1.⁠ ⁠Requirements Analysis

The team collected all system goals and expectations in this stage. The project needed to build an application which allowed users to create purchase demands for inventory items and generate related sum totals.

Hospital staff can use this system to submit their purchase requests.

It determines all costs from submitted requests.

A status approval depends on how much the total cost reaches.

The system creates new requisition and approval tracking IDs one at a time.

Displays and summarizes requisition statistics.

The system required support for many different requests plus staff information such as employee names, employee IDs and request dates.

2.⁠ ⁠System Design

The system architecture split its requirements into separate parts for better management of the complexity. Key design decisions included:

The system stores all requisition functions in a single class-based design named RequisitionSystem.

Creating methods for:

Collecting staff information (staff\_info)

The system gathers item specifics and determines expenses (Requisitions\_details).

Determining approval status (requisition\_approval)

Displaying output (display\_requisitions)

The system creates new IDs based on its counting system.

Our application saves several requisition records under all\_requisitions list.

Our simple approval process depends only on two possible choices.

3.⁠ ⁠Implementation (Coding)

During this phase the team wrote Python programming code to build the system per design plans. These are the main steps we put into effect:

Input collection via input() functions.

The program deal's with incorrect price data through try-except blocks.

We combine all item prices plus an approval system with conditions to make decisions.

Our system automatically creates new reference numbers for purchase orders once the system approves them.

The system needs to process 4 requisitions with automatic repetition of this operation.

Output display for each requisition and overall statistics.

4.⁠ ⁠Testing

We tested the system by running every function to verify the following aspects.

Accurate total cost calculations.

The system properly identifies approval status by checking the defined limits.

Proper generation of unique requisition and approval IDs.

Output formatting for readability.

Handling of invalid inputs (e.g., non-numeric prices).

Our tests confirmed the system logic by submitting purchase requests at lower, average and increased numbers.