Matthew Abney, Ethan Vaughan, Chris Ward

CSC 440 – Applied software engineering

Team Project: grade and gpa calculator and degree progress system

September 14, 2019

Requirements specifications report

Table of Contents

[Table of Figures 2](#_Toc22410035)

[I. Introduction 3](#_Toc22410036)

[A. Problem Statement 3](#_Toc22410037)

[B. Proposal 3](#_Toc22410038)

[II. System Description 3](#_Toc22410039)

[III. System Requirements 4](#_Toc22410040)

[A. Functional Requirements 4](#_Toc22410041)

[B. Nonfunctional Requirements 12](#_Toc22410042)

[IV. Use Case Diagram 13](#_Toc22410043)

[V. Class Diagram 15](#_Toc22410044)

[VI. Sequence Diagrams 16](#_Toc22410045)

[VII. Activity Diagrams 25](#_Toc22410046)

[VIII. State Diagrams 26](#_Toc22410047)

[IX. Database Diagram 26](#_Toc22410048)

[X. Conclusion 26](#_Toc22410049)

[XI. Data Dictionary 27](#_Toc22410050)

# Table of Figures

[Figure 1 Adding a Grade 4](#_Toc22410051)

[Figure 2 Submitting a Grade to the Database 4](#_Toc22410052)

[Figure 3 Performing What-If Grade Calculation 5](#_Toc22410053)

[Figure 4 Viewing What-If Results 5](#_Toc22410054)

[Figure 5 Delete Grade 6](#_Toc22410055)

[Figure 6 Choose a Grade to Delete 6](#_Toc22410056)

[Figure 7 GPA Calculator Button 6](#_Toc22410057)

[Figure 8 Student GPA Screen 7](#_Toc22410058)

[Figure 9 Completed Classes Overview 7](#_Toc22410059)

[Figure 10 Adding a Completed Class 8](#_Toc22410060)

[Figure 11 Success Message 8](#_Toc22410061)

[Figure 12 Modifying a Grade 8](#_Toc22410062)

[Figure 13 Deleting a Current Class 9](#_Toc22410063)

[Figure 14 Adding a Class 9](#_Toc22410064)

[Figure 15 Info for a new Current Class 10](#_Toc22410065)

[Figure 16 Student Progress Overview 10](#_Toc22410066)

[Figure 17 What-If GPA Results 11](#_Toc22410067)

[Figure 18 Viewing Current Courses 12](#_Toc22410068)

[Figure 19 Use Case Diagram 13](#_Toc22410069)

[Figure 20 Class Diagram of Grade, GPA, and Degree Tracking System 15](#_Toc22410070)

[Figure 21 Sequence Diagram for Viewing Assignment Grades 16](#_Toc22410071)

[Figure 22 Sequence Diagram for Adding an Assignment Grade 17](#_Toc22410072)

[Figure 23 Sequence Diagram for Deleting an Assignment Grade 17](#_Toc22410073)

[Figure 24 Sequence Diagram for Modifying an Assignment Grade 18](#_Toc22410074)

[Figure 25 Sequence Diagram for Performing What-If Grade Calculation 18](#_Toc22410075)

[Figure 26 Sequence Diagram for Adding a Current Course 19](#_Toc22410076)

[Figure 27 Sequence Diagram for Deleting a Current Class 19](#_Toc22410077)

[Figure 28 Sequence Diagram for Adding a Completed Course 20](#_Toc22410078)

[Figure 29 Sequence Diagram for Deleting a Completed Course 21](#_Toc22410079)

[Figure 30 Sequence Diagram for Modifying a Completed Course 22](#_Toc22410080)

[Figure 31 Sequence Diagram for Tracking Degree Progress 23](#_Toc22410081)

[Figure 32 Sequence Diagram for Performing What-If GPA Audit 24](#_Toc22410082)

[Figure 33 Activity Diagram for Viewing Assignment Grades 25](#_Toc22410083)

[Figure 34 ER Diagram 26](#_Toc22410084)

# Introduction

## Problem Statement

Computer science students at Eastern Kentucky University need to find out their grades in certain classes. They will have their grades on several completed assignments, with more assignments to come. They need to know what grade(s) they will need on future assignment(s) in order to receive, say, 89.5% overall in a course. Perhaps they would like to know what grade they will get overall in a course, assuming they make, say, 75% on remaining assignments and/or tests.

Additionally, computer science students needto keep track of their GPA. They may want to know their GPA in different areas, such as their major GPA, overall GPA, and GPA in supporting courses. And, before a semester is over, a computer science student would like to know what effects different final grades will have on the student’s overall GPA.

One final piece of information that computer science students would like to keep track of is their progress towards their concentration specific C.S. degree. C.S. students would like to track which classes they need to take (general education courses, supporting courses, core courses, etc.).

## Proposal

My solution to the needs of computer science students at Eastern Kentucky University is a grade and GPA calculator, with added functionality to keep track of a student’s progress towards degree completion. The grade calculator would allow students to record grades on assignments/tests and perform “what-if” scenarios, showing them what grades they would need on remaining coursework in order to receive a certain final grade overall (such as 88%) and what grade they would end up with in a course if they received a specified grade on remaining coursework.

For the GPA portion of my application, the application would calculate a student’s overall GPA as they enter different final grades. The app would show a student what GPA he/she would have if certain final grades were achieved in current or future courses.

Lastly, this app would track a student’s progress towards a degree concentration. It is tailored specifically for computer science students at Eastern Kentucky University, taking one of EKU’s C.S. concentrations.

# System Description

The project is to build a system that allows students to input grades in for individual classes to help calculate the grade in that class, also allowing to calculate GPA, and know how close they are to fulfilling their majors’ requirements. The system shall give the student options to add or remove grades as needed while doing the same for classes current or completed. The system shall allow the student to modify the grades or classes. The system shall allow the student to perform what-if scenarios on grades and overall GPA.

# System Requirements

The system is required to give information to EKU computer science students regarding their degree progress, GPA, and grades in current classes. The system tracks the student’s GPA as he/she submits final grades, and allows a student to see what GPA he/she would have if he/she received certain final grades in future classes (that is, this system can perform “what-if” calculations). The system uses the information about completed classes to show a user his/her progress towards the different C.S. concentrations that EKU offers (users may view their progress under the different concentrations, so a user is not “stuck” viewing progress for merely one concentration). Users may submit their grades for assignments, quizzes, tests, etc. in current classes and view their overall grade in the courses. Then, they may ask the calculator what grade they would need on remaining coursework in order to finish with a certain grade overall.

## Functional Requirements

R1. The system shall allow a user to submit grades for completed assignments/tests.

* 1. After the user has chosen a course to view (see R12), he/she shall click the “Add Grade” button.
  2. The system shall create a new row that the user can place the grade information into.



Figure Adding a Grade

* 1. The user shall enter the assignment/test description (e.g. “Assignment 1”), the grade received on the assignment (as a percentage), and the assignment’s weight (as a percentage).
  2. The user shall press the “Submit Changes” button.



Figure Submitting a Grade to the Database

* 1. The system shall recalculate the user’s overall grade in the course and send the new grade information to the database.

R2. The system shall allow a user to figure out what grades will be needed on remaining assignments in order to receive a desired overall grade in a course. Additionally, the system shall tell the user what grade he/she will have in the course, given a certain grade on remaining coursework.

2.1. The system shall display the user’s grades for that class.

2.1.1. The system shall show the assignment names, grades received for the assignments, and weight of the assignments.

2.1.2. The system shall calculate the user’s current grade in the course (calculated using grades the user has already submitted) as well as the “percentage points” earned for that class.

2.2. The user shall enter the desired grade, as a percentage, in the box “What if I made \_\_\_ on remaining coursework?”.



Figure Performing What-If Grade Calculation

2.2.1. This part of the what-if calculation will figure out what grade the student would end up with in the course if he/she made the specified grade on all remaining coursework.

2.3. The user shall enter the desired grade, as a percentage, in the text box “What do I have to make on remaining coursework to get \_\_\_ overall?”

2.3.1. This calculation will show the user what grade he/she must make on remaining coursework in order to receive specified grade overall in the course.

2.4. The system shall perform the calculations and display the results.



Figure Viewing What-If Results

R3. The system shall allow a user to delete a grade.

3.1. The user shall click the “Delete a Grade” button.



Figure Delete Grade

3.2. The system shall unhide a combo box that contains all the grades for the class.

3.2.1. The system shall also change the text in the button from “Delete a Grade” to “Delete”.



Figure Choose a Grade to Delete

3.3. The user shall select the grade from the combo box that he/she wishes to delete.

3.4. The user shall click the “Delete” button.

3.5. The system shall hide the combo box, reset the text in the button to “Delete a Grade”, remove the deleted grade from the list of grades, recalculate the user’s current grade and percentage points in the course, and remove the grade from the database.

R4. The system shall allow a user to add a final grade for a completed class.

4.1. The user shall click on the “GPA Calculator” button from the start screen.



Figure GPA Calculator Button

4.2. The system shall calculate the user’s GPA by fetching the user’s final grades.

4.3. The user shall click on the “Add/View Classes” button.



Figure Student GPA Screen

4.4. The system shall display a combo box with the user’s completed classes.

4.4.1. When the user selects a new class from the combo box, that class’s credits, the user’s final grade, and the semester taken shall be displayed.



Figure Completed Classes Overview

4.5. The user shall click the “Add a Class” button.

4.6. The system shall prompt the user to enter the information about the class (class name, such as “ENG101”, credits, final grade (A, B, etc.), category (such as “Gen ed element 6”), and semester, such as “fall 2013”).



Figure Adding a Completed Class

4.7. The user shall click the “Submit” button.

4.8. The system shall add the class to the combo box of classes.

4.8.1. The screen for adding a class will disappear and a success message will be displayed.



Figure Success Message

4.8.2. The class and its information will be sent to the database.

4.8.3. The user’s GPA will reflect the new final grade submitted.

R5. The system shall allow a user to modify a grade.

5.1. The user shall click on the part of the grade (assignment/test name, grade received, weight) and change the text as needed.



Figure Modifying a Grade

5.2. The user shall click on the “Submit Changes” button (see Figure 2).

5.2.1. If the user only wanted to see what effect the modified grade would have without submitting it to the database, he/she shall click on the “Refresh Calculations” button and the system shall display what grade the user would have overall given the modified grade(s).

5.3. The system shall refresh the user’s current grade in the course. The grade modification(s) shall be sent to the database.

5.4. The system shall display a success message to the user.

R6. The system shall allow a user to delete a current class.

6.1. The user shall click the “Delete Class” button.



Figure Deleting a Current Class

6.1.1. The “Delete Class” button shall change to “Confirm”.

6.2. The user shall click on the “Confirm” button, to confirm that the class should be deleted.

6.3. The system shall remove the class from the screen, display a new class, and submit the changes to the database.

R7. The system shall allow a user to add a current class.

7.1. The user shall click the “Add A Class” button.



Figure Adding a Class

7.2. The system shall unhide the following text boxes: class name (where the user puts the class’s 6-character name, such as “ENG101”), initial assignment description (for the first graded assignment/test, example: “Exam 1”), grade received (grade received on the initial assignment/test, e.g., 88.5%), weight (weight of the initial assignment/test, e.g., 10%), and semester (such as “fall 2015”).



Figure Info for a new Current Class

7.3. The user shall enter all the above information and click “Submit”.

7.4. The system shall send the data to the database and add the class to the drop-down menu.

R8. The system shall track a user’s degree progress.

8.1. The user shall click the “View Degree Progress” button (see Figure 2) from the initial screen.

8.2. The system shall display the user’s progress, based on the information already submitted about the user’s completed classes (hours, class type, etc.).

8.3. The user shall click the degree concentration he/she is seeking, and the system shall adjust the student’s progress, based on the newly selected computer science concentration.



Figure Student Progress Overview

R9. The system shall allow a user to calculate future GPA, given certain final grades in classes.

9.1. The user shall go to the GPA calculator by clicking the “GPA Calculator” button on the initial screen.

9.2. The system shall display the user’s current GPA, based on final grades that have been submitted previously for completed classes.

9.3. The user shall enter the final grade and credit hours for some classes.

9.4. The user shall click on the “Calculate” button.

9.5. The system shall display what GPA the user will have based on the submitted final grades and credit hours and the information just submitted.



Figure What-If GPA Results

R10. The system shall allow a user to delete a completed class.

10.1. From the initial screen, the user clicks on the “View Completed Classes” button.

10.2. The system displays a screen that shows all the user’s completed classes in a drop-down box.

10.3. The user selects the class he/she wishes to delete.

10.4. The user clicks the “Delete Class” button (see Figure 9).

10.5. The system asks the user to confirm the delete.

10.6. The user clicks the “Yes” to confirm.

10.7. The system deletes the class from the database and the drop-down box and displays a success message.

R11. The system shall allow a user to modify a completed class (e.g., change the class’s category, credit hours, final grade, etc.).

11.1. The user shall go to the final grades overview for completed classes.

11.1.1. From the initial screen, user clicks on “View Completed Classes”.

11.2. The user makes desired changes.

11.3. The user clicks the “Submit Modifications” button (see Figure 9).

11.4. The system shall ask the user to confirm the changes.

11.5. The user shall click the “Yes” button to confirm changes.

11.6. The system shall submit the changes to the database and display a success message.

R12. The system shall allow a user to view current classes.

12.1. The user shall select the semester for the class that he/she wishes to view classes.



Figure Viewing Current Courses

12.2. The system shall fetch all the classes for the chosen semester and place them in a dropdown box.

12.3. The user shall select the course from the dropdown that he/she wishes to view grades for.

12.4. The system shall fetch all available grades for the selected course and put them in the jTable. The system shall also calculate the user’s current grade in the course.

## Nonfunctional Requirements

NR1. All buttons must have a purpose

* 1. When a button is pressed it will have an outcome of some sort, whether it be visible or not.

NR2. System shall not crash under invalid data input.

2.1. When the user enters data into the system that is wrong data type or just invalid the system will error check and inform the user.

NR3. Any information in the database should be secure.

3.1. Only those who have access to that information should receive that information.

NR4. The functions of this system should be consistent and reliable.

4.1. The system should always output the same content given the same input.

# Use Case Diagram

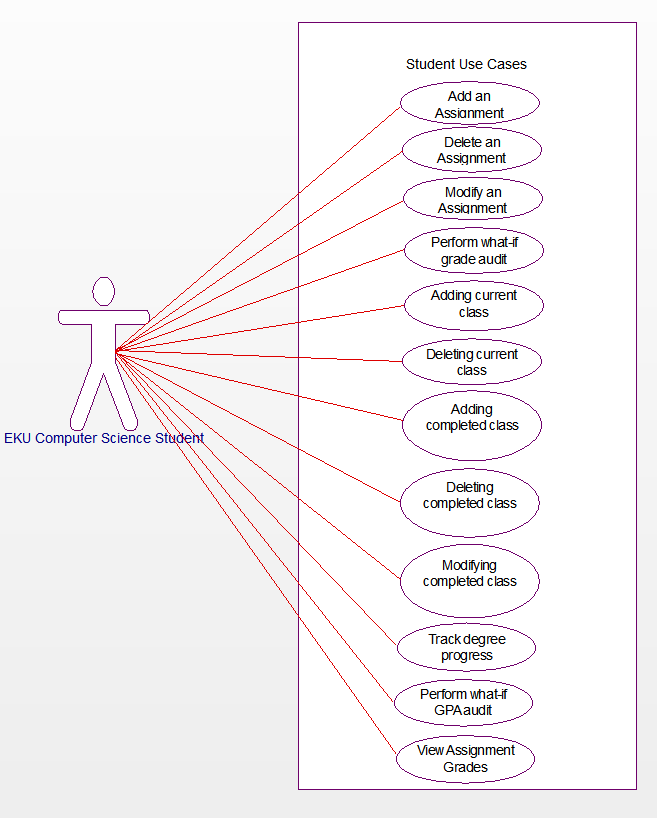


Figure Use Case Diagram

Adding a grade – Use Case to add a grade for a class to the system.

Deleting an assignment grade – Use Case to delete an assignment/test/quiz grade from the system.

Modifying an assignment grade – Use Case to modify a grade in the system.

Perform what-if grade audit – Use Case for a student to see what his/her grade will be given certain grades on future assignments.

Delete a current class – Use Case to delete a current class.

Adding a current class – Use Case to add a current class. Once a current class is added, user can submit grades for that class.

Adding a completed class – Use Case to add information about a completed class (such as class name, final grade, etc.).

Deleting a completed class – Use Case to delete a completed class.

Modifying a completed class – Use Case to modify information about a completed class (e.g., class name).

Track degree progress – Use Case to see how close you are to graduating.

Perform what-if GPA audit – Use Case to see what your GPA will be given certain final grades.

View Current Classes – Use Case for the student to see what their current classes are.

# Class Diagram

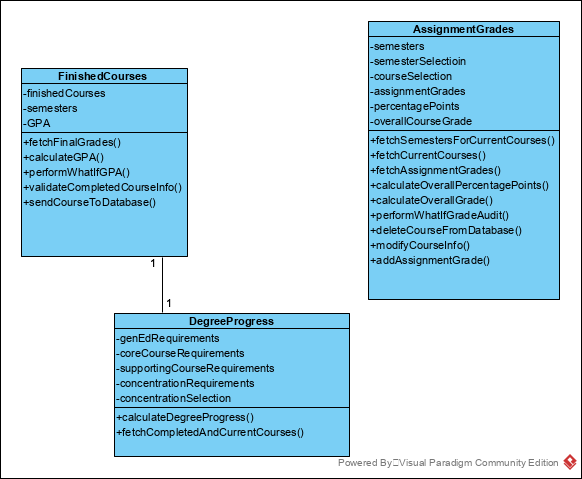


Figure Class Diagram of Grade, GPA, and Degree Tracking System

The above figure shows the classes and their relationships. Notice that the AssignmentGrades class is unrelated to the other two classes. The is because the grades for assignments/tests/quizzes is a different part of the overall system than tracking a student’s degree progress and checking GPA. However, the degree progress requires knowledge of completed courses, so there is association between that class and the finished courses.

# Sequence Diagrams

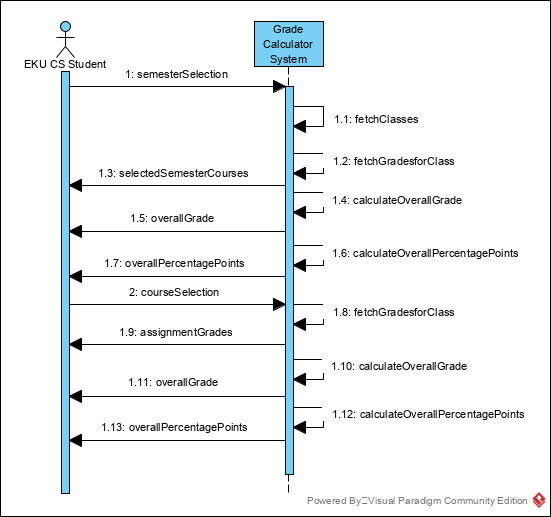


Figure Sequence Diagram for Viewing Assignment Grades

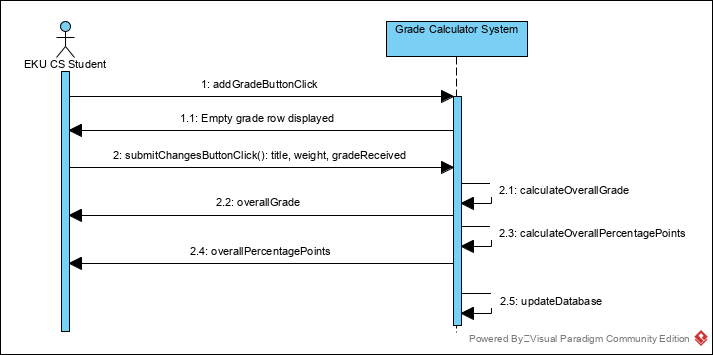


Figure Sequence Diagram for Adding an Assignment Grade

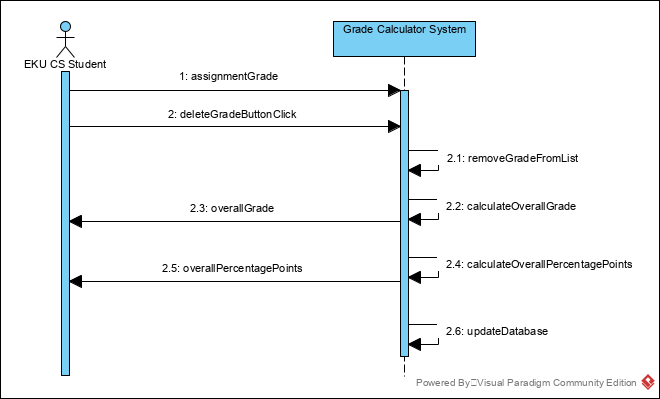


Figure Sequence Diagram for Deleting an Assignment Grade

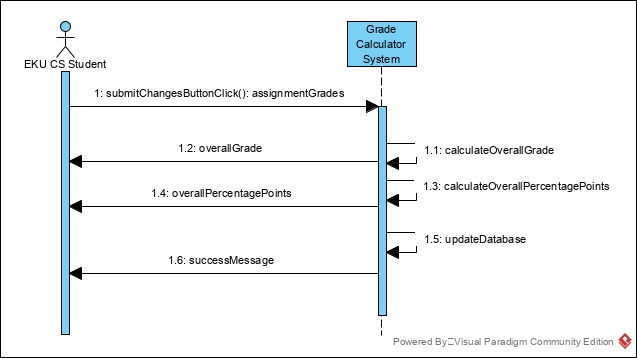


Figure Sequence Diagram for Modifying an Assignment Grade

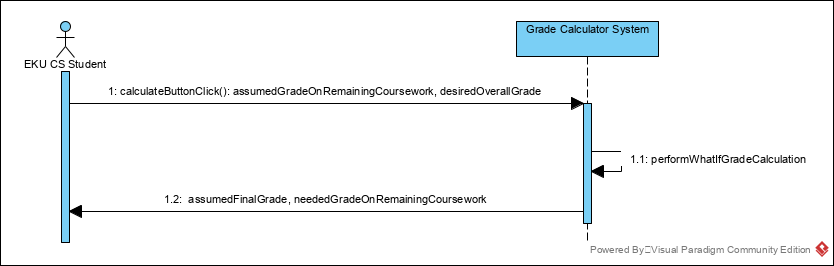


Figure Sequence Diagram for Performing What-If Grade Calculation

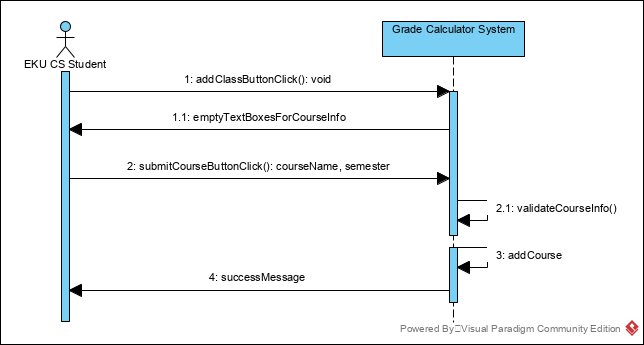


Figure Sequence Diagram for Adding a Current Course

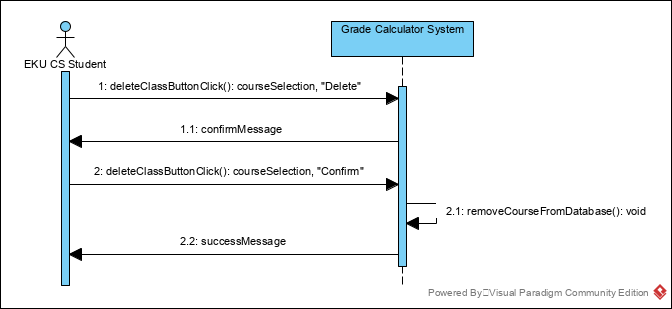


Figure Sequence Diagram for Deleting a Current Class

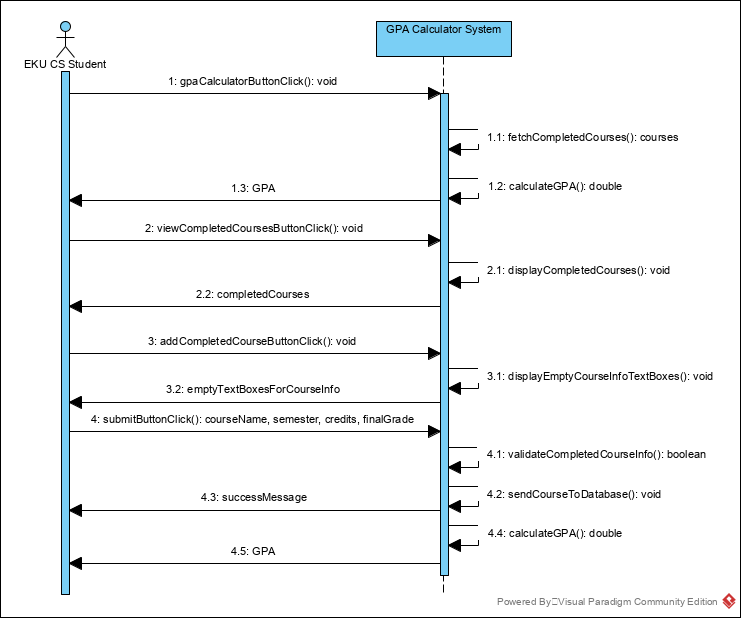


Figure Sequence Diagram for Adding a Completed Course

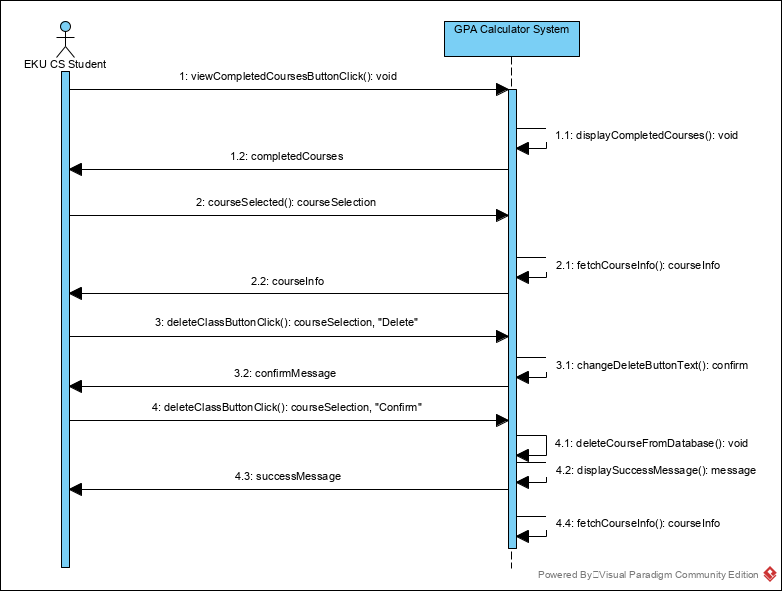


Figure Sequence Diagram for Deleting a Completed Course

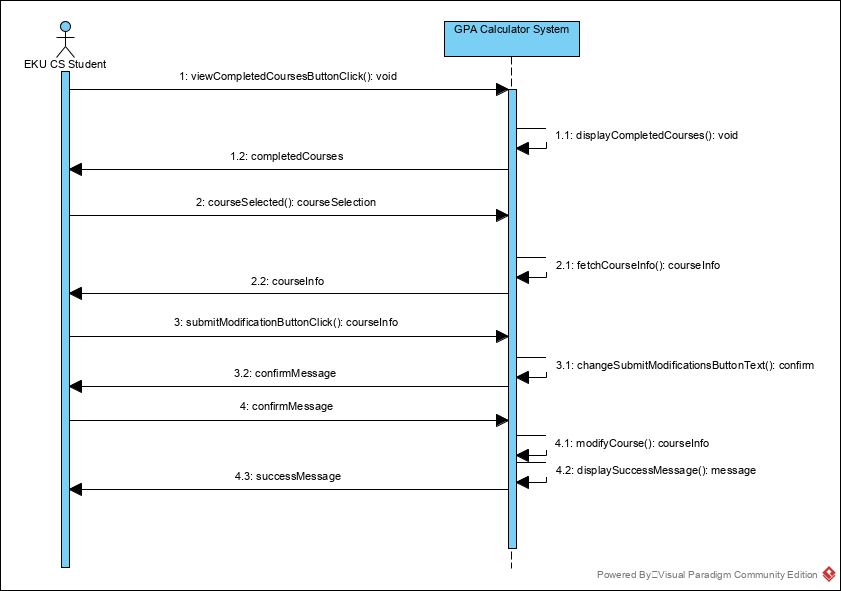


Figure Sequence Diagram for Modifying a Completed Course

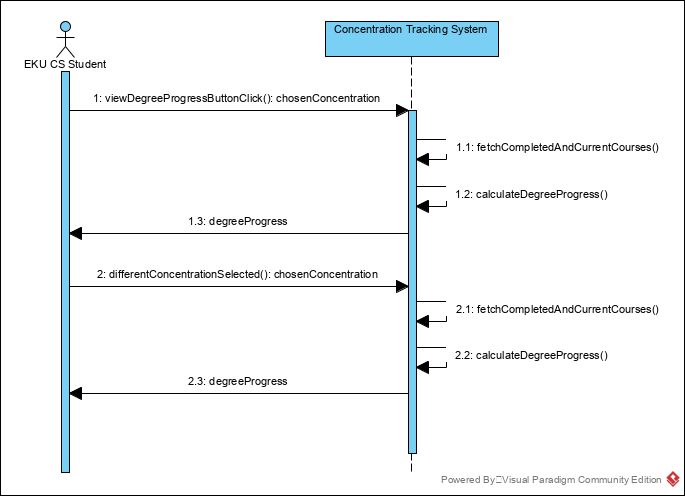


Figure Sequence Diagram for Tracking Degree Progress

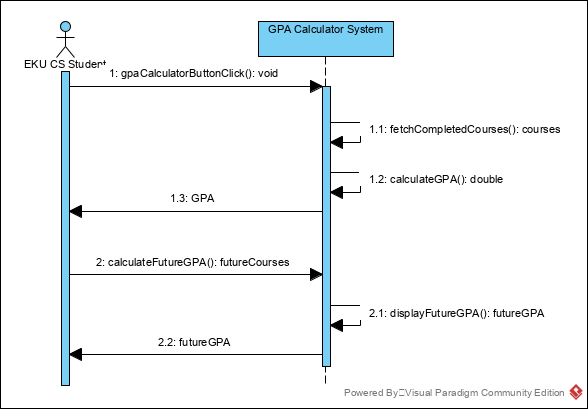


Figure Sequence Diagram for Performing What-If GPA Audit

# Activity Diagrams

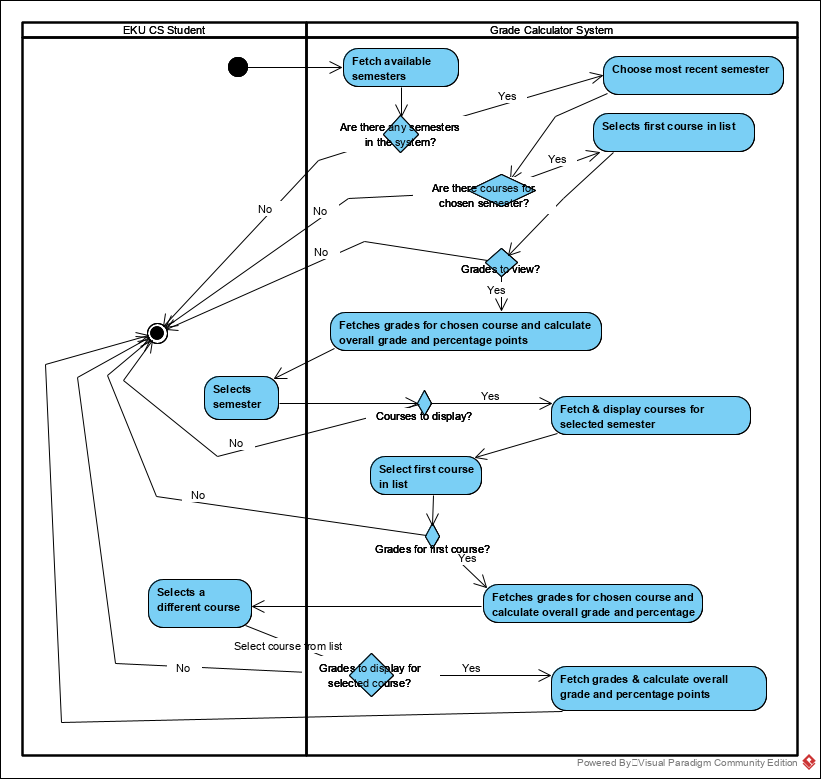


Figure Activity Diagram for Viewing Assignment Grades

# State Diagrams

# Database Diagram



Figure ER Diagram

# Conclusion

For the EKU C.S. student, earning grades is a difficult yet important task. It is stressful to keep track of grades in several different classes, and many times students find themselves worrying about what final exam grades they will need to pass a course or receive 89% in a class. At other times, students might wonder what will happen to their GPA given certain final grades. And at other times, students may wonder if they are on track to graduate. This system makes the EKU computer science student’s job of tracking grades and degree progress much easier. It allows students to quickly check their course grade, GPA, and degree progress, as well as to see what impact future grades will have on their overall course grade and what impact final grades in future courses will have on their overall GPA.

In this report, I have outlined the capabilities of the system, constraints, and overall structure. Suggestions on how to improve the system are welcome!

# Data Dictionary

**Assignment grades** – grades that are not final grades. An example of an assignment grade would be a midterm grade. It has a name (“midterm”), the student received a grade on it (e.g. 92%), and it has a weight (e.g. 20% of the overall course grade). Additionally, this grade belongs to a specific class (e.g. CSC440) which is taken during a specific semester (“fall 2019”, for example).

**assignmentGrade** – one single assignment grade. Example: title: “Test 1”, weight: 15%, gradeReceived: 94.5%.

**assignmentGrades** ­– a data structure that holds all the assignment/test/quiz grade titles, their weights, and grade received for a specific course.

**Assumed grade on remaining assignments** – this is a grade a user assumes she will make on the rest of the assignments/tests in a course. For example, suppose that 25% of the coursework has not yet been graded, and the user would like to see what overall grade she will have if she makes 87% on this remaining 25% of coursework. The grade calculator will show the user what her final grade in the course would be, given this information.

**assumedFinalGrade (double)** – this is the grade a user will have overall in the course, if she makes the grade stored in assumedGradeOnRemainingCourseWork.

**assumedGradeOnRemainingCoursework (double)** – this is one of the inputs a user will use when he/she is performing a what-if grade audit. A user would like to see what overall grade he/she will have in a course if this grade is made overall on remaining coursework.

**chosenConcentration (string)** – the concentration that an EKU C.S. student has chosen. For example: “Statistical Computing”.

**Completed assignments/tests** – assignments/tests that count towards a student’s final grade in a class. Every test/assignment/quiz score must consist of the name of the task, its weight, and the grade the student received on it.

**Completed/finished classes/courses** – classes that a student has finished and has received a final grade for.

**completedCourses** – a data structure holding information about all the completed courses entered into the system by the student. Each item in this data structure is of type **courseInfo.**

**Concentration** – the computer science specific degree track that a student is enrolled in. As of September 2019, there are five computer science concentrations offered by Eastern Kentucky University. They are general C.S., computer technology, interactive multimedia, artificial intelligence in data science, and digital forensics and cybersecurity.

**concentrationRequirements** – a data structure containing the courses that are needed to satisfy computer science concentration-specific course requirements.

**coreCourseRequirements** – a data structure containing the courses that are needed to satisfy core course requirements.

**courseInfo** – a data structure containing information about one completed course, such as course title, final grade, credits, and semester taken.

**courseSelection (string)** – the title of a course that has been selected from a list of courses. E.g. “ENG101”.

**Current class/Current course** – a class that a student is currently enrolled in. These are classes that a student would ordinarily enter information about current grades (grades for assignments, quizzes, homework, etc.).

**degreeProgress** – a data structure containing information about a student’s progress towards completing their chosen C.S. concentration degree requirements.

**desiredOverallGrade (double)** – used when performing a what-if grade audit. A user would like to see what grade he would need to make on, say, the final exam in order to get this overall grade in the course.

**EKU C.S. Student/EKU CS Student** – a computer science student at Eastern Kentucky University.

**Final grade** – the grade awarded to a student at the end of a class. This system is setup to handle “A” (for excellent, worth 4 quality points), “B” (for good, worth 3 quality points), “C” (for average, worth 2 quality points), “D” (for poor, worth 1 quality point), “F” (for failed, worth 0 quality points), “S” (for satisfactory, does not count toward student’s GPA), and “W” (for withdrawn, does not count toward student’s GPA) as final grades.

**Future class** – a class that a student has not yet taken but plans on enrolling in. This term is usually used in the context of the GPA calculator, when a student would like to know what GPA he/she would have in the future, if certain final grades are made in future, unfinished courses.

**futureCourses –** a data structure holding credits and final grades for future courses. It is used to calculate what a student’s GPA will be in the future, given certain final grades on the courses.

**futureGPA (double) –** the GPA a student will have in the future, assuming they make the final grades specified by futureCourses and given the final grades earned in completedCourses.

**genEdRequirements** – a data structure containing the courses that are needed to satisfy general education requirements.

**GPA** – Grade Point Average. GPA is a student’s weighted average of final grades. Calculated by dividing a student’s total number of grade points received by the number of credit hours attempted.

**gradeReceived (double) –** the percentage grade a student received on an assignment/test.

**neededGradeOnRemainingCoursework (double)** – this is the grade that a user will need to score on remaining coursework in order to finish with an overall average defined by desiredOverallGrade.

**overallGrade (double)** – a student’s overall grade in a course, calculated by dividing percentage points by the sum of weights on completed assignments/tests.

**overallPercentagePoints (double)** – the overall percentage points a student has earned in a (current) course. For example, if a student takes a midterm test and scores 97% and that test’s weight is 10%, then that student has earned 9.7 percentage points (0.97 \* 10 = 9.7).

**Percentage point** – a percentage point is that part of an assignment/quiz/test grade that has been earned by a student. For example: if a student earns 85% on Homework 1, which is weighted at 10%, then the student is awarded 8.5 percentage points.

**selectedSemesterCourses** – data structure containing all the courses for a chosen semester.

**semesterSelection (string)** – the semester that has been selected by a student or the system. Example: “SPRING 2014”.

**supportingCourseRequirements** – a data structure containing the courses that are needed to satisfy supporting course requirements.

**title (string) –** the name of the assignment/quiz/exam/test grade submitted. For example, “Assignment 2”, “Exam 1”, etc.

**weight (double)** – the percentage worth of a grade. For example, the final test in a course might be worth 25% of overall course grade.

**What-if grade audit/calculation** – this calculation is useful if a user has submitted some, but not all, of the coursework for a class. The user may ask the system what grade he would need on remaining coursework in order to receive a certain, desired grade overall. Additionally, a user may ask the system what grade she would receive in the course if he were to get a certain grade on remaining coursework.