Manual and Mobile Testing Metrics.

MANUAL TESTING CONCEPTS.

1.What is software testing?

Software Testing is the process of Verification and Validation of Implicit and Explicit Customer Requirements for Positive and Negative Test Scenarios.

Where, **Verification** is inspection of physical layouts and interface according to the customer requirements.

Validation is observing the behaviour of internal functionalities according to the requirements.

Implicit Requirements are the unstated default features/functionalities.

Explicit Requirements are the customer stated features/functionalities.

Test Scenarios describes any functionalities that can be tested.

Positive Scenarios are the acceptable/expected values/outcomes of the cases/functionalities.

Negative Scenarios are the unacceptable/unexpected values/outcomes of the cases/functionalities.

2. What is a Test Case?

A Test Case is a documented report, which is a step by step instruction to repeat user work flow/scenarios for +ve and -ve test conditions with respect to specified requirements.

Structure of a Test Case:

1. Test Case ID- unique ID assigned to test case.
2. Test Description- Steps/events to be followed/performed to achieve functionality.

Manual and Mobile Testing Metrics.

3. Pre-Conditions – Conditions or pre requisites to achieve functionality
4. Test Data – Data required to perform functionality.
5. Expected Result – The action which the steps/event expected to perform.
6. Actual Result – The action which steps/events actually performed.
7. Test Result – Assessing Test case is Pass/Fail according expected and actual results.

Sample Test Case:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1																13/08/2020			
2																			
3	Testcase ID		Testcase_Description			Precondition			Testdata			Expected_result			Actual_result			Result(Pass/Fail)	
4	TC1		Log In into Facebook with valid credentials 1. Go to homepage of Facebook Screen			Browser should be installed.			https://www.facebook.com/			Opens Facebook login page.			facebook login page displayed			pass	
5			2. Place cursor in Email or phone field			User must have a facebook account.			abc@gmail.com										
6			3. Enter a valid Email or phone																
7			4. Place cursor in password field						********										
8			5. Enter valid password																
9			6. Click on Log in button									Navigates to your facebook newsfeed page.			successfully navigated to newsfeed			pass	
10																			
11																			
12	TC2		Log In into Facebook with invalid credentials 1. Go to homepage of Facebook Screen			Browser should be installed.			https://www.facebook.com/			Opens Facebook login page.			facebook login page displayed			pass	
13			2. Place cursor in Email or phone field			User must have a facebook account.			++abc@gmail. com										
14			3. Enter a **invalid** Email or phone																
15			4. Place cursor in password field						********										
16			5. Enter valid password																
17			6. Click on Log in button									A message displaying "wrong credentials".			error message thrown successfully			pass	
18																			

3. Roles and Responsibilities of a software Test Engineer.

- Analyse the requirement document which is named differently among the companies like SRS(software requirement specifications), CRS(client requirement specification), BRS(business requirement specifications) etc., and prepare a detailed understandable document.
- Should be able to involve in preparing test plan.

  Where, **Test Plan** is a document having the systematic and detail approach for testing a software.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Test plan contents are: 1. **Scope and Objectives**

2. Approach

3. Resources Required

4. **Time Estimation**

5. Risks

- Should be able to prepare test scenarios/testcases according to requirements.
- Should be able to understand and execute the test cases which helps to validate the test coverage of a software.

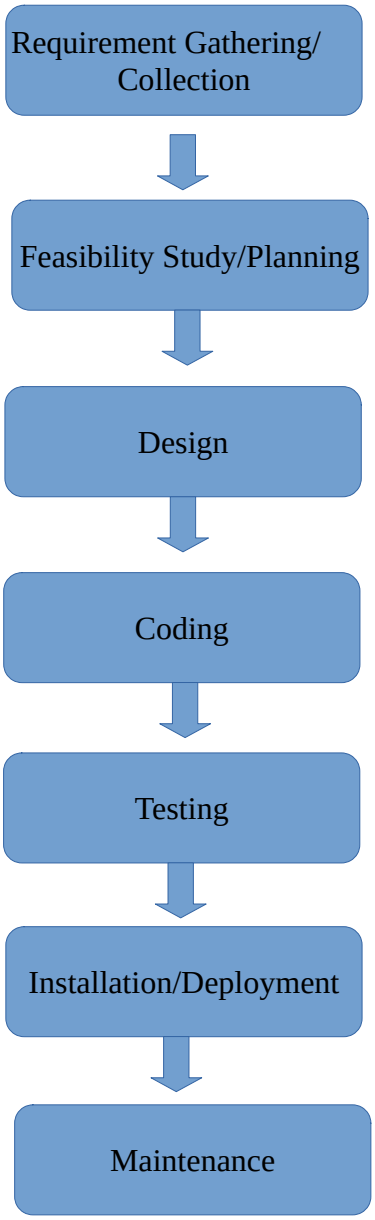
  Where, **Test Coverage** is the ratio of total number of test cases executed to the total number of test cases applicable to the software.

  Note: Test Coverage should be 100% to release a software to the end users/customers.
- Should be able to track the defects and should prepare a detailed log of it while reverting back to Developers.
- Where, **Defect** is the deviation between actual result to the expected result with respect to the customer requirements.
- Should be able to prepare status reports regularly which helps for further references.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

4. Software Development Life Cycle S.D.L.C.



Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Requirement Gathering/ Collection:

Business Analysts of the company communicates with the clients and gather the requirements which will be in the form of documents namely called as SRS, CRS, BRS, BRD etc.,

Feasibility Study/ Planning:

The group of business analysts, Hr's, Financier's, Architects of the company studies the business requirements of the clients and plans according to budget, resources, time etc.,

Design:

Further to the study and planning the software's internal and external development is done by using High-level and Low-level designing.

Where, **High-level designing** is the overall system designing covering the architecture and database design.

Low-level designing is the design detailing of the High-level using flow charts, class diagrams etc.,

Coding:

developers develops the software according to the design using the suitable stated tools.

Testing:

test engineers make sure to outsource a quality and efficient products using various testing techniques.

Deployment:

after the completion of testing and defect fixing resulting a 100% test coverage the product is released to the end users i.e., the customers.

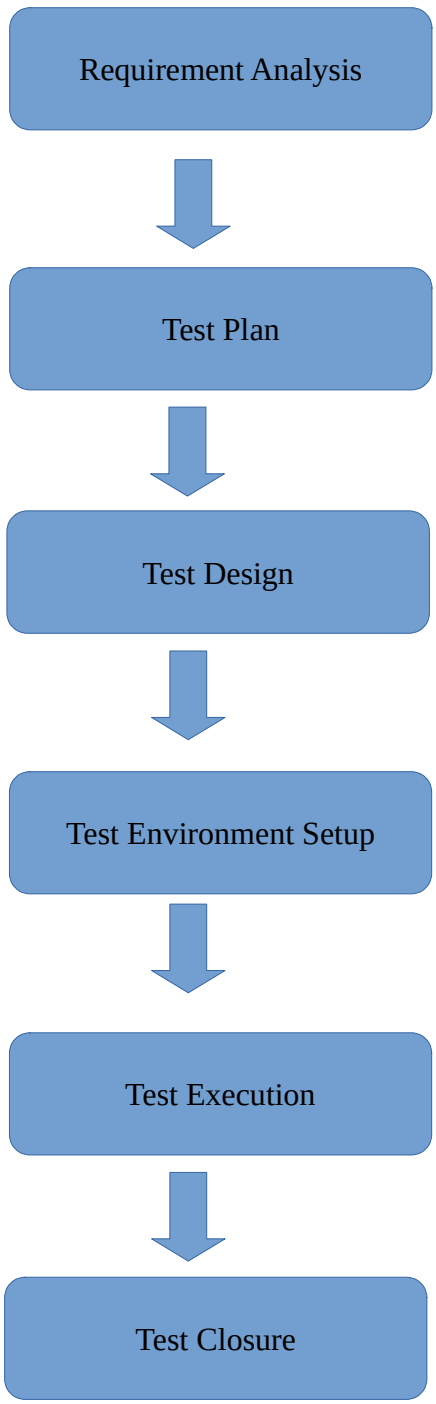
Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Maintenance:

it is the support given to the software after deployment.

5. Software Testing Life Cycle S.T.L.C.



Manual and Mobile Testing Metrics.

Requirement Analysis:

In requirement analysis test team studies/analyses the requirement document given by the client and verify whether it is testable or not and if the requirements aren't testable they further communicates with stakeholders of the client.

Test Planning:

The test team leads, managers along with the test engineers schedule the time periods, calculate the resources needed and cost estimations along with the selection of tools(if the testing is automation).

Test Design:

The testing team develop the test cases, test scripts, test data and which is further reviewed by the test team leads, managers. In this phase they also prepare the **Requirement Traceability Matrix R.T.M.**

where, **RTM** is document that helps to map and trace the requirements with respect to test cases. It helps to validate that all requirements are checked with test cases to make sure that no functionality in the requirement is missed during the software testing.

REQUIREMENTS TRACEABILITY MATRIX					
Project Name: Online Flight Booking Application					
Business Requirements Document BRD		Functional Requirements Document FSD			Test Case Document
Business Requirement ID#	Business Requirement / Business Use case	Functional Requirement ID#	Functional Requirement / Use Case	Priority	Test Case ID#
BR_1	Reservation Module	FR_1	One Way Ticket booking	High	TC#001 TC#002
		FR_2	Round Way Ticket		TC#003 TC#004
		FR_3	Multicity Ticket booking	High	TC#005 TC#006
BR_2	Payment Module	FR_4	By Credit Card	High	TC#007 TC#008
		FR_5	By Debit Card	High	TC#009
		FR_6	By Reward Points	Medium	TC#010 TC#011

Manual and Mobile Testing Metrics.

Test Environment Setup:

The testing team prepares all the software and hardware needs of the testing for the client/ architect team specified environment and tools here they also performs the smoke test.

Where, **Smoke Testing :** is the testing done by the experts to validate all the test cases to make sure that there is no **Show Stoppers**(Show stopper is a situation where the flow is blocked and user can't proceed further).

Test Execution:

here the test team starts executing the test cases based on the planned test cases to report pass/fail results. If the report is fail then the issue is reverted back to the developers.

Test Closure:

phase where the test coverage is 100% with minimum cost and time and also without having Critical and High defects ensuring maximum quality and efficiency of the product/software.

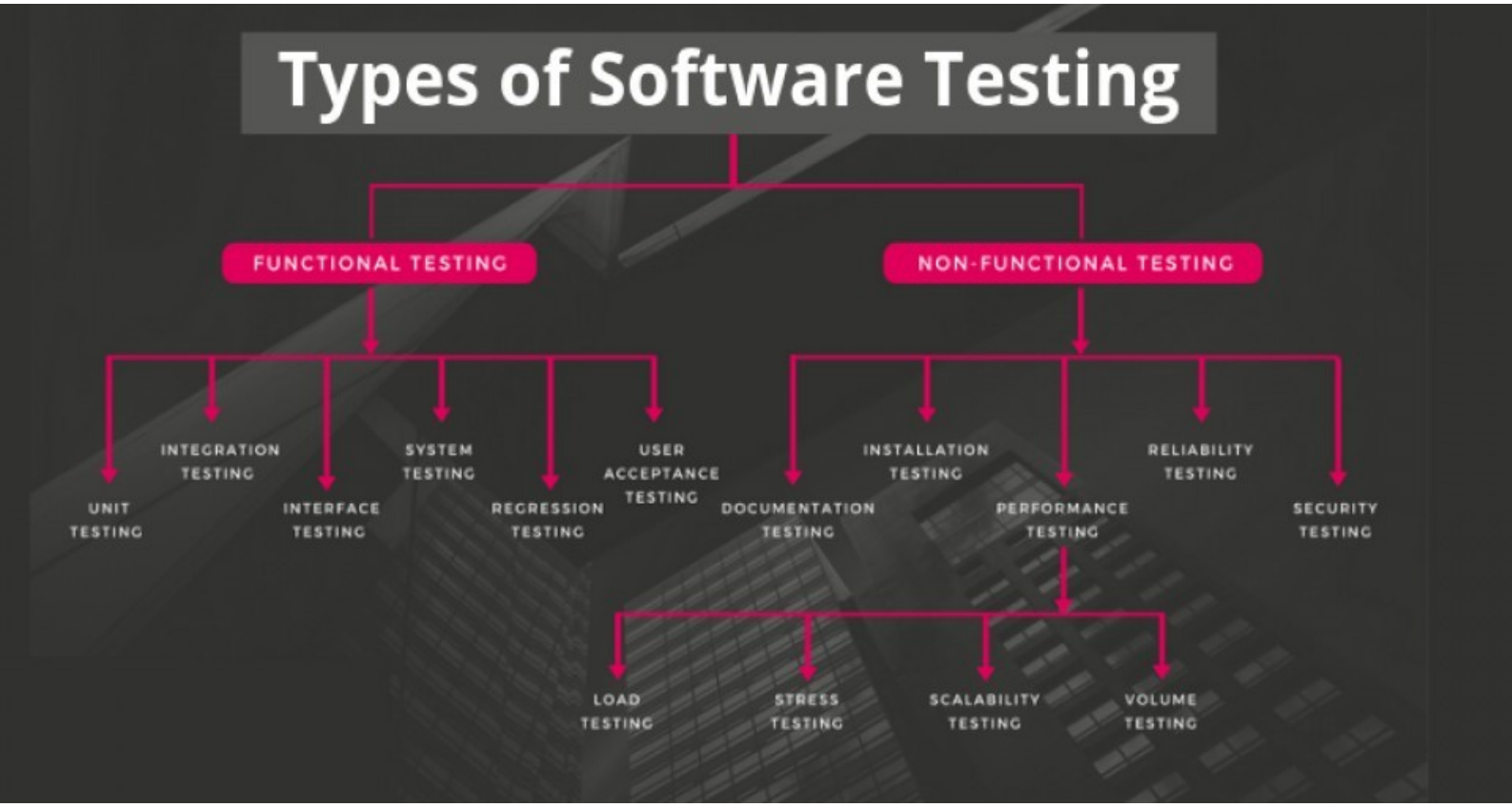
6. Types of Testing.

Testing is majorly of two types :

- **Functional Testing**
- **Non-Functional Testing.**

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.



Functional Testing:

1. Unit Testing:

This testing is done by developers, where the individual units/components are validate that they are performing as per the design.

Where, **unit** is the smallest part that can be tested in any software usually has 2 or more inputs and an output.

Benefits: cost-effective, debugging is easy.

Tools: Junit, Nunit, Jmockit, EMMA etc.,

2. Integration Testing: Here the smallest parts i.e., the units are integrated into one and are tested to study the interaction between the integrated units. It has different kinds of approaches like big bang, top-down, bottom-up, hybrid.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

3. System Testing:

Here the complete integrated software is tested to study that the software compliances with the specified requirements.

Usually system testing is done after the integration testing and before the acceptance testing.

4. Interface Testing:

Here the test engineers verify the communication between the different system involved in the software i.e, the server execution, redirections, error handling, security etc., are performing well if all the software and hardware components are given.

5. Regression Testing:

It is type of testing where the tester tests the pre-existing functionalities when ever there is a change or modification on the code base.

6. User Acceptance Testing:

It is the testing done before the software releasing into the market to ensure that all the functionalities are according to the client requirements and assess whether to ready release or not.

Usually it is done by customer support team, customers, end users etc.,

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Non-Functional Testing:

1. Documentation Testing:

It helps to estimate testing efforts required and test coverage which is done against the documents like test cases, test data, test scripts, RTM to make sure that they are efficient and are accurate accordingly.

2. Installation Testing:

It is quality assurance type of testing where the test engineers will do full/partial or upgrades to install and uninstall to study the behaviour of software accordingly.

3. Performance Testing:

It is of four types:

- **Load Testing –** type of performance testing to study the behaviour of system/software with increasing work load.
- **Stress Testing –** to study the behaviour of software with the load beyond or anticipated work load.
- **Endurance Testing –** to study the behaviour of the software when the significant work load is given continuously.
- **Spike Testing –** to study the behaviour of the software when the load is increased suddenly.

4. Reliability Testing: It is a type where the software is tested that it is fault free and is yielding the same result which is intended to be over period of time.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

5. Security Testing:

It is a type where the software is tested that it uncovers the vulnerabilities, threat, risks and prevent it from attacks. The main purpose of this testing is to find all the sensitive parts of the software and make them secure.

Other testing types:

Smoke Testing: is the testing done by the experts for checking the readiness of the build and also to validate all the test cases to make sure that there is no **Show Stoppers**.

Sanity Testing: is the type of testing done after every build with change in code base, addition of functionality, bug fixation etc., to make sure that the build is working as expected.

Smoke and Sanity are build verification type of testing.

Retesting: is the type where the testcases failed in earlier execution are ensured to pass test after the defects are fixed.

7. S.D.L.C. Models:

I. **Waterfall Model**

II. **Spiral Model**

III. **Iterative Model**

IV. **V-Model**

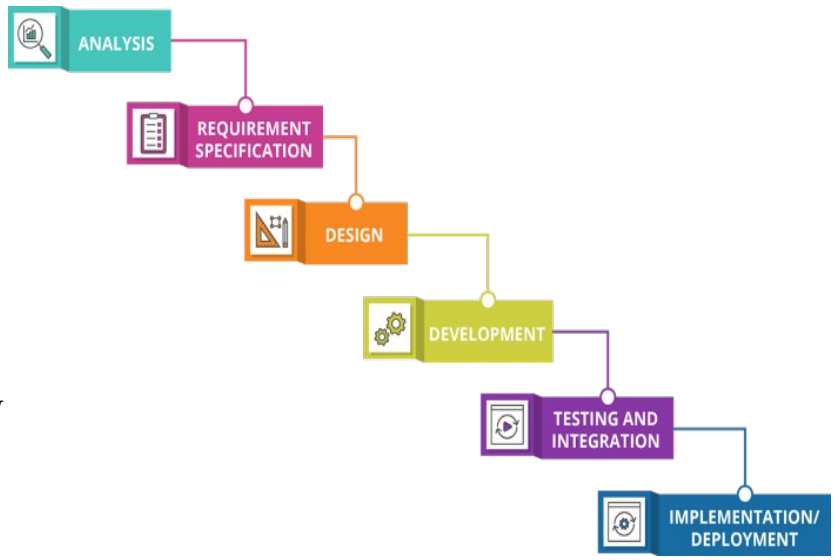
V. **Agile Model**

VI. **Prototype Model**

Manual and Mobile Testing Metrics.

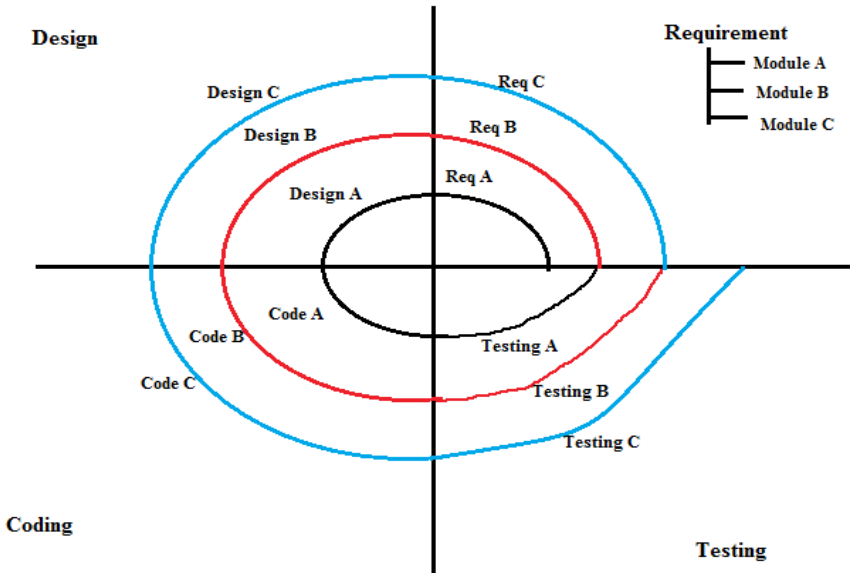
1. Waterfall Model:

- waterfall model is the traditional model of SDLC
- It is used when the requirements are stable.
- Mostly all the small and simple applications follow this development cycle.
- Short term application are suitable to follow.



2. Spiral Model:

- spiral model usually followed by the company when ever the customer is not sure about the requirements or the requirements are not stable.

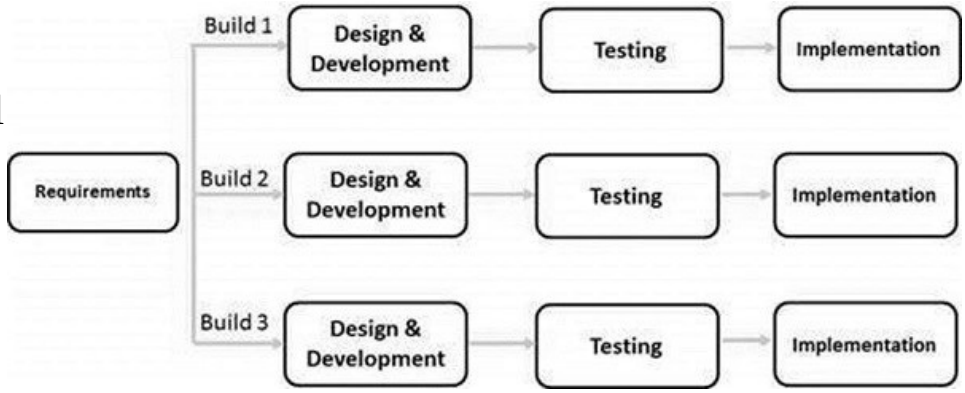


- Here the requirements are developed part by part.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

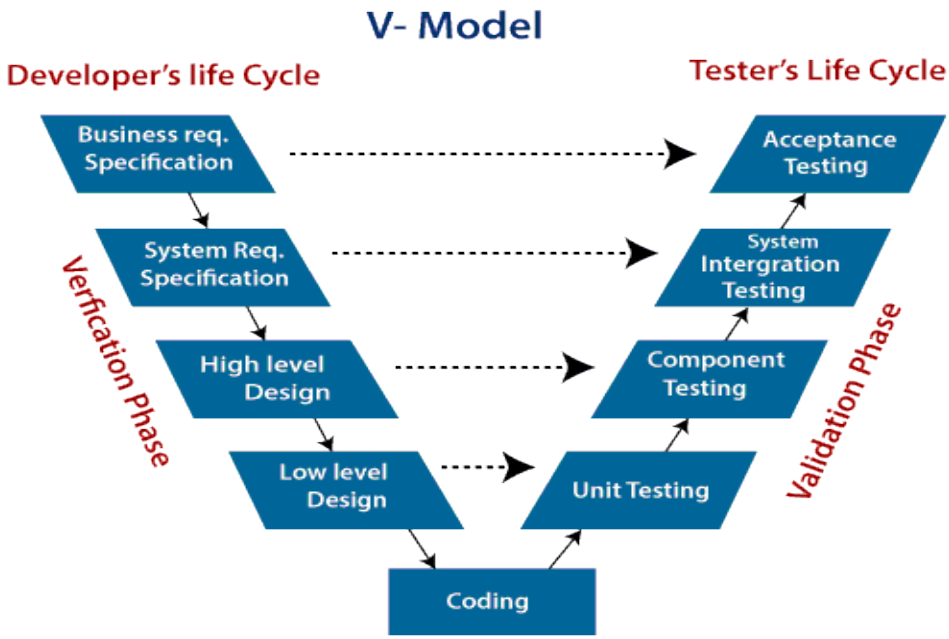
3. Iterative Model:



- In iterative model the system is developed through repeated cycles i.e., by adding feature by feature in step by step means.
- Parallel development of builds can be planned here.

4. V-Model:

- Verification and Validation Model popularly known as the V-Model.
- In this Model the development and the testing phases are done parallelly.



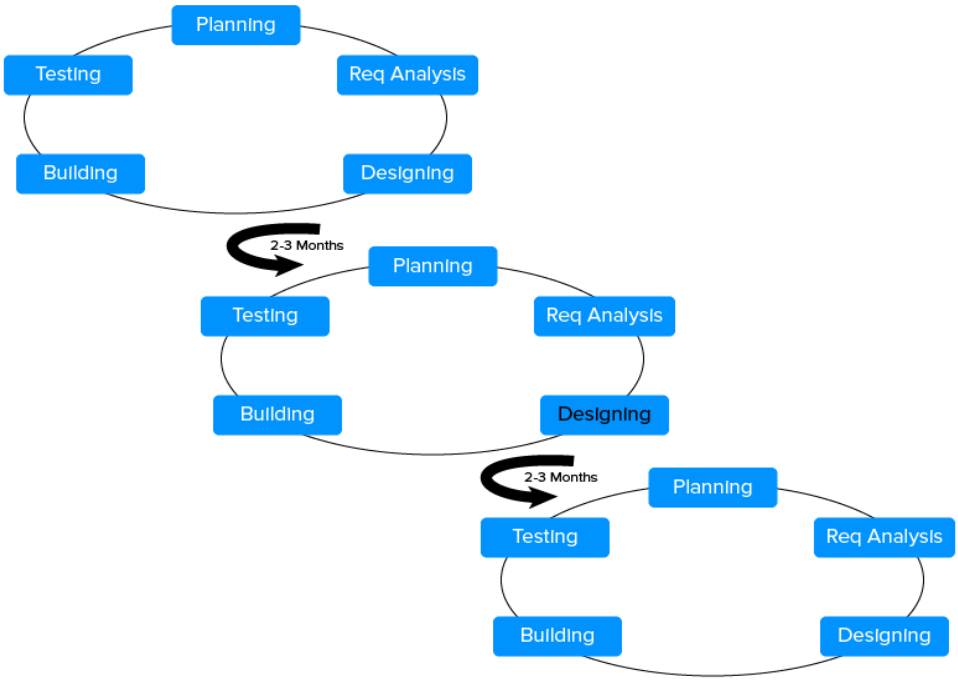
Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

5. Agile Model:

- In agile model the project is developed and tested in continuous iterations over a sprint.

  Where, **Sprint** is a time period set for a certain task/build/iteration

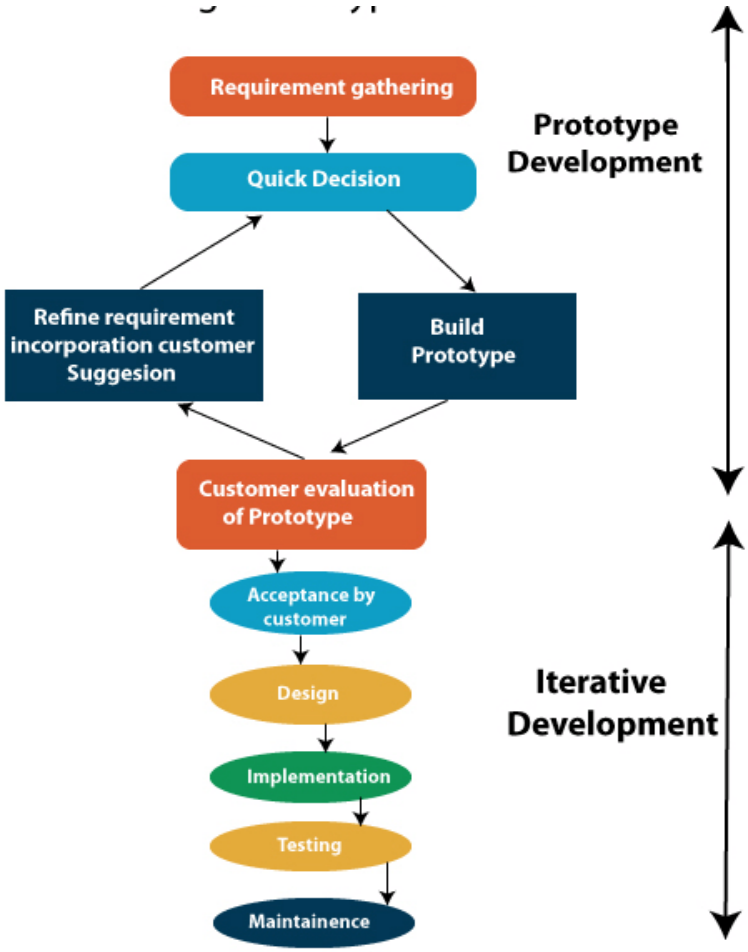


- Agile helps to demonstrate the deliverable at the end of every sprint so that the client/user can validate and suggest further modifications.
- **Scrum Methodology** is a kind of framework to manage the agile way of software development. Here the Scrum teams make sure that the project is moving forward with high level performance and also to communicates with stake holders to ensure they building the desired product.

Manual and Mobile Testing Metrics.

6. Prototype Model:

- This kind of development is used when the system is of short term and when the requirements are unstable.
- The development is done in 2 phases initially build a prototype and soon after the customer accepts without any code change or defects they are further pushed into iterative development phase.
- The continuous development of prototypes according to the requirements and functionalities and pushing them as one into implementation is known as the **Rapid Application Development (R.A.D.) Model.**



Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

8. Why Testing is needed?

1. **It Saves Money:** It will save money in long run, as software development is a long process by identifying bugs and defects at early stages saves money.
2. **Security:** It helps to study the sensitivities and vulnerabilities of the software so that the information is safe and secured.
3. **Product Quality:** It helps to ensure that the product is good, effective and efficient.
4. **Customer Satisfaction:** It helps to ensure best user experience which helps to gain the user trust and so business to reach heights.

9. Defect and defect management.

Defect: is a deviation between the expected result and the actual result with respect to the customer requirement.

Bug: mistake or mismatch in system identified by tester during the phase of testing.

Bug Report is a document forwarded to developer having the following information.

1. Defect_ID	2. Defect Description	3. Version	4. Steps
5. Date Raised	6. Reference	7.Detected By	8. Status
9. Fixed by	10. Date Closed	11. Severity	12. Priority.

Failure: Issues faced by the customer after the system deployed.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Fault: A condition that causes the software to fail, fault results in performance deviations in functionalities.

Error: A error is the mistake done by the developer during the code base development. Error can be easily identified and rectified.

Defect Management: Defect management is done by rating the defect in 2 ways

1. **Severity:** It is the extent to which the defect can create an impact on the software. Severity can rated in 4 types

- **Critical Severity:** when the defect creates a high business impact in the work flow of the system with 0 work around i.e., the customer can't proceed further. Critical Severity defect is a shop stopper.
- **High Severity:** when the defect creates a high business impact in the work flow of the system but having 1 or 2 work around i.e., the customer can proceed further by trying an alternate way.
- **Medium Severity:** when the defect create a low business impact in the work flow of the system but having many work around i.e., layout mismatches, toggle raising unnecessarily etc.,
- **Low Severity:** when the defect has no business impact but having a cosmetic defect i.e., the colour of background not compensating the text to understand etc.,

2. **Priority:** It indicates that how soon the defect should be resolved.

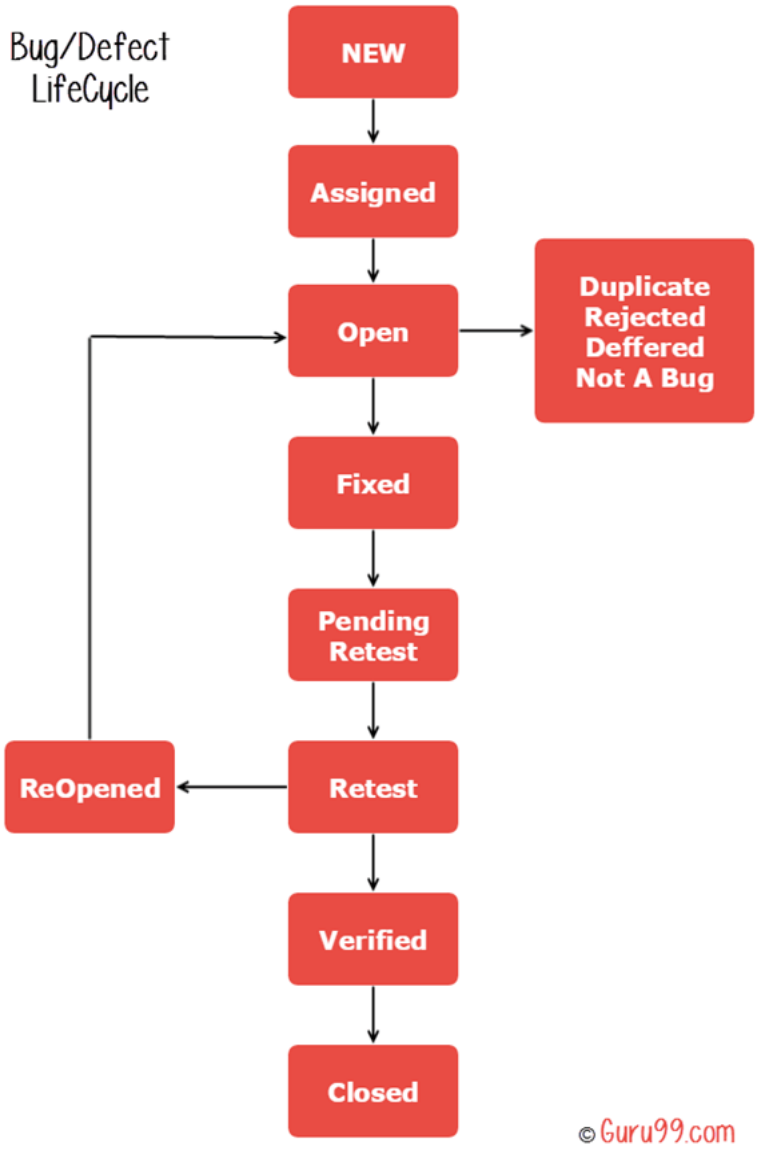
Priority can be rated in 3 ways

- **High Priority :** All critical and high severity defects are High Priority defects
- **Medium Priority and Low Priority** varies between medium and low severity according to the conditions.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Defect Life Cycle:



- **New:** the very first time the defect found by test engineer.
- **Assigned:** the defect found by the test engineer is redirected to the developer for defect fixation.
- **Open:** Developer analyses and work on fixing it.
- From OPEN if the bug inappropriate developer will mark it as follows

  Rejected- if the developer feels that the bug is not genuine developer states the bug as rejected

  Duplicate- if the bug is repeated twice or the bugs if mentioning the same concept then it is stated as Duplicate

  Deferred- if the bug is expected to fix in the next release then it is stated as deferred.

Documented by: **Pranay Koppada**

- After Bug Fixation it is directed to **Test/Retest** phase where the test engineer tests and assess the working efficiency.
- After the Test/Retest if the issue is still existing the it will be **Re-Opened** and again developer study analyse and if it is appropriate fixes it if not mark it as rejected, deferred, duplicate.
- In **Test/Retest** phase if the bug is fixed it is directed again it ia **Verified** by testing repeatedly to ensure the work flow.
- If verification is done perfectly and there is no bug/defect found then the Bug is **Closed** and feature/module/product is approved.

10. JIRA.

JIRA is a product of **Atlassian** which is designed for

- **Project Management**
- **Issue-Tracking**
- **Bug-Tracking for agile software and business teams**

Why JIRA

- JIRA has everything to manage project and teams
- JIRA allows you to customize according to needs
- JIRA has countless add-ons to further enhance the functionality
- JIRA has many features that helps to make our work easier

Products of JIRA

1. **JIRA Core -** Helps to set up business workflows.
2. **JIRA Software -** Helps to setup business workflow along with agile way to manage and integrate software development

Manual and Mobile Testing Metrics.

3. **JIRA Service Desk -** Helps to manage interaction between the clients to faster and efficient responses
4. **JIRA Ops -** Helps to manage incidents used to centralize the alerts to notify right people at the right time.

*** What is QA and QC**

QA stands for Quality Assurance, which is a process or set of processes used to measure and assure the quality if the product ensuring defect prevention.

It is Process Oriented. It is Managerial Tool.

It is Verification Process.

QC stands for Quality Control, which is the process of ensuring products and services meets the customer requirements and it mainly focuses on defect identification.

It is Product oriented. It is Corrective Tool.

It is Validation Process.

* **Monkey Testing:** It is a type where the test engineers provide random inputs to check the system behaviour and also to check the possibilities of system to crash.

*** Exploratory Testing:** It is a type where the test engineers tends to learn, design and execute simultaneously i.e., it do not have a proper structure to follow test engineers identifies the functionalities and design and execute.

***Ad-hoc Testing:** It is a type of testing performed without planning and documentation and the objective is to divide the system randomly into subparts and check their functionality.

Manual and Mobile Testing Metrics.

Mobile Testing

1. Mobile Operating Systems.

Android and IOS(Iphone Operating System) are the most popular mobile operating systems in today's market. Where Android is the Google designed OS and IOS is the apple designed OS. Android is open system where user can customize the devices but IOS is a closed system with layered architecture with strict device permissions.

The initial release of Android (android version 1.0)is 23 September 2008 and IOS(Iphone OS 1) is 29 July 2007

current version of Android is Android 10(September 2019) and IOS is version is 13.7(September 2020)

version list:

Android:

Code name	Version numbers
No codename	1.0
No codename	1.1
Cupcake	1.5
Donut	1.6
Eclair	2.0 - 2.1
Froyo	2.2 - 2.2.3
Gingerbread	2.3 - 2.3.7
Honeycomb	3.0 - 3.2.6
Ice Cream Sandwich	4.0 - 4.0.4
Jelly Bean	4.1 - 4.3.1
KitKat	4.4 - 4.4.4
Lollipop	5.0 - 5.1.1
Marshmallow	6.0 - 6.0.1
Nougat	7.0
Nougat	7.1.0 - 7.1.2
Oreo	8.0
Oreo	8.1
Pie	9.0
Android 10	10.0

IOS:

Version	Build
3.1.3	7E18
4.2.1	8C148
5.1.1	9B206
6.1.6	10B500
7.1.2	11D257
9.3.5	13G36
9.3.6	13G37
10.3.3	14G60
10.3.4	14G61
12.4.8	16G201
13.7	17H35
14.0 beta 7	18A5369b

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

2. Mobile Applications and their types.

Mobile Application: a mobile application is a software application developed for use on small, wireless computing devices such as smartphones and tablets. Mobile applications can come preloaded or can be downloaded from the device app store. The popular smartphones that support mobile applications are Android, IOS etc.,

Android applications are enclosed with **.apk(Android package)** extension and IOS applications are enclosed with **.ipa(Iphone app store package)** extension.

Different aspects comparing IOS and Android:

- **Type of Software Source – android** is an open source and that enables the customisation of applications easy using the android software development kit where IOS is not an open source and the applications developed are needed to be tested and released only by apple corporation
- **Type of Hardware – android** is supported in many of the devices and IOS is supported in the devices which are made by apple.
- **App Store Ecosystem – android** play store is the official area to engage with android applications but also the packages are available in websites external drives etc., where as , applications of IOS are duly restricted to its app store and access any other ways is denied.

Mobile applications are generally of three types:

1. **Native Applications**
2. **Web Applications**
3. **Hybrid Applications**

Manual and Mobile Testing Metrics.

1. Native Mobile Applications:

A native application is a downloadable mobile application which is installed and run directly on the device. Native mobile application is developed /implemented for a specific device operating system (android, IOS) and are always in the device's coding language. For example android applications are coded on Java**,** IOS applications are coded on **objective-C or Swift,** Windows are coded on **.Net**.

Generally they are developed specifically for one platform and can take full advantages of all the device features.

Examples: Iphone default dailer, browser, camera etc.,

2. Web Applications:

Mobile applications which are not downloadable from the device traditional app stores such as the Apple App Store or Android Play Store. These applications comes with the website itself which are made to fit any device, coding is done in any language and it can run on all platforms.

Example: m.facebook.com, m.redbus.com etc.,

3. Hybrid Applications:

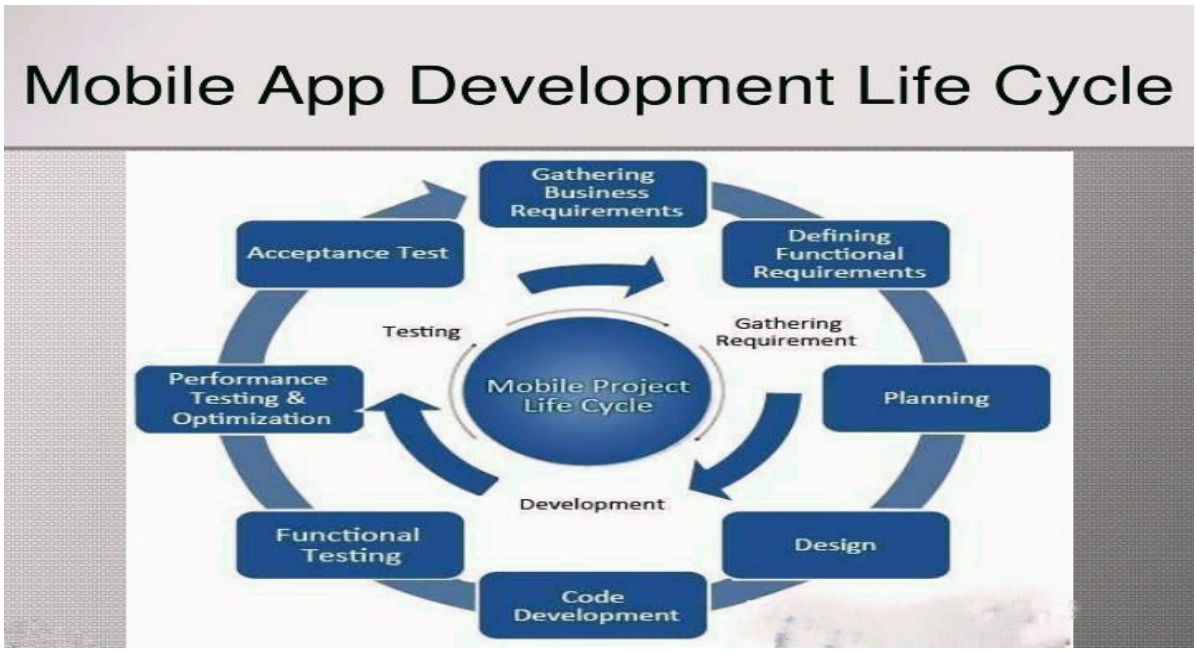
Mobile applications which are developed using "Open Web" technologies such as HTML, CSS, JavaScript etc., and then they are packaged up into fully native applications. They will also be available in the device application store and can take advantages of device features and also can run on all platforms.

Example: instagram, facebook, gmail etc.,

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

3. Mobile Application Development Life Cycle:



4. Mobile Application Testing and Types

Mobile testing is an important aspect of mobile application's development stage. It differs significantly from traditional software testing. Mobile applications need to be tested on a variety of software and hardware platforms and under different network connectivity conditions. It is important that testing is done thoroughly so that there are no bugs in app.

Mobile Testing Types:

- **Functional Testing**
- **Performance Testing**
- **Memory Leakage Testing**
- **Security Testing**
- **Usability Testing**
- **Installation Testing**
- **Compatibility Testing**
- **Interrupt Testing**
- **Recoverability Testing**

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

1. **Functional Testing:**

The Functional Testing of Mobile Application is a process of testing functionalities of mobile applications like user interactions as well as testing the transactions that users might perform.

The main purpose of mobile application functional testing is to ensure the quality, meeting the specified requirements, reducing the risk or errors and to ensure customer satisfaction

Test Scenarios:

- To validate whether all the required mandatory fields are working as required.
- To validate that the mandatory fields are displayed in the screen in a distinctive way than the non-mandatory fields.
- To validate that the device is able to perform required multitasking requirements whenever it is necessary to do so.
- To validate that the application allows necessary social network options such as sharing, posting and navigation etc.
- To validate that the application supports any payment gateway transaction such as Visa, MasterCard, PayPal etc. as required by the application.

2. Performance Testing: Verification of the response time of the application under various load conditions. This is a type of Non-Functional Testing.

It is used to:

- To determine whether the current network coverage is able to support the application at peak, average and minimum user levels

- To validate whether the response time of the application is as per as the requirements.
- To validate that the battery consumption, memory leaks, resources like GPS, Camera performance is well within required guidelines.
- To evaluate whether the battery life can support the application to perform under projected load volumes.
- To validate the network performance while moving around with the device.

3. Memory Leakage Testing:

Here the test engineers make sure that the application is utilising minimal memory area on different devices to ensure that the application is not closed by OS environment as it will shut the application which is utilising large amount of memory. By this testing they ensure that the application is not effecting the device performance.

4. Security Testing:

This type testing is to make sure that they uncover all the sensitive and vulnerable parts of the application are secured from malicious attacks and user data is not accessed by attackers.

It is used mainly to:

- To validate whether an application is not permitting an attacker to access sensitive content or functionality without proper authentication.
- To identify the dynamic dependencies and take measures to prevent any attacker for accessing these vulnerabilities.

- To prevent from SQL injection related attacks.
- To analyse the data storage and data validation requirements
- To protect against malicious client side injections.
- To protect against malicious runtime injections.

5. Usability Testing:

It is performed to make sure that the application is quick and easy step wise flow with less complications than difficult application with many procedures to follow.

It is ensure that the application is easy-to-use, intuitive in nature so that the end user find no difficulty while accessing/using it.

It is used to:

- To ensure that the buttons should have the required size and be suitable to big fingers.
- To ensure that the buttons are placed in the same section of the screen to avoid confusion to the end users.
- To ensure that the icons are natural and consistent with the application.
- To ensure that the buttons, which have the same function should also have the same colour.
- To ensure that the text is kept simple and clear to be visible to the users.

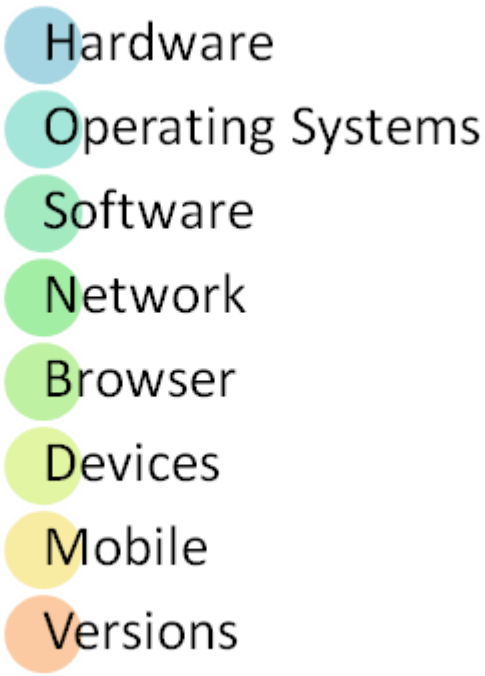
Manual and Mobile Testing Metrics.

6. Installation Testing:

This the testing where the test engineer make sure that the application is installing/uninstalling/updating properly and they are not affecting the navigation or the flow of other applications.

7. Compatibility Testing:

Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, network environments or devices.



It is a non-functional testing.

Compatibility testing is done on following aspects as shown in fig.

Compatibility Testing is of two types:

1. **Backward Compatibility Testing:** is to verify the behaviour of the developed hardware/software with the **older versions** of the hardware/software.
2. **Forward Compatibility Testing:** is to verify the behaviour of the developed hardware/software with the **newer versions** of the hardware/software.

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

8. Interrupt Testing:

This Testing that deals with- how an application reacts to interruption and resumes to its previous state.

The interruption are mainly as follows:

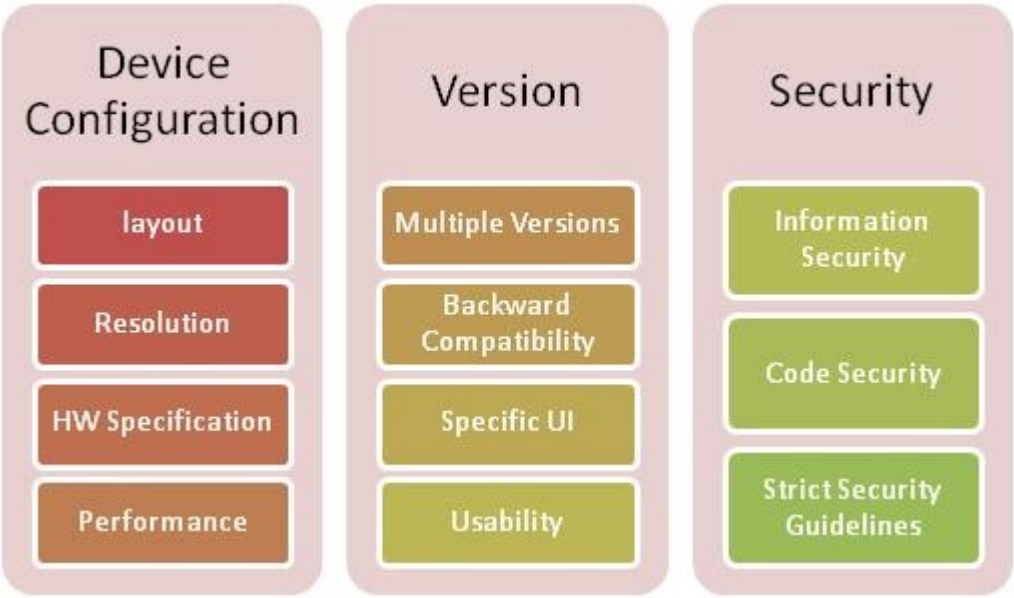
- **Battery low**
- **Battery full- when charging**
- **Incoming phone call**
- **Incoming SMS**
- **Incoming Alert from another mobile application**
- **Plugged in for charging**
- **Plugged out from charging**
- **Device shut off**
- **Application Update reminders**
- **Alarm**
- **Network connection loss**
- **Network connection restoration**

9. Recoverability Testing:

In this testing mainly checks how an app works on transactions when there is an app failure, and also analyses whether the application is able to recover the data after suspended connections that lead to failure.

Manual and Mobile Testing Metrics.

5. Challenges that are involved in Mobile Testing



a) **Device Configuration**:

1. Layout:

The layout of each device could be different. Android phones are available in different layout and sizes. This poses a challenge for the testers to test the application on every possible device. On other hand, apple has limited device portfolio which can be tested easily.

2.Resolution:

Different layout and sizes, results in resolution to vary from device to device. Chances are there that android app may not function as intended on all the devices. This means android app needs to be optimized for each device. This increases the number of scenarios to be tested thus increasing the testing efforts.

Manual and Mobile Testing Metrics.

3. Hardware Specifications:

Hardware specifications also need to be taken into consideration during. The processor speed and memory plays a significant role for certain apps to function smoothly. An application designed for high end phones might not work with low end phones. Hence while testing these points' needs to be remembered.

4. Performance:

With so many device variants available in market it should be ensured that code is written in such a way that the resource utilization is balanced.

b) OS versions

1. Multiple Versions:

Google does not encourage OEMs to use the latest version. This means that newly released phones might be using an older version of android. Unlike Apple devices which always come with the latest iOS, android devices could be running on older version such as ice-cream sandwich. The availability of multiple versions of the android and the resulting OS fragmentation causes testing apps on various versions of android to be challenging.

2. Backward Compatibility:

In Android mobile application testing, the tester need to test the Android API's with the older versions, ensuring that the applications run as desired.

3. Specific UI:

Since android is open source, Original Equipment Manufacturer (OEMs) are free to build their own UI. Hence UI such as TouchWiz by Samsung, Sense by HTC, Xperia by Sony, and ZenUI by Asus varies in look and feel. So application needs to be tested for specific UI.

Manual and Mobile Testing Metrics.

4. Usability:

It is essential that usability experience remains same over different devices and the versions. Thus multiple use cases needs to be developed as per the OS version.

c) Security:

1. Information Security:

with multiple devices and Operating Systems, it becomes a tedious task to ensure that application is not vulnerable to malicious attacks. A security issue present in the previous version of the android has to be addressed through security update in the app to avoid any information stealing.

2. Code Security Testing:

During Application development, developers can install the necessary APK files on device using Google Play store. There are tools available in the market that allows users to have unauthorized access to source code of the android APK files. Therefore additional security testing needs to be done for android OS to ensure that the application code is secured.

Also android is a target for hackers due to its large user base. Hence comprehensive testing is required for an app.

3. Strict Publishing Guidelines:

Google does not check the app before they are made available in the Play Store. Google only scans the Play store for malicious content. Hence it is easy for malicious apps to remain in the store till somebody reports about it. So tester needs to ensure that there are no security loopholes in the application.

Manual and Mobile Testing Metrics.

6. Real Devices, Simulator, Emulator.

Real Devices Testing: Testing on a real device allows you to run your mobile applications and checks its functionality. Real devices testing assures that your application will work smoothly on customer handset.

Mobile devices are many in number in terms of brands, versions, screen resolutions, layout etc., and so the real device testing is a challenge for mobile test engineers. To overcome this challenges face by the test engineers **Simulators and Emulators** are used.

Simulator: simulator is partial re-implementation of original software that replicates the functionalities of a real device virtually. It is written in high level to imply the internal behaviour of firmware.

Emulator: Emulator is complete re-implementation of original software of that replicates the functionalities of a real device virtually. It is written in machine level to imply mobile external features process management, transactions.

7. Mobile Testing Tools

- **APPIUM**
- **XCODE**
- **ADB**

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

1. APPIUM:

APPIUM is a freely distributed open source mobile application testing framework. Appium allows native, web and hybrid mobile application testing and support automation test on real devices as well as emulator or simulator both. It has no dependency on device OS so it will work for both andriod and IOS. It supports all languages that have selenium client libraries like Java, objective-c, Js, node js, PHP etc.,

2. Xcode:

Xcode is a set of software package(a set of interrelated programs that work together) used to write software for MacOS, IOS devices etc., Xcode is a IDE with editors, compliers and other software tools that work together to write a software, compile it, load it on a device, debug it.

3. ADB(android debugging bridge):

Android has an integrated development kit which enables the user to create , modify, compile, debug android applications. ADB is a command line tool. It is used to bridge communication between an emulator instance (Android device) and background running daemon process (server). ADB is connected to device generally by USB but it also supports WiFi connections**.**

Installing and Setting up ADB connections:

1. Download the Windows zip from Google.

2. Extract it somewhere - for example, %USERPROFILE%\adb-fastboot

Documented by: **Pranay Koppada**

3. On Windows 10:

- Open the Start menu, and type “advanced system settings”
- Select “View advanced system settings”
- Click on the Advanced tab
- Open the “Environment Variables” window
- Select the Path variable under “System Variables” and click the “Edit” button
- Click the “Edit Text” button
- Append ;%USERPROFILE%\adb-fastboot\platform-tools to the end of the existing Path definition (the semi-colon separates each path entry).

4.Install the universal adb driver, and reboot.

Setting up ADB:

To use adb with your device, you’ll need to enable developer options and USB debugging:

1. Open Settings, and select “About”.
2. Tap on “Build number” seven times.
3. Go back, and select “Developer options”.
4. Scroll down, and check the “Android debugging” or “USB debugging” entry under “Debugging”.
5. Plug your device into your computer.
6. On the computer, open up a terminal/command prompt and type adb devices.
7. A dialog should show on your device, asking you to allow usb debugging. Check “always allow”, and choose “OK”.

Congratulations! adb is now ready to use with your device.

Manual and Mobile Testing Metrics.

ADB Commands:

1. Basics:

adb devices (lists connected devices)
adb root (restarts adbd with root permissions)
adb start-server (starts the adb server)
adb kill-server (kills the adb server)
adb reboot (reboots the device)
adb devices -l (list of devices by product/model)
adb shell (starts the backround terminal)
exit (exits the background terminal)
adb help (list all commands)
adb -s <deviceName> <command> (redirect command to specific device)
adb –d <command> (directs command to only attached USB device)
adb –e <command> (directs command to only attached emulator)

2. Package Installation:

adb shell install <apk> (install app)
adb shell install <path> (install app from phone path)
adb shell install -r <path> (install app from phone path)
adb shell uninstall <name> (remove the app)

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

3. Paths:

/data/data/<package>/databases (app databases)
/data/data/<package>/shared_prefs/ (shared preferences)
/data/app (apk installed by user)
/system/app (pre-installed APK files)
/mmt/asec (encrypted apps) (App2SD)
/mmt/emmc (internal SD Card)
/mmt/adcard (external/Internal SD Card)
/mmt/adcard/external_sd (external SD Card)

adb shell ls (list directory contents)
adb shell ls -s (print size of each file)
adb shell ls -R (list subdirectories recursively)

4. File Operations:

adb push <local> <remote> (copy file/dir to device)
adb pull <remote> <local> (copy file/dir from device)
run-as <package> cat <file> (access the private package files)

5. Phone Info:

adb get-state (print device state)
adb get-serialno (get the serial number)
adb shell dumpsys iphonesybinfo (get the IMEI)
adb shell netstat (list TCP connectivity)
adb shell pwd (print current working directory)

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

adb shell dumpsys battery (battery status)
adb shell pm list features (list phone features)
adb shell service list (list all services)
adb shell dumpsys activity <package>/<activity> (activity info)
adb shell ps (print process status)
adb shell wm size (displays the current screen resolution)
dumpsys window windows | grep -E 'mCurrentFocus|mFocusedApp' (print current app's opened activity)

6. Package Info:

adb shell list packages (list package names)
adb shell list packages -r (list package name + path to apks)
adb shell list packages -3 (list third party package names)
adb shell list packages -s (list only system packages)
adb shell list packages -u (list package names + uninstalled)
adb shell dumpsys package packages (list info on all apps)
adb shell dump <name> (list info on one package)
adb shell path <package> (path to the apk file)

7. Configure Settings Commands:

adb shell dumpsys battery set level <n> (change the level from 0 to 100)
adb shell dumpsys battery set status<n> (change the level to unknown, charging, discharging, not charging or full)
adb shell dumpsys battery reset (reset the battery)
adb shell dumpsys battery set usb <n> (change the status of USB connection. ON or OFF)
adb shell wm size WxH (sets the resolution to WxH)

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

8. Device Related Commands:

adb reboot-recovery (reboot device into recovery mode)
adb reboot fastboot (reboot device into recovery mode)
adb shell screencap -p "/path/to/screenshot.png" (capture screenshot)
adb shell screenrecord "/path/to/record.mp4" (record device screen)
adb backup -apk -all -f backup.ab (backup settings and apps)
adb backup -apk -shared -all -f backup.ab (backup settings, apps and shared storage)
adb backup -apk -nosystem -all -f backup.ab (backup only non-system apps)
adb restore backup.ab (restore a previous backup)

9. Logs

adb logcat [options] [filter] [filter] (view device log)
adb bugreport (print bug reports)

10. Permissions

adb shell permissions groups (list permission groups definitions)
adb shell list permissions -g -r (list permissions details)

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

8. Miscellanous:

***Crash:** a crash is an unexpected termination of an application and throws an error message to the user (“ application stopped unexpectedly”). If an application gets crashed it doesn’t needed it to be running in foreground any active process related to that application can cause a crash.

Crash log: crash log is the report which contains all runtime logs of an operating system.

Retrieving Crash log using ADB: adb logcat AndriodRuntime:E*:S

ANR(application not responding): it is an unexpected termination of an application which is running in the process memory and an error message is thrown saying “application not responding”. If ANR is thrown the application will be still in the process memory and can be able to recover when the exception is cleared.

***Logcat:** logcat is an internal command line functionality of the abd tool in the android SDK that dumps all log messages of a file, application, device accordingly. Generally a **log** is a class that has reports of errors, warnings, system info, debugging info etc.,

command: adb logcat

***Bug-report: bug-report** contains the devices logs, stack traces(error report created whenever there is an application crash) and other diagnostic information which help you find and fix bugs in your application.

There are 2 ways to retrieve a bug-report

Manual and Mobile Testing Metrics.

- Through the device using the developer options and saving a bug report
- Through adb by connecting the device and retrieving the report using the command

  adb bugreport E:\Reports\MyBugReports

LINUX/UNIX Commands

File Commands		
1.	ls	Directory listing
2.	ls -al	Formatted listing with hidden files
3.	ls -lt	Sorting the Formatted listing by time modification
4.	cd dir	Change directory to dir
5.	cd	Change to home directory
6.	pwd	Show current working directory
7.	mkdir dir	Creating a directory dir
8.	cat >file	Places the standard input into the file
9.	more file	Output the contents of the file
10.	head file	Output the first 10 lines of the file
11.	tail file	Output the last 10 lines of the file
12.	tail -f file	Output the contents of file as it grows,starting with the last 10 lines
13.	touch file	Create or update file
14.	rm file	Deleting the file
15.	rm -r dir	Deleting the directory
16.	rm -f file	Force to remove the file
17.	rm -rf dir	Force to remove the directory dir
18.	cp file1 file2	Copy the contents of file1 to file2
19.	cp -r dir1 dir2	Copy dir1 to dir2;create dir2 if not present
20.	mv file1 file2	Rename or move file1 to file2,if file2 is an existing directory
21.	ln -s file link	Create symbolic link link to file

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Process management		
1.	ps	To display the currently working processes
2.	top	Display all running process
3.	kill pid	Kill the process with given pid
4.	killall proc	Kill all the process named proc
5.	pkill pattern	Will kill all processes matching the pattern
6.	bg	List stopped or background jobs,resume a stopped job in the background
7.	fg	Brings the most recent job to foreground
8.	fg n	Brings job n to the foreground

Searching		
1.	grep pattern file	Search for pattern in file
2.	grep -r pattern dir	Search recursively for pattern in dir
3.	command | grep pattern	Search pattern in the output of a command
4.	locate file	Find all instances of file
5.	find . -name filename	Searches in the current directory (represented by a period) and below it, for files and directories with names starting with filename
6.	pgrep pattern	Searches for all the named processes , that matches with the pattern and, by default, returns their ID

File permission		
1.	chmod octal file	Change the permission of file to octal,which can be found separately for user,group,world by adding, • 4-read(r) • 2-write(w) • 1-execute(x)

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

System Info		
1.	date	Show the current date and time
2.	cal	Show this month's calender
3.	uptime	Show current uptime
4.	w	Display who is on line
5.	whoami	Who you are logged in as
6.	finger user	Display information about user
7.	uname -a	Show kernel information
8.	cat /proc/cpuinfo	Cpu information
9.	cat proc/meminfo	Memory information
10.	man command	Show the manual for command
11.	df	Show the disk usage
12.	du	Show directory space usage
13.	free	Show memory and swap usage
14.	whereis app	Show possible locations of app
15.	which app	Show which applications will be run by default

Compression		
1.	tar cf file.tar file	Create tar named file.tar containing file
2.	tar xf file.tar	Extract the files from file.tar
3.	tar czf file.tar.gz files	Create a tar with Gzip compression
4.	tar xzf file.tar.gz	Extract a tar using Gzip
5.	tar cjf file.tar.bz2	Create tar with Bzip2 compression
6.	tar xjf file.tar.bz2	Extract a tar using Bzip2
7.	gzip file	Compresses file and renames it to file.gz
8.	gzip -d file.gz	Decompresses file.gz back to file

Documented by: **Pranay Koppada**

Manual and Mobile Testing Metrics.

Network		
1.	ping host	Ping host and output results
2.	whois domain	Get whois information for domains
3.	dig domain	Get DNS information for domain
4.	dig -x host	Reverse lookup host
5.	wget file	Download file
6.	wget -c file	Continue a stopped download

Shortcuts		
1.	ctrl+c	Halts the current command
2.	ctrl+z	Stops the current command, resume with fg in the foreground or bg in the background
3.	ctrl+d	Logout the current session, similar to exit
4.	ctrl+w	Erases one word in the current line
5.	ctrl+u	Erases the whole line
6.	ctrl+r	Type to bring up a recent command
7.	!!	Repeats the last command
8.	exit	Logout the current session

Documented by: **Pranay Koppada**