|  |
| --- |
| Designer drug database |
| Design Document |
| Senior Project, CIS 4911- U01 |
| **Professor: Seyedmasoud Sadjadi Mentor: Dr. Luis Arroyo** |
| **Team Member: Carlos Dominguez** |
| **2/2/2015** |

|  |
| --- |
|  |

# Copyrights and Trademarks Notices

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

All rights reserved. No part of the Designer Drug Database System Project or its documentation may be reproduced or transmitted in any form or by any means without prior written consent of the DDD Team.

# Abstract

The Designer Drug Database project is designed to provide a way to manage electronically information gathered by researchers at the Forensic Research Institute of FIU as well as allow them to share important information with the world of Academia and the Professionals Community. The system is sophisticated and highly user friendly in this aspect.

This document contains information about the design of the entire 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.

# Table of Contents

Table of Contents

[Copyrights and Trademarks Notices 1](#_Toc418130623)

[Abstract 2](#_Toc418130624)

[Table of Contents 3](#_Toc418130625)

[1. Introduction 5](#_Toc418130626)

[1.1 Problem definition. 5](#_Toc418130627)

[1.2 Design methodology used 5](#_Toc418130628)

[1.3 Definitions, acronyms, and abbreviations. 5](#_Toc418130629)

[1.3.1 Definitions 5](#_Toc418130630)

[1.3.2 Acronyms and abbreviations 6](#_Toc418130631)

[1.4 Overview of document 6](#_Toc418130632)

[2. System Design 7](#_Toc418130633)

[2.1 Overview 7](#_Toc418130634)

[2.2 Subsystem Decomposition 7](#_Toc418130635)

[2.3 Hardware and Software Mapping. 8](#_Toc418130636)

[2.4 Persistent Data Management. 8](#_Toc418130637)

[2.5 Security/Privacy 9](#_Toc418130638)

[3. Detailed Design 10](#_Toc418130639)

[3.1 Overview 10](#_Toc418130640)

[3.2 Static model 10](#_Toc418130641)

[3.3 Dynamic model 11](#_Toc418130642)

[3.4 Code Specification 11](#_Toc418130643)

[4. Glossary 12](#_Toc418130644)

[6. 1.3.1 Definitions 12](#_Toc418130645)

[12. 1.3.2 Acronyms and abbreviations 12](#_Toc418130646)

[22. Appendix 13](#_Toc418130647)

[5.1 Appendix A - Use case diagram for use cases being implemented. 13](#_Toc418130648)

[5.2 Appendix B - Use cases being implemented (from the RD). 14](#_Toc418130649)

[5.3 Appendix C 23](#_Toc418130650)

[5.4 Appendix D - Diary of meeting and tasks. 25](#_Toc418130651)

[Sprint 1 25](#_Toc418130652)

[Sprint 2 26](#_Toc418130653)

[Sprint 3 26](#_Toc418130654)

[Sprint 4 27](#_Toc418130655)

[Sprint 5 27](#_Toc418130656)

[6. References 28](#_Toc418130657)

# Introduction

The Designer Drug Database System is a system designed to easily add new compounds and manage the information through a centralized user friendly web application. This application was done with a responsive web design to be friendly in phones, tablets and any other device that you access the web application from.

## 1.1 Problem definition.

Dr. Arroyo and the team at the Forensic Research Institute produce several reports and useful information from research they perform at the Institute. One of the challenges they have face is sharing the results that yields from their research. The purpose of this project will be to provide the Institute with a software solution that allows them to share the research result with the world of Academia and the Professionals Community

## 1.2 Design methodology used

The Designer Drug Database 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. Using this methodology, I have used the different models such as sequence diagrams, and class diagrams.

## 1.3 Definitions, acronyms, and abbreviations.

Following we will have the definitions. Acronyms and abbreviations that are used in this document.

### 1.3.1 Definitions

**Single-page application (SPA):** is a web application or web site that fits on a single web page with the goal of providing a more fluid user experience akin to a desktop application.

**Platform independent framework:** is a framework that allows programmers to create one application that can be seen over different platforms.

**Native system:** is a system that can only be seen in its own target devices, such as apple devices or android devices.

**Administrator:** Registered user with credentials. User has elevated privileges and can give access to other users.

**User:** Any general person who uses the system

### 1.3.2 Acronyms and abbreviations

**Admin:** Administrator

**CSS :** Cascading CIS Style Sheets

**DB:** Database

**FIU:** Florida International University

**GUI:** Graphical User Interface

**HTML:** Hypertext Markup Language

**JS:** JavaScript

**MVC:** Model View Controller

**PHP:** PHP: Hypertext Preprocessor

## 1.4 Overview of document

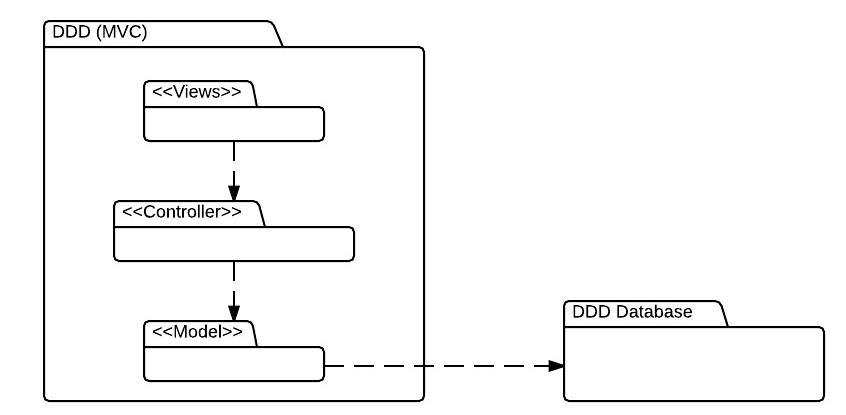
This document contains information about the design of the entire project. The following chapters on this document are System Design, and Detail Design. The system design chapter provides information about the system decomposition, hardware and software mapping, persistent data management, and security/privacy. 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.

# System Design

This chapter gives a high level description of the system design for the Designer Drug Database 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.

## 2.1 Overview

The Designer Drug Database has been developed based Model-View-Controller (MVC). This architecture separates the representation of data (view), the logic of the system (controller), and the data itself (model).



## 2.2 Subsystem Decomposition

* **DDD View**

This subsystem has all forms that are presented to the user. Thus the user can only interact directly with this subsystem. This subsystem varies its presentation according to who is interacting with it, none of the three users of the system have the same options. Once the user make a request through this subsystem then the request is passed to the Controller subsystem.

* **DDD 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. When a request is received from the view module, it is analyzed and depending on the request it goes or not to the model.

* **DDD Model**

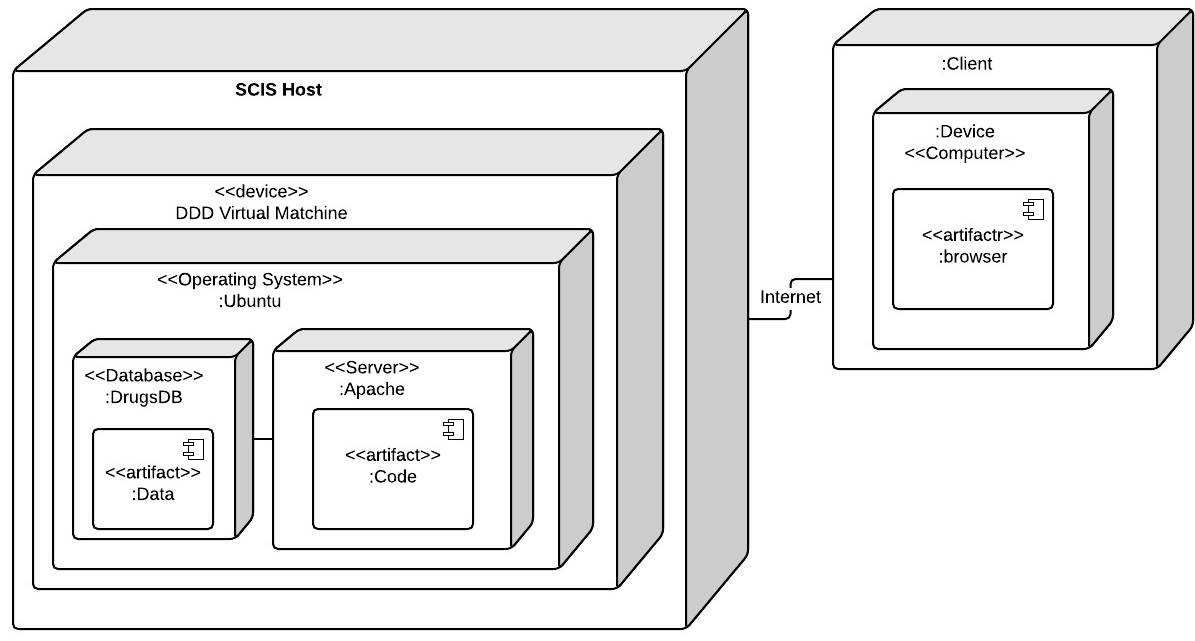
This subsystem provides a way to interact with the data so that it can be stored or retrieved from the repository. It receives request from Controller and performs it by requesting to the database.

* **DDD Database**

This is the module where all the Information is store

## 2.3 Hardware and Software Mapping.

The hardware and software mapping as the following figure shows, consist of two hardware devices and all the software running on them. On the device that is accessing the single page application the app is fun and any data request is sent to the webserver, who responds with data, then the requested device process the information and display it.



## 2.4 Persistent Data Management.

The persistent data to be stored will be stored in databases. The following relational tables are the way in which the data is to be stored.

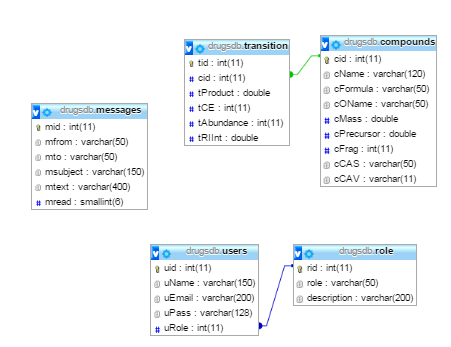
* **User Information:**

The system needs to save the user information of those who have the right to modify different modules of the system. In this case the system will save the name, email and password of admins and lab operators.

* **Compound Information**

Compound information is also saved in the database. In this case although the data being stored has changed from sprint to sprint, the current information being stored is Compound name, 1. Other Names, Formula, Mass , Precursor, Product, No. of Transitions, Frag, CE, Abundance, Relative Ion Intensity , CAS, Cayman #

In general the database look like the following



## 2.5 Security/Privacy

User authentication will use the sessions as a way to maintain persistence between all the pages and as a way to authenticate that a user has the right to access to certain information.

Encryption of data is only used for password and the type of encryption in use will be MD5.

Moreover, PDO with prepare statements have been use to prevent SQL Injections.

# Detailed Design

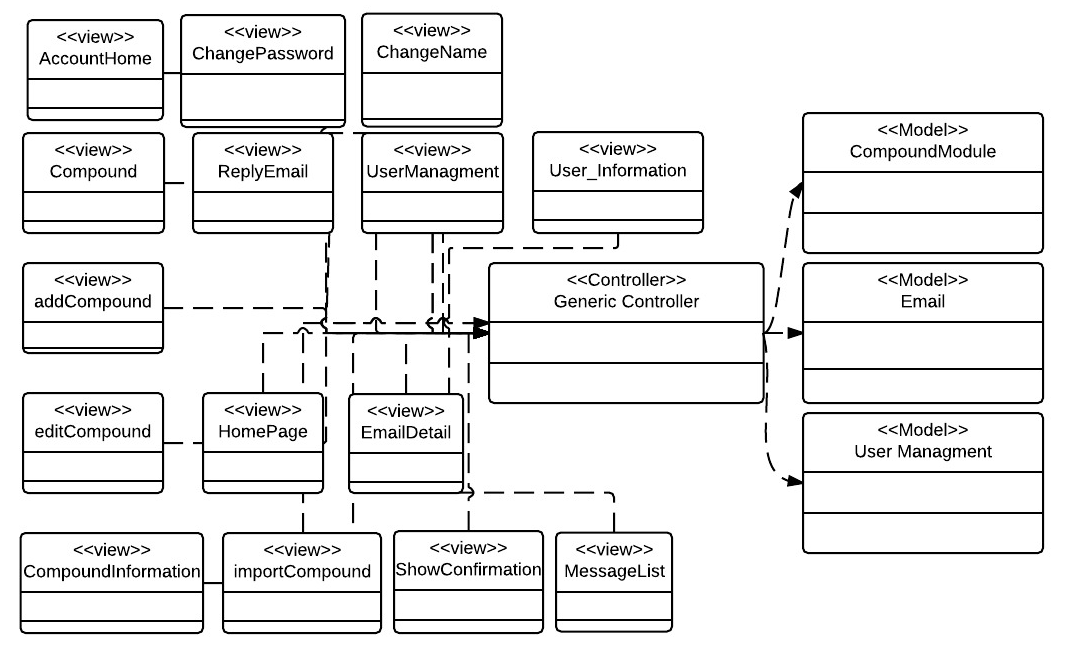
The Detail Design chapter gives a better understanding of the static and dynamic models for the Designer Drug Database System. The document describes the behavior and structure of each subsystem, the different classes along with their methods, and attributes that are part of the system. In addition a minimal class diagrams is provided to help understanding the relationship between classes, and 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, the document describes the class interfaces as well as the constraints for the main control object in each system.

## Overview

The Designer drugs database is basically composed of four subsystems, three of them fall inside the DDD 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 DDD 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.

## 3.2 Static model

Here are provided minimal class diagrams. Detailed class diagrams have not been placed here. However, detail class diagrams can be found in the Appendix.



## 3.3 Dynamic model

The Sequence diagram although not included here, can be found in appendix.

## 3.4 Code Specification

There are several things that need to be specified.

1. Client Code and how was it organized
2. Server Code and how is it organized

Client Code:

For the client Code there are several pages that need to be visited:

**/index.php**: contains the main page with the view that is filled by every page

**/functions/function.js**: contains the generic controller that serves to a controller to every view and model. It also contains the route that ANGULARJS uses to dynamic reroute different pages. Also it contains the Controller used by the navigation bar and lastly it contains some global javascript variables

**/pages/….html**: This folder pages contains all the pages or views of the system. The pages that contains a JavaScript function called cntrl which are the portion that belongs to the generic controller for each specific view or page.

Server Code:

For server code you have to look in the folder/request/. This folder contains all php server processing pages (Model). In this case I have:

CompoundModuleClass.php and CompoundModuleCall.php: where all compound information is processed at the model level

UserModuleClass.php, UserModuleCall.php: where you get all user information at the model level

AuthModuleClass.php and AuthModuleCall.php: where all the authentication is done.

MsgModuleClass.php and MsgModuleCall.php: where all messages are processed at model level.

# Glossary

1. Following we will have the definitions. Acronyms and abbreviations that are used in this document.

### 1.3.1 Definitions

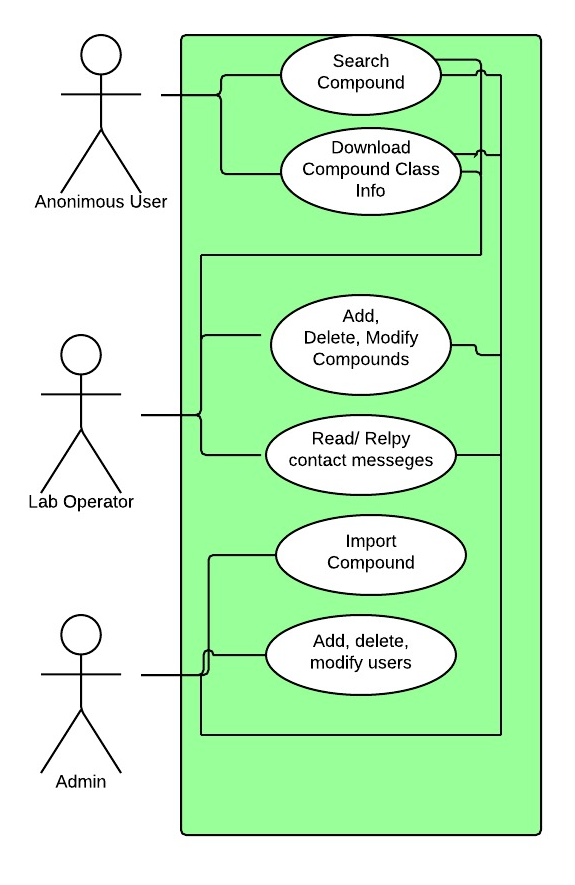
1. **Single-page application (SPA):** is a web application or web site that fits on a single web page with the goal of providing a more fluid user experience akin to a desktop application.
2. **Platform independent framework:** is a framework that allows programmers to create one application that can be seen over different platforms.
3. **Native system:** is a system that can only be seen in its own target devices, such as apple devices or android devices.
4. **Administrator:** Registered user with credentials. User has elevated privileges and can give access to other users.
5. **User:** Any general person who uses the system

### 1.3.2 Acronyms and abbreviations

1. **Admin:** Administrator
2. **CSS :** Cascading CIS Style Sheets
3. **DB:** Database
4. **FIU:** Florida International University
5. **GUI:** Graphical User Interface
6. **HTML:** Hypertext Markup Language
7. **JS:** JavaScript
8. **MVC:** Model View Controller
9. **PHP:** PHP: Hypertext Preprocessor

# Appendix

## 5.1 Appendix A - Use case diagram for use cases being implemented.



## 5.2 Appendix B - Use cases being implemented (from the RD).

|  |
| --- |
| Name: User Login |
| Participating actor: a non-logged in user |
| Entry condition:   * User is in login page. * User is not logged in. |
| Exit condition:   * User is on the account main page |
| Event flow:   1. User clicks the Login button 2. System loads login page 3. User enters email and password on their respective boxes and click submit button 4. System loads the account main page |
| Alternative Exit condition: System says that login and password are incorrect. |

|  |
| --- |
| Name: Modify Name of User |
| Participating actor: a logged in user. |
| Entry condition:   * User is logged in. * User is in Account Page |
| Exit condition:   * Confirmation name of user is changed is shown. |
| Event flow:   1. User clicks on its name 2. System shows a view to change the name of the user 3. User replaces old name with new name and clicks submit 4. System changes the name of the user and shows a confirmation. |

|  |
| --- |
| Name: Modify user password |
| Participating actor: a logged in user. |
| Entry condition:   * User is logged in. * User is in Account Page |
| Exit condition:   * Confirmation that password has been changed |
| Event flow:   1. User clicks on the password link 2. System shows a view to change the password 3. User enters old, new and confirmation of new password. 4. System changes the password of the user and shows a confirmation. |

|  |
| --- |
| Name: Search For Compound |
| Participating actor: any user of the system |
| Entry condition:   * User is in main page or home page |
| Exit condition:   * User found the data of the search compound |
| Event flow:   1. User enters the email or formula of compound in the search criteria box and click the search button. 2. Systems show the information of the compound |
| Alternative Exit condition: System does not find a compound and show a message saying the search was unsuccessful. |

|  |
| --- |
| Name: Download Compound Class |
| Participating actor: any user of the system |
| Entry condition:   * User is in main page or home page |
| Exit condition:   * User saved to his computer the desired data |
| Event flow:   1. User searches for the class of the compound he desires to download in the class download list and clicks on it 2. A window is pop up to ask user where he wants to save the information 3. User pick location where information will be saved and click save 4. System downloads the file |

|  |
| --- |
| Name: add new Compound |
| Participating actor: either an admin or a labOP |
| Entry condition:   * User is logged in the system * User is in compound page. |
| Exit condition: Compound data is added in the system |
| Event flow:   1. User click on the add compound button 2. System shows add compound form page 3. User selects the picture of the compound, fill the fields: name, formula, other name, mass, fragments, retention time, precursor, CAS. User add the corresponding transitions and click submit all button. 4. System confirm that compound has been saved |

|  |
| --- |
| Name: delete Compound |
| Participating actor: either an admin or a labOP |
| Entry condition:   * User is logged in the system * User is in the compound page. |
| Exit condition: Compound data is deleted from the system |
| Event flow:   1. User enters the name or formula in the search box and clicks the search button. 2. System searches for compound and goes to modify compound view 3. User clicks delete compound button. 4. System deletes the compound and shows a confirmation |

|  |
| --- |
| Name: edit Compound |
| Participating actor: either an admin or a labOP |
| Entry condition:   * User is logged in the system * User is in their account home page. |
| Exit condition: Compound data is updated in the system |
| Event flow:   1. User enters the name or formula in the search box and clicks the search button. 2. System searches for compound and goes to modify compound view. 3. User edit any of the information on the modify compound view and click the update button. 4. System updates the compound and shows a confirmation |

|  |
| --- |
| Name: import Compound |
| Participating actor: either an admin or a labOP |
| Entry condition:   * User is logged in the system * User is in the compound page. * File to be imported has a csv extension and has the following rows: Compound Name, Compound Class, Formula, Mass, Precursor, Product, No. of Transitions, Frag, CE, Abundance, Relative Ion Intensity, RT (Zorbax), CAS |
| Exit condition: Compounds data in csv file and pictures has been added to the system |
| Event flow:   1. User click in the import compound button 2. System shows the import page 3. User clicks the choose file to choose location of csv file from import csv panel 4. A pop up menu will show for user to choose the file he wishes to import. 5. User selects the csv file and clicks submit button. 6. System saves the compounds information and shows a confirmation. 7. User click the choose file to pick the location of the folder where the compound pictures are at, from import picture panel. 8. A pop up menu will show for user to choose the folder where pictures of compounds are 9. User selects all the pictures to be upload and clicks the submit button 10. System upload all the pictures associated with the compounds uploaded previously. |

|  |
| --- |
| Name: Add New User |
| Participating actor: admin user |
| Entry condition:   * User is logged in the system * User is in the user management page. |
| Exit condition: User invite is sent and confirmation is shown |
| Event flow:   1. User enters the email of the user to be invited and clicks the invite button. 2. System sends an email to the user and shows a confirmation. |

|  |
| --- |
| Name: edit User Rights |
| Participating actor: an admin |
| Entry condition:   * User is logged in the system. * User is in the user management page. * User is not trying to changes rights to itself. |
| Exit condition: User access right is updated |
| Event flow:   1. User enters the email of the user to be searched and clicks the search button. 2. System shows a view with the searched user information. 3. User picks the downgrade or upgrade option 4. System updates user rights and show confirmation. |

|  |
| --- |
| Name: delete User |
| Participating actor: an admin |
| Entry condition:   * User is logged in the system * User is in the user Management page. * user is not trying to delete itself |
| Exit condition: User is deleted |
| Event flow:   1. User enters the email of the user to be searched and clicks the search button. 2. System shows a view with the user information. 3. User clicks the delete button 4. System deletes the user and shows confirmation. |

|  |
| --- |
| Name: Add New Compound Class |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the compound class page. |
| Exit condition: Compound Class is added and confirmation is shown |
| Event flow:   1. User Clicks compound classes button. 2. System shows the Compound Classes view 3. User clicks button add class 4. System shows view to add new class 5. User enters the class name and clicks submit. 6. System saves the new class and shows confirmation. |

|  |
| --- |
| Name: Edit Compound Class Name |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the compound class page. |
| Exit condition: Compound Class name is updated and confirmation is shown |
| Event flow:   1. User Clicks compound classes button. 2. System shows the Compound Classes view 3. User clicks on the class that is going to be updated 4. System shows view where the class name can be updated 5. User updates the class name and clicks submit. 6. System updates the class name and shows confirmation. |

|  |
| --- |
| Name: delete Compound Class |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the compound class page. * Compound class to be deleted has no compounds associated with it |
| Exit condition: Compound Class is deleted and confirmation is shown |
| Event flow:   1. User Clicks compound classes button. 2. System shows the Compound Classes view 3. User clicks on the class that is going to be deleted. 4. System shows view with the class name to be deleted 5. User clicks the delete button. 6. System deletes the class and shows confirmation. |

|  |
| --- |
| Name: Add Read contact email |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the Messages page. |
| Exit condition: Message content is shown to the user |
| Event flow:   1. User Clicks on a message to be read from list of messages 2. System shows the message content. |

|  |
| --- |
| Name: reply contact message |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the Messages page. |
| Exit condition: Message reply is sent and confirmation is shown. |
| Event flow:   1. User Clicks on a message to be replied from list of messages 2. System shows the message content. 3. User clicks on the reply button 4. System shows a view to reply the message 5. User enters the text to be replied and click reply button 6. System replies the message and show a confirmation |

|  |
| --- |
| Name: delete message |
| Participating actor: admin or LabOP user |
| Entry condition:   * User is logged in the system * User is in the Messages page. |
| Exit condition: Message content is shown to the user |
| Event flow:   1. User Clicks on a message to be deleted from list of messages 2. System shows the message content. 3. User clicks the delete button 4. System deletes the message and shows confirmation message. |

## 5.3 Appendix C

The following are the classes that are needed for the Designer Drug Database system.

CompoundModuleClass.php

- compRequest()

- getComp()

- getCompbyId()

- addComp()

- editComp()

- deleteComp()

- impPics()

- impComp()

- getTrans()

- addTrans()

- editTrans()

- deleteTrans()

- countComp()

- checkVariableNotEmpty(valueOfVariable, nameOfVariable)

- checkEmailIsLoggedUser()

- checkisAdmin()

- checkisAdminOrLabOP()

- checkEmailIsNotLoggedUser()

- requestDatabase(returnValue)

- returnJson(result)

- returnCSV(result,title)

UserModuleClass.php

- userRequest()

- findUser()

- upgradeUser()

- downgradeUser()

- deleteUser()

- addUser()

- getMyInfo()

- setMyName()

- setMyPass()

- signUP()

- countUsers()

- checkVariableNotEmpty(valueOfVariable, nameOfVariable)

- checkEmailIsLoggedUser()

- checkisAdmin()

- checkisAdminOrLabOP()

- checkEmailIsNotLoggedUser()

- requestDatabase(returnValue)

- returnJson(result)

AuthModuleClass.php

- authRequest()

- testLoginInfo(email,password)

- testSesionInfo()

- checkUserLoggedIn()

- logOut()

- checkVariableNotEmpty(valueOfVariable, nameOfVariable)

- requestDatabase()

- returnJson(result)

MsgModuleClass.php

- userRequest()

- deleteMsg()

- newMsg()

- newReply()

- getMsg()

- modMsg()

- countMsg()

- checkVariableNotEmpty(valueOfVariable, nameOfVariable)

- requestDatabase()

- returnJson(result)

## Appendix D - Diary of meeting and tasks.

### Sprint 1

|  |  |
| --- | --- |
| Diary Entry 1 | |
| Date | Monday, February 2rd, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Talk about the system that needs to be done. * Create system user stories. |
| Summary | * Explanation of the current system. * Brief definition of the functionalities to be implemented. * Created some user stories based on the explanation |
| Assigned Tasks | * Get familiar with the tools to be used. * Create a mock up for acceptance. |

|  |  |
| --- | --- |
| Diary Entry 2 | |
| Date | Friday, February 6th, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show the mock up to where it was. * Show authentication system. |
| Summary | * Showed the mocked up to where it was done. * Talked about authentication system and showed what I had done. |
| Assigned Tasks | * Continue working on first Sprint |

|  |  |
| --- | --- |
| Diary Entry 3 | |
| Date | Friday, February 13th, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show what was done on Sprint 1 * Assigned for User Stories for Sprint 2 |
| Summary | * Showed the mocked up. * Show all that was done during Sprint 1. * New user story were assigned |
| Assigned Tasks | * Get ready to start spring 2 next Monday |

### Sprint 2

|  |  |
| --- | --- |
| Diary Entry 4 | |
| Date | Friday, February 27th, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show what was done in Sprint 2. * Assign User Stories to Sprint 3 |
| Summary | * Showed what was done in sprint 2. * Dr. Arroyo assigned user stories for sprint 3 |
| Assigned Tasks | * Get ready to start spring 3 next Monday |

### Sprint 3

|  |  |
| --- | --- |
| Diary Entry 5 | |
| Date | Friday, March 20th, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show what was done on Sprint 3 * Assigned for User Stories for Sprint 4 |
| Summary | * Show all that was done during Sprint 3. * New user story were assigned for Sprint 4 * Dr. Arroyo pointed out that different compounds could have the same formula, yet provide different information. |
| Assigned Tasks | * Get ready to start spring 4 next Monday * Fix what needed to be fixed on Sprint 4 regathing the new clarification on compounds formulas. |

### Sprint 4

|  |  |
| --- | --- |
| Diary Entry 5 | |
| Date | Friday, April 3rd, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show what was done on Sprint 4 * Assigned for User Stories for Sprint 5 |
| Summary | * Show all that was done during Sprint 4. * New user story were assigned for Sprint 5 * Dr. Arroyo said he wanted to change some of the information the institute is going to be sharing. |
| Assigned Tasks | * Get ready to start spring 5 next Monday * Fix the information that is to be shared on Sprint 5. |

### Sprint 5

|  |  |
| --- | --- |
| Diary Entry 5 | |
| Date | Friday, April 17th, 2015 |
| Location | OE building |
| Start | 1:00 PM |
| End | 2:00 PM |
| Attendees | * Carlos Dominguez * Dr. Luis Arroyo |
| Agenda | * Show what was done on Sprint 4 * Assigned for User Stories for Sprint 5 |
| Summary | * We discussed the agenda.. He was happy with the end result |
| Assigned Tasks | * No more tasks assigned. |

# 6. References