## SOFTWARE TESTING LIFE CYCLE:

It contains the below phases:

* + 1. Software test plan.
    2. Software test design.
    3. Test Execution.
    4. Result analysis.
    5. Reporting & BLC.
    6. Delivery and maintenance.
    7. Test summary report/ Build postmortem report.

#### Software test plan:

* + Plan is a strategic document which describes how to perform a task in an effective and efficient way.
  + Software test plan is also a strategic document which describes how to perform testing in an effective and efficient way. The test plan will be prepared by the test lead; once it is prepared it will be sent to the testing team for review.
  + Based on the test plan we are responsible to perform testing.
  + It contains below activities or Index.

### Test plan Index

1. Objective
   1. Scope of testing
2. Reference documents
3. Test Items
   1. Features to be tested
   2. Features not to be tested
4. Test strategy
   1. Testing types
      1. Functional testing types
5. Test environment
6. Test pass/Fail criteria
7. Defect analysis and closure
8. Test Deliverables
9. Automation testing
10. Risks and contingencies
11. Hardware and software requirements
12. Resource plan
13. Test summary report/ Build postmortem report.

##### Objective:

The main purpose of the test plan will be described here. It contains scope of testing.

##### Scope of testing:

What kinds of testing’s the testing team is responsible to test on the application is known as scope of testing?

**Ex**: Testing team is responsible for manual testing and automation for the project.

##### Reference Documents:

The list of documents which the test lead used to prepare the test plan will be described here. Test lead will use SRS documents to prepare the test plan.

##### Test Items:

* 1. **Features to be tested:**

The list of functionalities or modules which the team is responsible for, will be

described here and also the list of testing’s which the testing team is performing on the modules will be described here.

**Ex**: Testing team is responsible for Book a Flight, Book a hotel and managemy booking.

For the above modules they are responsible for manual testing and automation.

##### Features not to be tested:

The list of modules and testing’s which the testing team is not responsible for will be described here.

**Ex:** Testing team is not responsible for payment modules and also they are not responsible for performance testing, Load testing, Stress testing.

##### Test strategy:

*Strategy* means the list of steps which we are going to take to accomplish the plan.

* The test strategy means the list of functional testing types. Which the testing team is going to take to perform testing is known as test strategy.
* We will perform all the functional testing types like regression, re testing, etc… on the application
* **In short, plan means what to do. Strategy means how to achieve the plan.**

##### Test Environment:

Environment means the system which we are going to use to deploy the build and to test the application is known as the test environment.

**Ex:**Machine type : Windows server enterprise OS : Windows

Processor : Intel Xeon CPU

Memory : 4GB/2.13 GHZ

Hard disk : 150GB

Data base : Microsoft SQL server 2008 standard edition Web server : IIS 7.0

Client : Microsoft internet explorer, Firefox, Google chrome

##### Test pass fail/criteria:

If any test case is deviating from the expected result, then it will be treated as failure or bug.

Every bug is having the criteria or bug type.

It is of five types

1. Blocker
2. Very High
3. High
4. Medium
5. Low

##### Defect Analysis Closure:

At the time of delivering the build if any bugs/defects are available it will be analyzed by the testing team with project manager. If any bug is not necessary to be fixed then it will be closed.

##### Test Deliverables:

The list of modules which we are going to deliver to the client known as *deliverable*s.

All the modules will be divided into multiple phases and also the lead will be providing the targeted deadline (delivery date).

|  |  |  |
| --- | --- | --- |
| **Phase No** | **Modules** | **Dead Lines (Date of Delivery)** |
| **1** | 1. **BookaFlight** 2. **ManageMy Booking** 3. **PNR Status** | **30th Jun** |
| **2** | **4. Flight Schedules** | **31st July** |
| **3** | 1. **Corporate Benefit** 2. **Spice connect** | **30th Sept** |

##### Automation testing:

The number of modules which the testing team is going to automate will be described here and also the automation tool and strategy which the test engineers are going to follow will be described here.

##### Risks and contingencies:

The list of risks which the team is going to face while executing the project and also with the related solution will be described here.

|  |  |
| --- | --- |
| **Risks** | **Contingencies** |
| **Resource shortfall** | **Maintain buffer resources** |
| **Continuous Requirement Changes** | **Analyze the requirements** |
| **Lack of peer reviews** | **Monitor Peer reviews** |

##### Hardware & Software requirements:

The number of machines like laptops, mobiles, printers etc… required for the testing with related software will be described here.

##### Resource Plan:

The numbers of resources required for manual testing, automation testing, database testing will be described here.

##### Test summary report/Build postmortem report:

Once the testing is completed the test lead has to prepare the test summary report, it contains the summary of the testing.

#### Software Test Design:

The process of writing the test cases on the test case template after understanding all the requirements is known ‘*software test design’*.

* Every organization will be having their own template based on that template; the test engineer is responsible to write the test cases.
* We are having the below templates to write the test cases. It contains CoverSheet, Test cases, Test data, Traceability matrix and Test Report

##### Cover sheet:

Module name :

Total no. of test cases :

No. of P1 test cases :

No. of P2 test cases :

No. of P3 test cases :

No. of P4 test cases :

**Requirement ID:** The requirement number for which we are writing the test cases will be described here.

**Test types:** The test case type is known as test type. It is of five types.

* + GUI
  + Validation
  + Positive test case (or) Functional positive test case.
  + Negative test case (or) Functional negative test case
  + Database test case

**Positive test case:** Test the application with all the valid data is known as positive test case.

**Negative test case:** Test the application with at least one invalid data is known as Negative test case.

**Priority:** It describes how important the test case is.It is below types: P1, P2,P3 and P4.

**P1:** If the test case is describing the main functionality of the application/module then it will be treated as P1.

The main functionality means if the test case failed we can’t continue the testing further, so that priority is ‘P1’.

**P2:** If the test case is describing the field level functionality then the priority is ‘p2’.

Field level test case means, if it is failed we can continue the testing but it is important to be there in the application as per client requirement.

**P3:** All GUI test cases are comes under P3 priority.

**P4:**Test engineer is having the option to give the suggestion to the application. Those suggestions will be captured in the form of test cases and then the priority is ‘P4’.

**Test case ID:** The serial number of the test case will be described here.

**Test Scenario:**Scenario means a flow or the end user used way. The requirement will be divided into all the end user used flows or scenarios and those will be described here. The test engg has to identify the maximum possible flows(Scenrios) for the requirement or user story

**Pre-condition:**The condition which is required to test the scenario will be described here.

**Test Steps:**The list of steps which are required to execute the scenario will be described here. Based on the test steps the test engg will execute on the application or build

**Expected Result:**At the time of writing the test cases we won’t be having the application with us. So we will be expecting the result for the scenario. That expected result will be updated inthe expected result column.

**Actual Result:**It will be updated at the time of executing the test cases.The test engineer will observe the actual behavior of the application for the scenario and it will be updated here.

**Result:**Once the test execution is completed the test engineer will compare the actual result with expected result, if both are matching then he will update the result as pass, if not he will update it as fail.

**Comments:**The BA or client will provide the comments here.

Refer Gmail login TCs Document

**Test design techniques:**

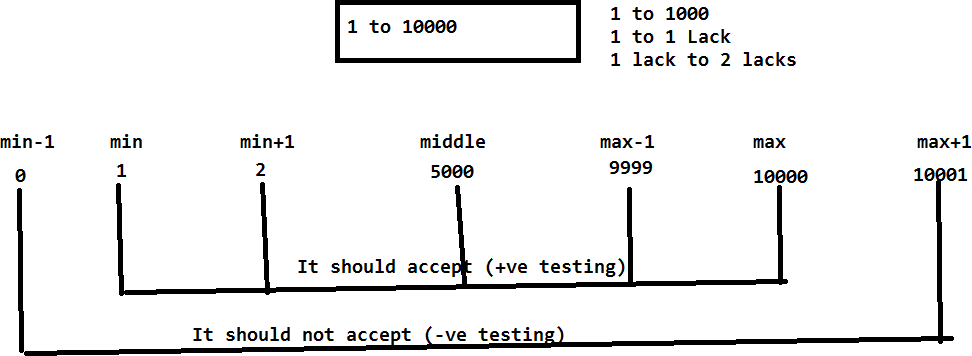
To perform testing in an effective and efficient way, we need to follow the below test design techniques.

* 1. Boundary value analysis (BVA)
  2. Equivalence class partition (ECP)
  3. Error Guessing
  4. Decision Making

##### Boundary value analysis (BVA): 1- 100, 0, 1, 2, 50, 99, 100, 101

Whenever we have the requirement to test the range like 1 to 100 or 1 to 1000 or 1 to 1 lack or 1 lack to 2 lacks then it is not possible to perform the exhaustive testing (complete testing). So we need to apply the BVA technique.

* + Divide the range to multiple boundaries like min-1, min, min+1, middle, max-1, max and max+1.
  + To perform the positive testing, test the field with min, min+1, middle, max-1, and max. Where it should accept. (Its +ve Test case)
  + To perform negative testing, test the field with min-1 and max+1. Where it should not accept. (Its -ve Test case)
  + If it is working as per the above then we can conclude that it is accepting only the range.



**+Ve Test Scenario: Verify the field with the boundaries like** min, min+1, middle, max-1, and max

-Ve Test Scenario: **Verify the field with the boundaries like** min-1 and max+1

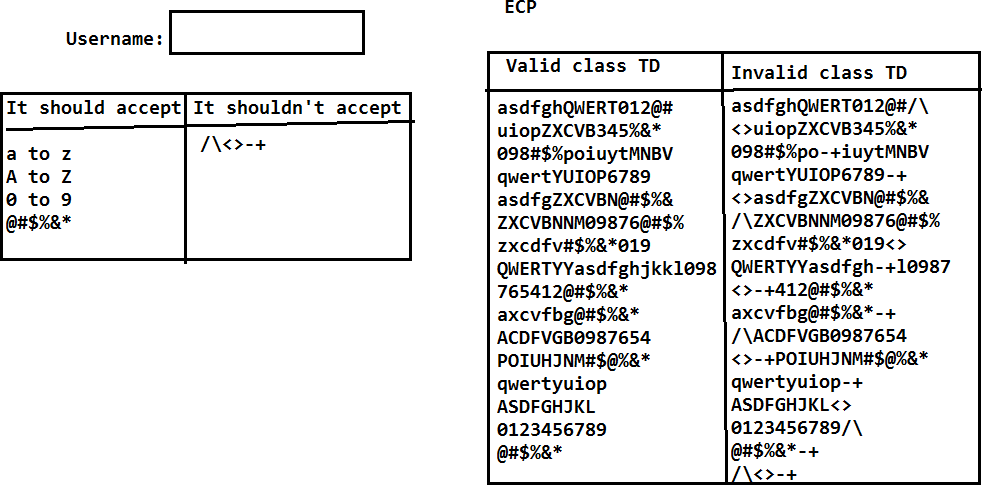
##### Equivalence class partition(ECP):

Whenever we are having the special requirement like check whether the field (user name or password) is accepting the characters like a to z, A to Z, 0 to 9 and #%@$&\*. At the same time the field should not accept the special characters like <>-+/\.

* + In this scenario it’s not possible to perform the exhaustive testing with all the characters. So we need to follow the ECP technique.
  + Divide equally the test data into two classes.

a. Valid test data class b. Invalid test data class

* + Prepare the test data with all the possible ways.
  + To perform positive testing, tests the field with valid test data. Where it has to accept. (Its +ve Test case)
  + To perform negative testing, test the field with invalid test data. Where it should not accept.(Its -Ve Test case)
  + If it is working as expected in the above we can conclude that it working as per the requirement.



##### Error guessing:

Whenever any bug is identified by the test engineer then it should be reported to the developer where he will fix it and send it back to the testing team. The test engineer will check if the bug is really fixed or not. At the same time he has to guess the errors or bugs in the related functionalities. He has to perform the testing in the related functionalities also. It is known as ‘Error guessing’.

Ex: In PR page alert msg is not displaying, it was fixed by developer and tested by the tester. Where the alert msg is properly working in PR page. Now the test engg.. has to go the related functionalities like Admission advice and admission then look (guess) for the similar kind of bug.

##### Requirements Traceability Matrix: RTM

It is to track back whether the test engineer has covered all the test cases for the entire requirements or not.

Based on the traceability matrix the lead or the client will track whether the test engineer has covered all the test cases or not.

|  |  |  |  |
| --- | --- | --- | --- |
| **Req ID** | **No of TCs** | **Test case id** | **Comments** |
| 1 | 1 | 1 |  |
| 2 | 1 | 2 |  |
| 3 | 1 | 3 |  |
| 4 | 1 | 4 |  |
| 5 | 1 | 5 |  |
| 6 | 1 | 6 |  |
| 7 | 1 | 7 |  |
| 8 & 9 | 5 | 8 to 12 |  |
| 10 |  | Not yet Implemented | Requirement is not clear. Waiting for BA comments |

#### Test Execution:

* The process of executing the test cases on the build in test environment is known as test execution. Whenever the build is released to the testing team the test engineer has to review the SRN document to know the build status.
* Based on the SRN document the test lead will deploy the build and the testing team will perform sanity test.
* Once the sanity test is completed, the sanity test results are mailed to the developer.
* If sanity test is passing the testing team will continue to execute the test cases, if sanity test is failed, the testing team will reject the build back to the development team.
* While executing the test cases the test engineer will observe the actual behavior of the application for the scenario and it will be updated under the actual result field. The same will be continued for all the test cases.

#### Result analysis:

* While executing the test cases the test engineer will update the actual result field then he will compare the actual result with expected result, if both are matching then he will provide the result as pass else he will update as fail.
* For pass we will give the green color, whereas for the fail we will provide the red color. Test execution and result analysis, both are parallel process.

**Note:** Once the test cases execution is completed we are responsible to execute all the types of functional testing's on the application to identify the bugs.

##### How many test cases can we write in a day?

It all depends on all the requirements and test engineer but an average we can write around 40-50 test cases in a day. It means we are taking approximately 8-10 minutes for one test case

to analyze the requirement and to prepare the test case on the test case template with the test data.

##### How many test cases we can execute in a day?

It also depends on the test cases and the application but on an average we can execute 50-60 test cases in a day because to review the test case and execute it on the application.

We are taking around 5-8 minutes to execute one test case on an average.

#### Reporting:

* + The process of reporting/sending the bugs (failed test cases) to the developer is known as Reporting.
  + It is two types.
  1. Reporting the Bugs by using XL files.
  2. Reporting the bugs by using reporting tools. JIRA, ALM, BugZilla

##### Report the bugs by using XL file:

It was the old process we used to have the below template to update the bug and send it to the developer.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bug ID | Bug title/ Summary | Status | Severity | Prior ity | Bug descrip tion | Screens hot | Build Versio n | Report ed  By | Assigned  to | Comments |
| 1. |  |  |  |  |  | | | | | |

##### Bug ID:

The serial number of the bug will be described here.

##### Bug title/Summary:

The actual result of the bug will be described here.

##### Status:

Based on the bug the test engineers as well as the developer are responsible to give the status. It is below of types.

New - TL, DL

OPEN - DL, DM

ACCEPTED/REJECTED/DEFERRED/NEED MORE INFO

In Progress

Fixed

Ready for Retest

Reopen/ Closed

ACCEPTED

In Progress

Fixed

Ready for Retest

Closed

##### New:

Whenever the test engineer identifies any bug. Initially the status of the bug is New. The new bug will be reported to the developer.

##### Open:

The developer will check that whether the new bug is really a bug or not. If yes then we will update the status from new to open.

##### Fixed/Verified:

Developer will take some time to fix the open bug once it is fixed he will update the status from open to fixed. Fixed bug will be sent to the test engineer.

##### Closed:

Test engineer will check whether the fixed bug is really working as expecting or not. If it is working then we will update the status from fixed to closed. Closed state is the end of the Bug.

##### Re open:

The fixed bug will be tested by the test engineer; if it is not working as expected then he will update the status from fixed to reopen and the reopen bug will be sent back to the developer.

The developer will check that whether it is really a bug or not, if yes he opens it, fix the bug and send it back to the testing team.

##### Rejected/Not a Bug/Hold/Differed:

When the test engineer identified any bug it will be reported to the developer with new status. If developer is not accepting a bug then he will update a status from new to Rejected/Not a bug and it will be sent back to the testing team.

##### Severity:

It describes how seriously the bug impacted the application on testing. Severity means seriousness of the bug. It is below types.

##### Blocker:

If the bug is blocking the entire testing of the module then the severity or type of a bug is Blocker.

##### Very high:

If the bug is blocking partially the testing of the module then the severity of the bug is very high.

##### High:

If the bug is blocking only the specific scenario of the module then the severity is high.

##### Medium:

All GUI bugs severity is medium.

##### Low:

Test engineer is having the option to give the suggestion also. So the suggestion will be reported in the form of bug, where the severity is low.

##### Priority:

Priority describes in which order the bug has to be fixed by the developer. Based on the severity the test engineer will provide priority to the bug as below

|  |  |
| --- | --- |
| **Severity** | **Priority** |
| Blocker/Urgent/critical Very high  High  Medium Low | P1 P2 P3 P4  P5 |

##### Description:

The detailed steps to produce/get the bug will be described here. Based on the steps developer will check that whether it is really a bug or not.

##### Screenshot:

The test engineer will capture the screenshot of the bug and it will be uploaded in the bug template. It is to prove the reported bug is valid and also to understand about the bug.

##### Build version:

The build number on which the test engineer identified the bug will be described here.

##### Reported by:

The test engineer who identified the bug will describe here.

**Assign to:**The developer name or the developer lead name, who is going to fix the bug will be described here.

##### Comments:

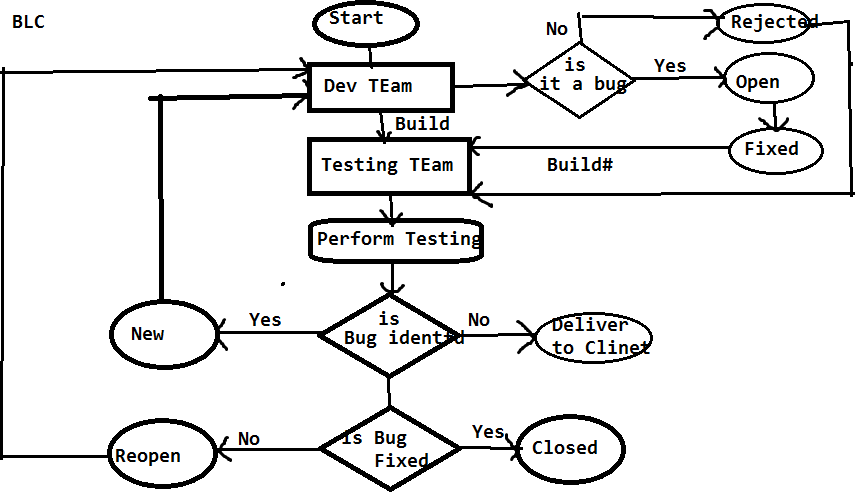
Both test engineers, developer can ask the questions in the form of comments.

**Note:**The XL file reporting will consume lots of time so our plan is to use the reporting tools.

Ex:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Phases** | **BUG ID** | **Bug Title / Summary** | **Status** | **Severity** | **Priority** | **Bug Description** | **Screenshot** |
| I | 1 | Application is not displaying both date  pickers | New | Blocker | P1 | 1. Open Spicejet.com 2. Click on Roundtrip radio button 3. Application is not displaying   both the date pickers | D:\Nagesh\SPicejet |
| II | 2 | Spicejet name is displaying as spacejet | New | Medium | **P4** | 1. Open Spicejet.com 2. Observe the Spicejet logo 3. Its displaying as spacejet | D:\Nagesh\SPicejet |
| III | 3 | Oneway radio button is not  displaying | New | VeryHigh | P2 | 1. Open Spicejet.com 2. Oneway radio button is not available | Path |
| I | 4 | Student check box is not  available | New | High | P3 | 1. Open Spicejet.com 2. Student check box is not available | Path |
| II | 5 | Cheange the color of spicejet home page  to blue | New | Low | P5 | 1. Open Spicejet.com | Path |
| III | 6 | Spicejet club link is not navigating  to Spicejet club page | New | Blocker | **P1** | 1. Open Spicejet.com 2. Click on Spiceje Connect link 3. Spiceje Connect link is not navigating to MySpicetrip page | Path |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| I | 7 | Application is not maintaing GUI | New | Medium | P4 | 1. Open <http://selenium4testing.com/hms> 2. Login into the application 3. Click on Search Registration 4. Application is not maintaining GUI | [Spicejet\_GUI.png](file://localhost/D:/Nagesh/Spicejet/Spicejet_GUI.png) |
| II | 8 | Admission worklist GUI is not maintaining  properly | New | Medium | p4 | 1. Open <http://selenium4testing.com/hms> 2. Login into hms with user1/user1 3. Click on ADT 4. Click on Admission worklist 5. Observe the GUI, its not maintaining properly | D:\Nagesh\hms\_AD |
| III | 9 | Adult field is not displaying on the  page | New | Blocker | P1 | 1. Open [http://spicejet.com](http://spicejet.com/) 2. Observe all the fields 3. Adult dropdown is not   available | D:\Nagesh\Spicejet |
| I | 1 | Application is not displaying Hyderabad and Bangalore | New | VeryHigh | P2 | 1. Open [http://spicejet.com](http://spicejet.com/) 2. Click on LeavingFrom field 3. Application is not displaying Hyderabad and Bangalore | D:\Nagesh\Spicejet |



Q: what is the difference between severity and Priority

Q: what is thedifference between Priority in test cases and Priority in bug template

**Q.**If the developer is not accepting your bug then how you will prove that yours is valid bug?

**A:** Based on the bug description, SRS document, screenshot we will try to prove that the bug is valid if is not accepting it then I will take the sever log to prove the bug is valid, if still not accepting it and then I will send it to a BA, project manager and finally client.

**Q.** Explain the scenario where the bug is having high severity with low priority and low security with high priority?

##### A:Severity Priority

Blocker - P1

High security Very high - P2 High priority High - P3

Medium - P4

Low security Low - P5 Low priority

We have two bugs one is Blocker another one is medium. The blocker will be having high priority and medium will be having low priority.

Based on the severity the test engg.. will provide priority. Based on the priority the dev team is responsible to fix

But the development lead is having the option to change the priority, depends on the situation.

* The bugs which are related to the current phase delivery will be converted to high priority irrespective of the severity.
* The bugs which are not part of the current delivery will be converted to low priority irrespective of the severity.

##### Phase Bug Id Bug title/Summary Status Severity Priority

* 1. **1.** Spice jetname is displaying New Medium P4---P1

As space jet

1. 2. Spice jet connect link is not New Blocker P1 P4

Navigating spice jet connect page

##### Test Report/Build status report:

Once the test case execution is completed on the build then test engineer is responsible to send a test report to the lead as well as client. It below format **Build status Report/Test Report**

|  |  |
| --- | --- |
| **Build Status Report / Test Report** | |
| **Test Engg Name:** |  |
| Build No | 1 |
| Login Credentials |  |
| Browser | FF, IE GoogleChrome |
|  |  |
| **Test Matrics** |  |
| Total no of testcases | 200 |
| No of Test cases executed | 150 |

|  |  |
| --- | --- |
| No of Test cases pending | 50 |
| No of Test cases Pass | 100 |
| No of Test cases Fail | 50 |
| No of Test cases Skipped | 10 |
| No of Bugs Reported | 3 |

##### Test Metrics:

Metrics means measurement of the task. Test metrics means measurement of the testing.

##### Pending:

If the developer has not giving functionality at all then those test cases can’t be executed. It’s comes under pending.

##### Skipped:

Developer has given the functionality, but we are unable to test the functionalities, because of the dependent functionalities failed.

Ex: if Login failed, we can’t execute compose. Compose test cases comes under skipped.

* The reporting will be continued until the build is stable, stable means majority of the test cases are pass and no blocker bugs are available in the reporting tool.
* The stable build will be delivered to the client.

##### Q: Explain me the reporting structure in your organization

Report the Bugs By using Reporting Tools:

* Any reporting tool having two kinds of users: One is admin user and another one is End user.
* Admin user: The admin user is responsible to create the project, create users like test engineers, developers…etc. He will assign the user to the project
* End user: He is responsible to use (report) or receive the bugs ex: test engineers, developers…etc.

Ex: QC, JIRA, Bugzilla, Redmine, Test manager etc… BugZilla:

* Access the Bugzilla by using selenium4testing.com
* Then click on Bugzilla.
* Login into the Bugzilla as a test engineer ([jan30selenium@gmail.com](mailto:jan30selenium@gmail.com)& password : selenium)
* By using Bugzilla we can perform below activities.

1. Reporting a Bug.
2. Search for the bugs.
3. We can take the report.
4. Preference.