1. Planning  
   I determined in the planning stage that the system would let staff prepare requisitions as well as total costs while evaluating approval requests according to specified monetary thresholds. Key considerations included:  
     
   Effecting process automation works to reduce traditional documentation practices.  
   Staff member identification together with their names will appear for tracking purposes.  
   The system includes auto-approving requisitions that fall below $500 through approval rules.

2. Analysis

During my analysis phase I detailed necessary user specifications while defining system operational procedures.  
  
The system requires users to provide Staff ID and Name together with Date and Item Names and Prices.  
Processes: Total calculation, approval decision.  
Requisitions will generate details that show their current status including the reference number for approvals.  
Using object-oriented programming would help system organization through logical separation of system functions such as data collection and numerical calculation as well as approval management.

3. Design

This system includes the following design structure:

The RequisitionSystem class serves to contain every application functionality.

Methods to:

Collect staff info (staff\_info)

The active method collects item information along with total calculation (requisitions\_details).

Determine approval status (requisition\_approval)

Display requisition details (display\_requisitions)

A unique requisition ID that auto-increments.

A continuous process enables the system to process several requisitions simultaneously.

4. Implementation (Coding)

The system developers used Python to create the implementation under an object-oriented framework. Key implementation points:

A global counter managed the generation of unique requisition identification numbers.

The program acquired user input through input() functionalities.

The conversion function float() protected the program from encountering mathematical errors.

I stored various items through multiple tuples arranged in a list.

The program uses conditional statements to perform automatic approval when the total reaches below $500.

5. Testing

Basic testing took place by performing manual operations.

The program goes through various test inputs to check the system functionality.

The system automatically approves only those requisitions with amounts under $500.

Requisition details print correctly.

Unique IDs are assigned.

The test function test\_requisition\_system() received partial implementation but the developers did not complete it. The system can accept future automation as a method to enhance its reliability.

6. Deployment

The system operates through a console interface approach in its current version. The development phase was executed through terminal-based execution of the Python file. No external dependencies were required.

The application would benefit from web-based deployment through conversion to a web interface or desktop graphical user interface with tools like Flask and Tkinter or PyQt.

**Conclusion**

The SDLC process enabled me to develop a well-structured requisition system. All phases contributed to molding the path and quality which the final end product would take. The foundation established can be developed to create a stronger application system.