1. **Software inspection**
    a. Inspection team – The team will consist of a Moderator, Author, Inspector and Tester. The Moderator will be responsible for controlling the meeting, the Tester will be the one going over the code, being inspected in the meeting, the Inspector will provide insights into improving the process and the functionality being reviewed, whereas the Author will be the one who wrote the code, whose role would be to answer questions related to it.
    b. The inspection procedure involves Planning, Overview, Preparation, Inspection Meeting, Rework, and a Follow-up meeting.
    c. Requirements checklist based on the one in the Lab 02 document.
        i. Each functional requirement relates to the business logic that the system will have, and each non-functional requirement provides some benefit to enhancing the user experience.
        ii. Each requirement possesses a unique id.
        iii. No technical specifications in the requirements
        iv. Requirements are as short and concise as possible and provide no unnecessary details.
        v. Requirements are free of weak words and are verifiable based on their contents alone.
        vi. Non-functional requirements have some passive voice usage.
        vii. The requirements do not state what the system should not do.
        viii. Requirements are complete.
        ix. Different tests for CRUD operation may be derived from some of the requirements.
        x. Requirements do not collide with each other.
        xi. Requirements are all viable.
        xii. There are no preconditions in the requirements.
        xiii. General exception scenarios have been explored.
        xiv. No references between requirements have been made.
        xv. Requirements have not been evaluated against any business value.
        xvi. No priority has been assigned to requirements as they are all to be implemented as a part of the Minimum Viable Product for the railway ticketing system.
2. **Self-code review**
    a. Self-code review checklist
        i. Does the code written fulfill all functionalities described in the relevant requirements?
        ii. Is the code easy to read and understand? In essence, does the code fulfill clean code standards?
        iii. Are error cases properly handled?
        iv. Is the code unit tested?

v. Are coding standards adopted by the team upheld in the code?
vi. Are more common code standards upheld in the code?