SWENG Assignment 2 - Calculator Web App

Jamie Taylor - 20333503
Abdelaziz Abushark - 20332134
Cormac Feely - 20332223
Jacob Wilson Sharpe - 20332475

Brief Description

The Flask web app we created takes a mathematical expression from the user in the form of a String input through the text box and validates and evaluates the expression, displaying the result to the user. The evaluation supports all the required expressions in the assignment declaration, those being:

1. Handling integers and floating-point numbers.
2. Capable of evaluating addition, subtraction, multiplication, division, and power (^)
3. Capable of evaluating expressions with brackets
4. Capable of evaluating natural log and exp.
5. Results are rounded off to 3 decimal places.
6. It can handle errors by returning error messages.

The App is tested on commits following the CI workflow on GitHub. New releases are pushed to Docker Hub following the CD workflow set up on GitHub.

Features of Each Release

Version 1.5: Complete Command Line Calculator

This version focused on the calculator logic and functionality.

The version implemented a calculator which took its user input through the command line before the web app was developed. It implemented the required calculator functionality, including the features as follows: addition, subtraction, multiplication, division, logarithmic and exponential functions on integers and floating-point numbers. This version also updated the CI to accommodate the updated calculator.

Version 2.1: Web App Release

This version focused on implementing the web app.

In this release, a single-page HTML front-end was developed using flask. This release connects the front-end and the previously implemented calculator logic of the previous release but modified its input to take in a String from the web app’s user interface.

Contributions

Backend Team

Abdelaziz: The task was divided into 2 parts, backend, and frontend in which we decided to split the work into 4 different parts. I worked on the backend part of the calculator in which I implemented reverse polish notation. I handled the first 3 features for the calculator which are:

- It can handle both integers and floating-point numbers.
- It should be capable of performing addition, subtraction, multiplication, division, and power (^).
- It should be able to work with expressions containing brackets.

I wrote half the unit test cases. In addition, I set up the CI workflow on GitHub, so the application is tested whenever new code is committed.

Jamie: I was part of the backend team. I developed features 4 and 5 for the calculator:

- Perform exponential and logarithmic functions
- Round the result to three decimal places

I also wrote half of the unit tests to test the calculator and handled some bugs that arose. I set up the CD workflow to push a release to Docker Hub. I did this in two versions, once for our initial command line version of the calculator, and then I updated it for the web application.

Frontend Team

Cormac: I helped design the frontend of the application by adding information to the calculator page to properly describe its functionality. Furthermore, I also edited the page so that it would display more cleanly, and the result section of the page would always display so that components wouldn't jump around as an expression was inputted.

Jacob: I helped design the frontend of the application along with Cormac. I wrote this report and recorded the video demo for the final deliverable.

Link to the Repository:

https://github.com/azizosharke/calculatorWebAppSWE

Commit Log