EECS 348 Group 10

Web Calculator for University of Kansas Students
Software Architecture Document

Version 2.0

Revision History

Date	Version	Description	Author
11/12/2023	1.0	Initial version	Harlan Williams
12/03/2023	2.0	Update changes to project structure	Harlan Williams

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a representation of the major architectural design of the system as well as specific details of the architecture.

1.2 Scope

This document covers the design of the Web-based calculator app at a system level and class level.

1.3 Definitions, Acronyms, and Abbreviations

This document uses the same definitions, acronyms, and abbreviations described in the Project Plan.

1.4 References

This document references the Project Plan for definitions, acronyms, and abbreviations, the Software Requirements Specification for the functionalities and constraints of the software to be developed, and the UML class diagrams 'Calculator_Class_Diagram.pdf' and 'Website_Class_Diagram.pdf' in the project documentation folder.

1.5 Overview

This document details how the architecture of the system is represented, the system requirements and constraints that informed the architectural decisions, the specific details of the system architecture, including its major subsystems and packages, and a description of how the architecture contributes to the constraints and non-functional requirements of the system.

2. Architectural Representation

The architecture is represented using UML class diagrams to represent the modules of the architecture and the connections between them.

3. Architectural Goals and Constraints

The web-app is a single class that uses the cpp-httplib library from https://github.com/yhirose/cpp-httplib to serve an index and about page, and route POST requests from users inputting values into the calculator to the calculator module.

The calculator module will consist of three classes to perform the data manipulation necessary to achieve the main functionality. These three classes will be an interface class, a tokenizer class, and an evaluator class. This architecture creates separation between the functionalities of the calculator processes, which allows for modifications to individual processes without affecting other processes.

4. Logical View

4.1 Overview

The project consists of an App that handles requests from a web page with the calculator interface, and a calculator module that evaluates strings as arithmetic expressions and returns their evaluated value.

4.2 Architecturally Significant Design Modules or Packages

The significant design objects are represented pictorially in the class diagrams located in the project documentation folder.

5. Interface Description

There will be a web interface consisting of endpoints where users can access the calculator to perform operations.

In the calculator endpoint, the user will be able to enter an expression in a text box or form, and submit their input. The app then sends the input to the calculator module, which both performs error checking and evaluates the expression.

6. Quality

Distributing the calculator module's functionality between a tokenizer and evaluator logically separate the tasks required to evaluate the expression and allow the calculator to perform error checking of specific errors in each logical step.