**Software Testing Simple**

**notes**

**SDLC**

customer: request a features

server: product owner write a user stories against the requirements

cook: dev writes the code

chef: QA Performs testing

customer: accepts the features

Tester "breaks the things"

**Software testing:**

The process of verifying and validating that a software is bug free and meets all the requirements with handing all the exceptional and boundary values

**Verification:** Are we building the product, right? It includes checking documents, designs, codes, and programs.

**Validation:** Are we building the right product? It includes testing and validating the actual product. end user need.

**Bug:**

an error flaw or fault in a computer program that causes it to produce an incorrect or unexpected result, or to behave in unintended ways

**Types of bugs:** Design, Functionality, spelling, compatibility, localization, performance, usability, security

**UI-testing (design):** Layout, Font, Color, Content, Images

**1. Layout Design bug**

* **Misalignment**: double margin float bug, staircase effect, IE6 box model bug, misbehaving floated layout, float drop bug, guillotine bug, IE resize bug.
* **Overlapping:** auto overflow
* **Missing content/ images**

**Spacing:** linewidth, line height, unwanted space between list items, vertical spacing bug.

**2. Font Design bugs**

* **Different type:** The font would be of the same family but the different type(e.g.: Helvetica lite was the font but developer had put Helvetica regular)
* **Different family used**
* **Mismatching of font size**

**3. Color Design Bugs**

* **Color mismatch:** mismatch of color code due to multiple reasons
* **Button:** button color should have changed when disabled but got missed

**4. Content**

* **Upper/Lower case differences**
* **Mismatch of letters/ words**
* **Spelling mistakes**

**Functional Testing**

System is tested against the functional requirements/ specifications

Block-box testing: you dont know ( and you dont care) how it works under the hood

User Action 🡪 system reaction

**Compatibility testing**

Type of Software testing to check whether you software is capable of running on different hardware, operating Systems, application, network environments or mobiles devices.

Hardware, Operating Systems, Software, Network, Browser, Devices, Mobile, Versions

**Localization testing**

Product is checked is assure that it behaves according to the local culture or settings.

Language, Currency, Date and time format.

**Performance testing**

Determining the speed, responsiveness and stability of a software program or device a workload. e.g black Friday sale.

**Usability testing**

Usability testing is a method used to evaluate how easy a website is to use( user Experience (UX)

**Security testing**

Testing technique to determine if an information system protects data and maintains functionality as intended.

**Ad-hoc testing**

Randomly testing the app, also called monkey testing.

Bug reporting : title, issue type, summer (what, where, how) description( steps), evidence(SS, Videos)

**Bug Triage:** Meeting where each bug is prioritized based on its severity, frequency, risk, etc

* To evaluate, prioritize and assign the resolution of defect
* We cannot fix every bug
* Ship with minor defects is OK
* Drive a car with scratches

**Severity and Priority**

* Severity is defined as the degree of impact on the development of application.
* Priority is defined as the order in which a defect should be fixed.

**Priority Classes**

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

**Severity Tips**

If the occurrence of a minor-defect is frequent in the code, it can be more severe. So from a user’s perspective, it is more serious even though it is a minor defect.

|  |  |  |
| --- | --- | --- |
|  | Critical | Non-Critical |
| Priority – urgent | Key feature does not work. | Company logo is the wrong color |
| Priority – low | Feature that is rarely used does not work | The caption on an image is written in the wrong font. |

Test Case:

* A set of actions executed to verify a particular feature or functionality of the software .
* It clarifies what needs to be done to test a system.

**Positive vs negative test cases.**

**Smoke Testing**

* Build Verification.
* Small set of test cases to ensure that the most important function works
* It shows if a build is stable enough to proceed with further testing.
* Bugs are usually P1(Critical priority)
* Happy Path
* Like a Test Drive of a Car.

Test Plan

* Document that describes software testing scope and activates
* Basis for formally testing any software/ product in a project
* It allows you to structure your efforts and see a bigger picture
* Never build a house without a Plan!

**Browsers**

* Google Chrome
* Safari
* Internet Explorer
* Mozilla Firefox
* Edge

Always ask from PO which browser we should use for testing.

Emulators/ simulators (Android studio, Xcode).

Cloud services( sauce Labs, Browser Stack).

**Cross Browser Testing**

* Test your application in multiple browsers and making sure that it works consistently and as in intended without any dependencies or compromise in Quality
* This is applicable to both web and mobiles applications.

**Test Plan**

* Document that describes software testing scope and activities.
* Basic for formally testing any software/product in a project.
* It allows you to structure your efforts and see a bigger picture.
* Never build a house without a plan.
* <http://softwaretestingfundamentals.com/test-plan/>

**Regression Testing:**

Type of Software Testing to confirm that a recent program or code change has not affected existing functionality/ features. Full or partial section of already executed test cases which are re-executed to ensure existing functionalities work fine.

**When to do?**

* Change in requirements and code is modified according to the requirements.
* New feature is added to software
* Defect fixing
* Performance issue fix

**Types of Regression**

* Retest All
* Regression Test Selection
* Prioritization of test Cases

**Regression Retest All:**

* This is one of the method for Regression Testing in which all the tests in the existing test bucket or suite should be re-executed.
* This is very expensive as it requires huge time and resources.

**Regression Test Selection:**

* Selecting the most appropriate test case to make sure the unmodified part of the program is error free.
* Identify the effected parts of the program – When a particular module is changed, other parts of the program are bound to be affected.
* Test Case Selection – Involves a selection of “test cases” from an already defined “test suite”.

**Prioritization of test Cases**

* Prioritize the test cases depending on business impact, critical and frequently used functionalities.
* Selection of test cases based on priority will greatly reduce the regression test suite.

Good number of the defects reported by customers were due to last minute bug fixes creating side effects and hence selecting the test cases for regression testing is an art and not that easy.

Effective regression Test ca be done by selecting the following test cases:

* Frequent Defects
* Visible to the users
* Core Features of the product
* Recently changed.

**Positive Testing**: Testing by Providing the valid data as input. Application does what it supposed to do.

**Negative testing**: Testing by providing invalid data as input. Application does not do anything that it is not supposed to do.

**Example:** The customer is eligible for a life assurance discount if they are at least 18 and no older than 56 years of age. **Boundary Value Analysis**

* Testing technique in which the test cases are designed to include values at the boundary.
* Positive Testing: input data is with in the boundary ( Base value)
* Negative Testing: input data is outside the boundary value limits ( one above and one below the base value)
* A system can accept the number from 18 to 56. All other number are invalid values. Boundary values 17,18,19 and 55, 56,57 will be tested.

**Equivalence Partitioning**

* Testing technique which divides the input data into many partitions.
* Values from each partition must be tested at least once.
* Partitions with valid values are used for positive testing
* Partitions with invalid values are used for negative testing.
* There are three partitions 0-18, 19 -56, 56+. Sample values: 5,40,76 can be taken from each part to test the scenarios. A lot of test cases all combined.

**Web Architecture**

* Front-End – part of application with which the user interacts directly. (Client Side)
* Back-End – part of application that is not directly accessed by the user, typically responsible for storing and manipulating data (Server Side)

**How it works**

* Browser: User Requests a page (HTTP Protocol)
* Back-End: Sends it to back as HTML (standard language for web browsers)
* Browser: Analyzes the page (parsing)
* Browser: Downloads all additional assets (scripts, fonts, images) talks to 3rd party back-end (payments, analytics, maps)
* Browser: Visualizes the page (rendering)

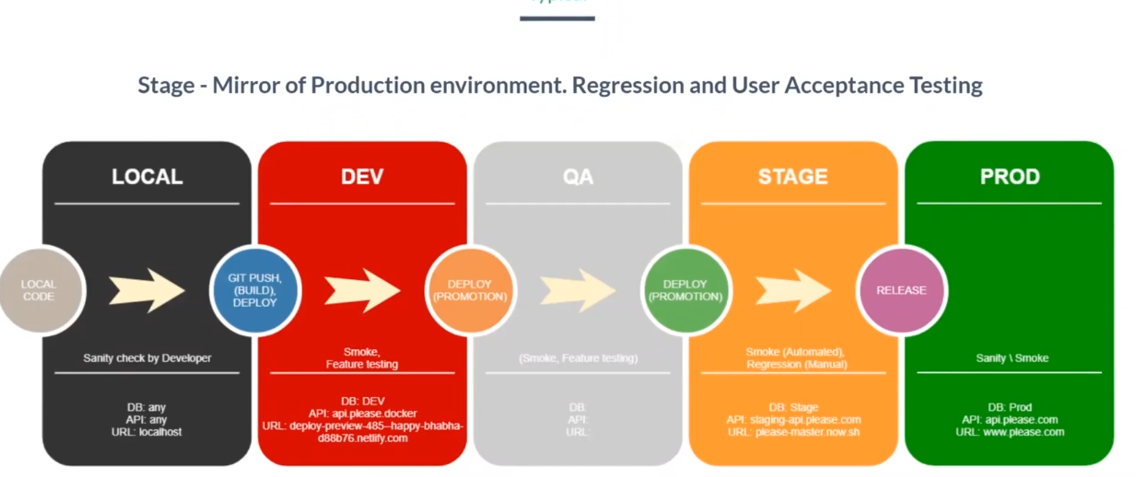
**How to debug front-end or back-end issues ?**

* Open the dev tools
* Go to network tab ( we have all the request)
* Header/ Response of specific request

**Build:** Compilation process, where source code files( test) are converted into executable code.( Dev to QA)

Deployment:

Release: Distribution of the final version of an application to customers. Based on Regression QA report. ( QA to Client)

****