Tools asked in Schneider:

vsts

postman

**Testing Fundamentals:**

**Prioritizing test cases-P1, P2, P3, P4-**

These are priotrity test cases-Urgent(p1), High(p2), medium(p3) and low(p4). Mention the requirement number just beside the testcase and vice versa. This would make the tester easy to map requirements with test cases which is traceability matrix nothing but test coverage.

**Smoke tests vs Sanity tests:**

Smoke:

This is nothing but the build verification tests(BVT).As and when we receive build(every new build) we test the application for critical issues and see that the build is validated and accepted, For example Patient Central is not logged in with pcuser. Typically 30 mins should be taken to run the smoke tests. Efficient for nightly and weekly builds. Few smoke test cases will be written and they will be run for each new build.

Sanity:

Verifies small bug fixes or small code changes for some specific functionality. Verifies requirements are met or not. It is also a Subset of acceptance testing. No testcases will be present.

Smoke VS Sanity:

a. Smoke is for testing critical issues as and when we receive the build. Sanity is testing for small and minor issues as and when we receive the build.

b. Smoke testing is based on test cases. Sanity is conducted without test cases but domain knowledge is required.

c. Smoke test is part of Regression testing and Sanity is subset of Acceptance testing.

For Banking domain, separate Testing is followed:

a. Requirements gathering testing

b. Requirements review

c. Regression testing-End -to-End transactions.

d. Database testing--Storing all the transactions in web server.

e. Security testing—Authentication-Restricting unauthorized access and allowing authorized access to the application.

f. User Acceptance testing--Business Requirements/User Requirements

Transactions that happens in banking:

i. Incoming and Outgoing transactions (cr/dr)

ii. Billings

iii. Loans

iv. Tradings

v. Fixed deposits

vi. Credit card transactions

vii. Interbanking transactions

viii. Cash/Cheque/DD/ATM Withdrawal

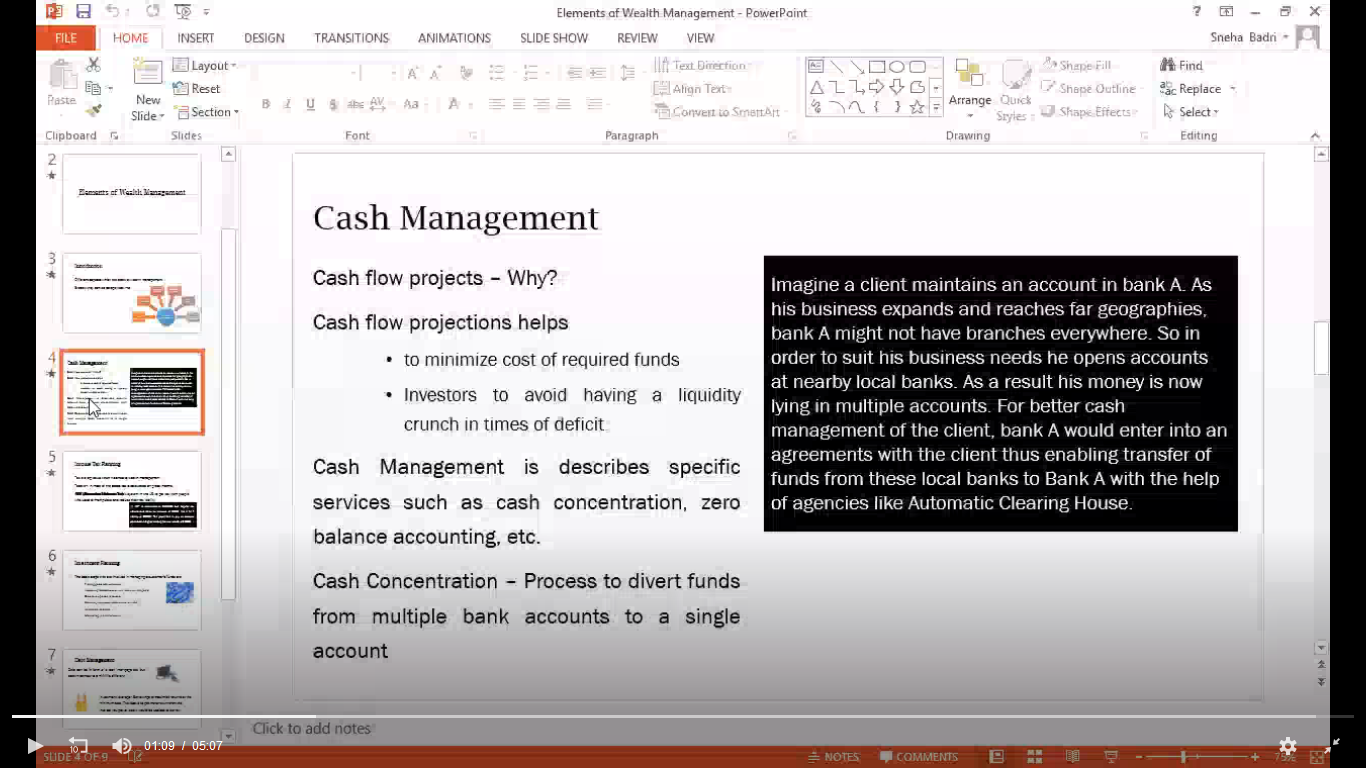
ix. Insurance transactions.

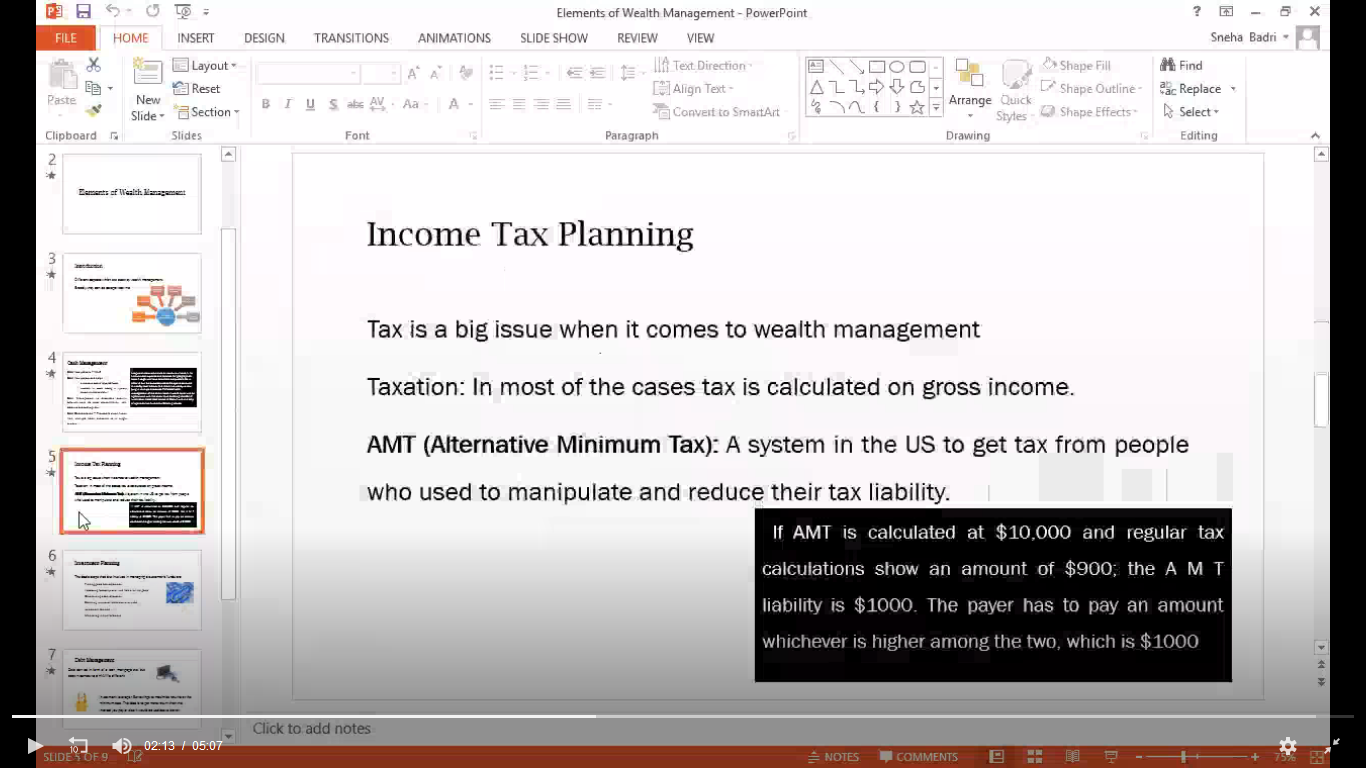
Two types of Banking

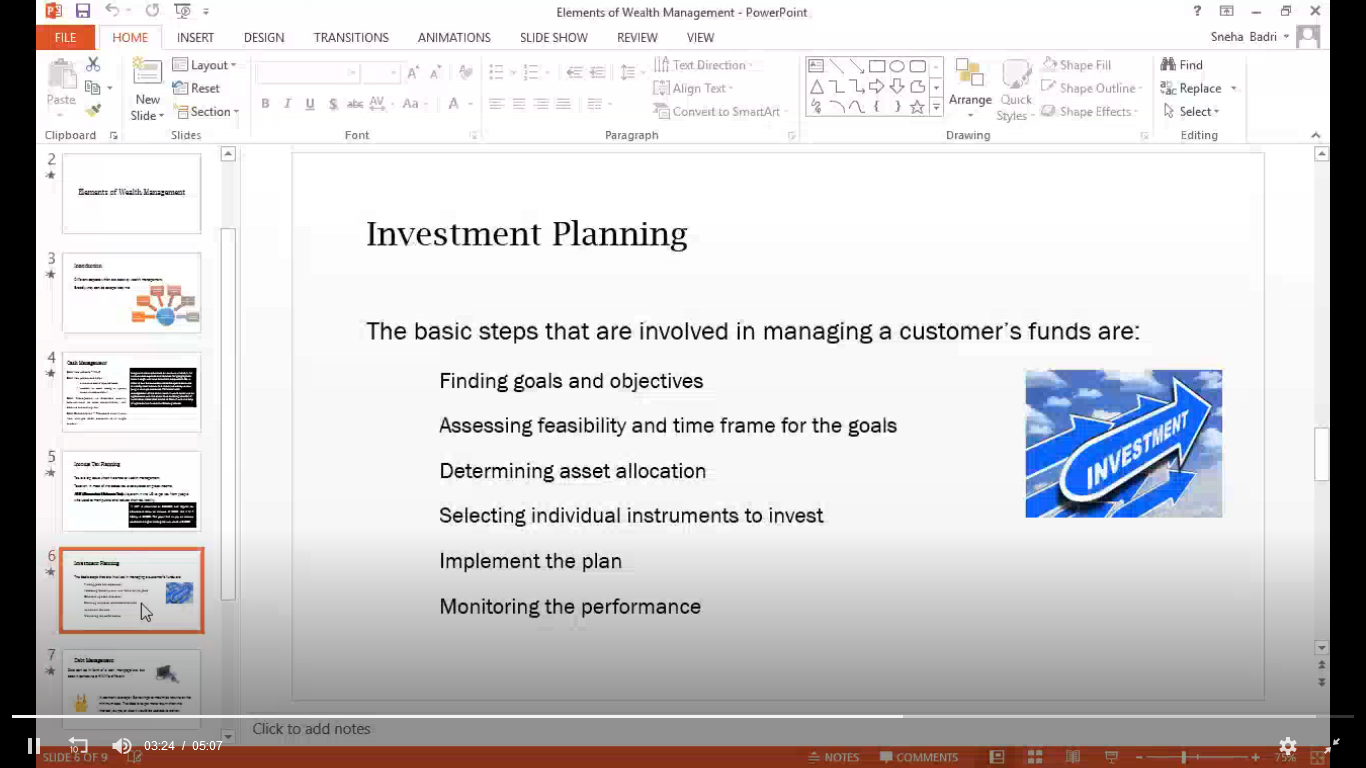
Private Banking and Wealth Management (Wings2 from ultimatix course id:59417)

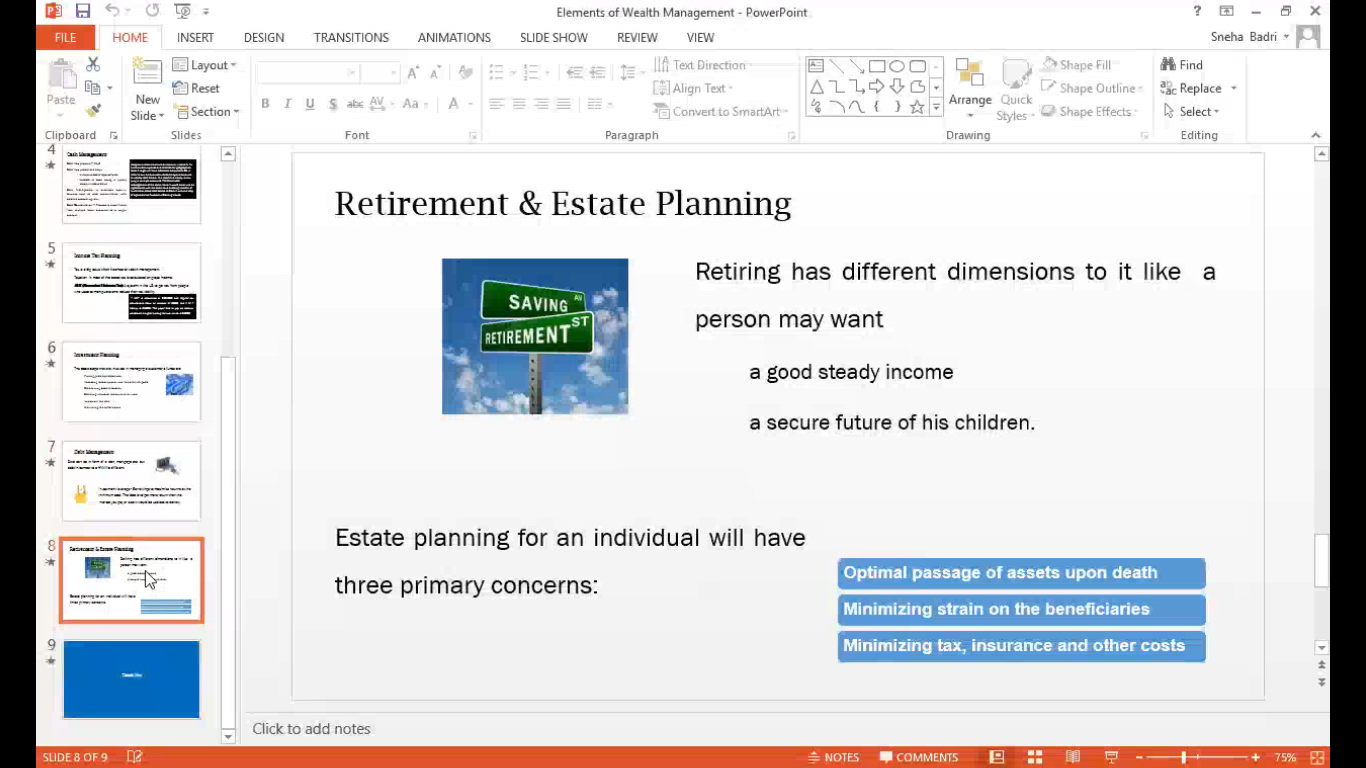
Private banking is for HNWI(High Net Worth of Individuals). Few banks offer services for these people.

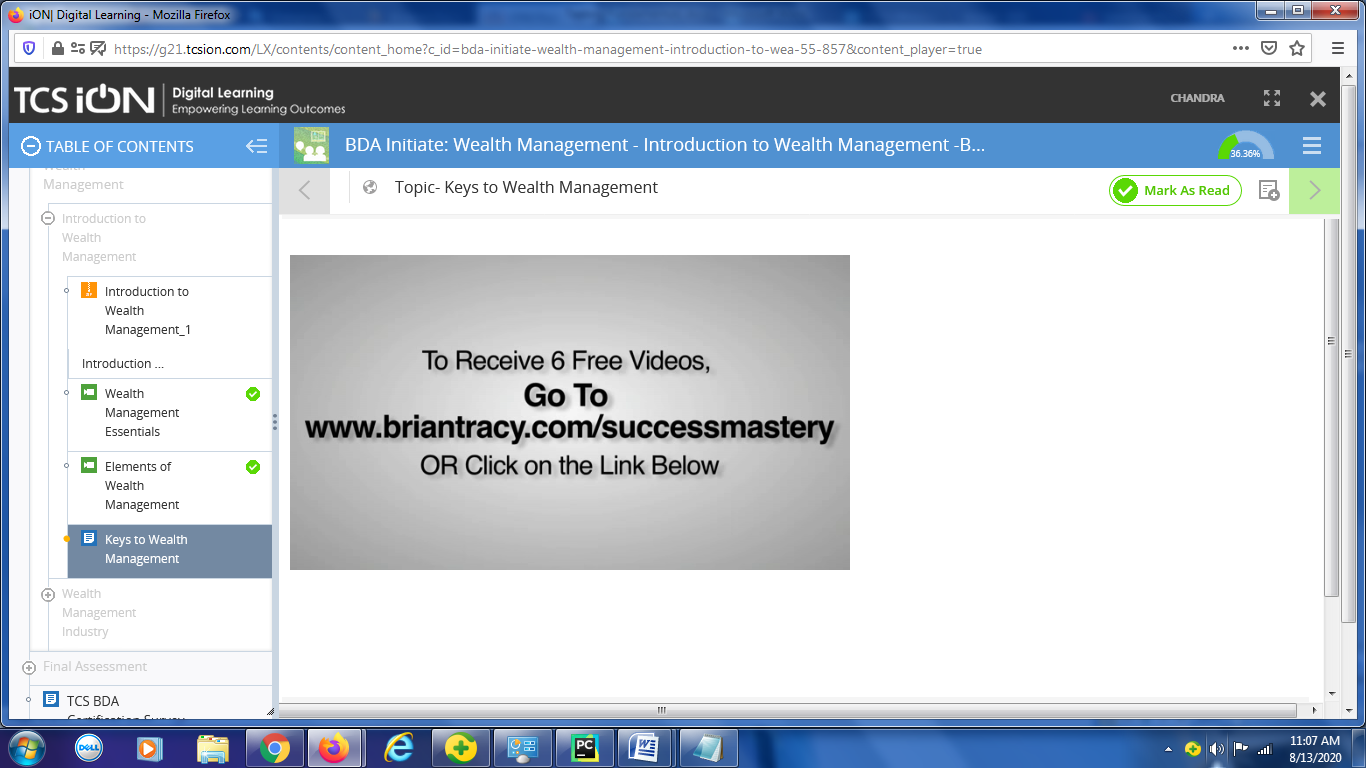
Wealth Management- Tax management, Risk Management, Cash management, Investment planning, retirement planning, Estate planning.



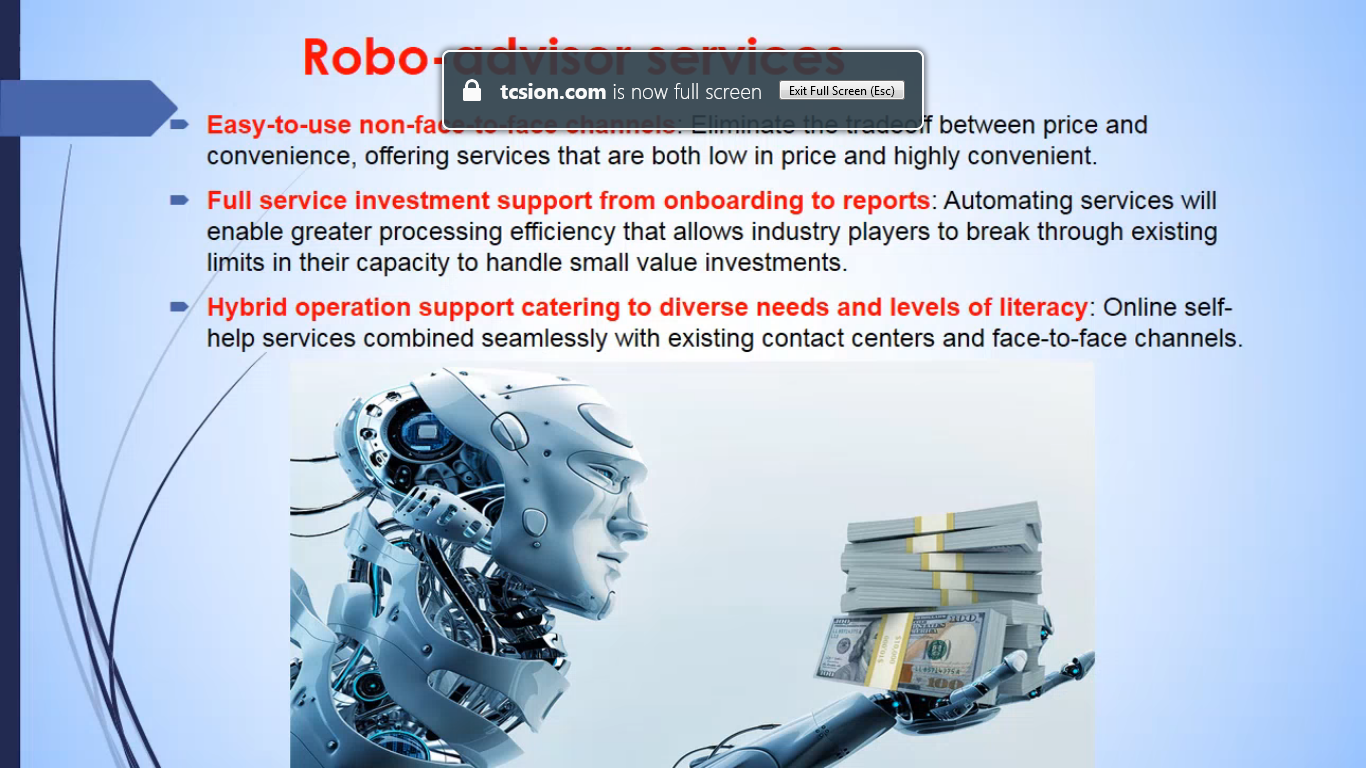














STLC (software test life cycle):

i. Test Requirement Analysis

ii. Test Planning

iii. Test Case preparation

iv. Test environment Setup

v. Test case execution

vi. Test Closure

User Stories-How will you derive the test cases

**Traceability matrix**

For Defect we attach test case in Jira. When we do this automatically that particular test case shows defect number.

Test case description should contain actual behaviour happening in the application and what is expected behaviour in the application. Expected is based on requirement.

Actual is wrong way of behaving and Expected is right way of the behaviour.

Negative testing is one approach for testing the negative scenarios.

What are the pre-conditions before running the tests:

For ex: Setting the parameters in patient central and then running the test. This could be changing the country, local settings and features for doctor.

Sprint release:

Once in a month.

Follow Agile process:

a. Scrum team.

b. Daily standup calls for statuses.

c. Updates on yesterday’s tasks and today’s tasks.

d. Scrum master records the session and sends the stats each day of each individual to the customer as well as team.

e. Demos to the customer about the fixes that were given by the dev team. The same will be noted by scrum master.

f. Process Improvements in the application.

g. Backlog refinements- product backlog and sprint backlogs. A product backlog is the one where the product owner gives the few selected requirements (not all) to the developers. These few selected requirements were kept in one folder and the dev team takes those requirements and will start working on these selected requirements.

This is nothing but product backlog. Sprint backlog is the developer chooses few requirements from the product backlog requirements and will keep in his/her folder which is nothing but the sprint backlog.

h. Retrospective meetings -What went wrong and what went right. Max 3hrs.

i. Noting down the steps for various functionalities so that repeatedly we need not ask questions.

**Product Backlog:**

Product Owner (Usually client) derives the requirements (Req1, Req2, Req3...Req100) from stakeholders, dev, marketing teams and scrum master. These will be kept in one folder. This folder is called product backlog.

**Sprint Backlog:**

Product Owner refines the product backlog (product backlog refinement) in-order to see which are feasible, flexible and easy to implement by Dev team. These will be kept in another folder and this becomes sprint backlog. Now the Sprint backlog is ready for Dev team and hence Sprint backlog is derived from product backlog.

Daily scrum should happen for 15 mins.

Scrum team:

a. Product Owner

b. Scrum Master

c. Development Team- Who has courage and move ahead. Includes BA, QA, Dev, Dev Validators, Scrum Master.

**Grooming session:**

Priority based user stories are discussed in grooming session and it becomes too lengthy at times. Instead the team discusses on product backlog refinement which is going to be implemented in next sprint.

Increment=Work done by previous sprint+Work done in current sprint.

Test Data:

Data used for executing test cases. For ex:username, environment(tjwl,djwl,ppr,dv1,dv2,sqa),

Username:<Dr. name>, pcuser(patient central user), country name, region,

Password:

Parameters:

**Challenges:**

a. Change in requirements such as change in translation texts, functionality changes at later stages or just before the release- This leads to changes in design, development, test cases, test data, information to be shared with multiple stake holders.

b. Issues found in production environments.

c. Misinterpretation of requirements.

d. Out-of-box thinking. This is when we need to work on a particular feature where minimal requirements were given by client and max output is expected. For example: A particular dialog box has to be popped up during particular screen for 3 mins and that pop up has mandatory fields (5-6 fields such as emailid, name, phone, address, country) which needs to be filled. So while filling the popup should not be closed and there should be some more waiting period. In this case the idle time should be considered and from that idle time the popup should be closed. Also the popup should not be visible to some users who are in Inactive state. These Limited users can login but cannot see the popup.

e. Keeping deadlines to complete the tasks.

f. Using production data in local. At times the data created in production is used in local environment to reproduce the issue. Then when some of the data such as production email is used, then mail will be sent to customer instead of receiving at the developer end.

g. Replication of issue in different environments/PCs. Possible reasons could be cache not cleared or at times need to try in incognito mode/private window, parameters/features are different.

h. At times we get illegal exceptions and null pointer exceptions. Need to repeat the steps for reproducing is a challenge.

i. Testing the application under deadlines

Risks:

Service Level Agreements:

STLC:

a. Analyzing the requirements

b. Deriving the Test cases from requirements and user stories

c. Executing the regression testing

d. Finding the defects, reporting the same in Jira, assign it to Dev. Tag the test case to defect.

e. Fail the test case.

f. Once the bug is fixed, retest the same in env. If it is fixed, close the defect with all the test data.

g. If not reopen the same.

**1)** important functionality of your project

**2)** high-risk modules of the project

**3)** which functionality is most visible to the user

**4)** which functionality has the largest safety impact

**5)** which functionality has the largest financial impact on users

**6)** which aspects of the application are most important to the customer

**7)** which parts of the code are most complex, and thus most subject to errors

**8)** which parts of the application were developed in rush or panic mode

**9)** what the developers think are the highest-risk aspects of the application

**10)** what kinds of problems would cause the worst publicity

**11)** what kinds of problems would cause the most customer service complaints

**12)** what kinds of tests could easily cover multiple functionalities