Student Course Recommendation Application

CSTP2104 Windows App Programming

Matt Caballero

000440554

**Project Description**

Overview

Our department, Computer Systems Technology, has been manually enrolling students by inputting each individual student information and scrolling up and down files to look for students' information. This process was sufficient at first but as more students come in, it eventually became too much of a hassle for them.

This is a project for my CSTP 2104 Windows Application Programming class. We were tasked with creating an application that would help some, if not all, of the problems the department is facing with student enrollment.

Objectives

The main objective is to automate the process of enrolling a student. But since we have a limited amount of time, we narrowed our scope to what we think are the most reasonable features and functionalities to have.

The main functionality is to be able to give out the courses a student can take with considering the prerequisites of each course. The application will also be able to filter out these recommendations by school year and school term.

Additional features that we have implemented is the searching for a student, displaying that student’s information, and displaying the courses the student has taken and is currently taking.

Design Constraints

The capability to automatically transfer or put data into the application is not implemented. Transferring of data is to be done from database to database. There is also the option to manually put down information to the database or use queries.

Environment

This is a Windows application and is limited to Windows operated devices only.

**Requirements**

Non-Functional Requirements

*Operational Requirements*

For the application to properly operate its functionalities, there must be sufficient data and information stored in the database. The application is heavily dependent on the database and without it properly set up, the application is useless.

*Performance*

For now, the application is designed for faculty use only. This set up can be handled by a centralized database with little to no performance issues. In the course of time, the application might be distributed to different users such as students. With this amount of usage, the database would not be able to handle traffic. Having a distributed database from the start or upgrading to it will surely increase performance of the application.

*Security*

The application stores sensitive information about the students. We have not dealt with application security because it is not part of our scope. The department has specific security requirements and procedures that they can implement on the application on their own.

Functional Requirements

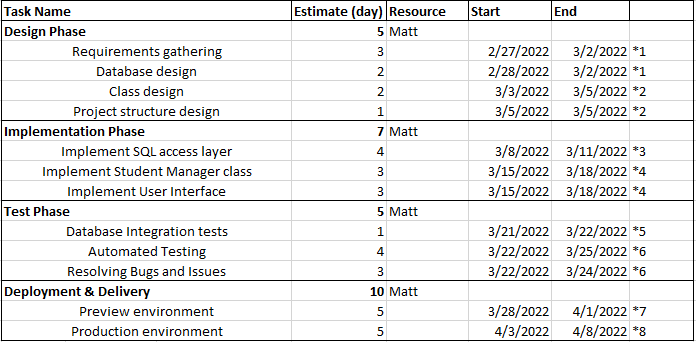
*Use Case 1 - Searching for student information*

|  |  |  |
| --- | --- | --- |
| **Use Case Scenario** | A faculty member wants to look for a student’s information to see if he/she has been enrolled in the department. | |
| **Actors** | Faculty members | |
| **Stakeholders** | Department Head, Student | |
| **Preconditions** | If the faculty member does not have the student’s id or name, the student must give out his/her student id or name to the faculty member. | |
| **Postconditions** | Faculty member should be able to determine whether the student is enrolled in the department or not. | |
| **Activities Flow** | **Actor Action** | **Systems Response** |
| 1. User inputs the student id or name   3. User successfully knows if the student is enrolled in the department or not | 1. Application displays a list of students with the name and student id. |
| **Exception Conditions** | 1. User types in the wrong student id or name | |

*Use Case 2 - Suggesting student courses to take*

|  |  |  |
| --- | --- | --- |
| **Use Case Scenario** | A student wants to know what course he/she can take next term. A faculty member advises the student. | |
| **Actors** | Faculty Member,  Student | |
| **Related Use Cases** | Use Case 1 – Searching for Student information | |
| **Stakeholders** | Department Head | |
| **Preconditions** | If the faculty member does not have the student’s id or name, the student must give out his/her student id or name to the faculty member. | |
| **Postconditions** | Faculty member should be able to advise the student on what courses he/she can take for the next term | |
| **Activities Flow** | **Actor Action** | **Systems Response** |
| 1. Faculty member inputs the student’s id or name  3. Faculty member clicks on the correct student on the list  5. Faculty member inputs the year and next term  7. Faculty member advises student | 2. Application displays a list of students with the name and student id  4. Application displays a list of courses the student is eligible to take  6. Application filters the list of courses based on the year and term. |
| **Exception Conditions** | 1. User types in the wrong student id or name | |

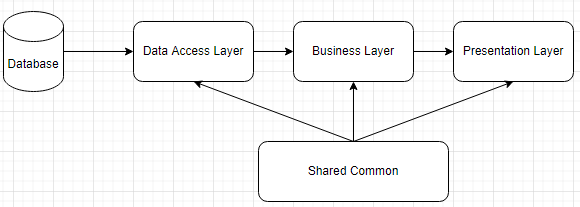
**Project Plan**



**Project Design**

Structural Design

*High level structural design*



Data Access Layers - handles all the queries required to grab data from the database.

Business Layer – The functions needed for the application such as Get Recommended Courses for a student and Get Student Information and stored in this layer. It uses the data from the data access layer to carry out these functions.

Presentation Layer – This layer is where the User Interface is created. The results of the functions from the business layer are displayed here.

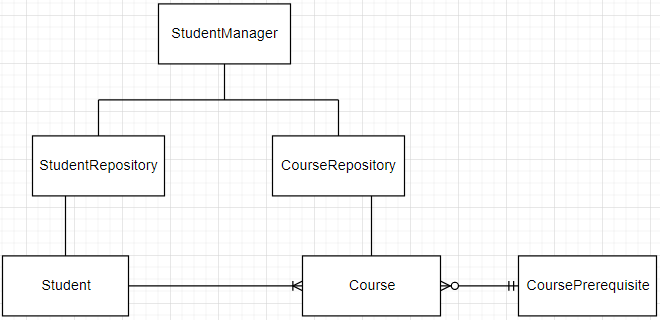
Shared Common – All entities, interfaces, modules, etc... required by the 3 layers are stored here. All 3 layers can independently access shared common.

*Design Concepts*

Every software process is characterized by basic concepts along with certain practices or methods. Methods represent the manner through which the concepts are applied. As modern technology replaces older technology, many changes occur in the methods that are used to apply the concepts for the development of software. However, the fundamental concepts underlining the software design process remain the same, some of which used in the application are

* Abstraction
* Architecture
* Patterns
* Modularity

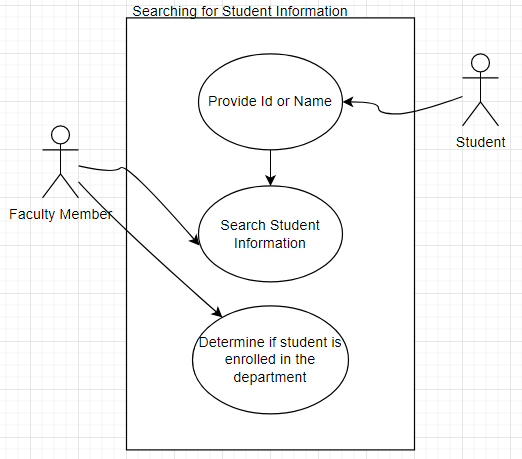
*Class Diagrams*



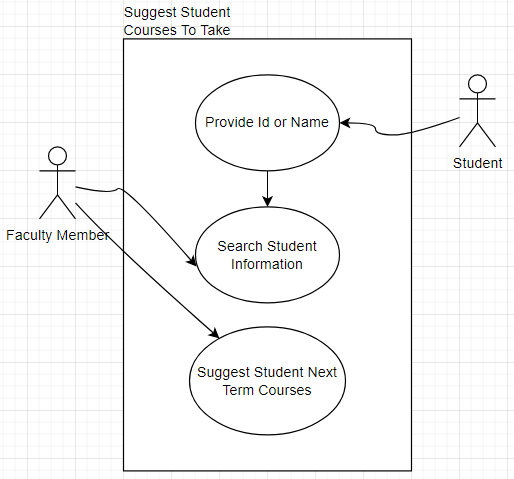
Behavioral Design

*Use Case Diagrams*

Use Case 1

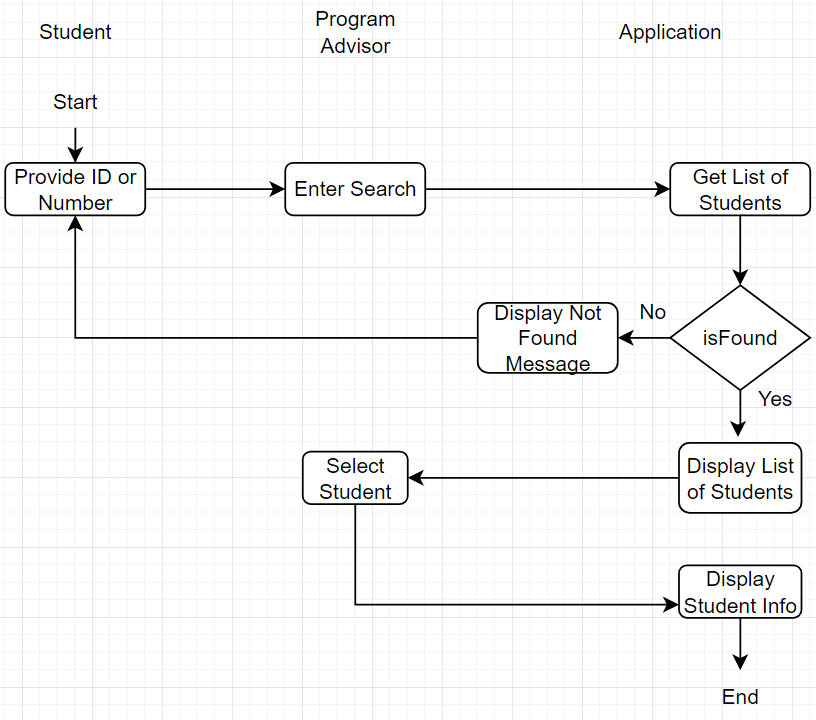


Use Case 2

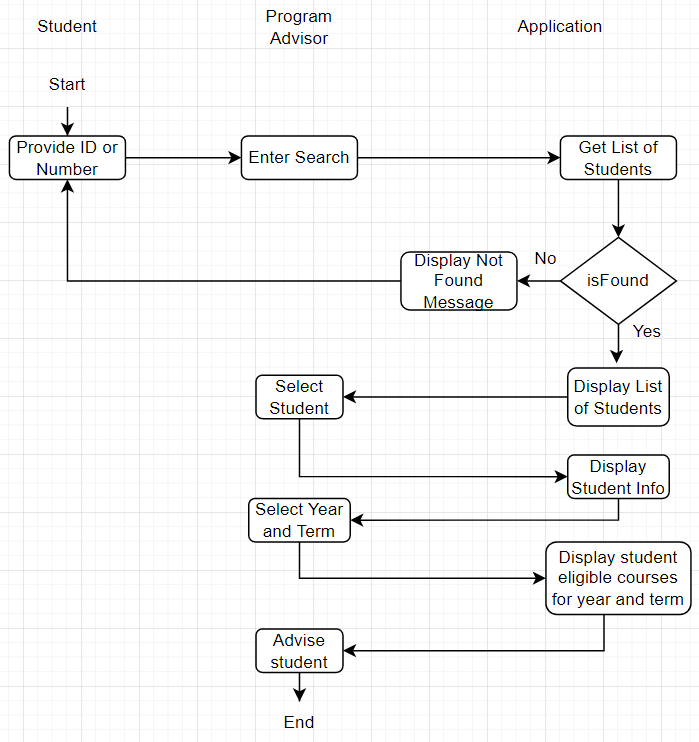


*Activity Diagrams*

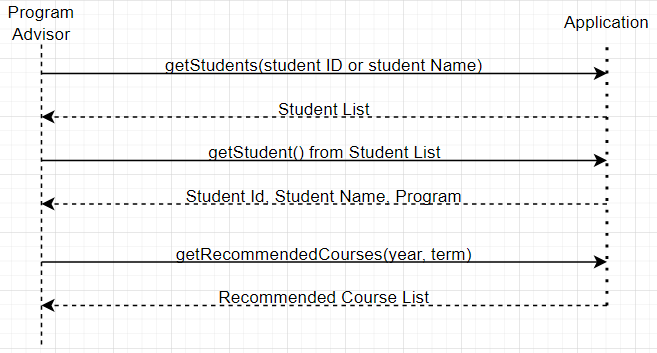
Searching for student information



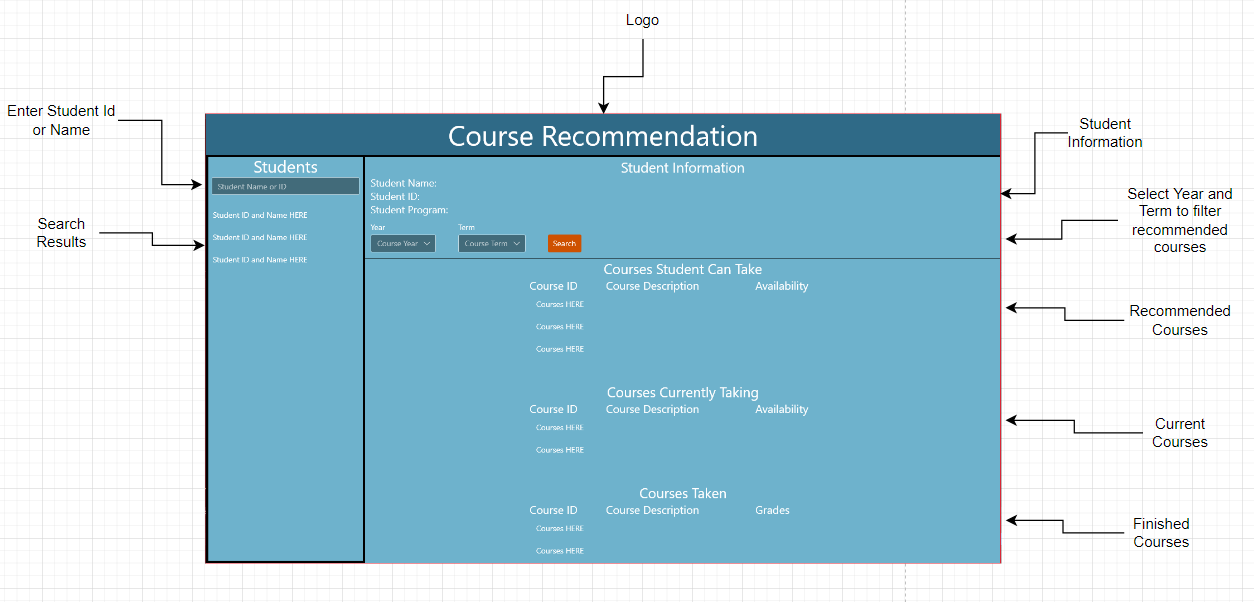
Suggest student courses to take



*Sequence Diagram*



Wireframe



Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Steps | Data | Expected Result | Status |
| Verify if getStudents() return list of students | 1. Enter student id or name  2. Press Enter Key | Student Id: 10564  Student Name: Matt | Show 10564 in the list or display all students with the name Matt | Pass |
| Verify if student information displays | 1. After searching, select a student from the list | Student Id: 10564  Student Name: Matt | Show information about student Matt 10564 | Pass |
| Verify if getRecommendedCourses() works | 1. Select year and term  2. Click Search | Year: 2023  Term: 2 | Display all eligible courses for a specific student filtered with year and term | Pass |
| Verify if getRecommendedElectiveCourses() works | 1. Select year and term  2. Click Search  3. Select Electives | Year: 2023  Term: 2 | Display all eligible elective courses for a specific student filtered with year and term | Pass |
| Verify if getAllowedStudents() works | 1. Enter a course ID  2. Enter a Program ID  3. Click Search | CourseID: ACC201 ProgramID: AP | Display all eligible students for the course | Pass |

**Project Implementation**

Hardware Platform

Application is only available on PC, Laptop, and Tablets.

Software Platform

Application is only available on Windows 10 Operating Systems and above.

Language / Framework / Libraries

Developed using C# with Microsoft.NETCore.App and Microsoft.Windows.SDK.NET.Ref.

Packages include:

* Microsoft.ProjectReunion(0.8.6)
* Microsoft.ProjectReunion.Foundation(0.8.6)
* Microsoft.ProjectReunion.WinUI(0.8.6)
* System.Data.SQLClient(4.8.3)

Development Tools

The project was developed using Visual Studio 2019 IDE loaded with ASP.NET and web development, .NET desktop development, Universal Windows Platform development, and .NET cross-platform development.

**User’s Guide**

Installation

*System requirements and recommended system*  *specs*

Recommended Requirements:

* 2.8Ghz processor. Quad-core or better recommended
* 6-8GB of RAM
* 1-2GB of available space.

Minimum Requirements:

* 1.8Ghz processor.
* 2GB of RAM
* 800MB of available space

*Installing the application*

Search for the application on the Microsoft store, click Get. The application will automatically download and install on your device.

How Tos

*Opening the app*

To open the app, simply double click on the icon on your desktop. If this does not work, right click on the icon and click Open.

*Searching for students*

Enter the Student ID or Name on the search bar located on upper left side of the screen then press the Enter key. If the search was successful, it will display a list of the students with the given name or ID. Select the searched student on the list to display the student’s information.

*Getting the suggested courses for a student*

There will be a drop-down list named Year and Term after successfully searching for a student and displaying the student’s information. Select a year and term for which you want to filter the courses with. Click search and it should display a list of courses the student is eligible to take given the year and term.

**Conclusion**

Lessons learned

This project has taught me a lot about the process of developing a software. Every milestone achieved gave me a feeling of success and satisfaction, but of course it was not possible without the failures and mistakes I have encountered along the way. I have learned that designing a database is quite challenging but ultimately rewarding in the end. The queries I made were perfected through a trial-and-error method.

The structural design and architecture made a lot of sense because it kept everything grouped together. If I had to find something such as a cause of an error, I instantly knew where to look. The windows UI was something new to me and I had to do some research to properly maneuver around it.

In the end, all these amounts to me improving my skills in properly developing a software.

Known Issues

*Existing problems*

One problem I believe needs to be addressed is inability to put in information from the application to the database. Information needs to be manually inputted in the database. The process of manually typing information into the database is tricky and might cause the application to crash if not done properly.

*Open or half-done features*

There is a feature where you input a course and it would display all the student eligible to take the course. The feature works perfectly on the back-end side, but it was not implemented on the front-end side of the application.

Future Improvements

An additional feature I would like to add in the application is the ability to put information from the app to the database. Adding a student or adding a course for example. I believe this would complete the application and fulfill its purpose.

**References**

Centralized vs Distributed Database

(<https://www.geeksforgeeks.org/difference-between-centralized-database-and-distributed-database/>)

Continuous Delivery vs Continuous Deployment (<https://harness.io/blog/continuous-delivery/continuous-delivery-vs-continuous-deployment/>)

Software Design & Concepts

(<https://ecomputernotes.com/software-engineering/principles-of-software-design-and-concepts#Developing_a_Design_Model>)