Ethan Raby

[EKU – CSC 440: Applied SOFTWARE ENGINEERING]  [October, 29, 2021]

grade management system

Table of Contents

[Table of Figures 2](#_Toc89187698)

[I. Introduction 3](#_Toc89187699)

[A. Problem Statement 3](#_Toc89187700)

[B. Proposal 3](#_Toc89187701)

[II. System Description 3](#_Toc89187702)

[III. System Requirements 3](#_Toc89187703)

[A. Functional Requirements 3](#_Toc89187704)

[B. Non-Functional Requirements 9](#_Toc89187705)

[IV. Use Case Diagram 10](#_Toc89187706)

[V. Data Flow Diagram 11](#_Toc89187707)

[VI. Structure Chart 14](#_Toc89187708)

[VII. Data Store (Database) Design 14](#_Toc89187709)

[A. ER Diagram 14](#_Toc89187710)

[B. Table Schema 15](#_Toc89187711)

[VIII. Algorithm Design 15](#_Toc89187712)

[IX. Conclusion 16](#_Toc89187713)

[X. Data Dictionary 16](#_Toc89187714)

# Table of Figures

Figure 1: The Login Screen 3

Figure 2: The Main Menu 4

Figure 3: Invalid File or File Name 4

Figure 4: Invalid File Contents 5

Figure 5: Successful Course Upload 5

Figure 6: The Edit Grade Screen, Prior to Selecting Course 6

Figure 7: Invalid StudentID at Editing Grade 6

Figure 8: The Edit Grade Screen, After Selecting Course 7

Figure 9: Successful Grade Change 7

Figure 10: Invalid StudentID at Transcript Printing 8

Figure 11: Printing a Student Transcript 8

Figure 12: Excel Contents Example 9

Figure 13: Grade Report Use Case Diagram 10

Figure 14: Level 0: Context Level 11

Figure 15: Level 1 from P0 (Delete record is performed in modify) 11

Figure 16: Level 2 from P1 12

Figure 17: Level 2 from P2 12

Figure 18: Level 2 from P3 13

Figure 19: Level 2 from P4 13

Figure 20: Structure Chart of the System 14

Figure 21: ER Diagram of the Database 14

Figure 22: Decision Tree for the program 15

# Introduction

## Problem Statement

EKU’s Registrar Office plans to provide a transcript program for the teachers to use. The office needs a software system that can provide the following operations: Add new grades, edit grades, and create a transcript. The transcript should be printed out and include the student name, ID, overall GPA, and a list of all courses the student has taken.

## Proposal

We propose the following software system to help EKU with the desired functionality.

# System Description

The system will allow teachers to upload the grades of an entire class into the database. The grades must be uploaded in an Excel file. It will then allow any grade to be updated and changed or deleted if necessary. The system will only accept grades in the form of A, B, C, D, F. Each course should have a credit hour attached to it between 1 and 5 and will use it to calculate an overall GPA for the printable transcript.

# System Requirements

## Functional Requirements

1. The system shall allow a user to login to the system.
   1. The system shall prompt the user to enter a username and password.

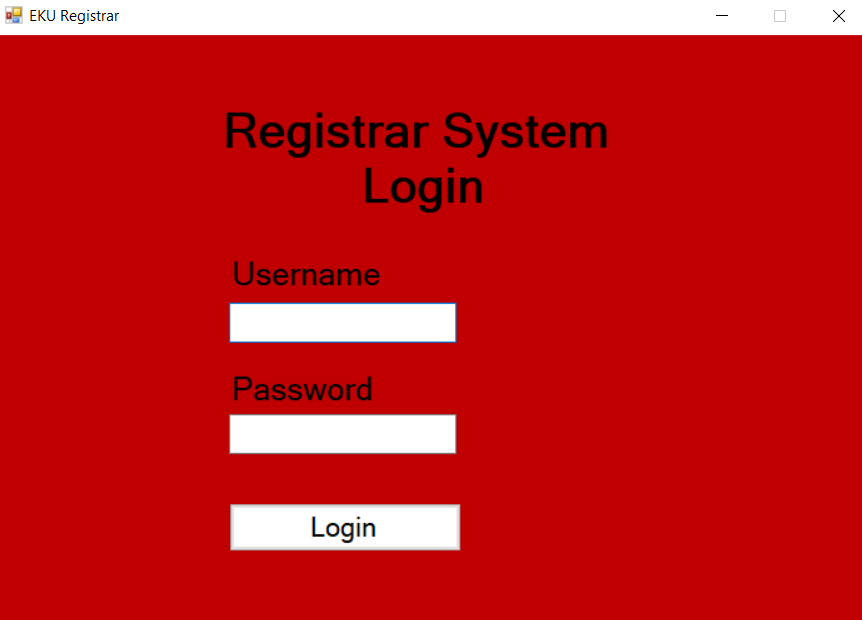


Figure 1: The Login Screen

* + 1. If the login information is invalid, the system shall display an “Invalid Username or Password” error message to the user and return to Step 1.1.
    2. If the login is valid, the system shall continue to Step 1.2.
  1. The system shall display the main menu.

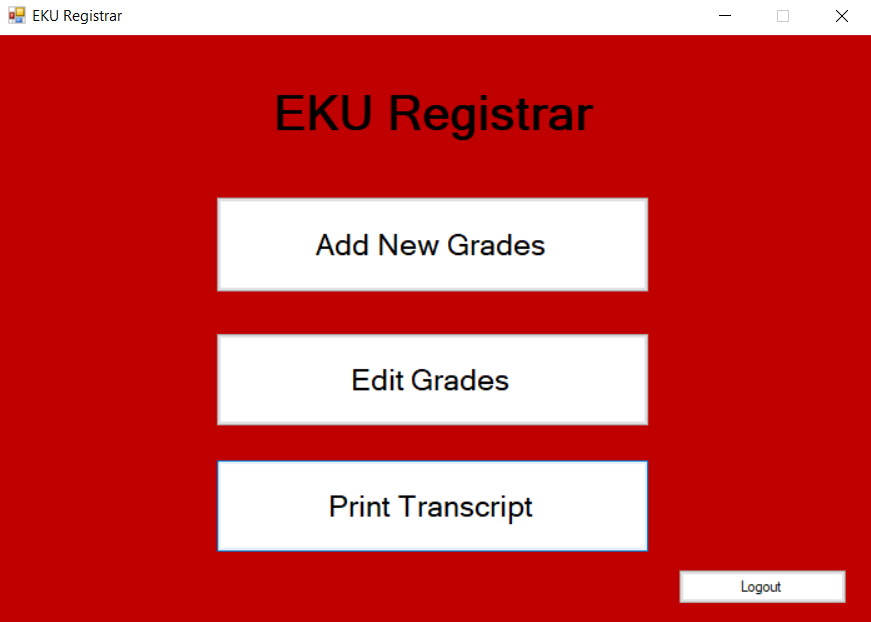


Figure 2: The Main Menu

1. The system shall allow a user to add new grades to a course for each student to the database.
   1. The user shall click the “Add New Grades” button from the main menu.
   2. The system shall open a file explorer browser.
   3. The user shall navigate to the appropriate Excel file and select it.
      1. If the file is not valid, an “Invalid File” error message will be displayed, and the system shall return to Step 2.1.

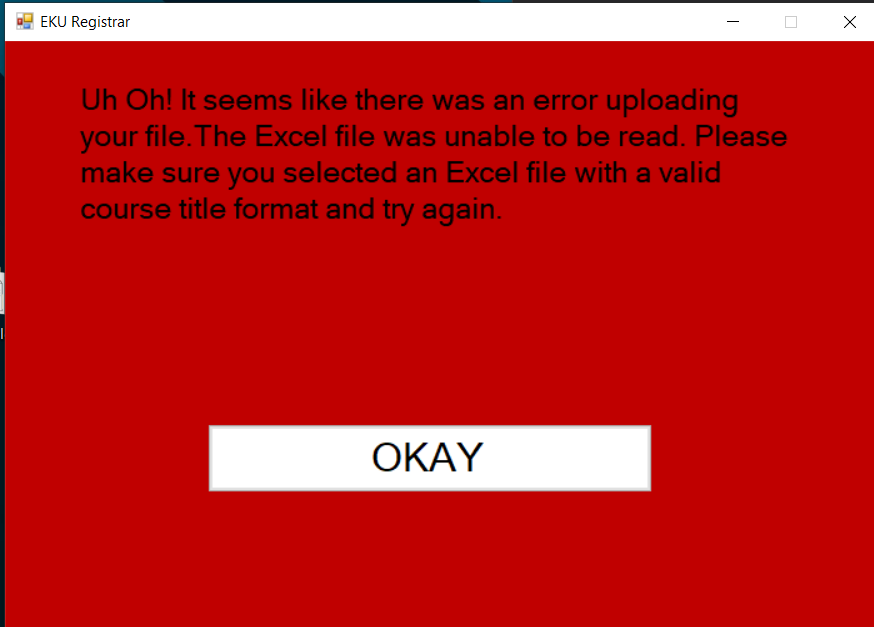


Figure 3: Invalid File or File Name

* + 1. If the file is valid, the system shall accept the file and continue to Step 2.4.
  1. The system shall read the file.
     1. If the format is incorrect, the system shall display an “Invalid File Format” error message and return to Step 2.1.

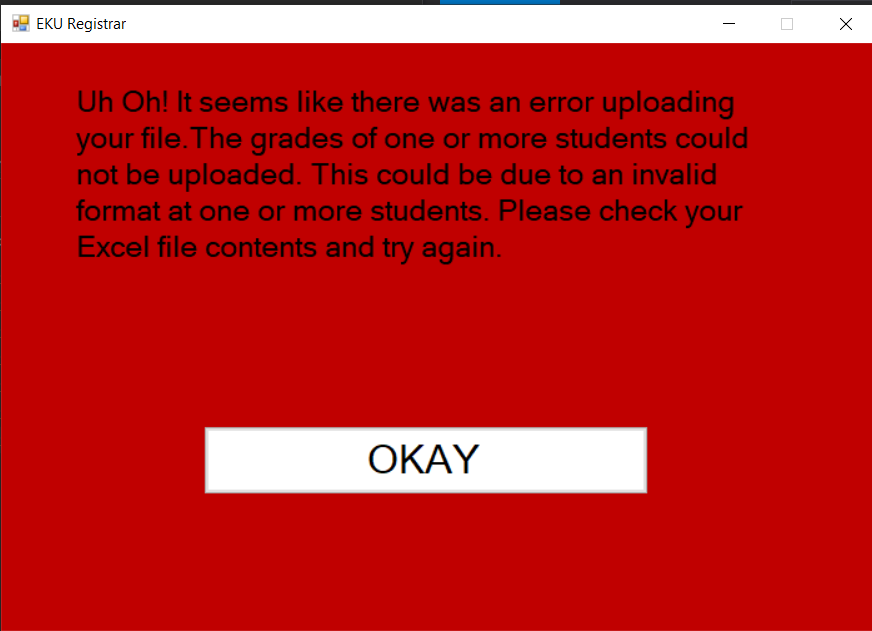


Figure 4: Invalid File Contents

* + 1. If the file is in valid format, the system shall display a “Successful Upload” message to the user and update the database.

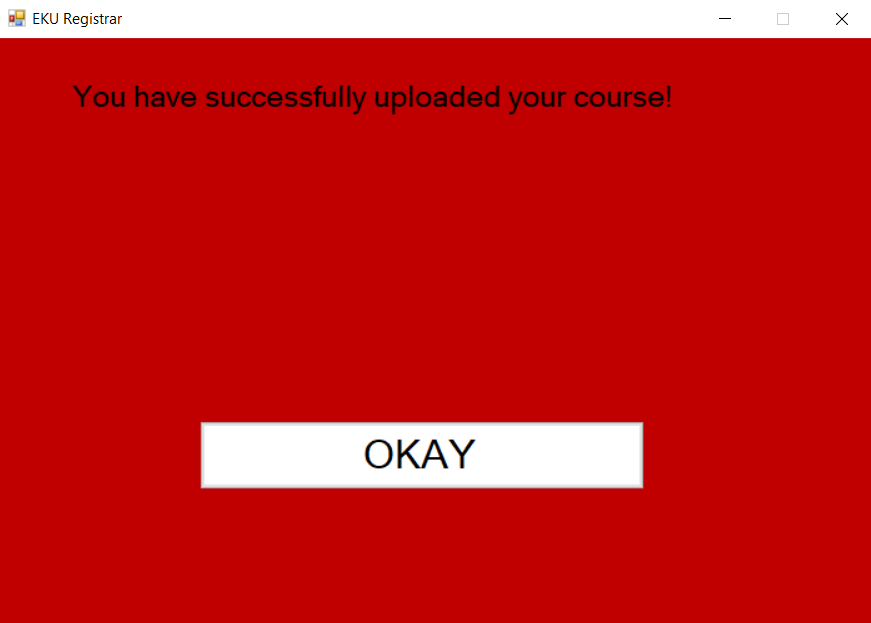


Figure 5: Successful Course Upload

* 1. The system shall return to the main menu.

1. The system shall allow a user to edit a grade for a student.
   1. The user shall select the “Edit Grade” button from the main menu.
   2. The system shall prompt the user for the student ID.

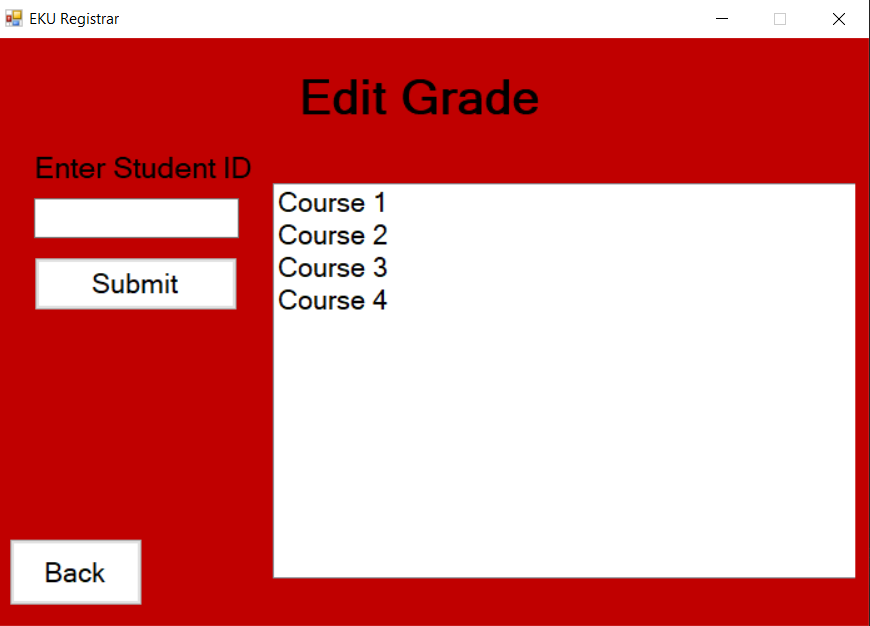


Figure 6: The Edit Grade Screen, Prior to Selecting Course

* 1. The user shall enter the appropriate user ID.
     1. If the ID is invalid, the system shall display an “Invalid Student ID” error message to the user and return to Step 3.2.

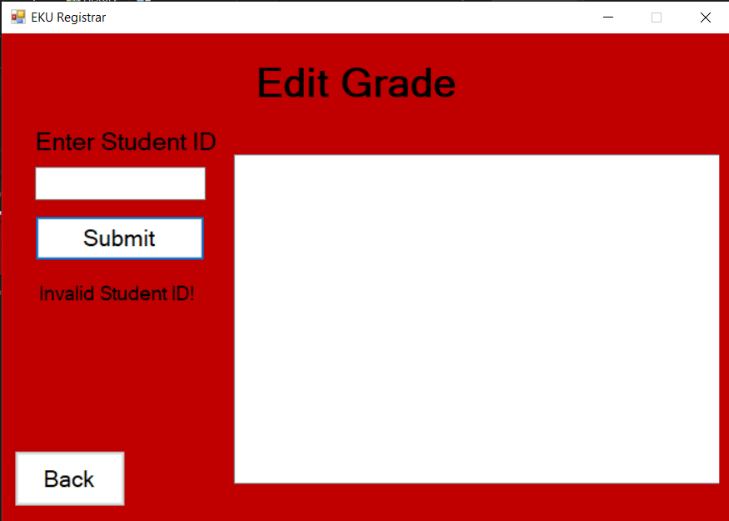


Figure 7: Invalid StudentID at Editing Grade

* + 1. If the ID is valid, the system shall continue to Step 3.4.
  1. The system shall display the students’ current courses. (The list will remain empty until a valid student ID has been entered).
  2. The user shall select the course they wish to edit.
  3. The system shall display the student name, current grade, and course information.

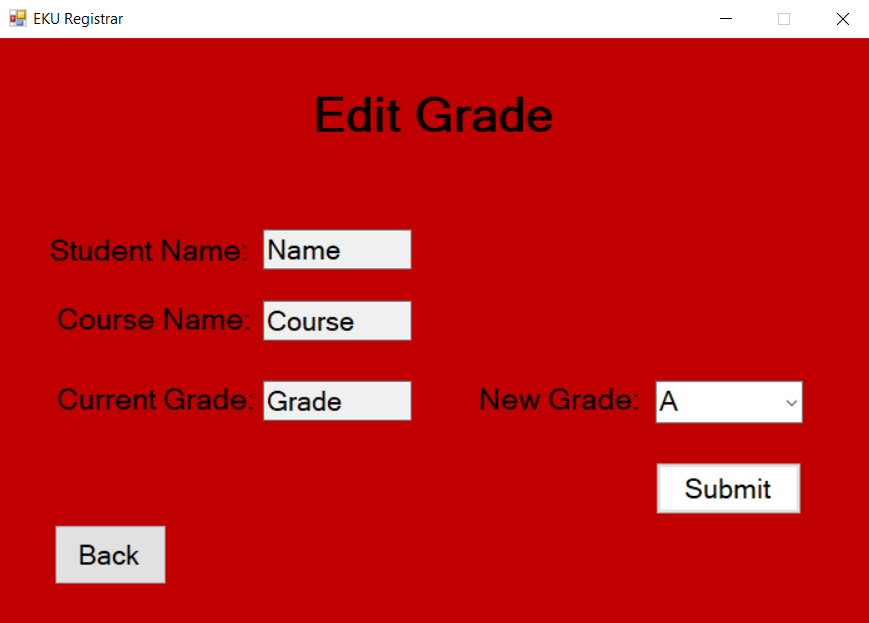


Figure 8: The Edit Grade Screen, After Selecting Course

* 1. The user shall change the grade.
  2. The user shall press the “Submit” button.
     1. If the grade is a valid change, the system shall update the database and display a “Successful Edit” message and continue to Step 3.9.

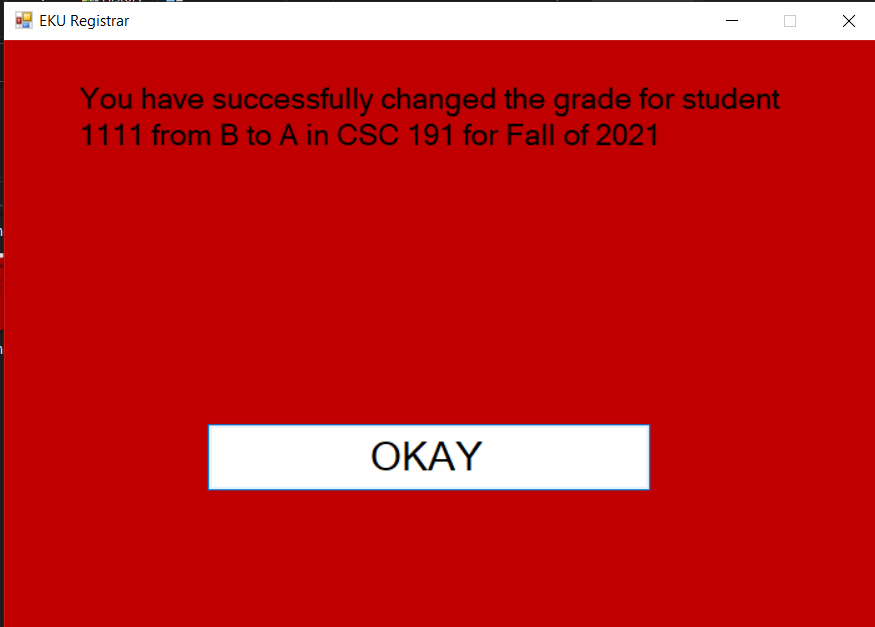


Figure 9: Successful Grade Change

* + 1. If the grade change fails, the system shall display an “Error Updating Grade” message and return to Step 3.1.
  1. The system shall return to Step 3.1.

1. The system shall allow a user to print a transcript for a student.
   1. The user shall select the “Print Transcript” button from the main menu.
   2. The system shall prompt the user for the student ID.
   3. The user shall enter the appropriate student ID.
      1. If the ID is invalid, the system shall display an “Invalid ID” error message to the user and return to Step 4.2.

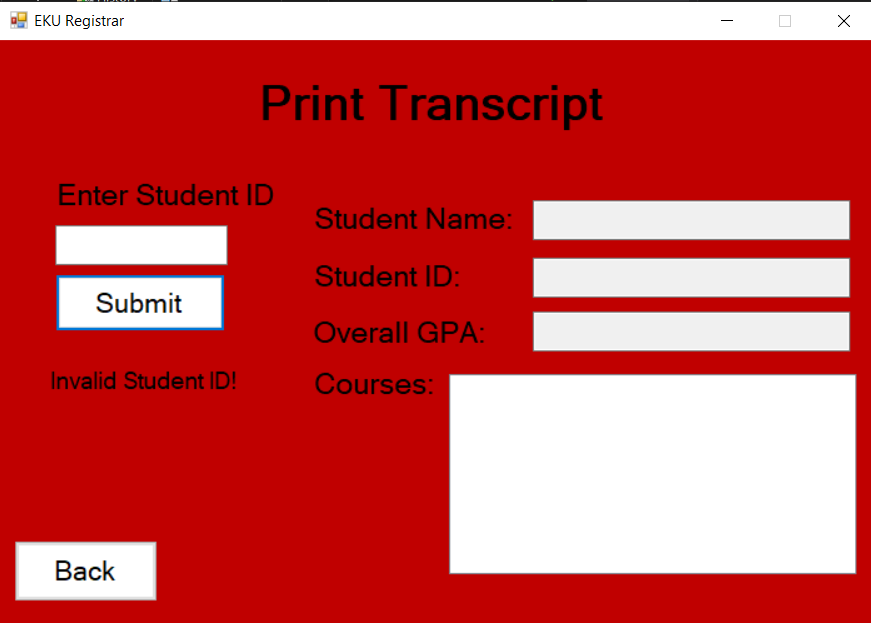


Figure 10: Invalid StudentID at Transcript Printing

* + 1. If the ID is valid, the system shall continue to Step 4.4.
  1. The system shall retrieve the appropriate information from the database.
  2. The system shall print the transcript including the student name, student ID, overall GPA, and a list of all completed courses.

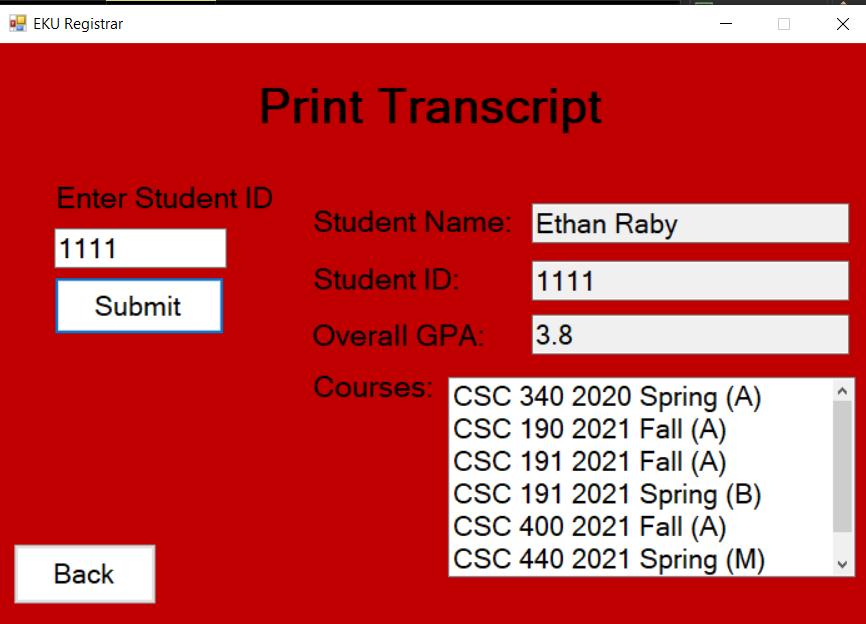


Figure 11: Printing a Student Transcript

* 1. The system shall return to the main menu.

1. The system shall allow a user to log out of the system.
   1. The user shall select the “Logout” button from the main menu.
   2. The system shall display a “Goodbye” message to the user.
   3. The system shall return to the login screen.

## Non-Functional Requirements

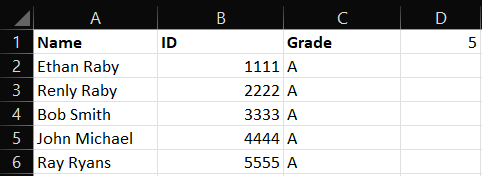
1. The grades for a student should be in A/B/C/D/F/M format.
2. M is a grade option indicating a missing grade that will **not** be included in the calculation of GPA.
3. W is a grade option to denote a student dropped a class. \*Note: A withdrawn course does not count towards a student’s GPA.
4. Once a student is withdrawn from a course, the ability to change the grade is removed, although it is still present on the transcript.
5. A withdraw must be done through the edit grade function.
6. The username and password must consist of integers only, no longer than 9 digits long.
7. Each course should have the number of credit hours, ranging from 1 to 5 inclusive, indicated in cell D1.
8. If the number of credit hours is not specified, the system will default to **three** credit hours. \*Note that an invalid value will **not** result in an error and will be treated as if no hour was specified.
9. Each Excel file should follow the following format: **[Course Prefix] [Number] [Year] [Semester].** Example: CSC 440 2021 Fall
10. The contents of the Excel file should follow the following format: 

Figure 12: Excel Contents Example

\*Note that if a student’s information is incorrectly formatted, they will be skipped, however the other students will be added. Once the errors are fixed, the file can be reuploaded in its entirety.

# Use Case Diagram

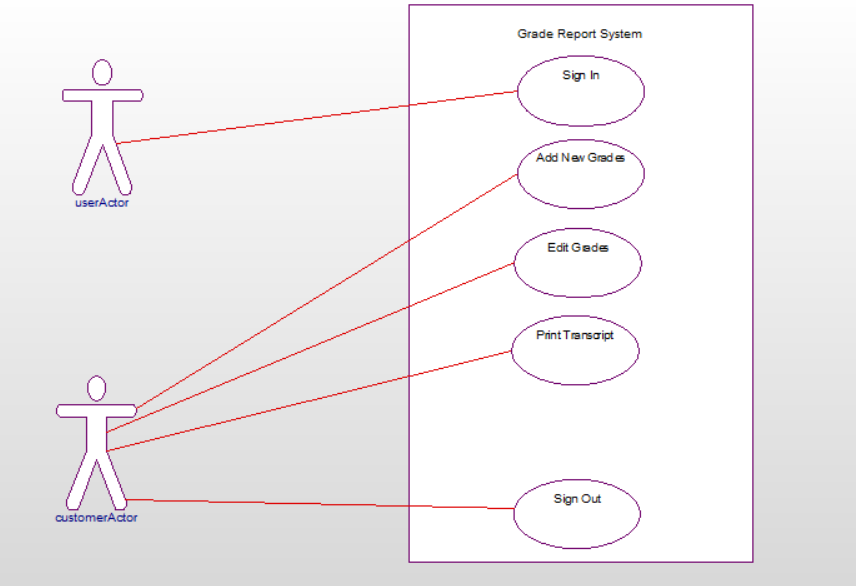


Figure 13: Grade Report Use Case Diagram

A use case diagram explains all the functionality the system can perform. As seen in the image above, all of the use cases were derived from the problem statement in the System Description (II).

# Data Flow Diagram



Figure 14: Level 0: Context Level



Figure 15: Level 1 from P0 (Delete record is performed in modify)



Figure 16: Level 2 from P1



Figure 17: Level 2 from P2



Figure 18: Level 2 from P3



Figure 19: Level 2 from P4

The data flow diagram (DFD) maps the flow of information for a process or system. They are used to analyze a system, beginning at the basic operation of a system, with each level becoming more and more advanced.

# Structure Chart



Figure 20: Structure Chart of the System

The structure chart is built from the DFD and helps in the design of the system. The different rectangles represent different functions performed by the system, constructed in a top-down manner. The connecting lines show a relationship between the processes, where the one above it will invoke the one underneath it.

# Data Store (Database) Design

## ER Diagram

Diagram

Description automatically generated

Figure 21: ER Diagram of the Database

The ER diagram is the roadmap of a database, showing all of the tables and relations that will be used. Each rectangle is an entity, containing the attributes in rectangles. The diamonds represent a relationship between two entities.

## Table Schema

User(ID, PIN)

Student(ID, name, overallGPA)

Course(coursePrefix, courseNum, year, semester, hours)

Takes(studentID, coursePrefix, courseNum, year, semester, grade)

The table schema is a written representation of the ER diagram, where underlined words are considered ‘primary keys’ to the data. The ‘foreign key’ helps keep a relationship between two different tables. This schema is used to make the tables in a database. Due to the nature of the system, there is no relation between the Student table and the User table. This is because any user can access any student, so we do not need to keep track of what user can manage what student.

# Algorithm Design



Figure 22: Decision Tree for the program

The decision tree above shows the possible outcomes for an action. A decision tree shows the steps that may be taken, and each end node, or leaf, has the final action in that path. The algorithm implemented in this program will be able to handle these possibilities. For every action, there is a possibility of some error occurring, or for the process to be completed. The errors will be handled by displaying the type of error that has occurred and returning the previous step until the correct path is taken.

# Conclusion

All desired featured have been successfully implemented, as well as the design of a user-friendly interface. The system allows a user to sign in, import grades from a file, add new grades, edit existing grades, and print a transcript. These functionalities are described in section III. Two other diagrams, found in section V and VI describe how the data flows through the system, as well as the functions that will be called. These sections also contain a short description of the diagram, as well as its role in design. The database design, including tables and a graphical representation, has been listed in section VII. The program and GUI were written in C# using Visual Studio; however, the implementation was done functionally, not object orientated. The database was implemented using HeidiSQL.

# Data Dictionary

Attribute – A piece of information which determines the properties of an item in the database.

Entity – Any item or person that has data which should be stored into the database.

Foreign Key – Column(s) that correspond to the primary key(s) in another table to keep a relationship between entities.

Primary Key – A unique piece of data in a table that can identify an entire row of data. An example of this would be your social security number, that will identify you, and only you.

Relation – An association between two tables.

Table – A set of data elements in a database. Each row represents a different entity, and each column represents attributes of that entity.