1. Lab02  
   1. Software Inspections of User Requirements:

a.) Team for Software Inspections:

* Inspection Leader: [Name]
* Inspector 1: [Name]
* Inspector 2: [Name]
* Developer 1: [Name]
* Developer 2: [Name]

b.) Software Inspections Procedure:

* Introduction:
* Review the purpose and scope of the inspection.
* Emphasize the importance of identifying issues early in the development process.
* Distribution of Requirements:
* Assign sections of user requirements to each inspector.
* Developers review their corresponding sections independently.
* Individual Review:
* Each inspector individually reviews the assigned requirements.
* Developers ensure understanding and feasibility.
* Inspection Meeting:
* Scheduled meeting to discuss findings.
* Inspectors and developers present their observations and concerns.
* Discussion and Clarification:
* Address questions and seek clarification from the requirements author.
* Resolve any discrepancies or ambiguities.
* Documentation of Issues:
* Document identified issues and suggestions for improvement.
* Revised Requirements:
* Author revises the requirements based on feedback.
* Updated version circulated for a follow-up inspection if necessary.
* Approval:
* Once all issues are addressed, the final version is approved.

c.) Requirements Checklist:

RQ1 - RQ16:

Review each requirement against the corresponding checklist items.

Document any deviations, inconsistencies, or missing elements.

2. Self-Code Review:

Code Structure:

Is the code logically organized?

Are functions well-defined with meaningful names?

Variable Naming:

Do variable names accurately represent their purpose?

Are names concise and following a consistent style?

Dynamic Pricing Logic:

Does the pricing logic align with the specified dynamic pricing rules?

Have all discount scenarios been considered?

Function Documentation:

Are functions documented with comments?

Do comments provide clarity on the purpose of each section?

Error Handling:

Is there appropriate error handling for invalid inputs?

Does the code gracefully handle unexpected scenarios?

Execute Self-Code Review:

Go through the code line by line, comparing it against the checklist.

Identify any potential issues, inconsistencies, or areas for improvement.

Make necessary corrections and document any lessons learned.

Ensure the code aligns with the dynamic pricing requirements outlined in Lab 1.

Note: The provided code is a simplified example, and adjustments may be needed based on the actual system requirements and architecture.