# Test Plan for Convert Number to Words Application.

## I. Test valid input:

Verify that the application accurately converts numerical inputs to their corresponding word format in dollars and cents.

### 1. Positive Integers

Description: Test small integers to ensure basic conversions are accurate.

Example test case:

Input: 1

Expected Output: "One Dollar"

### 2. Fractional Amounts

Description: Ensure the application correctly handles decimal values, representing them as parts of a dollar and cents

Example test case:

Input: 0.01

Expected Output: "One Cent"

Input: 1.2 (should be interpreted as 1.20)

Expected Output: "One Dollar and Twenty Cents"

### 3. Large Numbers

Description: Test the application's ability to handle and accurately convert large numbers extending into thousands, millions, and trillions.

Example test case:

Input: 1000

Expected Output: "One Thousand Dollars"

Input: 10000

Expected Output: "Ten Thousand Dollars"

Input: 1000000

Expected Output: "One Million Dollars"

Input: 1000000000000 (One Trillion)

Expected Output: "One Trillion Dollars"

## II. Test invalid input

Enter an invalid input and verify that the application returns accurate error messages when enter invalid inputs.

### 1. Negative integer and decimal number

Description: Ensure the application correctly negative number correctly by return a correct error message.

Example test case:

Input: -1

Expected Output: "Input must not be negative."

Example test case:

Input: -10.12

Expected Output: "Input must not be negative."

### 2. non-numeral input:

Description: Ensure that the application only takes valid number and no other characters.

Example test case:

Input: abc

Expected Output: " The input is invalid, please enter a valid number."

### 3. More than two decimals

Description: Make sure that the system only takes up to two decimal places.

Example test case:

Input: 10.245

Expected Output: " Input has more than two decimal places. Please limit it to two."

## III. Test response time

Description: Make sure the application is response in a timely manner.

Example test case:

Randomly enter inputs and click submit quickly.

Expected Output: The return results such as words and error messages should be no more than a second.

## IV. Test Tools and Approach:

1. Manual Test: Manually enter the input and see if the expected result matched the output

2. Unit Test: Create the test case with XUnit and see if all the test cases are passed.