**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, a generation flowchart (in form of a table) has been added in order to aid the path flow of a program in the 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, System Design, and Detail Design. The introduction briefly defines the problem, the software development process used in this project, and terminology. The system design chapter provides information about the system decomposition, hardware and software mapping, persistent data management, and security/privacy. Furthermore, the detail design document has data about the static and dynamic models, and code specification. In addition, this document also contains glossary, appendix, and references.

Contents

[1. Introduction 4](#_Toc404533088)

[1.1. Problem Definition 4](#_Toc404533089)

[1.2. Design Methodology 4](#_Toc404533090)

[1.3. Terminology 4](#_Toc404533091)

[1.3.1. Definitions 4](#_Toc404533092)

[1.3.2. Acronyms 4](#_Toc404533093)

[1.3.3. Abbreviations 5](#_Toc404533094)

[1.4. Overview of the System 5](#_Toc404533095)

[2. System Design 6](#_Toc404533096)

[2.1. Overview 6](#_Toc404533097)

[2.2. Subsystem Decomposition 6](#_Toc404533098)

[2.3. Hardware and Software Mapping 7](#_Toc404533099)

[2.4. Persistent Data Management 8](#_Toc404533100)

[2.5. Security / Privacy 11](#_Toc404533101)

[3. Detail Design 12](#_Toc404533102)

[3.1. Overview 12](#_Toc404533103)

[3.2. Static Model 12](#_Toc404533104)

[3.3. Dynamic Model 12](#_Toc404533105)

[3.4. Code Specification 12](#_Toc404533106)

[4. Glossary 13](#_Toc404533107)

[4.1. Definitions 13](#_Toc404533108)

[4.2. Acronyms 13](#_Toc404533109)

[4.3. Abbreviations 13](#_Toc404533110)

[5. Appendix 14](#_Toc404533111)

[5.1. Appendix A – User Case Diagram (Use Cases being implemented) 14](#_Toc404533112)

[5.2. Appendix B – User Cases being implemented 14](#_Toc404533113)

[5.3. Appendix C – Documented Class Interfaces 14](#_Toc404533114)

[5.4. Appendix D – Diary of meetings and tasks. 15](#_Toc404533115)

[6. References 23](#_Toc404533116)

## Introduction

This chapter introduces the University Catalog Management System version 2.0 including the problem that this project is solving, the design methodology used in this project and the terminology that is used in this document.

## 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.

In addition, the creation of new content for the catalog must be managed as well. Presently, this content can be added without passing any previous revision, which can really take a wrong direction in no previous check in exist. Finally, the creation of path flows in a program are currently being generated manually, which is time consuming because it needs to be done for all programs in the catalog university.

## Design Methodology

The Catalog Management System has been developed using the Agile Software Development process. This process provides an adaptive planning, and evolutionary development of the system, which is broken down through different iterations. Furthermore, continuous improvement can be easily achieved, and changes can be adapted without making major changes within the system.

The main goal of this methodology is to facilitate early releases with minimal bugs at the end of iteration. In other words, the system can be developed until each functionality is implemented as the customer wants it; even when multiple changes have been required down the path of development.

## 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

This document is organized as follow:

Chapter 2 contains information about how the system is decomposed into subsystems, how software and hardware is mapped, how the data management is being achieved as well as its security and privacy.

Chapter 3 describes the design of the entire system. In addition, details of how the system work is shown by using static and dynamic models. Finally, it provides code specification, which describes attributes and methods being used as well as their constraints.

Chapter 4 gives a glossary of terms that were used throughout this document.

Chapter 5 is the appendix of the document.

Chapter 6 contains the references of information used in this document if any.

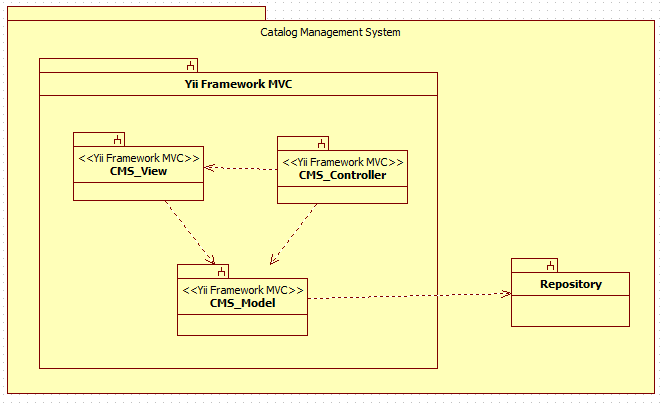
## System Design

This chapter gives a high level description of the system design for the Catalog Management System. It starts by giving and overview of the system by describing the architectures used for this project. Then it explains how the system is decomposed in the different subsystems. Furthermore, it describes how hardware and software are mapped. It also addresses how data is management occurs. Finally, it explains how security and privacy is implemented.

## Overview

The Catalog Management System has been developed based on two architectures. The first one is Model-View-Controller (MVC) which works as the primary architectures. This architecture separates the representation of data (view), the logic of the system (controller), and the data itself (model).

The other is the Data Repository pattern. This pattern is used since all data is stored in a central database created for this project.



## Subsystem Decomposition

The Catalog Management System consists of four major subsystems, three inside Yii Framework MVC subsystem, and the other is the Repository subsystem.

* **CMS\_View**

This subsystem has all forms that are presented to the user, so that he/she can interact with the system. In addition, this subsystem modifies what is being displayed base on the type of user that is currently interacting with the system. In this subsystem, the user can enter the requests that he/she needs to perform. Once the user has made a request the CMS\_View subsystem interacts with the CMS\_Controller to process this request.

* **CMS\_Controller**

This subsystem has the logic of the system. Most of the data that is needed to be either displayed or stored goes through here. Furthermore, the CMS\_Controller subsystem handles all commands received by the users in order to process their requests. Based on the request this controller will interact with the CMS\_Model subsystem to either retrieve or store information in the database.

* **CMS\_Model**

This subsystem provides a way to interact with the data so that can be store or retrieve from the repository. The CMS\_Model subsystem receives commands from the CMS\_Controller subsystem so that it can get or send information to the repository which contains all data for the system.

* **Repository**

This is the database where all information of the Catalog Management System is stored. Here is stored data for the users, catalogs, majors, minors, certificates, groups, set, and courses, among others. This subsystem is in direct contact with the CMS\_Model subsystem. Once the CMS\_Model sends a request the repository performs the action on the data.

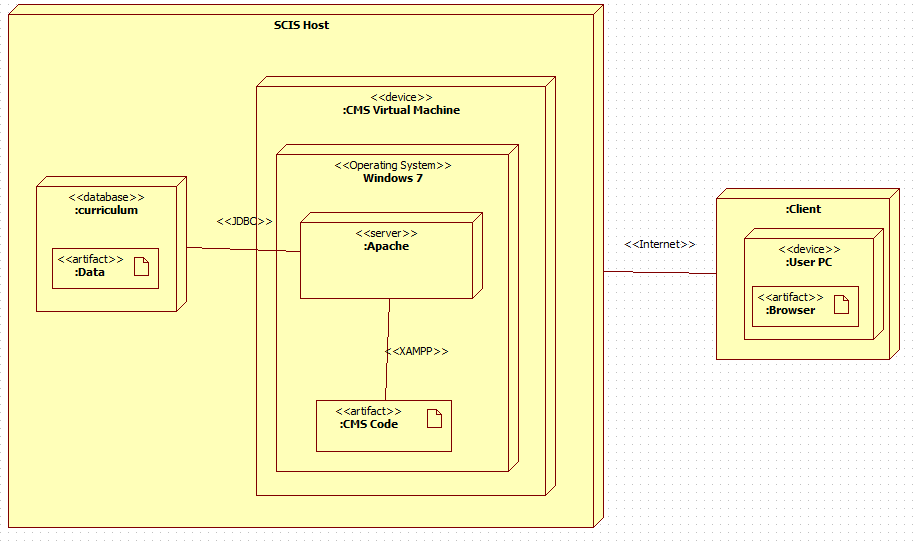
Each subsystem is being used in all use cases that have been implemented on the Catalog Management System project.

## Hardware and Software Mapping

The Catalog Management System is hosted on a virtual machine provided by the School of Computing and Information Sciences. This VM runs on Windows 7. Furthermore, it is important to mention that the database “curriculum” is hosted by the SCIS as well. In addition the most current version of the code for the system is stored in the VM so that it can be accessed through the internet. In order to be accessible, the Apache server must be running all the time on the virtual machine. This server connects the code for the system with the database by using XAMPP.

On the other hand, the user can access the system using any PC that has an internet browser.

In order to better understanding of the hardware and software mapping for the Catalog Management System, a deployment diagram is provided below.



## 

## Persistent Data Management

**User Information Storage**

The system requires storage of the information of the user that interacts with the system. Among the information about that is stored from the user is the last time he/she logged in into the system, the username, password, time when the user was created, the type of user he/she is, and whether this user is currently active or not.

**Catalog Storage**

The system requires storage of the different catalog that have been activated in the past, the ones that are proposed, the ones that are prospective, as well as the one that is currently active.

**Current Storage**

The system requires storage of the current data. This current data refers to the name, catalog id number, as well as the id for the majors, minors, certificates, tracks, groups, sets, and courses that are contained in the different catalogs of the system.

**History Storage**

The system requires storage of the current data. This history data refers to the description, identifier id number, which is the link to the current data, minimum number of credits, major number of credits, and in some cases notes for the majors, minors, certificates, tracks, groups, sets, and courses that are contained in the different catalogs of the system.

**Relation Storage**

The system requires storage of the relation that exists between majors and tracks, tracks and groups, minors and groups, certificates and groups, groups and sets, and finally sets and groups. This storage has as main purpose to save the relation between the groups mentioned above.







## Security / Privacy

**User authentication process**

The Catalog Management System has three main users: Administrator, Advisor, and regular visitor or student.

In the case of administrators and advisors, they need to log in with their credentials. On the main webpage of the system, users need to look for the log in button which is located at the end of the action bar (blue colored). Once the log in page is visible, user must provide their credentials, and the click in login, just pressed enter. The system will identify whether the user is administrator or advisor type, and provide them with their respective access to functionalities they can perform.

In the case of regular visitors or student, they are no required to log in to interact with the system. However, they do have access to a minim number of functionalities only.

**Encryption of data**

Currently, for data encryption, the CMS is using functions provided by Yii - framework. The “encrypting” function take the password the user inputs and converts it to a certain type of string that is composed of different characters, symbols, and digits. At the end the password that is stored in the database is complete changed to what the user entered.

**Use of firewalls and security servers**

The information held in the “curriculum” database is currently hosted by the School of Computing and Information Sciences (SCIS). This database can only be accessed when connecting directly to the “fiu-cis” network.

The latest version of the system is currently hosted by SCIS as well. This code is in a virtual machine that is directly connected to the “curriculum” database. This connection allows that users accessing to the system through the VM can access data stored in the main database.

## Detail Design

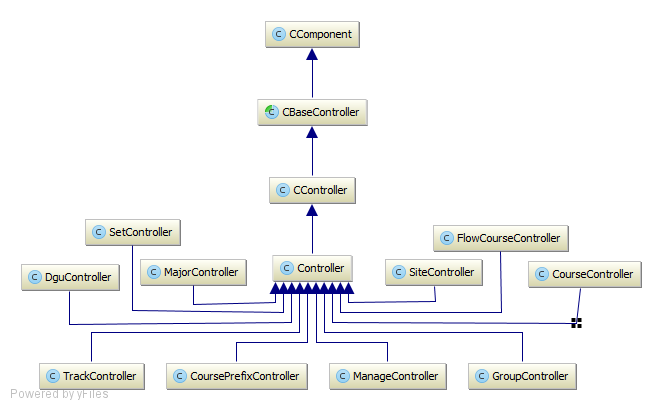
The Detail Design chapter gives a better understanding of the static and dynamic models for the Catalog Management System. Section 3.1 describes the behavior and structure of each subsystem. Section 3.2 shows the different classes along with their methods, and attributes that are part of the system. Furthermore, a minimal class diagrams is provided to help understanding the relationship between classes. Moreover, section 3.3 contains the dynamic model of the system. Here, we can find the sequence diagrams for the use cases that are being implemented for the current version of the project. Main algorithms that have been used are included as well in this section. Finally, section 3.4 describes the class interfaces as well as the constraints for the main control object in each system.

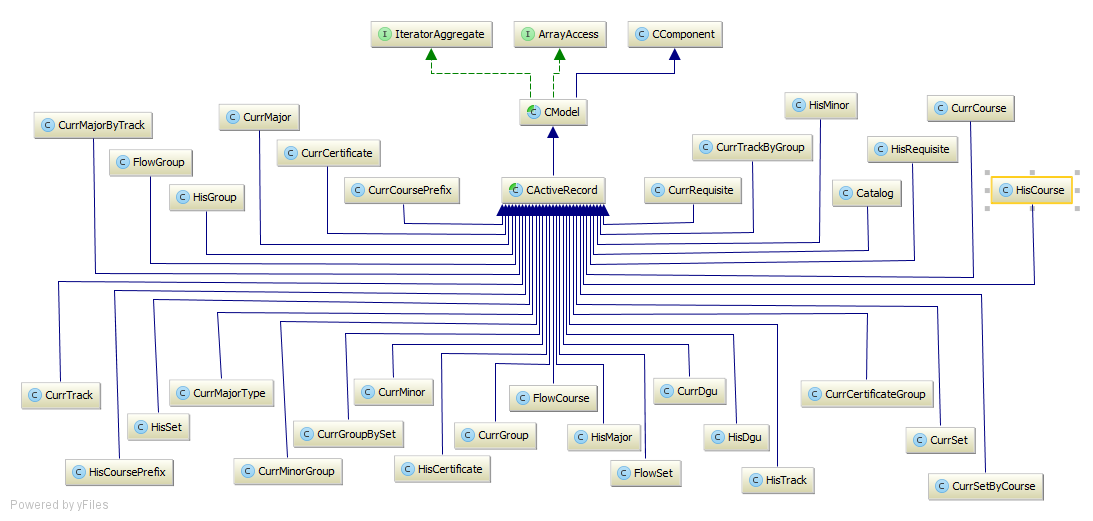
## Overview

The Catalog Management System is basically composed of four subsystems, three of them fall inside the Yii MVC, and the other is the repository. This first subsystem is the controller which is in charge of controlling and processing all command within the system. Without this subsystem, the CMS would be just a bunch of forms and death data. The second subsystem is the view, which allows the user to interact with system. This subsystem makes possible that user can see and enter data to the repository. The third subsystem is the model, which allows the system to interact with the data stored in the database. This means to retrieve or send data to be stored. Finally, the last subsystem is the repository, which has the data for the system.

## Static Model

Here are provided minimal class diagrams for the CMS\_Controller subsystem as well as the CMS\_Model subsystem. Detailed class diagrams have not been provided in this document due to low visibility of methods, and fields.





## Dynamic Model

state machine diagram for the main control object in each subsystem. Include the design of the ***main algorithms*** used in the problem solution. Refinement of the sequence diagram from the analysis model. Place diagrams inline

## Code Specification

describe the class interfaces (attributes and method signatures) and constraint (invariants, pre-condition and post-conditions) for the main control object in each system. Code should be in Appendix C

## 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 – User Case Diagram (Use Cases being implemented)

## Appendix B – User Cases being implemented

## Appendix C – Documented Class Interfaces

Documented class interfaces (code) for the subsystem(s) you will implement and the constraints.

## Appendix D – Diary of meetings 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: |

## References