Manual Testing Preparation

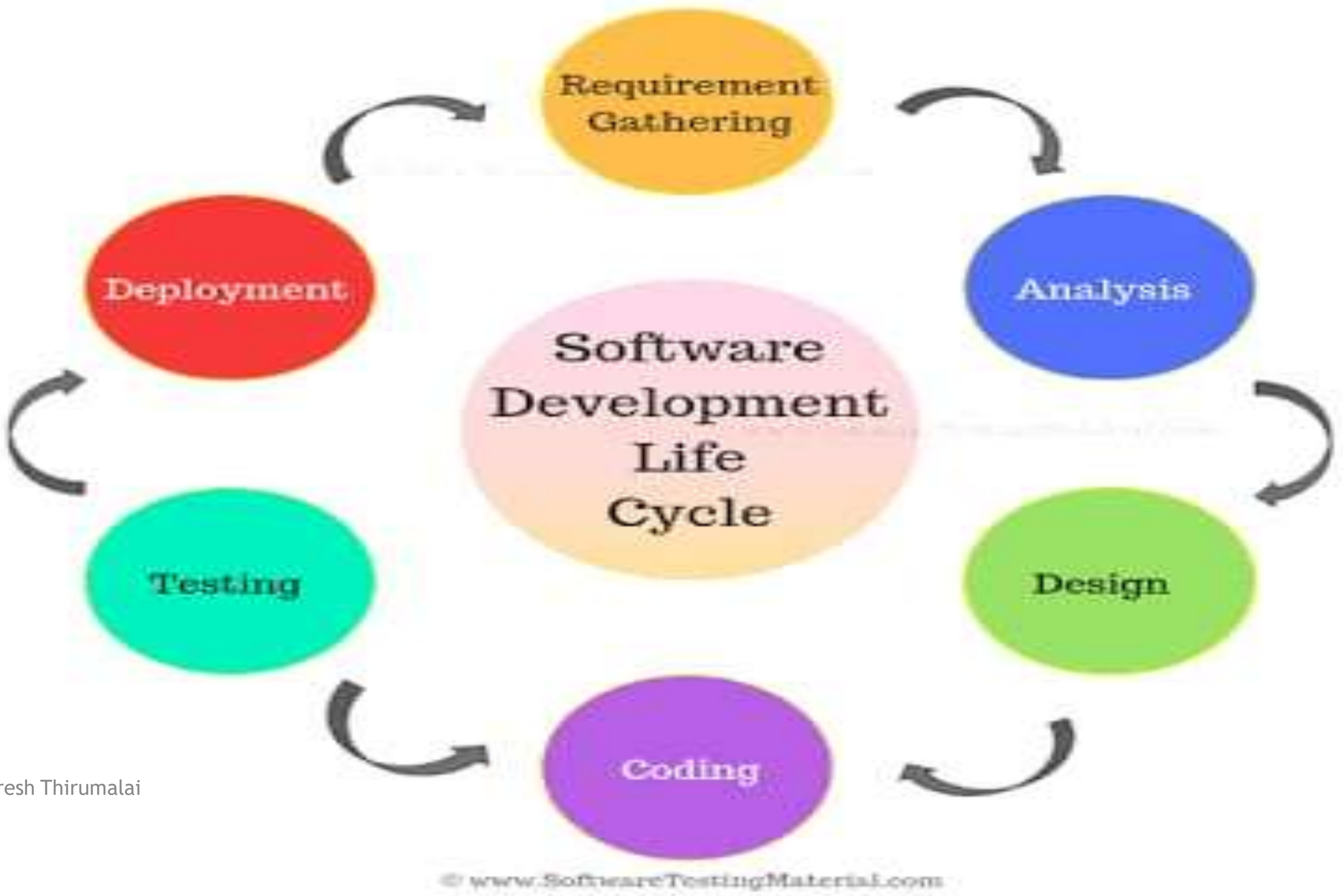
By **Suresh Thirumalai**

Index

Mr. Suresh Thirumalai

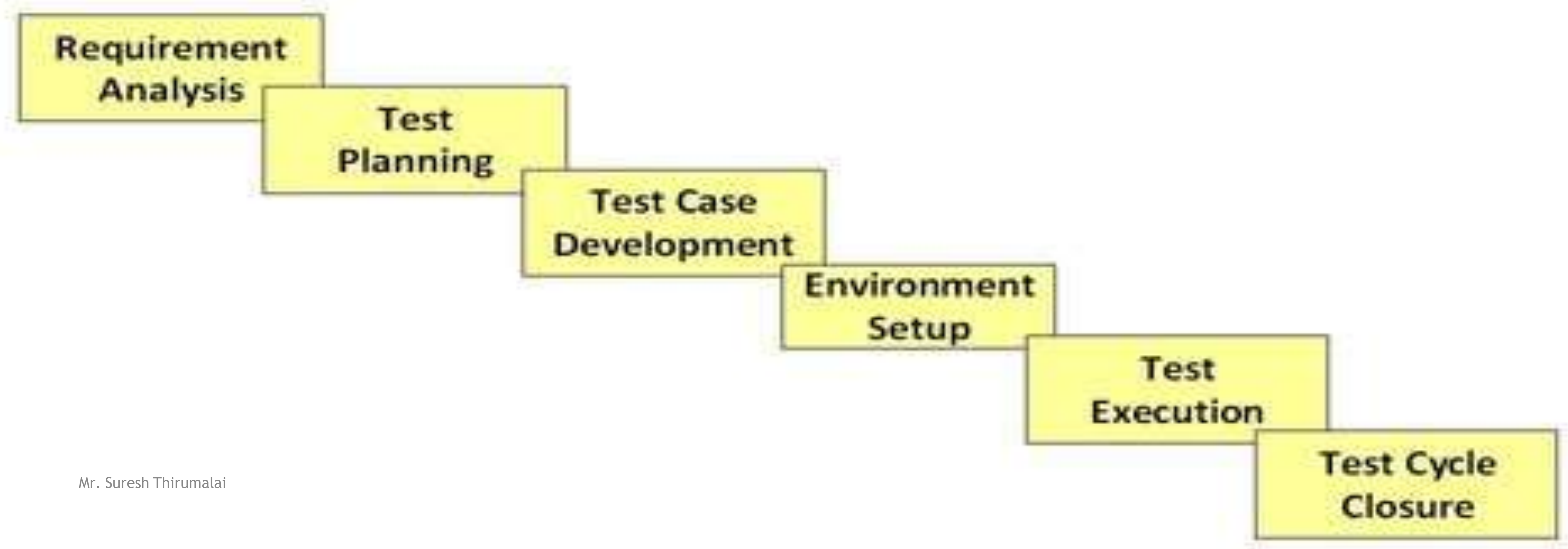
16. SDLC

Software Development Life Cycle


Requirement Gathering
Analysis
Design
Coding
Testing
Deployment
Software Development Life Cycle
© www.SoftwareTestingMaterial.com

17. STLC

Software Testing Life Cycle



Phase 1:

Requirement Analysis

- Test Team studies the requirements to identify the testable requirements
- QA team interact with Client, BA, Tech Leads, System architects
- To understand the requirements in detail

- Deliverables

  RTM

  Automation Feasibility Report

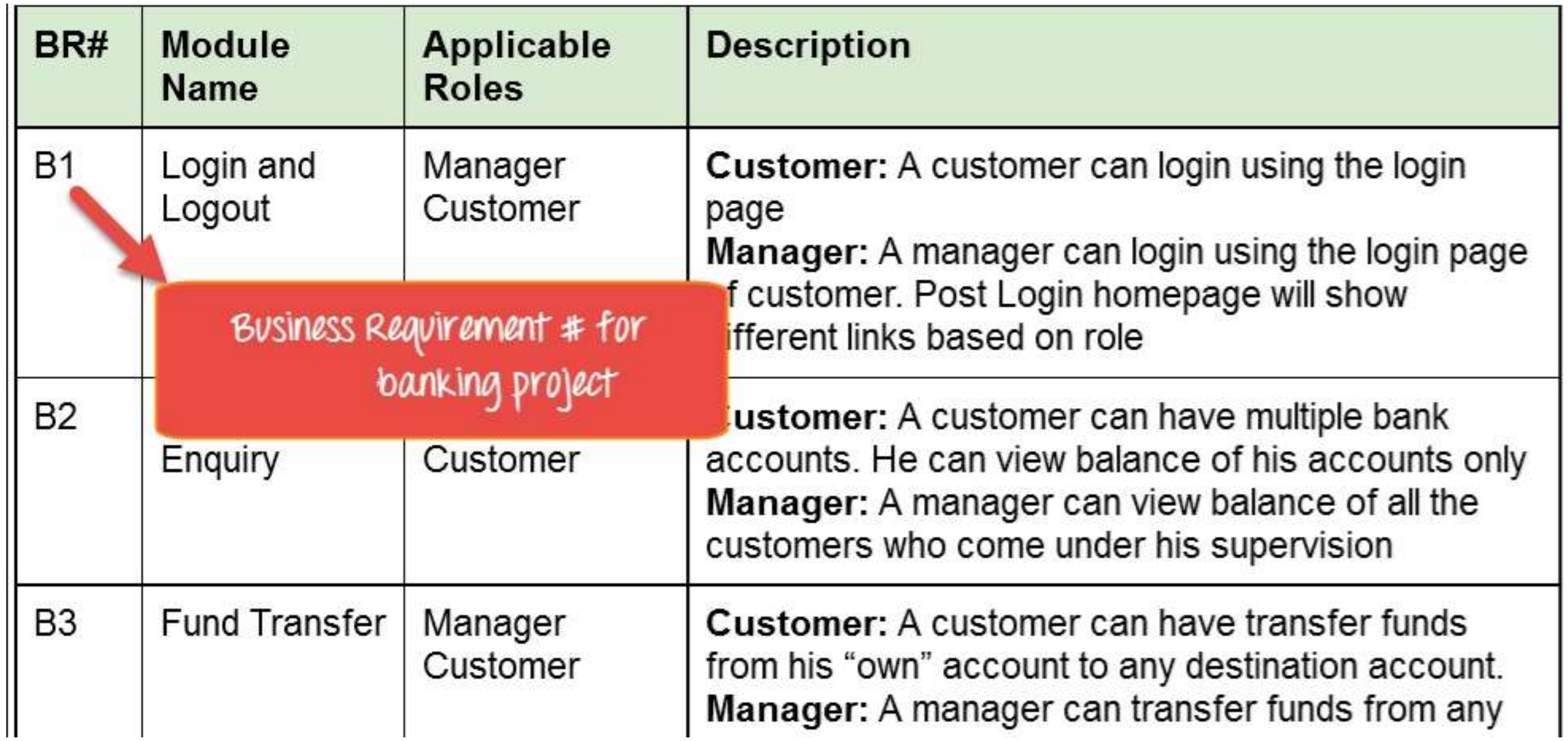
Requirements Traceability Matrix (RTM)

- It is a document that maps and traces the user requirement with test cases.
- It is used to track the requirements and to check the current project requirements are met

Advantage of RTM

- It confirms 100% test coverage
- It highlights any requirements missing or document inconsistencies
- It shows the overall defects or execution status with a focus on business requirements
- It helps in analyzing or estimating the impact on the QA team's work with respect to revisiting or re-working on the test cases

Business Requirement Document (BRD)

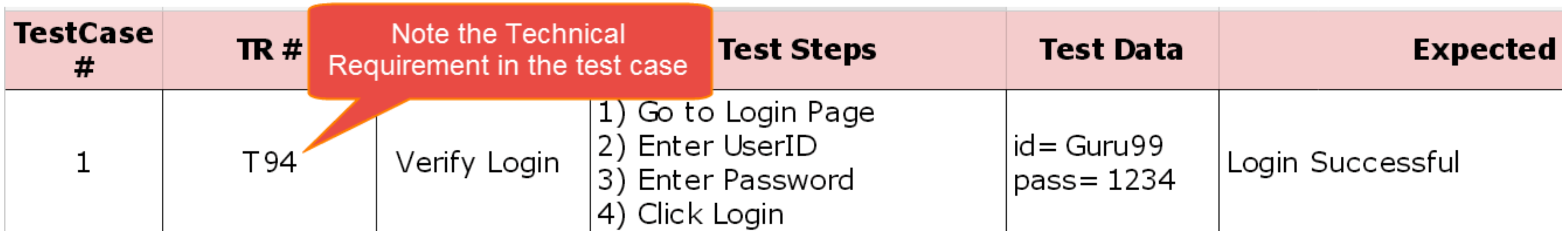


BR#	Module Name	Applicable Roles	Description
B1	Login and Logout	Manager Customer	**Customer:** A customer can login using the login page **Manager:** A manager can login using the login page f customer. Post Login homepage will show ifferent links based on role
B2	Enquiry	Customer	**ustomer:** A customer can have multiple bank accounts. He can view balance of his accounts only **Manager:** A manager can view balance of all the customers who come under his supervision
B3	Fund Transfer	Manager Customer	**Customer:** A customer can have transfer funds from his "own" account to any destination account. **Manager:** A manager can transfer funds from any

Business Requirement # for banking project

Mr. Suresh Thirumalai

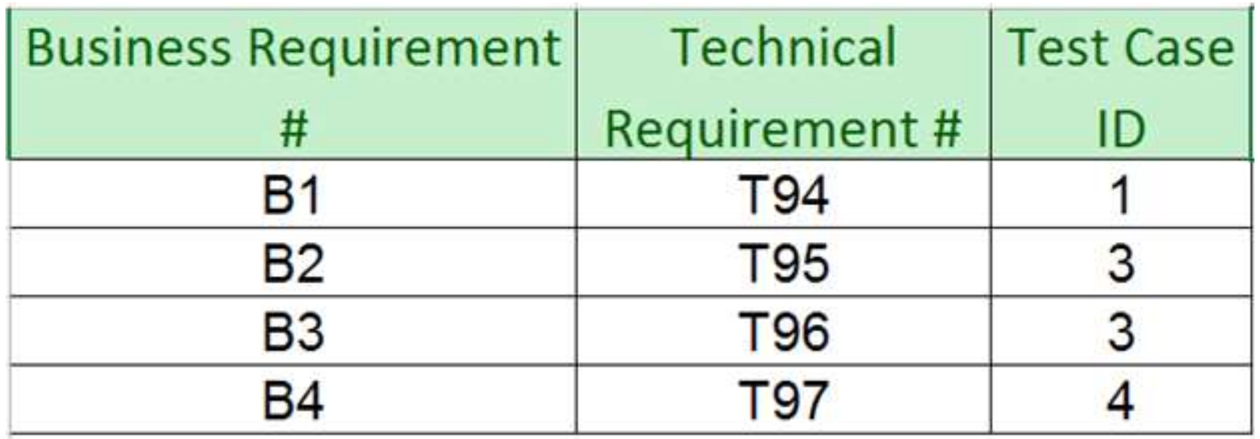
Technical Requirement Document (TRD)



Note the Technical Requirement in the test case

TestCase #	TR #		Test Steps	Test Data	Expected
1	T94	Verify Login	1) Go to Login Page 2) Enter UserID 3) Enter Password 4) Click Login	id= Guru99 pass= 1234	Login Successful

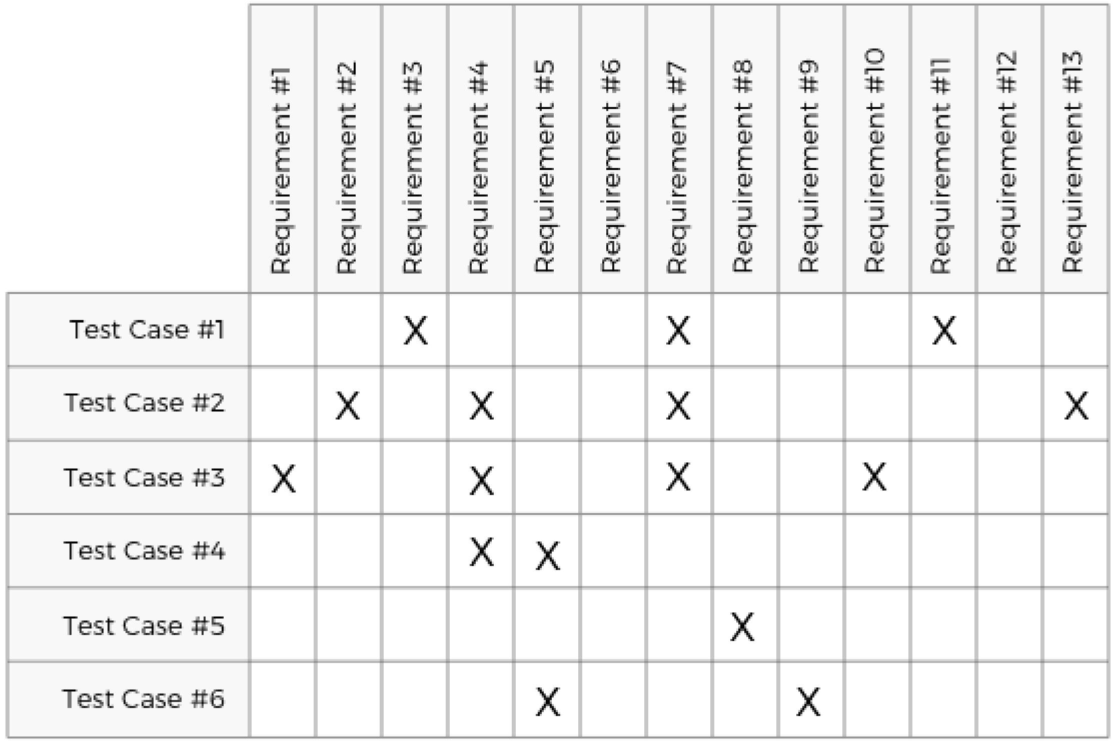
RTM



Business Requirement #	Technical Requirement #	Test Case ID
B1	T94	1
B2	T95	3
B3	T96	3
B4	T97	4

Requirement Traceability Matrix

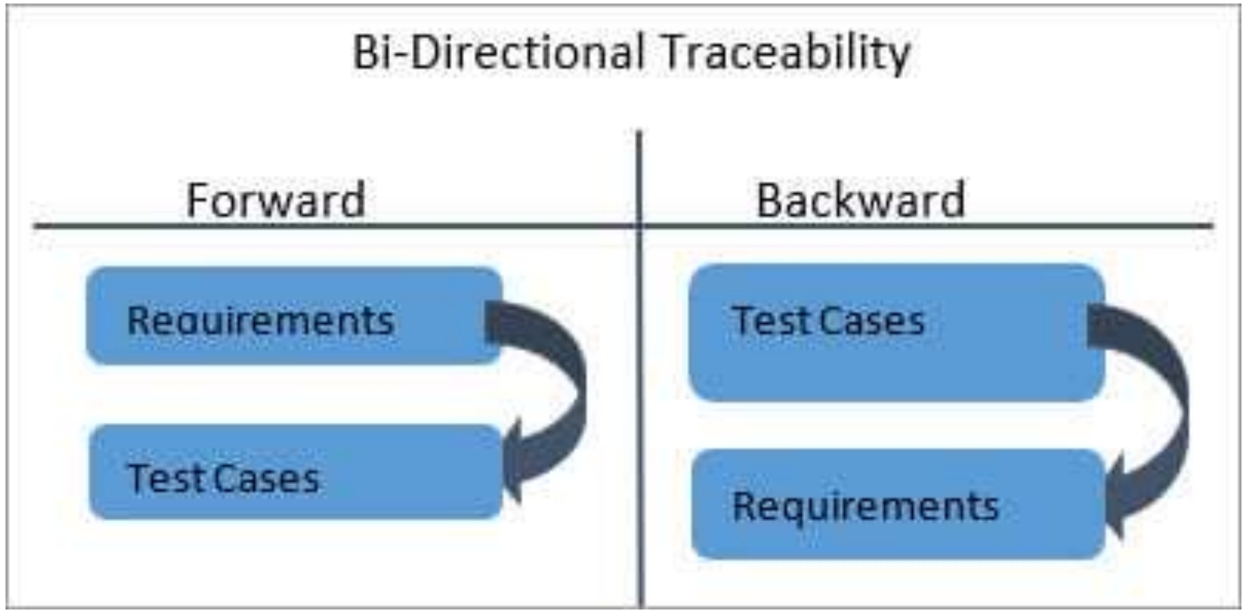
RTM Table



	Requirement #1	Requirement #2	Requirement #3	Requirement #4	Requirement #5	Requirement #6	Requirement #7	Requirement #8	Requirement #9	Requirement #10	Requirement #11	Requirement #12	Requirement #13
Test Case #1			X				X				X		
Test Case #2		X		X			X						X
Test Case #3	X			X			X			X			
Test Case #4				X	X								
Test Case #5								X					
Test Case #6					X				X				

Types:

- Forward Traceability
- Backward Traceability



Phase 2:

Test Planning

- Senior QA Manager determines Effort & Cost estimates
- Preparation of Test Plan Document for various types of testing
- Tool selection
- Resource planning and determining R&R
- Training Requirements

- Deliverables:

  Test Plan Document

  Effort Estimation Document

Test Plan Document

- Test Plan is a document describing the

  scope,

  approach,

  resources, and

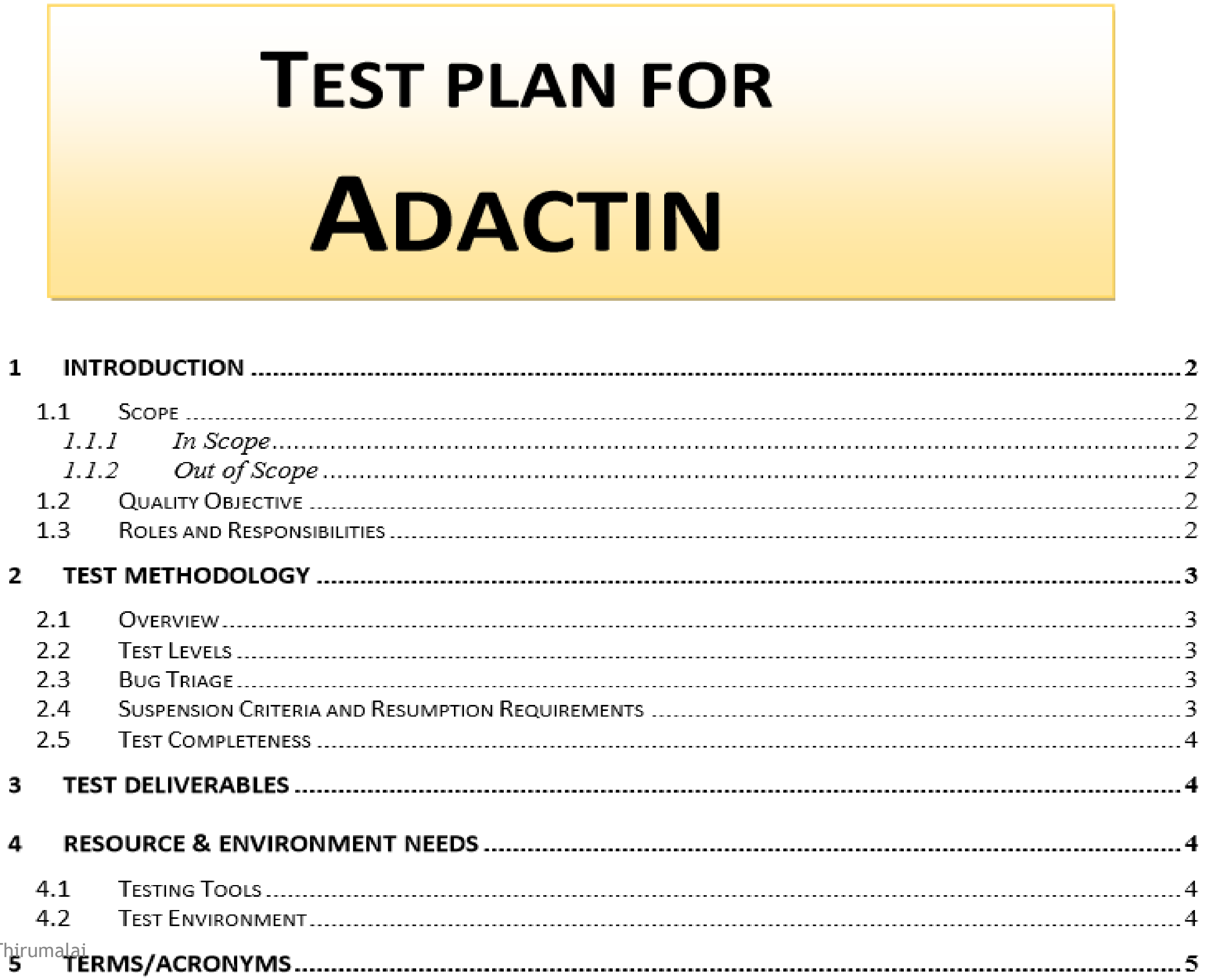
  schedule of planned test activities.

- it deals with test coverage, features to be tested, features not to be tested, estimation, scheduling and resource management.

Create Test Plan Document

Follow the below steps below to create a test plan.

1. Analyze the product
2. Design the Test Strategy
3. Define the Test Objectives
4. Define Test Criteria
5. Resource Planning
6. Plan Test Environment
7. Schedule & Estimation
8. Determine Test Deliverables



TEST PLAN FOR ADACTIN

Phase 3:

Test Case Development

- Create test Cases, Automation Scripts
- Review and Verify test cases & test scripts
- Create Test Data

- Deliverables

    Test Cases/Scripts

    Test Data

Phase 4:
Environment Setup

- Understand required architecture, environment set-up
- Prepare H/W , S/W requirement list
- Setup Test Environment and test data
- Perform Smoke Test on the build to do a readiness check of the given environment

- <u>Deliverables</u>

  Environment Ready with Test Data Setup

  Smoke Test Results

Test Environment Setup

- System and applications
- Test data
- Database server
- Front-end running environment
- Client operating system
- Browser
- Hardware includes Server Operating system
- Network
- Documentation required like reference documents/configuration guides/installation guides/ user manual

Phase 5:
Test Execution

- Execute tests as per Plan
- Document test results and log defects for failed cases
- Bugs will be reported to dev team for correction
- Retest the Defect fixes
- Track the defects to closure

- <u>Deliverables</u>

  Test Cases – Updated Result

  Defect Reports

Phase 6:

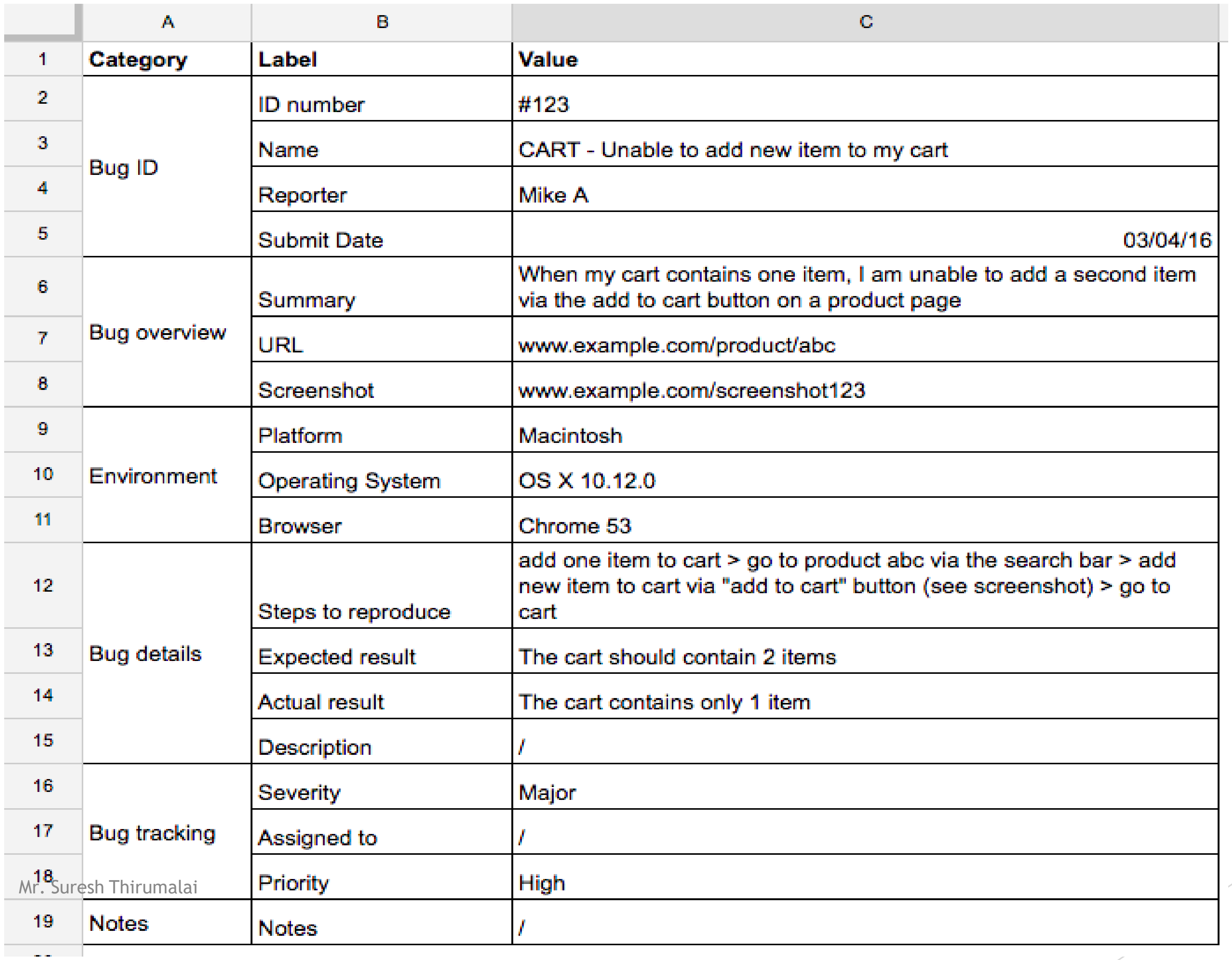
Test Cycle Closure

- Prepare Test closure report
- Evaluate cycle completion criteria
- Prepare test metrices.
- Test Result analysis

- <u>Deliverables</u>

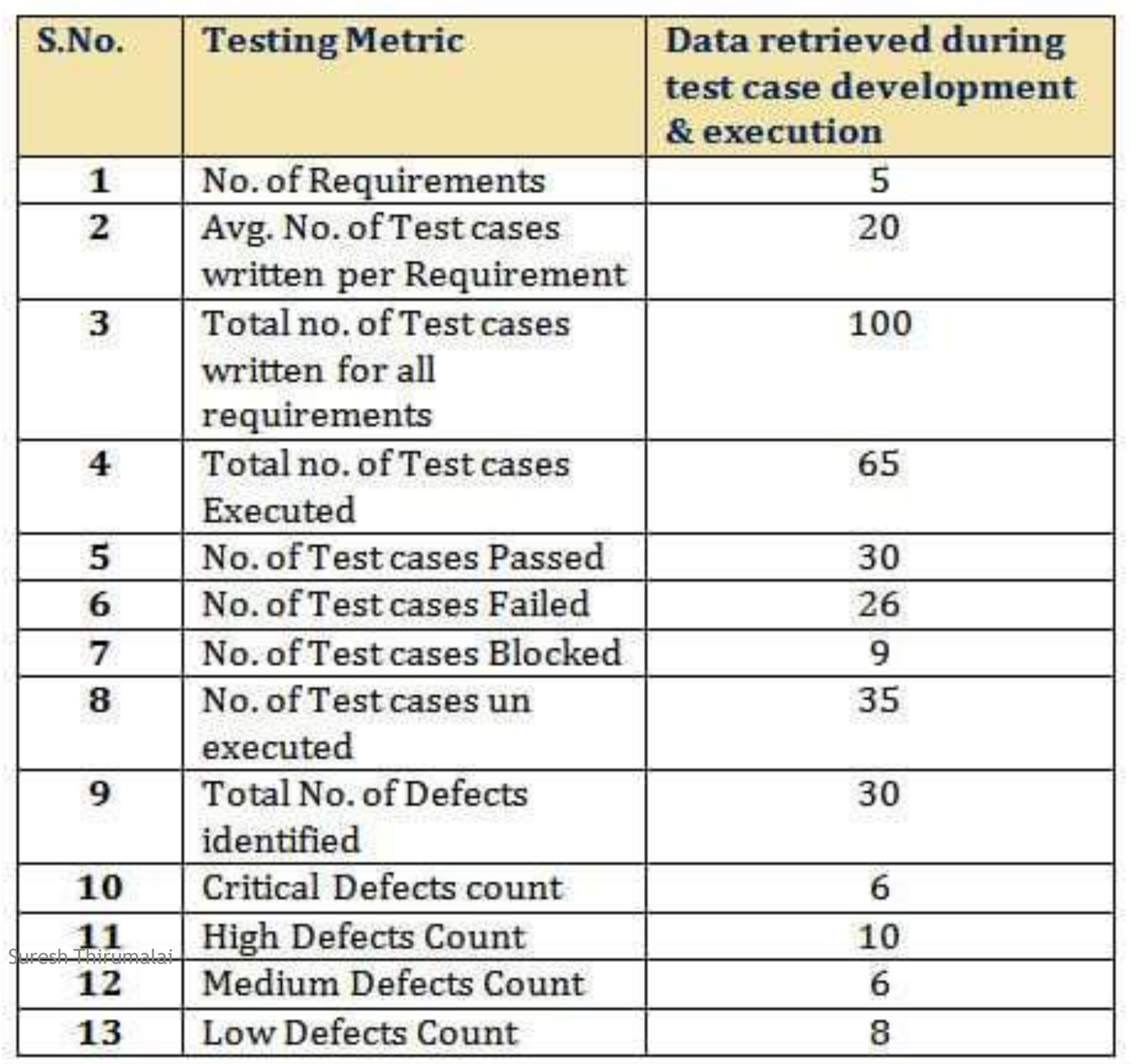
  Test Closure Report

Bug Report



	A	B	C
1	**Category**	**Label**	**Value**
2	Bug ID	ID number	#123
3		Name	CART - Unable to add new item to my cart
4		Reporter	Mike A
5		Submit Date	03/04/16
6	Bug overview	Summary	When my cart contains one item, I am unable to add a second item via the add to cart button on a product page
7		URL	www.example.com/product/abc
8		Screenshot	www.example.com/screenshot123
9	Environment	Platform	Macintosh
10		Operating System	OS X 10.12.0
11		Browser	Chrome 53
12	Bug details	Steps to reproduce	add one item to cart > go to product abc via the search bar > add new item to cart via "add to cart" button (see screenshot) > go to cart
13		Expected result	The cart should contain 2 items
14		Actual result	The cart contains only 1 item
15		Description	/
16	Bug tracking	Severity	Major
17		Assigned to	/
18		Priority	High
19	Notes	Notes	/

Test Summary Report: Testing Metrics



S.No.	Testing Metric	Data retrieved during test case development & execution
1	No. of Requirements	5
2	Avg. No. of Test cases written per Requirement	20
3	Total no. of Test cases written for all requirements	100
4	Total no. of Test cases Executed	65
5	No. of Test cases Passed	30
6	No. of Test cases Failed	26
7	No. of Test cases Blocked	9
8	No. of Test cases un executed	35
9	Total No. of Defects identified	30
10	Critical Defects count	6
11	High Defects Count	10
12	Medium Defects Count	6
13	Low Defects Count	8

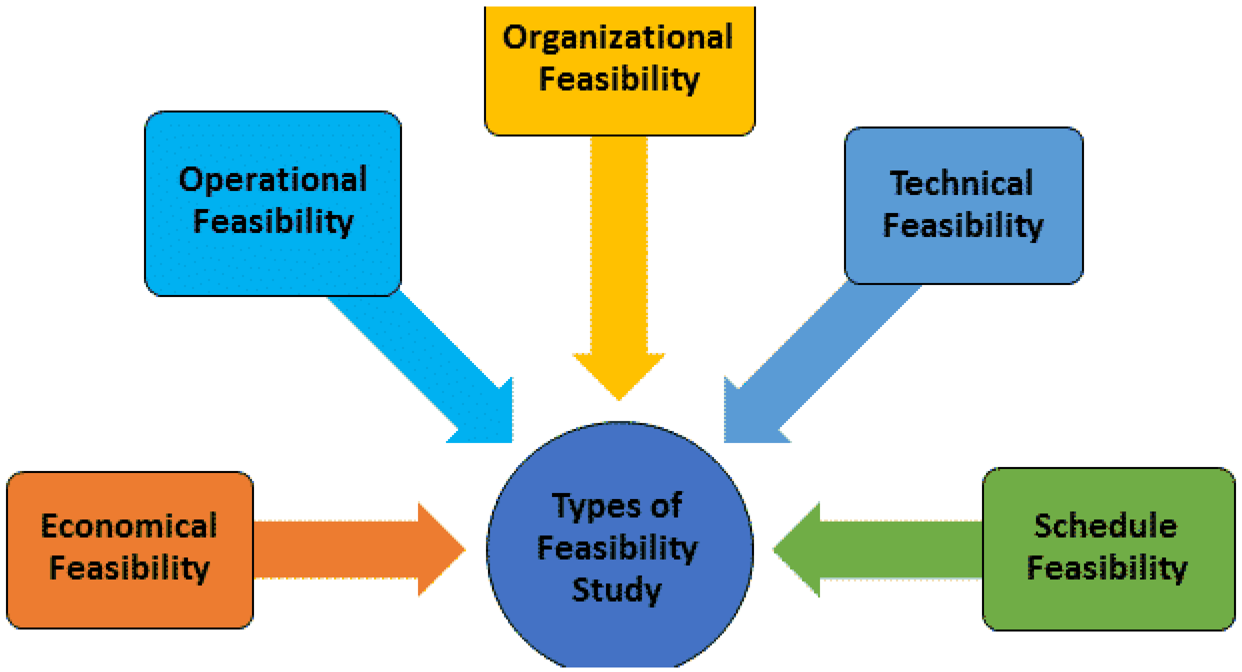
Test Strategy

- A strategy plan for defining the testing approach, what you want to accomplish and how you are going to achieve it.
- Writing a Test Strategy effectively is a skill every tester should achieve in their career. It initiates your thought process which helps to discover many missing requirements. Thinking and test planning activities help a team to define the Testing scope and Test coverage.
- It helps Test managers to get the clear state of the project at any point. The chances of missing any test activity are very low when there is a proper test strategy in place.

Feasibility Study

- **Feasibility Study:**
  Feasibility study explores system requirements to determine project feasibility. There are several fields of feasibility study including economic feasibility, operational feasibility, technical feasibility. The goal is to determine whether the system can be implemented or not. The process of feasibility study takes as input the requirement details as specified by the user and other domain-specific details. The output of this process simply tells whether the project should be undertaken or not and if yes, what would the constraints be. Additionally, all the risks and their potential effects on the projects are also evaluated before a decision to start the project is taken.

Types of feasibility Study



Story Point Estimations

- Story point Estimations are done in Agile projects using different techniques like
- Planning Poker,
- Bucket System,
- Affinity Mapping,
- T-Shirt Sizes (sizes: XS (Extra Small), S (Small), M (Medium), L (Large), XL (Extra Large))
- Dot Voting

Important Questions in STLC

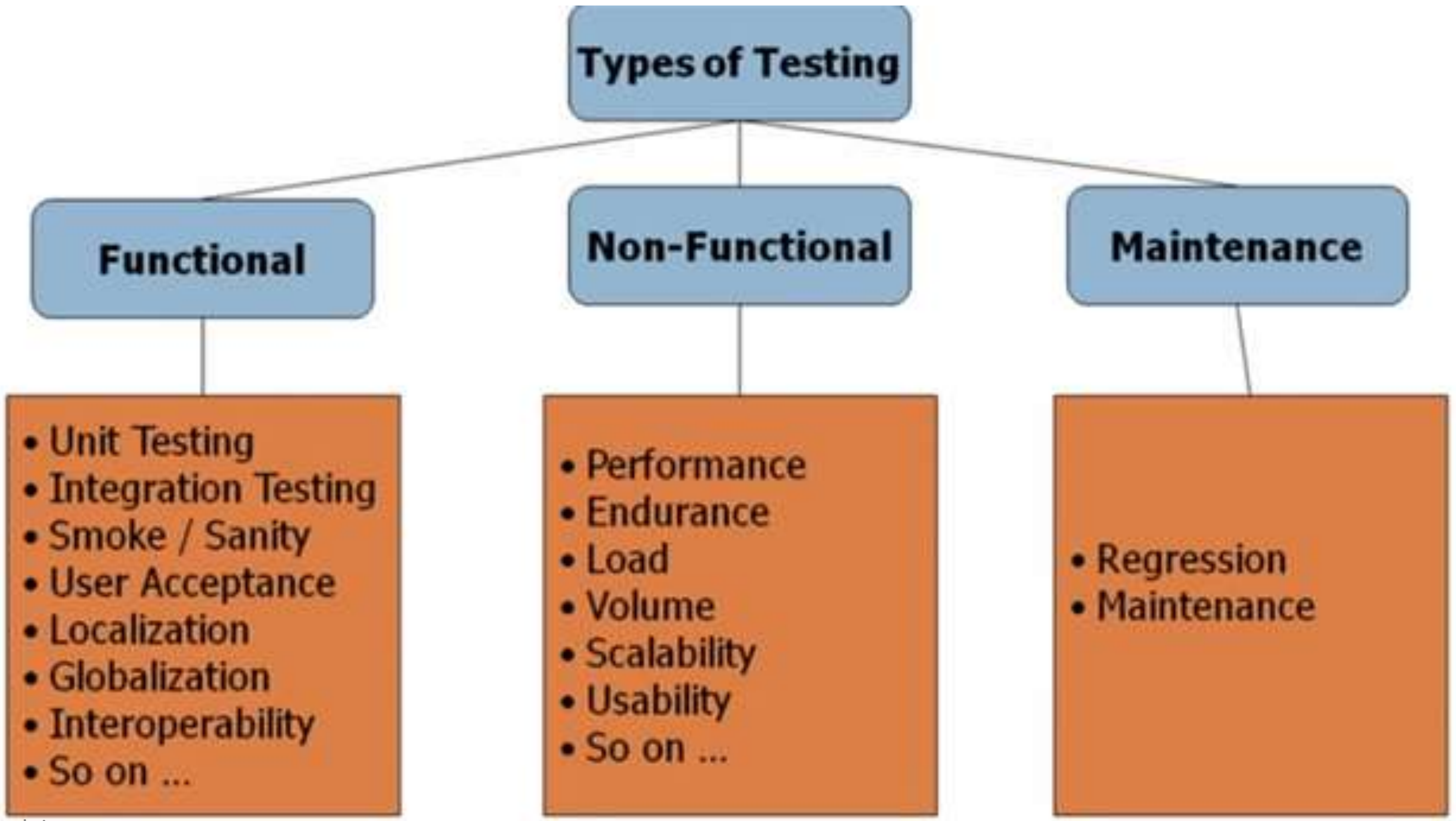
1. How to create RTM?
2. How to create Test Plan?
3. How to prepare Feasibility Report?
4. How to estimate Effort?
5. What is Test Strategy?

Software Testing

- The process, to evaluate the functionality of a software application.
- Activity to check whether the actual results match the expected results and to ensure that the software system is defect/bug free
- It can be done manually or using automated tools

Manual Testing
Automated Testing

Types of Testing (Manual/Automation)



1. Functional Testing

- Checks whether the application is providing all the functionalities that were mentioned in the functional requirement of the application.

2. Non-Functional Testing

- To check **non-functional** aspects (performance, usability, reliability, etc) of a software application.
- It is designed to **test** the readiness of a system as per **nonfunctional** parameters which are never addressed by functional **testing**

Data entry successful!
Is takes 5 minutes just to save a file?!
Functional Testing
Non-functional Testing
Buzzle.com

3. Maintenance

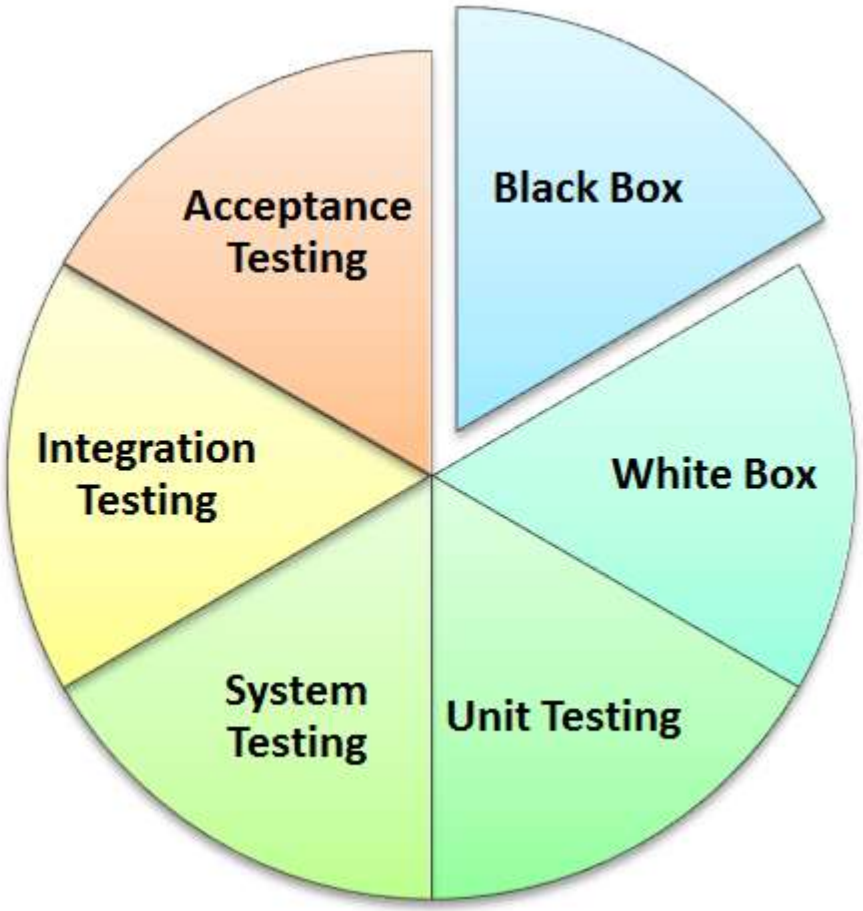
- Regression Test

Slide No: 182

18. Manual Testing (Functional)

- It is a type of software testing where testers manually test cases without using any automation tools.
- It is most primitive of all testing types and helps to find bugs in the software system.
- Any new application must be manually tested before its testing can be automated.
- Manual testing requires more effort but is necessary to check automation feasibility.

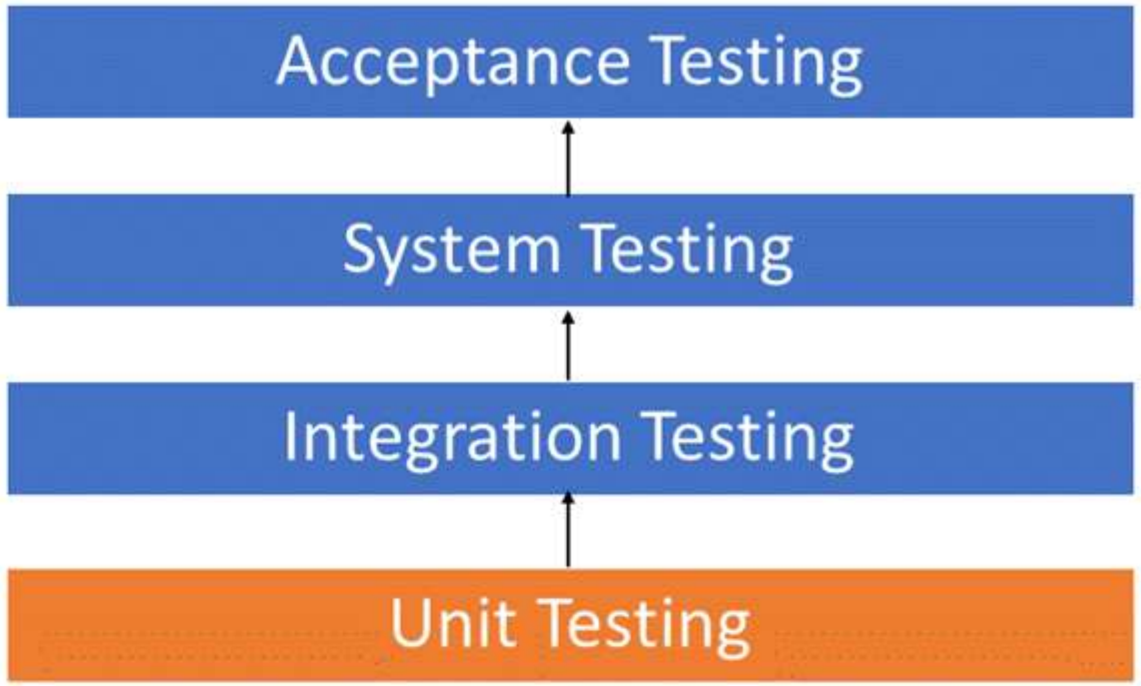
Techniques in Manual testing



How to perform a Test:

- 1. Read and understand the software project documentation / guides
- 2. Draft Test cases that cover all the requirements mentioned in the documentation
- 3. Review the test cases with Team Lead, Client
- 4. Execute the test cases
- 5. Report Bugs
- 6. After bugs fixed, execute failed test cases.

Software Testing Levels (Hierarchy)



1. Unit Testing

- Individual units or components of a software are tested
- It is done during the development of application by developers
- A unit may be an individual function, method, procedure, module, or object

<u>Tools</u>

JUnit

NUnit

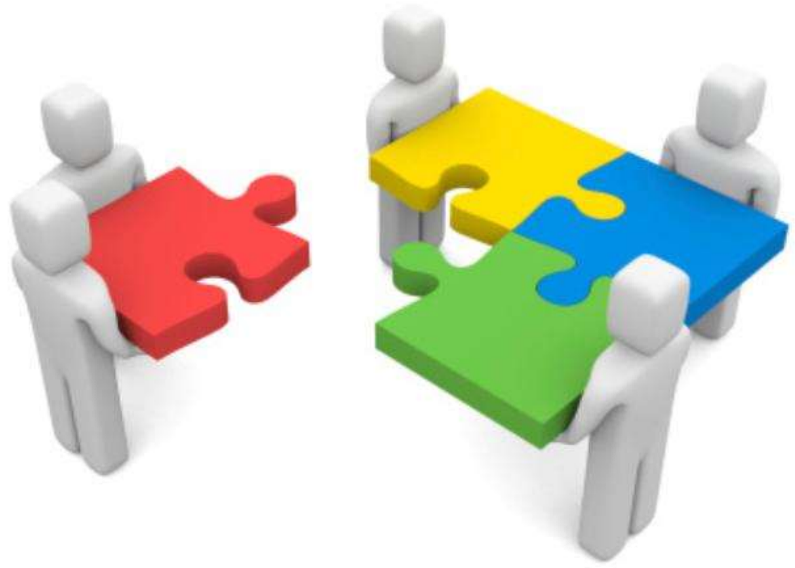
Mr. Suresh Thirumalai

2. Integration Test

- Software modules are integrated logically and tested as a group
- It focuses mainly on the interfaces and flow of data/information between the modules

Example:

- An application has 3 modules:
- Login Page, Mail box, Delete emails



3. System Testing

- It validates the complete and fully integrated software product
- Purpose is evaluate the end-to-end system specifications

<u>Different Types of System Testing</u>

Smoke Test,

Sanity Test,

Regression Test,

Black Box,

White Box,

User Acceptance Test,

Black Box Testing

- *It* is tested **without looking**

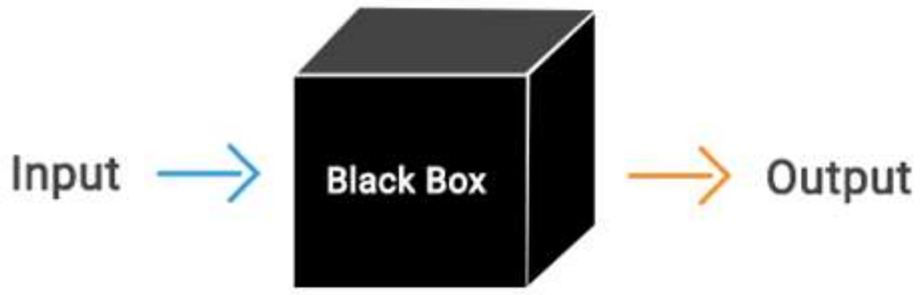
  at the internal code structure,

  implementation details and

  knowledge of internal paths of the software.

- This type of testing is *based entirely on*

  software requirements and specifications.



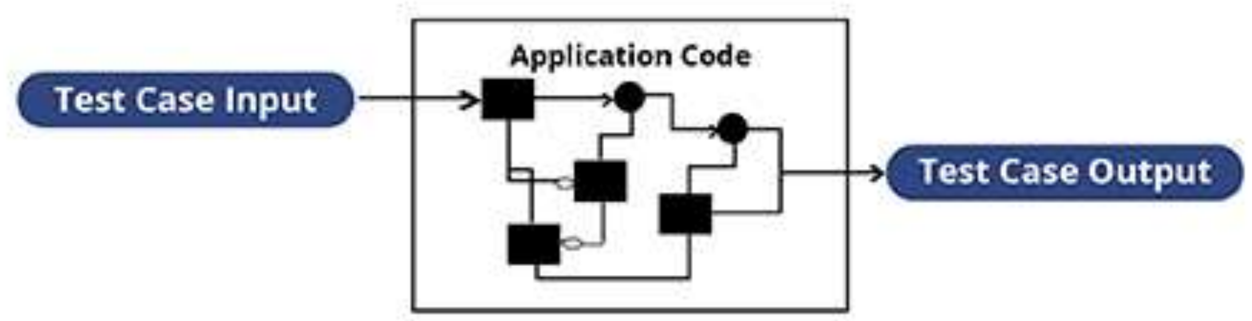
White box testing

- Testing software's <u>internal</u> <u>structure</u>, <u>design</u>, and <u>coding</u>.
- In this type of testing, the <u>code</u> <u>is</u> <u>visible</u> to the tester.
- It focuses primarily on verifying

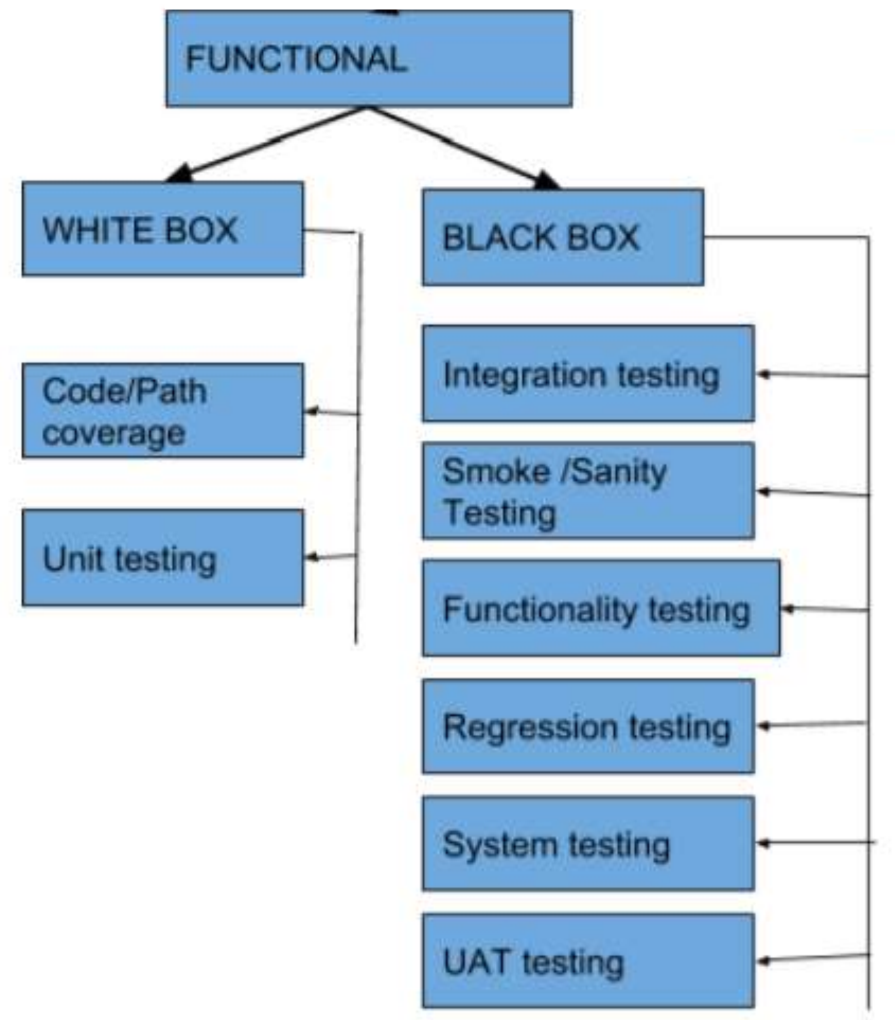
  <u>the flow of inputs & outputs through the application</u>,

  <u>improving design and usability</u>,

  <u>strengthening security</u>.

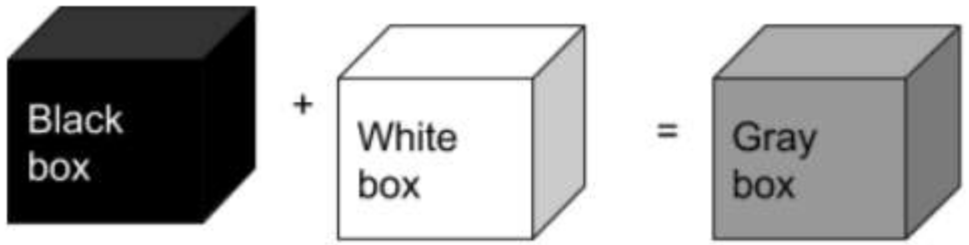


Black Box Vs White Box



Grey box testing

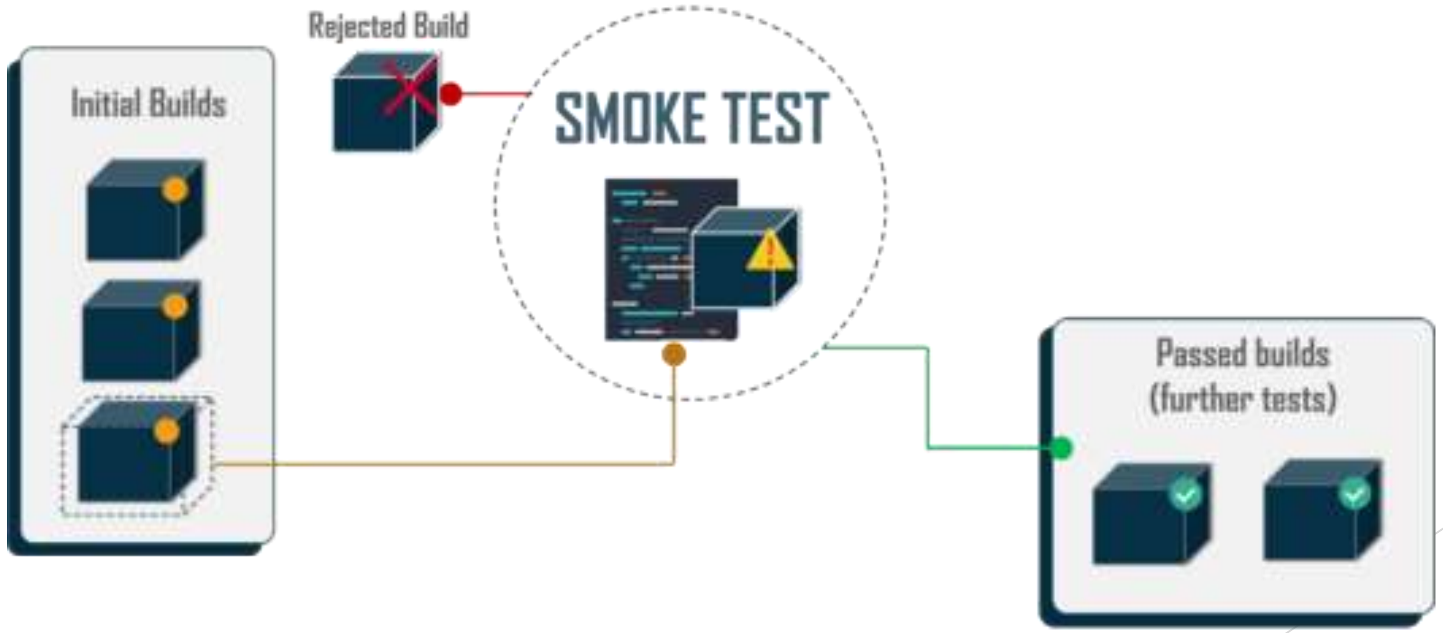
- Gray-box testing is a combination of white-box testing and black-box testing.
- **Grey box testing** is when the **tester** has a partial understanding of the internal structure in the system under **test**.
- **Grey box testing** is a process for debugging **software** applications by making an input through the front-end, and verifying the data on the back-end



Mr. Suresh Thirumalai

Smoke Testing

To confirm that the **critical functionalities of the program are working fine** after software build

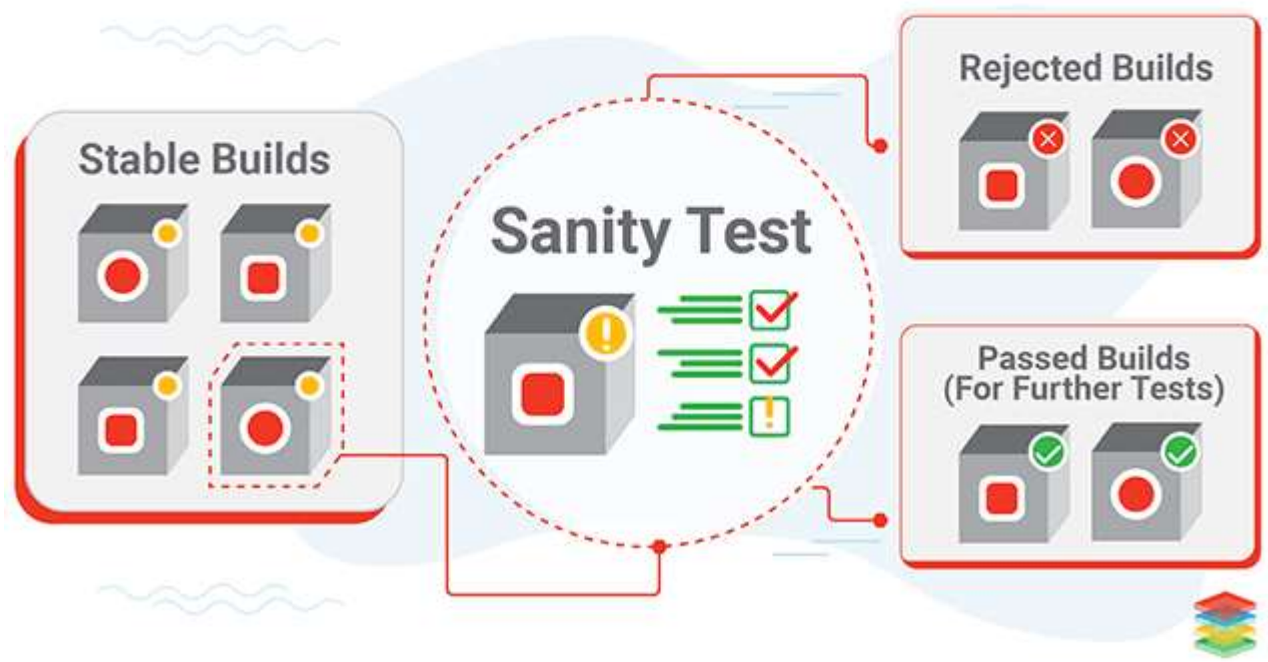


When Smoke Testing?

- "before" any detailed functional or regression tests are executed on the software build.
- To decide if a build is stable enough to proceed with further testing
- **Why?**
  - If the build is imperfect/defective, further testing would be waste of time and resources.
  - To detect early major issues.

Sanity Testing?

To check the **new functionality/bugs have been fixed.**



Smoke vs Sanity

Smoke Test

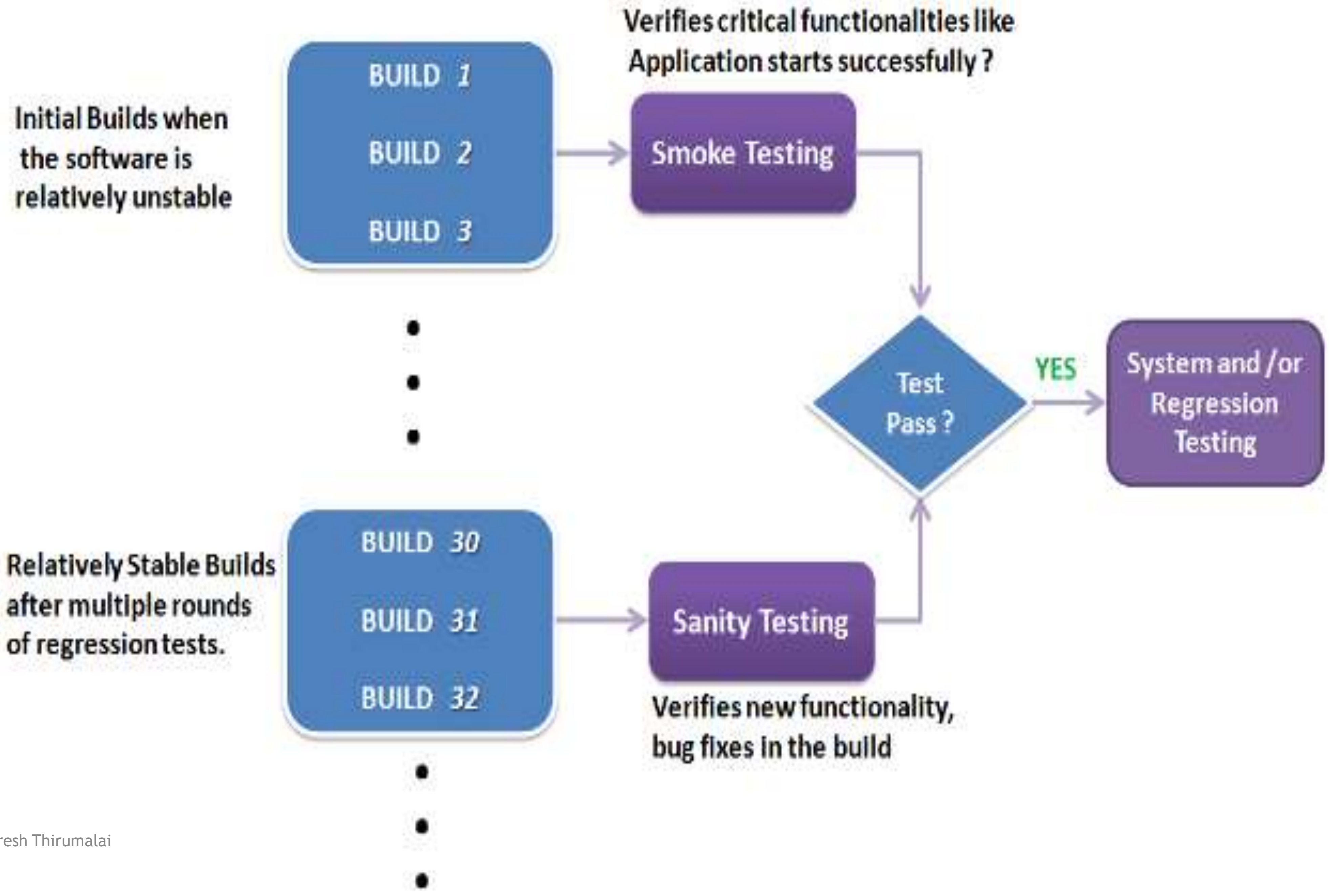
It is like General Health Check Up

Sanity Test

It is like specialized health check up

Ad-hoc Testing

- When a software **testing performed** without proper planning and documentation, it is said to be **Adhoc Testing**. Such kind of **tests** are **executed** only once unless we uncover the defects. **Adhoc Tests** are done after formal **testing is performed** on the application.

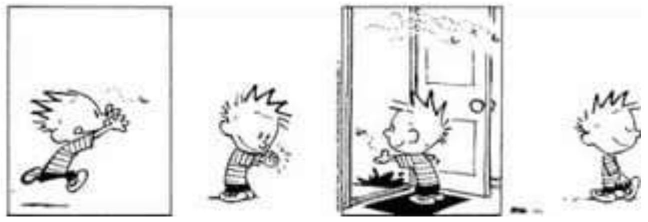

Verifies critical functionalities like Application starts successfully ?
Initial Builds when the software is relatively unstable
BUILD 1
BUILD 2
BUILD 3
Smoke Testing
Test Pass ?
YES
System and /or Regression Testing
Relatively Stable Builds after multiple rounds of regression tests.
BUILD 30
BUILD 31
BUILD 32
Sanity Testing
Verifies new functionality, bug fixes in the build

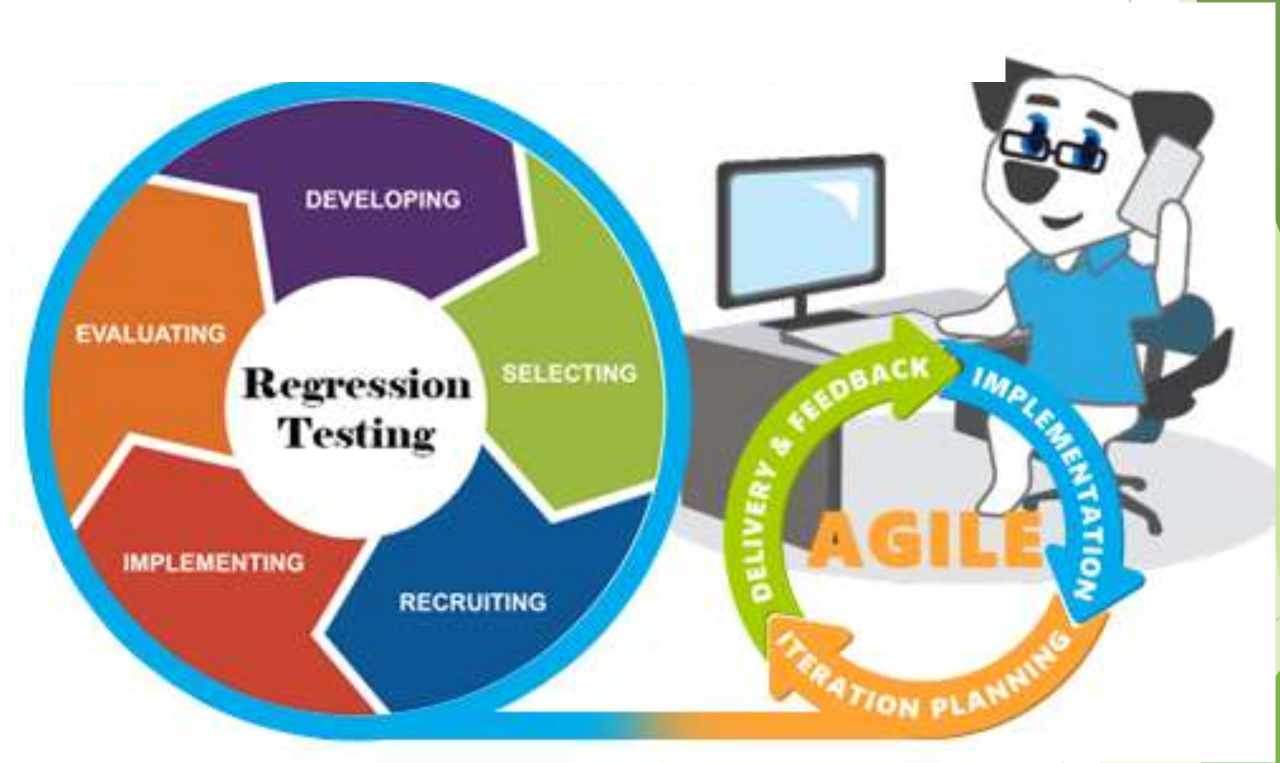
Regression Testing

- ✓ It is done to verify that a code change in the software does not impact the existing functionality of the product.
- ✓ This is to make sure the product works fine with new functionality, bug fixes or any change in the existing feature.
- ✓ Previously executed test cases are re-executed in order to verify the impact of change.

Regression

Regression:
"when you fix one bug, you introduce several newer bugs."





When Regression Testing?

- After the changes in requirements and code modified according to the requirement
- After new feature added to the software
- After defect fixed
- After performance issue fixed
- After alterations to an application's hosting environment.

Selecting Test Cases for Regression Testing

1. Select test cases for Regression; there are recent code changes /functional changes
2. Select test cases that map to the business requirements
3. Select test cases for Regression testing in areas with frequent bugs/defects
4. Select test cases for Regression testing of the areas which are visible to the user
5. Select all integration test cases for Regression testing
6. Select all complex test cases for Regression testing
7. Select test cases based on priorities for Regression testing
8. Select test cases for Regression testing based on criticality and impact of bug fixes
9. Select a sample of successful and failed test cases for Regression testing
10. Functionalities which are more visible to the users

Regression check is covered under <u>two categories</u>:

Sprint Level Regression

End to End Regression

1) Sprint Level Regression

- It is done mainly for the new functionality or the enhancement that is done in the latest sprint. Test cases from the test suite are selected as per the newly added functionality or the enhancement that is done.

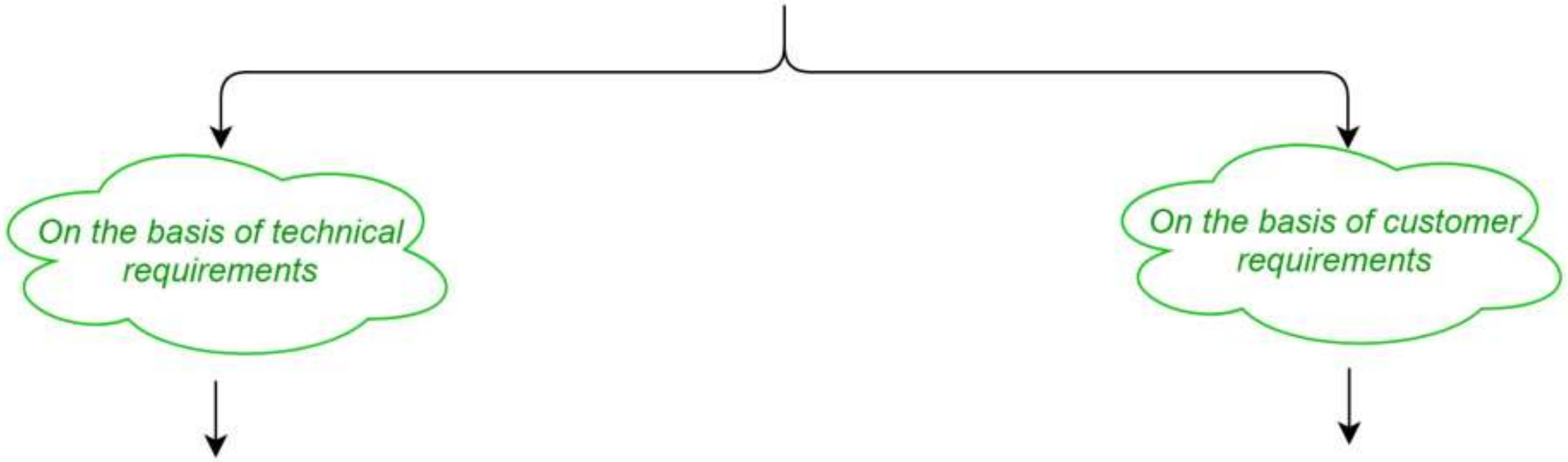
2) End-to-End Regression

- End-to-End Regression includes all the test cases that are to be re-executed to test the complete product end to end by covering all the core functionalities of the Product.

Regression Testing Steps:

- Select the Tests for Regression.
- Choose the apt tool and automate the Regression Tests
- Verify applications with Checkpoints
- Manage Regression Tests/update when required
- Schedule the tests
- Integrate with the builds
- Analyze the results

Prioritization


On the basis of technical requirements
On the basis of customer requirements

Priority Code 1: Essential Test Case

Priority Code 2: Important Test Case

Priority Code 3: Execute, if resources permits

Priority Code 4: Not important Test Case

Priority Code 5: Redundant Test Case

Priority Code 1: Important for the customer

Priority Code 2: Required to increase the customer satisfaction

Priority Code 3: Helps to increase the market share of the product

UAT – User Acceptance Testing

- Testing a software by the user or client

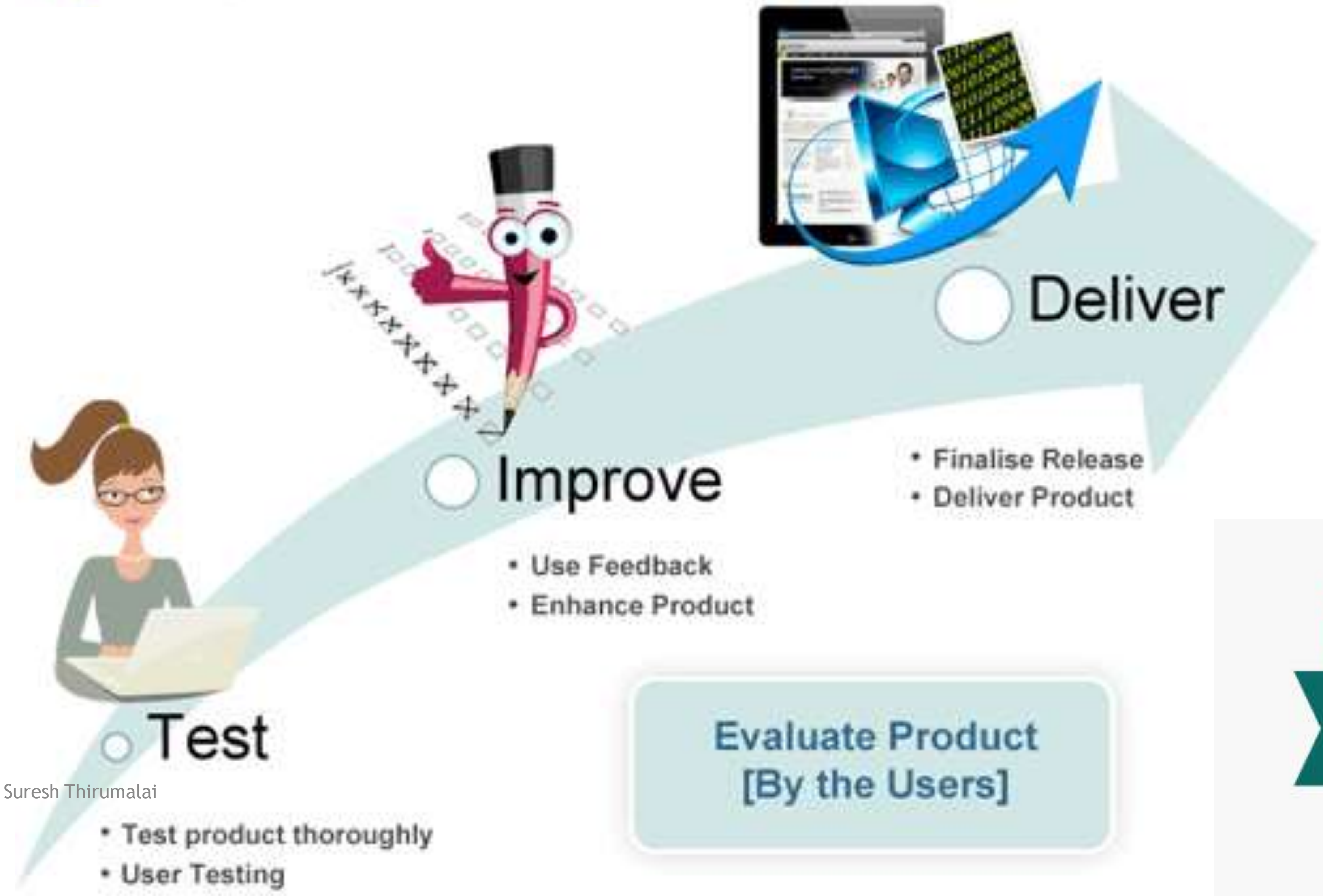
  to determine whether it can be accepted or not.

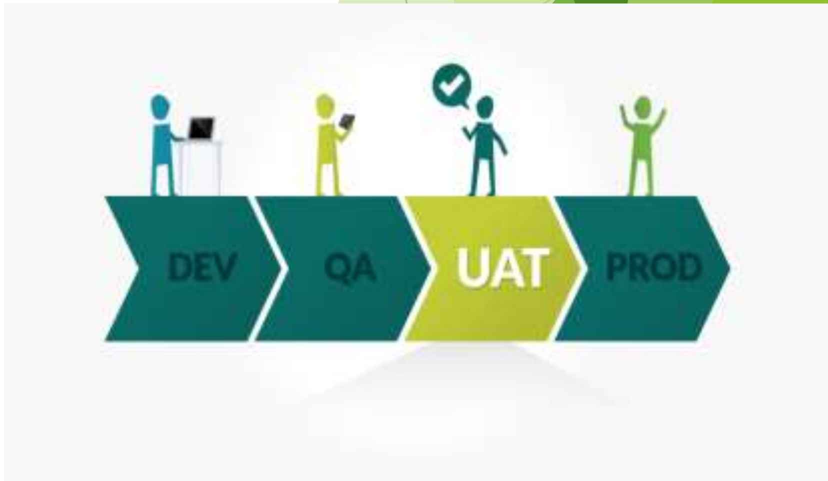
- This is the final testing performed

  once the functional, system & regression testing

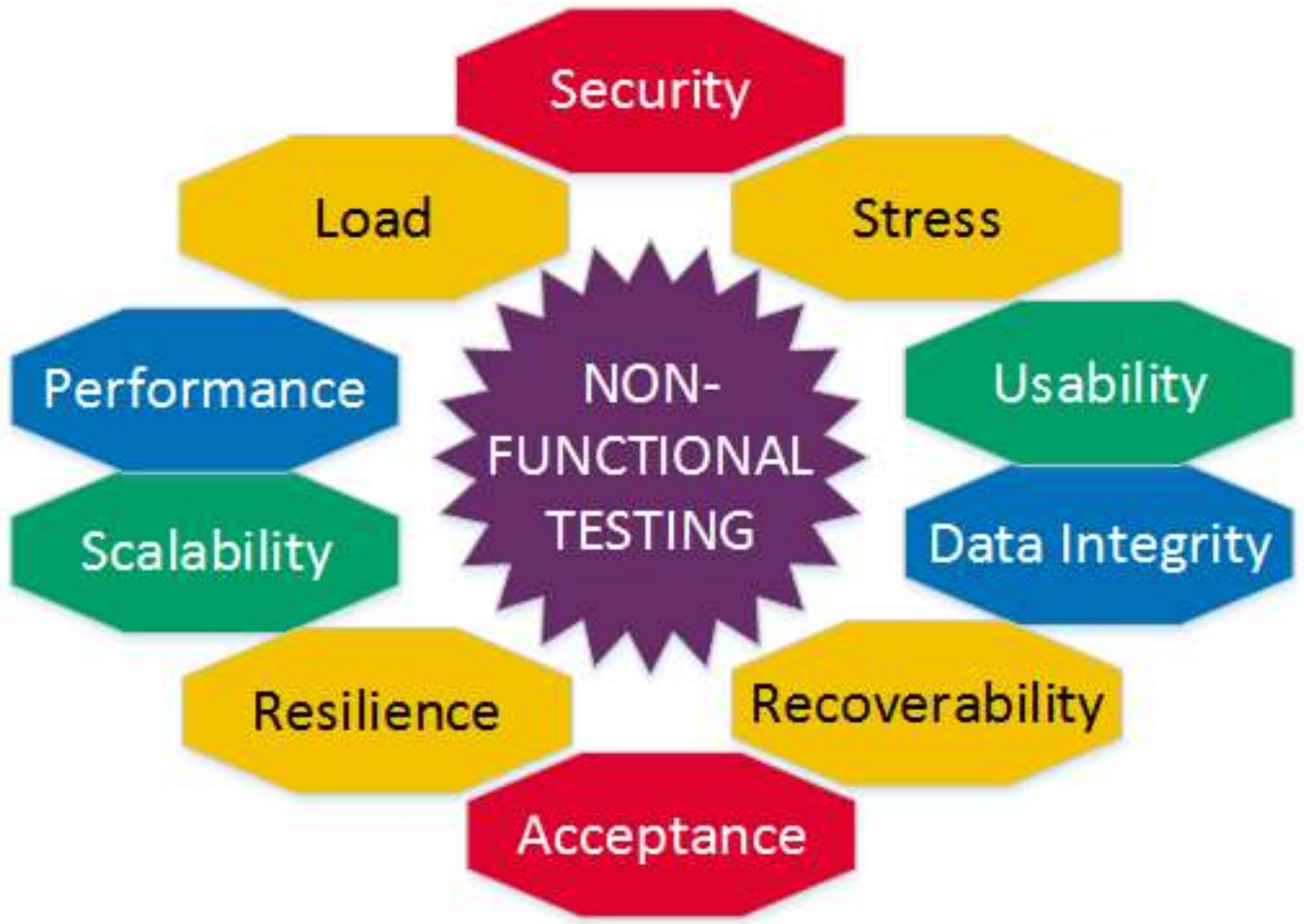
  are completed

After UAT…


Deliver
• Finalise Release
• Deliver Product
Improve
• Use Feedback
• Enhance Product
Test
• Test product thoroughly
• User Testing
Evaluate Product [By the Users]


DEV
QA
UAT
PROD

Non-Functional Testing



TESTING	Description	Tools
Performance Test	To Verify Speed, response time, reliability	
Load Test	How application behaves when multiple users access simultaneously	Apache JMeter, LoadNinja, LoadRunner, WebLoad, Load Complete, NeoLoad, LoadView
Stress Test	To verify the stability and reliability	
Volume Test	To analyse the system performance by increasing the volume of data in DB	
Stability Test	To check if the app will crash at any point of time	
Compatibility Test	Software should be installable on all versions of OS	

Functional Testing VS Non-Functional Testing

Parameters	Functional testing	Non-Functional testing
Objective	To verify and validate the software actions that define the process	To verify and validate the performance of the system
Key focus area	Concentrates on the user requirements	Concentrates on the user expectations
Ease of use	Easy to execute black-box test cases	Easy to execute white-box test cases
Functionality	Describes what the system should do	Describes how the system should work
Execution	It is executed before Non-functional testing	It is executed after Functional testing is executed

How to write a Test Scenario?

- Test Scenario
- Test Case
- Test Step
- Actual Result
- Expected Result
- Status
- Comment

Test Scenario and Test cases

- Test Scenario:

  It is one liner statement which tell us about what to **test**

  Ex: **Check Login functionality**

- Test Case:

  Test scenario consists of a detailed **test** procedure

  Here Login Test Scenario has 4 test cases.

  Ex: Check Using **valid** User Id and **valid** Password

  Check Using **Invalid** User Id and valid Password

  Check Using valid User Id and **Invalid** Password

  Check Using **Invalid** User Id and **Invalid** Password

Test Scenario vs Test cases

- **Test Scenario**

  " What to be tested“

- **Test case**

  " How to be tested“

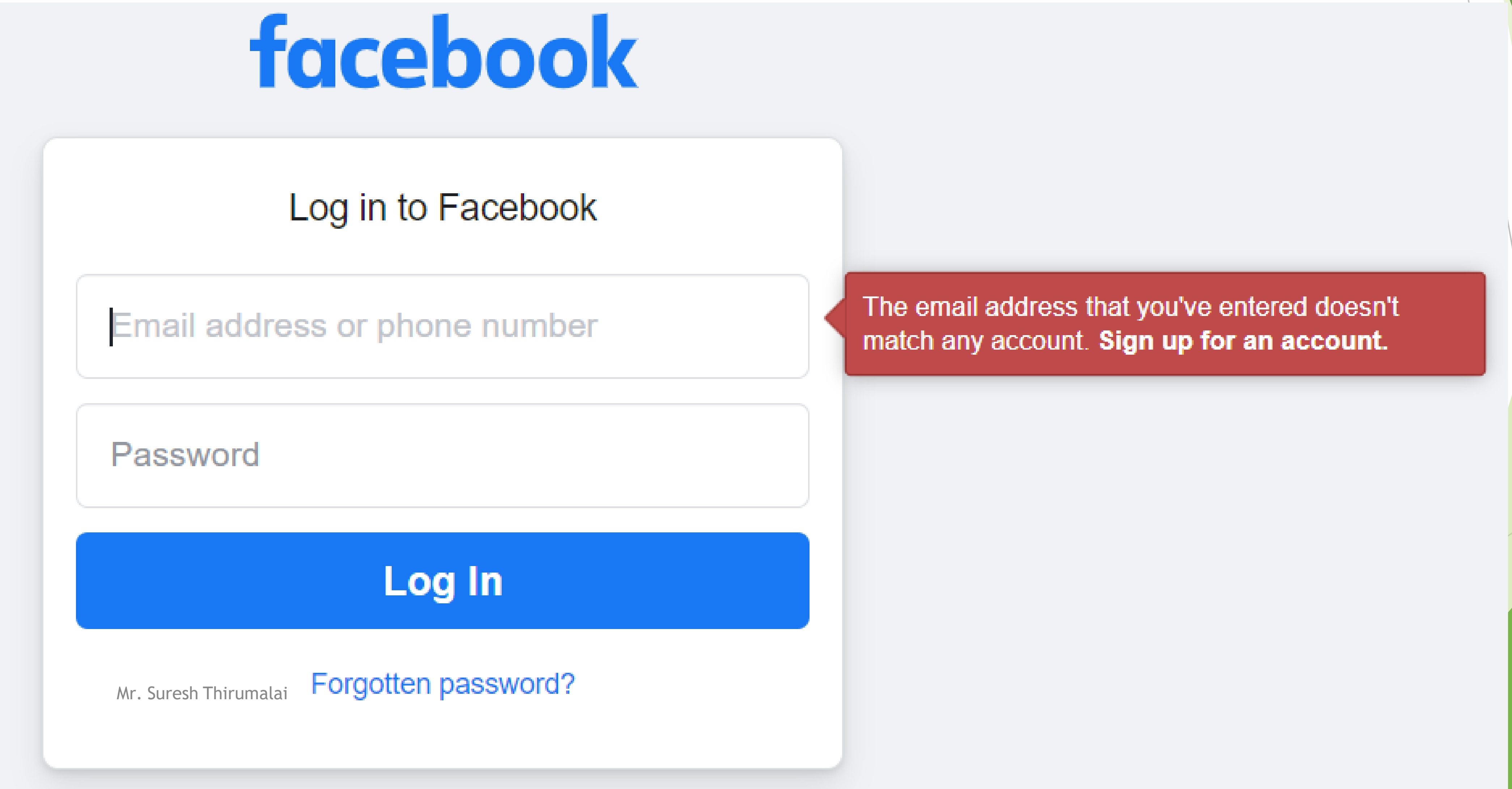
- **Test Script:**

  Test scripts are used in automated **testing**.

  Sometimes, a set of instructions (written in a human language), used in manual **testing.**

Scenario Id	Test Case Id	Test Case	Test Case Description	Test Step	Test Step Description	Expected Result	Actual Result	Status	Comment
TS001	TC001	Checking login function using valid Login and valid Username	Test the login functionality of Amazon application to make sure user is allowed to the site using valid user name and password	1	Launch the application	Application launched properly	Application launched successfully	Pass	
				2	Navigate to the login Page	Username and password fields are displayed	Username and password fields are displayed	Pass	
				3	Enter valid username in username field	Username field accepts input value as valid username	Username input accepted	Pass	
				4	Enter valid password in the password field	Password field accepts input value as valid Password	Password input accepted	Pass	
				5	Click on Login Button	Navigated to HomePage and shows name of the user in homepage	Name of the user is displayed in the homepage	Pass	

Negative Test case:


facebook

Log in to Facebook

Email address or phone number

Password

Log In

Forgotten password?

The email address that you've entered doesn't match any account. **Sign up for an account.**

Verification vs Validation

- Verification:

  The verifying process includes checking documents, design, code, and program.

  (Ex: Spelling mistake)

- **Validation**:

  It is a dynamic mechanism of Software testing and validate the actual Product.

  (Ex: **Actual** meets **Expected**)

Entry Criteria :

- Test environment availability
- Test scope definition and finalization
- Types of testing defined & finalized
- Test cycles & schedule is planned & Signed off
- test resources available
- Software under test knowledge is available
- Stakeholders are identified and involved
- Code is deployed in test environment
- Test data is available in the test environment.
- Exit criterias are signed off

Exit Criteria:

The following exit criteria should be considered for completion of a testing phase:

- Ensuring all critical Test Cases are passed
- Achieving complete Functional Coverage.
- Identifying and fixing all the high-priority defects
- Deadlines meet or budget depleted.
- Desired and sufficient coverage of the requirements and functionalities under the test.
- All the identified defects are corrected and closed.
- No high priority or severity or critical bug has been left out.
- Testing de-scoped due to any reason.
- Sign-off received from all stakeholders

Testing can be stopped when: 1

<u>Requirements</u>:

- 100% Requirements coverage is achieved.

<u>Defects</u>:

- Defined / Desired Defect count is reached.
- All Show Stopper defects or Blockers are fixed and No known Critical / Severity 1 defect is in Open Status.
- All High Priority defects are identified and fixed.
- Defect Rate falls below defined acceptable rate.
- Very few Medium Priority defects are open and have a workaround in place.
- Very few low priority open defects that do not impact software usage.
- All High Priority defects are re-tested and closed and corresponding Regression scenarios are successfully executed.

Testing can be stopped when: 2

<u>Test Coverage:</u>

- Test Coverage should be 95% achieved.
- Test case Pass Rate should be 95%. This can be calculated by formula
  - (Total No of TCs Passed / Total number of TCs) * 100.
- All critical Test cases are passed.
- 5% Test cases can be failed but the Failed Test cases are of low priority.
- Complete Functional Coverage is achieved.
- All major functional / business flows are executed successfully with various inputs and are working fine.

<u>Deadlines</u>:

- Project Deadline or Test Finish deadline is reached.

Testing can be stopped when: 3

Test Documents:

- All Test Documents / deliverables (Example – Test Summary Report) are prepared, reviewed and published across.

Budget:

- Complete Testing Budget is exhausted.

“Go / No Go” Meetings:

- “Go / No Go” meeting has been conducted with stakeholders and a decision is made whether the project should go to production or not.

Manual to Automation prerequisite

1. Build should always be stable.
2. Get the functionalities to repeat.
3. Filtering the **automated** test cases.
4. Skilled and experienced resources.
5. A module or application that does not change frequently.
6. Segregation of test cases that needs to be **automated**.
7. Use of reusable functions and procedures

Sample Test Cases

1. OTP

2. ATM

3. Add to Cart

Scenarios to Test OTP (One-Time Password)

1) OTP should be generated within time period.

2) Limitations of number of OTP generation for single authentication.

3) It is received only on registered Mobile Number / E-mail Address.

4) Network delay for expiry of One-Time Password.

5) Verify that once expired, it should not be used for any authentication.

6) Verify that once used, it should not be allowed to use again.

7) Verify that resend OTP functionality is working properly.

8) Verify that once user resent the OTP, the old one should be of no use.

9) Availability of Help and Documentation Link for OTP usage.

10) Verify for Case Sensitiveness.

11) Check for types of characters OTP supports: Only Digits, Only Alphabets, Alphanumeric.

12) How many times user can provide invalid OTP?

13) After multiple invalid try, verify that system temporarily blocks the account.

14) Verify that after temporary blocking of account, system does not send the one-time password.

15) Provide an invalid Phone Number or E-Mail address and submit the OTP. Check the validation.

16) Are the one-time password patterns are predictable?

Test Cases for ATM

Given below are the various test cases for ATM.

1) Verify if the card reader is working correctly. A screen should ask you to insert the pin after inserting the valid card.

2) Verify if the cash dispenser is working as expected.

3) Verify if the receipt printer is working correctly. Which means it can print the data on the paper and the paper comes out properly.

4) Verify if the Screen buttons are working correctly. **For touch screen:** Verify if it is operational and working as per the expectations.

5) Verify if the text on the screen button is visible clearly.

6) Verify the font of the text on the screen buttons.

7) Verify each number button on the Keypad.

8) Verify the functionality of the Cancel button on the Keypad.

9) Verify the text color of the keypad buttons. The numbers should be visible clearly.

10) Verify the text color and font of the data on the screen. The user should be able to read it clearly.

11) Verify the language selection option. If the messages or data are displayed in the selected language.

12) Insert the card, the correct pin, and print the receipt for available balance.

13) Verify the receipt printing functionality after a valid transaction. Whether the printed data is correct or not.

14) Verify how much time the system takes to log out.

15) Verify the timeout session functionality.

16) Verify the deposit slot functionality depending on its capability (Cash or cheque or both) by inserting a valid cheque.

17) Verify using different cards (Cards of different banks).

Verifying the Message

18) Insert the card and an incorrect PIN to verify the message.

19) Verify the message when there is no cash in the ATM.

20) Verify the messages after a transaction.

21) Verify if a user will get a correct message if a card is inserted incorrectly.

Messages for each and every scenario should be verified.

Cash Withdrawal

22) Verify the cash withdrawal functionality by inserting some valid amount.

23) Verify if a user can perform only one cash withdrawal transaction per PIN insert.

24) Verify the different combinations of operation and check if there will be a power loss in the middle of the operation.

Negative Test cases

25) Verify the functionality by entering a wrong pin number for 3 or more times.

26) Verify the card reader functionality by inserting an expired card.

27) Verify the deposit slot functionality by inserting an invalid cheque.

28) Verify the cash withdrawal functionality by inserting invalid numbers like 10, 20, 50 etc.

29) Verify the cash withdrawal functionality by entering an amount greater than the per day limit,

30) Verify the cash withdrawal functionality by entering an amount greater than per transaction limit.

31) Verify the cash withdrawal functionality by entering an amount greater than the available balance in the account.

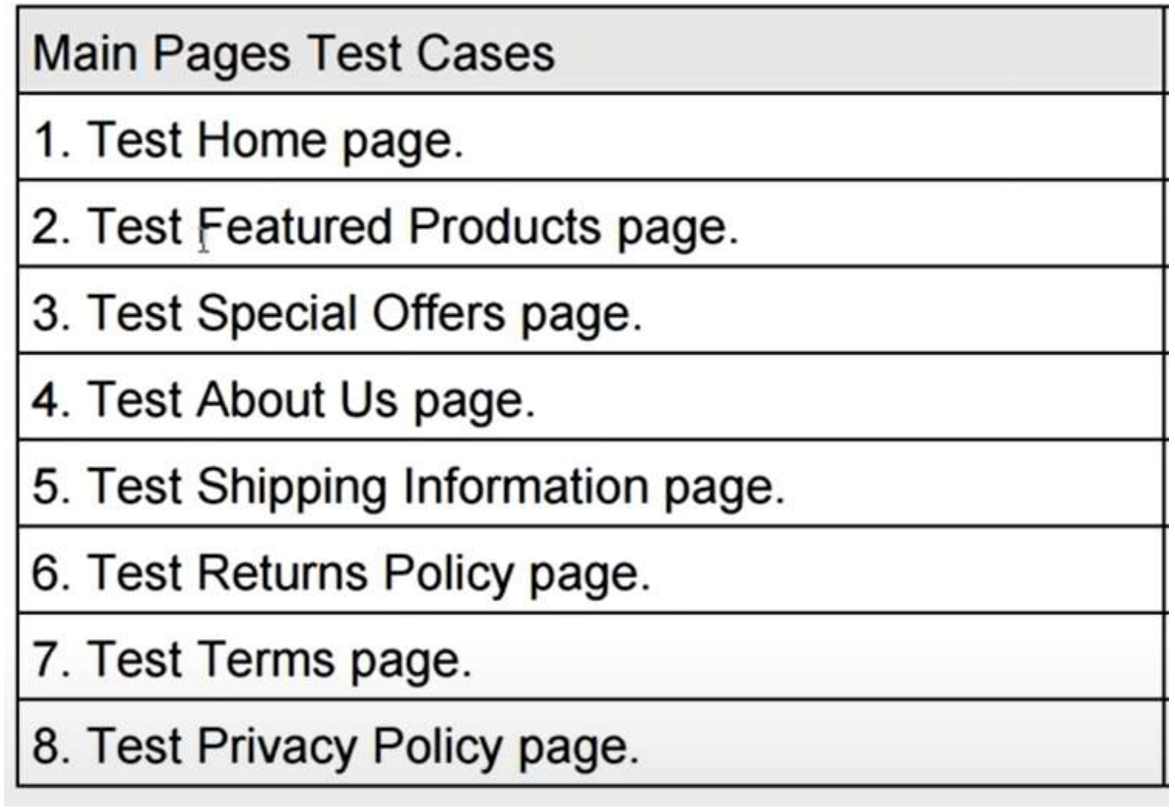
E-commerce Websites: Shopping cart

- Add one item to the cart and verify.
- Increase the quantity of the item from the cart and verify.
- Add the same item multiple times and verify.
- Add multiple items of different types and verify.
- Remove some items from the cart and verify.
- Remove all items from the cart and then verify.
- Click on an item in the cart and verify that the user is redirected to the product detail page.
- Check that the price of the cart is discounted when we apply a valid coupon.
- Check that the price of the cart is not discounted when we apply an invalid coupon.
- Add item(s) to the cart, close the browser and reopen the same site.
- Add item(s) to the cart, close the browser and reopen the same site.
- Verify the product QTY field when the product is out of stock.
- Verify that the user is able to add a text note for all products.
- Verify that the user is able to add any or all products to his wishlist by clicking on the wishlist link.
- Verify that the user is able to mark his order as a gift.
- Verify that the user is able to add any a gift message.
- Verify that the user is redirected to the checkout page after clicking on the checkout button.
- Verify the cart total when the exclusive tax is enabled from the admin end.
- Verify the cart total when the inclusive tax is enabled from the admin end.

Checkout flow

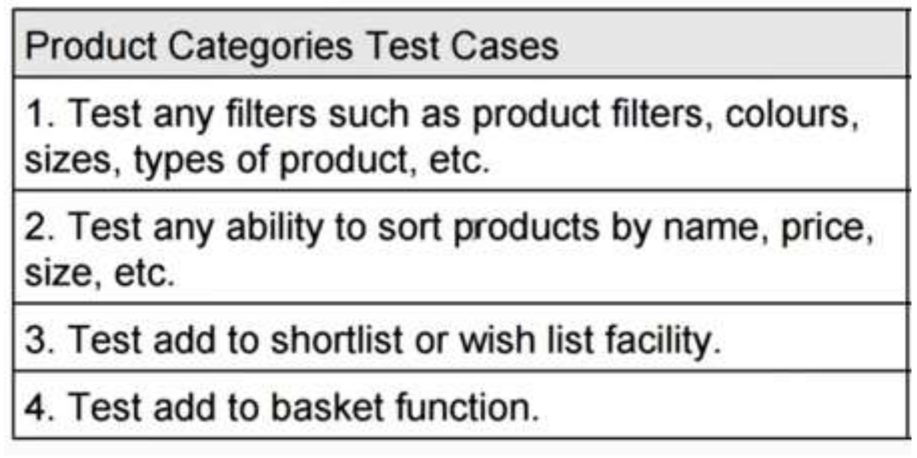
- Ensure that user can access the Checkout Page only after adding the product to the cart.
- Ensure that Checkout Address Page consists of all the details of the product such as Name, Quantity, Amount, etc.
- Ensure that only registered users are allowed to access the Checkout Address Page.
- Ensure that Name, Street Address, City, State, Country, Postal code is the mandatory field in the Checkout Address page (Shipping Address).
- Ensure that Name, Street Address, City, State, Country, Postal code is the mandatory field in the Checkout Address page (Billing Address).
- Ensure that error message is displayed when the user enters invalid input in all the mandatory field in the checkout address page.
- Ensure that Back to cart link is available so that user can modify the cart content if needed.
- Ensure that user is redirected to Checkout payment page only after entering valid mandatory details in the checkout Address page.
- Ensure that error message is displayed when the user leaves any one of the mandatory fields in the checkout Address page.
- Ensure that Next Button and Cancel Button are available in the Checkout Address Page.
- Check that a user is able to apply a valid coupon.
- Verify the error message for the invalid coupon.
- Check that user is able to make his order as a gift.
- Ensure that the user is able to add any gift message.
- Check order price when gift and wrapping charges are applicable.
- Check the shipping price.
- Check the shipping price tax.
- Check the payment options.
- Ensure that Name on Card, Card Number, Expiration date, CWW2 are mandatory fields in the Payment Page.
- Ensure that error message is displayed when the user enters invalid input in all the mandatory field on the Payment Page.
- Ensure that error message is displayed when the user leaves any one of the mandatory fields in the Payment Page.
- Ensure that the user is redirected to the third party (Vender) page.
- Ensure that the user is redirected to the third party (PayPal) page if he is going to pay through a third party payment gateway (PayPal).
- Place an order where order total price is Zero (0).
- Place an order where the total discount is more than the entire order.
- Place an order by double clicking on the submit Button.

Modules for E-Commerce site

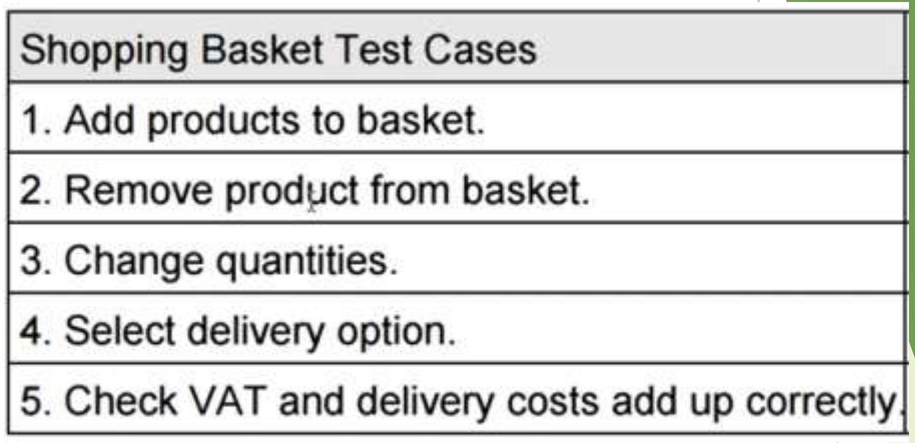


Main Pages Test Cases
1. Test Home page.
2. Test Featured Products page.
3. Test Special Offers page.
4. Test About Us page.
5. Test Shipping Information page.
6. Test Returns Policy page.
7. Test Terms page.
8. Test Privacy Policy page.

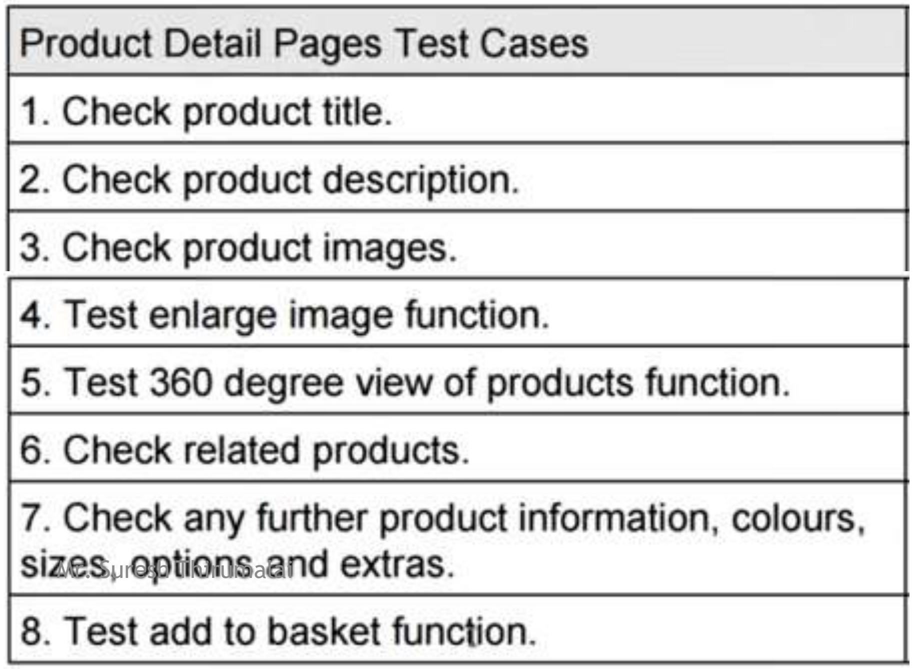
Test Cases



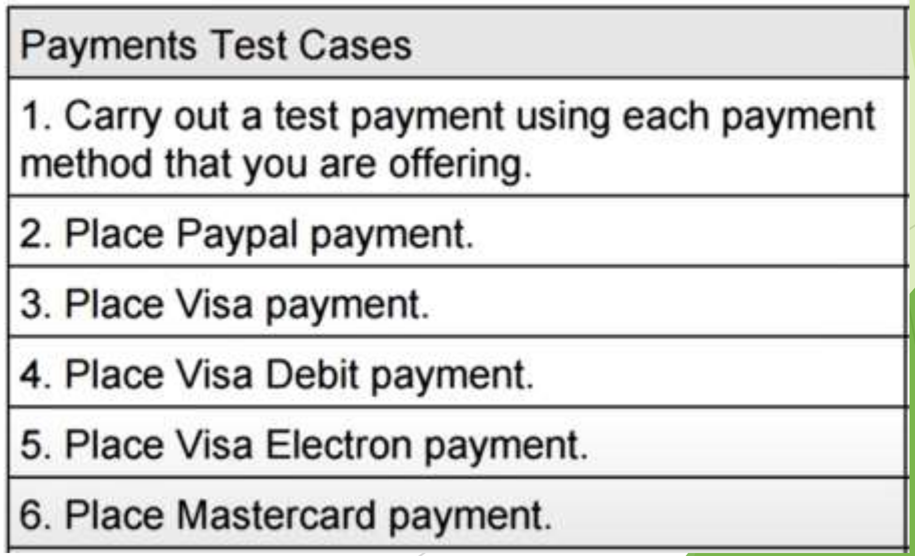
Product Categories Test Cases
1. Test any filters such as product filters, colours, sizes, types of product, etc.
2. Test any ability to sort products by name, price, size, etc.
3. Test add to shortlist or wish list facility.
4. Test add to basket function.



Shopping Basket Test Cases
1. Add products to basket.
2. Remove product from basket.
3. Change quantities.
4. Select delivery option.
5. Check VAT and delivery costs add up correctly.

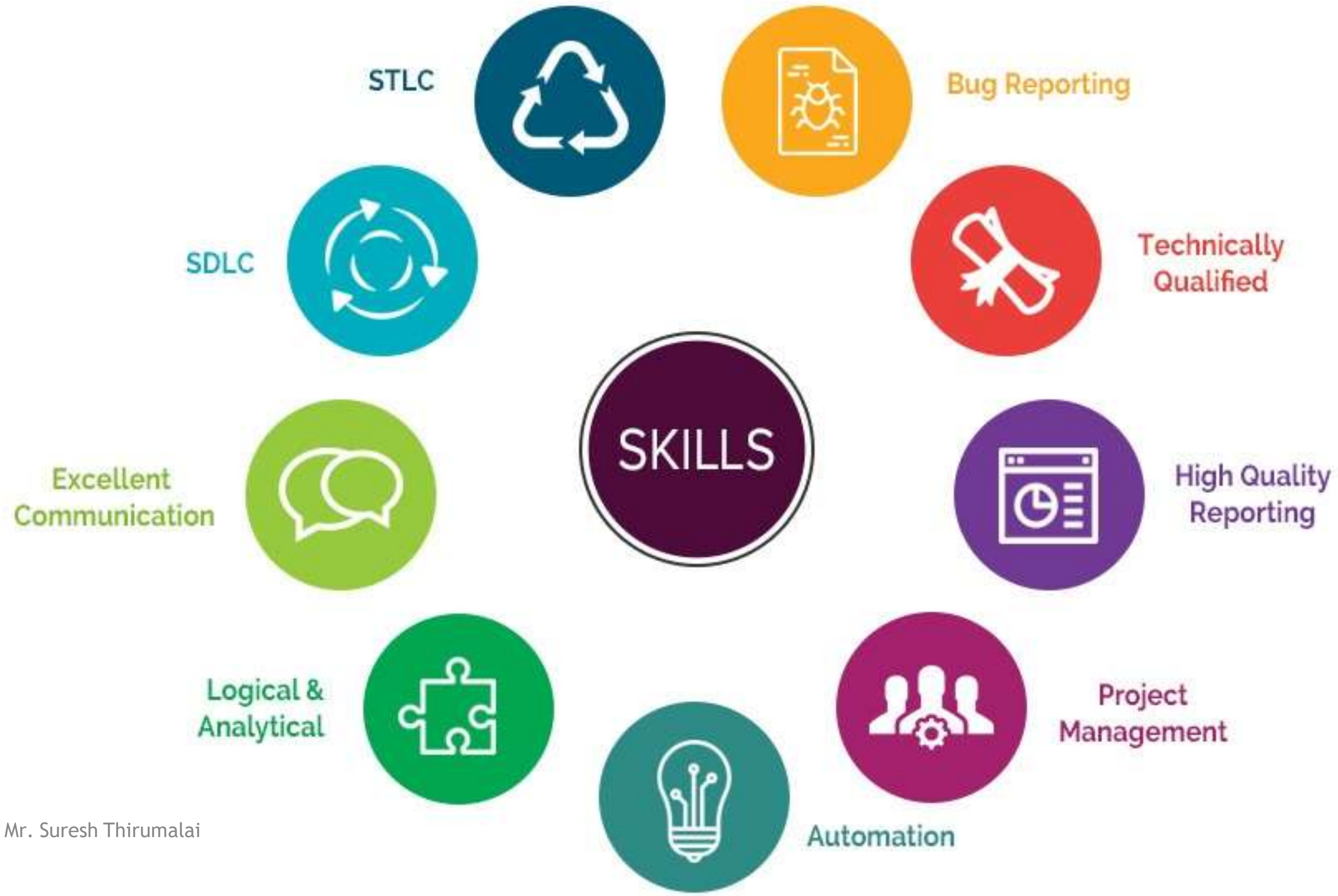


Product Detail Pages Test Cases
1. Check product title.
2. Check product description.
3. Check product images.
4. Test enlarge image function.
5. Test 360 degree view of products function.
6. Check related products.
7. Check any further product information, colours, sizes, options and extras.
8. Test add to basket function.

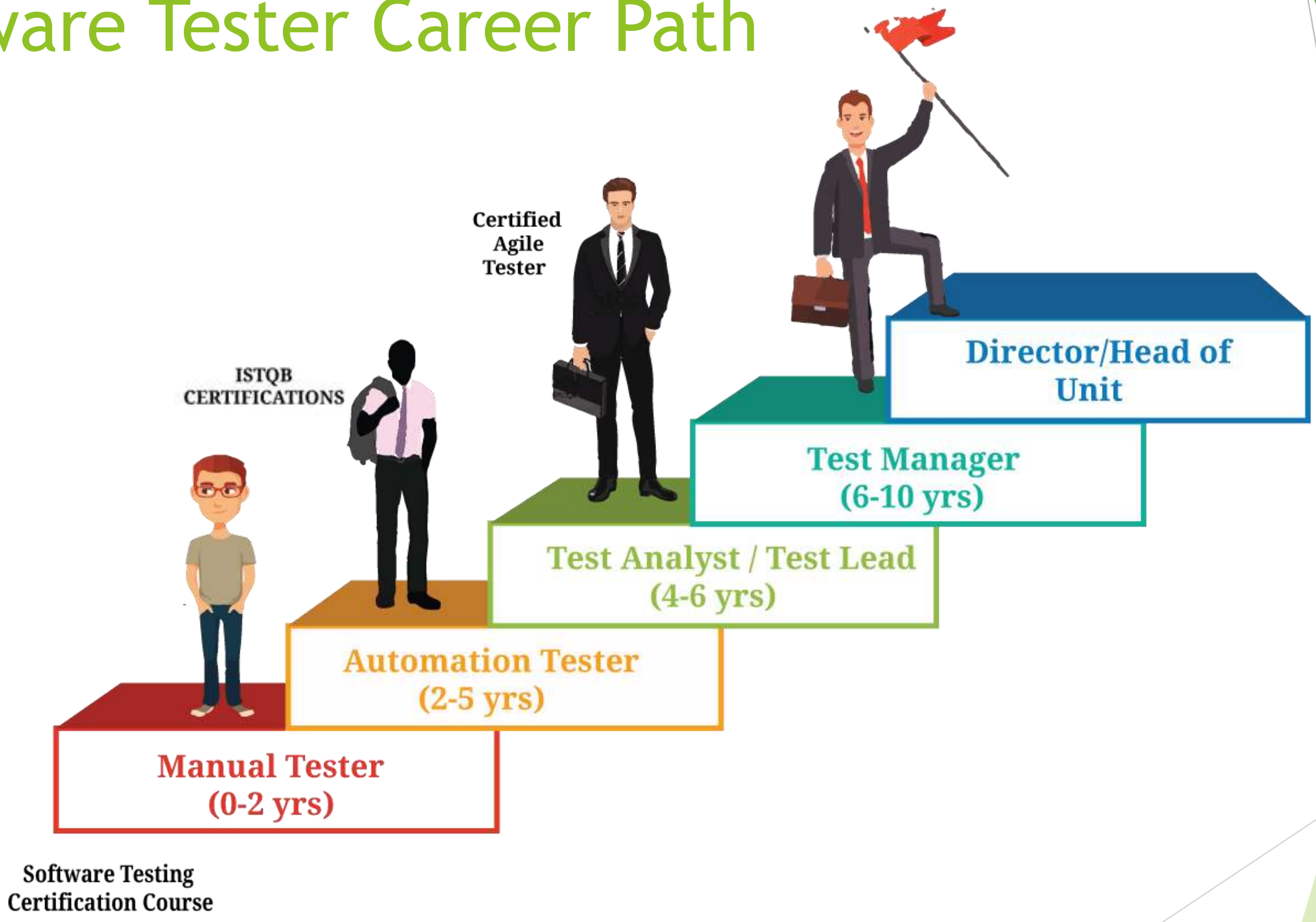


Payments Test Cases
1. Carry out a test payment using each payment method that you are offering.
2. Place Paypal payment.
3. Place Visa payment.
4. Place Visa Debit payment.
5. Place Visa Electron payment.
6. Place Mastercard payment.

Skills required to become a Software Tester



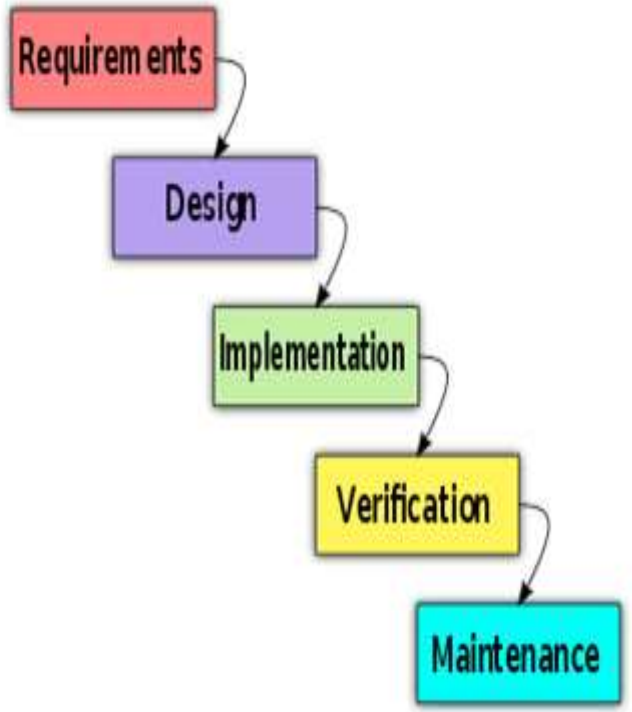
Software Tester Career Path



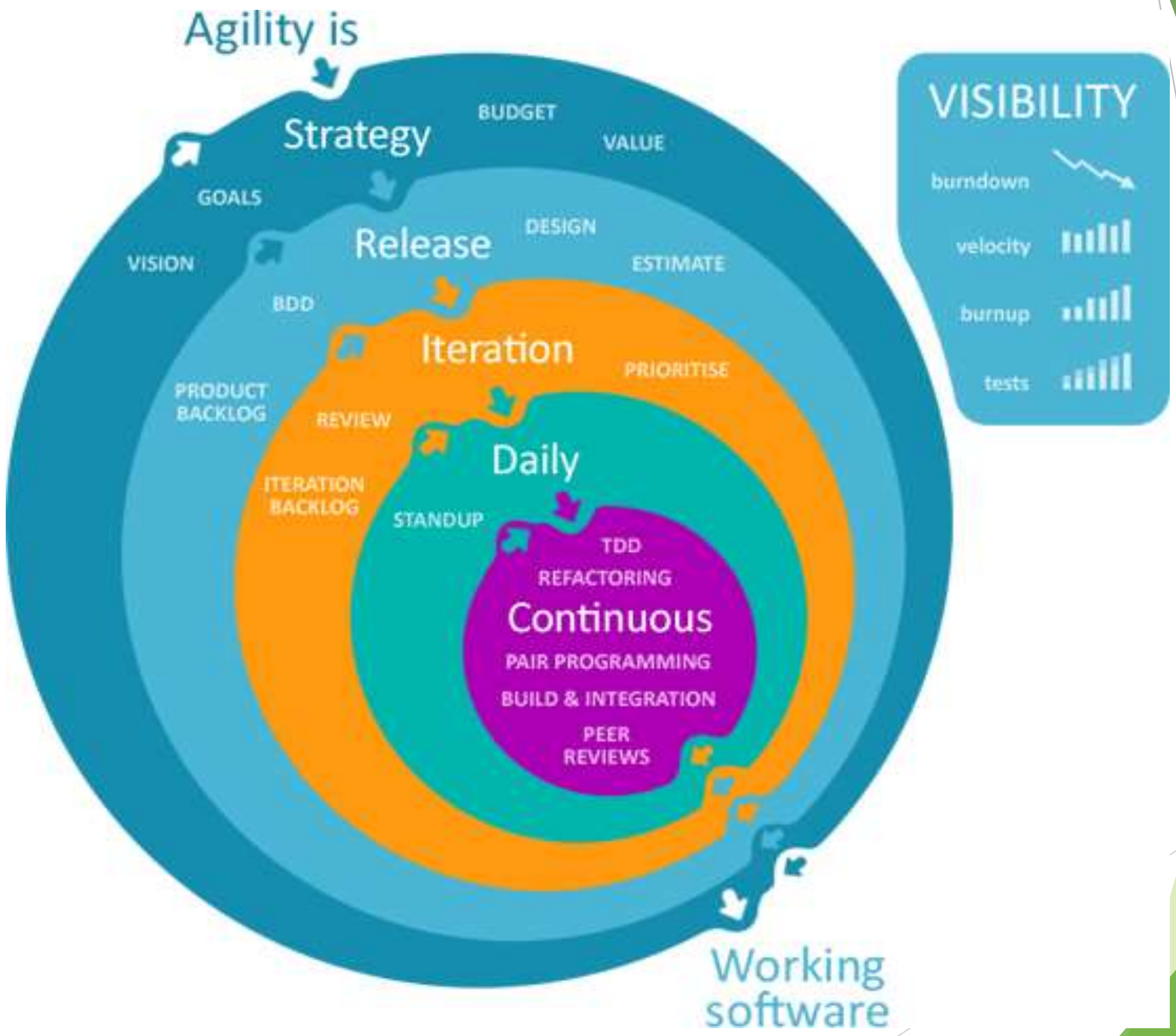
Mr. Suresh Thirumalai

Software Methodologies

Waterfall Model

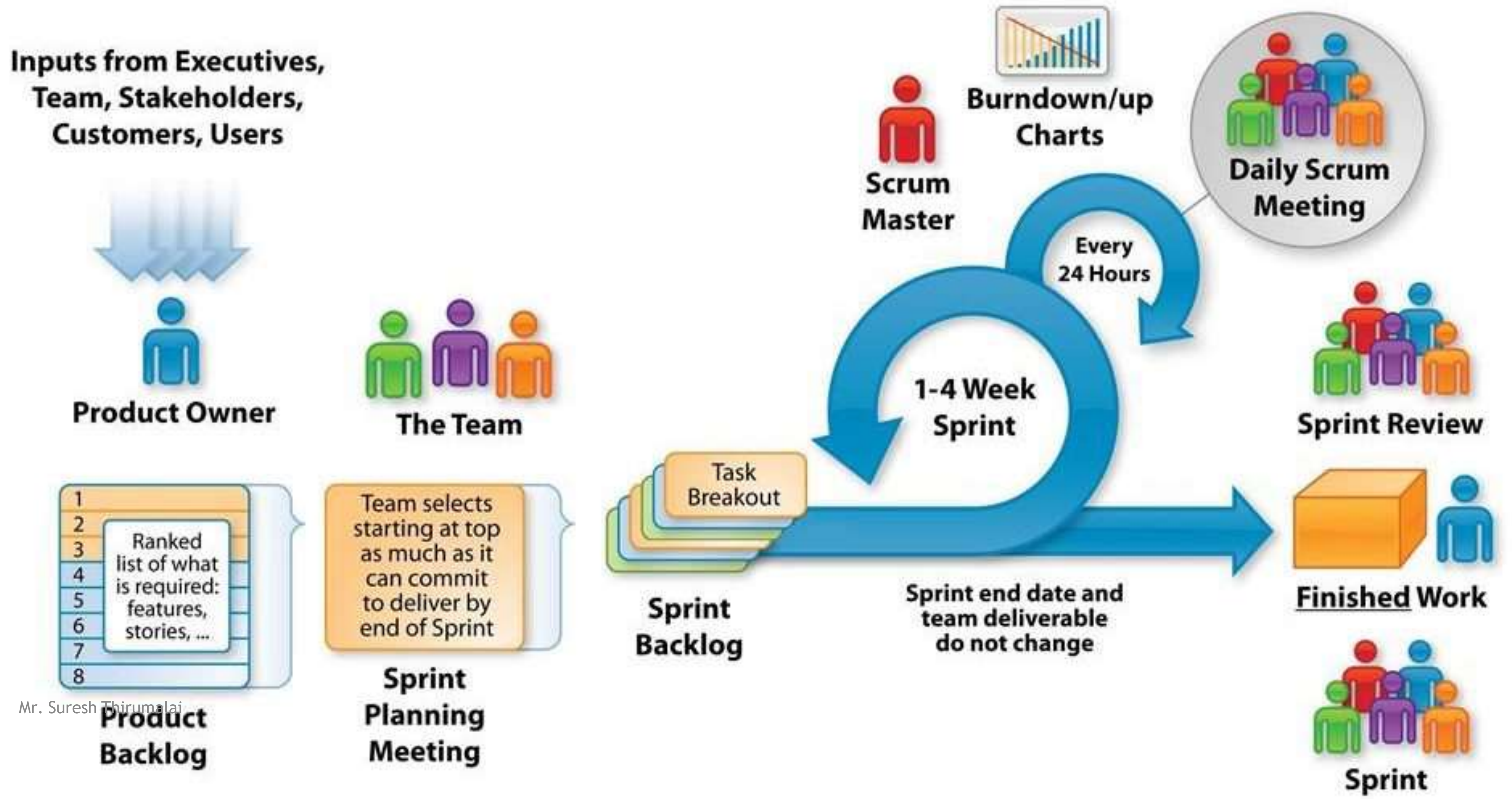


Agile Methodology



Mr. Suresh Thirumalai

Agile - Scrum Methodology



Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint end date and team deliverable do not change
Sprint Review
Finished Work
Sprint

Mr. Suresh Thirumalai

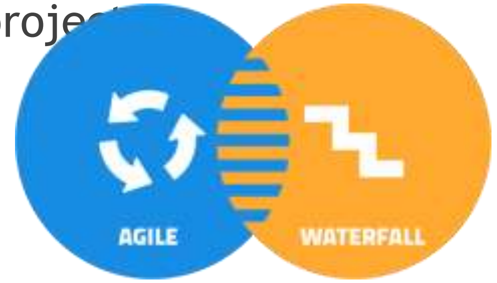
19. Agile

- It is a **practice** that promotes **continuous iteration** of development and testing throughout the software development lifecycle of project.
- Both Dev and testing activities are concurrent unlike Waterfall model.

Agile emphasizes on 4 core values:

- Individual and team interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile	Waterfall
Incremental and iterative approach	Sequential from start to end point
It is broken into individual models	Not broken into individual
Customer see the product frequently - can make the changes to the proje	Customer can only see the product at end of the project
Error can be fixed middle of the project itself	Whole product is tested in end. If error is found, the project has to start from the beginning
Regression testing is implemented in Every Iteration as new functions or logic are released	Only after the development phase, the testing phase is executed
Testers and developers work together	Testers and developers work separately
It requires close communication with developers and tester together to analyse requirements and planning	Developers does not involve in requirement and planning process. Usually, time delays between tests and coding.



Agile

- At the end of every sprint, UAT is performed
- When an iteration end, Shippable features of the product is delivered to the customer.

  New features are usable right after shipment

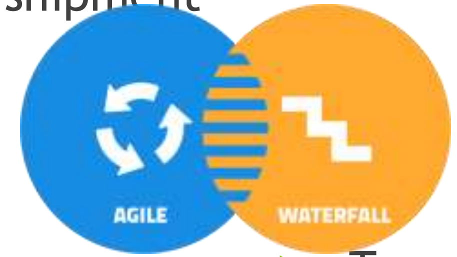
  It is useful when you have good

Contact with customers

- Unstructured
- Small projects can be implemented quickly. Large projects is difficult to estimate time
- Documentation attends less priority than software development

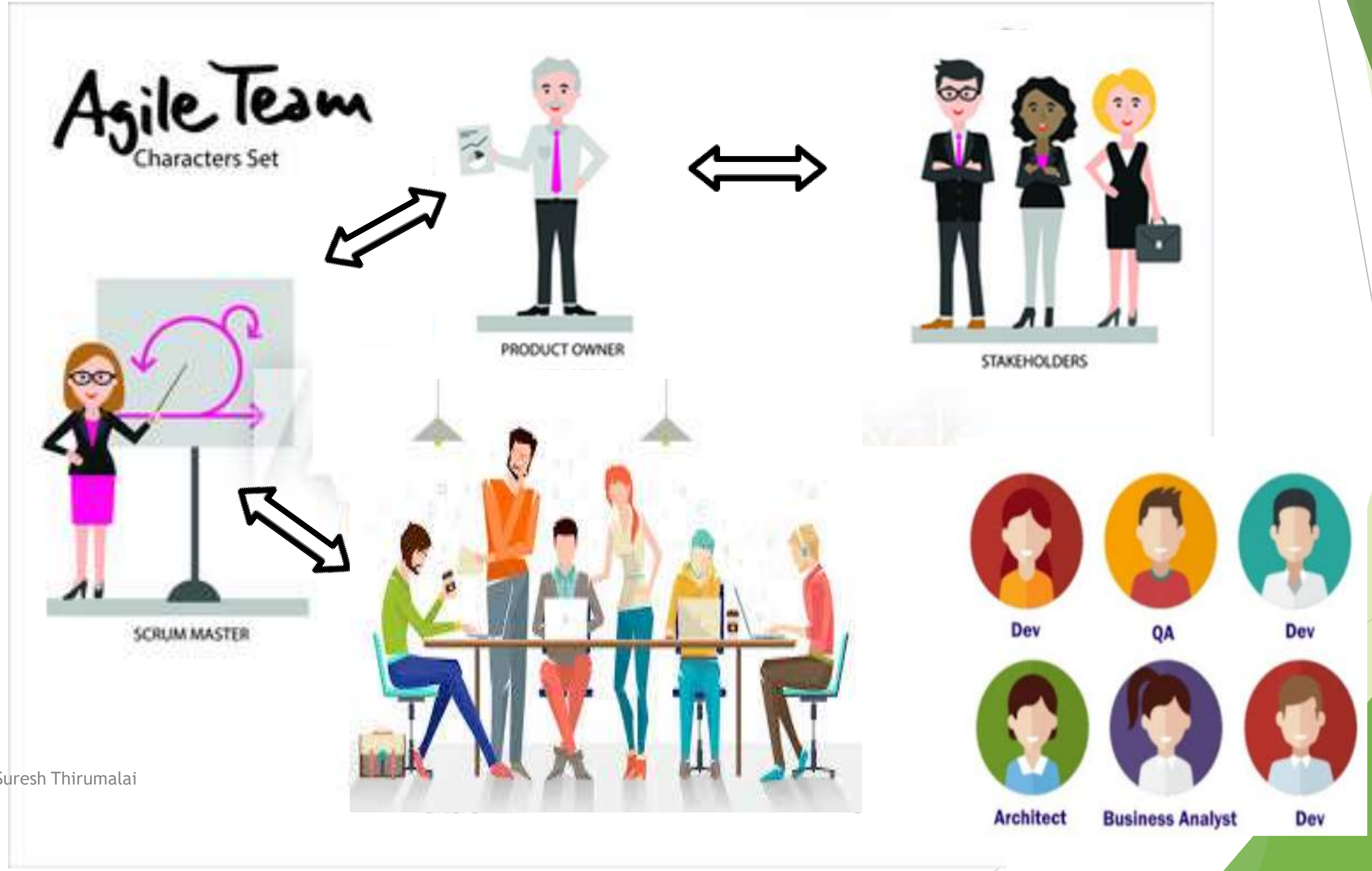
Waterfall

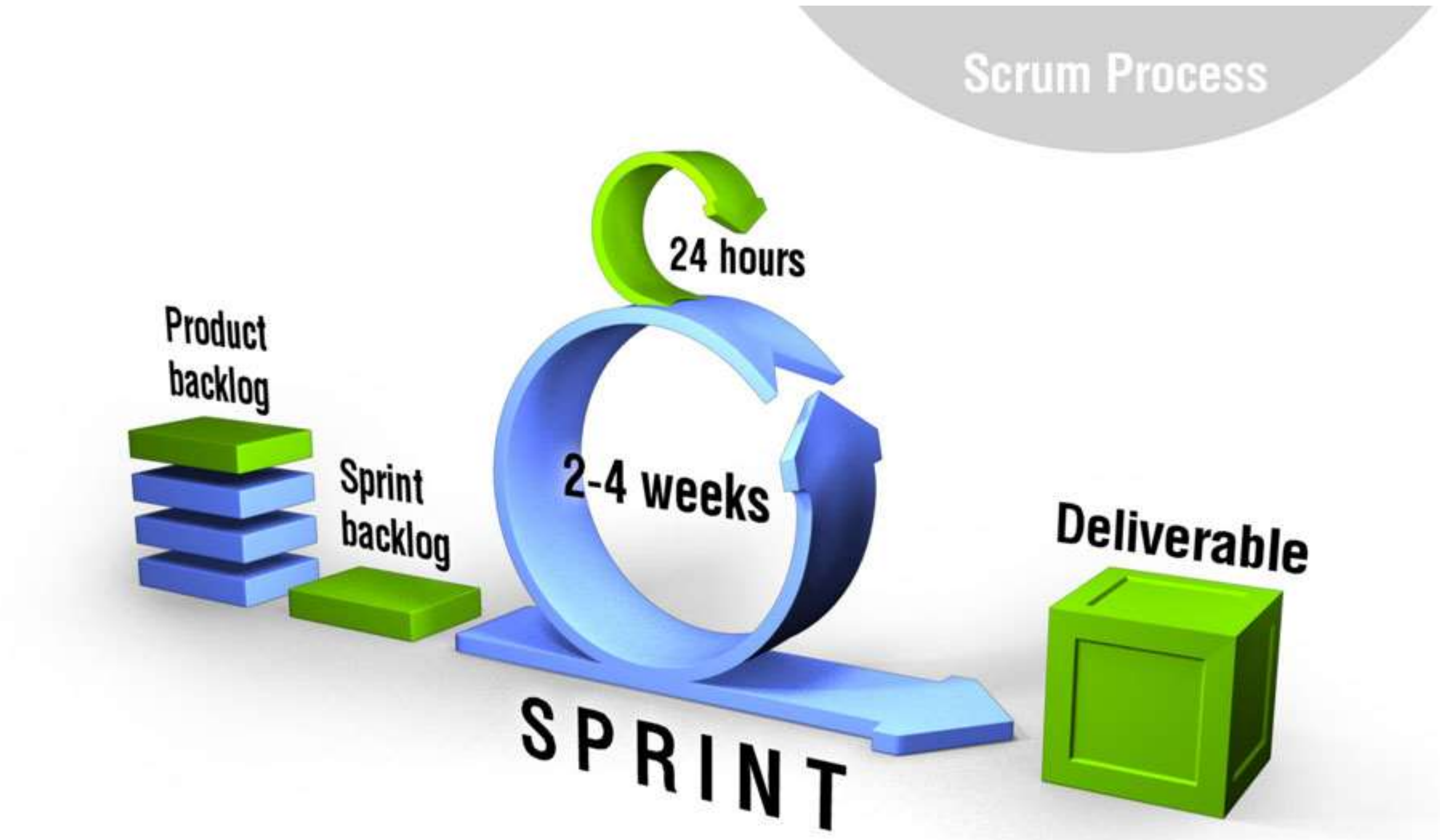
- At the end of the project only UAT is performed
- All features developed are delivered at once after the long implementation phase.



- Too planned and secured
- All sorts of project can be estimated and completed
- Documentation is a top priority and can even use for upgrade the software with another team

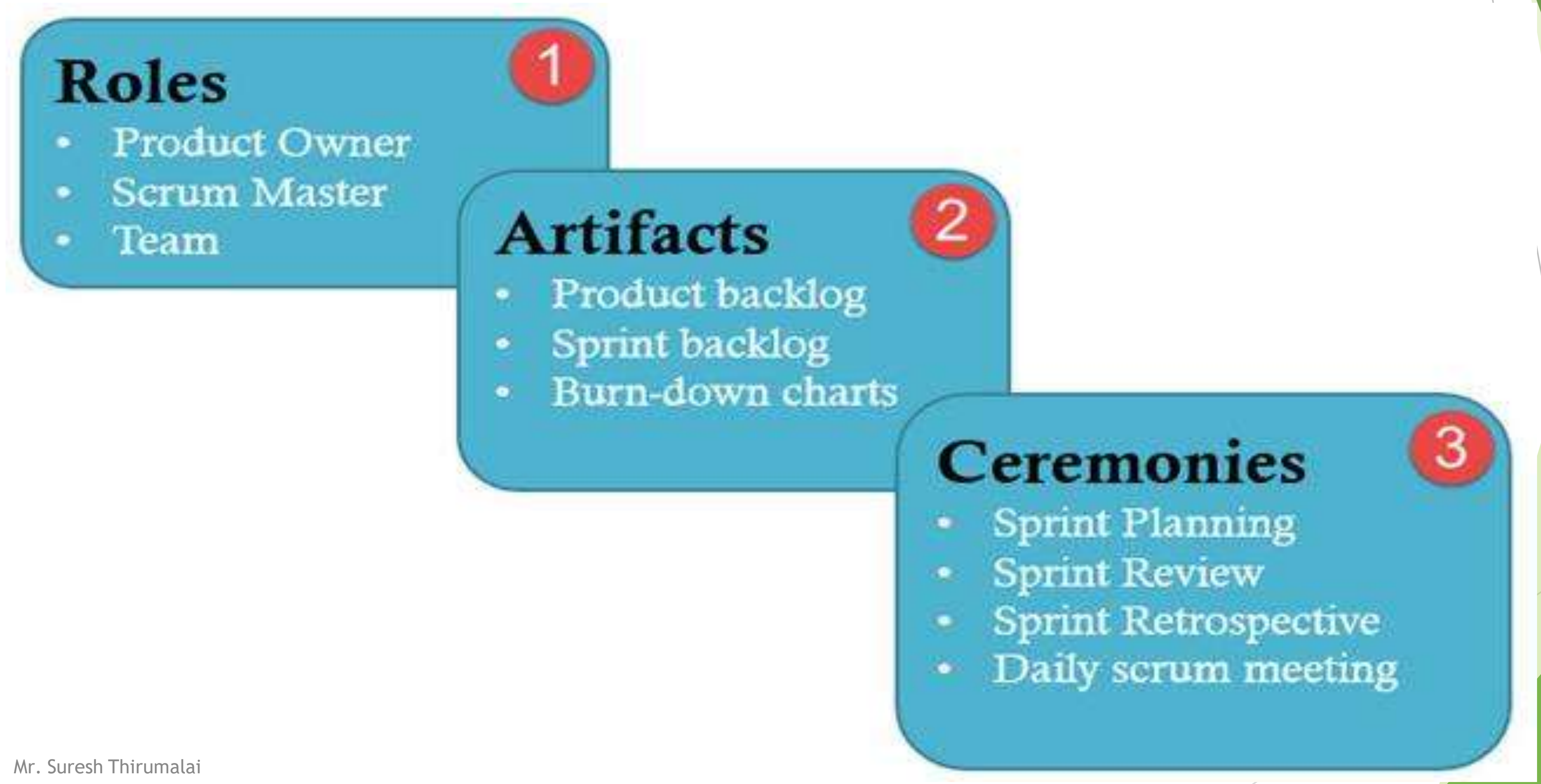
Scrum Framework





Scrum is an agile development method

1. Roles



Roles 1
- Product Owner
- Scrum Master
- Team

Artifacts 2
- Product backlog
- Sprint backlog
- Burn-down charts

Ceremonies 3
- Sprint Planning
- Sprint Review
- Sprint Retrospective
- Daily scrum meeting

Role: 1
Scrum master

Responsibilities

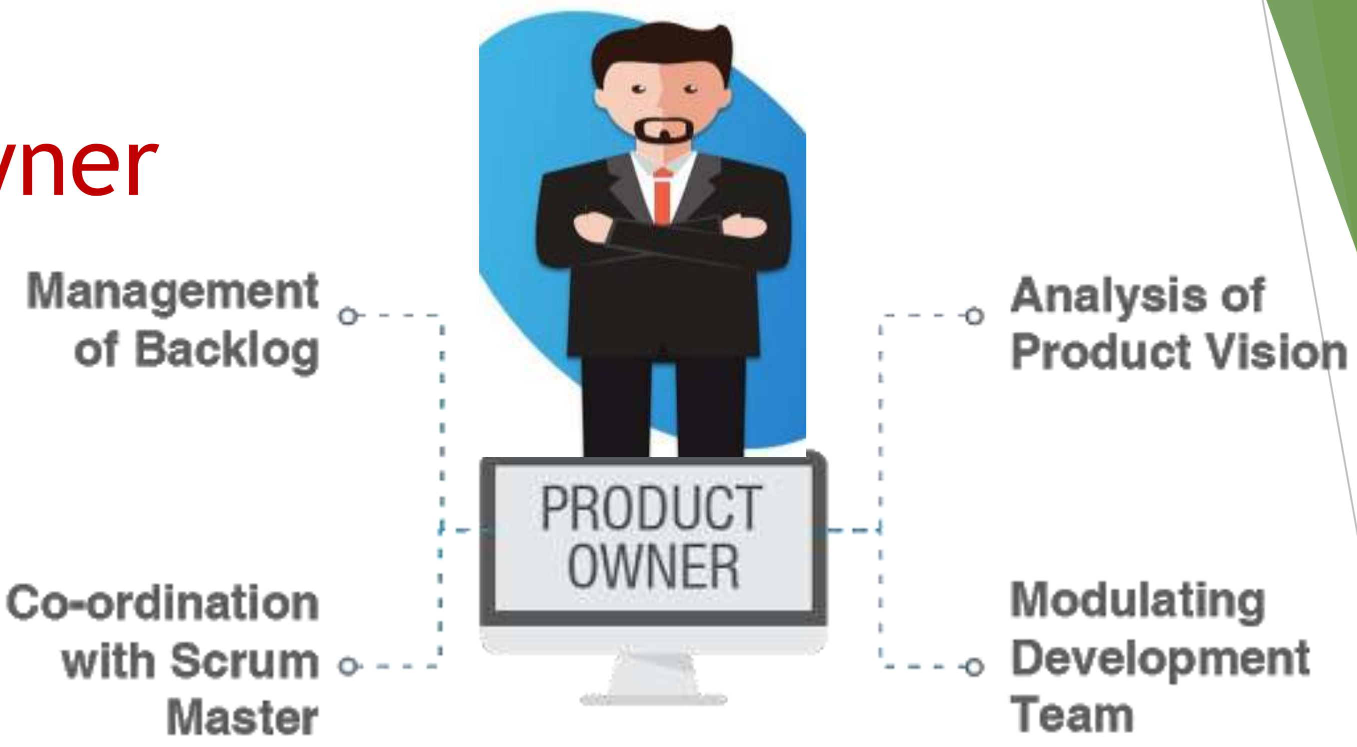
- 1. He setting up the team and look after the team's productivity
- 2. Invites to the daily scrum, sprint meetings
- 3. Removing obstacles to progress
- 4. Coordinates with all roles and functions

Owns the process
Protects team
Not the boss
Facilitator

Mr. Suresh Thirumalai

Role: 2

Product Owner


Management of Backlog
Co-ordination with Scrum Master
PRODUCT OWNER
Analysis of Product Vision
Modulating Development Team

Responsibilities

- 1. He defines features of the Product
- 2. Creates product backlog
- 3. Decides the release date and corresponding features
- 4. Prioritizes the backlog (features) according to the market value
- 5. Getting delivery of the functionality at each iteration from scrum team

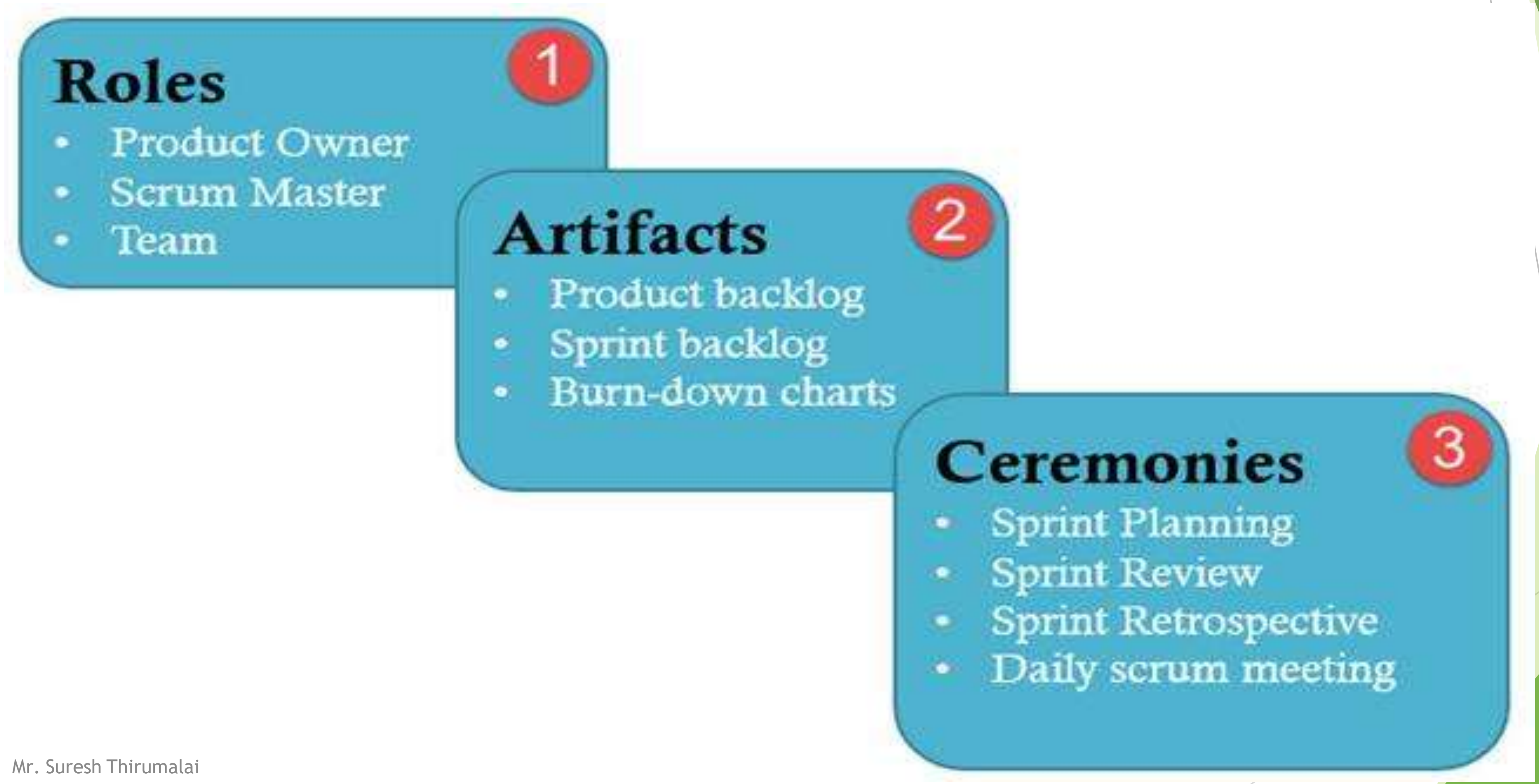
Role: 3

Scrum Team

Responsibilities

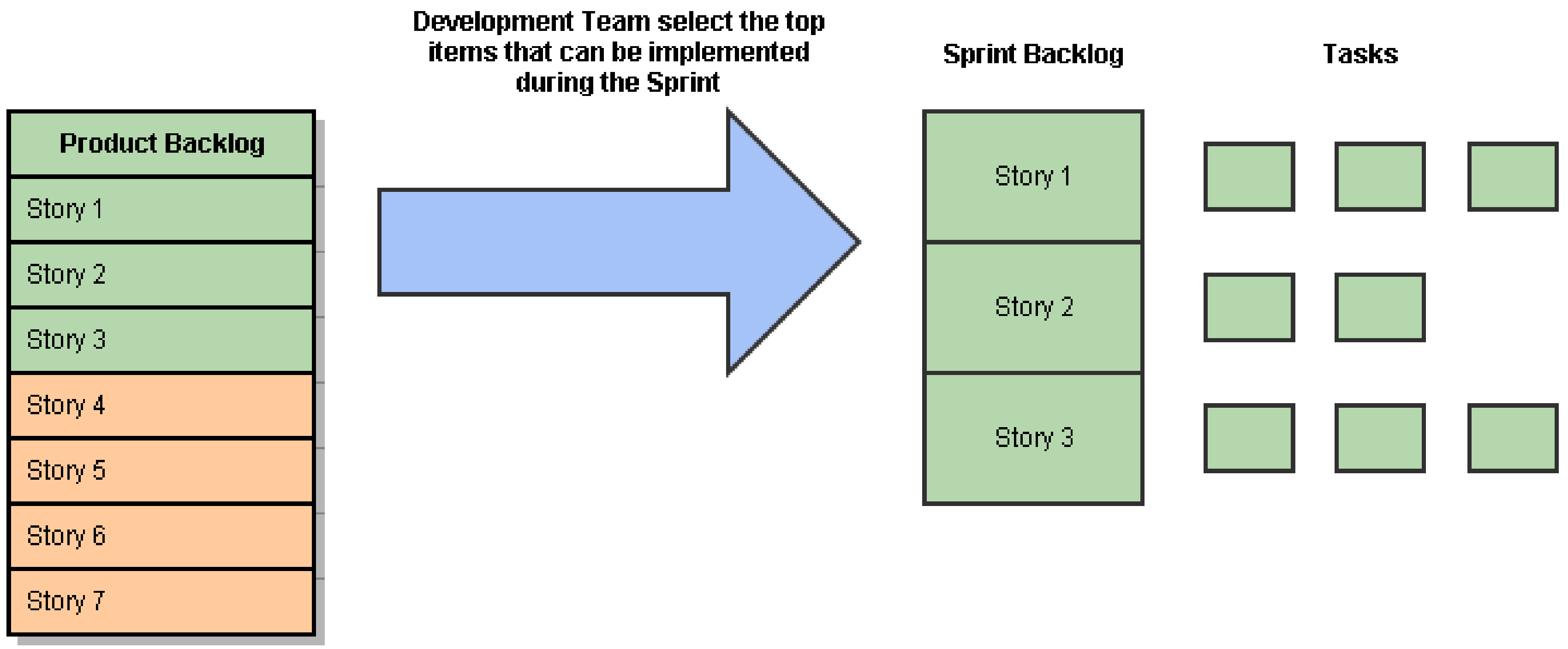
- 1. Manages its own work
- 2. Organizes the work to complete the sprint or cycle

2. Artifacts


Roles
1
Product Owner
Scrum Master
Team
Artifacts
2
Product backlog
Sprint backlog
Burn-down charts
Ceremonies
3
Sprint Planning
Sprint Review
Sprint Retrospective
Daily scrum meeting

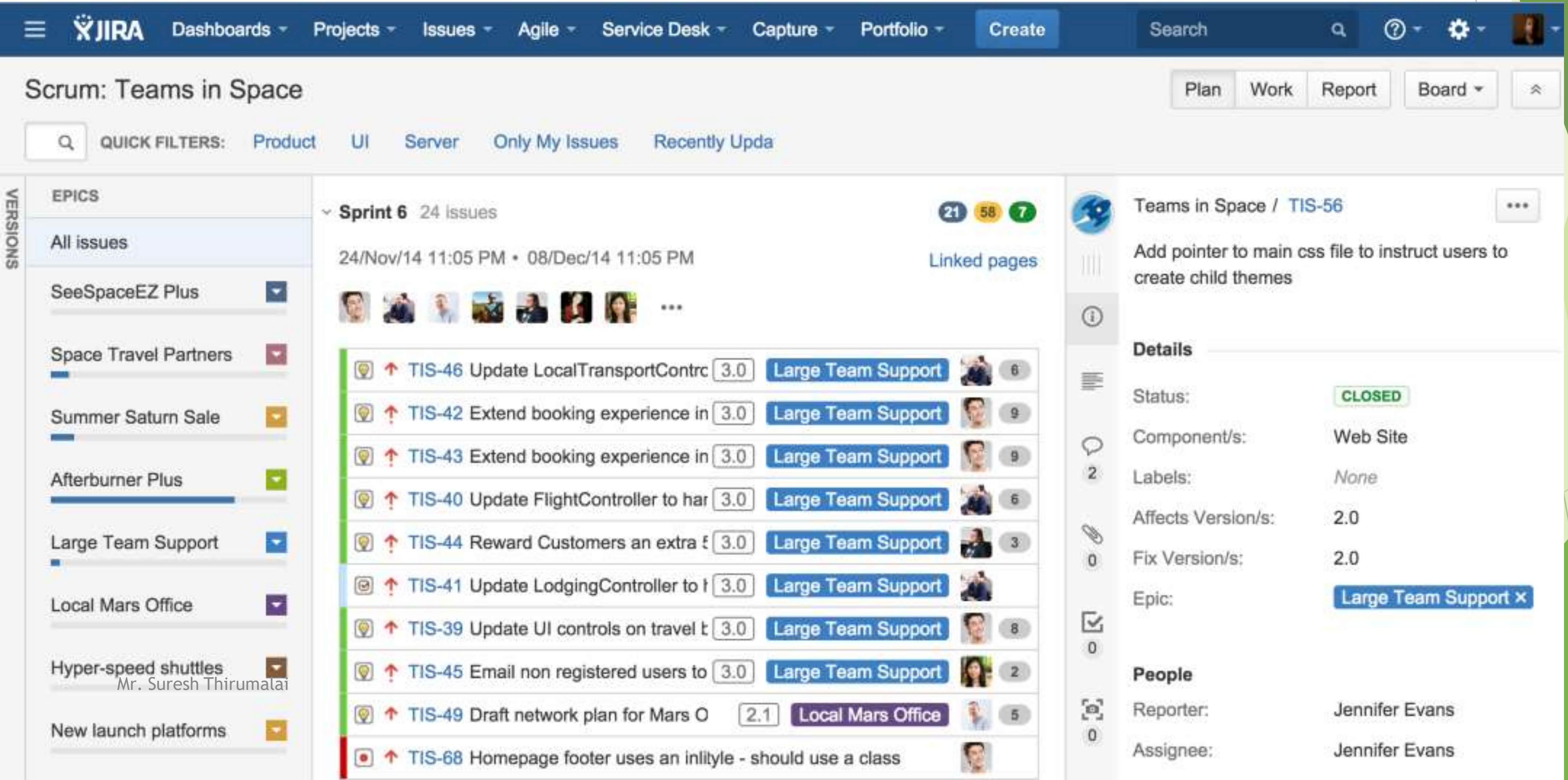
Sprint

- It is a set period of time to complete the user stories, decided by the product owner and developer team, usually 2-4 weeks of time.



User stories

- They are short explanation of functionalities of the system under test



JIRA Dashboards Projects Issues Agile Service Desk Capture Portfolio Create Search

Scrum: Teams in Space

Plan Work Report Board

QUICK FILTERS: Product UI Server Only My Issues Recently Upda

VERSIONS

EPICS

All issues

SeeSpaceEZ Plus

Space Travel Partners

Summer Saturn Sale

Afterburner Plus

Large Team Support

Local Mars Office

Hyper-speed shuttles

New launch platforms

Sprint 6 24 issues 21 58 7

24/Nov/14 11:05 PM • 08/Dec/14 11:05 PM Linked pages

TIS-46	Update LocalTransportContrc	3.0	Large Team Support	6
TIS-42	Extend booking experience in	3.0	Large Team Support	9
TIS-43	Extend booking experience in	3.0	Large Team Support	9
TIS-40	Update FlightController to har	3.0	Large Team Support	6
TIS-44	Reward Customers an extra !	3.0	Large Team Support	3
TIS-41	Update LodgingController to I	3.0	Large Team Support	
TIS-39	Update UI controls on travel t	3.0	Large Team Support	8
TIS-45	Email non registered users to	3.0	Large Team Support	2
TIS-49	Draft network plan for Mars O	2.1	Local Mars Office	5
TIS-68	Homepage footer uses an inlityle - should use a class			

Teams in Space / TIS-56

Add pointer to main css file to instruct users to create child themes

Details

Status:	CLOSED
Component/s:	Web Site
Labels:	None
Affects Version/s:	2.0
Fix Version/s:	2.0
Epic:	Large Team Support

People

Reporter:	Jennifer Evans
Assignee:	Jennifer Evans

1. Product Backlog

- This is a repository where requirements are tracked with details on the no of requirements (user stories) to be completed for each release.
- It should be maintained and prioritized by Product Owner, and it should be distributed to the scrum team.

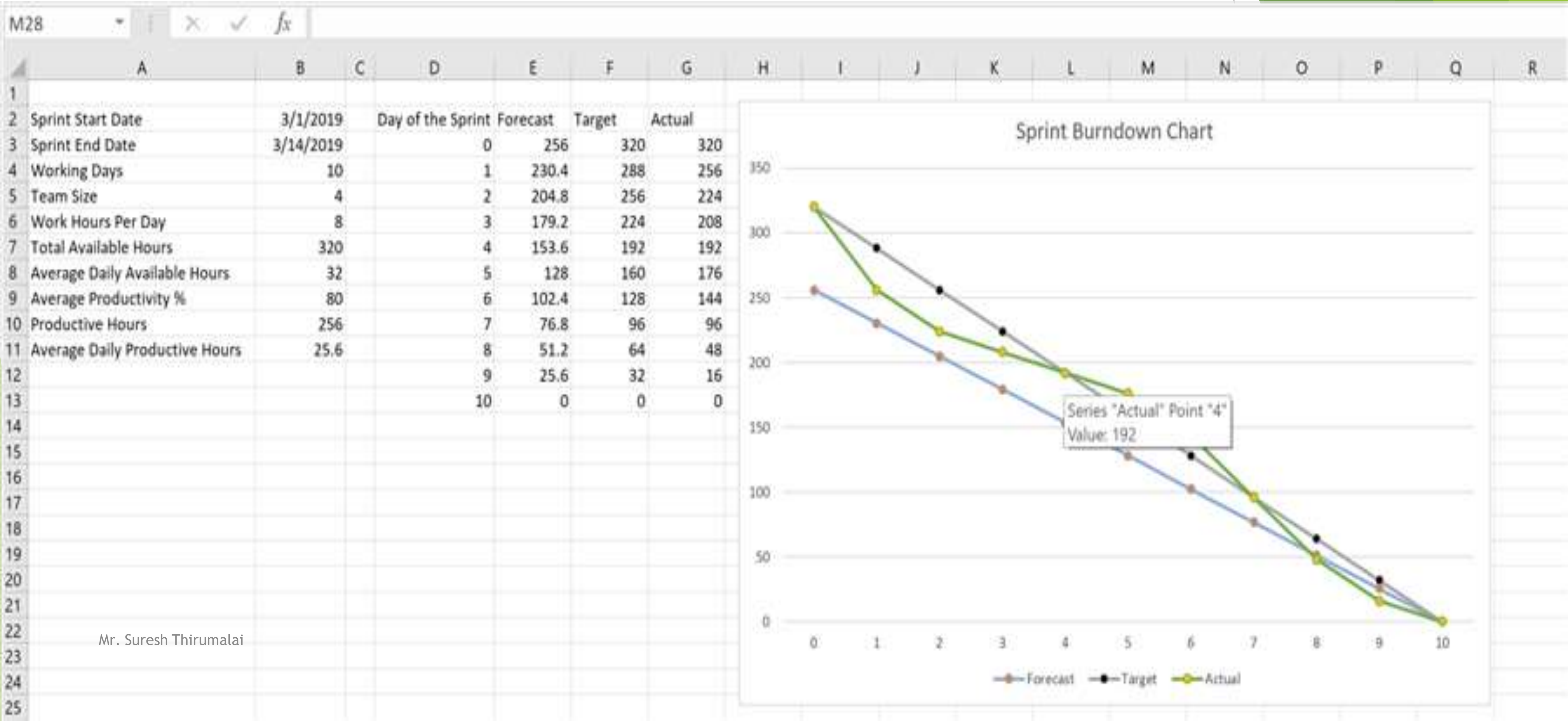
2. Sprint backlog

- It is a set of user stories to be completed in a sprint
- During the sprint backlog, work is never assigned, and the team signs up for work on their own.
- It is owned the estimated work remaining is updated daily.
- It is the list of task that has to be performed in sprint

3. Burndown chart

- A **burndown chart** is a graphic representation of how quickly the team is working through a customer's user stories, an agile tool that is used to capture a description of a feature from an end-user perspective.
- The **burndown chart** shows the total effort against the amount of work for each iteration

How to create Burndown chart



M28

Sprint Start Date	3/1/2019
Sprint End Date	3/14/2019
Working Days	10
Team Size	4
Work Hours Per Day	8
Total Available Hours	320
Average Daily Available Hours	32
Average Productivity %	80
Productive Hours	256
Average Daily Productive Hours	25.6

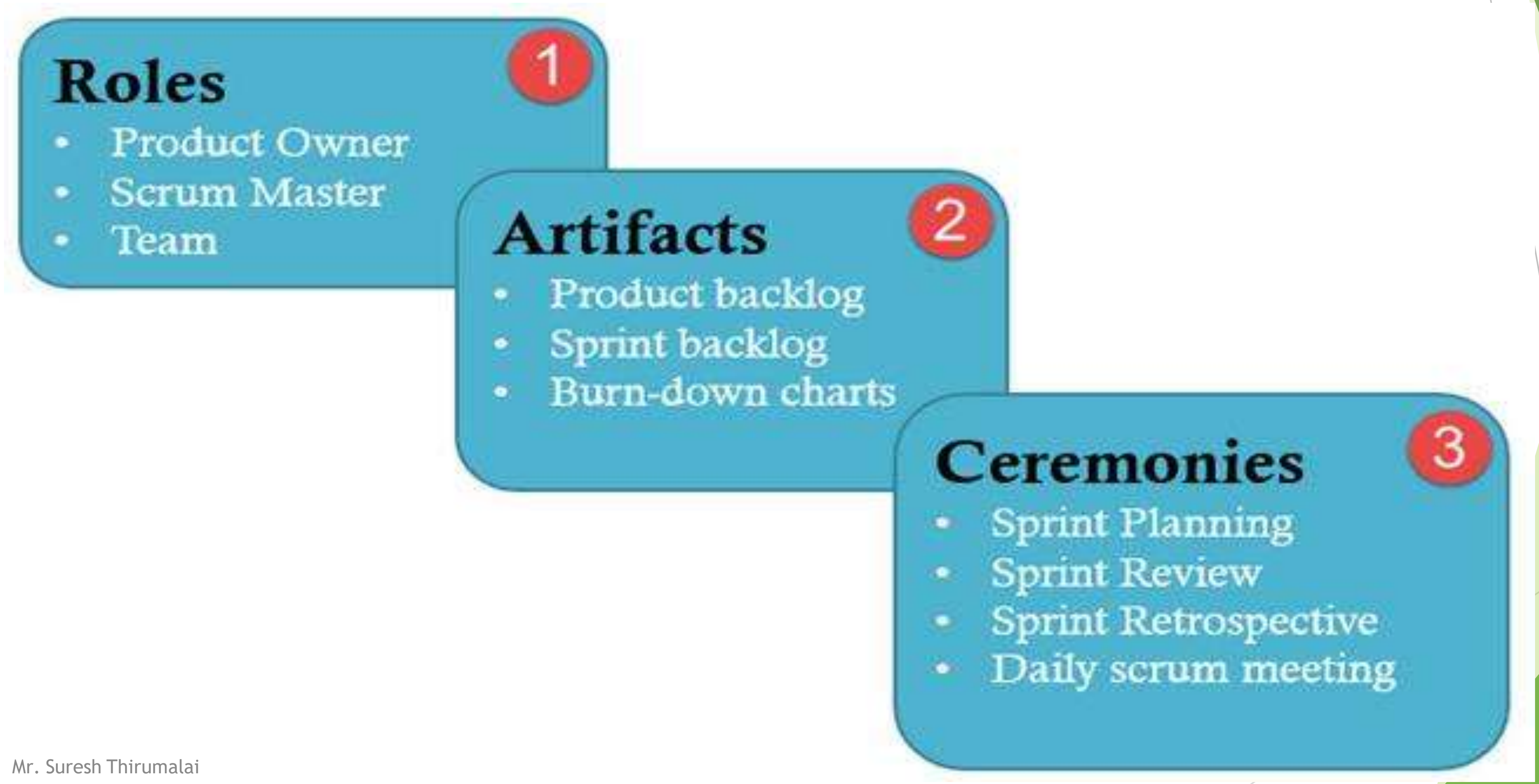
Day of the Sprint	Forecast	Target	Actual
0	256	320	320
1	230.4	288	256
2	204.8	256	224
3	179.2	224	208
4	153.6	192	192
5	128	160	176
6	102.4	128	144
7	76.8	96	96
8	51.2	64	48
9	25.6	32	16
10	0	0	0

Mr. Suresh Thirumalai

Process flow of scrum

- Each iteration of a scrum is known as Sprint
- Product backlog is a list where all details are entered to get end-product
- During each sprint, top user stories of Product backlog are selected and turned into sprint backlog
- Team works on the defined sprint backlog
- Team checks for the daily work
- At the end of the sprint, team delivers product functionality

3. Ceremonies



Roles 1
- Product Owner
- Scrum Master
- Team

Artifacts 2
- Product backlog
- Sprint backlog
- Burn-down charts

Ceremonies 3
- Sprint Planning
- Sprint Review
- Sprint Retrospective
- Daily scrum meeting

1. Sprint Planning Meeting

- Tester should pick a user story from product backlog that should be tested
- Tester should decide how many hours (Effort Estimation) it should take to finish testing for each of selected user stories.
- As a tester, He/she must know what sprint goals are.
- Tester should contribute to the prioritizing process

What happens in a Sprint?

- Support developers in unit testing
- Test execution is performed in a lab where both tester and developer work hand in hand.
- Defects are analysed during scrum meeting.
- Defects are retested as soon as it is resolved and deployed for testing
- Attend all daily standup meeting to speak up
- Schedule testing with CI system.
- Accomplish the automation test by utilizing various open source or paid tools available in the market.
- Review CI automation results and send the Reports
- At the end of the sprint, the tester also does acceptance testing (UAT) in some case and confirms testing completeness for the current sprint

2. Daily Scrum Meeting

Cisco webex

Participants –

Client, (Product owner)

Scrum Master, (Lead)

Dev Team,

Automation Team

Functional Team

(Manual)

Microsoft Teams

skype

7.30PM

3. Sprint Review Meeting

- The **sprint review** is an informal **meeting** which the development team, the **scrum** master, the product owner and the stakeholders will attend.
- The team gives a demo on the product and will determine what are finished and what aren’t.
- It is mandatory to deliver a potentially shippable product increment at the end of every **Sprint.** So, **Sprint Review meeting** is held at the end of each **Sprint**.

Mr. Suresh Thirumalai

4. Sprint Retrospective Meeting

- As a tester, He will figure out what went wrong and what went right in current sprint
- As a tester, He identifies lesson learned and best practices.

Sprint of our Project

- 2 weeks of sprint
- 1.5 years – 35 sprint
- Last Sprint 35
- Current Status:
- First Sprint?

POC

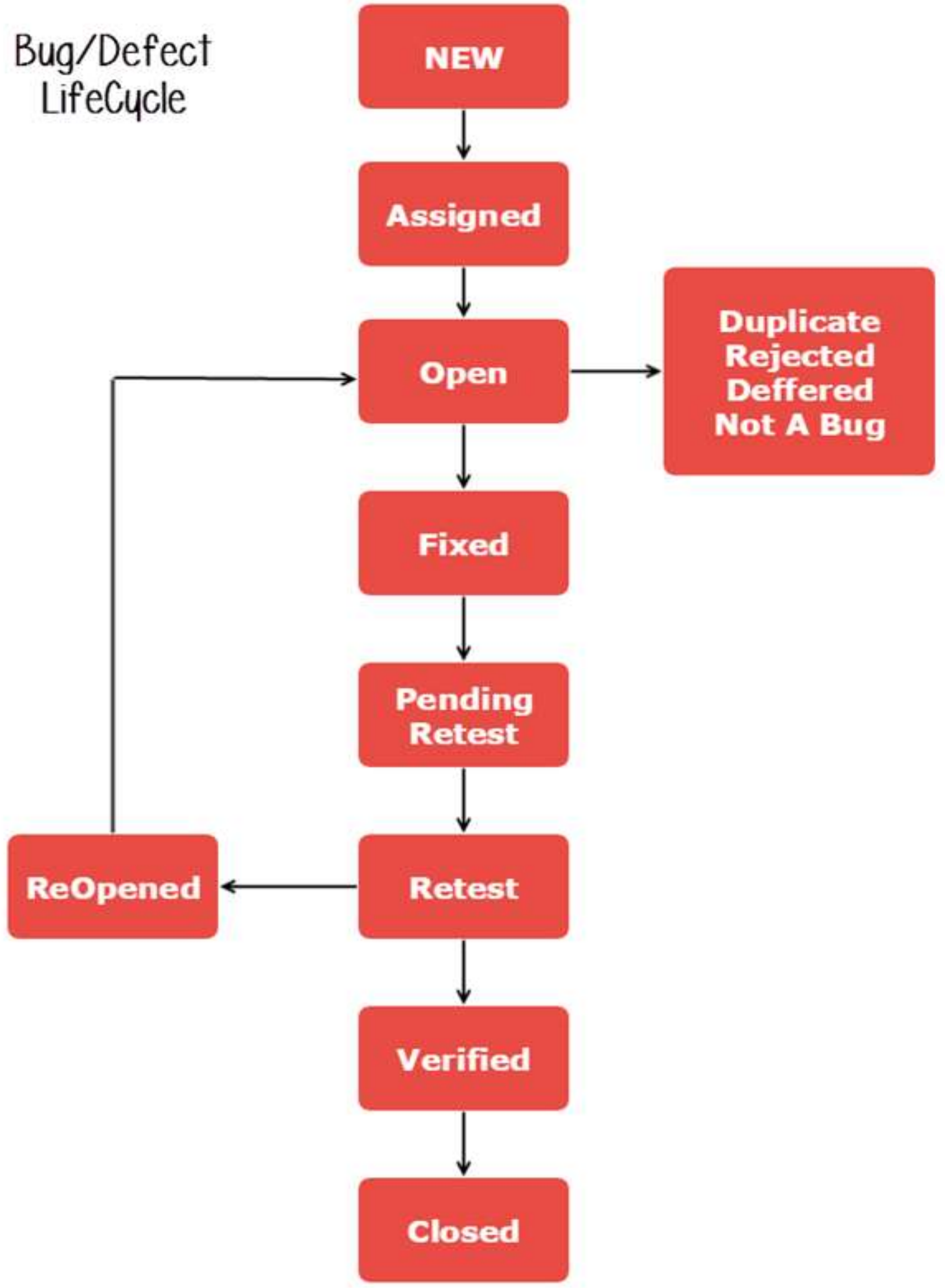
20. JIRA

Defect Tracking Tool

To create a issue,

1.	**Project**	:	Name of the Project
2.	**Issue Type**	:	Bug/Epic/Story/Task/Subtask/New Feature
3.	**Summary**	:	One line summary
4.	**Description**	:	Details of the issue
5.	**Priority**	:	High/Medium/Low/Lowest
6.	**Labels**	:	Specific type of issue
7.	**Issue**(Related To)	:	User can link the Issue ID to the linked issue
8.	**Assignee**	:	Who is responsible to fix this issue.
9.	**Epic Link**	:	If the issue belongs to any of those Epic Link
10.	**Sprint**	:	In which Sprint, this issue addressed

Defect Life Cycle



Mr. Suresh Thirumalai

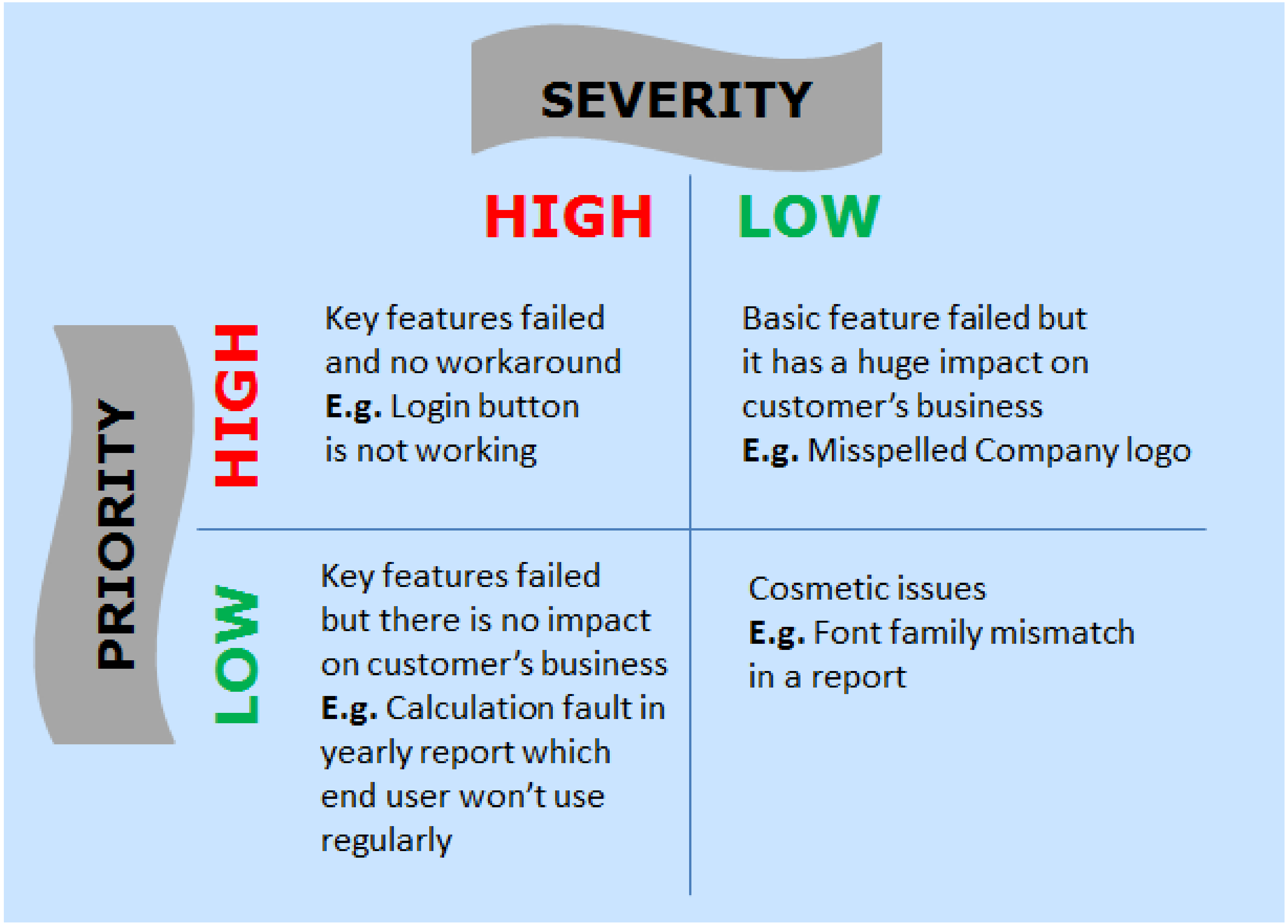
Priority vs Severity

Severity:

How severe defect **is** affecting the functionality.

Priority:

How fast defect has to be fixed



PRIORITY \ SEVERITY	HIGH	LOW
HIGH	Key features failed and no workaround **E.g.** Login button is not working	Basic feature failed but it has a huge impact on customer's business **E.g.** Misspelled Company logo
LOW	Key features failed but there is no impact on customer's business **E.g.** Calculation fault in yearly report which end user won't use regularly	Cosmetic issues **E.g.** Font family mismatch in a report

Mr. Suresh Thirumalai

Velocity

- **Velocity** is a measure of the amount of work a Team can tackle during a single Sprint and is the key metric in Scrum.
- **Velocity** is calculated at the end of the Sprint by totaling the Points for all fully completed User Stories

How to Calculate Velocity?

- Simply add up the total of story points completed from each sprint, then divide by the number of sprints

What is Deferred

- **Deferred:**

The bug, changed to **deferred** state means the bug is expected to be fixed in next releases. ...

If the tester feels that the bug no longer exists in the software, tester changes the status of the bug to “closed”.

- Low priority and low severity

QA - QC - QE

QA (Quality Assurance) defines the Testing Process

QC (Quality Control) is actual testing performed on the Software

QE (Quality Engineering) - As Testers are writing the code for testing the Software,

- QC is also referred as QE these days.

Error – Defect - Bug

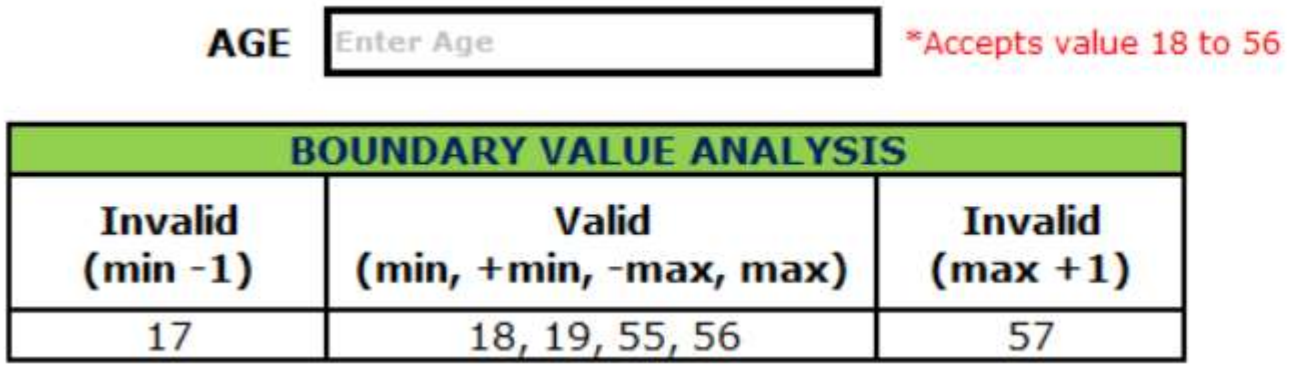
A **mistake/error** in coding is called **Error**, **error** found by tester is called **Defect**, **defect** accepted by development team then it is called **Bug**,

Testing Techniques (or)
How to design the testcases?

- Software Testing Techniques helps to design better test cases.
- Manual Testing Techniques help reduce the number of test cases to be executed while increasing test coverage.
- They help identify test conditions that are otherwise difficult to recognize.

Boundary Value Analysis
&
Equivalence Partitioning

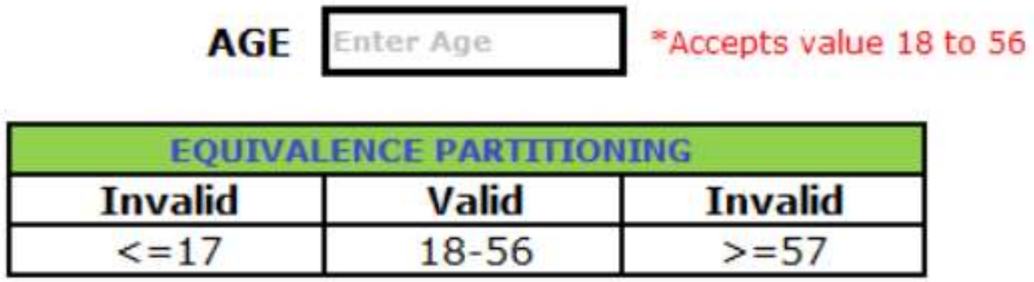
- **Boundary Value Analysis** is the process of testing between extreme ends or boundaries between partitions of the input values.



AGE Enter Age *Accepts value 18 to 56

BOUNDARY VALUE ANALYSIS		
Invalid (min -1)	Valid (min, +min, -max, max)	Invalid (max +1)
17	18, 19, 55, 56	57

- **Equivalence Partitioning** is testing technique used to divide the set of test conditions into partitions, and then a single value from each of those partitions are tested.



AGE Enter Age *Accepts value 18 to 56

EQUIVALENCE PARTITIONING		
Invalid	Valid	Invalid
<=17	18-56	>=57

Thank you

You are the Best