FAQ’s:

1. Introduce about yourself
2. Tell me about your current project
3. Tell me about your roles and responsibilities
4. Tell me one critical defect you found

When I performed functional testing on the MMP project, I noticed that the UI was accepting and displaying multiple patients with different names but the same SSN. I reviewed the MMP data dictionary document and noticed that the PATIENT table had a unique constraint on the SSN column. But within the TOAD database management toolset, I realized the constraint was missing.

I reviewed the DB creation scripts in Git and discovered that the “CREATE PATIENT table add constraint” statement was missing. I wrote a defect in JIRA socimenting this.

Another defect I discovered just last month was that a patient (low privilege user) can log in as administrator. This is a critical defect because this allows a patient access the medical reports and profiles of other patients.

1. Explain me about STLC (Software Testing Life Cycle)

The STLC consists of a series of stages, each performed sequentially to verify the quality of a software product. Each stage has specific activities, deliverables, and entry and exit criteria.



Requirements analysis: Here the QA analyst reviews requirements and identifies which requirements can be tested. Test environment is reviewed. Perform automation feasibility analysis. Deliverables: RTM, Automation feasibility report.

Test Planning: QA team performs cost and effort estimates for the project. Test tools (if any) are selected. Resources are planned(how many testers, which modules will they test). Deliverables: Test plan document. Effort estimation document.

Test Case Development: Test cases are reworked. Test data is generated. Automation test code is written. Deliverables: Test data, test scripts, reworked test case document

Test Environment setup: With an understanding of the application architecture (with input from Dev team) and the operating environment (OS) of the AUT, the QA team sets up the test machines and other equipment needed for test case execution. Test Data is populated in the Databases. This stage can be performed in parallel with the Test case development stage.

Test Execution: During this phase, the testers will carry out the testing based on the test plans and the test cases prepared. Bugs will be reported back to the development team for correction and retesting will be performed.

Test Cycle Closure: Testing team will meet, discuss and analyze results and strategies that have to be implemented in the future, taking lessons from the current test cycle. The idea is to remove the process bottlenecks for future test cycles and share best practices for any similar projects in the future.

1. What are the test case methodologies that you used in your project?

MMP uses the Agile methodology. In Agile methodology, software is developed in incremental, rapid cycles. Interactions amongst customers, developers and client are emphasized rather than processes and tools. The agile methodology focuses on responding to change rather than extensive planning.

1. What are black box testing techniques that are available (Here question 6 and 7 are same)

Black box testing is a testing technique in which the functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications. we just focus on inputs and output of the software system without bothering about internal knowledge of the software.

Various blackbox techniques are:

* **Equivalence Class Testing:** It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.

***Equivalence Partitioning Technique*** is also known as ***Equivalence Class Partitioning or ECP***. The idea behind this technique is to divide/partition a set of test conditions into groups or sets that can be considered the same or equivalence .This is used to reduce the total number of test cases to a finite set of testable test cases, still covering maximum requirements.

One test value is picked from each class while testing. This is because we are assuming that all the conditions in one partition will be treated in the same way by the software. If one condition in a partition works, we assume all of the conditions in that partition will work and if one condition fails it is assumed that all others in the partition will fail and there is no point in testing others.

* **Boundary Value Testing:** Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test cases. It is most suitable for the systems where an input is within certain ranges.
* **Decision Table Testing**: A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

1. Tell me what test cases you write for the field which is taking values range from 1 to 1000. Use Boundary Value Analysis/ECP.

***Test cases for input box accepting numbers between 1 and 1000 using***

***Equivalence Partitioning:***

1. *One input data class with all valid inputs. Pick a single value from range 1 to 1000 as a valid test case. If you select other values between 1 and 1000 then result is going to be same. So one test case for valid input data should be sufficient.*
2. *Input data class with all values below lower limit. I.e. any value below 1, as a invalid input data test case.*
3. *Input data with any value greater than 1000 to represent third invalid input class.*

So using *equivalence partitioning* you have categorized all possible test cases into three classes and that can be (-10, 100 & 1010). Test cases with other values from any class should give you the same result.

Boundary Value Analysis:

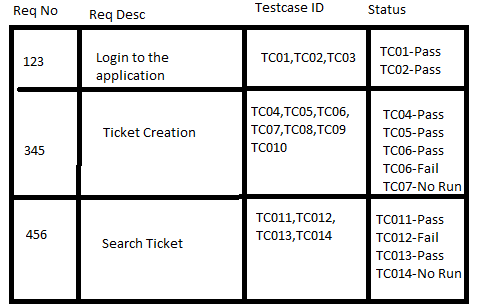
Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values. The basic idea in boundary value testing is to select input variable values at their:

1. Minimum (1)
2. Just above the minimum (3)
3. A nominal value (52)
4. Just below the maximum (98)
5. Maximum (100)
6. What is Requirement Traceability Matrix and why we use them. Can you explain to me the template of the RTM

Requirement Traceability Matrix or RTM is a document that maps and traces user requirement with test cases. The main purpose of Requirement Traceability Matrix is to see that all test cases are covered so that no functionality should be missed while doing Software testing.

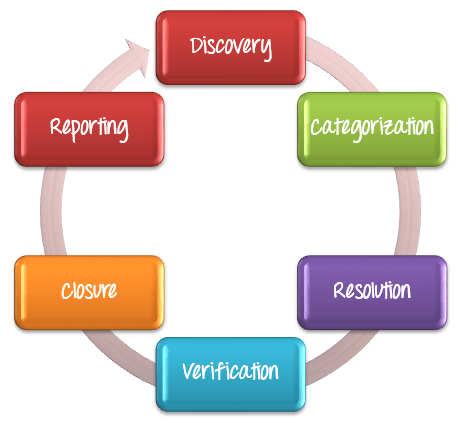
A RTM must contain at least the following fields:

* Requirement ID
* Requirement Type and Description
* Test Cases with Status



1. Explain to me about Defect Management Process

DMP is the process used by the QA and Dev teams to discover, categorize, report, resolve, verify and close defects.



Discovery: The QA team executes test plans and discovers a defect. Defect state is accepted (if Dev Team accepts it)

Categorization: QA Lead categorizes a defect by priority into critical/high/medium/low. Defect state is categorized.

Resolution: Dev Team modifies code to address the defect. Defect state is Resolved.

Verification: After the development team **fixes** the defect, the testing team **verifies** that the defects are actually resolved. Defect state is Verified. If the QA team cannot verify that the defect is fixed, it is sent back to the Dev Team. Defect state is verification failed.

Closure: Once a defect has been resolved and verified, the defect is changed status as **closed**.

The defect is tracked using tracking tools like JIRA, clearquest so that all stakeholders are updated about its status.

1. Tell me what are the different status of defects(Here questions 10 and 11 are almost same)

See above.

1. What is the difference between Severity and Priority

Severity is the degree of impact a[Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)has on the development or operation of a component application being tested.

1. How do you decide Severity and Priority

*Defect severity can be categorized into four class*

* **Critical**: This defect indicates complete shut-down of the process, nothing can proceed further
* **Major**: It is a highly severe defect and collapses the system. However, certain parts of the system remain functional
* **Medium**: It causes some undesirable behavior, but the system is still functional
* **Low**: It won't cause any major break-down of the system

Priority is defined as the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved.

*Defect priority can be categorized into three class*

* **Low:**The Defect is an irritant but repair can be done once the more serious Defect has been fixed
* **Medium:**During the normal course of the development activities defect should be resolved. It can wait until a new version is created
* **High:**The defect must be resolved as soon as possible as it affects the system severely and cannot be used until it is fixed

1. Give me one example for severity is Critical and Priority as low

For a flight operating website, a defect in reservation functionality may be of high severity but can be a low priority as it can be scheduled to release in a next cycle.

1. Give me one example for severity is Low and Priority as high.

 A logo error for any shipment website, is normally considered low severity as it not going to affect the functionality of the website. But if the customer does not want to accept the software with the wrong logo, it becomes a high priority defect.

1. What is Regression Testing and why we need to this testing

What: Regression Testing is used to confirm that a recent program or code change has not adversely affected existing features. It is a full or partial selection of already executed test cases which are re-executed to ensure existing functionalities work fine.

Why: Regression testing is done to make sure that new code changes should not have side effects on existing functionalities. It ensures that the old code still works once the new code changes are done.

1. How do you decide which test cases has to be considered for Regression Testing

A good number of the defects reported by customers were due to last minute bug fixes creating side effects. Selecting the[Test Cases](https://www.guru99.com/test-case.html)for regression testing is an art and not that easy.  It requires thorough knowledge of end user requirements. Effective Regression Tests can be done by selecting the following test cases -

* Test cases which have frequent defects
* Functionalities which are more visible to the users
* Test cases which verify core features of the product
* Test cases of Functionalities which has undergone more and recent changes
* All Integration Test Cases
* All Complex Test Cases
* Boundary value test cases
* A sample of Successful test cases
* A sample of Failure test cases

1. What is Build Verification Test

Build Verification Test is a set of tests run on every new build to verify that build is testable before it is released to test team for further testing. These test cases are core functionality test cases that ensure application is stable and can be tested thoroughly.

The build acceptance test is generally a short set of tests, which exercises the mainstream functionality of the application software. Any build that fails the build verification test is rejected, and testing continues on the previous build (provided there has been at least one build that has passed the acceptance test).

1. What is Smoke Test(Here questions 18 and 19 both are almost the same)

A “Smoke Test” is a cursory test that ensures the basic functionality of the application works. The primary outcome is validating that the build can be considered for further testing, and ultimately early problem detection. If a smoke test fails, there are serious blocking issues that need to be addressed quickly. Early detection is important so that a larger group of people does not become blocked by installing a bad build.

**Smoke tests are:**

* Extremely fast to run
* Ideally, automated
* Run on every build, including interim builds generated throughout the day
* Focus only on critical functions (e.g. a user can login)
* Shallow and wide (touch as many parts of the app as possible)
* Small in number (i.e. 1 or a few smoke tests per build)
* Typically built into other, more thorough, test cases
* Executable in minutes (not hours)

The term “Smoke Test” comes from the hardware industry. Once a circuit is created, the very first test that a hardware engineer performs is turning on the power. If it catches fire and/or smokes, you know you have a problem. No functionality is tested at this point, other than powering on. No smoke means that further functional testing can be performed (or that you have no power at all!).

BVTs, or Build Verification Tests (also called Build Acceptance Tests, or Sanity Tests),  are a superset of smoke tests, but in some cases the terms are used interchangeably. BVTs are the top priority test cases that exercise basic functionality in the build, slightly more thoroughly than smoke tests. BVTs ensure that the daily build is usable for testing.

BVTs are:

* Still run as part of the overall test pass (your highest priority tests)
* Typically automated
* Run immediately after a daily build is produced
* Executable in minutes
* Executable in minutes (not hours)

1. What is Sanity Testing?

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing.

Sanity testing is very similar to smoke testing, but the following differences are notable.

Smoke testing performed on a particular build is also known as a build verification test.

Smoke testing exercises the entire system from end to end.

Sanity testing exercises only the modules that changed in the build, not the entire system.

Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine

Sanity Testing is done to check the new functionality/bugs have been fixed.

1. Difference between Regression and Sanity Testing.

Sanity Testing is a surface-level testing, meaning that a tester checks whether whole functionality of the software works in a proper way. Regression Testing does not imply a surface-level testing.

But Sanity Testing is a part of Regression Testing.

Testers carry out Sanity Testing if they have limited time for performing testing. Regression Testing is carried out when there is enough time for it.

Testers perform Sanity Testing manually. Testers can perform Regression Testing manually or with the help of different [automated tools](http://qatestlab.com/resources/whitepapers/guide-to-test-automation-tools-2017-2018/).

Sanity Testing does not influence the product cost. Since Regression Testing takes much time and efforts, it increases the product cost.

1. Difference between Regression and Re-Testing

Regression Testing is carried out to confirm whether a recent program or code change has not adversely affected existing features

Re-testing is carried out to confirm the test cases that failed in the final execution are passing after the defects are fixed

Defect verification is not the part of Regression Testing

Defect verification is an important part of re-testing

Regression testing checks for unexpected side-effects

Re-testing makes sure that the original fault has been corrected

Regression testing is only done when there is any modification or an

Re-testing executes a defect with the same data and the same environment with different inputs with a new build

1. Did you do Compatibility Testing if so what are the browsers and Operating Systems used to do. Did you find any differences.

Compatibility Testing is a type of Software testing to check whether the software is capable of running on different hardware, operating systems, applications, network environments or[Mobile](https://www.guru99.com/mobile-testing.html)devices.

I performed cross browser automated testing while using Selenium to automate login and register patient functions within MMP. The “Browser Type” parameter in testing xml file allowed me to test the MMP application with Chrome, Edge, and Firefox. Functionality worked in all 3 browsers.

1. What is difference between Verification & Validation

Verification is the process of checking documentation (test plans, design documentation, RTM) during internal review meetings, walkthroughs or inspections. Verifications finds defects early in the QA cycle and does not involve executing the AUT.

Valoidation is the process of testing the AUT by executing test plans. Validation is performed after verification and does involve executing the AUT.

1. What is static testing and dynamic testing

Under**Static Testing (Verification)**, code is not executed. Rather it manually checks the code, requirement documents, and design documents to find errors. Hence, the name "static". Static testing involves manual or automated reviews of the documents. This review is done during an initial phase of testing to catch[Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)early in STLC. It examines work documents and provides review comments.

Dynamic testing (validation) executes the software and validates the output with the expected outcome. Dynamic testing is performed at all levels of testing and it can be either black or white box testing.

1. You raised defect and developer is not accepting the defect what you do?

I would carefully consider the input from the developer.

I would then review the business requirements document and the test cases/plans derived from it using the RTM. This would me allow me to verify that the test case and the test inputs are valid. If I still feel there is a defect, I will provide additional evidence of failure (screenshots/video recordings). If necessary I will try to reproduce the defect at the developer’s desk. If there is still disagreement I would escalate the issue by seeking input from the QA manager.

1. How do you make sure you cover all the testing for given requirements? OR How do you make sure you have not missed anything in your testing(Here answer is question 9 i.e. RTM)

See answer to question 9.

1. What you do if sufficient time is not given for testing

I would work with the QA manager to identify high risk test cases that can be tested in the timeframe available. Then execute these test cases as efficiently as possible.

Criteria for identifying high risk test cases:

* Which functionality is most visible to the user?
* Which functionality has the largest safety impact?
* Which functionality has the largest financial impact on users?
* Which aspects of the application are most important to the customer?
* Which parts of the code are most complex, and thus most subject to errors?
* Which parts of the application were developed in rush or panic mode?
* Which aspects of similar/related previous projects caused problems?
* Which aspects of similar/related previous projects had large maintenance expenses?
* Which parts of the requirements and design are unclear or poorly thought out?
* What do the developers think are the highest-risk aspects of the application?
* What kinds of problems would cause the worst publicity?
* What kinds of problems would cause the most customer service complaints?

1. Difference between Test Scenario and Test Case

A Test Scenario is defined as any functionality that can be tested. It is a collection of test cases which helps the testing team verify a sequence (workflow) of user actions. Example Test scenarios: Patient Search, Patient registration etc.

A Test Case is a set of actions executed to verify a particular feature or functionality of the AUT. The Test Case has a specific test data, preconditions, certain expected and actual results developed for a specific test scenario. Thus a test case is a step within a test scenario. Example Test case for Patient Search Test scenario: In the text box enter string “Solomon” and hit “Submit”. Verify that the page displays all patient records with patient name “Solomon”.

1. When to stop testing

Exit criteria for a QA test project are shown below. Meeting any one of these criteria can be enough to stop testing.

* 1. Delivered build fails smoke testing. Build has to be rejected because basic functionality is broken.
  2. Out of time. SW has been tested reasonably well and we are out of time.

© Test cases completed with certain percentage passed and desired test coverage is achieved.

* 1. Bug rate falls below a certain level, and testers are not getting any high priority bugs.

1. What is Risk-Based Testing

There will be a very large number of test cases that need to be run to completely test an application. There may not be enough time to execute all test cases. RBT identifies and prioritizes the most critical test cases which must pass before the software can be released. Knowledge of the domain is required to identify these test cases. If all critical test cases pass, then other test cases can also be executed.

1. What is Alpha and Beta Testing

Alpha testing verifies that a SW product meets requirements in-house (that is at the development site). It is performed by the QA team in the factory prior to release. Beta testing is not done by the QA team, it is done by end-users at the customer site. Beta testing comes after alpha testing is complete and there is confidence that all requirements are met.

1. What UAT and did you ever involve in this testing

UAT is user-acceptance-testing. It is testing that involves the end users of the SW product so that they can decide if the product is good enough to be accepted. It is the last chance for a SW to be tested to see if it meets all requirements before it goes live. Hence it comes after the QA team has already completed system testing and verified that UAT can be done. The QA team can also be involved in the UAT to demonstrate the features to the customers.

At TMS, I had the opportunity to take part in UATs. I was the TMS QA representative in a UAT of OrbCAD for delivery to the SFMTA (San Francisco Municipal Transit Agency). The UAT lasted 10 days (2 weeks) during which I executed acceptance test procedures and demonstrated the features of OrbCAD. Some bugs were discovered during the UAT and TMS agreed to delayed acceptance until most of the bugs were fixed. Overall the customer was pleased with the features the product offered and were satisfied that requirements were being met.

1. What is difference between V-Model and Agile model(this can be with other model as well like Water Model)

**V**- **model** is an extension of the waterfall **model**. But unlilke the waterfall model, the development phase itself is broken into sub-phases.

In **V**-**model**, there is a corresponding testing phase for each software development phase. Testing in **V**-**model** is done in parallel to SDLC stage. Testing is done as a sub project of SDLC. So there is cross-communication between developers and testers within each sub-phase of the SDev phase. However other teams (business analysts, customers) are still not involved. This is the major difference between Agile and the V model.

1. What is the difference between the Waterfall model and the Agile model?

Waterfall is a sequential model of SW development where the SDLC is divided into phases: a requirements gathering phase, a design phase, a development phase, a Test phase. Each phase follows only after the prior phase ends. It can be good for projects where requirements are clearly defined and are not expected to change once the SDLC begins. Participants in each phase are distinct (Business analysts are involved in the RG phase, BAs and Developers in the design phase, developers in the Development phase, and QA team in test phase, customers in the UAT phase).

In agile, the SDLC is divided into short duration(2-3 week) sprints or iterations with a planned deliverable. Each sprint involves all the stake-holders (Business analysts, developers, QA team, and client/customer representatives). Hence there is greater communication and feedback among teams. Missed requirements and bugs are caught early. Hence the agile model is more flexible.

Waterfall model is project-focussed. Agile model is customer focussed.

In the waterfall model there is great emphasis on written documentation which acts as the interface between various phases of the SDLC. In the Agile model daily stand-up meetings allow the team to communicate verbally and there is less reliance on written documentation.

1. What is Defect Removal Efficiency

The defect removal efficiency (DRE) gives a measure of the development team ability to remove defects prior to release. It is calculated as a ratio of defects resolved to total number of defects found. It is typically measured prior and at the moment of release. DRE can be measured at each phase of the SDLC.

An average DRE score is usually around 85% across a full testing programme, however with a thorough and comprehensive requirements and design inspection process this can be expected to lift to around 95%.

1. Difference between Defect , Bug and Error

A mistake in coding is called error, an error found by a tester is called a defect, a defect accepted by the development team is called a bug.

1. What are different Test Levels

From conception to completion a software evolves in stages. The testing performed at each stage of the SDLC is called a level.

|  |  |
| --- | --- |
| [Unit Testing](http://softwaretestingfundamentals.com/unit-testing/) | A level of the software testing process where individual units of a software are tested. The purpose is to validate that each unit of the software performs as designed. |
| [Integration Testing](http://softwaretestingfundamentals.com/integration-testing/) | A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. |
| [System Testing](http://softwaretestingfundamentals.com/system-testing/) | A level of the software testing process where a complete, integrated system is tested. The purpose of this test is to evaluate the system’s compliance with the specified requirements. |
| [Acceptance Testing](http://softwaretestingfundamentals.com/acceptance-testing/) | A level of the software testing process where a system is tested for acceptability. The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery. |

1. What is difference between Test metrics and Traceability matrix?

A Test metric is something that you measure that can improve the QA process.

A test metric must be MMATT (measurable, maintained and updated regularly, actionable, trackable over time, tied to business goals). Test metric examples: Time to test, time to fix, escaped bugs)

A traceability matrix is a document/table that maps requirements to test cases. You could have requirements in the columns and test cases along the rows. An X in a cell indicates that a particular test case adequately verifies a requirement. Every requirement must be “covered” by at least one test case.