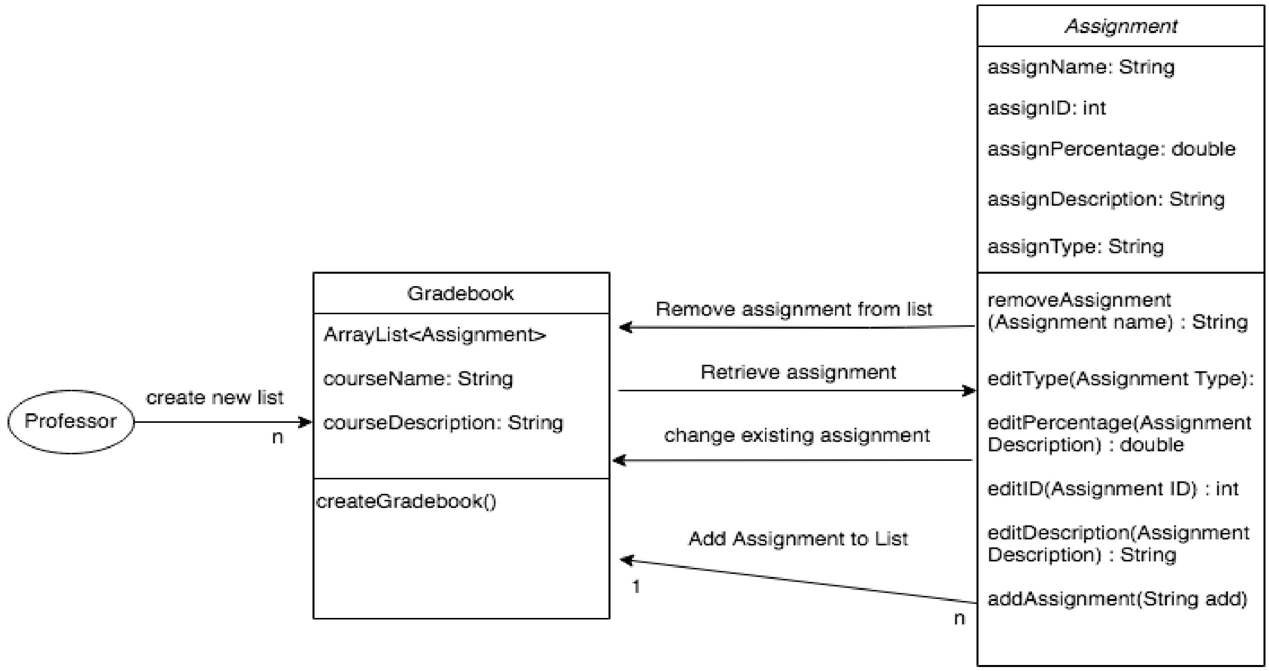
Class Gradebook Software Design Specification

Created By: Brandon Reiley, Bhavya Desai, and Erik Torres

Software Architecture Overview

Introduction: The purpose of this software design specification is to display and discuss the development of the class gradebook software system. The document will begin by detailing the class structure and architecture that is planned to be implemented into the system. The specification will then discuss the development plan and timeline for the software system and the planned responsibilities for the team.


Assignment
assignName: String
assignID: int
assignPercentage: double
assignDescription: String
assignType: String
removeAssignment (Assignment name) : String
editType(Assignment Type):
editPercentage(Assignment Description) : double
editID(Assignment ID) : int
editDescription(Assignment Description) : String
addAssignment(String add)
Gradebook
ArrayList<Assignment>
courseName: String
courseDescription: String
createGradebook()
Professor
create new list
n
Remove assignment from list
Retrieve assignment
change existing assignment
Add Assignment to List
1
n

The diagram above is a UML (Unified modeling language) class diagram. Each class has a name attached to them and contains attributes which are specified by instance variables. Furthermore, if needed, a class will contain operations or functions. The overall purpose of the class diagram is to display the structure of a Gradebook software system. It consists of 2 main classes: the Gradebook class and the Assignment class.

Class Descriptions:

- Gradebook class: This class is designed to hold and allow access to the assignment objects.

- Assignment: This class is used to create assignments that are essentially stored in a gradebook object.

Attributes of each class:

- Gradebook: The attribute ArrayList<Assignment> holds the assignments created in an ArrayList. Since each gradebook object is a gradebook for a course, the courseName attribute is used to distinguish between them. The courseDescription attribute is used to describe the course as needed. The courseName and courseDescription attributes are of datatype String.

- Assignment: The assignName attribute holds the name of an assignment and is of data type String. The assignID attribute holds the attached Identification number of an assignment and is of datatype int. The assignPercentage attribute is used to hold the percentage an assignment has for the total grade and is of datatype double. The assignDescription attribute is used to hold a description of an assignment; it uses a String datatype. The assignType attribute is of datatype String and holds information on the assignment type.

Operations of each class:

- Gradebook: The createGradebook() operation is used to create an assignment list for a course. It requires the user to input a name for the gradebook and a description.

- Assignment: The removeAssignment() operation removes an assignment from a gradebook object with the only parameter being the assignment name. The addAssignment() operation allows for the creation of an assignment. The editType() operation allows for the type of an assignment to be modified and accepts a String datatype for the parameter. The editPercentage() operation uses a double datatype for the parameter and modifies the percentage of an assignment. The editID operation uses a parameter of type int to modify an identification number for an assignment. The editDescription() operation allows the user to modify an assignment's description and uses a string for the parameter.

Development Plan and Timeline

The development for the software will begin with creation of the classes and related attributes. Once their development is complete, the associated operations will be implemented and tested to ensure they are stable and work as intended. First, the createGradebook operation will be created, followed by the addAssignment operation. Operations related to removing and modifying assignments will be implemented and tested. Lastly, after the overall system has been tested and the desired features are incorporated, it will be tested in a console-based environment.

All members of the group will engage in the programming process at different stages. Initial plans include Bhavya Desai and Erik Torres working on programming the classes and attributes and Brandon Reiley testing initial builds for functionality. Members may switch roles if needed.