Requirement

SDLC:

——> it is used to develop the software systematically.

1.Requirement :

—> here we are going to collect the req.

—> understanding the req.

2.Plan & Analyse:

—> req got filter and analysed accordingly.

3.Design:

—> we will create HLD & LLD

4.Coding:

—> here we will create actual project.

5.Testing:

—> here we will check all the functions are working according to client

6.Implementation & Maintenance:

—>changes & correction done in this stage

Models is SDLC: /SDLC Methods:

1.Waterfall Model:

——> we cannot go back and change anything

When to use: when the requirement is stable and clear.

When not to use: when the requirement changes

Baseline:——>it is also called checkpoint for the particular task.

2.Prototype model:

When: the requirement is not clear

—> here we are trying to confirm the req.

—> here we will create a small-small demo object

3.Increment model:

——>when requirement is clear but not complete.

Plan to implement:100 but we got only 20 now./

Release1 ——>it contain 20 req.

Release2——> it contain 30 req.

Release3

4.Spiral Model

—> when requirement is high risk

Req—>plan&analyse—>Risk identification—>design—>

Principles of Testing

What is Testing:

—> verification /checking /finding defect/ checking correctness ,completeness.

—> testing is not only used to identify the defect but also identify the performance ,security , usability

Why testing:

—> it is used to improve the level of quality

When testing:

—> after once requirement is confirmed.

—> static testing: it is used to check the testing in the doc level

How testing:[Manual Testing / Automation Testing]

There are 2 types of testing

1.Functional Testing:

—> it is used to validate the the working functionality of the specific object.

—> it is a data driven approach

2.Non-Functional Testing

—> checking the time taken to complete the process.

Performance / security / usability /configuration /compatability

Testing principles:

1.testing should start at the earlier stage of SDLC

STLC: testing levels

1.unit testing:

—> it is done by developer

—> it is used to verify each and every single features/module.

—> it is called development environment

Why it is required?

—-> we can’t relate a system if it not working 100%.

Software Release Matrix/Paramter:

1.100% functional coverage.

2.Integration testing:

—> it is used to verify the flow of the system.

—-> testing will be done by integrating one or more components together.

—> it is done by developer and some time independent tester

—> it is called development environment.

3.System testing:

—> it is used to check the complete business testing.

—> End-End Testing will be done here.

—> it is done by independent testing team

—> it is called Testing Environment

4.User Acceptance testing

—> client side testing

—> client will perform testing using Alpha & Beta testing

Alpha Testing: controlled environment testing

Beta Testing : Live Environment testing

Test Scenario:

—> possibilities of the object to test.

Ex:Login function

Scenario1 : Login Successful—-> when user enter correct details

Scenario 2: Login Unsuccessful——> when user enter enter incorrect details

Use case from Test Scenario:

1.Login use case

Testing Methods:

1.Black Box Testing :

—> it is also called data driven / input/output testing

—> it is used to check the behaviour of the application

—> it is done by the independent testing team

—> after white box will conduct the testing

Advantage of BBT:

—> it is used to improve the level of quality

2.White Box Testing

—> it is done by developer

—> it is used to check the logical condition of the program

—> after build will conduct the testing

Advantage of WBT:

—> it is used to complete the development of the product

UNIT TESTING ——————> Development Environment————> BlackBox & WhiteBox——>developer & Tester

INTEGRATION TESTING——> Development Environment————> BlackBox & WhiteBox——>developer & Tester

SYSTEM TESTING————-> Testing Environment—————> only BlackBox———> Tester

USER ACCEPTANCE TESTING———> Production Environment———>only BlackBox Testing——>Tester

Testing:

—> it is done by 2 different ways.

1.Positive testing:

—> it is used to check how the system should behave actually.

—> it is used to check according to the given conditions.

—> will always start with positive testing

—> check the system in proper way

2. Negative testing:

—> it is used to check whether the system not work the way it should n’t.

—> out of range

—> trying with different condition which is not mentioned

—> check the system un proper way

Scenario: check the user age limit for voting [18 - 60]

PT: age between 18 to 60

NT: age less than 18 & age greater than 60

Testing Techniques:Black Box Method:

—> Techniques are used to implement the testing or test case.

1. Equivalence Partitioning [EP]:

—> it is used when the given input is numeric and range level.

Ex:

Scenario: check the user age limit for voting [18 - 60]

Conditions. Valid Class Invalid class

Age 18 to 60 Any no between any no.of less than 18

18 To 60. Any no.of greater than 60

EP To Test case:

Positive Test case: 20, 30,55

Negative Test case: 15,12,9,80,75,65

Total test case : 20,30,55,15,12,9,80,75,65 [9]

Advantage:

—> small set of test case

Disadvantage;

——> only Assurance no guarantee

——> no proper guidelines to select inputs

2. Boundary value Analysis [BVA]:

Ex:

Scenario: check the user age limit for voting [18 - 60]

—> it is used to check the given conditions and boundaries of the given conditions.

X & Y | Boundaries of -1X+1 & -1Y+1

18 & 60 | Boundaries of 1-18 +1 & 1-60+1[18,60,19,17,61,59]

Total Test case : 6

Conditions. Lower Limit. Upper Limit

Age Between 18 To 60 Valid Invalid Valid Invalid

18 ,19. 17. 59, 60 61

Advantage:

—> very small set of test case

—> clear guidelines for selecting inputs.

Disadvantage;

——> only Assurance no guarantee

3. Descion table testing

4.state transition testing

5.Use case testing