"To err is human, but to really foul things up you need a computer." - Paul Ehrlich

Testing is a process focused on the goal of finding defects in the system.

What is Test Case?

A Test Case is a set of actions executed to verify a particular feature or functionality of your software application. Basically, a test case is a document, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement. Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

Test Case Parameters:

Following are the test case parameters, which are considered and included while writing a test case.

- Test Case ID
- Test Scenario
- Test Case Description
- Test Steps
- Prerequisite
- Test Data
- Expected Result
- Test Parameters
- Actual Result
- Environment Information
- Comments

According to IEEE 610, test case includes the following information:

- Pre-conditions
- Set of input values
- Set of expected results
- How to execute the test and check results
- Expected post-conditions

How to write a good test case?

To write a good and effective test case following points must be remembered.

1. Test Cases need to be simple and transparent:

Create test cases that are as simple as possible. They must be clear and concise as the author of test case may not execute them. Use assertive language like "go to home page"," enter data", "click on this" and so on. This makes the understanding the test steps easy and test execution faster.

2. Create Test Case with End User in Mind

Ultimate goal of any software project is to create test cases that meet customer requirements and is easy to use and operate. A tester must create test cases keeping in mind the end user perspective

3. Avoid test case repetition.

Do not repeat test cases. If a test case is needed for executing some other test case, call the test case by its test case id in the pre-condition column

4. Do not Assume

Do not assume functionality and features of your software application while preparing test case. Stick to the Specification Documents.

5. Ensure 100% Coverage

Make sure you write test cases to check all software requirements mentioned in the specification document. Use Traceability Matrix to ensure no functions/conditions is left untested.

6. Test Cases must be identifiable.

Name the test case id such that they are identified easily while tracking defects or identifying a software requirement at a later stage.

7. Implement Testing Techniques

It's not possible to check every possible condition in your software application. Testing techniques help you select a few test cases with the maximum possibility of finding a defect.

- **Boundary Value Analysis (BVA):** As the name suggests it's the technique that defines the testing of boundaries for a specified range of values.
- **Equivalence Partition (EP):** This technique partition the range into equal parts/groups that tend to have the same behavior.
- **State Transition Technique**: This method is used when software behavior changes from one state to another following particular action.
- **Error Guessing Technique:** This is guessing/anticipating the error that may arise while testing.This is not a formal method and takes advantages of a tester's experience with the application

8. Self cleaning

The test case you create must return the test environment to the pre-test state and should not render the test environment unusable. This is especially true for configuration testing.

9. Repeatable and self-standing

The test case should generate the same results every time no matter who tests it

10. Peer Review.

After creating test cases, get them reviewed by your colleagues. Your peers can uncover defects in your test case design, which you may easily miss.

CarMatch Case Study:

Test Case #:	Test Case Name: Process Payment
System: CarMatch System	Subsystem: Pay membership fee
Designed by: Rizwana Noor	Design Date:
Executed by: Sarfraz Ahmad	Execution Date: 03-03-2016
Short Description: Transfer the payment as a membership fee of car sharing scheme.	

Pre Conditions:
1. User must be registered in the system.
2. System must have all information about the user.
3. Car sharer match must be available for user.

Step	**Action**	**Expected System Response**	**Pass / Fail**	**Comment**
1	User enter the ATM card into the machine for transfer of payment.	System check the card credentials and prompts the user to enter PIN.		
2	User enters the PIN.	System checks the card validity and PIN entered by user with the card PIN number.		
3	User click for the transaction of amount.	System displays customer accounts and prompts the customer to choose a type of transaction.		
4	Customer selects Transfer amount, selects the account number, and enters the amount.	System checks the following: • The account is valid • The customer has enough funds in the account, • checks that the ATM has enough amount to transfer.		
5	Click transfer amount button.	System displays a message of “Successful transaction ” and ask for another transaction.		
6	User select “No” option.	System ejected the card.		

Post Condtions:
1. **Amount is deducted from user account.**
2. **Membership fee transferred to CarMatch system account.**

3. User successfully becomes member of car sharing schemes.