Approach Overview:

Decimal Handling:

Decision: The approach supports decimal inputs allow up to 2 decimal max.

Rationale: Users might input decimal numbers, which necessitates accurate handling for correct conversions.

BigInt Usage:

Decision: BigInt is used for large integer values (only positive input value)

Rationale: Regular numbers in JavaScript have limited precision, risking inaccuracies for large numbers. BigInt ensures precision and accuracy.

Test Plan

Test Case 1: Positive Whole Number

Input: 12345.00

Expected Output: "twelve thousand three hundred forty five dollars"

Test Case 2: Positive Decimal Number

Input: 12345.67

Expected Output: "twelve thousand three hundred forty five dollars and sixty seven cents"

Test Case 3: Zero

Input: 0.00

Expected Output: "zero dollars"

Test Case 4: Large Number (Decillion)

Input: 123456789012345678901234.56

Expected Output: "one hundred twenty three decillion four hundred fifty six nonillion seven hundred eighty nine octillion twelve septillion three hundred forty five sextillion six hundred seventy eight quintillion nine hundred one quadrillion two hundred thirty four trillion five hundred sixty two billion three hundred forty five million six hundred seventy eight thousand nine hundred one dollars and fifty six cents"

Test Case 5: Number with Only Cents

Input: 0.25

Expected Output: "twenty five cents"

Test Case 6: Number with Only Dollars

Input: 123456.00

Expected Output: "one hundred twenty three thousand four hundred fifty six dollars"

Test Case 7: Number with No Decimal Places

Input: 9876543210.00

Expected Output: "nine billion eight hundred seventy six million five hundred forty three thousand two hundred ten dollars"

Test Case 8: Number with Leading Zeros

Input: 00123.40

Expected Output: "one hundred twenty three dollars and forty cents"