Calculator Application Requirements

Feature 1: Unit Conversion Calculator

Rationale

This feature aims to provide users with a quick and easy unit conversion calculator.

Requirements

As a user, I want to be able to do unit conversions on a calculator with the following capabilities:

- Able to select from a list of units on the left input.
- Shortened list of possible units to convert to on the right input.
- Able to modify both input fields and change both values automatically.

Deliverables

Design:

- Direction: Draw diagrams illustrating components and data flow, preferably using UML. Any tool for designing the data flow is acceptable.

Code:

- Direction: Link to the code on GitHub (skeleton is acceptable in the initial development stage).

Test:

- Direction: Link to the test on GitHub, including unit tests to ensure accurate conversions and the ability to modify both sides.

Documents:

- Direction: Link to the document (manual) on GitHub, outlining user instructions and explanations for unit conversions.

Test Plan

- Multiple unit conversion tests to verify accurate results.
- Ensure the left unit reads accurately and adjusts the available right units.
- Unit tests to ensure both sides can be modified, and changes reflect on the other side.

Feature 2: Money Conversion Calculator

Rationale

This feature provides users with a currency conversion calculator for quick and easy money conversions.

Requirements

As a user, I want to be able to do currency conversions on a calculator with the following capabilities:

- Able to select from a list of units on the left and right inputs.
- Able to modify both input fields and change both values automatically.

Deliverables

Design:

- Direction: Draw diagrams illustrating components and data flow, preferably using UML.

Code:

- Direction: Link to the code on GitHub (skeleton is acceptable in the initial development stage).

Test:

- Direction: Link to the test on GitHub, including unit tests to ensure accurate currency conversions and the ability to modify both sides.

Documents:

- Direction: Link to the document (manual) on GitHub, outlining user instructions and explanations for currency conversions.

Test Plan

- Multiple currency conversion tests to verify accurate results.
- Ensure the left and right units read accurately and adjust available units accordingly.
- Unit tests to ensure both sides can be modified, and changes reflect on the other side.

Feature 3: Color Picker

Rationale

This feature allows users to customize the background color of their calculator, adding a personal touch.

Requirements

As a user, I want to be able to customize my calculator for a more personalized experience.

Deliverables

Design:

- Direction: Draw diagrams illustrating components and data flow, outlining the color customization process.

Code:

- Direction: Link to the code on GitHub (skeleton is acceptable in the initial development stage).

Test:

- Direction: Link to the test on GitHub, including tests for successful color customization.

Documents:

- Direction: Link to the document (manual) on GitHub, providing user instructions for customizing calculator colors.

Feature 4: Scientific Functions

Rationale

This feature provides users with scientific operations like trig functions, exponential functions, factorial, and square roots.

Requirements

As a user, I want to be able to perform scientific operations on a calculator with the following capabilities:

- Use of trig functions.
- Use of exponential functions.
- Use of factorial and square roots.

Deliverables

Design:

- Direction: Draw diagrams illustrating components and data flow for scientific functions.

Code:

- Direction: Link to the code on GitHub (skeleton is acceptable in the initial development stage).

Test:

- Direction: Link to the test on GitHub, including unit tests for each scientific function.

Documents:

- Direction: Link to the document (manual) on GitHub, providing user instructions for using scientific functions.

Test Plan

- Individual tests for each scientific function to verify accuracy.
- UI tests to ensure buttons for scientific functions are responsive and provide correct results.

Feature 5: Advanced Functions

Rationale

This feature provides users with advanced functions such as percentage calculations, radians/degrees mode, and constants like pi and e.

Requirements

As a user, I want to be able to perform advanced calculations on a calculator with the following capabilities:

- Percentage calculations.
    - Calculate tips (10%, 15%, and 20%) from a bill.
- Switch between radians and degrees.
- Use constants like pi and e.

Deliverables

Design:

- Direction: Draw diagrams illustrating components and data flow for advanced functions.

Code:

- Direction: Link to the code on GitHub (skeleton is acceptable in the initial development stage).

Test:

- Direction: Link to the test on GitHub, including unit tests for each advanced function.

Documents:

- Direction: Link to the document (manual) on GitHub, providing user instructions for using advanced functions.

Test Plan

- Individual tests for each advanced function to verify accuracy.
- UI tests to ensure buttons for advanced functions are responsive and provide correct results.