Project Name: Homework Tracker // Group 6

1. Overview

Homework Tracker is a software solution that helps students manage homework across the courses that they are taking. It aims to reduce the time and stress that comes from homework management and to simplify the process of tracking homework. HomeworkTracker presents class and assignment information in an easy to use graphical way. Additonally, homework tracker keeps track of when assignments are due and notifies the user, via the Windows 10 notification tray, when an assignment's deadline is within the current week.

2. Source Code

See Canvas for submission file: homework-tracker.zip

3. Test Plan

3.1 Levels of Testing

- Unit Tests:
    - (1) Assignment Constructor
        - Tests the Assignment class's constructor method to see if it properly creates Assignments, based on the passed arguments.
    - (2) GradeWeightCategory Total Points Method
        - Tests if a GradeWeightCategory's TotalPoints property returns the actual total amount of points the GradeWeightCategory has.
    - (3) Notification Constructor
        - Tests the Notification class's constructor method to see if it properly creates Notifications, based on the passed arguments.
- Integration Tests:
    - (4) Integrate the classes: Course, NotificationGenerator, and NotificationQueue
        - Tests if the notification generator creates and adds notifications to the queue by getting the due dates of assignments in a list of courses.
    - (5) Integrate the classes: GradeWeightCategory, and Assignment
        - Tests if the Assignments in a GradeWeightCategory have accurate grade percentages when the weight changes and multiple Assignments are added.

3.2 Test Report

Note: All tests were performed using Visual Studio's built in testing framework.

- Assignment Constructor
    - Sequence number:
        - 1
    - Condition to be Tested:
        - The value of the Name, Points, and DueDate properties of an Assignment after construction.
    - Test Case:
        - Test Data:
            - DateTime testDate = DateTime.Now;
            - Assignment assignment = new Assignment("Test", 100, testDate);
        - Expected Result:
            - assignment.Name == "Test" && assignment.Points == 100 && assignment.DueDate == testDate is true
        - Successful:
            - Yes

- GradeWeightCategory Total Points Property
    - Sequence number:
        - 2
    - Condition to be Tested:
        - The value of the TotalPoints property in the GradeWeightCategory class
    - Test Case 1:
        - Test Data:
            - A test GradeWeightCategory with no Assignments.
        - Expected Result:
            - testCategory.TotalPoints == 0 is true
        - Successful:
            - Yes
    - Test Case 2:
        - Test Data:
            - A test GradeWeightCategory with three Assignments. The total points across all the assignments is 60.
        - Expected Result:
            - testCourse.TotalPoints == 60 is true
        - Successful:
            - Yes

- Notification Constructor
    - Sequence number:
        - 3
    - Condition to be Tested:
        - The value of the Title and Message properties of a Notification after construction.
    - Test Case 1:
        - Test Data:
            - Notification notification = new Notification()
        - Expected Result:
            - notification.Title == “No Title” && notification.Message == “No Content” is true
        - Successful:
            - Yes
    - Test Case 2:
        - Test Data:
            - Notification notification = new Notification(“Test Title”, “Test Message”)
        - Expected Result:
            - notification.Title == “Test Title” && notification.Message == “Test Message” is true
        - Successful:
            - Yes

- Integrate the classes: Course, NotificationGenerator, and NotificationQueue
    - Sequence number:
        - 4
    - Condition to be Tested:
        - The test condition is the value of the NumNotifications property in our test NotificationQueue. NumNotifications is the number of notifications added to the queue by the NotificationGenerator when it is given a list of courses.
    - Test Case:
        - Test Data:
            - A list of 2 Courses, each with GradeWeightCategories that have Assignments in them. Two of the Assignments have a due date that is within a week. Two Assignments have a due date that is a month or more away. One Assignment has a due date that has already passed.

        - Expected Result:
            - Since two assignments are due within the current week, we would expect two notifications to be added to the queue. Therefore, notificationQueue.NumNotifications == 2 should be true.
        - Successful:
            - Yes

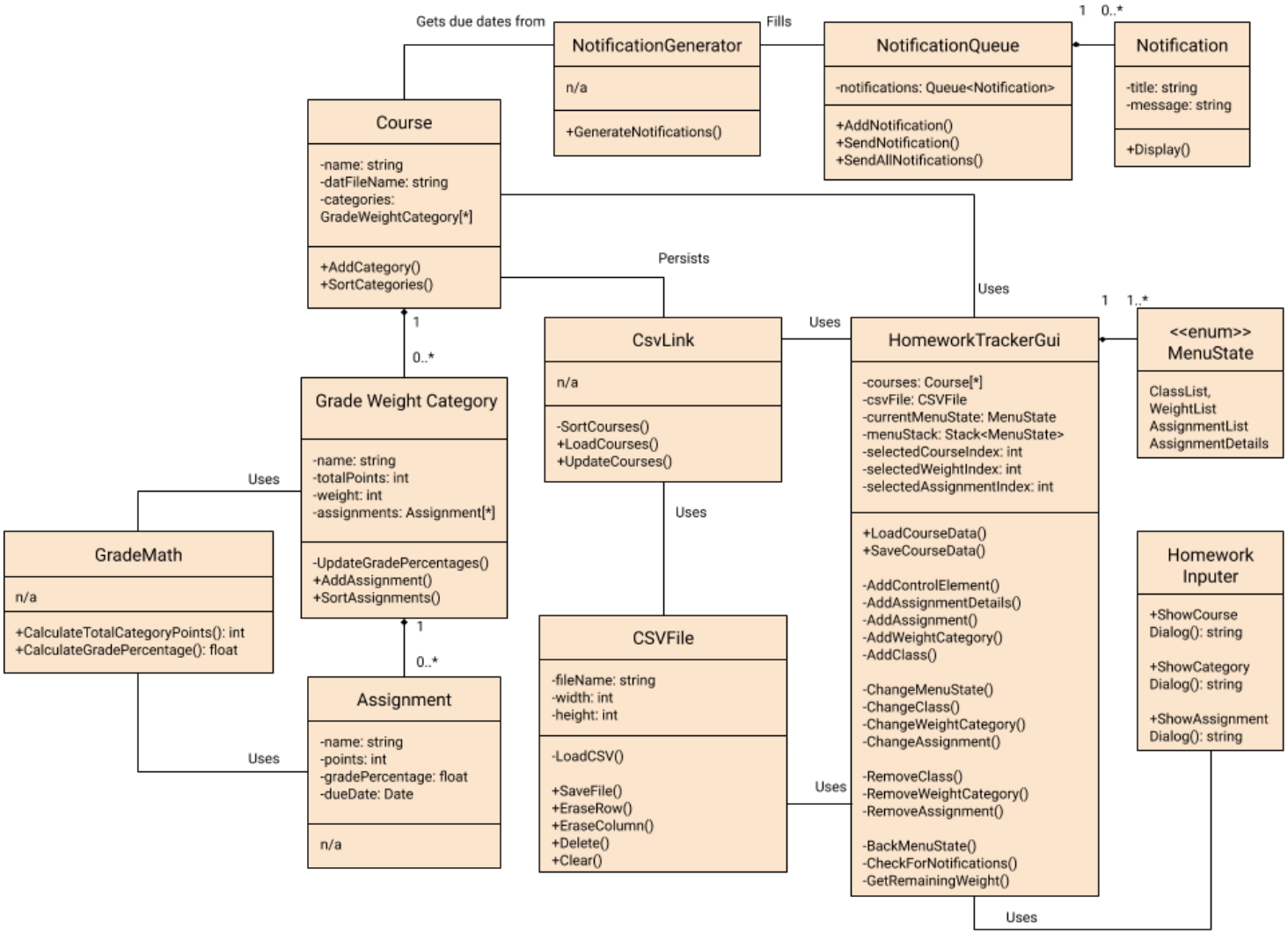
- Integrate the classes: GradeWeightCategory, and Assignment
    - Sequence number:
        - 5
    - Condition to be Tested:
        - The test condition is the value of the GradePercentage property in our test Assignment. GradePecatage is the percentage of the total grade an Assignment is worth, and it is dependent on the GradeWeightCategory the Assignment is in.
    - Test Case 1:
        - Test Data:
            - A GradeWeightCategory with 100 points, a weight changed from 70 to 50, and our test assignment as a member. The test assignment has zero points.
        - Expected Result:
            - testAssignment.GetGradePercentage() == 0 is true
        - Successful:
            - Yes
    - Test Case 2:
        - Test Data:
            - A GradeWeightCategory with 325 points, a weight changed from 70 to 50, and our test assignment as a member. The test assignment has 25 points.
        - Expected Result:
            - testAssignment.GetGradePercentage() == (25 / 325) * 50 is true
        - Successful:
            - Yes
    - Test Case 3:
        - Test Data:
            - A GradeWeightCategory with 25points, a weight changed from 70 to 50, and our test Assignment as a member. The test Assignment has 25 points.
        - Expected Result:
            - testAssignment.GetGradePercentage() == 50 is true
        - Successful:
            - Yes

4. Updates/Modifications

- Description of Changes:

  Due to how GUI design works in Windows Forms, we had to merge HomeworkDisplayer and GuiManager into one HomeworkTrackerGui class. Additionally, we still had some class members that pertained to functionality that we are not supporting in the prototype, so we removed those members. We also added a CSVFile class that we forgot to include in our SDD, but ended up needing. Finally, we added and modified some members and functions in our backend classes to better support the functionality we are providing in our prototype. For example, NotificationGenerator is now associated with Course instead of Assignment, because the NotificationGenerator needs to get information from all of a Course's Assignments to properly create all of the necessary Notifications. Another example is that CSVLink now only needs to persist Course to support the prototype's functionality. Our changes only impact our class diagram, all other diagrams are unaffected.

- Updated Class Diagram (constructors, properties, Windows Forms constructs, and other C# constructs are omitted for clarity):



5. User’s Guide

See Canvas for submission file: User Guide.pdf

6. Glossary

- Acronym definitions:
    - GUI: Graphical User Interface
    - CSV: Comma Separated Values
    - SDD: Software Design Document