|  |  |
| --- | --- |
|  | BYU-Idaho CS364  Grad Planner  Registration Integration  Software Design Document  Version 5.0  07/22/2020 |

# Authors

**Project Sponsor**

Jeff Lyon

**Project Manager**

Steven Milton

**Team Leaders**

|  |  |
| --- | --- |
| Group 1 | Nathan Stratford |
| Group 2 | Rick Edgemon |
| Group 3 | Govert Carreño |
| Group 4 | Jason Todd Jenkins |

**Individual Contributor**

|  |
| --- |
| **Group 1** |
| Fraser LeFevre  John Bake  Blake Pearson  Michael Nishiguchi |
| **Group 2** |
| Dalan Ienatsch  Ben Crowe  Josh Spendlove  Adam Nielsen |
| **Group 3** |
| Oscar Ampudia Lazo  Ryan Tennant  Andres Castro Guevara  Jacob Muhlestein |
| **Group 4** |
| Cristian Chuychuy  Ethan Picklesimer  Carlos Iribar  Jerrald Nelson |

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason for Changes | Version |
| Initial | 6/13/2020 | First Draft | 1.0 |
| Draft | 6/20/2020 | Second Draft | 1.1 |
| Draft | 7/4/2020 | Third Draft | 2.0 |
| Draft | 7/11/2020 | Fourth Draft | 3.0 |
| Draft | 7/19/2020 | Fifth Draft | 4.0 |
| Final Report | 7/22/2020 | Final changes to the report | 5.0 |

# Table of Contents

[1. Introduction 1](#_Toc46337636)

[1.1 Purpose 1](#_Toc46337637)

[2. Document Target Audience 1](#_Toc46337638)

[2.1 Developer 1](#_Toc46337639)

[2.1.1 Design Concerns 1](#_Toc46337640)

[2.2 Tester 2](#_Toc46337641)

[2.2.1 Design Concerns 2](#_Toc46337642)

[3. Scope 3](#_Toc46337643)

[4. Context 3](#_Toc46337644)

[5. References 5](#_Toc46337645)

[6. Glossary 5](#_Toc46337646)

[7. Requirements Traceability Matrix 8](#_Toc46337647)

[8. Use Cases 15](#_Toc46337648)

[8.1 Identified Stakeholders and Design Concerns 15](#_Toc46337649)

[8.1.1 BYUI Student 15](#_Toc46337650)

[8.1.2 Design Concerns 16](#_Toc46337651)

[8.2 Use Case Tables and Descriptions 18](#_Toc46337652)

[8.2.1 Use Case Template 18](#_Toc46337653)

[8.2.2 Degree Audit (REQ- 4) 19](#_Toc46337654)

[8.2.3 Class availability List (REQ- 5, 6, 42, 43, 44, 47) 22](#_Toc46337655)

[8.2.4 Class time (REQ- 9, 10) 23](#_Toc46337656)

[8.2.5 Class Status (REQ- 65) 25](#_Toc46337657)

[8.2.6 Registration status (REQ- 66) 27](#_Toc46337658)

[8.2.7 Course description (REQ- 15, 16, 49, 50, 51, 52) 30](#_Toc46337659)

[8.2.8 Required Classes (41) 32](#_Toc46337660)

[8.2.9 Automatic Registration (REQ- 48) 34](#_Toc46337661)

[8.2.10 Filter Data (REQ- 64) 35](#_Toc46337662)

[8.2.11 Opening Display Classes (REQ- 69) 37](#_Toc46337663)

[8.2.12 Export Schedule (REQ- 17) 39](#_Toc46337664)

[8.2.13 Menu Index (REQ- 2) 41](#_Toc46337665)

[8.2.14 Update Schedule (REQ – 25, 24) 43](#_Toc46337666)

[8.2.15 Weekly Schedule (REQ – 8, 12, 13, 14) 45](#_Toc46337667)

[8.2.16 REST-ful Web Service (REQ- 3, 53) 47](#_Toc46337668)

[8.2.17 Check Holds (REQ - 45) 49](#_Toc46337669)

[8.2.18 Enable or Disable Automatic Registration (REQ 46) 51](#_Toc46337670)

[8.2.19 Display Feedback (REQ – 26, 27, 28) 54](#_Toc46337671)

[8.2.20 Provide Feedback (REQ – 29, 30, 31, 32, 33) 57](#_Toc46337672)

[8.2.21 Degree (REQ-18, 19) 58](#_Toc46337673)

[8.2.22 Restrictions (REQ- 11, 67, 68) 60](#_Toc46337674)

[8.2.23 Login (REQ- 1, 7) 62](#_Toc46337675)

[8.2.24 Semester (REQ – 20) 64](#_Toc46337676)

[8.2.25 Grad Path (REQ - 21, 22) 66](#_Toc46337677)

[8.2.26 Possible Grad Classes (REQ - 23) 69](#_Toc46337678)

[8.2.27 User Interface (REQ - 34, 38, 54) 71](#_Toc46337679)

[8.2.28 New/Returning User (REQ - 35, 36, 37) 72](#_Toc46337680)

[8.2.29 Grad Planner Tour Button (REQ - 36) 74](#_Toc46337681)

[8.2.30 Grad Planner Tour (REQ - 37) 76](#_Toc46337682)

[8.2.31 Response Time (REQ - 39, 40) 78](#_Toc46337683)

[8.2.32 User Traffic (REQ – 55, 56, 57) 79](#_Toc46337684)

[8.2.33 Cookies (REQ - 58, 59) 81](#_Toc46337685)

[8.2.34 Secure Data Transfer (REQ - 60, 61, 62, 63) 82](#_Toc46337686)

[9. Design Overview 84](#_Toc46337687)

[9.1 System Architecture 84](#_Toc46337688)

[9.1.1 Architecture description 85](#_Toc46337689)

[9.1.2 View Classes 85](#_Toc46337690)

[9.1.3 Controller Classes 97](#_Toc46337691)

[9.1.4 Model Classes 97](#_Toc46337692)

[9.2 System Interfaces 100](#_Toc46337693)

[9.2.1 User Interface 100](#_Toc46337694)

[9.2.2 Software Interfaces 100](#_Toc46337695)

[9.3 Constraints and Assumptions 100](#_Toc46337696)

[9.3.1 Constraints 100](#_Toc46337697)

[9.3.2 Assumptions 101](#_Toc46337698)

[9.4 Error Handling 102](#_Toc46337699)

[9.4.1 Database Facade Class Commit Errors 102](#_Toc46337700)

[9.4.2 Invalid Data Errors 102](#_Toc46337701)

[9.4.3 I-Learn Importing Errors 102](#_Toc46337702)

[9.4.4 UI Controller Class Errors 102](#_Toc46337703)

[9.5 Component Design 103](#_Toc46337704)

[9.5.1 Degree Audit (REQ- 4) 103](#_Toc46337705)

[9.5.2 Class availability List (REQ- 5, 6, 42, 43, 44, 47) 104](#_Toc46337706)

[9.5.3 Class time (REQ- 9, 10) 106](#_Toc46337707)

[9.5.4 Class Status (REQ- 65) 107](#_Toc46337708)

[9.5.5 Registration status (REQ- 66) 108](#_Toc46337709)

[9.5.6 Course description (REQ- 15, 16, 49, 50, 51, 52) 108](#_Toc46337710)

[9.5.7 Required Classes (REQ- 41) 109](#_Toc46337711)

[9.5.8 Automatic Registration (REQ- 48) 111](#_Toc46337712)

[9.5.9 Filter Data (REQ- 64) 112](#_Toc46337713)

[9.5.10 Opening Display Classes (REQ- 69) 114](#_Toc46337714)

[9.5.11 Export Schedule (REQ- 17) 116](#_Toc46337715)

[9.5.12 Menu Index (REQ- 2) 117](#_Toc46337716)

[9.5.13 Update Schedule (REQ – 25, 24) 118](#_Toc46337717)

[9.5.14 Weekly Schedule (REQ – 8, 12, 13, 14) 120](#_Toc46337718)

[9.5.15 RESTful Web Service (REQ- 3, 53) (team 02) 120](#_Toc46337719)

[9.5.16 Check Holds (REQ - 45) (team 02) 121](#_Toc46337720)

[9.5.17 Enable or Disable automatic registration (REQ 46) 122](#_Toc46337721)

[9.5.18 Display Course Feedback (REQ – 26, 27, 28) (Team 2) 124](#_Toc46337722)

[9.5.19 Provide Feedback (REQ – 29, 30, 31, 32, 33) (Team 4) 125](#_Toc46337723)

[9.5.20 Degree (REQ-18, 19) 125](#_Toc46337724)

[9.5.21 Class Registration (REQ- 11, 67, 68) 128](#_Toc46337725)

[9.5.22 Login (REQ- 1, 7) (team 04) 130](#_Toc46337726)

[9.5.23 Semester (REQ – 20) (team 04) 130](#_Toc46337727)

[9.5.24 Grad Path (REQ - 21, 22) 131](#_Toc46337728)

[9.5.25 Possible Grad Classes (REQ - 23) (team 04) 133](#_Toc46337729)

[9.5.26 User Interface (REQ - 34, 38, 54) (team 04) 133](#_Toc46337730)

[9.5.27 New/Returning User (REQ - 35, 36, 37) (team 04) 133](#_Toc46337731)

[9.5.28 Response Time (REQ - 39, 40) (team 04) 133](#_Toc46337732)

[9.5.29 User Traffic (REQ - 55, 56, 57) (team 04) 133](#_Toc46337733)

[9.5.30 Cookies (REQ - 58, 59) (team 04) 133](#_Toc46337734)

[9.5.31 Secure Data Transfer (REQ - 60, 61, 62, 63) (team 04) 133](#_Toc46337735)

[10. Data Design 133](#_Toc46337736)

[10.1 Data Description 133](#_Toc46337737)

[10.2 Entity Relationship Diagram 133](#_Toc46337738)

[10.3 Data Dictionary 135](#_Toc46337739)

[10.4 SVG 136](#_Toc46337740)

[SVG Grad Planner logo 136](#_Toc46337741)

# List of Figures

Figure 1 Projected User Base Chart .....................................................................................................................................................11

Figure 2 Requirements and Courses screenshot ...................................................................................................................................26

Figure 3 Failed to register class screenshot ..........................................................................................................................................29

Figure 4 Class status screenshot ...........................................................................................................................................................31

Figure 5 Registration status screenshot …............................................................................................................................................33

Figure 6 Course description screenshot ...............................................................................................................................................35

Figure 7 Required classes screenshot ..................................................................................................................................................37

Figure 8 Automatic registration screenshot .........................................................................................................................................38

Figure 9 Filter data screenshot .............................................................................................................................................................42

Figure 10 Class first-time sign-up screenshot ......................................................................................................................................44

Figure 11 Export schedule screenshot .................................................................................................................................................46

Figure 12 Menu index mockup ............................................................................................................................................................48

Figure 13 Update schedule screenshot .................................................................................................................................................50

Figure 14 Weekly schedule screenshot ................................................................................................................................................52

Figure 15 Check hold screenshot ….....................................................................................................................................................62

Figure 16 Enable/disable automatic registration screenshot ................................................................................................................64

Figure 17 Course Feedback screenshot ................................................................................................................................................66

Figure 18 Degree requirements screenshot ..........................................................................................................................................71

Figure 19 Class restrictions error message screenshot .........................................................................................................................73

Figure 20 Grad planner login screenshot .............................................................................................................................................75

Figure 21 Grad path screenshot …........................................................................................................................................................79

Figure 22 Possible classes screenshot ..................................................................................................................................................81

Figure 23 Grad Planner tour launch button mockup ….........................................................................................................................87

Figure 24 Grad planner tour mockup …...............................................................................................................................................89

Figure 25 System Architecture Diagram .............................................................................................................................................96

Figure 26 Degree Audit Diagram ...…................................................................................................................................................115

Figure 27 Class Availability Diagram ................................................................................................................................................116

Figure 28 Login Relationship Diagram .............................................................................................................................................125

Figure 30 Export Schedule Diagram ..................................................................................................................................................128

Figure 31 Menu Index Diagram ….....................................................................................................................................................129

Figure 32 Update Schedule Diagram .................................................................................................................................................130

Figure 33 Weekly Schedule Diagram ................................................................................................................................................131

Figure 34 RESTful Web Services Diagram .......................................................................................................................................132

Figure 35 Check Holds Diagram .…..................................................................................................................................................133

Figure 36 Degree Path Diagram ..…..................................................................................................................................................139

Figure 37 User Controller Diagram ...................................................................................................................................................140

Figure 38 Class Registration Diagram ...............................................................................................................................................142

Figure 39 User Interface Diagram …..................................................................................................................................................143

Figure 40 Grad Path Diagram …...….................................................................................................................................................145

Figure 41 Entity Relationship Diagram .............................................................................................................................................147

# Introduction

## Purpose

The purpose of this Software Design Document is to define the design elements of the Grad Planner Registration Integration. The document uses standards-based on the “IEEE Standards for Information Technology - Software Design Descriptions” [1]. The document utilizes requirements from the “Software Requirements Specification” [2] document and provides sufficient detail to implement the software product to stakeholder specifications. The documentation will describe the system architecture, the user interface, and the way that the system is to be implemented to meet design concerns.

# Document Target Audience

## Developer

Developers are those involved in the development process of the Grad Planner Registration Integration application.

### **Design Concerns**

#### Developers will know of the expected inputs and outputs.

#### Developers will implement error handling and data validation in their code.

#### Developers will know what patterns and classes to be used.

#### Developers will know how the data is stored in the database.

## Tester

Testers are those that make test cases and assist in testing the Grad Planner Registration Integration application.

### **2.2.1 Design Concerns**

#### Testers will be given a set of the expected results based on the inputs.

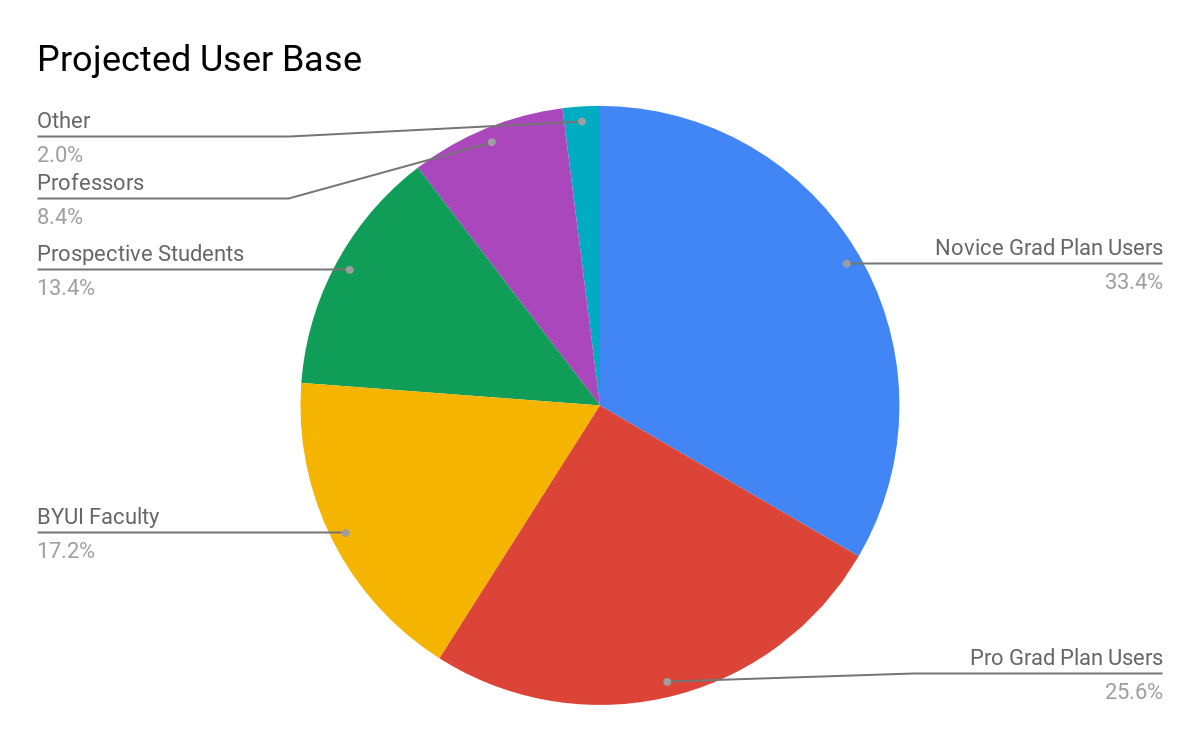
#### Testers should understand the processes associated to test all aspects of the product.

# Scope

This document supplies details for all features required in the initial release of the Grad Planner Registration Integration application. Specifically, internal data models, user interface, overall design views, and relevant use cases are defined.

# Context

The Grad Planner Registration Integration is an application to ease the course scheduling for college students at BYU-Idaho. The application will communicate via an API with the BYU-I Registration system currently in use by the school. The application will be accessed via mobile devices and computers on a web interface. The purpose is to assist students with an understanding of certificate or degree requirements and giving them the tools to schedule those courses over the student’s career at BYU-I.



***Figure 1 - Projected User Base Chart***

# References

1. IEEE, "IEEE Standard for Information Technology — Systems Design — Software Design Descriptions" pp. 1-35, Jul. 2009. [Online] Available: <http://ieeexplore.ieee.org/servlet/opac?punumber=5167253>
2. Grad Planner Registration Integration Software Requirements Specification
   * Version 1.4, Last Updated June 11, 2020

[3] “Encyclopedia Britannica” [Online] Available: <https://www.britannica.com/technology/graphical-user-interface>

[4] “Protecting Student Privacy” US Department of Education [Online] Available: <https://studentprivacy.ed.gov/faq/what-ferpa>

[5] Lexico Online Dictionary, [Online] Available: <https://www.lexico.com/en/definition/adviser>

# Glossary

|  |  |
| --- | --- |
| Terms | Definitions |
| Unfamiliar terms or acronyms | Any term that is not common or could be misunderstood or any acronym no matter how obvious. |
| Actor | Specifies a role played by a user or any other system that interacts with the subject. |
| Advisor | a teacher or staff counselor who helps a student plan a course of study. [5] |
| API | Application Programming Interface; a set of subroutine definitions, protocols, and tools for building application software |
| Backend | In software architecture, backend usually refers to the layer of programming that handles data (information) storage and business logic (the rules that govern the program). |
| Breadcrumbs | A small text path usually at the top of a page that shows where the user is on a web page. |
| Client | Part of software that accesses information from another part of the software program or from another piece of hardware via a server in order to perform its role in the program. |
| Cookie | A small packet of information a website sends your computer to keep track of your actions on the website. |
| Database | A structured set of data that is held on a computer. |
| Encryption | A string of characters used in a computer program used to alter and store data so that the information appears random and undecipherable. |
| Entity | Something structurally, functionally, or otherwise distinct from other elements |
| Facade | A simplified interface. |
| FERPA | Family Educational Rights and Privacy Act: The law that allows parents the right to access their children’s education records and have the records amended. [4] |
| GUI | Graphical User Interface: A computer program that allows a user to interact with a computer using symbols, visuals, and pointing devices. [3] |
| HTTP | Hypertext Transfer Protocol: The rules that govern the way a web server communicates with a web client (usually a browser) |
| HTTPS | Hypertext Transfer Protocol Secure: The secure version of HTTP, the S stands for secure. HTTPS uses encryption to make sure communication between your browser and server is secure. |
| ISC | A file extension used to |
| Parent User Case | The Parent User Story links to the higher-level use case that is the parent to a particular use case. |
| Pseudocode | The notation used to represent a simplified programming language. This is used to design computer programs. |
| Primary  Key | Unique field in a table which identifies each row in a table. Cannot have null values. |
| RESTful Web service API | REST is an acronym for Representational State Transfer, it allows two software programs to communicate with each other. |
| Stakeholder | A person with an interest or concern in something. Someone who has an interest in the part of the program where the stakeholder is listed. |
| UI | User Interface: The part of the computer program where the user interacts with the computer. |
| UML | Unified Modeling Language: A standard notation used to diagram classes in Object Orientated Programming. Graphical diagrams used to show how a software program is divided into “classes”. |
| Use Case | A list of actions or steps between an Actor and a system to complete a goal. Used in Software engineering and Business analytics. |
| User | A person that utilizes a computer or network service. |
| Query | A request to a database for information or data. Usually, it is to retrieve information, but a query can be used to manipulate data also. |

# Requirements Traceability Matrix

|  |  |  |
| --- | --- | --- |
| Entity | Description | Sub Entity |
| Login Page (REQ 1) 3.1.1 | A form prompting the user to log in to their account when they first enter the application, if authentication fails error handling to prompt the user to try again. |  |
| Index (REQ 2)  3.1.2 | 3 sections at top of page with links for “Courses”, “Credits,” and “Degrees” the links will be available by the user to switch between pages at any time. | Class  Course |
| RESTful Web Service API (REQ 3)  3.1.3 | A backend API to interface with backend data of the application. | Registration API  Class |
| Planned Courses  (REQ 4)  4.1.1 | A list showing courses that need to be fulfilled by the student and suggested courses according to prerequisites complete queried from the backend. | Registration API  Class  Course  Major  4.1.1.1, 4.1.1.2 |
| Query (REQ-5)  4.1.2 | A list of available classes and their times for a given course number. | Class  Course |
| Query Display (REQ-6)  4.1.3 | A function for the grad planner to display queried class data. | Class  Course |
| Authorized Login (REQ-7)  4.2.2 | A web page redirects to BYU-I (CES) login screen that will redirect back to the grad planner upon success. | User Login Information  4.2.2.1  4.2.2.2 |
| User’s current class schedule (REQ 8) | A list of the user’s current class schedule. | Class  4.2.3.1 |
| Class time (REQ 9, 10) | Indicate the class time. | 4.2.4.1, 4.2.5.1 |
| Popup message (REQ 11) | A physical message displayed to the user if class times overlap with current class schedule. | 4.2.6.1, 4.2.6.2  4.2.6.3, 4.2.6.4 |
| Weekly Schedule  (REQ 12-14)  4.3.4 - 4.3.6 | A graphical representation of the user’s class schedule for the week differentiating each course. | Class, Course  Month, Date  Day, Time  Assignment  4.3.4.1, 4.3.5.1  4.3.5.2, 4.3.5.4 |
| Differentiate Class Display (REQ-15)  4.3.7 | When a user clicks a course, the selected course’s description shall be displayed. | Class, Course  4.3.7.1, 4.3.7.2  4.3.7.3, 4.3.7.4 |
| Close course description (REQ-16)  4.3.8 | The user shall have the ability to close out of the course description. | Class  Course |
| Export schedule  (REQ 17)  4.3.9 | The user shall have the ability to export their schedule as an ISC file. | Class  Course |
| Degree (REQ 18 – REQ 19)  4.4.3 - 4.4.4 | A data model that describes all requirements to graduate with said degree. | Class, Department  Catalog Year  Major, Minor  Cluster, Course  4.4.4.1, 4.4.4.2  4.4.4.3 |
| Track (REQ 20)  4.4.5 | A list of semesters (Winter, Spring, Summer, and Fall) that a student is registered to attend classes for. | Class, Course  4.4.5.1, 4.4.5.2  4.4.5.3, 4.4.5.4 |
| GradPlan (REQ 21)  4.4.6 | A list of planned classes and completed classes organized by semester and year. | Semester, Class  Degree  4.4.6.1, 4.4.6.2  4.4.6.3 |
| Reset GradPlan (REQ 22)  4.4.7 | A function to reset the GradPlan to the shortest possible path to graduation A degree requirement that can be completed by two or more different options. Visibly differentiated from other requirements by a small icon in the Grad Planner GUI. When a course is unavailable for a user's selected semester or user prerequisites, other qualifying course suggestions will be shown. | GradPlan, Track  Degree  4.4.7.1, 4.4.7.2  4.4.7.3 |
| Alternative Requirements (REQ 23)  4.4.8 | The system shall be able to state alternative requirements. | Class, Department  Catalog Year  Major, Minor  Course, Degree  4.4.8.1 |
| Reschedule Course  (REQ 24)  4.4.9 | Once courses are inserted into the user's semester, the user can drag and drop that course tag into another semester. The user’s GradPlan will update with this action. | Class, Department  Catalog Year  Major, Minor  Course |
| Saved Schedule Change  (REQ 25)  4.4.10 | Course schedule will be saved to the database. | Class, Department  Catalog Year  Major, Minor, Course  4.4.10.1 |
| Feedback Menu Icon  (REQ 26)  4.5.3 | When the user clicks the icon at the left of the box that represents the class, the user shall see a menu icon with the “feedback” option. | User Interface |
| Feedback Score  (REQ 27)  4.5.4.1 | The interface shall display an aggregate score based on previous feedback for a course or instructor. | Course, Instructor |
| Feedback Comments  (REQ 28)  4.5.4.2 | The user shall have access to the comments from other students about the class or instructor. | Course, Instructor |
| Feedback Type Selection  (REQ 29)  4.5.5.1 | The feedback form shall provide the user with the option of selecting either instructor or course feedback. | Course, Instructor |
| Instructor Rating  (REQ 30)  4.5.5.2 | The user shall have the ability to rate his/her instructor using a numerical scale. | Course, Instructor |
| Course Rating  (REQ 31)  4.5.5.3 | The user shall have the ability to rate the class using a numerical scale. | Course, Instructor |
| Class Comment  (REQ 32)  4.5.5.4 | The user shall have the ability to leave a comment about his/her experience with the class. | Course, Instructor |
| Anonymity  (REQ 33)  4.5.5.5 | The ability for the user to leave feedback and comments without revealing the name, email, username, or other means of identifying. | User |
| Fluid Design  (REQ 34)  4.6.1 | Design standard to display all features for any screen size: Desktop, laptop, tablets, mobile phones. | Cluster |
| Grad Planner Tour Option  (REQ 35)  4.7.3 | The option that will allow new users to take a tour of the software. | User  4.7.3.1  4.7.3.2 |
| Grad Planner tour button (REQ 36)  4.7.4 | The user will have the option to take the tour through a button in the Grad Planner in the Registration Integration settings. | User |
| Grad Planner Tour  (REQ 37)  4.7.5 | Once the user decided to take the tour, the system shall display in a modal way how to subscribe for classes, how to plan classes, and how to leave feedback to classes taken and professors. | User  Tour |
| UI: Sub-Second  (REQ 38)  4.8.1 | UI - Back-end connection, time response, and time and performance. | Performance  4.8.1.1, 4.8.1.2 |
| The Backend: Sub-Second  (REQ 39)  4.8.2 | UI - Back-end connection and UI Fails. | Performance  4.8.2.1, 4.8.2.2  4.8.2.3, 4.8.2.4  4.8.2.5, 4.8.2.6  4.8.2.7 |
| The UI: Final Sub-Second  (REQ 40)  4.8.3 | UI - Back-end connection, time response, and time and performance. | Performance  4.8.3.1, 4.8.3.2  4.8.3.3, 4.8.3.4  4.8.3.5 |
| Required classes  (REQ 41)  4.9.1 | The feature that displays the required classes corresponding to the declared major. | User, Class  API, Major  4.9.1.1, 4.9.1.2  4.9.1.3, 4.9.1.4  4.9.1.5, 4.9.1.6 |
| Class Filter  (REQ 42)  4.9.2 | The feature that allows the user to filter available classes by online vs on-campus classes. | User, Class  Schedule, API  4.9.2.1, 4.9.2.2  4.9.2.3, 4.9.2.4  4.9.2.5, 4.9.2.6 |
| Class Filter  (REQ 43)  4.9.3 | The feature that allows the user to filter available classes by daytimes classes. | User, Class  Schedule, API  4.9.3.1, 4.9.3.2  4.9.3.3, 4.9.3.4  4.9.3.5, 4.9.3.6 |
| Class Filter  (REQ 44)  4.9.4 | The feature that allows the user to filter available classes by nighttime classes. | User, Class  Schedule, API  4.9.4.1, 4.9.4.2  4.9.4.3, 4.9.4.4  4.9.4.5, 4.9.4.6 |
| Automatic Registration restriction  (REQ 45)  4.10.1 | The feature that will block automatic registration if there is any hold. | Registration, Holds  4.10.1.1, 4.10.1.2 |
| Automatic Registration check  (REQ 46)  4.10.2 | The feature that will check if the user has enabled the automatic registration feature. | API  Class  Registration |
| Automatic Registration: class status  (REQ 47)  4.10.3 | The feature that will check if a class is available or closed before an automatic registration. | API  Class |
| Automatic Registration (REQ 48)  4.10.4 | The ability the user shall have to enable or disable automatic registration. | Registration  User |
| Request class  (REQ 49)  4.11.1 | Request from Backend the info from the class selected. | Class |
| Get class  (REQ 50)  4.11.2 | Get the data from Backend to the UI (Code, name, description, seats available, instructors, day/time). | Instructor  4.11.2.1, 4.11.2.2  4.11.2.3, 4.11.2.4  4.11.2.5, 4.11.2.6 |
| Display class  (REQ 51)  4.11.3 | Display the data when registering (Code, name, description, seats available, instructors, day/time). | GradPlan, Track, User  4.11.3.1, 4.11.3.2  4.11.3.3, 4.11.3.4  4.11.3.5, 4.11.3.6 |
| Link  (REQ 52)  4.11.4 | Link to class information in the UI. | API |
| Efficient Design  (REQ 53)  5.1.1 | Design standard that queries and responses from the Registration API shall be handled and displayed to the user in an appealing format in under 1 second. | Instructor, Catalog, Year, Major, Minor, Degree, Course, User, Registration API |
| Browser Capability  (REQ 54)  5.1.2 | Design standard in which the system is designed for and can be used on all browsers. |  |
| Capability  (REQ 55)  5.1.3 | Design standard in which the system must be capable of handling the student body of BYU-Idaho. |  |
| Scalability  (REQ 56)  5.1.4 | The system must be able to scale to handle an increase in the student body at BYU-Idaho. |  |
| Reliability  (REQ 57)  5.1.3 | The system must perform as expected across the platform. |  |
| Set Cookies  (REQ 58)  5.2.1 | Creates and reads cookies. The cookie ID will then be used as part of the HTTPS request so that users can securely access the registration system to add/drop classes. |  |
| Send Cookies  (REQ 59)  5.2.2 | A cookie ID will then be used as part of the HTTPS request so that users can securely access the registration system to add/drop classes. |  |
| HTTPS Standard  (REQ 60)  5.2.3 | Design standard that HTTPS will be supported and used, not HTTP. |  |
| Encrypt User Database Data  (REQ 61)  5.2.4 | Ensure that all user data is encrypted in the database. | Database |
| Encrypt API Send  (REQ 62)  5.2.5 | Secure user data when making calls to the API. | User, Registration API, Encryption |
| Decrypt API Receipts  (REQ 63)  5.2.6 | Secure user data when receiving data from the API. | User, Registration API, Decryption |
| Filter  (Req 64)  5.2.7 | Filter the data by user input. | User Input |
| Class Status  (REQ 65)  5.4.1 | Indicate the class status (Open/Closed/Reopened/Waitlist). | Class, Instructor, Course |
| Class Registration Status (REQ 66)  5.4.2 | Indicate if a class is taken, currently taking, or planned to take, showing the class code and name in assorted colors. | Class, Course, Registration API |
| User Restriction  (REQ 67)  5.4.3 | The system shall restrict the user registering for when desires or wants are not feasible. | Registration API  5.4.3.1, 5.4.3.2  5.4.3.3, 5.4.3.4  5.4.3.5, 5.4.3.6 |
| Error Message  (REQ 68)  5.4.4 | The system shall show a message to the user if there has been an error such as registering for too many credits. | Registration API  5.4.4.1, 5.4.4.2 |
| Opening the system  (REQ 69) 5.4.5 | Shows the beginning of the screen. Shows classes in an organized manner for the user. | User  5.4.5.1, 5.4.5.2 |

# Use Cases

This section addresses the identified stakeholders and their design concerns in section 12.1 In section 12.2, the document will provide several context viewpoints including UML communication diagrams to provide an overview of each viewpoint. It will also provide use cases that are design entities that pertain to the corresponding context viewpoint.

Each UML communication diagram displays the function calls that are associated with each use case. The function names are labeled with a preceding number indicating which use case the function call corresponds to.

**Example:** The function labeled *"0.1 addToDoAssignments()"* located in the To-Do Assignment List diagram (**Figure 17**), corresponds to use case 3.2.3.0 "Load List" while the function *"1.1 markAssignmentCompleted(Assignment)"* corresponds to use case *“3.2.3.1 Mark Assignment Done"*. Each of these functions are defined in section 4.2 within their respective classes.

## Identified Stakeholders and Design Concerns

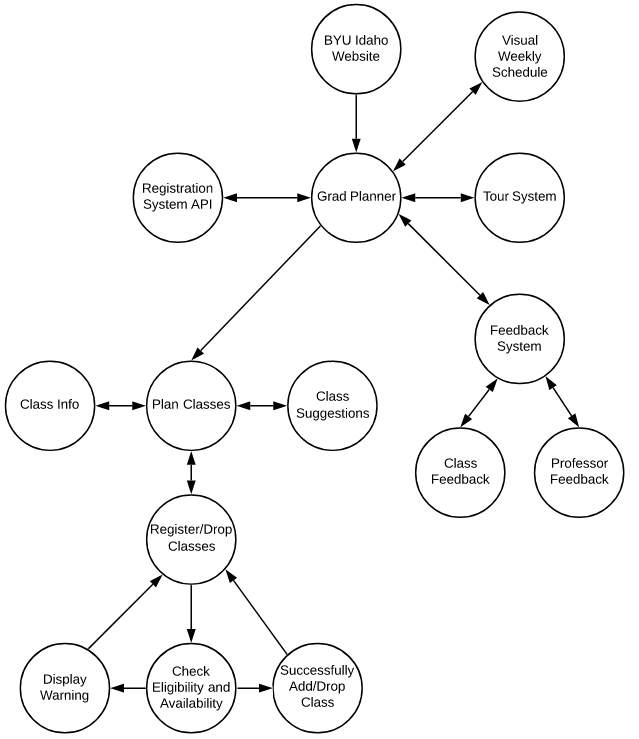
### **8.1.1 BYUI Student**

The BYUI student registers and uses the grad planner and registration system each semester and holds most of the design concerns.

### **8.1.2 Design Concerns**

* + - * The ability to log in without any complications (REQ 1)
        + A simple way to navigate through the application without getting lost (REQ 2)
        + The ability of the Grad Planner and the Registration system to connect without complications (REQ 3-6)
      * The security of the application (REQ 7)
        + Being able to have classes suggested based on classes already taken and current schedule (REQ 8-11)
      * The current school schedule shall be prominent on the page (REQ 12-14)
      * The ability to view details of each course (REQ 15-16)
      * A way to export the course schedule for the week. (REQ 17)
      * An accurate representation of which courses have been completed and which are still needed (REQ 18-19)
      * The representation of suggested course plans for upcoming semesters shall be explicit (REQ 20-22)
      * The ability to modify the course plan and save those changes (REQ 23-25)
      * Access to ratings/feedback for courses and instructors shared by other students (REQ 26-28)
      * The ability to filter classes based on type (Online, Daytime, Nighttime) (REQ 42-44)
      * The restrictions and holds on user’s account shall be displayed prominently (REQ 45)
      * The ability to enable automatic registration (REQ 46-48)
      * User shall be given an option to view class information with one click (REQ 49-52)
      * Quick and responsive to user input (under 1 second) (REQ 53)
      * Reliable performance regardless of browser (REQ 54)
      * The ability to handle the student body of BYUI (REQ 55)
      * The ability to handle an increase of student’s numbers (REQ 56)
      * Performance should be similar across all platforms (REQ 57)
      * Data must be handled and transmitted securely (REQ 58-64)
      * Classes show status and will only allow valid courses to be added (REQ 65-8)
      * Show a tutorial to a new user (REQ 69)

### **8.1.3 Functional Relationships**



**Figure 2 - Grad Planner Registration Integration Functional Diagram**

## Use Case Tables and Descriptions

User and System use cases provide design entities that show the interactions between the actors or personas and the system that are required. The following table is employed as a template to describe use cases in section 3.2. The left side of the table remains the same, while the right provides a description of the text that will be filled in according to the particular use case.

### **Degree Audit (REQ- 4)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 2 Requirements and Courses screenshot* |
| Page Title: | Degree Audit |
| Author: | Team 02 |
| Type: | Process |
| Purpose: | A tool that displays the current progress of a student in their currently selected major. This tool displays required and completed courses. |
| Parent User Story: | Grad Path |
| Stakeholders: | Student, Advisors |
| Pre-conditions/Product(s) Required: | The student must have a declared major. |
| Post- conditions/Product(s) Produced: | A table is displayed to the user with the Degree Audit. |
| Links: |  |
| SRS Document | REQ-4 |
| Description/Notes: | A Degree Audit shall show how the student is progressing on their current declared degree. The Audit shall display their completed courses as well as those they are still needing to complete, including the course code, course title, and total credits.   1. The user has a declared major. 2. The user shall select the option to have their graduation plan audited. 3. The graduation planner shall evaluate the requirements of the degree that has been chosen and compare it to the classes that the user has already completed, the courses the user has selected for future semesters as well as whether or not the user is meeting the Universities suggested credit hour load, and whether or not the user has selected enough credit hours per semester to graduate in as few years as possible. 4. The graduation planner shall display any warnings that the user's selections will not meet the above requirements. 5. The user will be given the option to correct the warnings as well as change any classes that have been selected. 6. When the user has passed the degree audit, all warnings have been cleared the degree audit will display that the graduation plan has been accepted. |

### **Class availability List (REQ- 5, 6, 42, 43, 44, 47)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 3 Available Class Search screenshot* |
| Page Title: | Class Registration |
| Author: | John Bake (Team 1) |
| Type: | Course |
| Purpose: | Allows student to search for available classes for a course. |
| Stakeholders: | Students |
| Parent User Story: | Grad Plan |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user must be logged in.  The class must be created. |
| Post-conditions/Product(s) Produced: | List of available classes. |
| Links: | None |
| SRS Document | REQ – 5, 6, 42, 43, 44, 47 |
| Description/Notes: | The user can search for available classes for a selected course filtering by class type. |

### **Class time (REQ- 9, 10)**



|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 3 Failed to register class screenshot* |
| Page Title: | Class Schedule |
| Author: | Dalan Ienatsch (Team 2) |
| Type: | Course |
| Purpose: | Verify times of enrolled courses don’t overlap with currently enrolled class times. |
| Stakeholders: | Students |
| Parent User Story: | Overlapping Classes. |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must be logged in.  The user must be enrolled in one or more classes. |
| Post-conditions/Product(s) Produced: | Error message. |
| Links: | None |
| SRS Document | REQ – 9, 10 |
| Description/Notes: | When the user attempts to register for a class:   1. The system checks the user’s currently registered class times. 2. If the user is attempting to enroll in a class that overlaps with a currently registered class’s time, an error message is displayed that the class is unable to be registered in. |

### **Class Status (REQ- 65)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 4 Class Status screenshot* |
| Page Title: | Class Registration |
| Author: | Adam Nielsen (Team 02) |
| Type: | Course |
| Purpose: | Indicates the status of the class as opened, closed, reopened, or waitlist. |
| Stakeholders: | Students |
| Parent User Story: | Grad Plan, Course |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The class must be created, and the user must be logged in to view the status of the class. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 65 |
| Description/Notes: | When the class is first created, the status will be “closed”. When the class is available to be registered, the status will be set to “opened”. The status will remain “opened” until the maximum number of students have registered for the class. When the maximum number of students have registered for the class, the status will be set to “waitlist”. If the maximum number of registered students has increased or a registered student has unregistered, students from the waitlist will be given the chance to register for the class. If there is still available room after waitlisted students have been given the chance to register, the class status will be set to “reopened”. |

### **Registration status (REQ- 66)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 5 Registration status screenshot* |
| Page Title: | Grad Planner |
| Author: | Adam Nielsen (Team 02) |
| Type: | Course |
| Purpose: | To indicate if a class is taken, currently taking, or planned to take. |
| Stakeholders: | Students |
| Parent User Story: | Grad Plan, Course |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must be logged in to view their own personal grad plan. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 66 |
| Description/Notes: | If a student has already taken a class, the class will have the status of “taken” and be displayed in the semester it was taken. If the student is currently taking a class or is registered to take a class, the status will display “currently taking”. If the course is in the planned courses of the students GradPlan, the course will be displayed as “planned to take”.  In the GradPlan, the registration status will be denoted by the color of the course. |

### **Course description (REQ- 15, 16, 49, 50, 51, 52)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 6 Course Description screenshot* |
| Page Title: | Grad Planner, Course Registration |
| Author: | Adam Nielsen (Team 02) |
| Type: | Course |
| Purpose: | To inform the user of the purpose and syllabus of the course. |
| Stakeholders: | Students |
| Parent User Story: | GradPlan, Course |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must be logged in. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 15, 16, 49, 50, 51, 52 |
| Description/Notes: | Contains a detailed description of the course provided by the department. |

### **Required Classes (41)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 7 Required classes screenshot* |
| Page Title: | List of Required Courses for Students Degree |
| Author: | Team 02 |
| Type: | Course |
| Purpose: | Help the user understand the required courses for their degrees. |
| Stakeholders: | Students |
| Parent User Story: | GradPlan |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must be logged in to view their own personal grad plan and then expand the requirements for their degree path. |
| Post-conditions/Product(s) Produced: | The user can drag and drop course selections into their grad plan for future semesters and be informed if that class is available and any course pre-requisites. |
| Links: | None |
| SRS Document | REQ – 41 |
| Description/Notes: | When the user clicks the folder tabs for General Education, Degree, or Elective Requirements the modal will show the requirements subcategories required for their degree and a list of course options per subcategory. The users completed courses will show in blue and the available ones in green. |

### **Automatic Registration (REQ- 48)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 8 Automatic registration screenshot* |
| Page Title: | Automatic Registration |
| Author: | Team 01 |
| Type: | Actor |
| Purpose: | The ability the user shall have to enable or disable automatic registration. |
| Parent User Story: |  |
| Stakeholders: | Student, Professor |
| Pre-conditions/Product(s) Required: | The user must be logged in. The user must be a current student. |
| Post- conditions/Product(s) Produced: | - |
| Links: | Additional references either inside this document or external references. |
| SRS Document | REQ- 48 |
| Description/Notes: | If the user has chosen classes before registration and added them to his plan, and the automatic registration is already chosen, the student will be registered during their registration period. |

### **Filter Data (REQ- 64)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 9 Filter data screenshot* |
| Page Title: | Filter Data |
| Author: | Team 01 |
| Type: | Process |
| Purpose: | Allows the user to search for a course that meets specific criteria. |
| Parent User Story: | Registration, Search |
| Stakeholders: | Student |
| Pre-conditions/Product(s) Required: | The user must be logged in. The user must be a current student. |
| Post- conditions/Product(s) Produced: | - |
| Links: | Additional references either inside this document or external references. |
| SRS Document | REQ- 64 5.2.7 |
| Description/Notes: | Filter data will allow the student to restrict which information is shown to them. Allowing them to search through less data when looking for a course. |

### **Opening Display Classes (REQ- 69)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 10 Class first time signup screenshot* |
| Page Title: | Sign up for the first time |
| Author: | Govert Carreño (Team 03) |
| Type: | Function |
| Purpose: | Help the user to understand how the systems work and show an automatic and complete plan. |
| Stakeholders: | Students |
| Parent User Story: | Log In |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must log in for the first time in the system. |
| Post-conditions/Product(s) Produced: | Store a tutorial flag as a cookie that it does not appear in future logins. |
| Links: | None |
| SRS Document | REQ – 69 |
| Description/Notes: | When the user opens the system for the first time, the system shall:   1. Show the beginning screen with labels showing each of the system functionalities. This is will be the tutorial with instructions for inexperienced users. 2. Organize the classes in separate boxes for each semester: The classes shall be automatically organized depending on the declared major   While the second step shall be made each time the user logs in, the first shall be available just the first time |

### **Export Schedule (REQ- 17)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 11 Export schedule screenshot* |
| Page Title: | Export Calendar Schedule |
| Author: | Andres Castro (Team 03) |
| Type: | Function |
| Purpose: | Export the Grad Planner calendar to sync with other Calendars (Like Outlook or Google Calendar) |
| Stakeholders: | Students |
| Parent User Story: | Weekly schedule |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | User logged Weekly Schedule opened |
| Post-conditions/Product(s) Produced: | A file or link (the user choice) downloaded from browser. |
| Links: | None |
| SRS Document | REQ – 17 |
| Description/Notes: | In the weekly schedule, the user shall have available the “export calendar” button, then the button shall display two options to export the calendar (link or file) when the user clicks one of them, the link will be shown, or the file will be downloaded. |

### **Menu Index (REQ- 2)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 12 Menu index mockup* |
| Page Title: | Menu Index |
| Author: | Team 03 (Jake Muhlestein) |
| Type: | Function |
| Purpose: | Show the Grad Planner in three different sections named “Courses,” “Credit,” and “Degree.” This will help the student navigate the Grad Planner to the functionality they wish to access. |
| Stakeholders: | Students |
| Parent User Story: | Log In |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The users must log in for the first time in the system and accessed the grad planner. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 2 |
| Description/Notes: | Each section within the application should maintain similar or identical styling. The current section shall be distinguishable by the individual user by use of visual cues such as highlighting and bolding of selected section’s link. Each section within the application will be available and accessible to the user at any given point. |

### **Update Schedule (REQ – 25, 24)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 13 Update schedule screenshot* |
| Page Title: | Update Schedule |
| Author: | Oscar Ampudia (Team 03) |
| Type: | Process |
| Purpose: | The user can move classes that are already placed in the user’s grad path to different semesters simply by dragging it. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Student |
| Pre-conditions/Product(s) Required: | The user is authenticated.  Add/drop a class. |
| Post-conditions/Product(s) Produced: | The schedule is updated based on the changes (add/drop course). |
| Links: | None |
| SRS Document | REQ-24, 25 |
| Description/Notes: | Once changes are made, they will be saved on the backend. |

### **Weekly Schedule (REQ – 8, 12, 13, 14)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 14 Weekly schedule screenshot* |
| Page Title: | Weekly Schedule |
| Author: | Oscar Ampudia (Team 03) |
| Type: | Process |
| Purpose: | Provide the student with a visual representation of the student’s weekly schedule for the current semester. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Student |
| Pre-conditions/Product(s) Required: | The user is authenticated.  Check if the user already accessed the Grad Planner Tour before.  The user is registered in a class. |
| Post-conditions/Product(s) Produced: |  |
| Links: | When the user clicks a course, the selected course's description shall be displayed.  The user shall have the ability to export their schedule as an ICS file. |
| SRS Document | REQ-8, 12, 13, 14 |
| Description/Notes: | The Weekly Schedule will be displayed on a portion of the screen. It will contain the current month, date and day of the week for each day, the assignments with a due date and time corresponding to the user’s time zone. Each of the user’s registered courses shall be differentiated graphically on the weekly schedule display. |

### **REST-ful Web Service (REQ- 3, 53)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | RESTful Web Service |
| Author: | Team 02 |
| Type: | Process |
| Purpose: | Connect to server. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | Web Service CRUD (create, read, update, delete) schema must be coded in program. |
| Post-conditions/Product(s) Produced: | The program connects to server via RESTful Web Service. |
| Links: | None |
| SRS Document | REQ – 3, 53 |
| Description/Notes: | When a student chooses a course a get statement is called then post to display the resource from the database that shows the information on the course |

### **Check Holds (REQ - 45)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 15 Check holds screenshot[[1]](#endnote-2)* |
| Page Title: | Check Holds |
| Author: | Team 02 (Ben Crowe) |
| Type: | Process |
| Purpose: | Check Holds will verify that the student has no holds placed on their registration. This will give the ability to have automatic registration as required by Requirement 45. |
| Stakeholders: | Students |
| Parent User Story: | GradPlan |
| Actor(s)/Persona(s): | Student |
| Pre-conditions/Product(s) Required: | 1. The user must be logged into the system.   One of the following must be true.   1. The user must be in the process of selecting new courses during registration. 2. The user must have automatic registration enabled. |
| Post-conditions/Product(s) Produced: | The user will be able to be registered for their chosen courses without further action. A student will not be able to register for a course when they have a hold. |
| Links: | None |
| SRS Document | REQ – 45 |
| Description/Notes: | Check Holds will be utilized in one of two situations during the open registration timeframe.   1. A user has an approved graduation planned filled out. Courses have been chosen for every semester in advance.    1. The user has selected automatic registration.    2. Check holds will be called to verify the user(student) has no holds placed on their registration.    3. If there is a hold the user will receive a warning and will not be able to continue with automatic registration until it is cleared.    4. If there are no holds the user will be registered for the classes in the upcoming semester without any further input by the user.    5. The user will receive a message that they have registered for the specified classes. 2. A user does not have a graduation plan filled out but chooses to use the grad planner to register for classes.    1. Check holds is called when a user (student) selects a course(s) from the grad planner during their registration period.    2. If there is a hold the user will receive a warning and will not be able to continue with registration until it is cleared.    3. If there are no holds the user will be registered for the classes in the upcoming semester.    4. The user will receive a message that they have registered for the specified classes. |

### **Enable or Disable Automatic Registration (REQ 46)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 16 Enable/disable automatic registration mockup* |
| Page Title: | Enable or Disable Automatic Registration |
| Author: | Team 02 |
| Type: | Process |
| Purpose: | To offer student a choice to enable or disable the automatic registration. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | After student creates their grad plan, they can choose to enable or disable the automatic registration by checking the checkbox selector. |
| Post-conditions/Product(s) Produced: | The selection is updated in database. |
| Links: | Checkbox |
| SRS Document | REQ – 46 |
| Description/Notes: | The enable or disable choice is for planned courses.   1. When registering for a new semester or creating a graduation plan, a box will be displayed enabling the user to be automatically registered for the classes they have selected in their graduation plan 2. During the user’s open registration period the graduation planner will automatically register them in the chosen classes if the automatic registration is enabled. 3. A user at any time may deselect the automatic registration option |

### **Display Feedback (REQ – 26, 27, 28)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | ***Figure 17 – Course Feedback screensot*** |
| Page Title: | Display Feedback |
| Author: | Team 02(Ben Crowe) |
| Type: | Process |
| Purpose: |  |
| Stakeholders: | Students, Professors |
| Parent User Story: | Grad Plan |
| Actor(s)/Persona(s): | Student |
| Pre-conditions/Product(s) Required: | 1. The user must be logged into the system. 2. The user must select a course from the list to view 3. The user must select either professor or course to view other Students feedback on course or the professor. 4. If no feedback is available, I.e. a previous student hasn’t written feedback, a notice will be displayed that no course or professor feedback is available. |
| Post-conditions/Product(s) Produced: | A display of course or professor for that course feedback so students can be more informed as to what expect from the course/professor with a number rating and how it will affect their workload for the semester they are planning. |
| Links: | None |
| SRS Document | REQ – 26,27,28 |
| Description/Notes: | 1. The user will click on a course (a link) from the course catalog or required degree classes they would like to research as a possible class to take for a given semester. 2. A pop up will display showing the course information, course number, prerequisites, credit hours. Users will be given an option to display student feedback. 3. If the user clicks on the student feedback option a box will appear given the user, the option to select the feedback students have given on the course or the professor. I.e. display course, professor feedback. 4. Depending on which option they choose (course, professor) the feedback previous students have given on the subject (course, professor) will be displayed along with a number rating. 5. To exit the feedback, display the user will be able to choose to exit the feedback screen entirely or choose the to view the feedback they didn’t select (course, professor) 6. Users will be given the option in all screens to exit the feedback display window. |

### **Provide Feedback (REQ – 29, 30, 31, 32, 33)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Provide Feedback |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Allow the user to provide feedback on classes and professors. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user opens a course that they have completed or a professor who taught a course that they have completed. |
| Post-conditions/Product(s) Produced: | The feedback is posted to the database. |
| Links: | None |
| SRS Document | REQ – 29, 30, 31, 32, 33 |
| Description/Notes: | The user shall have the ability to leave a comment about his/her experience with the class in the feedback modal. |

### **Degree (REQ-18, 19)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 18 Degree requirements screen shot* |
| Page Title: | Degree Requirements |
| Author: | Andres Castro (Team 03) |
| Type: | Process |
| Purpose: | Inform the user of the requirements necessary to apply for graduation. |
| Stakeholders: | Student |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Student |
| Pre-conditions/Product(s) Required: | The user must have a degree defined in his or her plan. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ- 18, 19 |
| Description/Notes: | The user shall have access to the requirements of their degree by clicking on the button "see details"  A window will be displayed with a checklist graph that will show the credits completed by the student in each category, internship status, and GPA required for graduation.  If the student has met one of the requirements, the graph will show it in green.  If the requirement has not been met, it will be gray. |

### **Restrictions (REQ- 11, 67, 68)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 19 Class restrictions error message screenshot* |
| Page Title: | Error Message |
| Author: | Govert Carreño (Team 03) |
| Type: | Function |
| Purpose: | Show the user the error description (when the user commits one) and restrict some features according to the university policies. These requirements are described in REQ-68 and REQ-69. More specific details available online: <http://www.byui.edu/Documents/student-records-and-registration/Registration%20FAQ/REGFAQsAugust.pdf> |
| Stakeholders: | Students |
| Parent User Story: | Class Registration |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user tries to register for a class in one or more of the following actions:   * The class is closed * The class has not opened seats * The class is already registered * The student has a class registered in the same schedule * If the total of credits for the semester is more than 32 * If there is any active restrictive hold |
| Post-conditions/Product(s) Produced: | A pop a message with the error description. |
| Links: | None |
| SRS Document | REQ – 11, 67, 68 |
| Description/Notes: | The function will stop any process in progress and shows the message with a detail restriction description. |

### **Login (REQ- 1, 7)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 20 Grad planner login screenshot* |
| Page Title: | Login |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Allow the user to log in and warn the user if the login fails. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | A compatible web browser such as Google Chrome, Firefox, Chromium, Safari, or Microsoft Edge.  The user enters the application.  The user enters their credentials.  The user has Duo Mobile. |
| Post-conditions/Product(s) Produced: | The user is logged in or a warning dialogue is displayed to the user. |
| Links: | None |
| SRS Document | REQ – 1, 7 |
| Description/Notes: | A form that prompts the user to log in that appears when the user enters the application.  Login success shall result in the application loading.  Login failure shall result in the application failing to load, prompting the user to insert their correct credentials.  The login verification shall be Multifactor Authentication. |

### **Semester (REQ – 20)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Track |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Show a list of semesters (Winter, Spring, Summer, and Fall) and the classes that a student is registered to attend. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 20, REQ-15 |
| Description/Notes: | The course should be organized into years and semesters.  On an off-track are differentiated graphically.  Complete, current and future semesters are differentiated graphically.  Class data is provided for the backend system.  Information about a course is displayed when the user clicks on the course see REQ-15. |

### **Grad Path (REQ - 21, 22)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 21 Grad path screenshot* |
| Page Title: | Grad Path & Reset |
| Author: | Team 03 (Jake Muhlestein) |
| Type: | Function |
| Purpose: | Show either the path of classes the student has already selected to complete their graduation or if no selection has been made the shortest path or required classes for the student to graduate. The user can reset their graduation path, which will default to the shortest path available. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user has logged in to grad planner site and has accessed their graduation path. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 21, 22 |
| Description/Notes: | The backend will query the school database to obtain a degree and associated path for the logged-in student. |

### **Possible Grad Classes (REQ - 23)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 22 Possible classes screenshot* |
| Page Title: | Select Grad Classes |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Show a list of Classes, with the Department, Year in the Major, and Minor for the Degree that a student is interested. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 23 |
| Description/Notes: | The courses should be organized into years and semesters in the Major/Minor of the students.  Class data is provided for the backend system. |

### **User Interface (REQ - 34, 38, 54)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Responsive Design |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Implement responsive design so that the grad planner shall display perfectly on any size screen or browser to create a pleasant interface. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students  Teachers |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 34, REQ-38, REQ-54 |
| Description/Notes: | The web pages should show an even display on different devices such as phones, tablets, and computers.  The interface should be even on all screen sizes as well and be compatible with each browser. |

### **New/Returning User (REQ - 35, 36, 37)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Grad Planner Tour Option |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | This option will allow new or returning users to take a tour of the software. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated.  The user is new.  Check if the user already accessed the Grad Planner Tour before. |
| Post-conditions/Product(s) Produced: | If the user clicks “Yes”, the user will see the Grad Planner Registration Tour.  If the user clicks “No”, the user won’t see the Grad Planner Registration Tour. |
| Links: | None |
| SRS Document | REQ – 35 |
| Description/Notes: | The Grad Planner Registration will look to see if the user has a cookie that would be deposited if the user previously accessed to the Grad Planner.  Show modal to prompt the user if they want to take the tour: “Would you like to take the tour of the Grad Planner.”  If the user clicks “Yes” the user will access the Grad Planner Registration Integration tour functionality and take a tour of the Grad Planner Registration Integration site. A cookie will be placed on the user’s computer indicating that the user has taken the Grad Planner Registration Integration tour and will no longer prompt the user to take the tour.  If the user chooses “No” a cookie will be placed on the user’s computer indicating that the user has refused to take the Grad Planner Registration Integration tour and will no longer prompt the user to take the tour. |

### **Grad Planner Tour Button (REQ - 36)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 23 Grad Planner tour launch button mockup* |
| Page Title: | Grad Planner Tour Button |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | The user will have the option to take the tour through a button in the Grad Planner in the Registration Integration settings. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated.  The user decided to not take the tour. |
| Post-conditions/Product(s) Produced: | The user can take the tour. |
| Links: | None |
| SRS Document | REQ – 36 |
| Description/Notes: |  |

### **Grad Planner Tour (REQ - 37)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | *Figure 24 Grad planner tour mockup* |
| Page Title: | Grad Planner Tour |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Teach the user how to use the app. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated.  The user decided to take the tour. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 37 |
| Description/Notes: | Show how to subscribe for classes, how to plan classes, and how to leave feedback to classes taken and professors. |

### **Response Time (REQ - 39, 40)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Response Time |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | The UI and Backend are sub-second on 95% of all transactions to be sub-second. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 39, REQ - 40 |
| Description/Notes: | The system is built to track transaction time between the UI to UI or UI to Backend. Verify with recorded data that this requirement is completed. |

### **User Traffic (REQ – 55, 56, 57)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | Scalability |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | The web pages should be having scalability, capability, and reliability as to fulfil what the user desires to complete. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students, Teachers |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 55, REQ – 56, REQ – 57 |
| Description/Notes: | The system will be able to support the number of students and faculty that will use the website at any given time. |

### **Cookies (REQ - 58, 59)**

|  |  |
| --- | --- |
| Data | Description |
| Page Title: | The cookie monster |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | The cookies will be sent from the webserver to the browser and then back to the webserver. This will allow the tracking of flow and user. |
| Stakeholders: | Students |
| Parent User Story: | N/A |
| Actor(s)/Persona(s): | Students |
| Pre-conditions/Product(s) Required: | The user is authenticated. |
| Post-conditions/Product(s) Produced: |  |
| Links: | None |
| SRS Document | REQ – 58, REQ - 59 |
| Description/Notes: | Creates and reads cookies. The cookie ID will then be used as part of the HTTPS request so that users can securely access the registration system. |

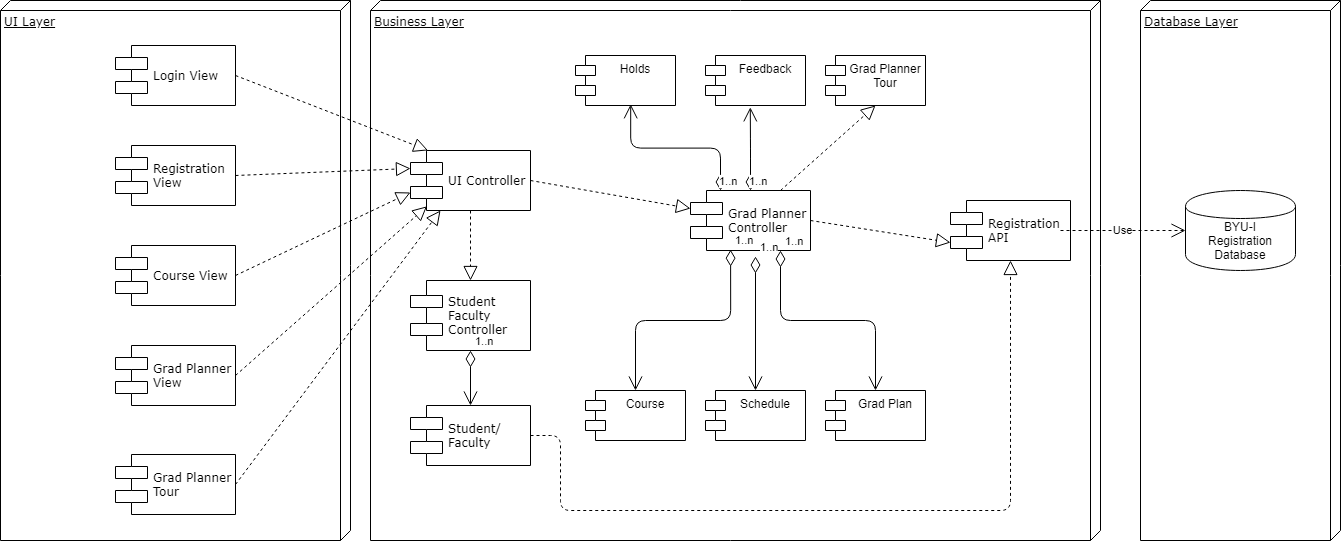
### **Secure Data Transfer (REQ - 60, 61, 62, 63)**

|  |  |
| --- | --- |
| Data | Description |
| Screenshot/Mockup: | The Screenshot/Mockup will contain a graphical illustration which will show an overlay of the use case under consideration. |
| Page Title: | Secure Data Transfer |
| Author: | Team 04 |
| Type: | Process |
| Purpose: | Secure data transfer during operations with the API. |
| Parent User Story: | - |
| Stakeholders: | Student |
| Pre-conditions/Product(s) Required: | A call is being made to the API. |
| Post- conditions/Product(s) Produced: | CRUD operations are encrypted. |
| Links: | - |
| SRS Document | REQ – 60, 61, 62, 63 |
| Description/Notes: | Design standard that HTTPS will be supported and used, not HTTP.  Secure user data when making calls to the API.  Secure user data when receiving data from the API. |

# Design Overview

The Design Overview will display a high-level representation of the Grad Planner Registration Integration Application's system design. The System Architecture section 4.1 presents a UML diagram to demonstrate the relationships between the software classes and how they will be integrated. This will provide the reader and user of the document a reference for the overall design. Further details of each design component are provided in section 4.2.

## System Architecture



***Figure 25 - System Architecture Diagram***

### **Architecture description**

The system architecture design represents the grad planner and all possible alternatives related to the project upon which it shall be built. The system design will help address any concerns for the stakeholders while helping to meet system requirements and ensuring maintained and consistent view. Class Descriptions

### **View Classes**

The view classes perform all the functions related to visually displaying the system to the user as well as allowing the user to interface with the system. They send signals to and receive data from the U.I. controller class.

#### View 1

|  |  |
| --- | --- |
| Class Name: Registration View | |
| Description: The registration view  The registration view is the screen that the student view while making decisions on which classes to register for a semester. Apart from registering, a student may decide to drop a class during a semester, receive warnings that may prohibit him or her from registering for a class, or it may be used just to gather information on a particular class that he or she may take in an upcoming semester. | |
| **Attributes (Fields)** | **Attribute Description** |
| Warnings | The user may receive a warning due to financial responsibility, not having an ecclesiastical endorsement, or registering for a semester not congruent with his or her on-track schedule. |
| Add/Drop | The user may add or drop a class to a current or future semester. |
| **Methods (Operations)** |  |
| Method | **Method Description** |
| Parameters: Add/Drop  Return: Boolean  Description: The method will return a Boolean verifying if a class or classes have been added or dropped. |
| **Method Pseudocode** |
| Add Drop (Add, Drop) {  IF Add is TRUE(  Add the selected class  ) ELSE IF drop is TRUE  Then drop the selected class  } |
| Method | **Method Description** |
| Parameters: Warning  Return: String  Description: The method will return a string that will show a warning if the user has a potential issue that must be fixed before registering for classes. |
| **Method Pseudocode** |
| Warning (warningType) {  IF student has warning (  Set warningType as the type of warning  )  RETURN warningType  } |

#### View 2

|  |  |
| --- | --- |
| Class Name: Login View | |
| Description: The login view  The login view is the screen that the student first authenticates to. The boxes of username and password are presented. The password is sent to the server and validated. The page also checks for a two-factor authentication cookie as given by the cookie monster. If the login succeeds and the two-factor authentication cookie is present, then the Login view proceeds to the destination view. | |
| **Attributes (Fields)** | **Attribute Description** |
| Warnings | The user may receive a warning due to a bad password or username. Also, if the 2-factor cookie is expired or doesn’t exist then the two-factor auth is performed. |
| Login | The user is required to log in. |
| **Methods (Operations)** | |
| Method | **Method Description** |
| Parameters: Login  Return: String  Description: The method will return a String verifying if the login is successful. |
| **Method Pseudocode** |
| Login(userID, pwd) {  IF userID == storedUser AND Hash(pwd) == storedPwdHash {  RETURN ‘Succeed’;  } ELSE {  RETURN ‘Login Error’  }  } |
| Method | **Method Description** |
| Parameters: 2nd Factor Authentication  Return: String  Description: The method will return a string that will show a warning if the user doesn’t have a valid factor Authentication Cookie. |
| **Method Pseudocode** |
| TwoFactorAuth(cookie) {  IF ValidateCookieByCookieMonster(cookie) == Valid {  RETURN ‘Valid Cookie’;  } ELSE {  RETURN ‘Cookie error’;  }  } |

#### View 3

|  |  |
| --- | --- |
| Class Name: Course View | |
| Description: The course view  The course view presents all the information about a class to the user. This information includes id, name, course description, teacher, code, availability, and prerequisite courses. This information should bring from the API. Also, this view provides the option to register for a class. | |
| **Attributes (Fields)** | **Attribute Description** |
| Course Id | Course Identification (unique). |
| Course Name | Name of the course. |
| Course Description | Description of the course. |
| Teacher | Name of the teacher. |
| Course Code | Course code. |
| Availability | When this class is available Ex.: Fall, Spring, etc. |
| Prerequisite course | List of class that are prerequisite to take this class. |
|  |  |
| **Methods (Operations)** |  |
| Method  Load Course | **Method Description** |
| Parameters:  Return: void  Description: This method gets the class info and display the information. |
| **Method Pseudocode** |
| loadCourse() {  Get class info  Show info in the view  } |
| Close View | **Method Description** |
| Parameters:  Return: void  Description: This method closes the course view. |
| **Method Pseudocode** |
| closeView() {  Dismiss open view  } |
| **Register** | **Method Description** |
| Parameters:  Return: void  Description: This method calls the API to register for a class. |
| **Method Pseudocode** |
| register() {  Check IF the class is available {  IF the class is available {  Call the API to register for a class  }  Close the view  }  } |

#### View 4

|  |  |
| --- | --- |
| Class Name: Grad Planner View | |
| Description: The Grad Planner View  The grad planner view is the interface the student will see for adding or moving classes in a certain window that will help navigate the course for his or her path to graduation. | |
| **Attributes (Fields)** | **Attribute Description** |
| Classes | Courses related to the student's major will be made present on the grad planner view. |
| Grad Plans | They will be made available as a resource to guide a student on which classes he or she should register for during a given semester. |
| Feedbacks | The feedbacks of classes and professors. |
| Student | The user logged into the Grad Planner. |
| **Methods (Operations)** |  |
| Display Grad Plans | **Method Description** |
| Parameters: userId  Return: display grad plans  Description: Displays the grad plans of the user. |
| **Method Pseudocode** |
| DisplayGradPlans(userId) {  Plans = GradPlanController request from API using the userId  Display table of grad plans  } |
| Display Classes | **Method Description** |
| Parameters: userId, semester  Return: display grad plans  Description: Displays the classes the user is taking each semester. |
| **Method Pseudocode** |
| DisplayClasses(userId, semester) {  Classes = ClassesController request from API using the userId and semester  Display table of classes  } |
| Display Feedback | **Method Description** |
| Parameters: class/professor  Return: display the feedback of the class/professor  Description: Displays the feedback for a class. |
| **Method Pseudocode** |
| DisplayFeedback(class, professor) {  Feedback = FeedbackController request from API using the class and professor  Display table of feedback  } |

#### View 5

|  |  |
| --- | --- |
| Class Name: Registration View | |
| Description:  The grad planner tour view consists of the optional guided tour presented to the user when first entering the grad planner. The tour guides the user through the many different functions of the Grad Planner.  The Grad Planner Registration will look to see if the user has a cookie that would be deposited if the user previously accessed to the Grad Planner  Show modal to prompt the user if they want to take the tour: “Would you like to take the tour of the Grad Planner.”  If the user clicks “Yes” the user will access the Grad Planner Registration Integration tour functionality and take a tour of the Grad Planner Registration Integration site. A cookie will be placed on the user’s computer indicating that the user has taken the Grad Planner Registration Integration tour and will no longer prompt the user to take the tour.  If the user chooses “No” a cookie will be placed on the user’s computer indicating that the user has refused to take the Grad Planner Registration Integration tour and will no longer prompt the user to take the tour. | |
| **Attributes (Fields)** | **Attribute Description** |
| Cookie | The cookie that is saved on the user’s browser. |
|  |  |
| **Methods (Operations)** |  |
| Prompt New User | **Method Description** |
| Parameters: N/A  Return: N/A  Description: The method will return a Boolean verifying if the user has a cookie saved. |
| **Method Pseudocode** |
| PromptNewUser() {  Cookie = ReadBrowserCookie()  IF Cookie is null or empty {  Dialogue. Show()  Dialogue.text = “Would you like to take the tour of the Grad Planner?”  IF user clicks “Yes” {  WriteBrowserCookie(“Took tour”)  TakeTour(TRUE)  } ELSE {  WriteBrowserCookie(“Refused tour”)  }  Dialogue.Hide()  }  } |
| Take Tour | **Method Description** |
| Parameters: isNewUser (Boolean)  Return: N/A  Description: This method starts the tour of the Grad Planner. |
| **Method Pseudocode** |
| TakeTour(isNewUser) {  IF (user clicks tour button in Grad Planner OR isNewUser) {  Then take the tour.  }  } |

### **Controller Classes**

The controller classes perform all the functions related to accessing and modifying data using the registration API. They send requests to the API for CRUD operations.

### **Model Classes**

The model classes represent different entities used in the operations of the system.

#### Student

|  |  |
| --- | --- |
| Class Name: Student | |
| Description: This class represents the student within the Grad Planner system. | |
| **Attributes (Fields)** | **Attribute Description** |
| Schedule | A list of courses the student is enrolled in. |
| Degree | The student’s chosen degree. |
| Completed Courses | A list of courses the student has completed. |

#### Course

|  |  |
| --- | --- |
| Class Name: Course | |
| Description: This class represents a course within the Grad Planner system. | |
| **Attributes (Fields)** | **Attribute Description** |
| Title | The name of the course. |
| Credits | The number of credits the course is worth. |
| Completed Courses | A list of courses the student has completed. |
| Instructor | The instructor of the course. |
| Seats | The number of open seats in the class. |
| Status | The current status of the class (i.e. Open, Closed). |
| Time | The day of the week and time this class meets, or if this is an online course. |

## System Interfaces

### **User Interface**

The user shall be able to navigate through the grad planner in an organized matter while comfortably registering for classes, planning for graduation, and having little to no concern for his or her future as a student at BYU-I.

### **Software Interfaces**

The software will need to interface with a database management system to retrieve data from and store data to a user account database. The connection will use a standard database connection technology.

## Constraints and Assumptions

### **Constraints**

#### Web Browser Compatibility

Due to the large number of web browsers available and limited resources to ensure compatibility with every web browser, these browsers will be tested for compatibility: Google Chrome, Firefox, Chromium, Safari, and Microsoft Edge. The browsers will be tested using their respective minimum versions that are available as of 2017.

#### Family Educational Rights and Privacy Act (FERPA)

The Grad Plan Registration Integration application will operate within FERPA, namely not disclosing educational records of any kind to parties other than the authorized user.

### **Assumptions**

#### University Authorization

#### It is assumed that the University will allow students to log in to their university account through the Grad Plan Registration Integration application and continue to pull course, assignment, group, activity, and grade information through the authorized account.

## Error Handling

### **Database Facade Class Commit Errors**

All commits to the database will be wrapped in a transaction. If any commit to the database fails during a procedure called in the database facade class, then the entire transaction will be rolled back, and an error message will be sent to the U.I. Layer.

### **Invalid Data Errors**

All data entered into the system by the user will be validated to determine if the data type entered matches with expected input (ex. a student name field should not contain any numbers). If the user attempts to enter invalid data into the system an exception will be thrown, and the user will be notified that the data is invalid.

### **I-Learn Importing Errors**

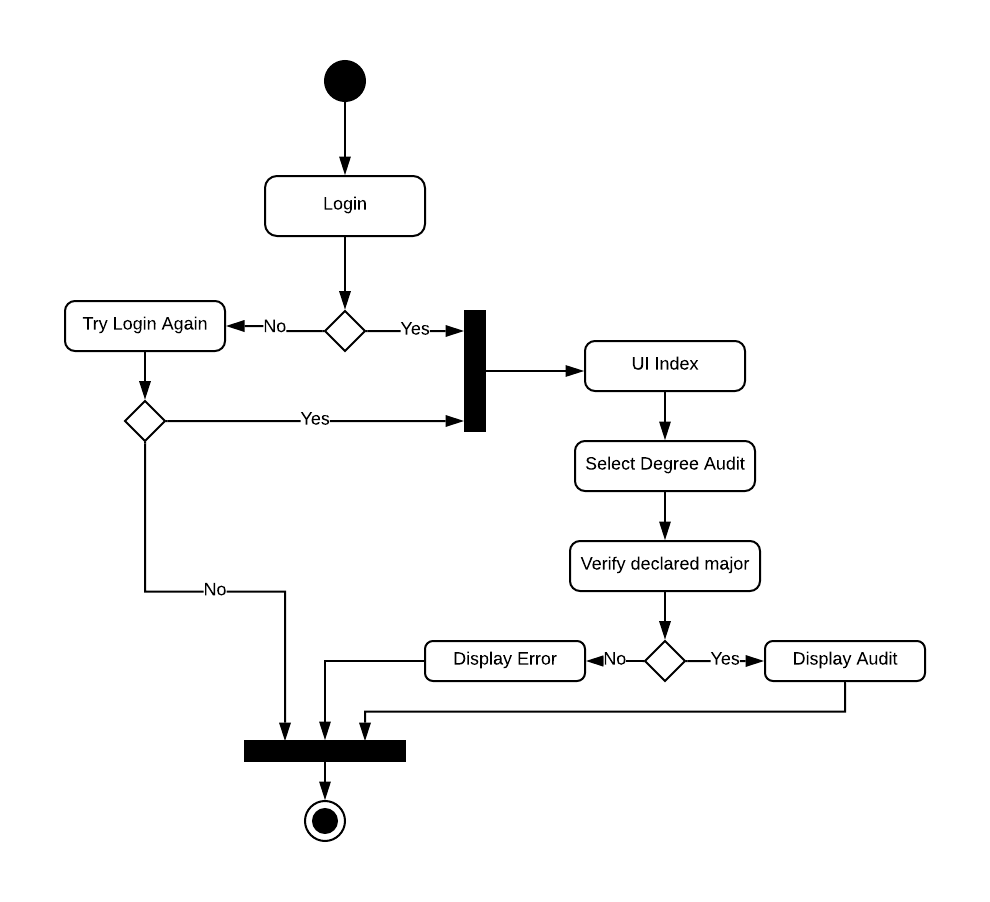
If a user's credentials do not match up to their log in information for I-Learn, an error will be thrown, and the user will be prompted to reenter their credentials or cancel importing their assignments from I-Learn. If any error results in the I-Learn API as it attempts to import assignment information into the system, the entire import process will be aborted, and the user will be notified that an import error occurred.

### **UI Controller Class Errors**

All data passing through the UI Controller will be validated before being sent to the UI layer. If the data appears to be corrupted a new request will be sent to the respective controller. If the same problem is retrieved a second time an error will be displayed to the user and sent to the administrative logs. If the information is missing or incomplete an error will be passed to the UI layer which will notify the user of the issue.

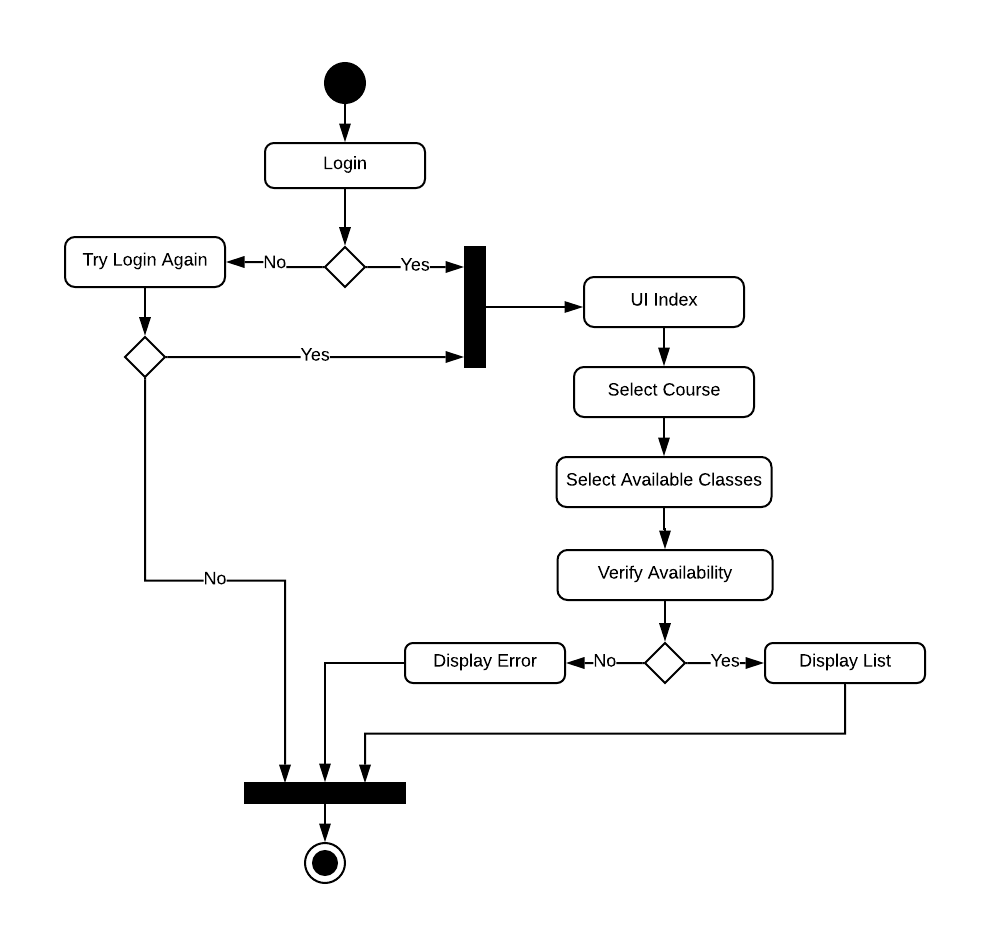
## Component Design

### **Degree Audit (REQ- 4)**



***Figure 26 – Degree Audit Diagram***

### **Class availability List (REQ- 5, 6, 42, 43, 44, 47)**



***Figure 27 – Class Availability Diagram***

### **Class time (REQ- 9, 10)**

|  |  |
| --- | --- |
| Class Name: ClassTime | |
| Description: This class describes a single class-time for a course. A course may have multiple class-times | |
| **Attributes (Fields)** | **Attribute Description** |
| Day | The day of the week (Monday, Tuesday, …). Stored as an enum. |
| StartTime | The time of day that the class-time starts stored as an integer representing the number of minutes into the day. Could also be stored in seconds.  Example:  0 = 12:00AM; 900 = 3:00PM; 1439 = 11:59PM |
| EndTime | The time of day that the class-time ends. See Start Time for details on values |
| **Methods (Operations)** | **Method** **Description** |
| **ConflictsWith(ClassTime)** | BOOL ConflictsWith(ClassTime time) {  TimeToGetBetweenClasses = 14min // a value greater than or equal to zero  IF Day is not equal to time.Day {  RETURN FALSE  }  IF StartTime or EndTime between (time.StartTime - TimeToGetBetweenClasses) AND (time.EndTime + TimeToGetBetweenClasses) (Inclusive) {  RETURN TRUE  }  IF time.StartTime or time.EndTime between (StartTime - TimeToGetBetweenClasses) AND (EndTime + TimeToGetBetweenClasses) (Inclusive) {  RETURN TRUE  }  RETURN FALSE  } |

### **Class Status (REQ- 65)**

|  |  |
| --- | --- |
| ENUM Name: ClassStatus | |
| Description: Represent the status of a class opened, closed, reopened, or waitlist. | |
| **Attribute (Fields)** | **Attribute Description** |
| OPEN = 0x1 | Course is open for adding. |
| CLOSED = 0x2 | Course is closed for adding. |
| REOPENED = 0x4 | Course has been reopened for adding. |
| WAITLIST = 0x8 | Course is available for wait listing. |

### **Registration status (REQ- 66)**

|  |  |
| --- | --- |
| ENUM Name: RegistrationStatus | |
| Description: Represent the registration status of a student in a course, taken, currently taking, planned to take. | |
| **Attribute (Fields)** | **Attribute Description** |
| TAKEN = 0x1, Navy Blue | Student has already taken this course. |
| CURRENTLY TAKING= 0x2  Baby blue | Student is currently enrolled in this course. |
| PLANNED TO TAKE= 0x4  Green | Student has planned to take this course in a future semester. |

### **Course description (REQ- 15, 16, 49, 50, 51, 52)**

|  |  |
| --- | --- |
| Class Name: CourseDescription | |
| Description: This class describes a description of a course. A course can only have one description. | |
| **Attributes (Fields)** | **Attribute Description** |
| Description | Description of the course. Stored as a string. |
| creditHours | Number of course Credit hours. Stored as an int. |

### **Required Classes (REQ- 41)**

|  |  |
| --- | --- |
| Class Name: Required Classes | |
| Description: This class contains the required courses for a given degree, the prerequisites for that course and it calls use registration status to display the classes that are planned, already taken. | |
| **Attributes (Fields)** | **Attribute Description** |
| General Education | GECourses = RegistrationAPIRequest  List of the required General education courses for a Bachelor of Science or Arts |
| ListOfClassesByMajor | MajorCourse = RegistrationAPIRequest  List of the required courses for the chosen Major  API request that has all information from Table on the Major Chosen |
| ListOfElectives | Electives = RegistrationAPIRequest  List of couses that will count as the required electives for a Bachelor of Science  API request that has all information from Table on the List of Electives |
| Major | Major = RegistrationAPIRequest  API request that has all information from Table on the Major Chosen |
| Prerequesites | Taken from the Major Object. List of the prerequisites for a given course |
| Registration Status | Status = RegistrationAPIRequest  API Request to verify the registration status of the student to determine they can register |
| **Methods (Operations)** |  |
| getMajor | String getMajor  Return Major |
| getRegistrationStatus | Int getRegistrationStatus  Return RegistrationStatus |
| getGeneralEd | String getGeneralEd  Return General Ed |
| getListOfClasses | String getListOfClasses  Return List of Classes |
| getListOfElectives | String get List of Electives  Return List of Electives |
| getPrerequisites | String getPrerequisites  Return Prerequesites |

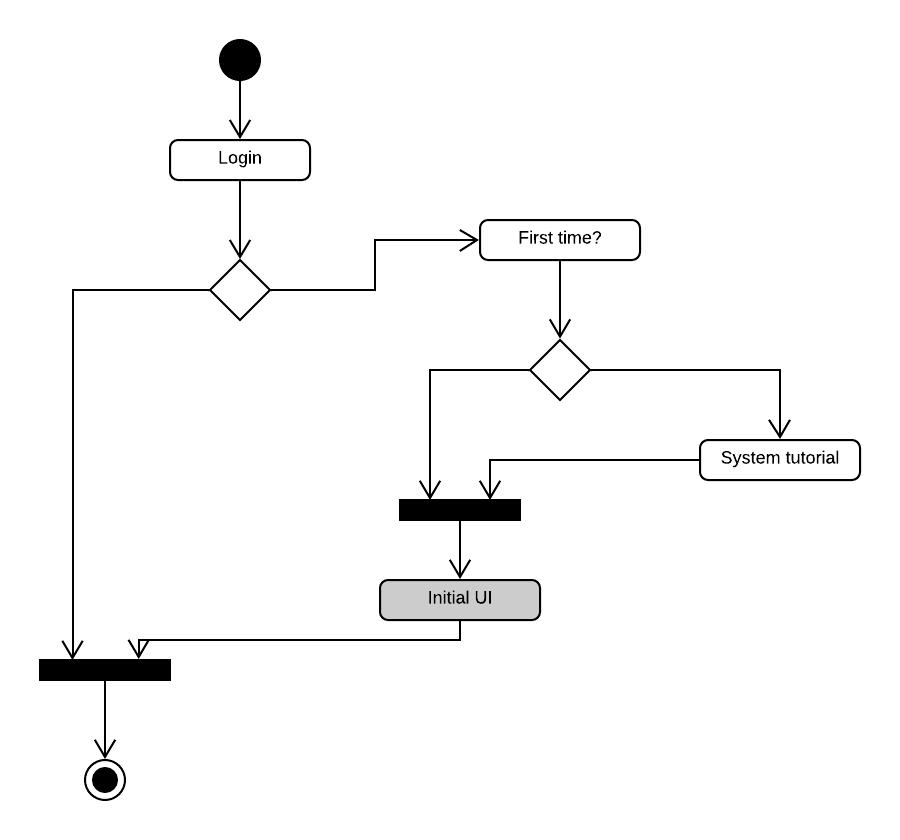
### **Automatic Registration (REQ- 48)**

|  |  |
| --- | --- |
| Class Name: AutomaticRegistration | |
| Description: This class describes whether a student has selected automatic registration. | |
| **Attributes (Fields)** | **Attribute Description** |
| autoReg | Whether automatic registration has been selected. Stored as a bool. |
| **Methods (Operations)** | |
| Is automatic registration selected | **Method Description** |
| Parameters: N/A  Return: N/A  Description: The method will return a Boolean verifying if the user has enabled automatic registration. |
| **Method Pseudocode** |
| autoRegIsSelected() {  Check IF automatic registration is selected {  IF it is then automatically register  } ELSE {  Do not automatically register  }  } |

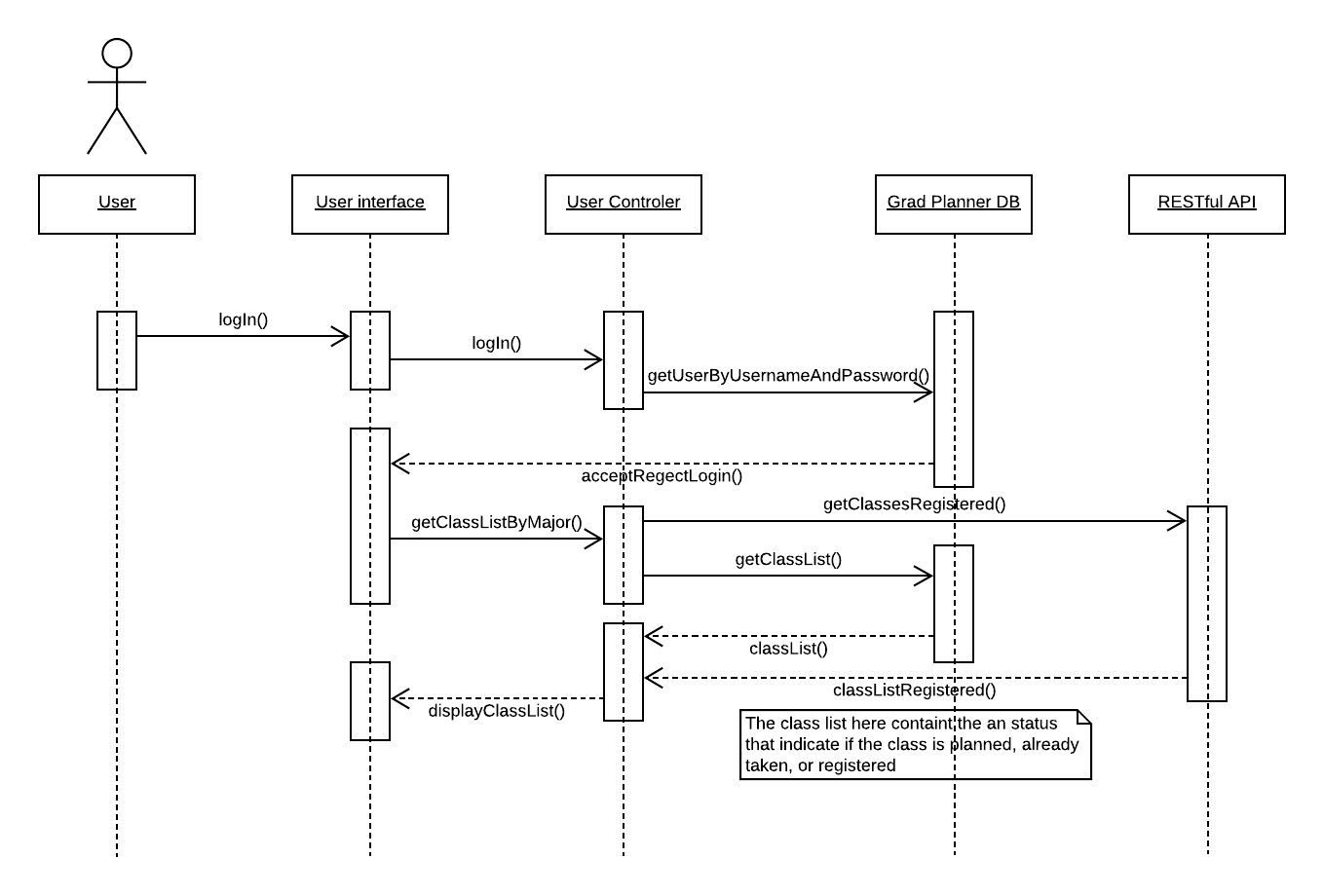
### **Filter Data (REQ- 64)**

|  |  |
| --- | --- |
| Class Name: FilterData | |
| Description: This class enables courses to be searched for that matches set criteria. | |
| **Attributes (Fields)** | **Attribute Description** |
| regExp | Word, phrase, or expression used to search for specific courses. |
|  |  |
| **Methods (Operations)** |  |
|  | **Method Description** |
| Parameters: String searchText  Return: Results  Description: Allows user to filter search results based on department, credit number, class type, and availability. |
| **Method Pseudocode** |
| FilterCourseData() | IF isLoggedIn {  Course course[];  For department of department {  IF searchDepartment matches department  Add department to course.result  }  For classType of type {  IF searchClassType matches class type {  Add department to course.classType.result  }  }  For credit of credit {  IF credit matches {  Add course.credit# to classType  }  }  } |

### **Opening Display Classes (REQ- 69)**

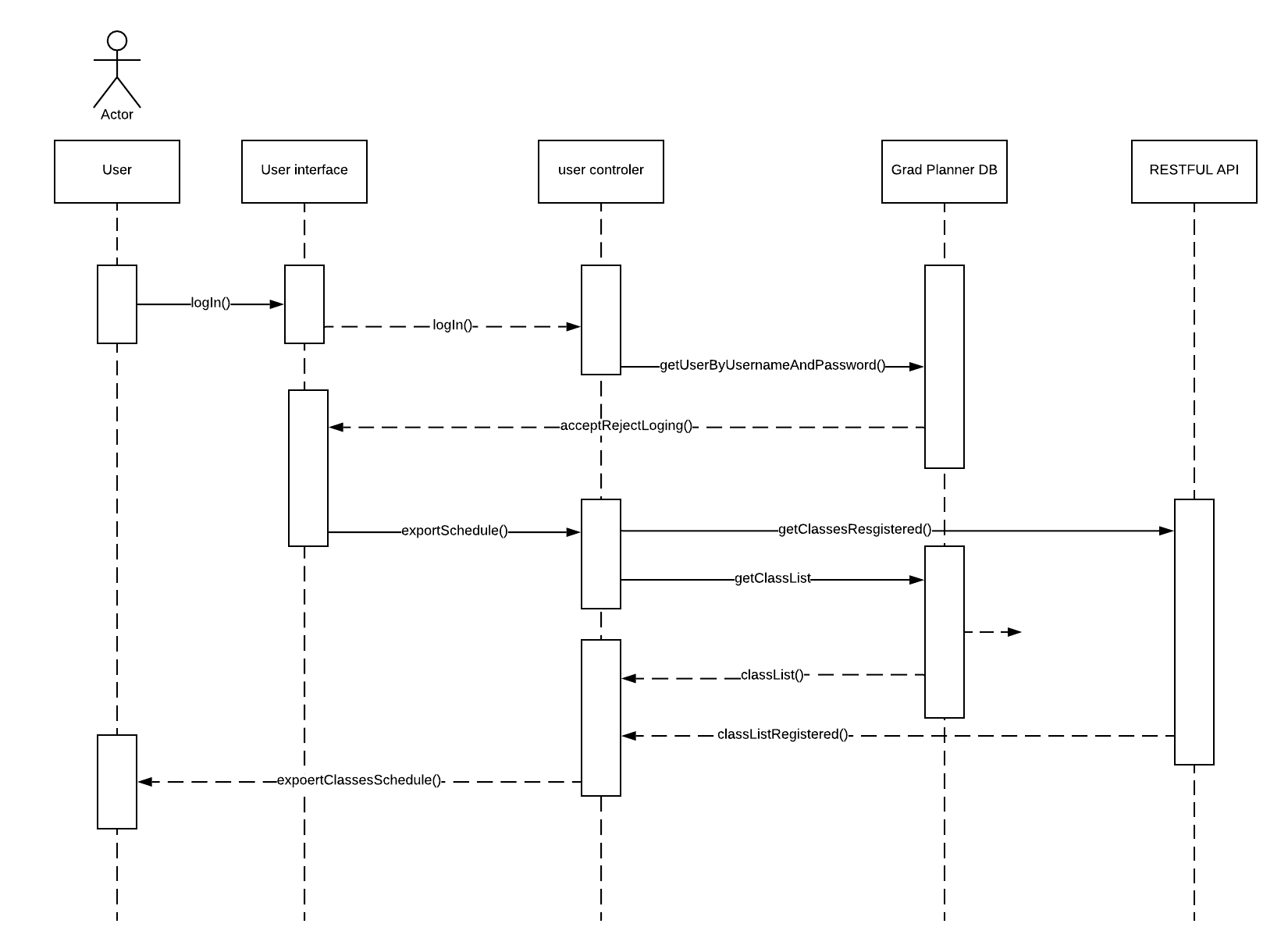


***Figure 28 - Login Relationship Diagram***



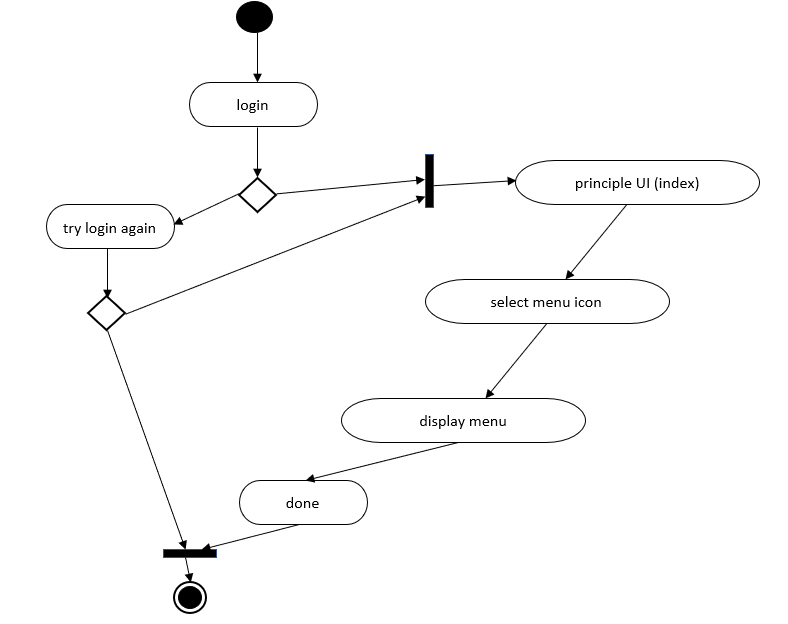
***Figure 29 - User Relationship Diagram***

### **Export Schedule (REQ- 17)**



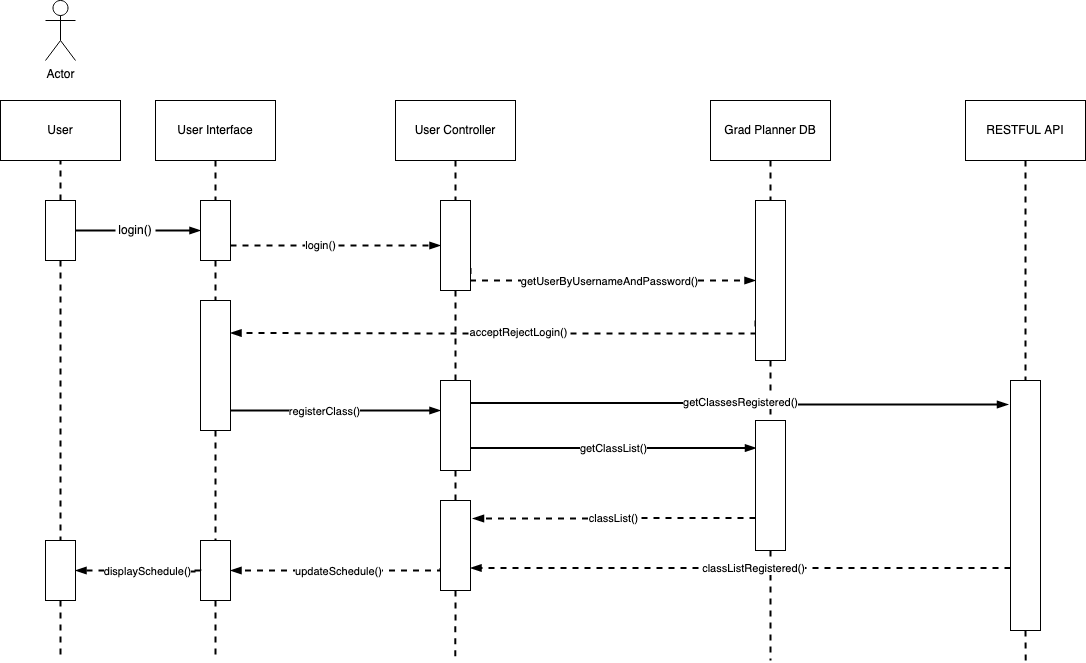
***Figure 30 – Export Schedule Diagram***

### **Menu Index (REQ- 2)**



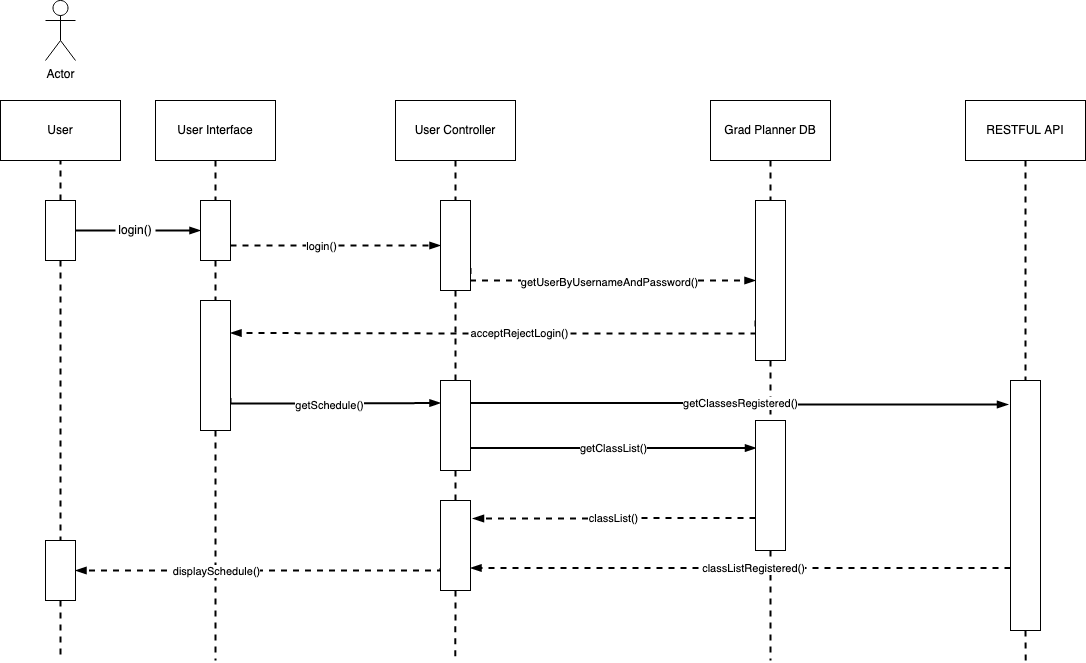
***Figure 31 – Menu Index Diagram***

### **Update Schedule (REQ – 25, 24)**



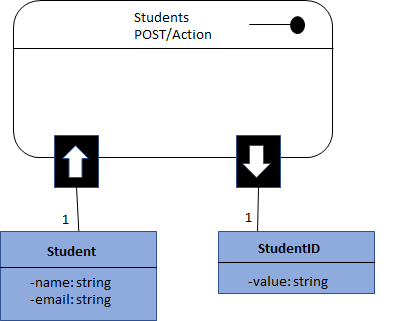
***Figure 32 – Update Schedule Diagram***

### **Weekly Schedule (REQ – 8, 12, 13, 14)**



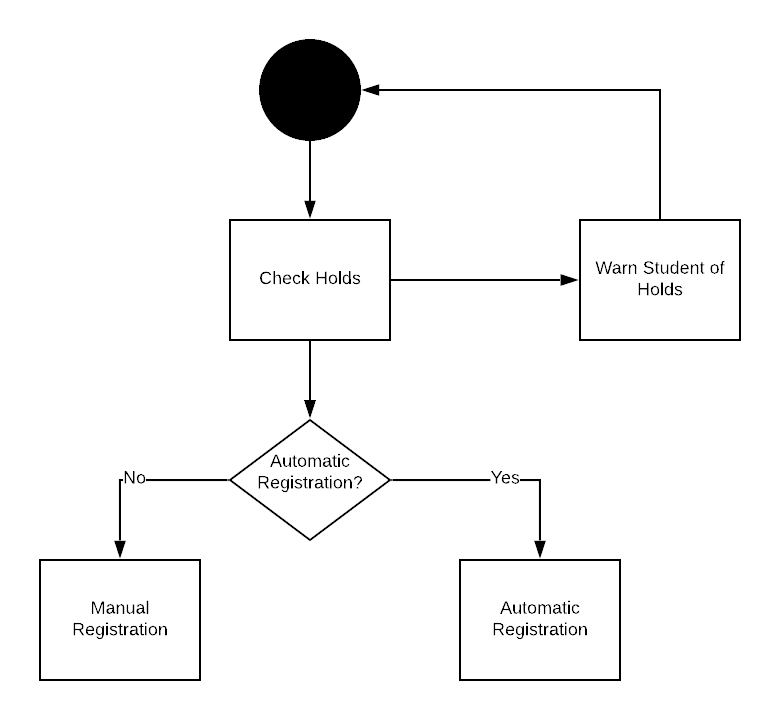
***Figure 33 – Weekly Schedule Diagram***

### **RESTful Web Service (REQ- 3, 53)**



***Figure 34 – RESTful Web Services Diagram***

### **Check Holds (REQ - 45)**



***Figure 35 – Check Holds Diagram***

### **Enable or Disable automatic registration**

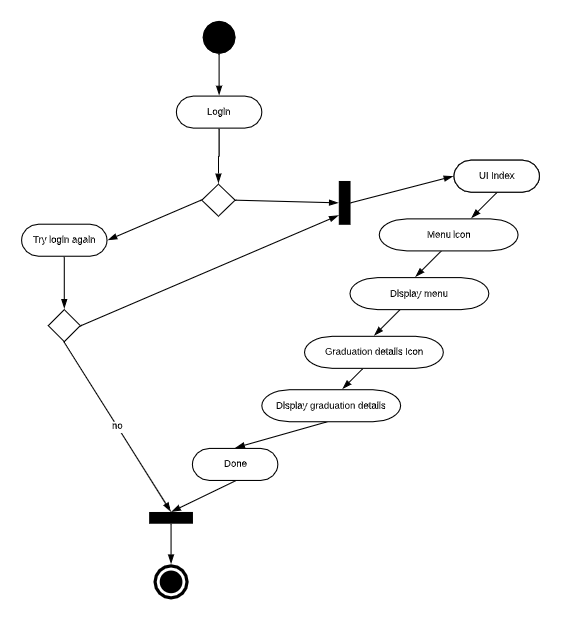
|  |  |
| --- | --- |
| Class Name: Enable or Disable automatic registration | |
| Description: This is a class that allows user to enable or disable automatic registration | |
| **Attributes (Fields)** | **Attribute Description** |
| autoReg | Whether automatic registration has been selected. Stored as a bool. |
| **Methods (Operations)** |  |
| Is automatic registration selected | **Method Description** |
| Parameters: N/A  Return: N/A  Description: The method will return a Boolean verifying if the user has enabled automatic registration. |
| **Method Pseudocode** |
| autoRegIsSelected() {  Check IF automatic registration is selected {  IF it is then automatically register  } ELSE {  Do not automatically register  }  } |
|  |  |
| autoRegEnabledMessage | Displays a message that the user has enabled auto-register. |
|  | **Method Pseudocode** |
|  | **AutoRegEnabledmessage**  **If autoReg**  **Print “You have chosen to have the Grad Planner Automatically register for the courses you have planned each semester”**  **Cin Y/N**  **Print “Thank You”** |

### **Display Course Feedback (REQ – 26, 27, 28)**

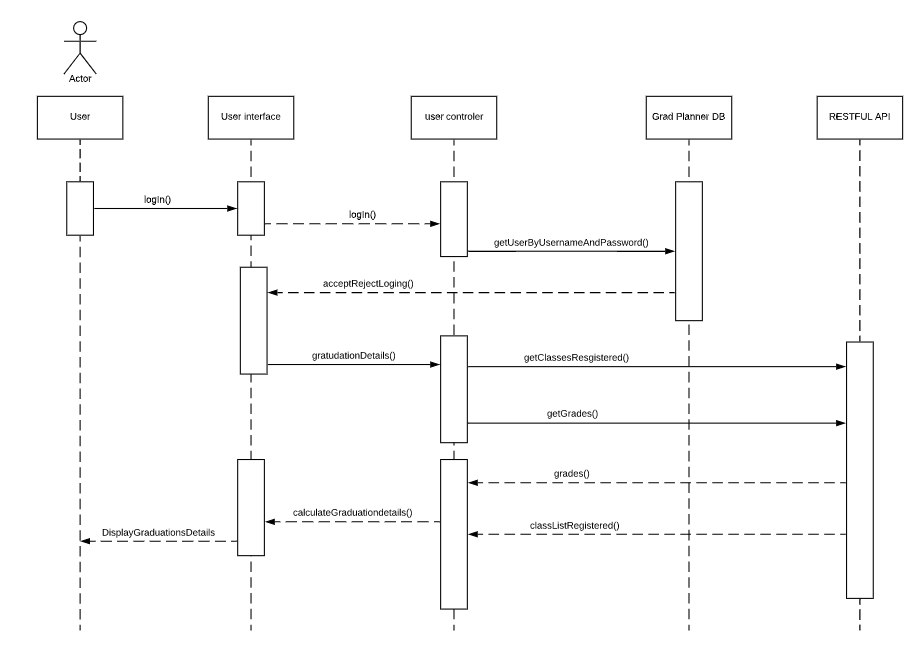
|  |  |
| --- | --- |
| Class Name: Display Feedback | |
| Description: This is a method that displays the feedback on a course. | |
| **Attributes (Fields)** | **Attribute Description** |
| courseFeedback | Object of CourseFeedback Class |
| **professorFeedback** | Object of ProfessorFeedback Class |
| **Methods (Operations)** |  |
| Display Course FeedBack | **Method Description** |
| Parameters: N/A  Return: N/A  Description: The method will be void. It will create a course Feedback and a Professor Feedback and display them. |
| **Method Pseudocode** |
| DisplayFeedback(class, professor) {  Feedback = FeedbackController request from API using the class and professor  CourseFeedback(Feedback)  ProfessorFeedback(FeedBack)  Display table of feedback  } |

### **Provide Feedback (REQ – 29, 30, 31, 32, 33) (Team 4)**

### **Degree (REQ-18, 19)**

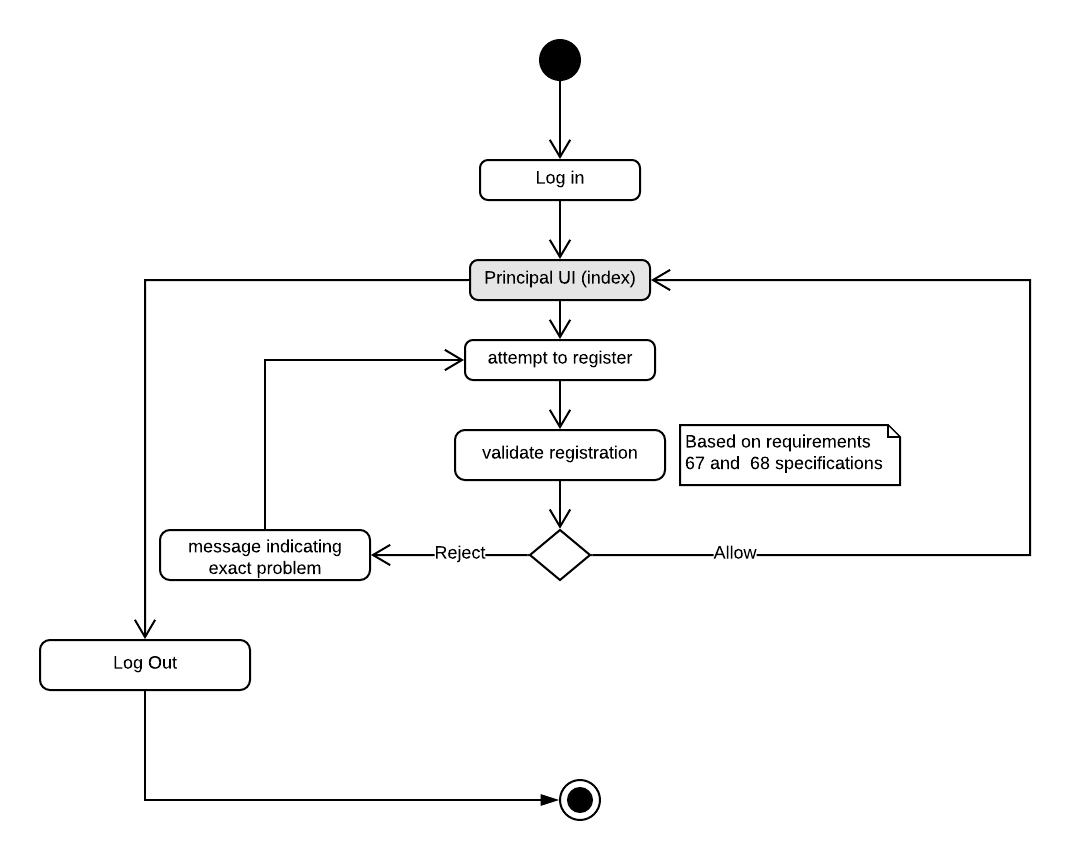


***Figure 36 – Degree Path Diagram***

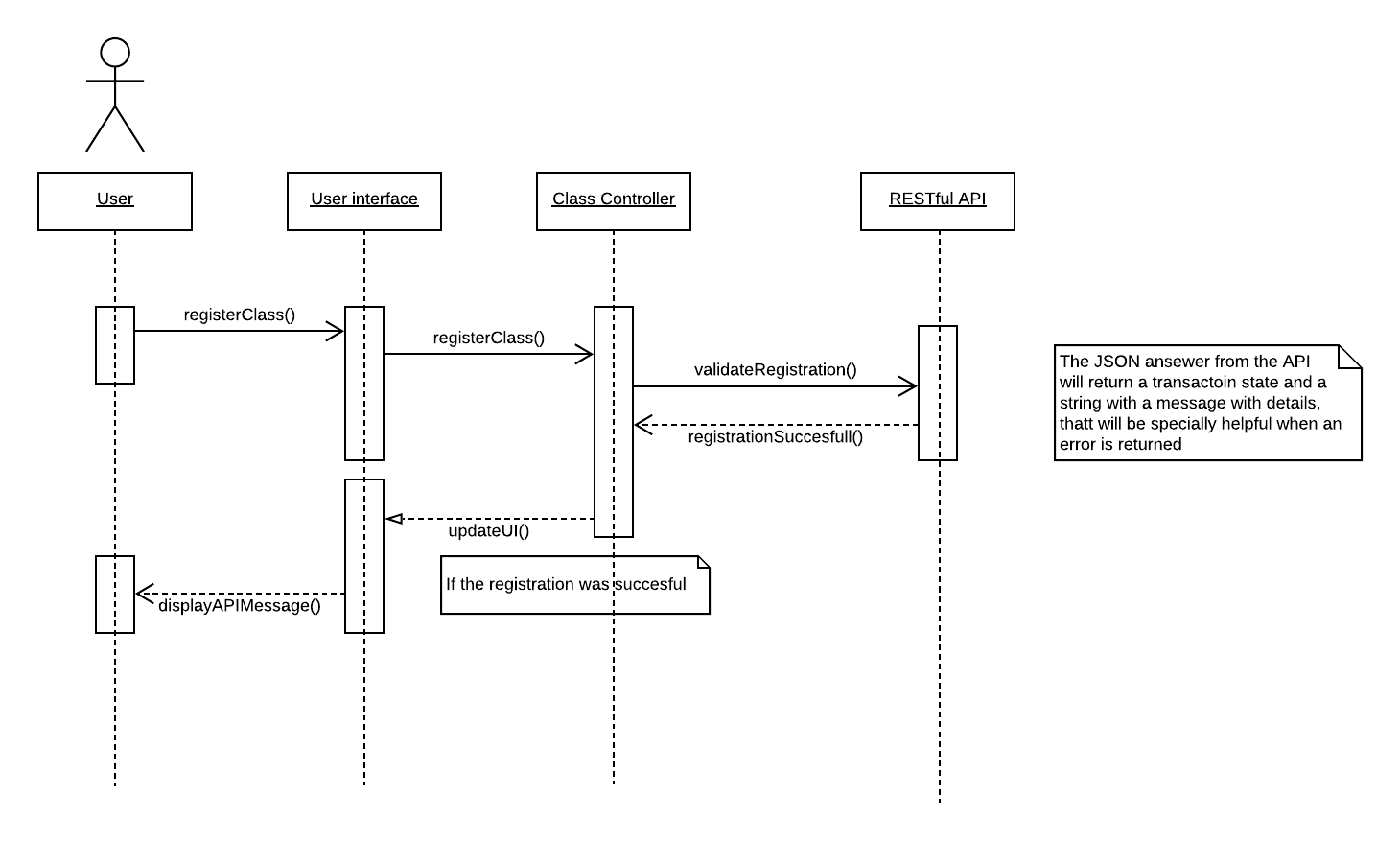


***Figure 37 – User Controller Diagram***

### **Class Registration (REQ- 11, 67, 68)**



***Figure 38 – Class Registration Diagram***

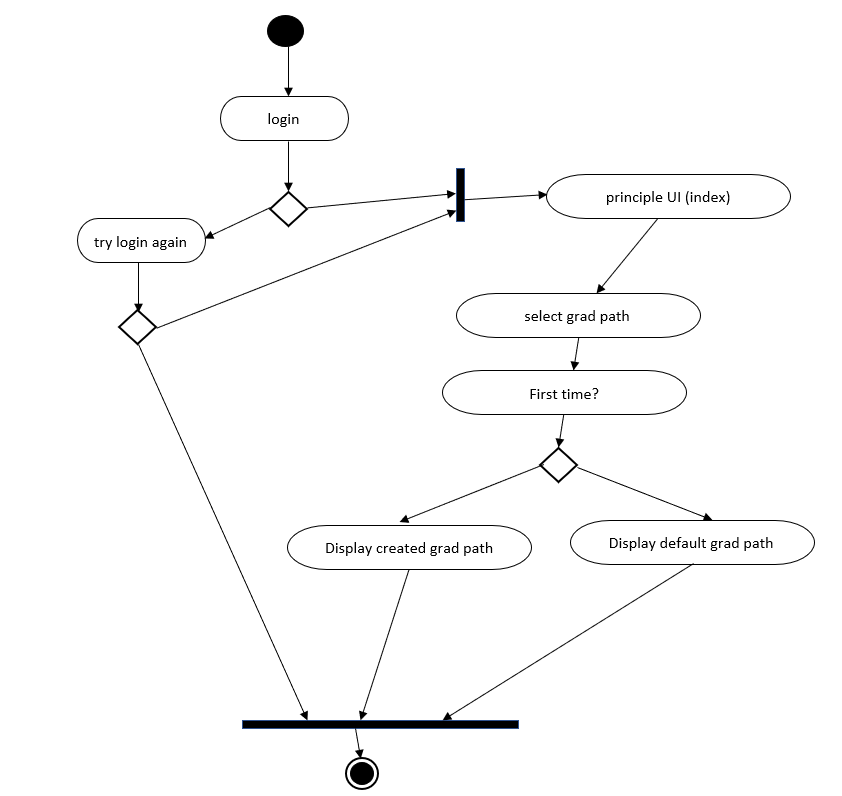


***Figure 39 – User Interface Diagram***

### **Login (REQ- 1, 7) (team 04)**

### **Semester (REQ – 20) (team 04)**

### **Grad Path (REQ - 21, 22)**



***Figure 40 – Grad Path Diagram***

### **Possible Grad Classes (REQ - 23) (team 04)**

### **User Interface (REQ - 34, 38, 54) (team 04)**

### **New/Returning User (REQ - 35, 36, 37) (team 04)**

### **Response Time (REQ - 39, 40) (team 04)**

### **User Traffic (REQ - 55, 56, 57) (team 04)**

### **Cookies (REQ - 58, 59) (team 04)**

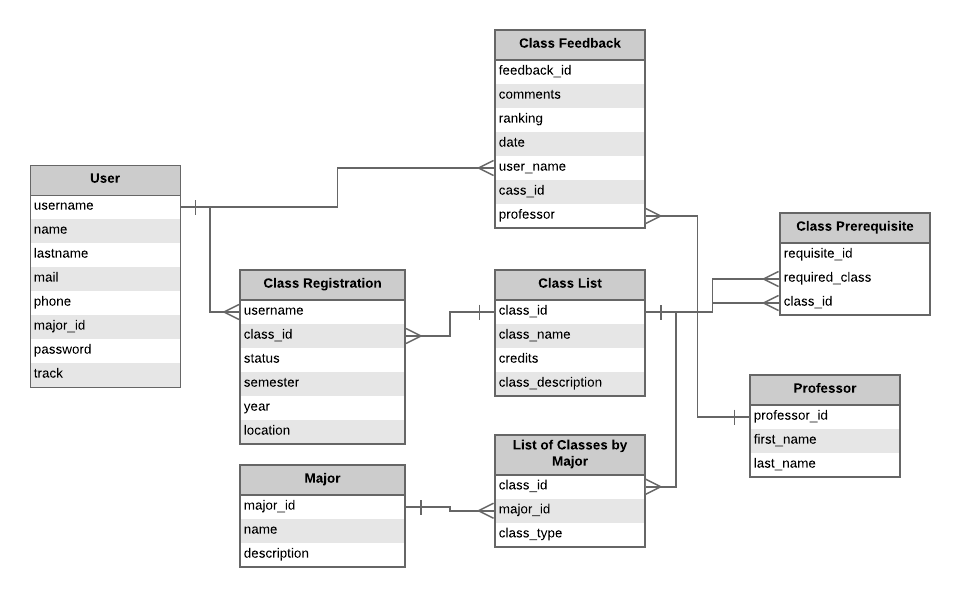
### **Secure Data Transfer (REQ - 60, 61, 62, 63) (team 04)**

# Data Design

## Data Description

The Grad Planner Registration Integration shall interface with the BYU course Registration System’s Informix Database through a RESTful web service API. Data requests shall be sent to the Informix Database using the SQL language. Data shall be communicated back to the Grad Planner Registration Integration as a JSON object. The JSON object shall then be analyzed by the client-side application and displayed to the user. Any changes made to the student’s registered classes shall then be sent to BYU’s course Registration System’s Informix Database as an SQL UPDATE

## Entity Relationship Diagram



***Figure 41 - Entity Relationship Diagram***

## Data Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table | Field | Type | Null | Default |
| **User** | user\_id | **PK** int | No | N/A |
|  | name | varchar(50) | No | N/A |
|  | lastname | varchar(50) | No | N/A |
|  | mail | varchar(50) | Yes | N/A |
|  | phone | varchar(50) | Yes | N/A |
|  | major\_id | int | No | 399 (General Studies Associate) |
|  | password | varchar(50) | No | N/A |
|  | track | int | No | 1 (Winter/Spring) |
| **Class Feedback** | feedback\_id | **PK** int | No | N/A |
|  | comments | varchar(500) | Yes | N/A |
|  | ranking | int | No | N/A |
|  | date | date | No | N/A |
|  | user\_id | **FK** int | No | N/A |
|  | class\_id | **FK** int | No | N/A |
|  | professor | **FK** int | No | N/A |
| **Class Registration** | user\_id | **FK** int | No | N/A |
|  | class\_id | **FK** int | No | N/A |
|  | status | int | No | N/A |
|  | semester | int | No | N/A |
|  | year | int | No | N/A |
|  | location | varchar(50) | No | N/A |
| **Class List** | class\_id | **PK** int | No | N/A |
|  | class\_name | varchar(50) | No | N/A |
|  | credits | int | No | N/A |
|  | class\_description | varchar(500) | Yes | N/A |
| **Class Prerequisite** | requisite\_id | **PK** int | No | N/A |
|  | required\_class | **FK** int | No | N/A |
|  | class\_id | **FK** int | No | N/A |
| **Major** | major\_id | **PK** int | No | N/A |
|  | name | varchar(50) | No | N/A |
|  | description | varchar(500) | Yes | N/A |
| **List of Classes by Major** | class\_id | **FK** int | No | N/A |
|  | major\_id | **FK** int | No | N/A |
|  | class\_type | int | No | 1 |
| **Professor** | professor\_id | **PK** int | No | N/A |
|  | first\_name | varchar(50) | No | N/A |
|  | last\_name | varchar(50) | No | N/A |

## SVG

## SVG Grad Planner logo

<path xmlns="http://www.w3.org/2000/svg" style="fill:#010002;" d="M223.336,148.384l-0.137-23.527l22.628-12.662L122.576,47.195L0,113.495l49.144,28.216 l0.098,16.766l0.01,1.339l0.449-0.215c-0.518,0.703-0.85,1.426-0.84,2.149c0.039,8.246,33.326,14.772,74.41,14.548 c41.064-0.215,74.302-7.122,74.273-15.349c0-0.723-0.381-1.426-0.889-2.149l0.449,0.215v-1.339l-0.088-16.834l21.309-13.258 l0.117,20.83c-2.345,1.006-3.976,3.312-3.957,6.009c0.02,3.537,2.892,6.399,6.458,6.37c3.586-0.02,6.429-2.912,6.409-6.439 C227.332,151.657,225.691,149.371,223.336,148.384z M123.241,170.621c-36.452,0.205-66.017-3.801-66.046-8.91 c-0.029-5.11,29.496-9.399,65.949-9.585c36.462-0.205,66.017,3.781,66.037,8.881 C189.209,166.098,159.703,170.426,123.241,170.621z M195.335,127.183c-4.934-5.188-22.618-18.886-72.426-18.602 c-49.877,0.264-67.336,14.128-72.211,19.394l-0.029-4.963c0,0,14.147-21.524,72.202-21.827 c58.025-0.313,72.436,21.045,72.436,21.045L195.335,127.183z M215.755,162.199l-2.511,36.433 c7.767-12.203,14.255-7.66,14.255-7.66l-0.156-28.832C218.998,165.414,215.755,162.199,215.755,162.199z"/>

1. [↑](#endnote-ref-2)