**CMSC 495 7383**

**Capstone Computer Science (2242)**

Professor Jeff Sanford

**Project Plan – Student Information Records System (SIRS)**

**Team Calculating How Not to Fail (CHNF)**

Coleman, Deven

Kardash, Mark

Kelley, Kendle

Ramirez, Arturo

Ross, Marcel

Versions History

|  |  |  |
| --- | --- | --- |
| Version | Change Description | Date |
| 1.0 | Initial submission for Project Plan | 03/26/2024 |
| 1.1 | Project plan updated to include Test Plans and User Guides | 04/02/2024 |
| 1.2 | New use cases and Database design diagrams | 04/09/2024 |
| 1.3 | Created a Requirement Traceability Martic, new appendices for test reports, contribution reports, and references. Updated Table of Contents. | 04/16/2024 |
| 1.4 | Updated the Requirement Traceability Martic, added new test results, contribution reports | 04/23/2024 |
| 1.5 | Updated GUI Test Cases, and Test Results | 04/30/2024 |
| 1.6 | Final submission | 05/7/2024 |

Project Plan

**TABLE OF CONTENTS**

[A.1 Project Introduction 4](#_Toc165986183)

[A.1.1 Introduction 4](#_Toc165986184)

[A.1.2 Overview 4](#_Toc165986185)

[A.1.3 Objectives 4](#_Toc165986186)

[A.1.4 Key Features 4](#_Toc165986187)

[A.2 Project Plan Description 5](#_Toc165986188)

[A.2.1 Description 5](#_Toc165986189)

[A.3 Scope 5](#_Toc165986190)

[A.3.1 Scope Management 5](#_Toc165986191)

[A.3.2 Scope Change 6](#_Toc165986192)

[A.4 Requirements 6](#_Toc165986193)

[A.4.1 Requirements Process 6](#_Toc165986194)

[A.4.2 Requirements Gathering 6](#_Toc165986195)

[A.5 Schedule 7](#_Toc165986196)

[A.5.1 Schedule 7](#_Toc165986197)

[A.5.2 Schedule Monitoring 9](#_Toc165986198)

[A.5.3 Monitoring and Reporting 9](#_Toc165986199)

[A.5.4 Schedule Tools and Outputs 9](#_Toc165986200)

[A.6 Resources 9](#_Toc165986201)

[A.6.1 Roles and Responsibilities 9](#_Toc165986202)

[A.7 Deliverables 10](#_Toc165986203)

[A.7.1 Deliverable List 10](#_Toc165986204)

[A.7.2 Requirements Review 10](#_Toc165986205)

[A.8 User Guide 11](#_Toc165986206)

[A.8.1 Purpose: 11](#_Toc165986207)

[A.8.2 Section 1: Installation 12](#_Toc165986208)

[A.8.3 Section 2: User Interface 13](#_Toc165986209)

[A.8.4 Section 3: SIRS Functions 14](#_Toc165986210)

[A.9 Test Plan 23](#_Toc165986211)

[A.9.1 Overview 23](#_Toc165986212)

[A.9.2 Testing Resource 23](#_Toc165986213)

[A.9.3 Types of Testing 23](#_Toc165986214)

[A.9.4 Test Case Template 24](#_Toc165986215)

[A.9.5 Test Objectives 25](#_Toc165986216)

[A.9.6 Test Case Scenarios 25](#_Toc165986217)

[A.10 System Design 27](#_Toc165986218)

[A.10.1 UML Class Diagram 27](#_Toc165986219)

[A.11 Appendix B. References 28](#_Toc165986220)

[A.12 Appendix C. Contribution Report 29](#_Toc165986221)

Project Introduction

Introduction

The Student Information Records System (SIRS) will streamline the process of managing grades by providing a centralized platform to input, track, and calculate grades for all assignments and discussions in a course. This saves time and effort for students who would otherwise need to manually calculate their grades or use multiple tools.

Overview

A simple application for UMGC students to be able to add all their gradable assignments and discussions from a course and then input the grades they received on the ones they have completed. Then, the user will be given an average grade they must receive on the rest of their assignments to finish the course with a selectable desired overall grade.

Users can set a desired overall grade for their course and monitor their progress towards achieving it. SIRS calculates the average grade needed on remaining assignments to meet the desired overall grade, empowering students to set realistic goals and work towards them effectively.

Objectives

* To develop a simple application for UMGC (University of Maryland Global Campus) students.
* Allow users to add all gradable assignments and discussions from a course.
* Enable users to input grades received on completed assignments.
* Calculate the average grade needed on remaining assignments to achieve a desired overall grade.

Key Features

* Easy-to-Use GUI to manage assignments and discussion grades.
* Accurate Assignment Weighting
* Predictive Grades
* Save/Load Feature (objective)
* Selectable Goal Final Grade adjusts predictive grades (objective)
* UMGC Grade Page Import Functionality (objective)
* Web and mobile accessibility application (objective)

Project Plan Description

Description

This Project Plan will be used to plan, manage, track, and evaluate SIRS performance. This plan includes a description of how we will conduct project administration, operations, and tracking. Team CHNF will collaborate closely with UMGC teaching staff to assess, plan, and execute the activities required to effectively perform all objectives for this project and course. Team CHNF members will operate within an 8-week quarter for rapid development and project assignments will be submitted weekly as instructed in the syllabus. Activities include the following:

* Understand and validate SIRS commitments.
* Gain a common understanding of key items, such as scope and deliverables.
* Establish the roles and responsibilities, communication management, and document change control.
* Collaborative development and shared understanding of project processes and plans, such as a Work Breakdown Structure (WBS), schedule, Unified Modeling Language (UML) diagrams tools.

At the conclusion of the project, Team CHNF will demonstrate the ability to design, develop, test, implement, and document in a current computer science application area. Additionally, Team CHNF will architect a reusable and scalable application using the most current technologies and approaches.

Scope

Scope Management

Scope management for the SIRS project will be the primary responsibility of the PM. Proposed scope changes may be initiated by the PM, professor, or any team member of the project team. All change requests will be submitted to the PM who will then evaluate the requested scope change. Upon acceptance of the scope change request the PM will follow a change control process. Upon approval of scope changes, the team will update all project documents and communicate the scope.

Various factors can cause a change to the scope of the program and various projects. Team CHNF will work closely to manage the scope and address any needed changes as they are discovered. The following are how scope changes will be identified during execution of the project:

* Refine the scope to a greater level of detail in conjunction with the intended client needs.
* Review all results of project meetings or review sessions with the client to determine if a scope change has been required.
* Include a roundtable to identify scope changes at all program and project team meetings.
* Assess the results of all deliverables reviews and approval cycles to determine if a scope change is required.

Scope Change

1. When the Team CHNF identifies a change of scope, a Scope Change will be assessed and be completed by the PM.
2. The PM will assign someone from the team to update the project schedule and review impact.
3. The PM will provide the team with a date by which a decision is required.
4. If Team CHNF approves the change, if applicable, the PM will revise the project plans to incorporate it.
5. If Team CHNF does not approve the change, the Change Request will be deleted, and no further work will be done on it.

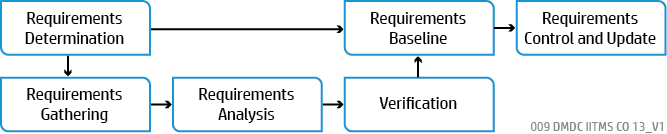
For areas where the scope of work is not quantified or is undefined in the course syllabus and/or as additional requirements in these or other new scope areas outside of the course syllabus become available, Team CHNF will determine the best way to address these requirements and develop a mutually agreeable plan to perform the work.

Requirements

Requirements Process

Requirements Management is a systematic approach to identifying, organizing, communicating, and managing the changing requirements of a system. SIRS will establish high-level processes for Requirements Management process for the project. Figure 1 presents the Requirements Management process.

Figure 1. Requirements Management Process



Requirements Management process is used to determine complete and accurate requirements prior to starting activities to meet the requirements.

Requirements Gathering

* Team CHNF will develop a simple application for University of Maryland Global Campus (UMGC) students.
* The application will allow users to add Gradable Assignments and discussions from a course (threshold).
* The application will allow users to input grades received on completed assignments (threshold).
* The application will calculate the average grade needed on remaining assignments to achieve a desired overall grade based on assignment weights (threshold).
* The application will provide a user-friendly interface for adding assignments and inputting grades (threshold).
* The application will include fields for assignment names, grades received, and possible grades (threshold).
* The application will allow users to save and load course grade data sets (objective).
* The application will allow users to import grades directly from the UMGC Course page (objective).
* The application will allow Users to select their desired overall grades (objective).
* The application will have support for web or mobile platforms (objective).
* The application will ensure compatibility with commonly used browsers and devices (objective).

Schedule

Schedule

Development of a project schedule is a key basis for developing resource and delivery estimates. The planned schedule indicates the individual activities that need to be completed, supports understanding planned completion dates of milestones, and forms the basis for understanding the status of a project. SIRS has provided a draft schedule shown in Table 1.

Table 1. Draft Schedule for SIRS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WBS | Task Name | Duration | Start | Finish |
| **~~1~~** | **~~Team Formation~~** | **~~7 Days~~** | **~~Wed 3/13/24~~** | **~~Tue 3/19/24~~** |
| ~~1.1~~ | ~~Member Assignment and Organization~~ | ~~7 days~~ | ~~Wed 3/13/24~~ | ~~Tue~~  ~~3/19/24~~ |
| **~~2~~** | **~~Project Initialization and Design~~** | **~~21 days~~** | **~~Wed 3/20/24~~** | **~~Tue 4/09/24~~** |
| **~~2.1~~** | **~~Determine Project Scope~~** | **~~14 days~~** | **~~Wed 3/20/24~~** | **~~Tue 3/26/24~~** |
| ~~2.1.1~~ | ~~Member Pitches~~ | ~~2 day~~ | ~~Wed 3/20/24~~ | ~~Fri 3/22/24~~ |
| ~~2.1.2~~ | ~~Team Vote~~ | ~~1 day~~ | ~~Fri 3/22/24~~ | ~~Sat 3/23/24~~ |
| ~~2.1.3~~ | ~~Member Role Determination~~ | ~~1 day~~ | ~~Sat 3/23/24~~ | ~~Sun 3/24/24~~ |
| ~~2.1.4~~ | ~~Draft Project Plan~~ | ~~3 day~~ | ~~Sun 3/24/24~~ | ~~Tue 3/26/24~~ |
| ~~2.1.5~~ | ~~Draft User Guide~~ | ~~7 days~~ | ~~Wed 3/27/24~~ | ~~Tue 4/02/24~~ |
| ~~2.1.6~~ | ~~Draft Test Plan~~ | ~~7 days~~ | ~~Wed 3/27/24~~ | ~~Tue 4/02/24~~ |
| **~~2.2~~** | **~~Project Design~~** | **~~7 days~~** | **~~Wed 4/03/24~~** | **~~Tue 4/09/24~~** |
| ~~2.2.1~~ | ~~Draft Database Design~~ | ~~5 days~~ | ~~Wed 4/03/24~~ | ~~Sun 4/07/24~~ |
| ~~2.2.2~~ | ~~UI Mockups~~ | ~~5 Days~~ | ~~Wed 4/03/24~~ | ~~Sun 4/07/24~~ |
| ~~2.2.3~~ | ~~Draft ERD, UML Class Diagrams, Use Cases~~ | ~~2 days~~ | ~~Sun 4/07/24~~ | ~~Tue 4/09/24~~ |
| **~~3~~** | **~~Project Development~~** | **~~21 days~~** | **~~Wed 4/10/24~~** | **~~Tue 4/30/24~~** |
| **~~3.1~~** | **~~Initial MVP Development~~** | **~~7 Days~~** | **~~Wed 4/10/24~~** | **~~Tue 4/16/24~~** |
| ~~3.1.1~~ | ~~MVP UI/Frontend Development~~ | ~~7 Days~~ | ~~Wed 4/10/24~~ | ~~Tue 4/16/24~~ |
| ~~3.1.2~~ | ~~Threshold Draft Database/Backend Development~~ | ~~7 Days~~ | ~~Wed 4/10/24~~ | ~~Tue 4/16/24~~ |
| **~~3.2~~** | **~~MVP Testing~~** | **~~4 Days~~** | **~~Wed 4/17/24~~** | **~~Sun 4/21/24~~** |
| ~~3.2.1~~ | ~~Use Case Verification~~ | ~~1 Day~~ | ~~Wed 4/17/24~~ | ~~Thur 4/18/24~~ |
| ~~3.2.2~~ | ~~Test Case 1-12 Testing~~ | ~~2 Days~~ | ~~Thur 4/18/24~~ | ~~Sat 4/20/24~~ |
| ~~3.2.3~~ | ~~MVP Testing Report~~ | ~~1 Day~~ | ~~Sat 4/20/24~~ | ~~Sun 4/21/24~~ |
| **~~3.3~~** | **~~Test Report Adjudication~~** | **~~3 Days~~** | **~~Sun 4/21/24~~** | **~~Tue 4/23/24~~** |
| ~~3.3.1~~ | ~~Post-Test Fix Planning~~ | ~~3 Days~~ | ~~Sun 4/21/24~~ | ~~Tue 4/23/24~~ |
| ~~3.3.2~~ | ~~Post-Test Fix Execution~~ | ~~3 Days~~ | ~~Sun 4/21/24~~ | ~~Tue 4/23/24~~ |
| **~~3.4~~** | **~~Final Program Development~~** | **~~4 Days~~** | **~~Wed 4/24/24~~** | **~~Sat 4/27/24~~** |
| ~~3.4.1~~ | ~~Frontend Development~~ | ~~4 Days~~ | ~~Wed 4/24/24~~ | ~~Sat 4/27/24~~ |
| ~~3.4.2~~ | ~~Threshold Draft Database/Backend Development~~ | ~~4 Days~~ | ~~Wed 4/24/24~~ | ~~Sat 4/27/24~~ |
| ~~3.5~~ | **~~Final Program Testing~~** | **~~3 Days~~** | **~~Sat 4/27/24~~** | **~~Tue 4/30/24~~** |
| ~~3.5.1~~ | ~~Test Case 1-12 Testing~~ | ~~2 Days~~ | ~~Sat 4/27/24~~ | ~~Mon 4/29/24~~ |
| ~~3.5.2~~ | ~~Final Testing Report~~ | ~~1 Day~~ | ~~Wed 3/11/24~~ | ~~Tue 4/30/24~~ |
| **~~4~~** | **~~Lessons Learned~~** | **~~7 Days~~** | **~~Wed 5/01/24~~** | **~~Tue 5/07/24~~** |

Schedule Monitoring

Schedule control will be accomplished using Trello and associated tools by the PM.

Monitoring and Reporting

Weekly reporting plans will be established to monitor and track resources and execution, and other relevant schedule elements.

Schedule Tools and Outputs

The Scheduling Tool, Trello, is used for all scheduling needs, Gantt Charts, progress reports, and resource reports, if required.

Resources

Roles and Responsibilities

Table 2. Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| Role | Responsibility | Resource |
| Project Manager | Conducts overall management of the project; Creates and defines team roles; Sets timelines for the team; Ensures requirements are correctly defined; Creates and manages critical schedules; Conducts QA/QC on team member’s work; Manages the GitHub repository; Resolves team conflicts; Creates and manages Project Plan document | Primary: Kendle Kelley  Secondary: Arturo Ramirez |
| Frontend Developer | Creates all program code for implementation of the project UI functionality; Creates and manages interfaces for backend (data) functionality; Interfaces with test engineer to refine frontend tests; Interfaces with UI designer to ensure correct functionality; Interfaces with documentation manager to ensure functionality is captured | Primary: Marcel Ross  Secondary: Deven Coleman |
| Backend Developer | Creates all program code for management and manipulation of data; Creates and manages interfaces for frontend (UI) functionality; Interfaces with test engineer to refine backend tests; Interfaces with documentation manager to ensure functionality is captured | Primary: Mark Kardash  Secondary: Kendle Kelley |
| UI Designer | Drafts and finalizes designs for UI; Interfaces with developers to support UI design; Ensures UI meets requirements; Resolves issues and conflicts between UI design and implementation | Primary: Marcel Ross  Secondary: Kendle Kelley |
| Testing Engineer | Designs comprehensive tests for each project requirement; Creates Test Plan document; Conducts tests to ensure project meets requirements; Captures test results in official documentation; Interfaces with team on test results | Primary: Deven Coleman  Secondary: Arturo Ramirez |
| Documentation Manager | Creates all official support documentation for project; Creates and manages the User Guide; Interfaces with team to capture essential information within documentation; Supports GitHub repository maintenance and documentation | Primary: Arturo Ramirez  Secondary: Kendle Kelley |

Deliverables

Deliverable List

Student Information Records System (SIRS) will submit reports, documents, and deliverables following CMSC 495 guidance, maintaining a high-quality standard regarding accuracy, clarity, valid specifications, and appropriate format. Submission of all documents will be on time on or before the due date as specified in Table 3.

Table 3. Deliverables List

|  |  |
| --- | --- |
| Deliverable | Due Date |
| ~~Project Plan~~ | ~~March 26, 2024~~ |
| ~~Users Guide and Test Plan~~ | ~~April 2, 2024~~ |
| ~~Peer Reviews #1~~ | ~~April 2, 2024~~ |
| ~~Project Design~~ | ~~April 9, 2024~~ |
| ~~Phase 1 Source~~ | ~~April 16, 2024~~ |
| ~~Phase 2 Source~~ | ~~April 23, 2024~~ |
| ~~Peer Reviews #2~~ | ~~April 23, 2024~~ |
| ~~Phase 3 Source~~ | ~~April 30, 2024~~ |
| ~~Peer Reviews #3~~ | ~~May 7, 2024~~ |
| ~~Final Project~~ | ~~May 7, 2024~~ |

Requirements Review

With the completion of the SIRS Prototype, the PM and Documentation Managers have reviewed the defined Requirements from Section A.4.2 and graded Team CNHF’s implementation of threshold and objective requirements. This grading can be found in Table 4.

Table 4. Requirements Review

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Threshold/Objective** | **Achievement Grading** |
| Team CHNF will develop a simple application for University of Maryland Global Campus (UMGC) students. | Threshold | Achieved |
| The application will allow users to add Gradable Assignments and discussions from a course. | Threshold | Achieved |
| The application will allow users to input grades received on completed assignments. | Threshold | Achieved |
| The application will calculate the average grade needed on remaining assignments to achieve a desired overall grade based on assignment weights. | Threshold | Achieved |
| The application will provide a user-friendly interface for adding assignments and inputting grades. | Threshold | Achieved |
| The application will include fields for assignment names, grades received, and possible grades. | Threshold | Achieved |
| The application will allow users to save and load course grade data sets. | Objective | Achieved |
| The application will allow users to import grades directly from the UMGC Course page. | Objective | Not Achieved |
| The application will allow Users to select their desired overall grades. | Objective | Achieved |
| The application will have support for web or mobile platforms. | Objective | Not Achieved |
| The application will ensure compatibility with commonly used browsers and devices. | Objective | Not Achieved |
| The application will allow users to delete previously added assignments | Objective (added via Change Request) | Achieved |

User Guide

Purpose:

**A Video Guide is available on YouTube:** <https://www.youtube.com/watch?v=R1o3ff7zR8M> This Student Information Records System (SIRS) User Guide is intended to give users easy to follow instructions on how to build/install SIRS and operate the program’s various features. *Note:* This static Users Guide will be updated with every major release. For all minor releases and prerelease candidates, dynamic updates can be found at the SIRS Project GitHub wiki located at <https://github.com/UselessFodder/umgc_SIRS/wiki> .

Section 1: Installation

**Step 1**: Check if you have Java install on your machine by opening command prompt, PowerShell, bash CLI and entering the following command:

Java -version

**Note:** if you do not have it, please go to <https://www.java.com/en/download/> and download to your machine.

**Step 2**: Install Eclipse IDE

a. Visit the link to install software, <https://www.eclipse.org/downloads/packages/installer>

b. Click on the link for your platform on the right side of the screen (under “Download”) and unzip the downloaded file to get a file called “Eclipse Installer

c. Run the Eclipse Installer. Select “Eclipse IDE for Java Developers” and then “Install” to install Eclipse.

d. Launch Eclipse.

e. The first time you run Eclipse; you need to choose a workspace (a folder that stores your projects).

**Step 3**: Setting up SIRS in Eclipse

a. Download the SIRS zip and unzip the files.

b. Open Eclipse and open the Java Workspace created above.

c. Import the SIRS project.

d. Press File –> Import

e. Then scroll down to Maven then click Existing Maven Projects.

f. In the popup, press Browse, and navigate to the unzipped SIRS downloaded. Choose that folder.

**Step 4**: Running the project.

a. To run your project code, press the small dropdown arrow next to the green play button.

A screenshot of a computer

Description automatically generated

Section 2: User Interface

Components consist of the following editable fields:

a. **Class Name**: Enter text and numbers to describe your class name.

b. **Assignment**: Enter text and numbers to describe your assignment name.

c. **Grade Received**: Enter the raw number score received on the specific task.

d. **Possible Grade**: Enter the total possible number score that can be achieved on this task.

e. **The Future Assignment Checkbox**: Allows you to enter the assignment without a grade received score.

f. **Add Assignment Button**: Once all fields above the button are fields you may click this button to add the assignment to the course.

g. **Calculate Grade**: Pop up window returns your current grade based on the assignment entered and notifies you if you are meeting your desired grade target.

h. **Save Data**: This allows you to save your data inputs into a dat file.

i. **Load Data**: This feature allows you to load inputs into the application to calculate grades.

j. **Delete Last Assignment**: The feature allows you to remove the most recent assignment added.

k. **Desired Overall Grade Dropdown**: This dropdown allows you to select a desired grade you hope to achieve in a specific course. This value is used by other components in the program to notify you if you will meet your target grade.

***A screenshot of a computer

Description automatically generated***

Section 3: SIRS Functions

This section will give detailed instructions on each major SIRS function with supporting graphics.

**Subsection a: Creating a New Course with Multiple Assignments**

**Step 1**: Enter required fields:

a. Class name

b. Assignment name

c. Grade Received

d. Possible Grade

**Step 2**: Click Add assignment.

**Step 3**: Repeat step 1 through 2 as many assignments as you would like to enter.

Your display should look like the example below:

Figure 1. Add New Course with Multiple Assignments

*A screenshot of a computer

Description automatically generated*

**Subsection b: Calculate Grade**

**Step 1:** Click the Calculate Grade Button

**Step 2**: Read response below for Calculate Grade Results prompt, base on the previous example in subsection a.

A screenshot of a computer screen

Description automatically generated

**Subsection c: Add Future Assignment**

**Step 1**: Enter another assignment with the following fields:

a. Class name

b. Assignment name

c. Possible Grade

**Step 2**: Click checkbox Future Assignment. See below for GUI before clicking Add Assignment.

A screenshot of a computer

Description automatically generated

**Step 3**: Click Add Assignment. See below for the future assignment added to the centered display field.

Note: There is an \* as a place holder for that specific assignment Grade Received and percentage score.

A screenshot of a computer

Description automatically generated

**Subsection d: Adjusting Desired Grade**

Step 1: Select the desired letter grade you would like to achieve in the course from the dropdown as seen below:

A screenshot of a computer

Description automatically generated

**Step 2:** Select A

**Step 3:** Click Calculate Grade and you should receive a prompt like in subsection b, Step 2. The display will like the GUI below:

A screenshot of a computer error

Description automatically generated

**Step 4**: Change Desire Grade to B from the dropdown.

**Step 5**: Click Calculate Grade and see new prompt with the new values needed to achieve new desired grade.

A screenshot of a computer error

Description automatically generated

**Subsection e: Saving a Course**

Step 1: Save course and assignment input by clicking Save Data.

A screenshot of a computer

Description automatically generated

Step 2: Select the Save\_Test.dat file and click save.

A screenshot of a computer error

Description automatically generated

Step 3: Click the close button

**Subsection f: Loading a Course**

Step 1: Launch the application.

Step 2: Click Load Data Button.

Step 3: Select the Save\_test.dat.dat file.

A screenshot of a computer

Description automatically generated

**Step 4:** Observe as the center display box will populate with the 4 test saved from subsection e.

**A screenshot of a test

Description automatically generated**

**Subsection f: Delete Last Assignment**

Step 1: Create a course with more than one assignment. Below is a GUI with all assignments entered.

A screenshot of a computer

Description automatically generated

Step 2: Click Delete Last Assignment button and watch the last assignment removed from the display box.

A screenshot of a computer

Description automatically generated

Test Plan

Overview

The purpose of the test plan is to communicate our testing approach to the project. The test plan defines the testing methods that will be used for verifying the Student Information Records System (SIRS) Releases throughout development and test phases.

The Test Plan provides a road map to:

* Who participates?
* What is to be tested?
* How testing will be conducted and recorded?

Testing Resource

Provides a brief description of the testing resources used during the different states of testing. As a minimum the following resources are included:

* Project Manager – The project manager ensures tasks are successfully completed according to the project schedule, including unit test, Quality Assurance (QA) test and User Acceptance Testing activities.
* Lead Tester – The lead tester defines the test plan and testing standards. Leads the development of all test cases used for QA testing. Oversees, manages, and reports on the testing resources, testing environments, as well as test execution and testing deliverables.
* Testers – The testers develop system test cases, execute QA testing, including functional and regression testing, and report the results of test execution.
* Developers – As part of the development phase, developers perform unit tests on the impacted modules. Developers also perform any necessary defect resolution during testing activities.

Types of Testing

The team will perform at minimal the following test below. Other tests may be performed at the discretion of the developers.

Table 4. Test Types

|  |  |  |
| --- | --- | --- |
| Test Type | Description | Owner |
| Unit | Confirms that the program logic within a module produces the expected output when given a known input. Tells the developer that the code is doing things right and ensures that the smallest testable components of an application perform a specific task successfully. | Developers |
| Functional | Confirms proper inter-operation with all prior software deliverables, interfaces, and other components. Validates that all the feature functions of SIRS work in unison. Validates that the release requirements work as expected. | Testers |
| Regression | Ensures that changes to the system’s software, hardware, or environment do not negatively impact unchanged areas of the software, and that the components, application and system still comply with the specified requirements. | Testers |
| User Acceptance | Validates that the system meets mutually agreed upon requirements | PM |

Test Case Template

Table 5. Test Case Template

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type | ID | Feature/ Component | Test Case | Input | Pass / Fail |
| Unit | 001 | UI | A pop-up shall notify user that they are about to submit a response to a form. | First  Last  test@email.com | Failed |
| |  | | --- | | Expected Results |   sA screenshot of a computer error  Description automatically generated | | | | | |
| |  | | --- | | Actual Results |   A screenshot of a form  Description automatically generated | | | | | |
| Summary: The Test did not perform as expected because the submit button on the UI did not execute the pop-up informing the user they are about to submit the form. | | | | | |

Test Objectives

The SIRS project team (Team CHNF) will conduct tests on various scenarios that will prove the successful outcome of the project requirements and address the expected failure outcomes of this project.

Table 6. Requirements Traceability Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement ID | Feature | Justification | Pass / Fail |
| A001 | Graphic User Interface | A GUI shall open when code is executed and closed when exit button is clicked. | All Passed |
| A002 | Adding Course | User shall create a new course to hold corresponding assignments | All Passed |
| A003 | Adding Assignments | User needs to be able to add a new assignment to track grade score. | All Passed |
| A004 | Inputting Grades | User needs to be able to add grades to assignments to track course progress. | All Passed |
| A005 | Calculation of Average Grade | The program shall calculate the average of all assignment entered the system. | All Passed |
| A006 | Saving and Loading Data Sets | The program shall save data inputs from the user to revisit upon next login. | All Passed |
| A007 | Importing Grades | The program shall allow users to import grades from other digital formats. | Not Executed / Feature not deployed |
| A008 | Selecting Desired Overall Grades | The program shall allow the user to select the course grade they are targeting and give grades needed to meet that desired grade. | All Passed |

Test Case Scenarios

The SIRS project team will test the following scenarios which will prove a pass or fail outcome for all test cases.

**User Interface**:

* Test Case T001: GUI displays when application is ran.
* Test Case T002: GUI closed with close icon selected.

**Adding Course:**

* Test Case T003: Adding a new course with valid details.
* Test Case T004: Adding a new course with invalid details.

**Adding Gradable Assignments:**

* Test Case T005: Adding a new assignment with valid details.
* Test Case T006: Adding a new assignment with invalid details.
* Test Case T007: Adding a future assignment with valid details.

**Inputting Grades:**

* Test Case T008: Inputting valid grades inputs for completed assignments.
* Test Case T009: Inputting grades with invalid data in Grade Received.
* Test Case T0010: Inputting grades with invalid data in Possible Grade.

**Calculation of Average Grade**:

* Test Case T0011: Calculating the average grade needed for a desired overall grade.

**Saving and Loading Data Sets:**

* Test Case T012: Saving a data set successfully.
* Test Case T013: Loading a saved data set successfully.

**Importing Grades:**

* Test Case T014: Successfully importing grades from UMGC Course page.

**Selecting Desired Overall Grades:**

* Test Case T015: Selecting desired overall grades and verifying functionality.

System Design

UML Class Diagram

A screenshot of a computer

Description automatically generated

Appendix B. References

References:

Institute, Project Management. (2013). *A guide to the project management body of knowledge (pmbok guide), fifth edition*.

[*http://library.books24x7.com.ezproxy.umgc.edu/toc.aspx?bookid=51356*](http://library.books24x7.com.ezproxy.umgc.edu/toc.aspx?bookid=51356)(Accessed: 19 March 2024)

Pargaonkar, S. (2024) ‘Software testing principles and Techniques’, *A Guide to Software Quality Engineering*, pp. 41–54. doi:10.1201/9781032702049-3.

(2024) *Software Testing, Verification and Reliability*, 34(2). doi:10.1002/stvr.v34.2.

*Software testing* (2024) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Software\_testing (Accessed: 30 March 2024).

Tsui, F., & Karam, O. (2011). Essentials of Software Engineering(3rd ed.). Jones & Bartlett Learning. <https://library-books24x7-com.ezproxy.umgc.edu/toc.aspx?bookid=51648> (Accessed: 19 March 2024).

Appendix C. Contribution Report

Contribution Reports

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Name** | **Contribution** | **Category** |
| 28-Mar | Kelley | Created this contribution tracker | Documentation |
| 28-Mar | Ramirez | Created drafted the Test Plan | Documentation |
| 29-Mar | Ross | Worked on Test Plan Objectives | Documentation |
| 29-Mar | Ross | Worked on Test Case Scenarios | Documentation |
| 29-Mar | Ross | Worked on Test Plan Objectives | Documentation |
| 30-Mar | Ross | References | Documentation |
| 31-Mar | Kardash | Reviewed and provided feedback on the Test plan | Documentation |
| 31-Mar | Kelley | Reviewed and Edited Test Plan | Documentation |
| 31-Mar | Kelley | Created Draft User Guide | Documentation |
| 31-Mar | Kelley | Created GitHub Repository | Planning |
| 31-Mar | Kelley | Initiated GitHub Wiki specifications | Documentation |
| 1-Apr | Ramirez | Created version table to track document changes | Documentation |
| 1-Apr | Ramirez | Updated reference sources | Documentation |
| 1-Apr | Coleman | Reviewed and provided feedback on the Test plan | Documentation |
| 2-Apr | Kelley | Added team members to GitHub Collaboration | Coordination |
| 2-Apr | Ramirez | Consolidate feedback, format document, standardize text, and posted final version for group submit. | Documentation |
| 5-Apr | Kelley | Reviewed UI Draft Mockups | Design |
| 6-Apr | Ramirez | Identified and created database attributes and entities | Design |
| 6-Apr | Ramirez | Created ERD in Crows Foot Notation | Design |
| 8-Apr | Ramirez | Created SIRS Metadata Table | Design |
| 8-Apr | Kelley | Reviewed Database Design | Design |
| 8-Apr | Ross/Coleman | UI Design Mock Up | Documentation |
| 8-Apr | Ross/Coleman | UI Use Case | Documentation |
| 9-Apr | Ramirez | Updated version table to track document changes | Documentation |
| 9-Apr | Kelley | Drafted Project Schedule | Documentation |
| 9-Apr | Kelley | Integrated Diagrams and UI Mockup into Plan | Documentation |
| 9-Apr | Kardash | Created a class diagram for application. Shared it with team for review. | Backend Dev |
| 12-Apr | Kardash | Created classes, attributes, and get/set statements in Java using Eclipse IDE. | Backend Dev |
| 12-Apr | Ramirez | Create and set-up Junit Test Suite for future regression testing | Testing |
| 13-Apr | Ramirez | Created a Requirements Traceability Matrix | Documentation |
| 13-Apr | Kelley | Created Git repository branches | Coordination |
| 14-Apr | Kelley | Merged team member-created UI and backend classes into Git branch | Coordination |
| 14-Apr | Ramirez | Create appendices for references, test results, and contribution report | Documentation |
| 14-Apr | Ramirez | Updated document table of contents | Documentation |
| 14-Apr | Coleman | Created a UI layout displaying the necessary fields and buttons using Java Swing in the Eclipse IDE | Frontend Dev |
| 14-Apr | Marcel | UI code mock-up built | Frontend Dev |
| 14-Apr | Ramirez | Updated initial test results base on code available to test | Documentation |
| 15-Apr | Kelley | Created constructors and operational functions for backend classes | Backend Dev |
| 16-Apr | Kelley | Re-tasked team members to complete project due to unexpected code setbacks | Planning |
| 16-Apr | Kelley | Created and completed Controller class and modified existing classes to complete MVC program construction | Frontend Dev |
| 16-Apr | Kelley | Submitted changelogs and created pull request for team review | Coordination |
| 16-Apr | Kelley | Created example Wiki page to support porting of Project Plan to GitHub wiki page | Documentation |
| 14-Apr | Ramirez | Code review merge request for final src submission | Documentation |
| 17-Apr | Kelley | Created detailed Phase 2 task list for each team member and established program goals for Phase 2 | Coordination |
| 19-Apr | Kardash | Created constructor to handle assignments that do not yet have an actual grade. | Backend Dev |
| 19-Apr | Kelley | Added error checks and exception handling to backend classes | Backend Dev |
| 21-Apr | Kelley | Generated GitHub Pull Request for review | Backend Dev |
| 21-Apr | Kardash | Attempted to create function calculating points remaining to achieve a desired grade. Function had no effect on code. | Backend Dev |
| 21-Apr | Coleman | Updated UI with new functionality Design and Theme | Frontend Dev |
| 21-Apr | Kelley | Merged team code updates into unified testing branch for test review | Coordination |
| 21-Apr | Kelley | Provided team direction and clarification for additional needed program functionality | Coordination |
| 22-Apr | Kelley | Final minor code corrections and bug fixes before testing | Backend Dev |
| 22-Apr | Ramirez | Code review and test the phase\_2\_test branch code | Testing |
| 23-Apr | Ramirez | Run manual functionality test on program and add test results to main project document | Documentation |
| 23-Apr | Ramirez | Gather and collect contribution report and add to main project document | Documentation |
| 23-Apr | Ross | Added Error Handling within the Controller class and check for the inappropriate entry types and throw an error with a pop up. | Frontend Dev |
| 23-Apr | Kelley | Updated UML class diagram with new class attributes | Documentation |
| 23-Apr | Kelley | Cleaned up and merged GitHub repo branches. | Coordination |
| 24-Apr | Kelley | Created detailed weekly task and goal list and posted it to the Discord | Coordination |
| 24-Apr | Kelley | Created a GitHub Pull Request for review to integrate the most up to date code into our main branch. Also posted PR description to Discord | Backend Dev |
| 24-Apr | Kelley | Discovered and documented an issue within the codebase that will prevent the project from achieving threshold functionality. This issue is in relation to how point values are handled as ints versus doubles | QA/QC |
| 24-Apr | Kelley | Created a new GitHub branch for Phase 3 functionality inclusion. | Backend Dev |
| 25-Apr | Kelley | Coded up important threshold functionality on the backend systems for calculating the percentage and points required for the user to achieve their desired grade. | Backend Dev |
| 26-Apr | Kelley | Wrote documentation on the interfaces for our frontend team to be able to utilize the new functionality coded within the backend | Coordination |
| 27-Apr | Kelley | Refactored three classes to handle point values as doubles (instead of ints) and modified all functions dealing with these values | Bugfixing |
| 27-Apr | Kardash | Reviewed latest changes made to Phase 2, per Kendle Kelley's request. Unable to test due to unexplained Eclipse error. | Backend Dev |
| 28-Apr | Kardash | Began creating Save/Load function. | Backend Dev |
| 28-Apr | Coleman | Added functionality to UI class to handle different inputs and general optimization | Frontend Dev |
| 28-Apr | Kelley | Recorded a How-To Guide video to show teammates how to operate GitHub and bring branches into Eclipse IDE | Documentation |
| 29-Apr | Kelley | Merged, bugfixed, and packaged all team member inputs into one branch for testing | Coordination |
| 29-Apr | Kelley | Assisted teammates in refactoring their code when bugs were revealed from branch merge | Coordination |
| 29-Apr | Kardash | Created the basis for a Save/Load function. Handed it over to Kendle Kelley for bug fixing. | Backend Dev |
| 30- Apr | Coleman | Updated controller class to handle the changes made to UI class | Frontend Dev |
| 30-Apr | Ramirez | Conduct unit and regression testing on the application as a whole | Testing |
| 30-Apr | Ramirez | Update test results in project documents and consolidate contribution reports. | Documentation |
| 30-Apr | Kelley | Bugfixed and refactored save/load functionality code after merging | Backend Dev |
| 30-Apr | Kelley | Recorded and posted to YouTube a SIRS Project Overview and Quick User Guide for the program's functionalities | Documentation |
| 30-Apr | Kelley | Builted Runnable Jar file and packaged for Peer Review | Coordination |
| 1-May | Kelley | Created detailed weekly task and goal list and posted it to the Discord | Coordination |
| 1-May | Kelley | Created a GitHub Pull Request for review to integrate the most up to date code into our main branch. Also posted PR description to Discord | Backend Dev |
| 2-May | Kelley | Assisted in troubleshooting Save/Load system | QA/QC |
| 3-May | Kelley | Merged Save/Load system into codebase | Coordination |
| 3-May | Kardash | First Attempt to Create JFileChooser for assignment selection. Result needed corrections. | Backend Dev |
| 4-May | Coleman | Added functionality and made adjustments to the UI | Frontend Dev |
| 4-May | Kardash | Made corrections to JFileChooser as suggested by Kendle Kelley. | Backend Dev |
| 4-May | Kelley | Resolved mission assignment data bug | Bugfixing |
| 4-May | Kelley | Created functionality to recall latest Assignment object created and remove it from the list, allowing an amount of database control | Frontend Dev |
| 4-May | Kelley | Created functionality to display a pop up with message data from any unhandled exceptions | Frontend Dev |
| 5-May | Coleman | Fixed issues with UI display | Frontend Dev |
| 5-May | Kardash | Viewed and acknowledged additional corrections to JFileChooser by Kendle Kelley. | Backend Dev |
| 5-May | Kelley | Merged all frontend dev work into combined backend branch | Coordination |
| 5-May | Kelley | Resolved significant amount of bugs caused by branch merge | Bugfixing |
| 5-May | Kelley | Final UI and Controller class minor bugfixes and functionalities added and merged into GitHub repository | Frontend Dev |
| 5-May | Kelley | Generated test datasets and executable JAR of final build for test phase. Directed group to test | Coordination |
| 5-May | Ramirez | Test Final code branch | Testing |
| 5-May | Ramirez | Update User Guide instructions and pictures | Documentation |
| 6-May | Ramirez | Draft and inspect final project plan documentation | Documentation |
| 6-May | Kelley | Reviewed Project Plan and Project Requirements. Scored all threshold and objective requirements achieved. | Documentation |
| 7-May | Kelley | Recorded Project Overview and Users Guide Tutorial video and posted to YouTube | Documentation |
| 7-May | Kelley | Conducted final checks of Project Plan, codebase, and deliverables and directed team on correct files to submit | Coordination |