INTERNSHIP TASK -3

DELIVERABLE: A FUNCTIONAL DESIGN WITH SIMULATION SHOWING EACH STAGE'S OPERATION.

To create a **functional design with simulation** showing each stage’s operation, we’ll need to break down the deliverable into structured steps. The design can be for a system, software, or a physical process. Let's break it down into a generic framework and then adjust based on your specific needs.

Steps to Achieve This Deliverable:

1.Define the Problem and Requirements

- What is the problem you are trying to solve?
- What are the input and output requirements?
- Any constraints or specific conditions to consider?

2.Design the System (High-level)

- **Input/Output Specifications**: What is the expected input and output at each stage?
- **System Architecture**: How will the different stages of the system work together?
- **Components**: What modules or components are involved in each stage?

3.Breakdown of Stages (Detailed)

- Identify each stage of operation within the system (e.g., initialization, processing, validation, final output).
- For each stage, describe the following:
  - Inputs to the stage
  - Operations or processes that occur
  - Outputs from the stage
  - Any error handling or exceptions

4.Simulation

- Provide a simulation (either in code or visual representation) that demonstrates the operation of each stage.
- Each stage should be shown in sequence, with sample inputs and outputs for clarity.

Depending on the complexity of the system, the simulation could involve:

- **Software simulation** (e.g., using Python, MATLAB, or another platform to demonstrate logic).
- **Visual simulation** (e.g., flow diagrams or state machines showing how each component interacts).
- **Testing and Validation**
- How do you validate that each stage is working as expected?
- Include test cases or sample data that confirms the design is functional.
- **Documentation**
- Provide clear documentation of the design, including diagrams (UML, flowcharts), descriptions, and the simulation code if applicable.

Example: Simple Automated Sorting System

1. Define the Problem & Requirements

- **Problem**: Create a system that takes in a list of numbers and sorts them.
- **Input**: An unsorted list of integers.
- **Output**: A sorted list of integers.
- **Constraints**: None for now (can be expanded later).

2. High-Level Design

- **System Overview**: The system should:
  - Accept a list of numbers.
  - Process the numbers and sort them in ascending order.
  - Output the sorted list.

3. Breakdown of Stages

- **Stage 1: Input Stage**
    - **Inputs**: List of integers.
    - **Operation**: Receive the input via a form or a function call.
    - **Output**: Pass the list to the next stage (processing).
- **Stage 2: Sorting Stage**
    - **Inputs**: List of unsorted numbers.
    - **Operation**: Sort the list using a sorting algorithm (e.g., bubble sort, quicksort).
    - **Output**: Sorted list of numbers.
- **Stage 3: Output Stage**
    - **Inputs**: Sorted list.
    - **Operation**: Display the sorted list.
    - **Output**: A message showing the sorted numbers.

4. Simulation (Code Example in Python)

```
def input_stage():
    # Example: Receive an unsorted list from the user
    unsorted_list = [34, 12, 5, 23, 89, 7]
    return unsorted_list
def sorting_stage(unsorted_list):
    # Sort the list using bubble sort
    n = len(unsorted_list)
    for i in range(n):
        for j in range(0, n-i-1):
            if unsorted_list[j] > unsorted_list[j+1]:
                unsorted_list[j], unsorted_list[j+1] = unsorted_list[j+1], unsorted_list[j]
    return unsorted_list
```

```
def output_stage(sorted_list):
    # Display the sorted list
    print("Sorted List:", sorted_list)
def main():
    # Run each stage sequentially
    unsorted_list = input_stage()
    sorted_list = sorting_stage(unsorted_list)
    output_stage(sorted_list)
main()
```

5. Testing and Validation

- **Test Case 1**: Input a list like `[34, 12, 5, 23, 89, 7]`. Ensure the output is `[5, 7, 12, 23, 34, 89]`.
- **Test Case 2**: Input an already sorted list and verify the system doesn't alter the order.

Expected Output

When you run the above code, the program will output:

- Sorted List: [5, 7, 12, 23, 34, 89]

Conclusion

- **Stage 1**: The input is taken as an unsorted list.
- **Stage 2**: The sorting algorithm (Bubble Sort) organizes the list in ascending order.
- **Stage 3**: The sorted list is displayed.