prospective catalog.
4. Use case ends when user clicks on post.

**Post condition(s):**

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

**Alternative Course of Action:** None.

**Related Use Cases:** UCMSv2 – 0001

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0004

**Name :** Create Accounts

**Details :** This use case allows a user with super admin privileges to create user accounts.

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “Create Accounts” button. After clicking on the button the user will be redirected to the create accounts page.
2. On this screen the user must first select the type of account to create.
3. The super admin must then enter the following information:
   1. Username, password, email, first name, last name, and birthday.
4. Once the information has been validated as the correct format the user can click on the button saying “Create Account” to create the account profile.

**Post condition(s):**

1. The user profile is added to the database.
2. Email is sent out to the users email address.

**Alternative Course of Action:** Instead of clicking the “Create Account” button a user can click “Cancel” to cancel the operation.

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0005

**Name :** Delete Accounts

**Details :** This use case allows a user with super admin privileges to delete user accounts.

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “Delete Accounts” button. After clicking on the button the user will be redirected to the delete accounts page.
2. On this page a table will be displayed showing a list of all users in the system.
3. Next to each user is a delete button.
4. When the delete button is clicked the user will be removed from the system.

**Post condition(s):**

1. The user profile is removed from the database.
2. Email is sent out to the users email address informing them that their account is no longer valid.

**Alternative Course of Action:** Instead of clicking the delete button on the table, a user can search for the user based on their email address.

**Related Use Cases:** UCMSv2 – 0004

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0006

**Name :** View old catalogs

**Details :** This use case allows a user with super admin privileges to view older catalogs

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “View Catalogs” button. After clicking on the button the user will be redirected to the catalogs page.
2. On this page the user will be able to view a list of all catalogs in the system.
3. Clicking on a catalog year will open up the catalog information for each major.

**Post condition(s):**

1. The user is able to view an older catalog.

**Alternative Course of Action:** Instead of clicking on a specific years catalog, a user can perform a filtered search to find their desired catalog.

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0007

**Name :** Update DGU Active Catalogs

**Details :** This use case allows a user with super admin privileges to update active catalogs.

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “Active Catalog” button. This will redirect users to a page where the active catalog is displayed.
2. To change

**Post condition(s):**

1. The users has been assigned as an admin.
2. An email is sent out informing advisers that a change has been made to the active catalog.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0007

**Name :** Update DGU Active Catalogs

**Details :** This use case allows a user with super admin privileges to update active catalogs.

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “Active Catalog” button. This will redirect users to a page where the active catalog is displayed.
2. From here the admin can click the “Edit” button to make changes to the active catalog. This will redirect users to the Edit page.
3. From here the admin can look through the catalog and edit based on degree path or department.
4. Once the changes have been made, user can click the “Save” button.

**Post condition(s):**

1. The changes are reflected in the database.
2. An email is sent out informing advisers that a change has been made to the active catalog.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0009

**Name :** Update Course Table

**Details :** This use case allows a user with super admin privileges to update a course table.

**Actors :** Super admin

**Pre-condition(s):**

1. Super admin must be logged in into the system.

**Description :**

1. On the super admins control panel, the user must first click the “Update Course” button. This will redirect users to a page where the active courses are listed.
2. From here the admin can search for a particular course to update.
3. Once the user has found the course to update, clicking on it will display the course information.
4. From here the admin can choose to remove the course or add new information to the description.
5. Once the changes have been made, user can click the “Save” button.

**Post condition(s):**

1. The changes to the course are reflected in the database.
2. An email is sent out informing advisers that a change has been made to a particular course.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0012

**Name :** View Old DGU Catalog.

**Details :** This use case allows Advisors and Admin to view old catalogs from his/her DGU.

**Actors :** Advisor, Admin

**Pre-condition(s):**

1. User 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 “Catalog” button.
2. User selects option “View Old Catalog” from the catalog page.
3. System must display a list of old catalogs.
4. User must select the catalog that he/she wants to see.
5. Use case end when system displays the catalog that user has selected.

**Post condition(s):** None

**Alternative Course of Action:** None.

**Related Use Cases:** UCMSv2 – 0001, UCMSv2 – 0019,

**Constraints:**

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

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

**Use Case ID :** UCMSv2 – 0013

**Name :** View Comments on Catalogs

**Details :** This use case allows a user with admin privileges to view comments on a certain catalog.

**Actors :** Admin

**Pre-condition(s):**

1. Admin must be logged into the system.

**Description :**

1. On the admins control panel, the user can click on the Catalogs button to view a list of all catalogs in the system.
2. From this screen a user can click on any catalog.
3. Once they have selected the catalog that they wish to view they will be directed to the view page for that catalog.
4. Comments can be viewed by scrolling down to the page of each catalog view per DGU.

**Post condition(s):**

1. N/A

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0014

**Name :** Create DGU Accounts

**Details :** This use case allows a user with admin privileges to create DGU accounts.

**Actors :** Admin

**Pre-condition(s):**

1. Admin must be logged into the system.

**Description :**

1. On the admins control panel, the user can click on the shortcut “Create DGU Account” button.
2. On the create accounts page the user would have to enter the following information:
   1. Username, password, email, first name, last name, and birthday.
   2. The account will also have to be assigned to a department.
3. Once the user has entered all of the required information the user can click “Create” to create the account on the system.

**Post condition(s):**

1. The DGU account has been created.
2. An email has been sent out confirming the creation of the account.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0015

**Name :** Delete DGU Accounts

**Details :** This use case allows a user with admin privileges to delete DGU accounts.

**Actors :** Admin

**Pre-condition(s):**

1. Admin must be logged into the system.

**Description :**

1. On the admins control panel, the user can click on the shortcut “Delete DGU Accounts” button.
2. From this screen the user will see a list of all DGU accounts.
3. The user can delete the accounts either by clicking the delete button next to the account. Alternatively user can click on the account to view information about the account and delete if through that screen.
4. Users can also complete a search to find a specific account.

**Post condition(s):**

1. The DGU account has been deleted and reflected in the database.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0016

**Name :** Assign DGU Advisers

**Details :** This use case allows a user with admin privileges to assign DGU advisers.

**Actors :** Admin

**Pre-condition(s):**

1. Admin must be logged into the system.

**Description :**

1. On the admins control panel, the user can click on the shortcut “Assign DGU Advisers” button.
2. Once the user is on the screen they can assign an adviser to a specific program by searching for the user account and the program.
3. Once the two fields have been entered the user can click the “Assign” button to assign that adviser to that DGU.

**Post condition(s):**

1. The user has been assigned.
2. An email has been sent out to the assigned user.

**Alternative Course of Action:** N/A

**Related Use Cases:** N/A

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0018

**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.

**Alternative Course of Action:** None.

**Related Use Cases:** UCMSv2 – 0003, UCMSv2 - 0020

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0019

**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.

**Alternative Course of Action:** None.

**Related Use Cases:** UCMSv2 – 0003, UCMSv2 – 0020

**Constraints:**

* **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.

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

**Use Case ID :** UCMSv2 – 0020

**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

**Alternative Course of Action:**

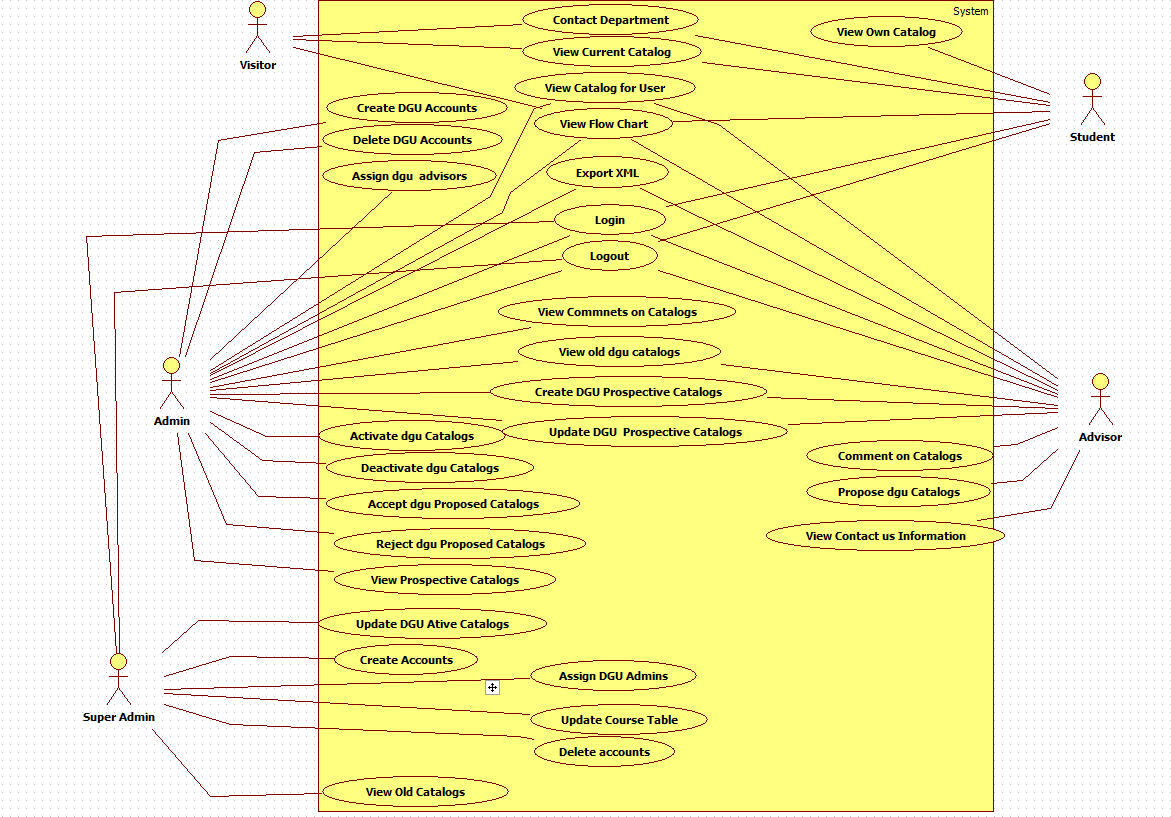
In the case that no prospective catalog has been proposed exist at the moment, system must display message “There are no prospective catalogs to be viewed.”

**Related Use Cases:** UCMSv2 – 0003

**Constraints:**

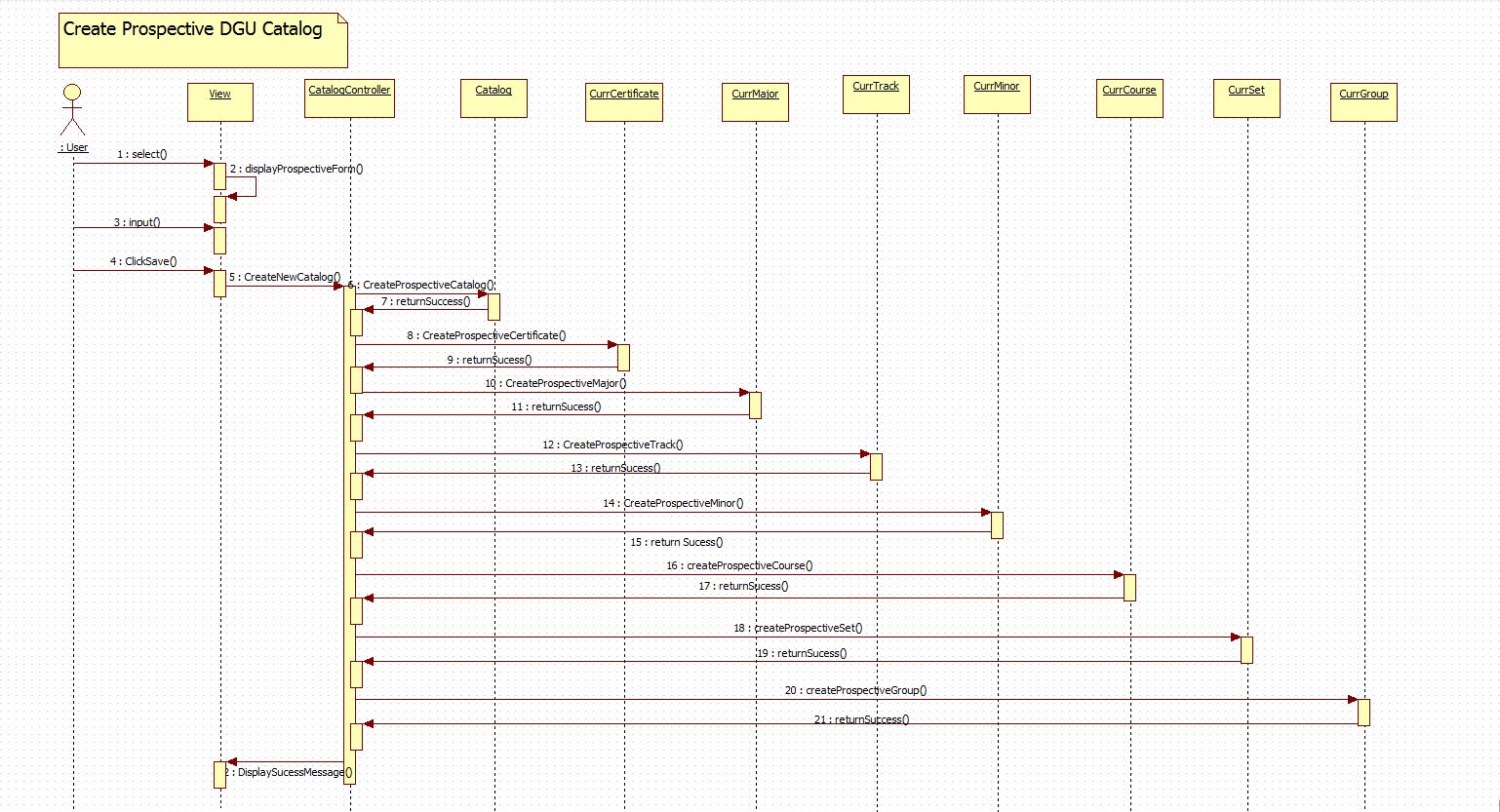
* **Usability**: The feature must be easy to follow.
* **Reliability**: System must work flawlessly 99% of the time.
* **Performance**: Catalog must be displayed save within 5s seconds.
* **Supportability**: This functionality must work for any browser.

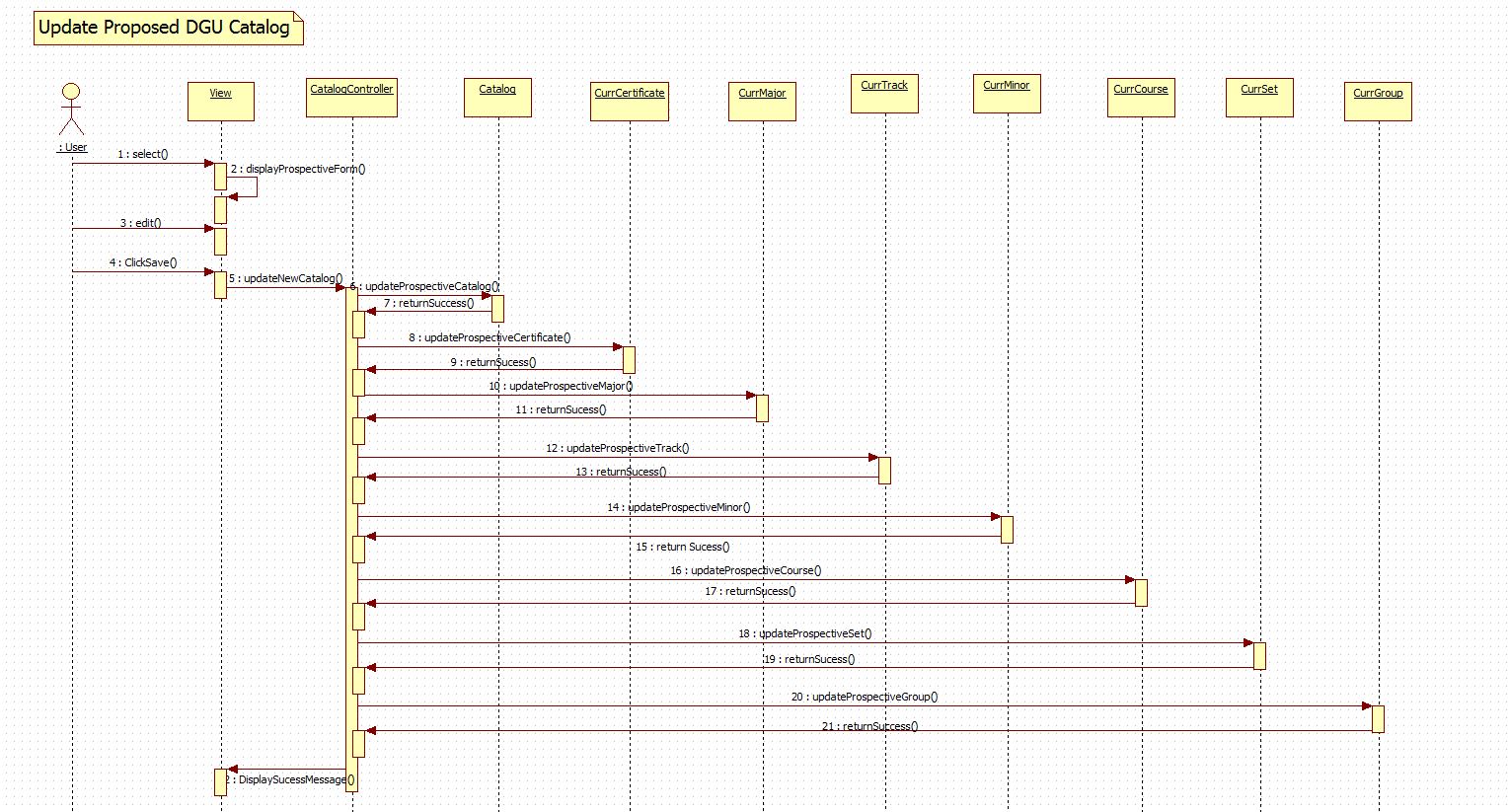
## Appendix B – Use Case Diagram Using UML

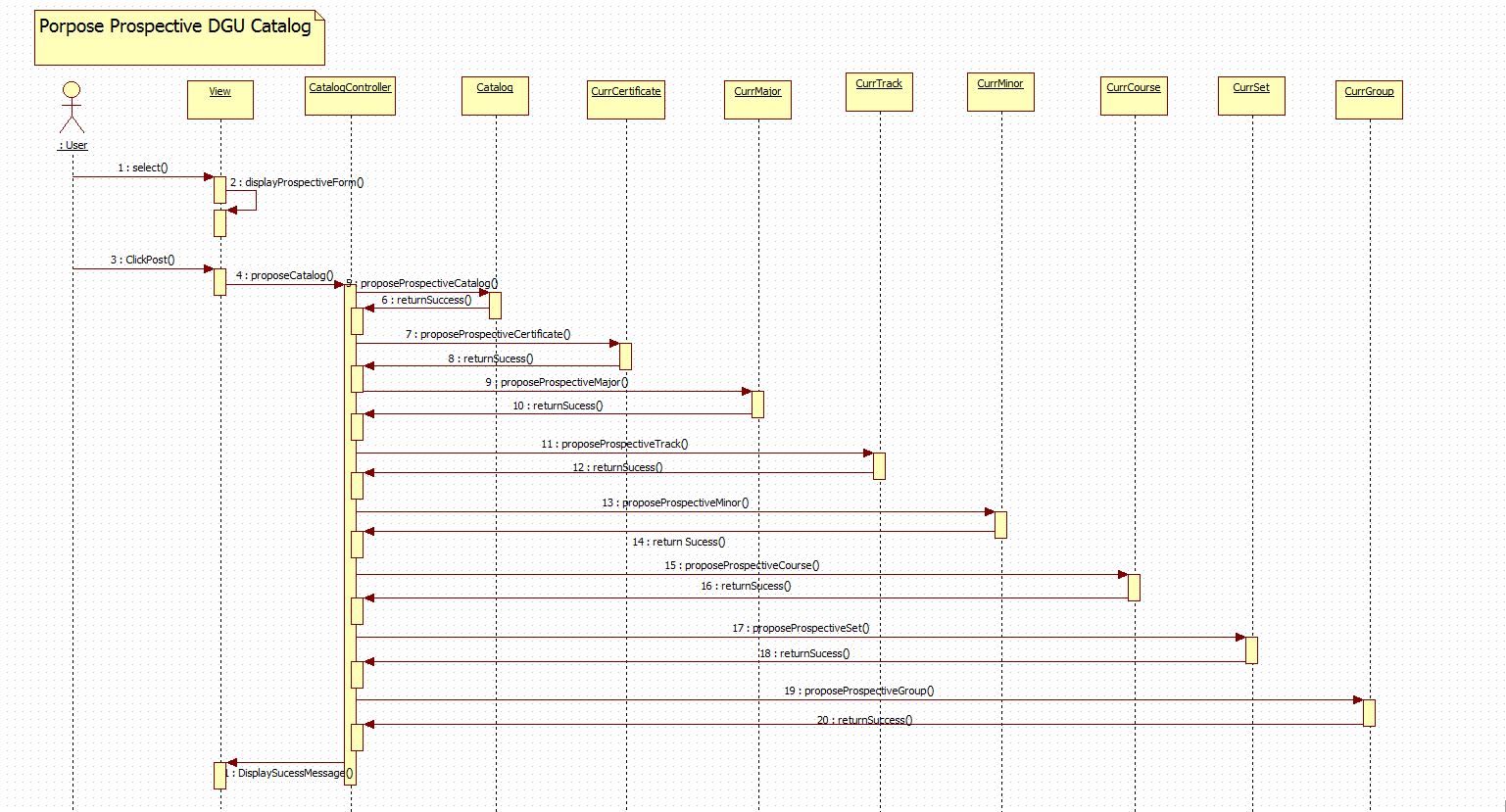


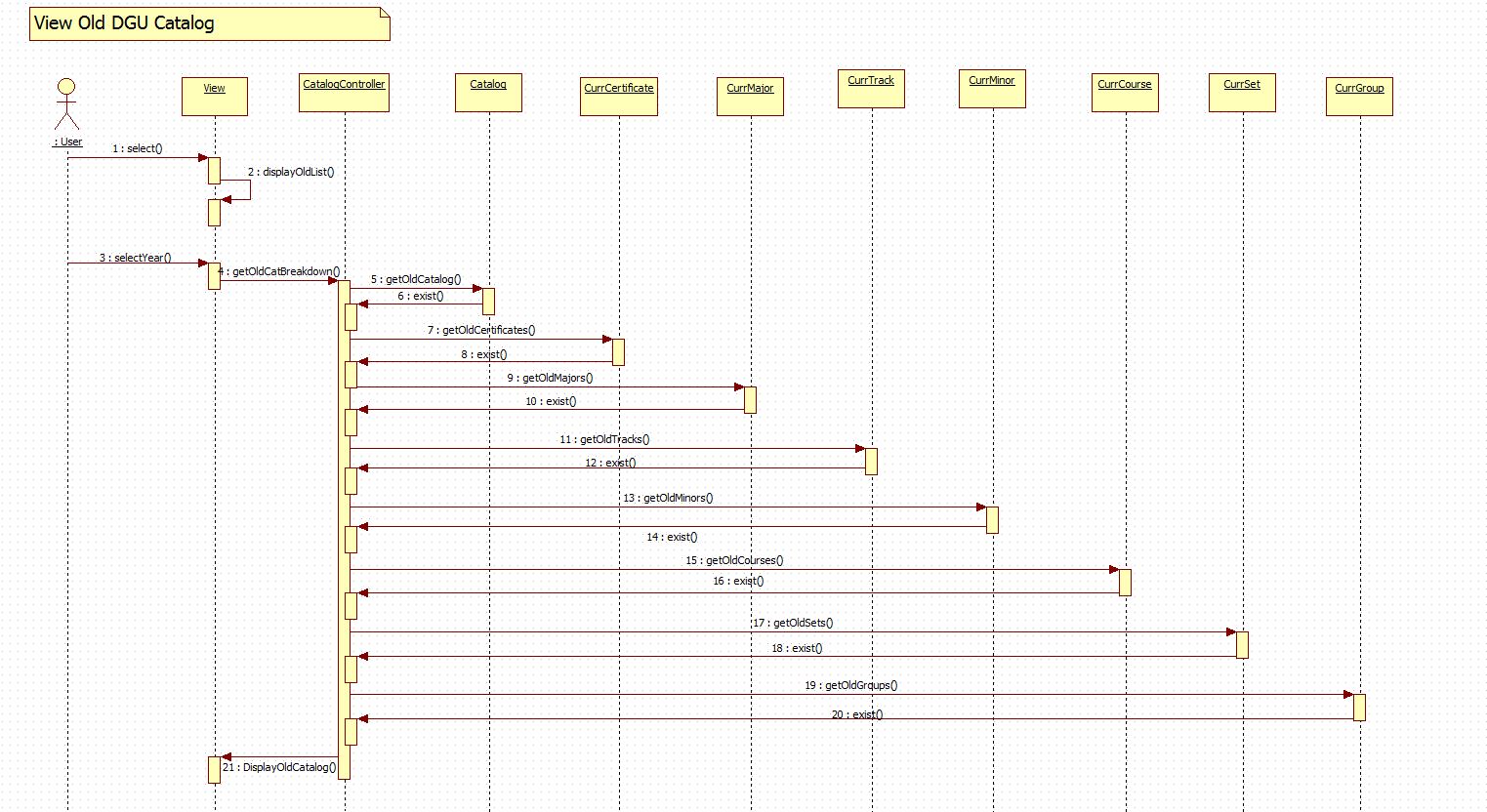
## Appendix C – Static UML Diagram

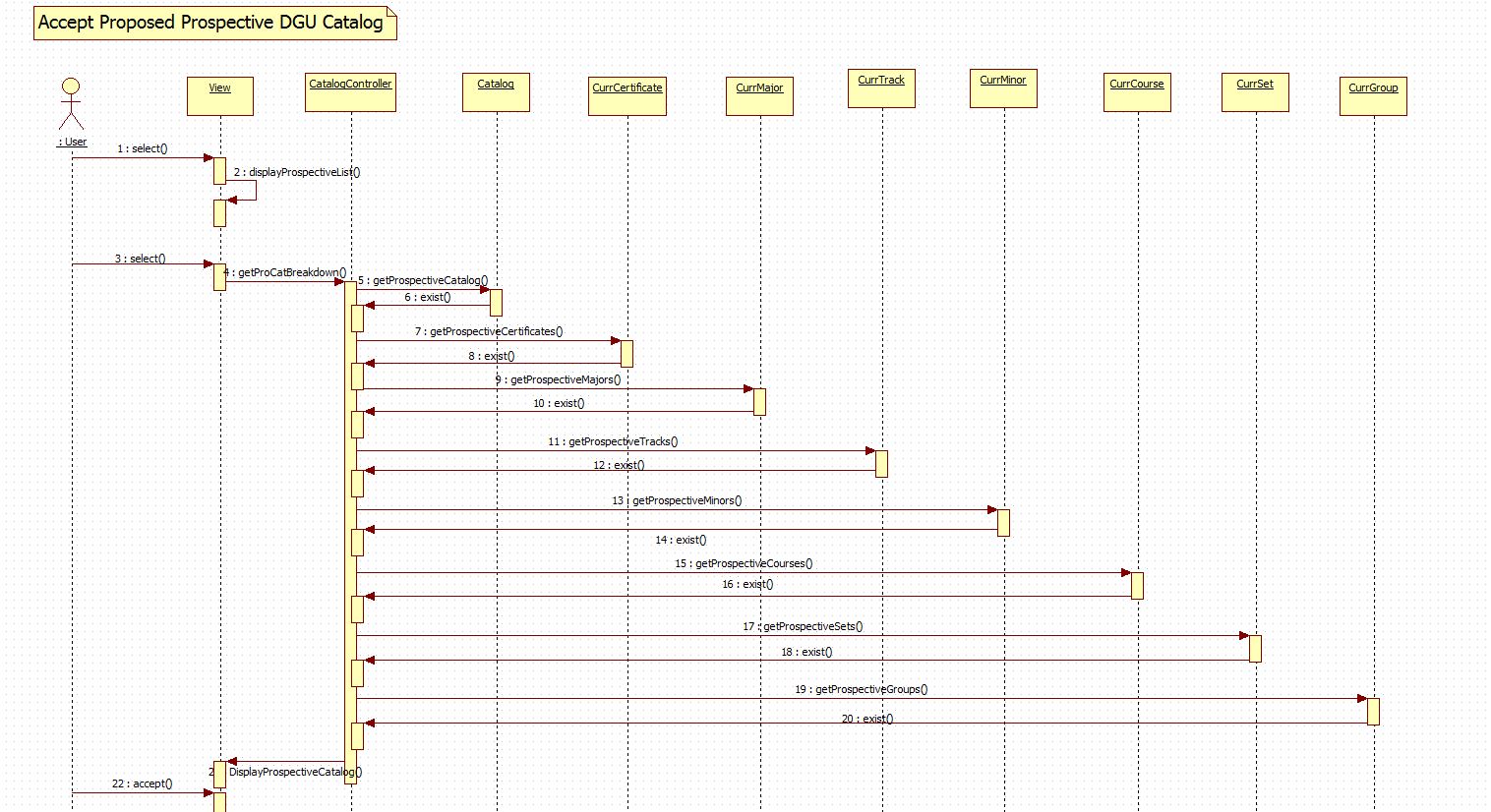
## Appendix D – Dynamic UML Diagram

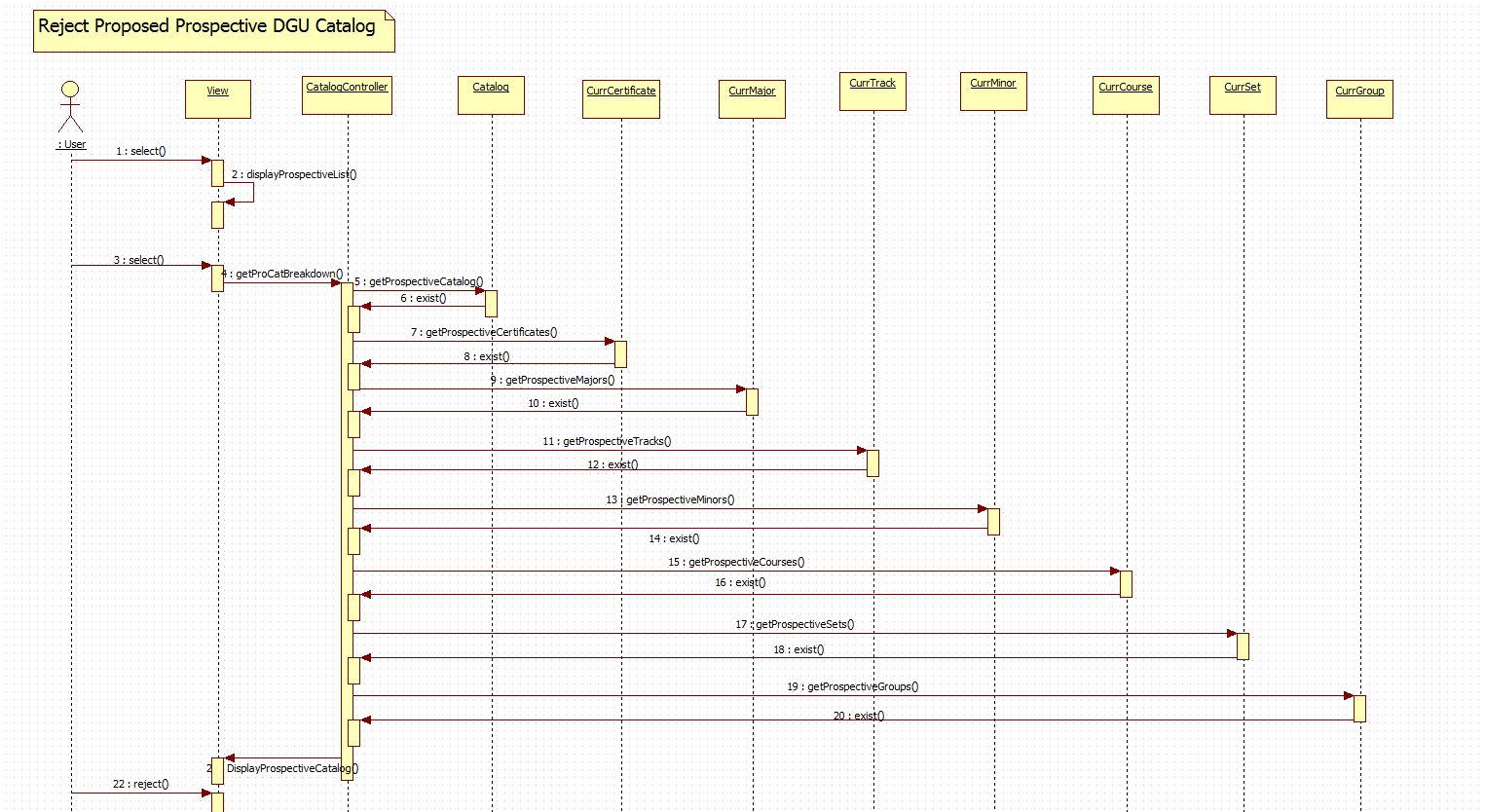


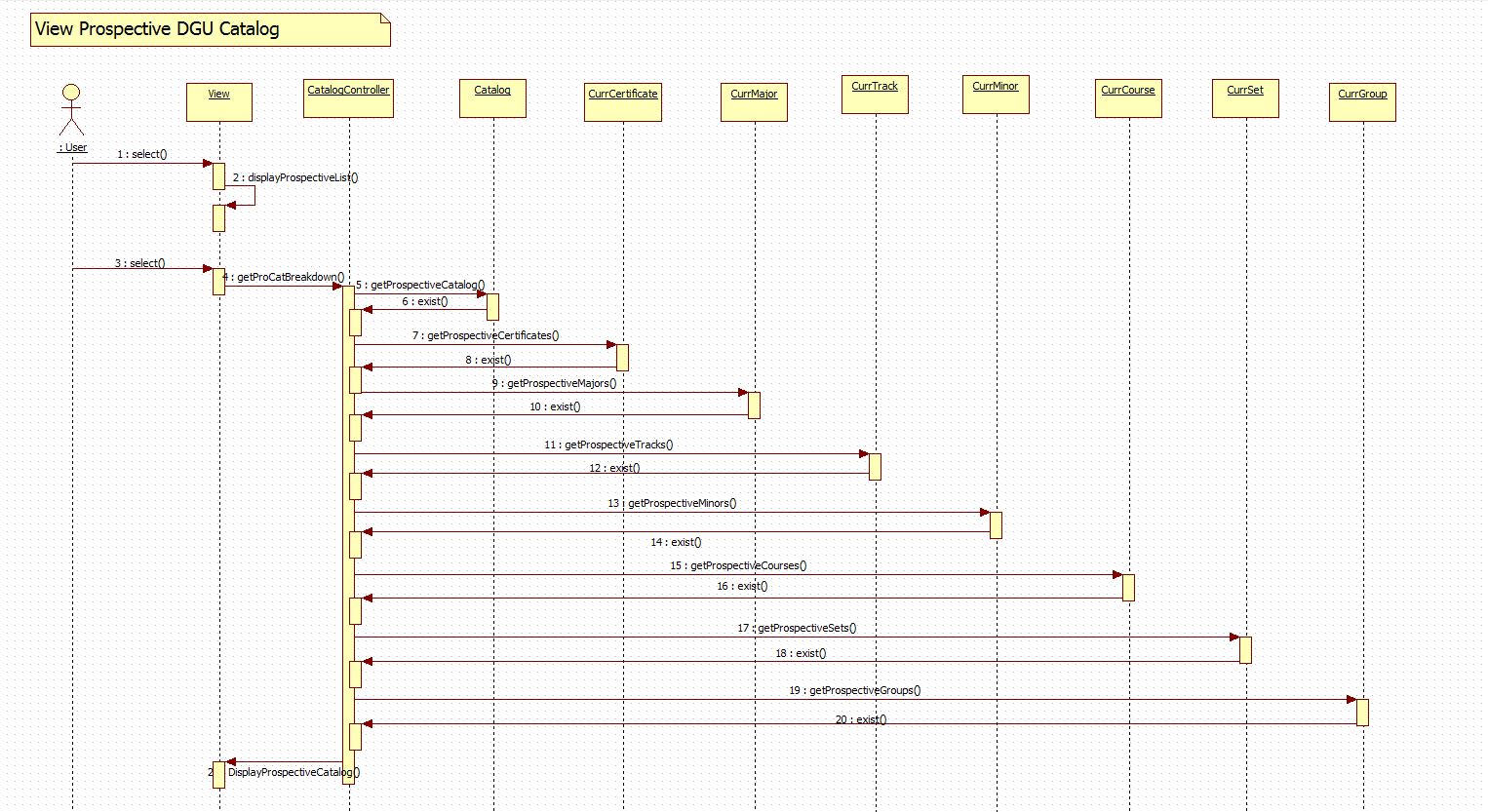












## Appendix E – User Interface Design

## 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