Assessments - Computations -

Design

Assessments: Computations

2/23/2022

2.1

By: Mitchell Boone, Michael Quinn, and Liam Costello

Table of Contents

Contents

[1. Document Description 3](#_Toc96508754)

[1.1. Introduction 3](#_Toc96508755)

[1.2. System Overview 3](#_Toc96508756)

[2. Design Considerations 4](#_Toc96508757)

[2.1. Assumptions and Dependencies 4](#_Toc96508758)

[2.2. General Constraints 4](#_Toc96508759)

[2.3. Goals and Guidelines 4](#_Toc96508760)

[2.4. Development Methods 5](#_Toc96508761)

[3. Architecture 5](#_Toc96508762)

[3.1. Architecture Strategies 5](#_Toc96508763)

[3.2. System Architecture 6](#_Toc96508764)

[4. Policies and Tactics 7](#_Toc96508765)

[4.1. Product Choices 7](#_Toc96508766)

[4.2. Testing 7](#_Toc96508767)

[4.3. Maintaining the Software 7](#_Toc96508768)

[5. Detailed System Design 8](#_Toc96508769)

# Document Description

## Introduction

This document will provide information regarding the design of the computations part of the Assessments software. It will provide specifically how this part of the software will work as well as different considerations and strategies used in the design process.

## System Overview

The computations part of the Assessments software will deliver outcome and assessment data as a CSV file for one or more courses in the Canvas Learning Manager System as well as provide the visualizations team with the data required to create their visual comparisons. There are two main parts of the computations part of the software:

* Calculate the average, median, percent of students above the average threshold of 3, and the standard deviation
* Create a CSV file for the comparison of all courses selected a well as a separate CSV file for each course

These tasks will provide two outcomes:

* A vector of course objects with the comparison data to be sent to the visualizations team
* Various CSV files to be stored separately for an alternative view of the data

# Design Considerations

## Assumptions and Dependencies

We are assuming in the design that the software will only be used on PC running the Windows 10+ operating system.

## General Constraints

The constraint with the design of the computations part of the software is time. If time permits we will be going ahead and adding the standard deviation, if not that computation will not be included.

## Goals and Guidelines

We are going to try and make it so we can send the data through a vector of course objects containing all the comparison data rather than requiring the visualizations team to read in the data from the CSV files, which would same time when running the software. We will make sure we get the CSV files generated first though in case the time constraint does not allow us to complete this goal.

## Development Methods

We will be using a divide and conquer technique when designing our part of the software, breaking the coding into four separate parts:

* Get the outcomes data from the Canvas API
* Take the data from JSON
* Compute the comparisons based on the outcomes
* Save the data as CSV files

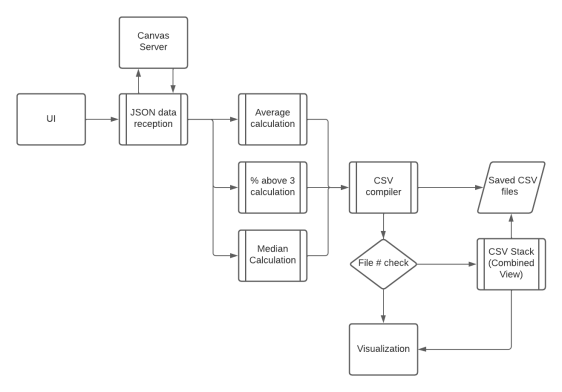
# Architecture

## Architecture Strategies

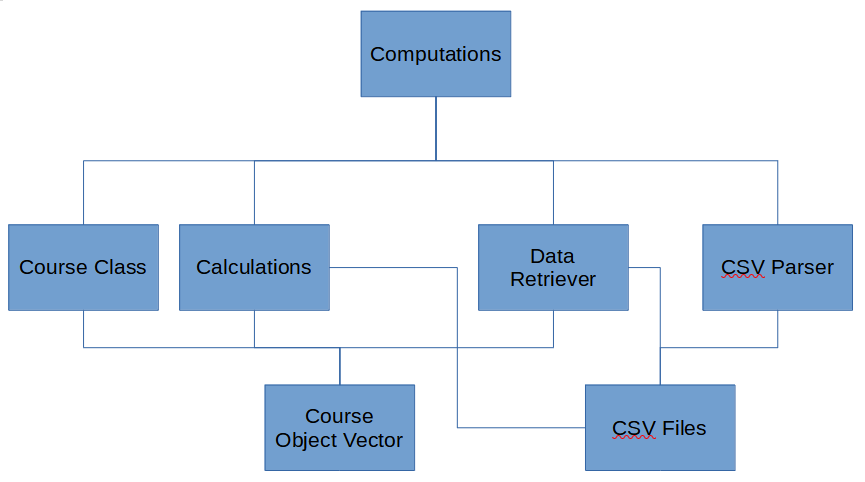
We will be utilizing Visual Studio 2019, CURL, JSON, BOOST, and C++ to develop the software. If we do not have time to add the course class or standard deviation features, we will look to leave the code in a good spot to allow for the addition of those features and leave helpful comments for feature developers of the software. Bitbucket will be used as our version control for this project. With Bitbucket each team will be using a separate .cpp file to allow us all to work on our parts without running into issues with the other teams’ code.

## System Architecture

Data Flow Diagram:



Hierarchy Chart:



# Policies and Tactics

## Product Choices

Visual Studio 2019 will be used for writing the code because it is installed on all the school computers and is free to download for at home use. We are using JSON, CURL, and BOOST because they were used in the previous version and should work fine with the scope of the project.

## Testing

The testing of the computations part of the software will be through the use of data that is in the same format as the data we will be receiving. A sample course with an instructor and two students will be used for running tests regarding how the program will work with the Canvas API.

## Maintaining the Software

We will look to keep the software in a good spot by writing good comments throughout our development and using Bitbucket version control. Doing this will allow us to develop the software without running into the issue of being stuck with a piece of software that does not work. It will also allow future developers of the program to be able to expand the software without struggling to find out how the code works.

# Detailed System Design

The software will work by first receiving the Canvas Course IDs and a Canvas API key from the UI. The outcomes data will be received from the Canvas API for the Canvas Course IDs and Canvas API key given using JSON and CURL. The data retrieved will be used to calculate the average, median, percent of students above the average threshold of 3, and standard deviation for both the individual courses as well as the combination of the courses. Once the calculations have been made, they will be used along with the data received from the Canvas API to generate a CSV file along with a vector of course objects. The course class will store the individual information for each of the courses. The CSV files will be saved in a folder and both the vector of course objects and the CSV files will be sent to the visualizations team to generate the visual diagrams.