

D424 – Software Engineering
Task 3



Capstone Proposal Project Name: WGU: SAM (Simplified Academic Management)

Student Name: Aron Fejes

Table of Contents**Table of Contents**

<i>Table of Contents.....</i>	<i>2</i>
<i>Application Design and Testing.....</i>	<i>4</i>
Class Design	4
UI Design	6
<i>Unit Test Plan.....</i>	<i>9</i>
Introduction	9
Test Plan.....	10
Specifications.....	12
Procedures.....	12
Results.....	13
Conclusion	15
<i>C2. Web App Hosting Information:.....</i>	<i>15</i>
<i>C3. GitLab Repository Details:.....</i>	<i>15</i>
Repository Link: WGU Student Repository - AFejes_Capstone	15
<i>C4. User Guide: Setting Up and Running the Application for Maintenance Purposes</i>	<i>16</i>
Prerequisites:	16
Cloning the Repository:	16
Setting Up the Development Environment:	16
Running the Application:.....	16

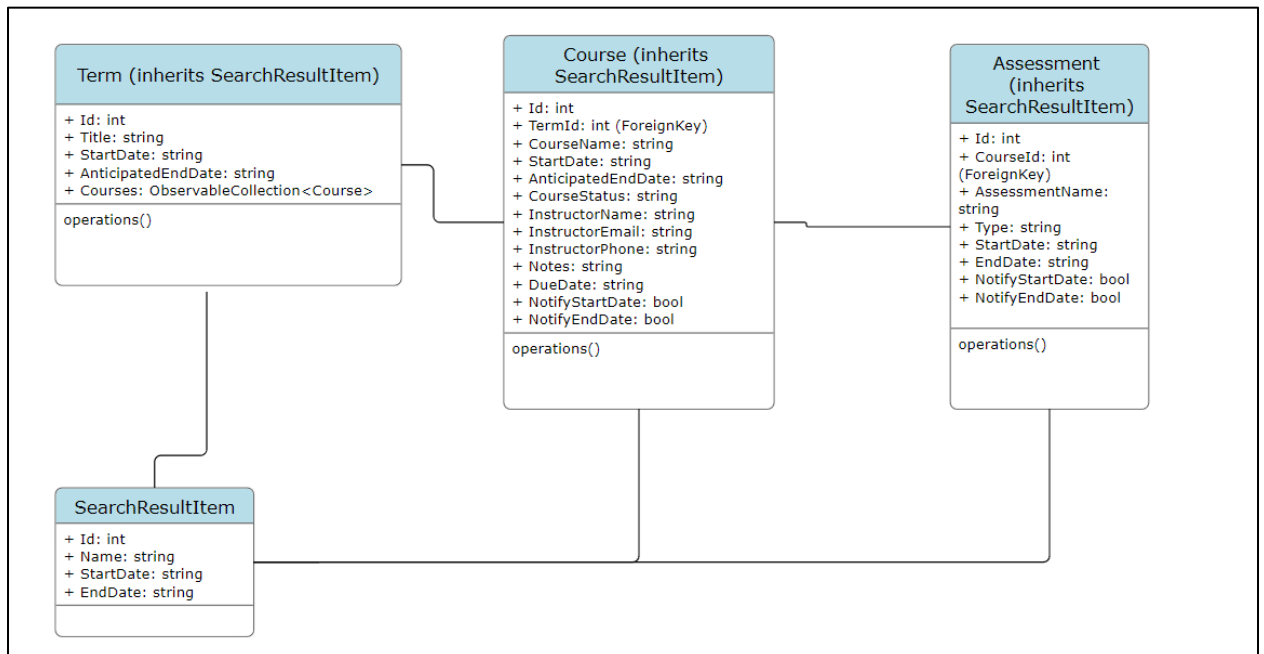
Maintenance Tasks:	17
Deployment and Builds:	18
Troubleshooting Common Issues:	18
<i>C5. User Guide: Running the Application from a User Perspective</i>	<i>19</i>
Introduction	19
Installation and Using the Application	19
Navigating the Application	19
Main Page (Academic Terms)	21
Term Details Page	22
Courses Page	23
Course Details Page	24
Reports Page	26
Generate PDF:	26
Assessment Data Overview:	26
Search Results Page	27
Shell Navigation	28
Conclusion	28

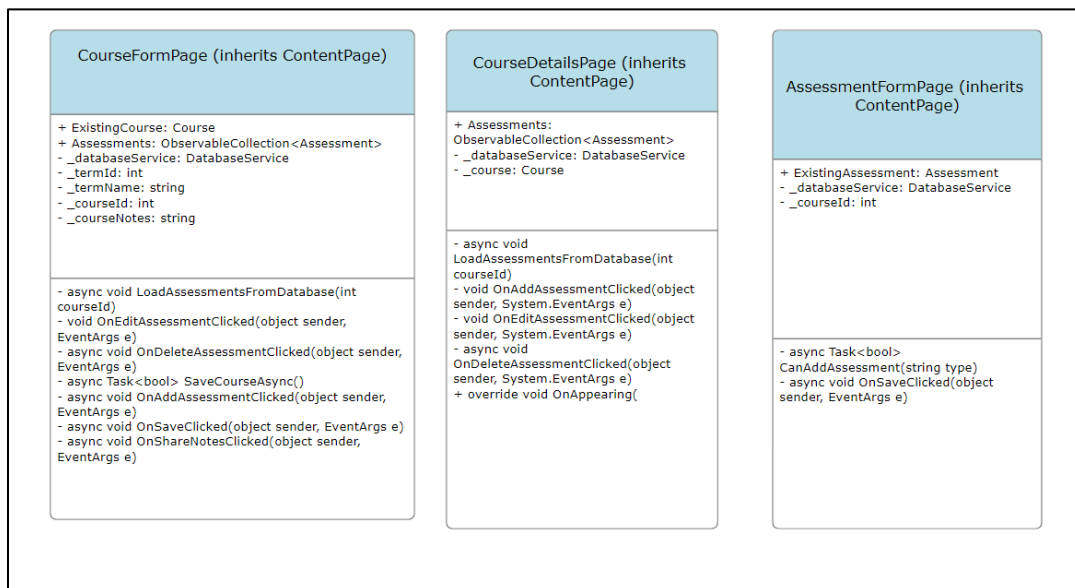
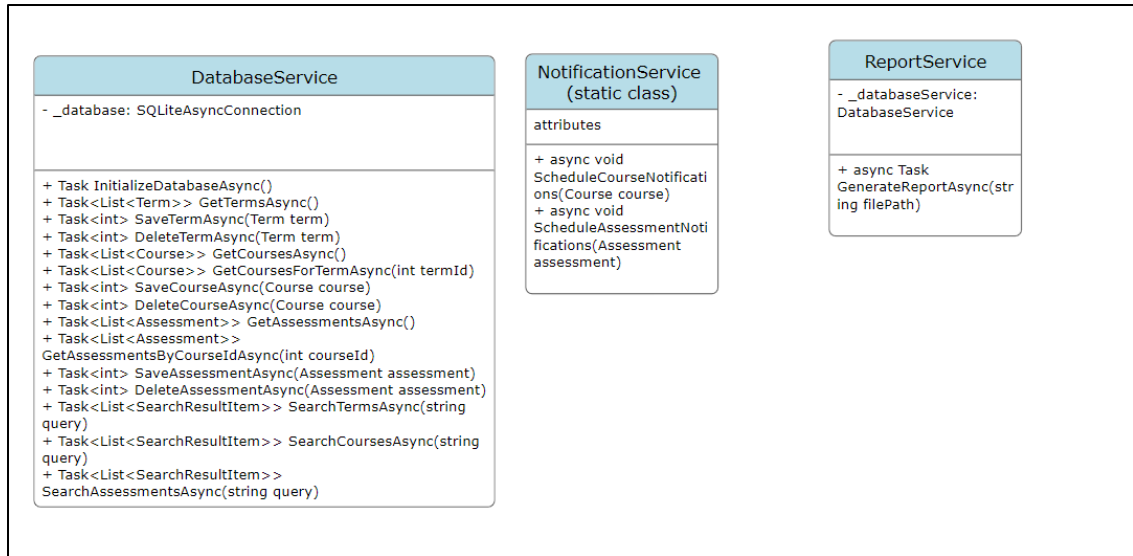
Task 3 Design Document

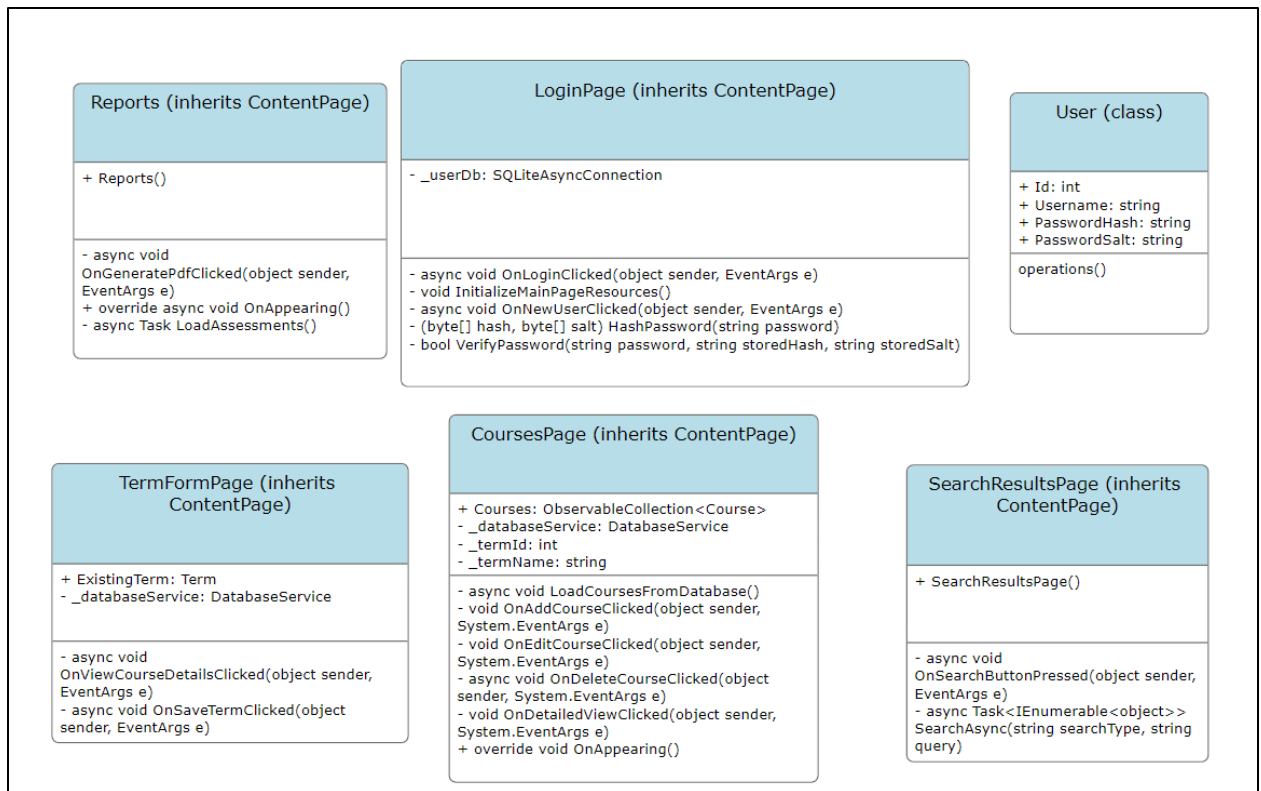
Application Design and Testing

Class Design

In the development of the Capstone project, I employed a comprehensive class design that ensured modularity, maintainability, and readability of the codebase. My design revolved around the principles of Object-Oriented Programming (OOP), ensuring encapsulation, inheritance, and polymorphism were appropriately utilized.







UI Design

The success of any application largely depends on its user interface. A well-thought-out and intuitive UI design not only enhances the user experience but also ensures efficient interaction with the application's functionalities. In the initial stages of my Capstone project, before diving deep into the coding phase, I laid out a low-fidelity wireframe. This wireframe served as a blueprint, helping us visualize the application's structure, layout, and the flow of user interactions.

[Home](#)
[Settings](#)
[Notifications](#)

Frame

WGU: SAM

Login

Username

Password

Login

Create Account

[Home](#)
[Settings](#)
[Notifications](#)

Frame

WGU: SAM

Existing Terms

Term 1 - Start Date / End Date

Term 2 - Start Date / End Date

Term 3 - Start Date / End Date

Term 4 - Start Date / End Date

Add New Term

[Home](#)
[Settings](#)
[Notifications](#)

Add/Edit Academic Term

WGU: SAM

Academic Term

Term Title

Start Date

01/01/2022

Anticipated End Date

06/01/2022

Save

Delete Term

Add New Course

List of Courses

Course 1

Start Date: 01/01/2022

End Date: 01/05/2022

Course 2

Start Date: 01/06/2022

End Date: 01/10/2022

Course 3

Start Date: 01/11/2022

End Date: 01/15/2022

Course 4

Start Date: 01/16/2022

End Date: 01/20/2022

Course 5

Start Date: 01/21/2022

End Date: 01/25/2022

Course 6

Start Date: 01/26/2022

End Date: 01/30/2022

[Home](#)
[Settings](#)
[Notifications](#)

Add/Edit Course Name

WGUSAM

Course Name

Course Name

Course 1

Start Date

01/01/2022

☐ Set Notification for Start Date

Anticipated End Date

01/05/2022

☐ Set Notification for End Date

Course Status

Select course status

Instructor's Name

Instructor's Phone

Instructor's Email

Optional Notes

Assessments

Assessment 1

Objective Assessment

Assessment 2

Performance Assessment

Share Notes

Add New Assessment

Save

Delete Course

[Home](#)
[Settings](#)
[Notifications](#)

Add/Edit Assessment

WGUSAM

Add/Edit Assessment

Type

Objective Assessment

Assessment Name

Start Date

01/02/2022

End Date

01/02/2022

☒ Set Notification for Start Date

☒ Set Notification for End Date

Save

Delete Assessment

Unit Test Plan

Introduction

Purpose

During the development of WGU: SAM (Simplified Academic Management), I implemented a combination of unit testing and integration testing to ensure the software's reliability and effectiveness. The primary aim of these tests was to validate the functionality of the application, particularly its database interactions and user interface behaviors. My tests yielded a high success rate, indicating that the application is robust and functions as expected. However, there were a few edge cases that were identified and addressed. Remediation involved revisiting the codebase to adjust logic or handle specific scenarios, followed by rerunning the tests to ensure accuracy.

Overview

The testing methodologies employed for WGU: SAM were necessary in ensuring the overall quality of the project. While unit testing was a consistent practice used throughout the development cycle for individual components, integration testing was used for validating the interactions between different parts of the application, such as the UI and the backend database.

For instance, a recurring method used in many parts of the application was the SQLite database operations, like Create, Read, Update, and Delete (CRUD). These operations are essential, and unit tests were designed to ensure that each operation functioned as expected.

Functions tested ranged from basic CRUD operations for terms, courses, and assessments to more intricate functionalities like generating reports or scheduling notifications. Each function underwent a series of tests, simulating various user inputs and scenarios.

Whenever errors or unexpected behaviors were identified, the development team collaborated to pinpoint the source of the issue. Typically, the errors were a result of overlooked scenarios or minor logical flaws. These were addressed by refining the code, improving error handling, or adding conditions to cater to specific scenarios. After adjustments, tests were rerun to ensure that the issues were effectively resolved and that no new issues were introduced.

In conclusion, the rigorous testing methodologies employed during the development of WGU: SAM ensured a high-quality, reliable application that meets user expectations and adheres to industry best practices.

Test Plan

Items

To execute the tests for the WGU: SAM application, the following items were required:

1. A development environment set up with the necessary SDKs and tools, including Microsoft MAUI and SQLite.
2. A test environment or emulator for simulating different device configurations.
3. A version control system (like Git) to manage code iterations.
4. Test data samples for database interactions.

Features

The features/functions tested include:

1. CRUD operations for:
 - a. Terms
 - b. Courses
 - c. Assessments
2. Report generation.
3. Search functionality across terms, courses, and assessments.

Deliverables

The tests produced:

1. Documentation of each test scenario, including input data, expected outcome, and actual outcome.
2. Code annotations highlighting critical sections and potential points of failure.
3. Logs capturing detailed information during test execution.
4. Error reports detailing failures or inconsistencies.

Tasks

The tasks required to complete the testing were:

1. Setting up the test environment.
2. Designing and documenting test cases.
3. Executing each test scenario.
4. Logging and documenting the outcomes.
5. Analyzing failures and implementing fixes.
6. Re-running tests after adjustments.
7. Finalizing test reports.

Needs

To conduct the tests, the following were necessary:

1. Microsoft MAUI SDK (latest stable version).
2. SQLite library for database operations.
3. Device emulators/simulators for various configurations.
4. Testing libraries/package, NUnit, for unit tests.

Pass/Fail Criteria

The success criteria for the tests were:

1. All CRUD operations should execute without errors and should match expected results.
2. Reports should be generated accurately and in the required format.
3. Searches should return accurate results based on the input query.
4. In case of a test failure:

5. issue was logged and documented with details of the input and observed outcome.
6. The development team collaborated to identify the source of the failure.
7. Remediation strategies were discussed, implemented, and the tests rerun.
8. All fixes and the reasons for failures were documented for future reference.

Specifications

In my project, WGU: SAM (Simplified Academic Management), testing played a pivotal role in ensuring that the system was reliable and robust. I employed unit testing, one of the most widely recognized methods in software development, to ascertain the correct functioning of individual units or components of the software. NUnit, a popular unit testing framework for .NET applications, was my tool of choice for this purpose.

Procedures

Setting Up the Environment:

- The first step involved establishing a conducive environment for testing. I installed the NUnit framework and its associated libraries.
- A mock database environment was set up to simulate the application's real database. This ensured that tests were isolated and that the real database remained unaffected during the testing phase.

Designing Test Cases:

- A comprehensive list of functionalities and scenarios requiring testing was compiled.
- Each function in my application had a corresponding test case. This approach was adopted to ensure extensive coverage, including positive cases (expected behavior), negative cases (handling of incorrect inputs or situations), and edge cases (unusual or extreme cases).

Execution:

- After designing the test cases, the tests were run using the NUnit testing framework.

- Every test's outcome was logged and documented for future reference. In instances where a test did not pass, the input and the observed outcome were detailed to facilitate debugging.

Iterative Process:

- As software development is often an iterative process, my tests were continuously updated and expanded upon to keep pace with the application's evolution.
- After the completion of each major development phase, the entire suite of tests was run to ensure there were no regressions and that recent changes hadn't inadvertently introduced bugs.

Generating Results:

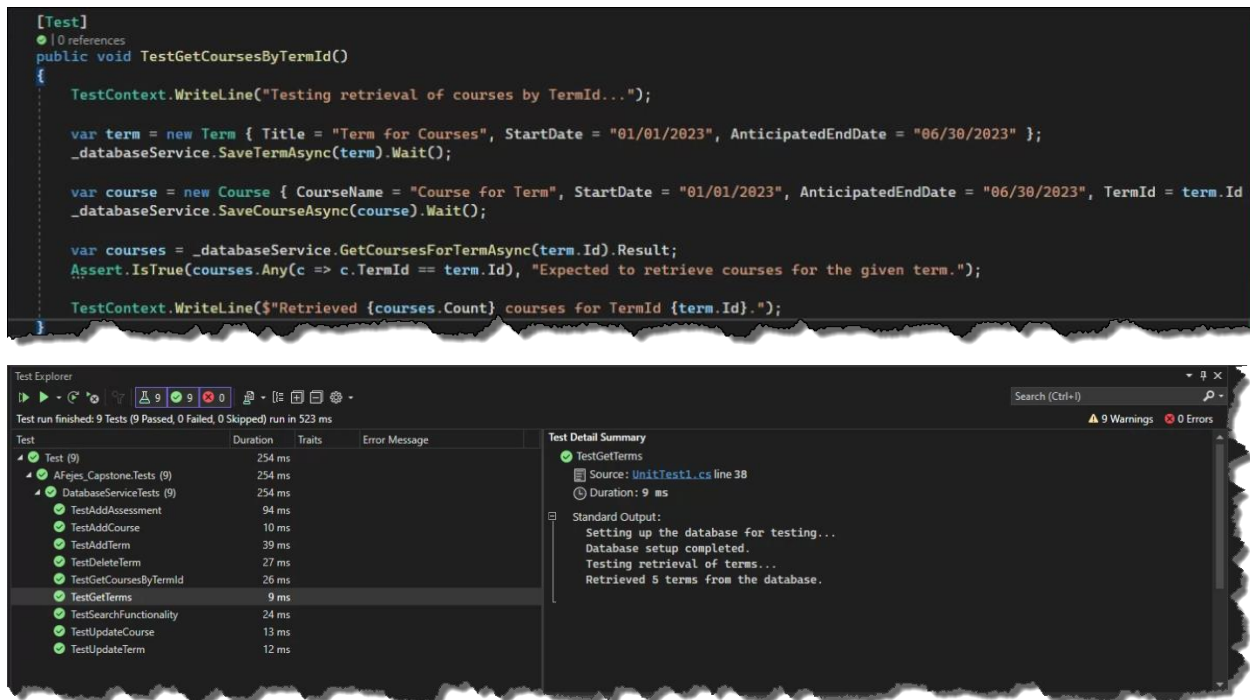
- Post the test execution, NUnit provided a detailed report of the outcomes, clearly indicating which tests passed and which did not.

Results

Unit testing provided valuable insights into the application's stability and highlighted areas that required improvement or fine-tuning.

CRUD Operations:

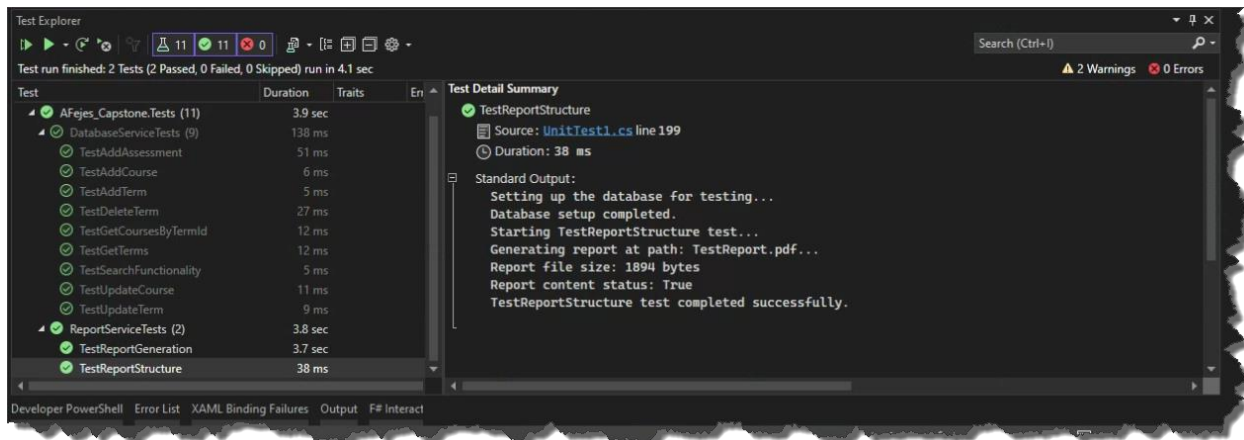
- **Creation:** Entities can be successfully added to the database. Tests confirmed the successful addition and the database returned a positive identifier for the newly added entity.
- **Read:** Retrieval of entities from the database was consistent and accurate. Tests confirmed that the entities retrieved were of the expected type and were not null.
- **Update:** Changes to existing entities were correctly reflected in the database. The tests ensured that updates did not inadvertently change unrelated attributes or records.
- **Deletion:** Entities can be successfully deleted from the database. The tests also verified the cascading nature of deletions where appropriate, ensuring related entities were also removed when a primary entity was deleted.



Report Generation:

- **Basic Functionality:** The software successfully generated reports in the specified format and at the determined file path.
- **Report Structure:** Preliminary tests confirmed that the generated report contained the expected content by checking various metrics, including file size.





Conclusion

The tests provided a comprehensive evaluation of the fundamental functionalities of the WGU: SAM software. While the software demonstrated proficiency in the tests conducted, it's crucial to recognize that these unit tests represent only one tier of the testing process. Additional testing layers, including integration and end-to-end tests, would further ensure the software's reliability in a production setting.

C2. Web App Hosting Information:

For this project, a web app with HTML code hosting is not applicable. The primary focus and development of this project revolves around creating a mobile application. Therefore, there isn't a hosted web version available.

For the source code, revisions, and further details, refer to the provided GitLab repository link in the following section.

C3. GitLab Repository Details:

For a comprehensive look into the source code, revisions, and collaborative history of the project, you can access the GitLab repository. The version of the code that is being discussed and analyzed in this submission can be found in the "master" branch.

Repository Link: [WGU Student Repository - AFejes_Capstone](#)

For clarity and transparency, always ensure to check the latest commits and updates on the "master" branch, as this represents the most recent and relevant version of the code for this submission.

<https://gitlab.com/wgu-gitlab-environment/student-repos/afejes21/d424-software-engineering-capstone>

C4. User Guide: Setting Up and Running the Application for Maintenance Purposes

Prerequisites:

- **Development Environment:** Ensure you have Visual Studio 2022 or later installed with .NET 7.0 SDK.
- **Mobile Emulators:** For testing purposes, ensure you have Android emulators or physical devices set up.
- **Database:** SQLite is used as the primary database. Ensure you have SQLite tools installed for debugging purposes.

Cloning the Repository:

1. Open your terminal or command prompt.
2. Navigate to the directory where you want to clone the repository.
3. Run the following command:

```
git clone https://gitlab.com/wgu-gitlab-environment/student-repos/afejes21/d424-software-engineering-capstone.git
```

4. Navigate to the cloned directory:

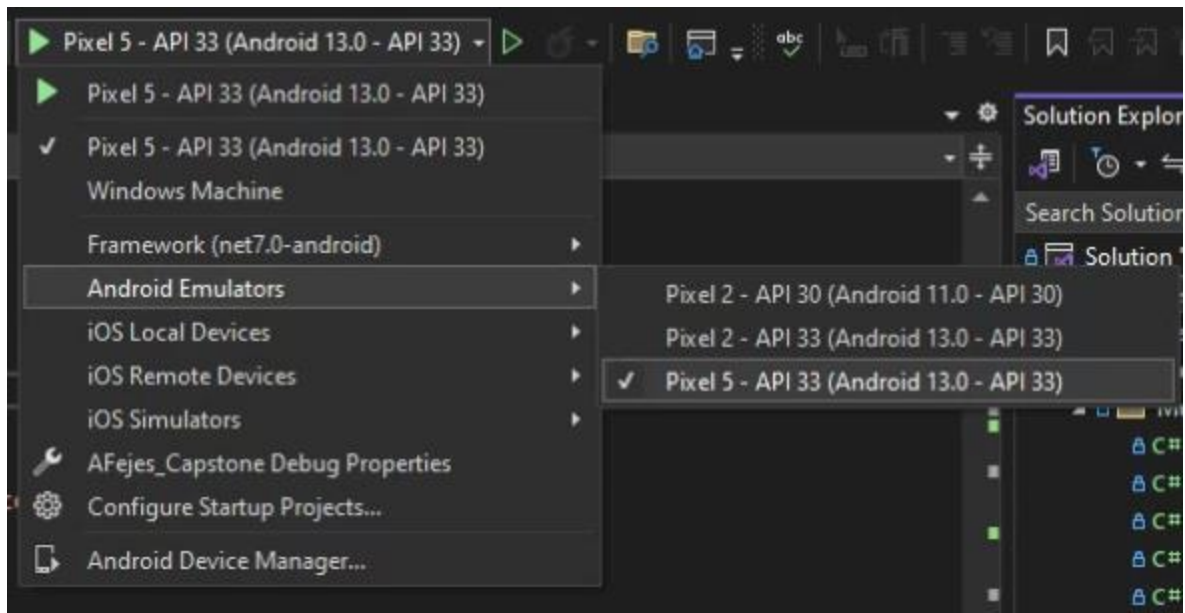
```
cd d424-software-engineering-capstone
```

Setting Up the Development Environment:

1. Open the solution file **AFejes_Capstone.sln** in Visual Studio.
2. Ensure all the NuGet packages are restored. If not, right-click on the solution and select "Restore NuGet Packages".
3. Set the desired startup project (e.g., AFejes_Capstone and not the Test project) by right-clicking on the respective project and selecting "Set as Startup Project".

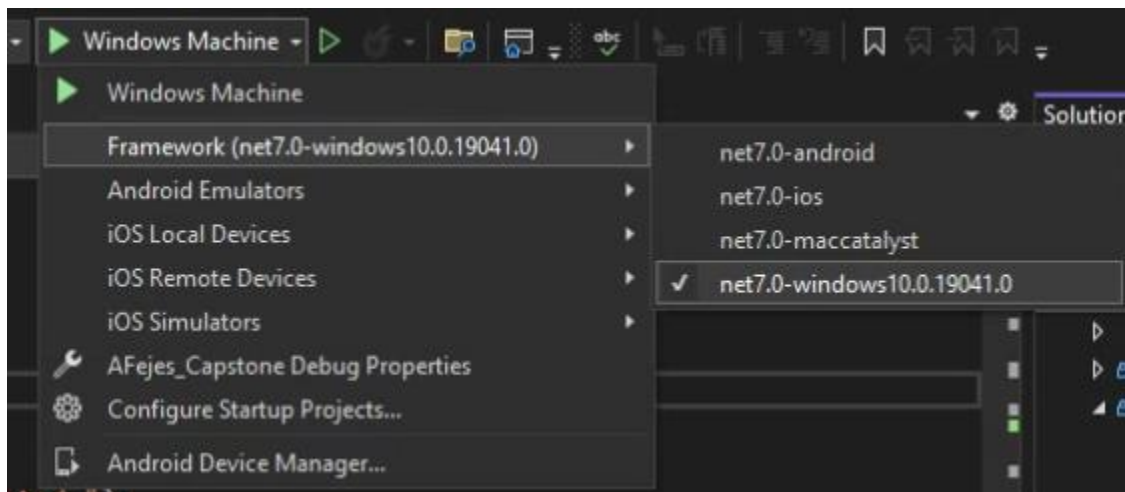
Running the Application:

- Android device/emulator
 1. To run on an Android device/emulator, ensure you have the AFejes_Capstone project set as the startup.
 2. Select an appropriate Android Emulator, or local device, from the drop-down



3. Press the **F5** key or click on the "Start Debugging" button.

- Windows Store
 1. To run on a Windows device, through the Windows Store, ensure you have the AFejes_Capstone project set as the startup.
 2. Select "net7.0-windows" from the drop-down



3. Press the **F5** key or click on the "Start Debugging" button.

Maintenance Tasks:

- **Database Schema Changes:** If there are any changes to the database schema, ensure you update the respective models in the **Models** folder. After updating, run the application to ensure SQLite tables are updated accordingly.

- **Adding New Features:** When adding new features or services, create separate classes in the respective folders (e.g., Services, Views, Models) to maintain the modular structure of the application.
- **Updating UI Components:** If updating or adding new UI components, ensure they are placed in the appropriate XAML files under the **Views** folder and in the correct Namespace.
- **Unit Testing:** Before pushing any major changes, ensure that all unit tests are run and passing. If adding new functionalities, consider adding relevant unit tests to the Test project in the solution.

Deployment and Builds:

To create a build for Android, right-click on the Android project and select "Publish". Follow the prompts to generate an APK or App Bundle.

Troubleshooting Common Issues:

- **Build Errors:** Ensure all NuGet packages are up-to-date and restored. Clean the solution (**Build > Clean Solution**) and rebuild.
- **Runtime Errors:** Check the stack trace for specific details. If related to the database, ensure that the SQLite schema matches the model classes.
- **Emulator Issues:** Ensure the emulator is running the correct OS version and has Google Play Services (for Android) updated.

C5. User Guide: Running the Application from a User Perspective

User Guide

Introduction

Welcome to the User Guide for the WGU: SAM application! This guide is designed to provide you with clear, step-by-step instructions on how to install, log into, sign up, and navigate through all the features of the application. Whether you're a new user trying to set up your academic management or an existing user looking to make the most out of the platform, this guide is for you. I've ensured that each step is detailed yet concise, making your experience seamless and intuitive.

Installation and Using the Application

Installation

- **Locate the Installation File:** Find the **WGUSAM.apk** file (or equivalent) for the application, or download from the Google Play store.
- **Run the Installer:** Double-click the installation file to initiate the setup process, or install from the Google Play store.
- **Follow On-Screen Instructions:** The installer will guide you through the setup process. Click "Next" as prompted, and choose your desired installation location.
- **Complete Installation:** Once the installation is complete, click on "Finish" to close the installer.

![Placeholder for Installation Image]

Navigating the Application

The WGU: SAM application is organized into various sections, allowing for efficient academic management:

Home/Main Page:

- View a list of academic terms.
- Edit, delete, or view courses related to a term.
- Add a new academic term.

Reports Page:

Generate and view reports related to assessments.

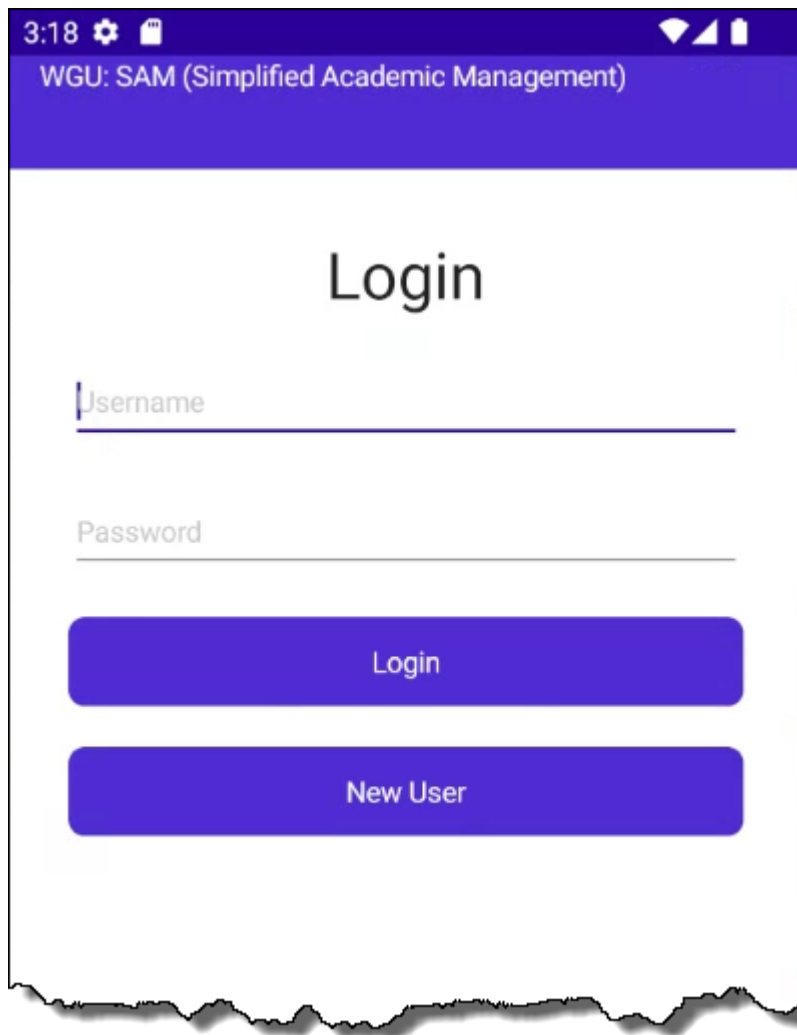
Search Page:

Find specific terms, courses, or assessments based on chosen criteria.

Each section is designed to offer specific functionalities, ensuring a streamlined user experience.

Logging In

- **Launch the Application:** Click on the WGU: SAM application icon on your device.
- **Input Credentials:** Enter your username and password in the respective fields.
- **Access the Platform:** Click the "Login" button.
 - **Don't have an account?** Click on the "New User" button to create your account.



Main Page (Academic Terms)

After successfully logging in, you'll land on the main page displaying a list of academic terms.

- **View Term Details:**

You can view the details of each term, such as the title, start date, and anticipated end date.

- **Edit Term:**

Click on the "✎ Edit Term" button to modify the details of a specific term.

- **Delete Term:**

Use the "🗑 Delete Term" button to remove a term.

- **View Courses for a Term:**

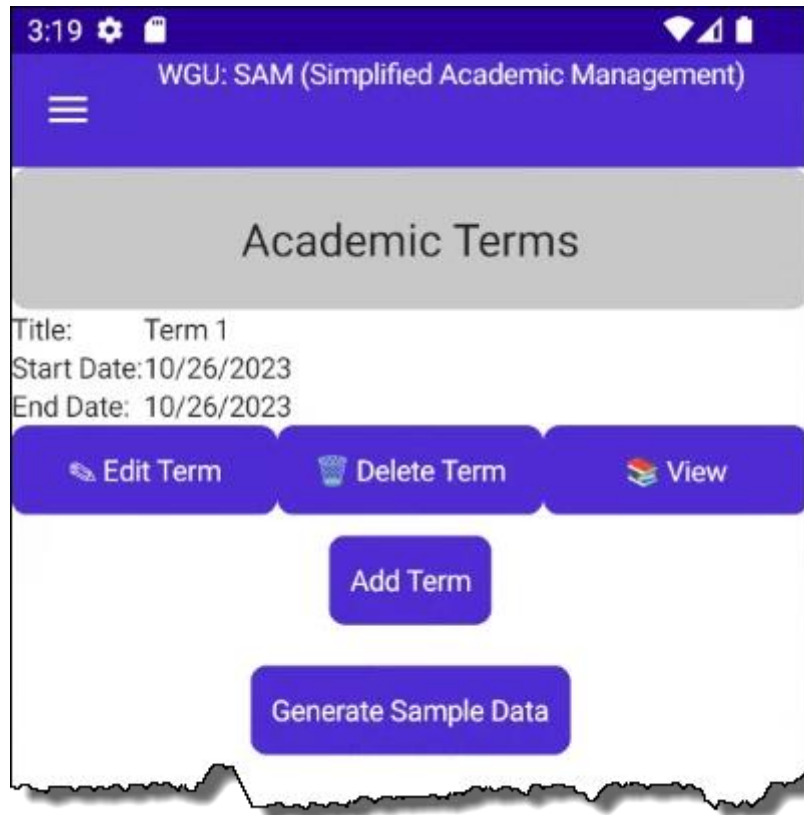
To view the courses associated with a term, click on the "📖 View Courses" button.

- **Add Term:**

To add a new academic term, click on the "Add Term" button at the bottom of the page.

- **Generate Sample Data:**

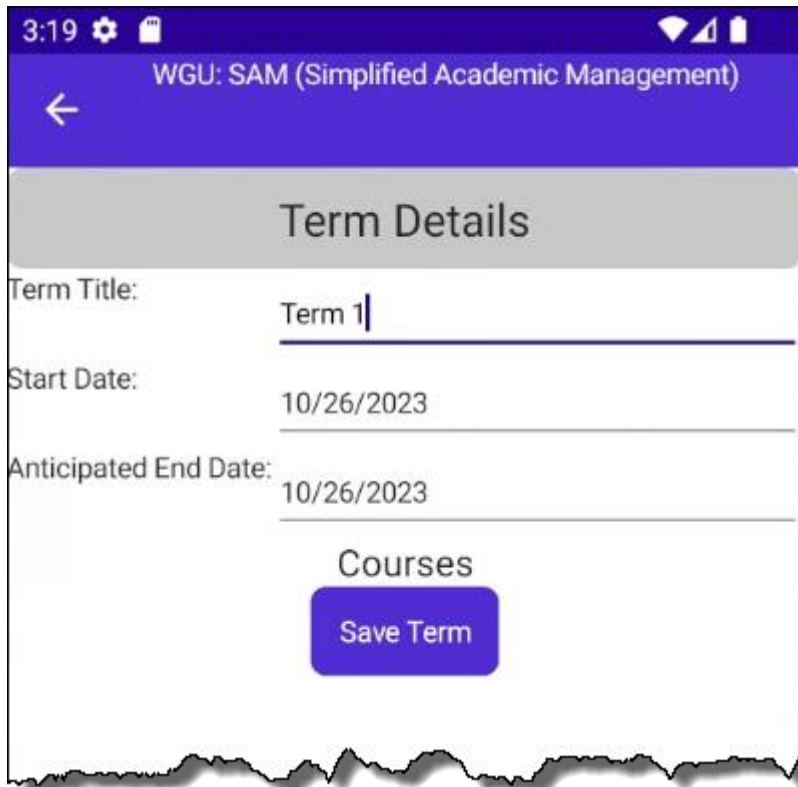
To add a new sample academic term, click on the "Generate Sample Data" button at the bottom of the page.



Term Details Page

On this page, you can add or modify term details.

- **Fill in Term Details:**
 - Provide the title, start date, and anticipated end date for the term.
- **Save:**
 - Click on the "Save Term" button to save the term details.



3:19

WGU: SAM (Simplified Academic Management)

←

Term Details

Term Title: Term 1

Start Date: 10/26/2023

Anticipated End Date: 10/26/2023

Courses

Save Term

Courses Page

This page displays a list of courses for a particular term.

- **View Course Details:**
 - Each course card displays the course name, start date, end date, instructor details, and other relevant information.
- **Edit Course:**
 - Use the "✎ Edit Course" button to modify course details.
- **Delete Course:**
 - Click on the "🗑 Delete Course" button to remove a course.
- **Add Course:**
 - To add a new course, click on the "Add Course" button at the bottom of the page.



Course Details Page

This page lets you add or modify course details.

- **Fill in Course Details:**
 - Provide details such as the course name, start date, end date, status, instructor information, and notes.
- **Assessments:**
 - View the assessments related to this course.
 - Add a new assessment by clicking on the "+" button.
- **Save:**
 - Click on the "Save" button to save the course details.

3:20







WGU: SAM (Simplified Academic Management)

Add Course for Term Term 1

Course 1

Start Date: 10/26/2023

Start Date Notification ☒

End Date: 10/26/2023

End Date Notification ☐

Due Date 10/26/2023

In Progress

Professor Tim

Test@Email.com

123-456-7890

test

Share Notes

Assessments

Reports Page

The reports page provides you with an overview of all assessments.

Generate PDF:

Click on the "Generate PDF" button to export the assessments data into a PDF format.

Assessment Data Overview:

Below the "Generate PDF" button, you will find a grid displaying all the assessments. This grid provides information about the ID, assessment name, type, start date, and end date of each assessment.

ID	Name	Type	Start Date	End Date
1	Test	Objective	10/26/202	10/26/202
2	Midterm Exam	Objective	2023-10-1	2023-10-1
3	Final Exam	Performance	2023-12-1	2023-12-1
4	Midterm Exam	Objective	2023-10-1	2023-10-1
5	Final Exam	Performance	2023-12-1	2023-12-1
6	Midterm Exam	Objective	2023-10-1	2023-10-1
7	Final Exam	Performance	2023-12-1	2023-12-1

Search Results Page

The search results page allows you to search for specific terms, courses, or assessments based on your chosen criteria.

Select Search Type:

From the dropdown menu, select the type of search you want to conduct: "Terms", "Courses", or "Assessments".

Search Bar:

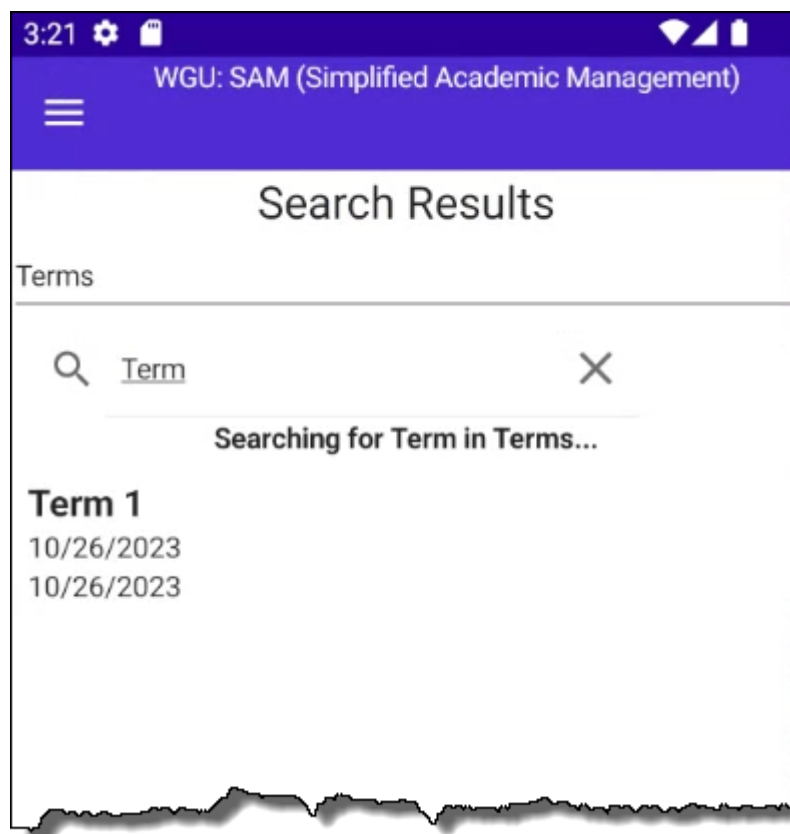
Enter your search criteria in the provided search bar and press the search button.

Status Label:

This label will inform you about the status of your search, whether it was successful, or if there were no results found.

Search Results List:

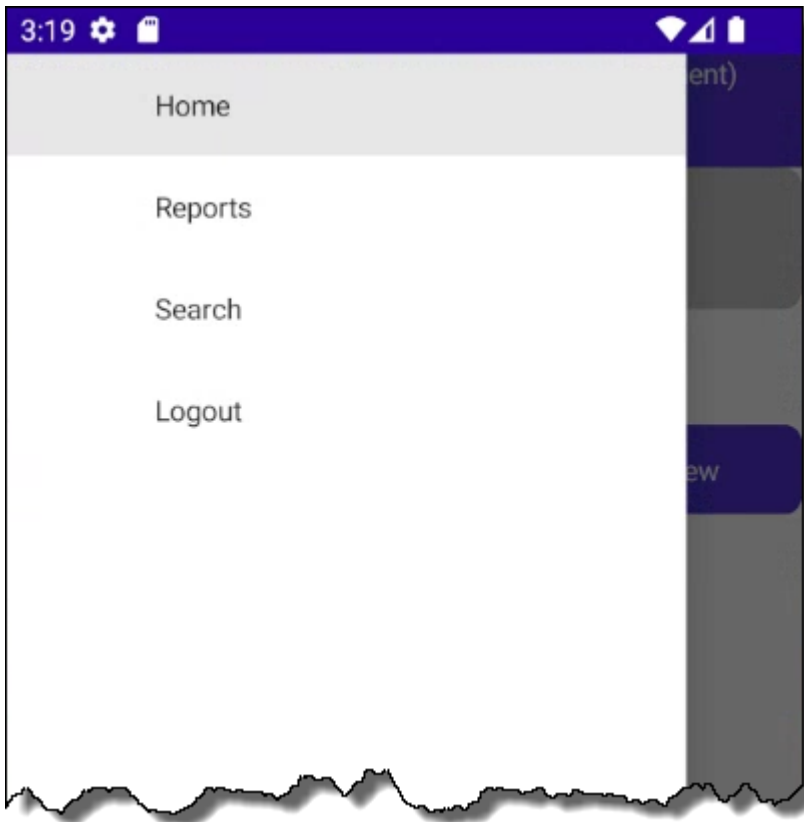
The results of your search will be displayed in a list format below the search bar. Each result will provide details such as the name, start date, and end date.



Shell Navigation

At any point, you can use the shell navigation to move between different sections of the application:

- **Home:**
 - Takes you back to the main page where you can view academic terms.
- **Reports:**
 - Access the reports section to view and generate various reports.
- **Search:**
 - Search for specific terms, courses, or assessments.
- **Logout:**
 - Log out of the application and return to the login page.



Conclusion

This guide aims to assist you in smoothly navigating through the WGU: SAM application. Thank you for choosing WGU: SAM for your academic management needs. I hope you find this guide helpful!