**1. Test case id:**

*Each test must have a unique id that can be mapped to the Story id or Req id in the tool used in the team, which helps to keep track and calculate the test coverage. You can refer to a requirement document matrix for the same*

**2. Description:**

*A brief description of what the functionality is about and test scenarios it is derived from, along with a brief idea of the business requirement, would be helpful for a new person to understand it in a better way*

**3. Module Name:**

*The module to which the test steps belong, which helps us later to find out the dependent and impacted modules which need to be prioritized for testing.*

**4. Complexity:**

*What is the complexity of the test case? It can be (Simple, Medium or Complex), which helps us to decide which one to pick first when a time crunch is there. We can simply filter out the complex scenarios and plan the testing.*

**5. Preconditions:**

*The ideal environment to perform the steps for both positive and negative scenarios.*

**6. Test steps:**

*Detailed steps which are followed by the user to perform the actions intended.*

**7. Test Data:**

*A comprehensive test data set that the user has to use to execute the test case. It can be dummy data created in lower environments, but it is always preferred to use real user data to get a better output.*

**8. Expected Results:**

*The result expected after completing the steps. It can be a message or another landing page, an error message (in case of a negative test case), or a confirmation message (for a successful transaction).*

**9. Actual Result:**

*Capturing the system's behavior after the steps are performed and comparing it to the expected result.*

**10. Status:**

*Set the status as PASS/Fail as per the previous step result. If the status is failed create and tag the bug for the specific test case.*

**11. Priority:**

*We need to set the priority of the test cases based on conditions like functionality which has the greatest impact on major workflows of a system and to be executed at all times or based on risks and coverage and functionalities which have less impact and can be neglected for a release.*