Software Requirements Specification

for

Grad Planner Registration Integration

Version 4.0

BYU-Idaho CS364 04 - Software Engineering

Instructor: Brother Lyon

June 11, 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 | 5/7/2020 | Template | - |
| W03 Ponder: SRS Version 0 | 5/8/2020 | Preliminary draft SRS | 0.0 |
| W04 Ponder: SRS Version 1 | 5/16/2020 | Updated draft SRS | 1.0 |
| W04 Ponder: SRS Version 2 | 5/23/2020 | Updated draft SRS | 2.0 |
| W04 Ponder SRS Final Version | 5/30/2020 | Final Version | 3.0 |
| W04 Ponder SRS Version 4 | 6/11/2020 | Changes suggested by sponsor and renumbered requirements | 4.0 |

Table of Contents

1. Introduction 1

1.1 Project Purpose 1

1.2 Project Scope 1

1.3 Document Conventions 2

1.4 Intended Audience 3

1.4.1 Developer 3

1.4.2 Project Manager 3

1.4.3 User 3

1.4.4 Tester 4

1.4.5 Document Writer 4

1.4.6 Team Leader 4

1.4.7 Sponsor 4

1.5 Product Scope 5

2. Product Overview 6

2.1 Product Perspective 6

2.1.1 Software Interfaces: 6

2.2 Product Functions 7

2.2.1 Registration 7

2.2.2 Drop Classes 8

2.2.3 Class Overlap 8

2.2.4 Weekly Schedule 8

2.2.5 Suggested Classes 9

2.2.6 Feedback 9

2.2.7 Optional Tour 10

2.2.8 Web Access 10

2.2.9 Navigation 10

2.2.10 Availability 10

2.2.11 Automatic Registration 11

2.2.12 Information 11

2.2.13 Required Classes 11

2.3 User Characteristics 13

2.3.1 Students 13

2.3.2 Support Technicians 13

2.3.3 Faculty Advisors 13

2.3.4 System Administrator 13

2.3.5 Database Administrator 14

2.4 Operating Environment 14

2.5 Design and Implementation Constraints 14

2.6 User Documentation 15

3. External Interface Requirements 15

3.1 User Interfaces 15

3.2 Hardware Interfaces 16

3.3 Software Interfaces 16

3.4 Communications Interfaces 17

4. System Features 17

4.1 Grad Planner System Backend Interface 17

4.2 Overlapping Classes 18

4.3 Weekly Schedule 19

4.4 Graduation Path 20

4.5 Feedback for courses and instructors 22

4.6 User Interface 23

4.7 Optional Tour 23

4.8 Performance Requirements 24

4.9 Classes Display and Filtering 26

4.10 Automatic Registration 28

4.11 Class Information Display 29

5. Other Nonfunctional Requirements 30

5.1 Performance Requirements 30

5.2 Security Requirements 30

5.3 Software Quality Attributes 31

5.3.1 Usability 31

5.3.2 Reliability 31

5.3.3 Portability 31

5.3.4 Maintainability 32

5.3.5 Reusability 32

5.3.6 Testability 32

5.4 Business Rules 33

6. Other Requirements 34

6.1 Appendix A: Glossary, Acronyms, and abbreviations 34

6.2 Appendix B: List of Figures 35

# Introduction

## Project Purpose

This document defines and provides detail on the requirements for the Grad Planner Registration Integration Project. The project is intended to provide students with a single tool for effectively and efficiently planning their college career and registering for those classes needed to complete all the degree requirements. To accurately describe stakeholder expectations of the system and its requirements, this document outlines the scope of the project, the intended audience, and the product features. Once a high-level view of the project is described, the system requirements and constraints are defined. After the core requirements are covered the system features and nonfunctional requirements will be outlined.

## Project Scope

The current Grad Planner tool allows students to create a plan detailing the classes to take for each semester of their college career. However, to register for those classes a student must access a separate registration system and search for each of the classes from the Grad Planner. The goal of the Grad Planner Registration Integration project is to create a tool that allows students to not only plan their classes but also register for those classes in the same tool. It will also include interfaces for viewing the student’s weekly schedule, providing course and instructor feedback, and touring the tool to become more familiar with using the Grad Planner.

The Course Planning Interface will create a course plan for all subsequent semesters based on the student’s degree requirements and completed courses. The student will be able to view information for each course and edit the course plan. This interface will also allow the student to directly register for available courses and drop existing courses. The system may also provide a way to register for all planned courses for the upcoming semester with a single click.

The Weekly Schedule Interface will provide the student with a visual representation of the student’s weekly schedule for the current semester. The System Tour Interface will provide interactive help to aid the student in more effectively using the system. The Feedback Interface will provide the student with a tool for providing feedback on both the instructor for each course and the course itself.

## Document Conventions

High-level requirements/features will be indented only one tab and will be a size 18 bold font, with a number in front of the requirement/feature indicating its ranking in the list of other high requirements/features. The most important high requirement/feature will be first on the list, and the list will start at number 1. Any sub-requirements/features under this first level will be indented one more tab than the previous, with a limit of two sub-levels per high requirement/feature. The sub-requirements/features will use numbers in a dot format to represent the subsections and will use a 12 or 14 font size. Sub-requirements/features will build on their parent by providing any specifications that might be needed.

The font will be a variation of Times New Roman, colored black. Any dramatization of text will only be done in italics (this means no exclamation marks). Underlining will only be used for title.

Semicolons will only be used in code snippets. Colons will only be used before a list. If there is a desire to use it in a sentence, reword it to avoid the possibility of improper usage. Any use of a question mark must be immediately followed by an answer (this means no sarcasm, hypothetical questions, or obvious questions without clearly stating the desired response for the reader).

Sentences will not exceed more than 40 words. Paragraphs will not exceed 10 sentences. This will increase the readability of our paper and will help us to formulate ideas in a way that increases understanding.

## Intended Audience

This document is intended for developers, project managers, users, testers, document writers, and any other individual interested in the design and development of the Grad Planner Registration Integration.

### Developer

The developers are the producers of a company. They work directly under the management of the project managers or a team leader. Developers take input and direction from the project manager and the team leads and build the product for the user. They will also take advice from stakeholders and users through the chain of command to better the project. There are many more developers then there are project managers or team leads. Developers will see the inner workings of the product and know exactly how the features are meant to work. The developers will read the fine details of the SRS to understand what the product needs to be.

### Project Manager

Project managers will lead and direct the developers, the testers, the documentation writers, and the marketing staff. The project manager’s job is to lead and direct the team to produce the product and get it into the user’s hand. They will direct the developers on what the user needs and what features to add to the product. The project manager will focus on the summary and the basics of the SRS to understand the overall product that needs to be created.

### User

Users are people who will be using the finished software product. Users are interested in why they should use this planning software, what features it has, and how it works. Users will use the SRS to learn how to use the application.

### Tester

Testers will use the SRS (Software Requirements Specification) to find the system features and their descriptions to produce accurate test cases. They put together special use cases where they run the defined software or system through a logical task and record what should happen. These are test cases, which provide valuable feedback to the developers and expose bugs.

### Document Writer

Document writers will be looking at the Overall Description and External Interfaces to create user manuals. They may go deeper if creating technical guides for advanced users. Like testers, they put together test cases, but to walk users or technicians through operation and troubleshooting.

### Team Leader

Anyone filling this position spends three-quarters of their time working on the actual project, and one quarter managing their team. They work directly with the Project Manager and relay any information received from the Project Manager directly to their team. This includes any information given to the Project Manager from the Sponsor. The team leaders are to coach or guide the writing process of their team, helping them produce the best work possible. After the work is produced, it is the responsibility of the team leader to deliver any progress directly to the Project Manager.

### Sponsor

The sponsor is the main point of reference for the Project Manager for improvements that must be made to the document. The instructions from the sponsor given to the Project Manager reach everyone through the hierarchy of our organization. There are periodic meetings held by the sponsor intended to inspire, course-correct, give advice, or help in another way so that we may produce the most meaningful, correct, and proper documentation.

## Product Scope

The product that will be developed shall be called the “Grad Planner Registration Integration.” The purpose of this application will be to provide a full-featured college graduation planner helping students with the registration process. The application will have different tools and features to assist with organizing the classes necessary to get a degree or certificate.

The objective of the “Grad Planner Registration Integration” is to create a more structured way of managing and scheduling the courses available to the student to get a specific degree or certificate desired.

The planner will allow the user to assign a degree or certificate program and allow the user to assign classes associated with that degree in each of the upcoming semesters. The planner will validate that the course is being taken in the correct sequence and that the course is available in that particular semester.

The planner will also keep track of the classes taken and keep a running count of credits and the accumulated GPA for the length of time the student has been attending the school. The course selection will be tied into the registration system and validate if the course is available and allow for registration of the class if the student is within their registration window. The college priorities will also apply so the students can get those classes most needed first to keep them on the graduation path. All this should give the student a good feeling about their future at the school and help them feel confident in their plan with minimal counseling necessary to set up their plan.

The goal of the “Grad Planner Registration Integration” application is to allow the student the ability to effectively and efficiently plan their college career and be able to register for those classes needed to complete all the requirements.

# Product Overview

## Product Perspective

This integration of the BYU-I grad planner with the registration system shall combine and enhance features of the grad planner to create a convenient and intuitive experience for the user to plan and register for classes at BYU-I. This product shall be an element of the BYU Idaho website and shall be integrated into the website. The grad planner shall grant the user the ability to register for and drop classes with similar functionality to the registration system. The grad planner will reflect course changes made in the registration system. The grad planner shall display views of the user’s weekly schedule for the semester. The grad planner shall provide a feedback system for classes, courses, and professors.

### Software Interfaces:

* Name: Grad Planner
  + - Mnemonic: N/A
    - Specification Number: N/A
    - Version Number: 2.0.1.320
    - Source: https://iplan.byui.edu/grad\_planner/index.html
    - Communication Interfaces: Local Network Protocols
    - Memory: N/A
    - Operations:
      * The user interface will consist of updating data from the grad planner to the registration or vice versa when the user has inserted new data.
      * Users shall have current feedback available about certain classes and professors to make. The grad planner has no feedback for students or teachers.
      * The new user shall be prompted by a touring tool that will help the user become familiar with the grad planner environment.
      * Users shall have access to a tool that will provide course suggestions.
      * The interface shall create a visible graph of registered classes as to alert the user if one class shall interfere with another in such a way as a time conflict.

## Product Functions

### Registration

The user shall have the ability to register for a class from the grad planner.

* UI (User Interface)
  + The user shall be able to view the class they want to register for with a click.
  + The user shall be able to click a button to register for the class.
  + The user interface shall send a request to the backend to register for the class.
  + The user interface shall update the user’s class schedule displayed to the user.
  + The user interface shall update the user’s expected graduation date displayed to the user.
* Backend
  + The backend shall verify the class is open.
  + The backend shall complete registration for students and class.
  + The backend shall update the expected graduation date.
  + The backend shall return the status to the user interface.
  + The grad planner shall calculate and update the expected graduation date when the user registers for a class.
  + The grad planner shall not allow the user to register for a class that overlaps another class.
    - Adding classes that have time overlaps will display an error to the user.
    - User will be able to see classes listed that have time overlaps but will not be able to add them.

### Drop Classes

The user shall have the ability to drop a class from the grad planner. (Execute requirements 1-3 in [section 4.1](#_Grad_Planner_System))

* UI (User Interface)
  + The user shall be to view the class they want to drop with a click.
  + The user shall be able to click a button to drop the class.
  + The user interface shall send a request to the backend to drop the class.
  + The user interface shall update the user’s class schedule displayed to the user.
  + The user interface shall update the user’s expected graduation date displayed to the user.
* Backend
  + The backend shall drop the class in the user’s schedule.
  + The backend shall update the expected graduation date.
  + The backend shall return the status to the user interface.
  + Dropped classes after the start of school will be added to the next semester unless otherwise specified by the student.

### Class Overlap

The grad planner shall display a message indicating if classes overlap when the user attempts to register.

### Weekly Schedule

The grad planner shall have a graphical view of the user’s weekly schedule for the current semester.

* UI (User Interface):
  + The interface shall display the student’s schedule for each day of the week.
  + The user shall hall have the ability to click on any item to see the description of the activity or assignment.
* Backend:
  + The weekly calendar shall connect to the I-Learn system to access course data.
    - Execute requirements in [Section 4.1](#_Grad_Planner_System).

### Suggested Classes

The grad planner shall display suggested classes to take each semester for the fastest path to graduation for the user.

* UI (User Interface)
  + The interface shall display a list of required courses that the user may add to the suggested schedule.
    - The interface shall provide a graphical element the user may use to remove a class from the suggested schedule.
    - The interface shall provide a graphical element the user may use to replace a class with one from a list of other available classes.
* Backend:
  + The classes shall be planned for each semester according to the declared major.
  + In the database, the major shall have a list of required and most common courses
    - Execute requirements in [Section 4.1](#_Grad_Planner_System)

### Feedback

The user shall possess the capability of leaving feedback for courses and professors.

* UI (User Interface)
  + Users shall be able to access a course feedback link for currently ongoing and enrolled courses for each student.
  + Users shall be guided through the completion of a feedback form and be returned to the previous page upon completion.
  + The feedback form shall provide users with a multiple-choice survey and have the option of leaving custom feedback through a textbox.
  + User feedback shall be anonymous from within the UI.
* Backend
  + The existing course evaluation system shall be utilized for the course and instructor feedback.
    - Execute requirements in [section 4.5](#_Feedback_for_courses)

### Optional Tour

The grad planner shall display an optional tour for new users of the system the first time they open the grad planner.

* UI (User Interface)
  + The tour prompt shall be displayed as a modal.
  + The tour prompt shall contain an option to either accept or decline the tour.
  + The tour prompt shall require the new user to accept or decline the tour to access the application.
  + The tour shall be accessible to the user with just one click.
  + Declining the tour shall display a prompt asking the user to confirm their selection.

### Web Access

The interface for the grad planner shall be consistent on all web-accessible devices for better human-computer interaction.

* UI (User Interface)
  + All features of the grad planner shall be accessible on all web-accessible devices.
  + Elements of the grad planner will scale according to user media-device size, resolution, and orientation.

### Navigation

### Availability

The grad planner shall have the option to filter through classes that are available specifically to online, daytime, and evening students

* Execute requirement 2 in [section 4.1](#_Grad_Planner_System)

### Automatic Registration

The user shall be automatically registered for planned courses for a semester on their registration date if all the following are met.

* The user does not have holds on their student account.
* The user has completed all prerequisites.
* The student has enabled auto-registration.
* The class is still open.

### Information

The user shall be able to view information on a class including when registering for an open class:

* The users shall see how many seats are available.
* The users shall see what professors are teaching the classes.
* The users shall see what day and time options are available.

### Required Classes

The grad planner shall display the required classes for the user’s current field of study. The user shall have the option to filter through classes that are available specifically to online, daytime, and evening students.

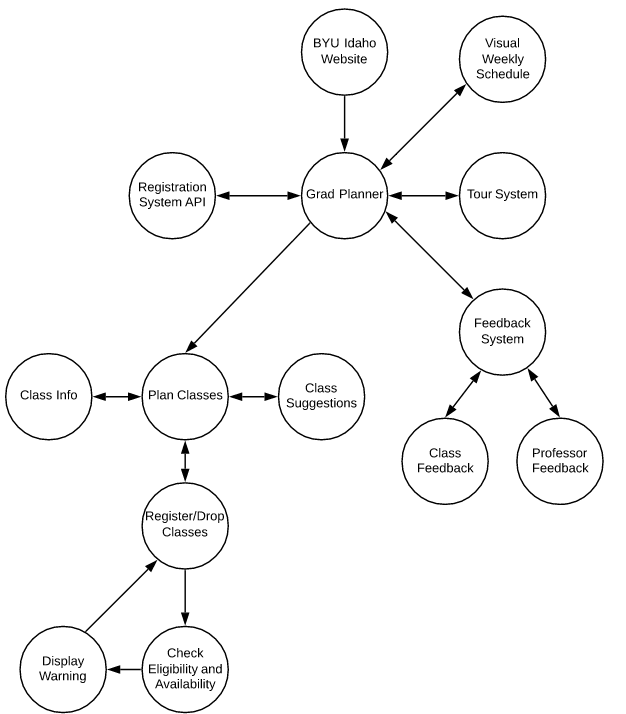


Figure 1- Functional Relationships

## User Characteristics

### Students

The primary audience for the Grad Planner Registration Integration will be the school's students. Although the system is owned and maintained by the university, it is students who will primarily be using the system. Because not all university students will have technical training or ability, ease-of-use is a crucial characteristic in the project’s design. The application will be simple enough that any student can use it without confusion. In addition, the application will not be too intrusive in the students’ workflow.

### Support Technicians

Support Technicians will be needed to assist students who struggle with the interface and who would not be able to register without this assistance. These people will most likely be advanced students who have been working with the planner for at least two semesters if not more. They will need access to the training documents for the software and any enhancement documents that have been completed.

### Faculty Advisors

These are the professors at school whose responsibility is to assist the students in planning their college careers. They will need to be able to access the student’s grad planner account and be able to guide the student through the process of selecting a degree or certificate program. They must also be able to assist the student in the registration process. They will be critical in helping the students get on the right path and have a good plan to advance in their programs.

### System Administrator

Daily use is expected by this user. A system administrator is expected to have an undergraduate degree in computer technology or similar field with industry certifications in computer fundamentals and operating systems. They have 5 or more years in a system support role. Their primary responsibility and privilege are seeing that the servers that host the app and customer accounts have high availability and uptime, but they will have little to no direct interaction with the grad planner application.

### Database Administrator

Daily to weekly interaction with the application's back end is determined by the number of database change requests. The Database Administrator (DBA) has a degree related to database administration or similar IT function and 3 or more years in a role managing the database of a large internet-enabled application. Their permissions shall enable them to update all aspects of the budget app and access all customer accounts. They will be trusted to delegate this authority to Customer Support and other IT workers. They will have very infrequent use of the front-end application.

## Operating Environment

Operating environment for the "Grad Planner Registration Integration":

* Informix Database
* REST API
* Windows, Mac, IOS, Android, Linux
* Chrome, Edge, Firefox
* SQL/Java/PHP

## Design and Implementation Constraints

* The global schema, fragmentation schema, and allocation schema.
* SQL commands for above queries/applications
* Combining various fragments.
* Implement the database using a centralized database management system.

## User Documentation

* User Virtual Walkthrough
* FAQ Webpage
* IT (Information Technology) Help Desk Number
* Administrative Operation Manual
* Assumptions and Dependencies
* The feedback system used to provide feedback for professors and classes that will be used is <https://www.ratemyprofessors.com/>.
* The Grad Planner Registration Integration will read and write from the BYU-I Registration System API.
* Holds from Ecclesiastical Endorsement, financial responsibility, or any other related subject will prevent users from registering for classes.

# External Interface Requirements

## User Interfaces

REQ-1: The Grad Planner Registration Integrationshall prompt the user to login upon accessing the application apart from the initial landing page. Login failure shall result in the application failing to load, prompting the user to insert their correct credentials.

REQ-2: The applicationshall be separated by major sections such as “Courses”, “Credits” and “Degree” as deemed necessary. 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.

REQ-3: The application shall interface with the data backend through a RESTful web service API. This API should enable a familiar experience across all supported platforms, including Web, Microsoft Windows, Mac, Linux, Android, and iOS. The data backend may be platform-agnostic but must provide a sustained performance of no more than 500-millisecond queries for no less than 90% of transactions in a 24-hour period.

## Hardware Interfaces

All server-side components shall be executed on server-class computers. All client-side components shall be executed on workstation-class, personal-class computers, and mobile devices.

## Software Interfaces

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.

The Grad Planner Registration Integration’s HTML files shall be hosted on BYU’s I-Learn server. The existing Registration System API shall be used to send and receive information.

## Communications Interfaces

All communication between the client and server will be served over HTTP Secure (HTTPS) protocol. The architecture shall follow the client-server model. API requests shall return information to the client as a JSON object. All API requests should be sent, received, and displayed within one second. User information and sensitive data transactions shall be encrypted to meet Advanced Encryption Standards (AES) guidelines. The application shall operate in a synchronized state with the registration system in that currently registered classes and declared degrees are consistent and accurate across both applications.

# System Features

## Grad Planner System Backend Interface

REQ-4: The grad planner backend software shall provide the ability to query and receive from the registration software planned course data (i.e., course code, course title, credit hours) for a specific user ID for the current registration period.

The query shall be able to be generated automatically at a scheduled time (i.e., the opening of the registration period).

The query shall be able to be generated on an ad hoc basis by user request.

REQ-5: The grad planner backend software shall provide the ability to query and receive from the registration database the class data (i.e., instructor, open seats, status, schedule, type) for all available classes for a specific course for the current registration period.

REQ-6: The grad planner backend software shall provide the ability to send the obtained class data to the grad planner frontend software.

## Overlapping Classes

The grad planner backend software shall provide the ability to send the obtained class data to the grad planner frontend software (REQ-6).

REQ-7 The system shall verify the student is securely logged in.

The login verification shall be Multifactor Authentication.

Password, Username, Duo Mobile passcode

REQ-8: The grad planner backend software shall have the student's current class schedule.

The backend software shall query for planned course data (REQ-4).

REQ-9: The grad planner backend software shall have the class time of the course for which the user is trying to register.

The backend shall send the obtained class data to the frontend software (REQ-6).

REQ-10: The grad planner backend software shall compare the possible overlapping course’s time to the times of all the student's registered courses.

The backend software shall query for planned course data (REQ-4).

REQ-11: If the course in question overlaps another course in the user’s current class schedule, a popup message shall be displayed to the user informing them of the overlapping class.

The backend software shall query for planned course data (REQ-4).

The backend software shall query for class data (REQ-5).

The backend shall send the obtained planned course data to the frontend software (REQ-6).

The backend shall send the obtained class data to the frontend software (REQ-6).

## Weekly Schedule

The backend shall send the obtained class data to the frontend software (REQ-6).

The system shall verify the student is securely logged in (REQ-7).

The backend shall have the student’s current class schedule (REQ-8).

REQ-12: On a portion of the page, the system shall display the student's current weekly schedule.

The backend shall send the obtained class data to the frontend software (REQ-8).

REQ-13: The weekly schedule shall display the following elements:

Current month

Date and day of the week for each day

Any user’s class assignments with a due date and time corresponding to the user’s time zone

For on-campus students, the course name, day, and time for each class

REQ-14: Each of the user’s registered courses shall be differentiated graphically on the weekly schedule display.

REQ-15: When the user clicks a course, the selected course's description shall be displayed.

The backend software shall query for planned course data (REQ-4).

The backend software shall query for class data (REQ-5).

The backend shall send the obtained planned course data to the frontend software (REQ-6).

The backend shall send the obtained class data to the frontend software (REQ-6).

REQ-16: The user shall have the ability to close out of the course description.

REQ-17: The user shall have the ability to export their schedule as an ICS file.

## Graduation Path

The backend software shall provide the ability to send the obtained class data to the grad planner frontend software (REQ-6).

The system shall verify the student is securely logged in (REQ-7).

REQ-18: The system shall have access to the user’s current declared degree and the user’s assigned track.

REQ-19: The system shall have access to the declared degree’s course requirements.

The backend software shall query for planned course data (REQ-4).

The backend software shall query for class data (REQ-5).

The backend shall send the obtained planned course data to the frontend software (REQ-6).

REQ-20: The system shall organize courses into years and semesters.

On and off tracks shall be differentiated graphically.

Completed semesters, current semesters, and future semesters shall be differentiated graphically.

The backend software shall query for class data (REQ-5).

Each semester shall have a list of completed or planned courses that occur during that semester.

The backend software shall query for class data (REQ-5).

Information about a course shall be displayed when the user clicks on the course.

REQ-21: The system shall display the user’s current path or if it is the user’s first time accessing the grad planner, the shortest path to graduation for the user’s declared degree.

The backend software shall query for class data (REQ-5).

The shortest path to graduation shall follow the user’s assigned track.

The shortest path to graduation shall follow the credit hours load recommended by the department of the user’s declared degree.

REQ-22: The user shall be able to reset their graduation path to the shortest path to graduation via a graphical element.

The backend software shall query for planned course data (REQ-4).

The backend software shall query for class data (REQ-5).

The backend shall send the obtained planned course data to the frontend software (REQ-6).

REQ-23: Graduation requirements that can be fulfilled by multiple courses shall be marked with a small, but visible, icon. The user shall be able to click this icon to view and select any of the possible courses that fulfill the requirement.

The backend software shall query for class data (REQ-5).

REQ-24: The user shall be able to click and drag a course to take place in a different semester.

REQ-25: Changes that the user makes to the schedule shall be saved in the database.

The backend shall send the obtained planned course data to the frontend software (REQ-6).

## Feedback for courses and instructors

The backend shall send the obtained planned course data to the frontend software (REQ-6).

The system shall verify the student is securely logged in (REQ-7).

REQ-26: 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

When the user clicks the element described in (REQ-24) the system will display a modal window with the feedback information:

REQ-27: The interface shall display an aggregate score based on previous feedback for a course or instructor.

REQ-28: The user shall have access to the comments from other students about the class or instructor.

The system shall provide the user a graphical element to access a feedback form.

REQ-29: The feedback form shall provide the user the option of selecting either instructor or course feedback.

REQ-30: The user shall have the ability to rate his/her instructor using a numerical scale.

REQ-31: The user shall have the ability to rate the class using a numerical scale.

REQ-32: The user shall have the ability to leave a comment about his/her experience with the class.

REQ-33: The feedback form shall provide users the option of disclosing identity but shall retain the student’s anonymity if the option is not selected.

## User Interface

REQ-34: Each of the system features shall be designed in boxes that the system could display every feature in small, medium, and screen sizes.

## Optional Tour

The backend shall send the obtained planned course data to the frontend software (REQ-3).

The system shall verify the student is securely logged in (REQ-5).

REQ-35: Once a user accesses the Grad Planner Registration Integration the website will look to see if the user has a cookie that would be deposited if the user has previously accessed the Grad Planner Registration Integration site.

If the user has been determined to have accessed the site, the user will be displayed the original Grad Planner Registration Integration site.

If the user has been determined to have not accessed the site, the Grad Planner Registration Integration site will be loaded but in an un-editable state with a modal \*Yes/No box prompting the user “Would you like to take the tour of the Grad Planner.”

If the user chooses “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” the Yes/No box will disappear, leaving the Grad Planner Registration Integration site accessible to the user. 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.

REQ-36: If the user has decided to not take the tour, there shall be a button to take the tour in the Grad Planner Registration Integration settings.

REQ-37: 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.

## Performance Requirements

REQ-38: The UI: Sub-Second

The user shall select an action on the web page to load another page or update the current page.

The UI shall send the start time of reload / new web page to the Backend.

REQ-39: The Backend: Sub-Second

The UI shall send the start time of reload / new web page.

The Backend shall record the start time and it sends the request for information or begins the load of a new page.

The Backend records how long from the time the request was received to when the data was retrieved, or a new web page is loaded.

The Backend shall write the total time it took since receiving the request to reload or build a new web page.

The Backend shall send both the UI start time and Backend total time to the UI.

If the UI fails to send start time, then a default time for UI to let the system know the UI part cannot be calculated.

If Backend fails, then a flag is added to total Backend time so that we can distinguish it from a successful response.

REQ-40: The UI: Final Sub-Second

The UI shall take the start time and Backend total time returned from Backend and the UI sends an asynchronous call to the Backend with the start time, Backend total time, and the current UI ending time.

The Backend records the asynchronous call from the UI.

The system shall be monitored daily, so that all page requests that had, UI start time, Backend total time, and UI ending time. This shall make verification of 95% sub-second response is being achieved and problem pages to be modified for speed.

The system shall be monitored daily so that all page requests that had missing times or errors in the request. This way the patterns can be tracked, and issues resolved.

The user shall navigate to different pages in the grad planner 95% of the time within a sub-second.

## Classes Display and Filtering

REQ-41: The grad planner UI shall display the required classes corresponding to the declared major of a plan they are viewing, including:

Course name.

Course code.

Brief description of the class.

Number of seats available out of total seats.

Selection of professors who are teaching the class.

Day and time options available for the class.

REQ-42: The Grad Planner Class Filter: Online

The user selects a filter checkbox for Online classes and then searches classes.

The UI shall send a flag to the Backend that says only return classes that have the Online flag.

The Backend query shall only return classes that have the Online flag in the database, that was received from the UI.

If the UI does not receive correctly formatted classes in the returned dataset, then it will display a user message, that it failed to return classes.

If the Backend does not receive a correctly formatted request, then the query isn’t performed, and an error is sent to the UI.

If the Backend does not receive a valid dataset from the query, then an error is sent to the UI.

REQ-43: The Grad Planner Class Filter: Daytime (8:00 am – 3:00 pm)

The user selects a filter checkbox for Daytime classes and then searches classes.

The UI shall send a flag to the Backend that says only return on-campus classes that have the Daytime flag.

The Backend query shall only return classes that have the Nighttime flag in the database, that was received from the UI.

If the UI does not receive correctly formatted classes in the returned dataset, then it will display a user message, that it failed to return classes.

If the Backend does not receive a correctly formatted request, then the query is not performed, and an error is sent to the UI.

If the Backend does not receive a valid dataset from the query, then an error is sent to the UI.

REQ-44: The Grad Planner Class Filter: Nighttime (3:01 pm – 10:00 pm)

The user selects a filter checkbox for Nighttime classes and then searches classes.

The UI shall send a flag to the Backend that says only return on-campus classes that have the Nighttime flag.

The Backend query shall only return classes that have the Nighttime flag in the database, that was received from the UI.

If the UI does not receive correctly formatted classes in the returned dataset, then it will display a user message, that it failed to return classes.

If the Backend does not receive a correctly formatted request, then the query isn’t performed, and an error is sent to the UI.

If the Backend does not receive a valid dataset from the query, then an error is sent to the UI.

## Automatic Registration

REQ-45: The Automatic Registration Backend software shall display restrictions and holds on the user’s account which shall prohibit a user from being automatically registered for future classes. Examples of the following would be:

Financially responsibility unfulfilled.

Emergency contacts are not filled out or updated.

Ecclesiastical Endorsement has not been completed.

REQ-46: The Automatic Registration Backend software shall provide the ability to query the database to verify if the user has enabled automatic registration.

REQ-47: The Automatic Registration Backend software will send a query to the database to check the open or closed status of a class.

REQ-48: The user shall have the ability to enable and disable Automatic Registration with the click of a switch.

## Class Information Display

REQ-49: The UI shall send the selected class to the Backend to retrieve information of the class.

REQ-50: The Backend system shall provide the UI with data about the open class:

Course name.

Course code.

Brief description of the class.

Number of seats available out of total seats.

Selection of professors who are teaching the class.

Day and time options available for the class.

REQ-51: The UI shall show information on a class when the user is registering for an open class, including:

Course name.

Course code.

Brief description of the class.

Number of seats available out of total seats.

Selection of professors who are teaching the class.

Day and time options available for the class.

REQ-52: The UI shall provide a link to the specific information of the class.

# Other Nonfunctional Requirements

## Performance Requirements

REQ-53: Queries and responses from the Registration API shall be handled and displayed to the user in an appealing format in under 1 second.

REQ-54: Reliability must remain consistent regardless of browser.

REQ-55: The system must be capable of handling the student body of BYU-Idaho.

REQ-56: The system must be able to scale to handle an increase in the student body at BYU-Idaho.

REQ-57: The system must perform as expected across the platform.

## Security Requirements

REQ-58: Set cookies after authenticating the user.

REQ-59: 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.

REQ-60: HTTP will not be supported. Only HTTPS.

REQ-61: Ensure that all user data is encrypted in the database.

REQ-62: Secure user data when making calls to the API.

REQ-63: Secure user data when receiving data from the API.

REQ-64: Employ the use of data filtering for the user input.

## Software Quality Attributes

Usability

To use the program, prior training by qualified personnel will not be necessary. The program will include a touring tool for new students who are familiar with the program, however**,** it is expected that at least 95% of students can use the program without difficulties. Those students with low computer skills may see an academic advisor for additional help.

Reliability

By default, the program will generate the fastest plan for grading students who use the tool, however, students will be able to create a new plan or make changes to the automatic plan according to their needs. Because the Registration System Informix Database provides information about classes, students can be sure that their information will be up to date.

Portability

Because the Grad Planner Registration Integration is a web application, it will be available in the main browsers such as Google Chrome, Mozilla Firefox, Safari, Internet Explorer and Opera and android browser, in such a way that this application can be used on devices with the following operating systems:

* Mac OS
* iOS
* Windows
* Android
* Linux

Its responsive design will make the application easy to use on cell phones, tablets, and computers.

### Maintainability

The Grad Planner Registration Integration will update according to changes and updates in BYU courses and programs. This includes creating new courses, new teachers, changes in class schedules and requirements.

This data will be updated through the BYU course Registration System’s Informix Database API, which will provide the names of the courses, teachers, classes, schedules, and requirements through a JSON object. Programmers should only make changes to the code for the creation of new courses and degrees.

### Reusability

Since popular web programming languages such as JavaScript, HTML, and CSS will be used to create the program, we estimate that at least 70% of the code will be reusable. This will allow other institutions to implement the software and reduce costs during the implementation process.

In case other institutions want to implement our software, our developers will make compatibility evaluations with the databases of other educational intuitions for their implementation.

### Testability

Each section of the code will be evaluated in a validation process and verification process. During the validation process, the developers will verify the quality of the code with considering aspects such as:

* Code size
* Loading times
* Efficiency

During the verification process, the necessary tests will be carried out to demonstrate that the program meets the requirements described in this document (sections 3 “External interface Requirements”).

## Business Rules

REQ-65: The system shall indicate the class status (Open/Closed/Reopened/Waitlist)

REQ-66: The system shall indicate if a class is taken, currently taking, or planned to take, showing the class code and name in assorted colors.

REQ-67: The system shall restrict a register for a class in the following situations:

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

REQ-68: The system shall show a message with an error in the following cases:

If the user plan to take more than 140 credits

If the user has planned to complete the career with all the required classes

REQ-69: When the user opens the system for the first time, the system shall:

Show the beginning screen with labels showing each of the system functionalities. This is will be the tutorial with instructions for inexperienced users.

Organize the classes in separate boxes for each semester: The classes shall be automatically organized depending on the declared major

# Other Requirements

## Appendix A: Glossary, Acronyms, and abbreviations

|  |  |
| --- | --- |
| Keywords | Definition |
| API | An application programming interface (API) is a computing interface that defines interactions between multiple software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. |
| RESTful | Representational state transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services. Web services that conform to the REST architectural style, called RESTful Web services, provide interoperability between computer systems on the Internet. RESTful Web services allow the requesting systems to access and manipulate textual representations of Web resources by using a uniform and predefined set of stateless operations. |
| HTTPS | Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL). |
| N/A | Not applicable |
| ICS | A media type which allows users to store and exchange calendaring and scheduling information such as events, to-dos, journal entries, and free/busy information. |

## Appendix B: List of Figures

[Figure 1- Functional Relationships 13](#_Toc41763542)