**Software testing Notes**

**Session -1:**

* **Introduction to Software Testing and its Significance**
* Software - > set of instructions to operate an application.
* Types of software - >system software (Ex: OS) , application software (Anything which we use in the computer like YouTube, face book)
* Software testing -> to evaluate and verify a software product to meet its functional requirements and thinks like an end user.
* Examples :
  + Amazon check out without credit card details,
  + Laser operation without accuracy,
  + Cheating Frenchmen sues Uber for a bug which revealed his trips with GF to his Wife.
* Software quality will be different for Producers and consumers -> From producer’s point of view, s/w should meet its requirement.
* From consumer’s point of view, s/w should meeting customer needs.
* These two things should be in the same line. if meeting requirements and meeting consumer needs is not the same, then product quality will be a big question mark.
* For that, more than a developer, a quality analyst must understand the product requirements and should forecast the expected outcome of a product.
* **History and Evolution of Software Testing:**
  + It's all started with programming and debugging phase, when finding errors during debugging was considered testing.
  + Software testing didn’t evolve in a single day; it took time and sweat to get it where it is today. Testing gurus like Hetzel and Dave Gelprin divide testing into five significant eras:
* ***Debugging-oriented era:*** 
  + This phase was during the early 1950s, when there was no distinction between testing and debugging. The focus was on fixing bugs.
  + Developers used to write code, and when faced with an error would analyse and debug the issues.
  + There was no concept of testing or testers.
* ***Demonstration-oriented era:***
* From 1960s to 70s, the distinction between debugging and testing was made and testing was carried out as a separate activity.
* During this era, the major goal of software testing was to make sure those software requirements were satisfied.
* As an example, the requirement might have been ‘We need a web application that displays a list of 10 products only’.
* ***Destruction-oriented era****:*
* From 1979 to 1982, the focus was on breaking the code and finding the errors in it.
* It illustrated the software engineering community’s desire to separate fundamental development activities, such as debugging, from verification.
* As an example, a tester would test software in such a way that it would break (e.g. entering letters in a field that should only accept numbers).
* There was **no defect prevention approach** during this phase.
* However, the destruction-oriented approach also failed because software would never get released because you could find one bug after another.
* Also, fixing a bug could also lead to another bug.
* ***Evaluation-oriented era****:* 
  + From 1983 to 1987, the focus was on evaluating and measuring the quality of software.
  + Testing improved the confidence index on how the software was working.
  + Testers tested until they reached an acceptable point, where the number of bugs detected was reduced.
  + This was mainly applicable to large software.
* ***Prevention-oriented era****:* 
  + 1988 to 2000 saw a new approach, with tests focusing on demonstrating that software met its specification, detecting faults and preventing defects.
  + Code was divided into testable and non-testable.
  + Testable code had fewer bugs than code that was hard to test. In this era, identifying the testing techniques was the key.
* Why do we do software testing
* To identify defects early
* Flawless system
* To increase overall quality of a system.
* What do we test

A product’s

* Functionality based on it’s requirement
* Usability and performance -
* Aesthetics of the product – how it helps user to understand its functionality
* Cosmetic issues
* Ex: <https://opensource-demo.orangehrmlive.com/index.php/dashboard>
* <http://www.leafground.com>
* Attributes of Software Testing
  + Functional suitability –works as per requirement
  + Reliability – fault tolerance when there is an issue with the network/internal error.
  + Usability – User friendly and it should not be boring to user’s eyes .
  + Performance efficiency –withstanding behavior against multiple load
  + Security - resistant to malwares and hacks
  + Compatibility – compatible to different OS.
  + Maintainability – reusability.

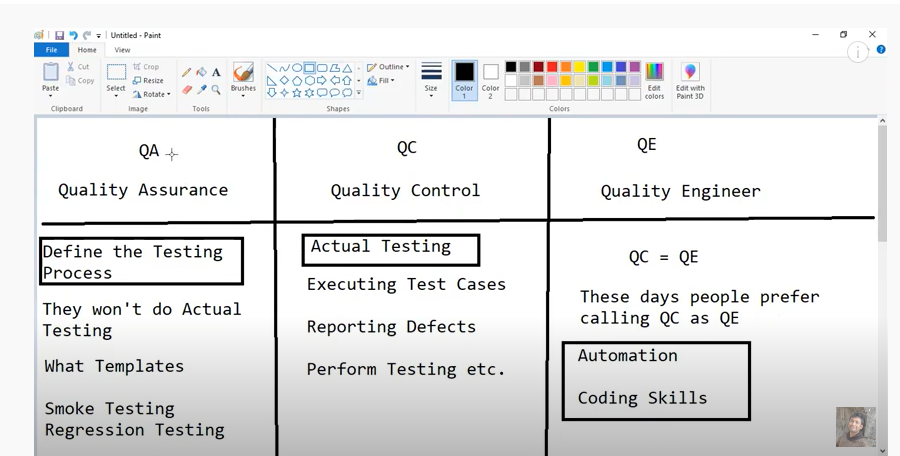


**Session -2:**

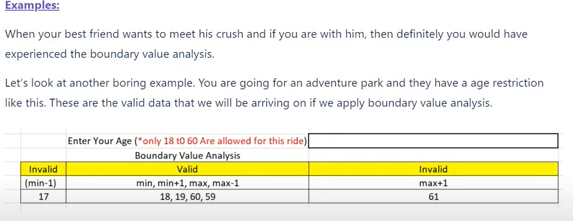
1. ***What is Verification***
   * **Process oriented activity**
   * S/W QA Team - First 15 days planning –next 15 days Development understanding - next 2months development –next 1 month testing. --🡪 This is verification
   * Quality Assurance
   * Proactive approach
   * Are we building a product right?(in right way)
   * SQA /TQ ->Software Quality Assurance Team / Technology Quality team is responsible for the process to work in proper way.
   * SQA is responsible for a tester to write proper test scenarios and test cases.
   * They will review our work and we will walk them through our test scenarios.
2. ***What is Validation***
   * **Product oriented activity**
   * Quality Control –if a product is working 100% correctly.
   * Are we building a right product?
   * Testers are being deployed for validating a product.
   * Reactive approach
3. ***Difference between Verification and Validation***
   * A 10th student is preparing for board exam. Daily morning his parents wakes him up at 4 AM.

They think that their son is preparing well and he will clear the exam. They do not know what he is studying – This is verification

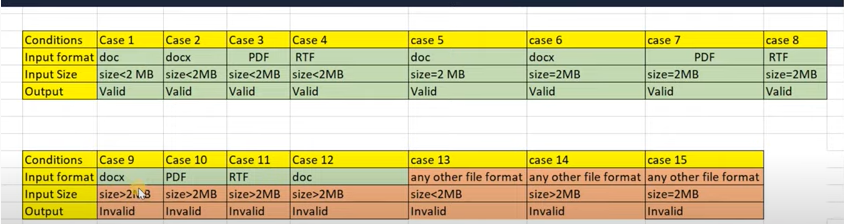
But when he goes to the school, there his teacher conducts a test and his potential about the subject is being validated. – This is validation

1. ***QA vs QC vs QE***
2. ***Principles of Testing:***
   * Testing shows the presence of defects, not their absence.
     + When a Bike is tested at a lab to give 80km mileage , but practically it will give maximum of 60km only on road.
     + If a software is 100% bug free , then why windows/Google is releasing versions in the product? Ex Win 7, Win 10.
   * Exhaustive testing is impossible.
     + We cannot test all type of probability and combination.
     + To test login functionality ,we can probably use this below combination,.

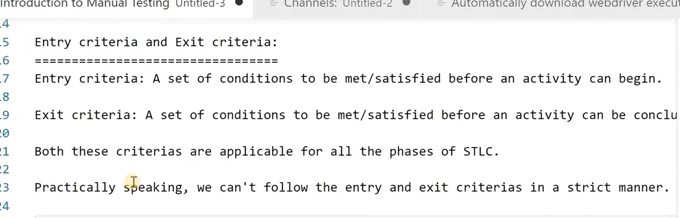
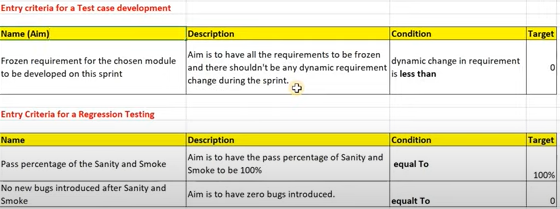
* But what if we need to test a SIGN UP form???
* There come our new techniques.
* **Boundary value Analysis** -(0 -100) 0 is a boundary ,100 is a boundary
  + **Lower boundary ,Upper boundary ,On Boundary**

****

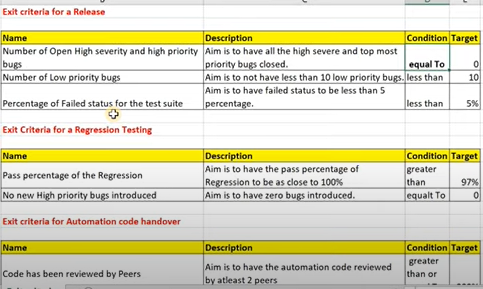
* **Decision Table Technique**
  + **Input and Op format is captured in Table format.**
  + **Example : Naukri -> Upload CV option**
  + Tester in parallel will start to write this as part of test design. Though product is not available to test, test scenarios will be prepared even before testing. So this tabular column will be shared to Dev team so Dev will write the code based on the below scenarios.

****

* Early testing saves time and money.
  + To find defects at early stage, start testing at early stage. So that the product will be a quality product.
  + Example of interview question: **In a 3 month project you find a bug in 15 days and another person finds a bug in 30days. Which one is better and how it’s better?**
  + **How - >**  the developed module in 15 days will comparatively less than 30days. SO dev can easily rectify its mistake. So early testing is good.
* Defects cluster together.
  + Grouping –Clustering
  + 80:20 principle -> In a class , only 10 -20% guys will be not interested to learn and they will disturb even hard working students. So prof will send these 20% guys from the class room.
  + Example: A dev does a mistake by putting $ in place of Rs, it has adversely effected into books,furntiures,electronics section in Flipkart app.
  + This is not 3 mistakes. but a one mistake,
* Beware of the pesticide paradox.
  + Update the test case as per new environment.
  + Example: Before 5 years, one has to test the app only in Laptop view, but now we have to test in tab view and mobile view also.
* Testing is context dependent.
  + Tester has to work based on its context.
  + Example: A fruit seller usually measures 500gm and if the fruit measures 520gm, he will neglect that 20gm and he gets the amount for 500gms.But what if we does that in Jeweler shop??
  + Example: Testing Approach for Google page, YouTube and Gmail is different.
* Absence-of-errors is a fallacy.
  + One software has to meet its Customer’s expectation. Though we give a error free product but if the customer spec is not relying to that, its waste.
  + Example: customer wants to build a mobile app, we developed everything perfectly in Android but the customer need is to create the app in IOS. So effort has gone in vein

1. ***When do we Test and When do we NOT: (Entry and Exit Criteria):***
   * It is a Guidelines to write a test case.
   * Entry Criteria:
   * ******
   * ******

Exit Criteria:



**Week -2**

**Session -1 :**

1. **Introduction to Types of Testing:**
   1. [**Accessibility testing**](https://www.perfecto.io/accessibility-testing)**-(Non functional testing):**
      1. Accessibility testing is the practice of making your web and mobile apps usable to as many people as possible.
      2. It makes apps accessible to those with disabilities, such as vision impairment, hearing disabilities, and other physical or cognitive conditions.
      3. How to perform Accessibility testing?
         1. Check for Transcripts. (For deaf and dumb)
         2. Read out captcha images
         3. Turn off mouse/track pad – since visually challenged person can’t see mouse pointer.
   2. **Black box testing:**
      1. Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure and implementation details.
      2. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is done from the end user's perspective.

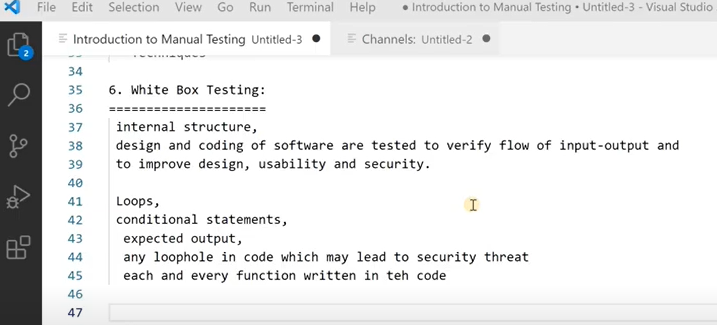


All these come under STLC (Software Testing Lifecycle)

The entire consumer will do black box testing.

**Example:**

1. While going to buy a bulb, we won’t test the diameter of a bulb rather we will test only if the bulb is producing illumination or not.
2. Google search engine term - > final product only will be tested by testing team. Functionality only will be known to testing team. Implementation logic will not be visible to user/tester.
3. How to do black box testing?
   1. Understand the end users requirements
   2. Creating a test strategy
   3. Designing the test cases with proper test case designing technique
   4. Creating the test cases
   5. Executing the test cases
   6. Raising a bug if any of the functionality is not working as expected
   7. **White-box testing:**
      * 1. To improve code quality and improve security features of a software.
        2. Developers will be doing white box testing. No testers will be doing white box testing.
        3. **UNIT TESTING is white box testing.**

****

* 1. **Integration testing:**
     1. To test the interfaces between the module.
     2. Example: In youtube live, when you try to screenshare option, you will do via 8x8 app.
     3. When you try to purchase a thing from a Ecommerce website, you will be re directed to banking website. So you have to test these integration part is working fine or not.
     4. **Two approaches:**
        1. **Top Down approach ->** Component 1 -> Component 2 (Ex: To test ,Youtube ->Share option -> Whatsapp/Telegram)
        2. **Bottom up approach ->** Component 2 ->Component 1(Ex: To test, From Whatsapp the shared youtube link is working fine or not)

**Interview Question:**

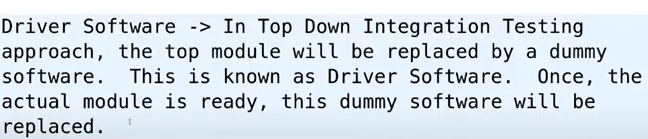
1. Diff btwn Unit testing and Integration testing

Ans: Unit testing -> testing the modules individually(Ex: testing youtube separately and whatsapp separately)

Integration testing -> To test the interfaces between the module.

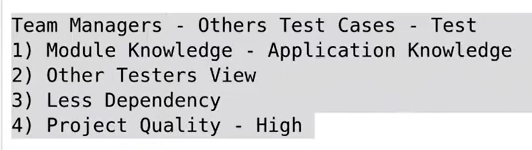
* 1. What if the component 1 software is not ready/under development , how can we test component 2??

At that time component 1 will be given as API/Dummy s/w ->Driver s/w

****

**Stub software -> In bottom up approach, if the component 2 is not ready/under development, then there will be a dummy software given to us. That will be known as Stub.**

**Note:**



* 1. **Load testing:**

Load Testing is a type of performance testing which determines the performance of a system, software product or software application under real life based load conditions.

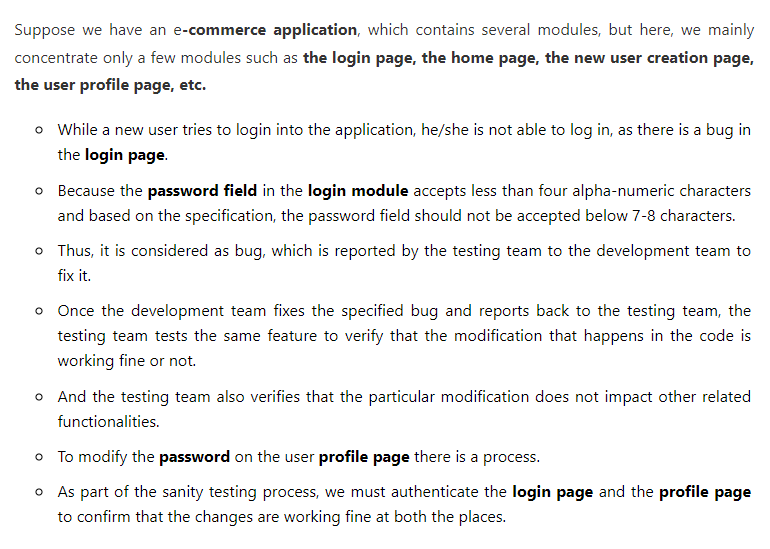
* 1. **Performance testing:**
     + 1. Performance Testing is a type of software testing that is carried out to determine system performance in terms of responsiveness, stability, scalability, reliability, speed and resource usage of your software and infrastructure.
       2. Testing the stability and response time of an application when there are less than or equal to desired load. -🡪 Load testing

|  |  |
| --- | --- |
|  | Load( No of users hitting the app in the desired time) |
| Performance testing | **Stability(able to handle the desired no of users )** 🡪 Load testing + Response (going for payment page by giving a req to the server and coming back to the server) |

* + - 1. Example: Board exam results -> traffic will be more while results is published.

Performance testing vs Load testing:

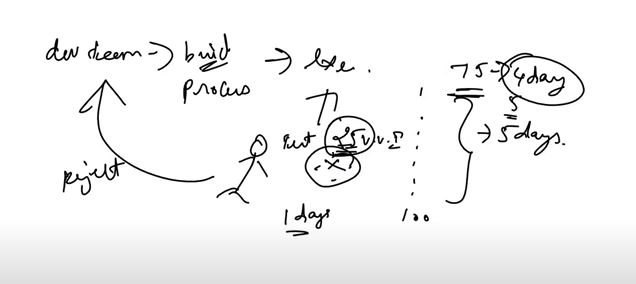
* 1. **Stress testing:** 🡪 Testing Testing the stability and response time of an application when there are More than or equal to desired load.
  2. **Sanity testing**: -> It is used to ensure that all the bugs have been fixed and no added issues come into existence due to these changes. Sanity testing is unscripted, which means we cannot documented it. It checks the correctness of the newly added features and components.
     + 1. Example:



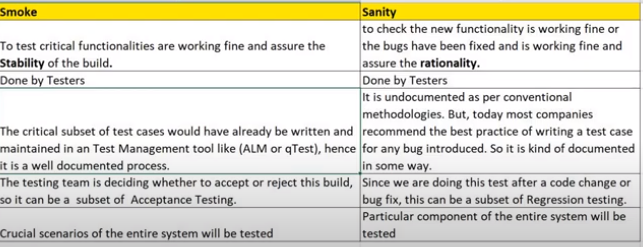
* 1. **Smoke testing/Confidence testing/Build verification testing**:
     1. Testing the basic & critical feature of an application before doing one round of deep, rigorous testing (before checking all possible positive and negative values) is known as smoke testing.
     2. In the smoke testing, we only focus on the positive flow of the application and enter only valid data, not the invalid data. In smoke testing, we verify every build is testable or not; hence it is also known as **Build Verification Testing.**
     3. To find if the latest put build is Stable build or not? Ex: Youtube introduce additional feature through a build, but when we test that functionality, videos are not properly playing( core functionality failed) means , it s not a successful build.
        + 1. Why do we need smoke testing?

When A Robot is created, it has to follow Isac Asimov rules. In Robo movie, there is scene where chitti will be introduced to the Scientist where scientist ask n number of questions to it, at that time a guy ask this question!! This robo will follow Asimov’s first principle?? If not they will reject this robot.So thumb rule should be followed for all products.

If Youtube is not playing videos properly while testing, who will waste their time to check if the CC/Subtitles part is working or not??



* 1. **Difference between Smoke and Sanity testing:**
     1. Smoke testing -> like a master check up
     2. Sanity testing -> Specialized check up(Ex: Eye check up/heart check up)
     3. Sanity testing -> Mini Regression testing



* 1. **Security Testing:**
     + - 1. To secure s/w from potential vulnerability , we would do security testing.
         2. There are 7 types of security testing in software testing. These are as follows:

**Vulnerability scanning:** An automated software scans a system against identified vulnerability.

**Security scanning:** This scanning can be performed for both Manual and Automated scanning. It identifies the network and system weaknesses. After that, it provides the solution as well

[**Penetration testing:**](https://www.edureka.co/blog/what-is-penetration-testing/) An analysis of a system to check for the potential vulnerabilities, if there is an external hacking attempt.

**Risk assessment:** The analysis of security risks observed in the organization is done here. Risks are classified as Low, Medium and High. It helps to provide measures in order to reduce the risks.

**Security auditing:** It’s a kind of internal inspection of applications and Operating systems for checking the security flaws.

**Posture assessment:** This combines security scanning, ethical hacking and risk assessments to represent overall security of the organization.

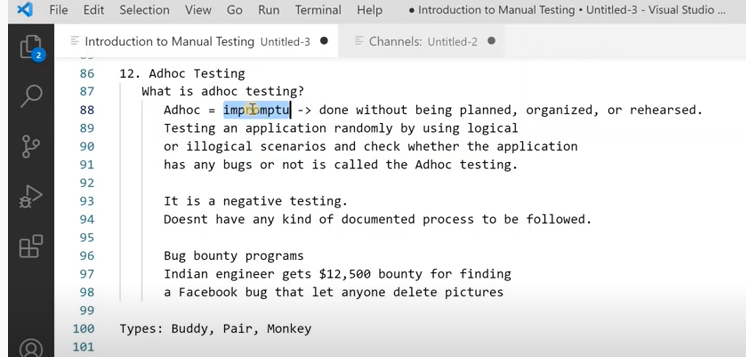
[**Ethical hacking:**](https://www.edureka.co/blog/ethical-hacking-tutorial/) The hackers attempt hacking in order to expose the flaws in the security system of the organization.

* 1. **Regression testing:**
     1. Regression testing is a software testing practice that ensures an application still functions as expected after any code changes, updates, or improvements.
     2. Regression testing is responsible for the overall stability and functionality of the existing features.
     3. Super set of Sanity testing.
     4. Covid vaccine - > works properly ->Sanity testing
     5. Because of covid vaccine, is there any problem in the body like lever failure/hear failure due to the vaccine it is regression testing.

**Re testing:**

* + Retesting is a process to check specific test cases that are failed due to bug/s in the previous execution is getting passed or not in the next execution done after the big fix.
  + Generally, testers find these bugs while testing the software application and assign it to the developers to fix it.
  + Then the developers fix the bug/s and assign it back to the testers for verification. This continuous process is called Retesting.
  + Key difference between Regression and Retest:
    - Regression will check if any other modules across the entire system is affected due to a code change or bug fix or new addition of functionality.
    - Retesting is just focusing on a particular failure test scenario due to a bug and verify it again after the bug fix.

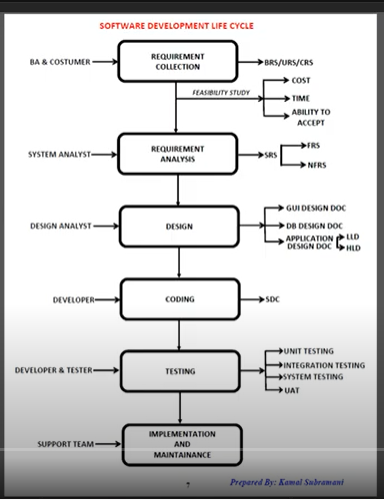
**Adhoc testing:**

****

**Week 3 Notes**

**Session -1 :**

**SDLC ->** it is defined as a step by step process to develop a software product with 100% quality.



**Requirement collection:**

Typically, requirements collection refers specifically to the practice of gathering the software requirements from the customer.

**Example:**

(In a Restaurant)Customer – Customer (In S/W Company)

(In a Restaurant )Server/Waiter – Business Analyst (In S/W Company)

(In a Restaurant)Chef – Developer(In S/W company)

Waiter writes down the need of the customer in a Restaurant

BA writes down the need of the customer -> Business Requirement Document /Specification

System requirement Specification -> Technical i/p will be given by customer(Ex: we need to use Java)

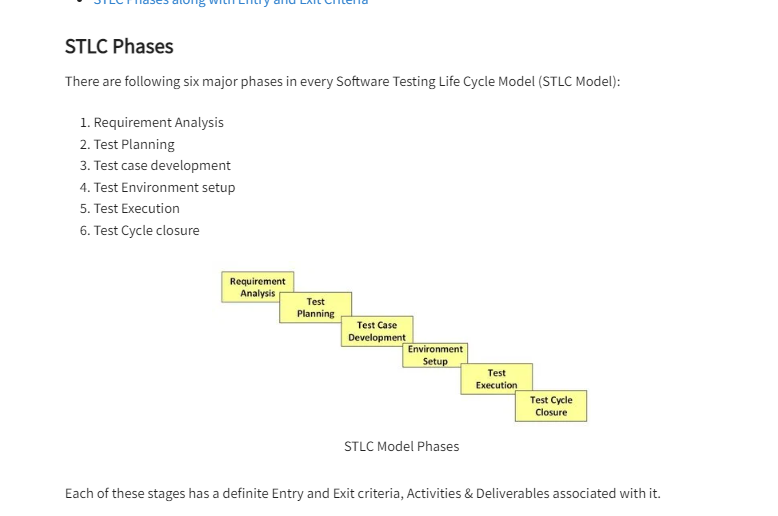
Business requirement Specification -> Business i/p given by customer (we need to build a website)

**Requirement Analysis/Planning:**

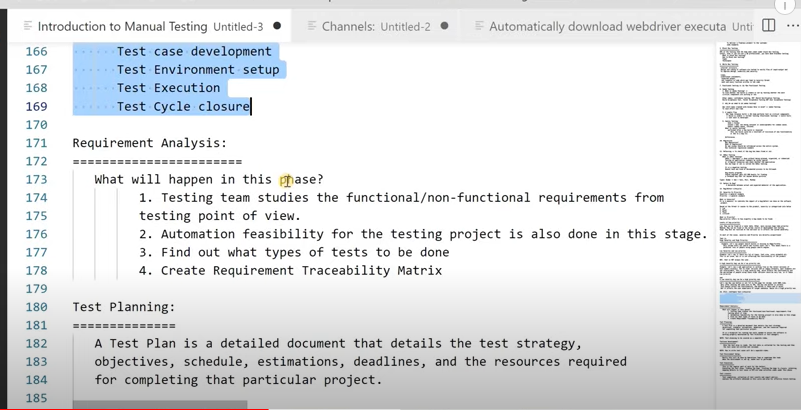
I/P -> BRS document given to Tester and Developer

**Etc….**

**STLC:**

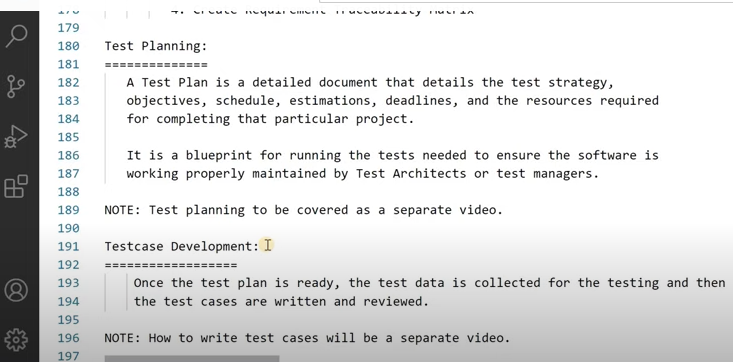
****

**Test Planning:**

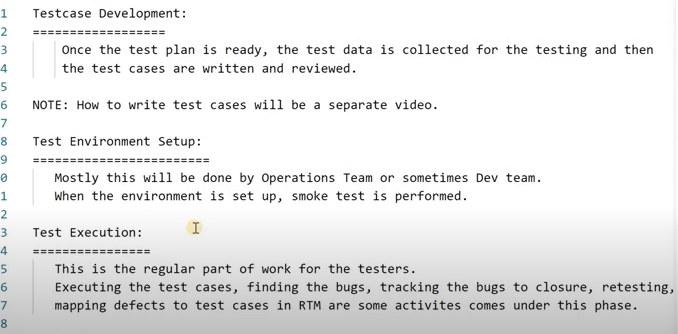
****

**Requirement Traceability matrix**

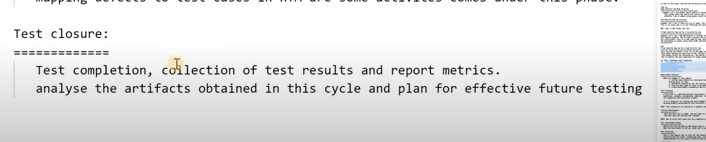
**Test case Development:**

****

**Test Env Set up:**

****

**Test Closure:**

****

**Week 3 –Session 2**

* **Risks associated with project and product:**

**Project Risk:**

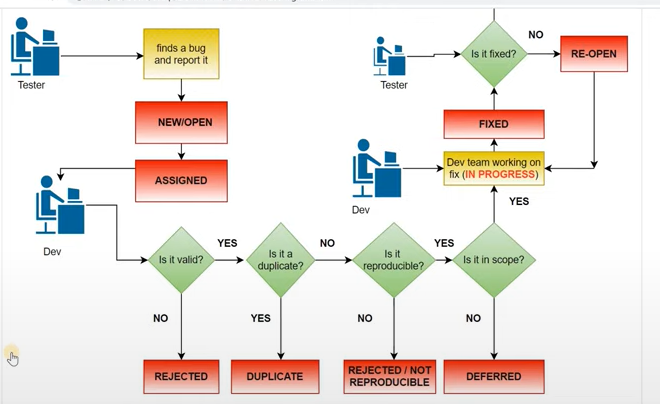
* **Delays in Delivery and Task completion -**
  + Heard this story before? The testing team was supposed to get stories on Monday, and it's already Friday !!
  + It's the most common project risk where there is a delay in completing the Development Task for a story. Therefore, there is a delay in the story delivery to the testing team.
* **Cost Challenges –**
  + Project funds and resources might get allocated to other high priority projects in the organization.
  + There could also be cost-cutting across an organization, which could lead to reduced funds and resources for the project. It will impact testing resources, as well. Can you relate to the risk that six folks would now do a work that was supposed to be done by ten people? Of course - the timeline always remains the same!
* **Inaccurate Estimates -**
  + Estimation of Home Page development for a website was 20 days of development and 7 days of testing. When actual work started, the team figures out that they will need 35 days of development and 12 days of testing.
  + Can you relate to this? When a project begins, then the high-level estimation happens according to which allocation of resources and funds takes place. These estimates likely turn out to be inaccurate when actual work starts. It could lead to delays, quality issues, or cost overruns for both development and testing teams.

**Product Risk:**

Product risks result from problems/defects with the delivered product.

Product Risks associate with specific quality characteristics of the product. Therefore they are also known as Quality Risks.

**Defect life cycle:**

****

**Defect triage:**

* Defect triage is a process where each bug is prioritized based on its severity, frequency, risk, etc.
* The goal of Bug Triage is to evaluate, prioritize and assign the resolution of defects.
* The frequency of defect triage meeting is decided according to the project schedule, number of defects in the system, overall project health, etc.
* Project Manager, Test Team Leader, Technical Lead, Development Team Leader are taken part in this meeting.
* Defects are analyzed and evaluated by the degree of their severity.

**Defect Density:**

