Software Quality Assurance and Testing (SEng4113) Lecture 04: Software Test Planning and Management

Belesti Y. (M.Sc.)

Debre Berhan University
Faculty of Computing
Department of Software Engineering

March 28, 2020

Outline

- Software test planning
- Software test plan components
- Software test management
- Software test management processes
- Software test management tools and techniques

Outline

- Test Planning
  - Test strategies
  - Schedule, Budgeting
  - Configuration management
- Test Design
  - Test scenarios
  - Test cases
  - Test data
  - Traceability metrics
- Test Execution
  - Build & release
  - Pre-QA checklist
  - Entry & exit criteria
  - Execution (Manual & Automated)
- Test Reporting
  - Test status reports
  - Test closure reports

Session Objective

- At the end of this session, students will be able to
  - Define software test plan
  - Explain importance of software test plan
  - Explain the components of software test plan
  - Describe how to develop software test plan

Software Test Planning

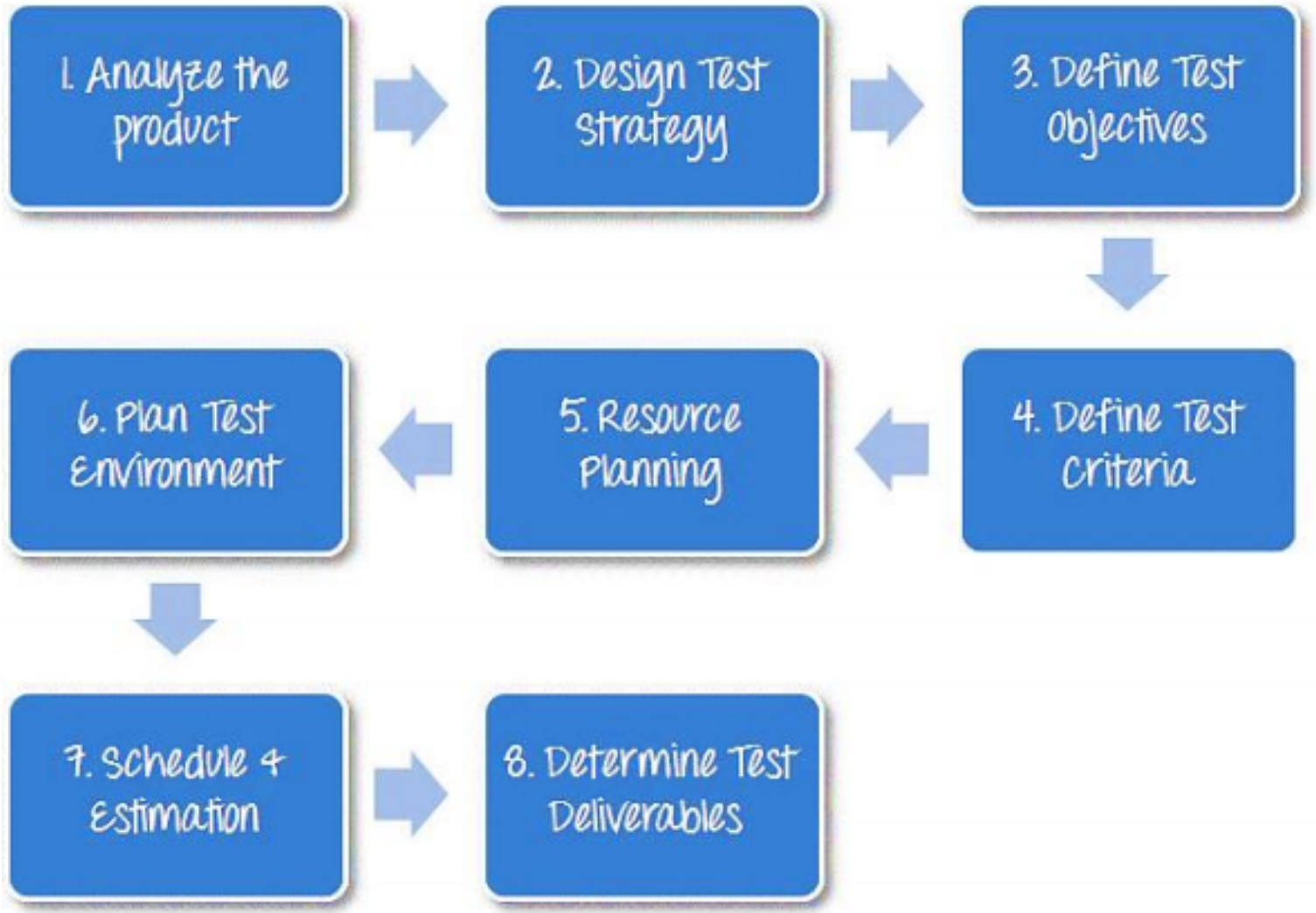
- Software test planning is the **practice of documenting software testing requirements in an organized manner**
- Its **deliverable is a test plan**
- A test plan is a **detailed document that outlines** the:
  - Test tasks, (what), test strategy, test approach (how)
  - Testing objectives (why), resources (manpower, software, hardware)
  - Test schedule (when), test estimation and test deliverables
- The test plan serves as a **blueprint** to conduct software testing activities as a **defined process**
  - Which is minutely **monitored and controlled by the test manager**

Importance of a Test Plan

1. Test plan helps us **determine the effort** needed to validate the quality of the application under test
2. Help people outside the test team such as developers, business managers, customers **understand the details of testing**
3. Test plan **guides our thinking**
   - It is like a rule book, which needs to be followed
4. Important aspects like test estimation, test scope, test strategy are documented in test plan, so it can be **reviewed by management team and re-used for other projects**

How to Write a Test Plan?

- Follow the **seven** steps below to create a test plan as per **IEEE 829**



How to Write a Test Plan? Analyze the Product

- How can you test a product without any information about it?
  - The answer is impossible
  - You must learn a product thoroughly before testing it
- You should **research clients and the end users** to **know their needs and expectations** from the application
  - Who will use the website?
  - What is it used for?
  - How will it work?
  - What software/ hardware the product uses?
- We have to **review product documentation** to understand all the **features** of the system as well as **how to use** it
  - If you are **unclear** on any items, you might **interview** customer, developer, designer to **get more information**

How to Write a Test Plan? Develop Test Strategy

- Test strategy is a **critical** step in making a test plan
- A test strategy document is a **high-level document** which is usually developed by test manager
- This document defines:
  - The projects **testing objectives and the means to achieve them**
  - Determines **testing effort and costs**
- You should follow **steps** below

How to Write a Test Plan? Develop Test Strategy: Define Scope of Testing

- Before the start of any testing activity, scope of the testing should be known. You must think hard about it.
  - The components of the system to be tested (hardware, software, middleware, etc.) are defined as "in scope"
  - The components of the system that will not be tested also need to be clearly defined as being "out of scope."
- Defining the scope of your testing project is very important for all stakeholders
- A precise scope helps you
  - Give everyone confidence & accurate information of the testing you are doing
  - All project members will have clear understanding about what is tested and what is not

How to Write a Test Plan? Develop Test Strategy: Define Scope of Testing

- To determine scope, you must have
  - Precise customer requirement
  - Project Budget
  - Product Specification
  - Skills & talent of your test team
- Now, you should clearly define the "**in scope**" and "**out of scope**" of the testing

How to Write a Test Plan? Develop Test Strategy: Identify Testing Type

- A testing type is a standard test procedure that gives an expected test outcome
- Each testing type is formulated to identify a specific type of product bugs
  - But, all testing types are aimed at achieving one common goal **early detection** of all the defects **before releasing the product to the customer**
- There are many testing types for testing software product
  - Your team cannot have enough efforts to handle all kind of testing
  - As test manager, you must set priority of the testing types
  - Which testing types should be focused for, for example, web application testing?
  - Which testing types should be ignored for saving cost?

How to Write a Test Plan? Develop Test Strategy: Document Risks and Issues

- Risk is futures uncertain event with a probability of occurrence and a potential for loss.
- When the risk actually happens, it becomes the issue.
- In the test plan, you will document those risks

Risk	Mitigation
Team member lack the required skills for website testing.	Plan training course to skill up your members
The project schedule is too tight; it's hard to complete this project on time	Set test priority for each of the test activity.
Test Manager has poor management skill	Plan leadership training for manager
A lack of cooperation negatively affects your employees' productivity	Encourage each team member in his task, and inspire them to greater efforts.
Wrong budget estimate and cost overruns	Establish the scope before beginning work, pay a lot of attention to project planning and constantly track and measure the progress

How to Write a Test Plan? Develop Test Strategy: Create Test Logistics

- In test logistics, the test manager should answer the following questions:
  - Who will test?
  - When will the test occur?
- You may not know exact names of the tester who will test, but the type of tester can be defined
- A Person having the following skills is most ideal for performing software testing:
  - Ability to understand customers point of view
  - Strong desire for quality
  - Attention to detail
  - Good cooperation

How to Write a Test Plan? Develop Test Strategy: Create Test Logistics

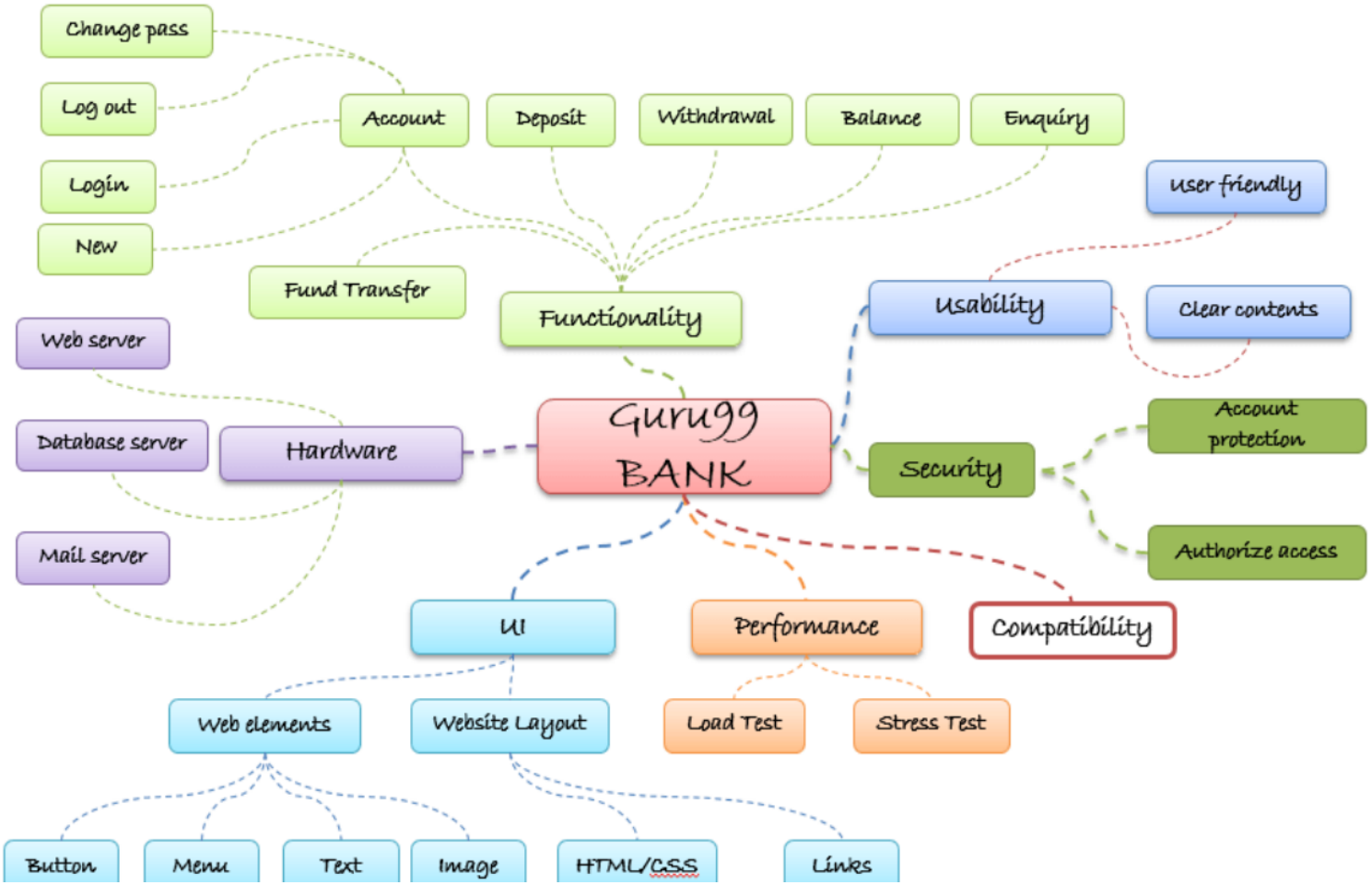
- When will the test occur?
- Test activities must be matched with associated development activities.
- You will start to test when you have all required items shown in following figure

How to Write a Test Plan? Define Test Objective

- Test objective is the overall goal and achievement of the test execution
- The objective of the testing is finding as many software defects as possible; ensure that the software under test is bug free before release
- To define the test objectives, you should do the following two steps:
  1. List all the software features (functionality, performance, GUI,) which will be tested
  2. Define the target or the goal of the test based on the above features

How to Write a Test Plan? Define Test Objective

- **Test objective of Guru99 Bank Testing Project**

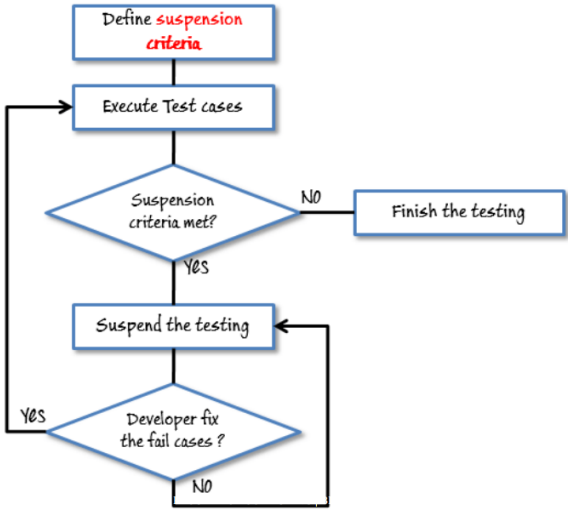


How to Write a Test Plan? Define Test Objective

- Based on above the features, you can define the test objective of the project guru99 as follows
  - Check that whether website guru99 functionality(account, deposit) is working as expected without any error or bugs in real business environment
  - Check that the external interface of the website such as UI is working as expected and meet the customer needs
  - Verify the usability of the website. Are those functionalities convenient for user or not?

How to Write a Test Plan? Define Test Criteria

- Test criteria is a standard or rule on which a test procedure or test judgment can be based
- Therere **two types of test criteria**:
  1. **Suspension criteria**: specify the critical suspension criteria for a test
     - ★ If the suspension criteria are met during testing, the active test cycle will be **suspended** until the criteria are resolved
     - ★ **Example**: if your team members report that 40% of test cases failed, you should suspend testing until the development team fixes all the failed cases



How to Write a Test Plan? Define Test Criteria

- Therere **two types of test criteria**:
  2. **Exit criteria**: specify the criteria that denote a successful completion of a test phase
     - ★ The exit criteria are the targeted results of the test and are necessary before proceeding the next phase of development
     - ★ **Example**: 95% of all critical test cases must pass
     - ★ Some methods of defining exit criteria are by specifying a targeted run rate and pass rate
  - **Run rate** is a ratio between number of test cases executed/total test cases of test specification
    - ★ For example, the test specification has a total of 120 TCs, but the tester only executed 100 TCs
    - ★ So, the run rate is 100/120 = 0.83 (83%)
  - **Pass rate** is ratio between numbers test cases passed / test cases executed
    - ★ For example, in the above example, 100 TCs executed and 80 TCs passed
    - ★ So, the pass rate is 80/100 = 0.8 (80%)

How to Write a Test Plan? Define Test Criteria

- This data can be retrieved in test metric documents
  - Run rate is mandatory to be 100%, otherwise a clear reason should be given
  - Pass rate is dependent on project scope, but achieving high pass rate should be a goal

How to Write a Test Plan? Define Test Criteria

- **Example**: Your team has already done the test executions. They report the test result to you, and they want you to confirm the Exit Criteria

- In this case, the run rate is mandatory to be 100%, but the test team only completed 90% of test cases
- It means the run rate is not satisfied, so do NOT confirm the Exit Criteria

How to Write a Test Plan? Resource Planning

- Resource plan is a detailed summary of all types of resources required to complete project task
  - Resource could be human, physical (equipment and materials) and financial needed to complete the test
- The resource planning is important factor of the test planning because it helps in determining the number of resources (employee, equipment) to be used for the project.
- Therefore, the test manager can make the correct schedule & estimation for the project
- This section represents the recommended resources for your project

How to Write a Test Plan? Resource Planning: Human Resource

- Members in your project team

No.	Member	Tasks
1.	Test Manager	Manage the whole project Define project directions Acquire appropriate resources
2.	Tester	Identifying and describing appropriate test techniques/tools/automation architecture Verify and assess the Test Approach Execute the tests, Log results, Report the defects. Tester could be in-sourced or out-sourced members, base on the project budget For the task which required low skill, I recommend you choose outsourced members to save project cost.
3.	Developer in Test	Implement the test cases, test program, test suite etc.
4.	Test Administrator	Builds up and ensures Test Environment and assets are managed and maintained Support Tester to use the test environment for test execution
5.	QA members	Take in charge of quality assurance Check to confirm whether the testing process is meeting specified requirements

How to Write a Test Plan? Resource Planning: Physical Resource

- **Example**: for testing a web application

No.	Resources	Descriptions
1.	Server	Install the web application under test This includes a separate web server, database server, and application server if applicable
2.	Test tool	The testing tool is to automate the testing, simulate the user operation, generate the test results There are tons of test tools you can use for this project such as Selenium, QTP…etc.
3.	Network	You need a Network include LAN and Internet to simulate the real business and user environment
4.	Computer	The PC which users often use to connect the web server

How to Write a Test Plan? Resource Planning: Financial Resource

- To estimate the financial requirement of a test, you need the following items:
  1. List of activities and their estimated direct and indirect costs must be calculated
  2. Items to be purchased and their estimated direct and indirect costs must be calculated
  3. Human resource to be hired
- There are three types of cost estimation techniques:
  1. **Preliminary** cost estimation - accuracy (-25%, +75%)
  2. **Budgetary** cost estimation - accuracy (-10%, +25%)
  3. **Definitive** cost estimation - accuracy (-5%, +10%)

How to Write a Test Plan? Plan Test Environment

- A testing environment is a setup of software and hardware on which the testing team is going to execute test cases
- The test environment consists of real business and user environment, as well as physical environments, such as server, front end running environment
- To finish this task, you need a strong cooperation between test team and development team
- You should ask the developer some **questions** to understand the web application under test clearly
  1. What is the maximum user connection which this website can handle at the same time?
  2. What are hardware/software requirements to install this website?
  3. Does the user's computer need any particular setting to browse the website?

How to Write a Test Plan? Schedule and Estimation

- In the test estimation phase, suppose you break out the whole project into small tasks and add the estimation for each task as shown below

Task	Members	Estimate effort
Create the test specification	Test Designer	170 man-hour
Perform Test Execution	Tester, Test Administrator	80 man-hour
Test Report	Tester	10 man-hour
Test Delivery		20 man-hour
Total		280 man-hour

- Then you create the schedule to complete these tasks
- Making schedule is a common term in project management
- By creating a solid schedule in the test planning:
  - The test manager can use it as tool for monitoring the project progress, control the cost overruns

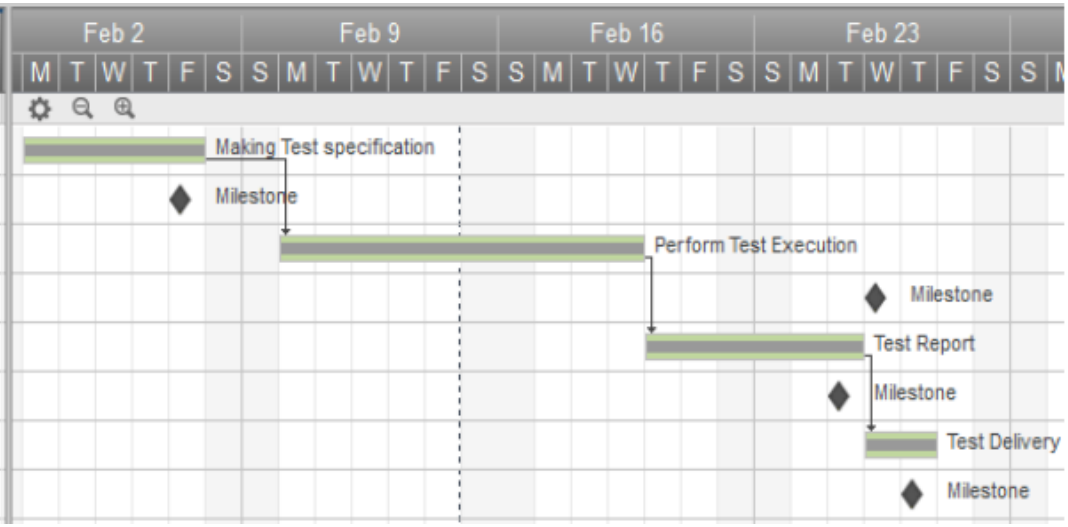
How to Write a Test Plan? Schedule and Estimation

- To create the project schedule, the test manager needs several types of input such as:
  - **Employee and project deadline**: the working days, the project deadline, resource availability are the factors which affected to the schedule
  - **Project estimation**: based on the estimation, the test manager knows how long it takes to complete the project (Expected time calculation)
    - ★ So, he can make the appropriate project schedule
  - **Project risk**: understanding the risk helps test manager add enough extra time to the project schedule to deal with the risks. (Buffer)

How to Write a Test Plan? Schedule and Estimation

- Suppose the boss wants to complete the project in one month, you already estimated the effort for each tasks in Test Estimation
  - You can create the schedule as shown below. (Gantt Chart)

Task Name	Start Date	End Date
Making Test specification	**02/03/14**	**02/07/14**
Milestone	02/07/14	02/07/14
Perform Test Execution	**02/10/14**	**02/19/14**
Milestone	02/26/14	02/26/14
Test Report	**02/20/14**	**02/25/14**
Milestone	02/25/14	02/25/14
Test Delivery	**02/26/14**	**02/27/14**
Milestone	02/27/14	02/27/14



How to Write a Test Plan? Test Deliverables

- Test deliverables are list of all the documents, tools and other components that have to be developed and maintained in support of the testing effort.
- There are different test deliverables at every phase of the software development lifecycle.
- Test Deliverables can be **grouped into three**
  1. Deliverables provided **before** testing
     - ★ Test plans document
     - ★ Test cases documents
     - ★ Test design specifications

How to Write a Test Plan? Test Deliverables

- Test Deliverables can be **grouped into three**
  2. Deliverables provided **during** testing
     - ★ Test scripts
     - ★ Simulators
     - ★ Test data
     - ★ Test traceability matrix
     - ★ Error logs and execution logs
  3. Deliverables provided **after** testing
     - ★ Test results/reports
     - ★ Defect report
     - ★ Release notes

How to Write a Test Plan? Test Plan Types

- **Test plan** is a **guideline** based on which test execution should be tracked
  - For successful testing and good product test delivery, it is important to **update and make required changes** in the plan as per **changes** in the any of the parameter which was **basis** of the test plan
- One can have the following **types of test plans**:
  1. **Master Test Plan**: a single high-level test plan for a project/ product that unifies all other test plans.
  2. **Testing-Level Specific Test Plans**: Plans for each level of testing
     - ★ Unit Test Plan
     - ★ Integration Test Plan
     - ★ System Test Plan
     - ★ Acceptance Test Plan
  3. **Testing-Type Specific Test Plans**: Plans for major types of testing like Performance Test Plan, Security Test Plan, etc

How to Write a Test Plan? Outline of a Test Plan

- Test plan format and content may vary depending upon different standards
- A test plan shall have the following structure:
  - Test plan identifier
  - Introduction
  - Test items
  - Features to be tested
  - Features not to be tested
  - Approach
  - Item pass/fail criteria
  - Suspension criteria and resumption requirements
  - Test deliverables
  - Testing tasks
  - Environmental needs
  - Roles and Responsibilities
  - Staffing and training needs
  - Schedule
  - Risks and contingencies
  - Approvals

How to Write a Test Plan? Outline of a Test Plan - Description

- **Test Plan Identifier**: Provides a unique identifier for the document
  - Every deliverable has a unique identification number which could be numeric or alphanumeric based on the company configuration management
- **Introduction**: Brief introduction about the project and objective of the current release
  - Project could be platform configuration tool and objective could new mobile App interface or new feature / enhancement in existing product or defect fixes
- **Test item**: Introduction and overview of Software Under Test
- **Features to test**: In scope features
  - This could be newly added or updated features
  - Indirect features that have technical or functional dependency on newly added or updated features

How to Write a Test Plan? Outline of a Test Plan - Description

- **Features not to test**: Out of scope feature
  - Excluded product features from current Test Plan
  - **You have to provide reason for exclusion**, like, non-impacted / less impacted / less priority features, as applicable
- **Approach**: Strategy to test the software
  - Includes types of tests and how to test
    - ★ Functional, performance, security testing
    - ★ Using manual only, automation only or combined [manual+automation] approach
- **Test deliverables**: All the deliverables from the testing e.g. approaches, test cases, reports etc.
- **Item pass/fail criteria**: Entry and Exit criteria for all items
  - Test Case: All Steps passed
  - Feature: All test cases executed and no defects are detected.
- **Testing tasks**: All tasks / steps to execute for test planning and execution
- **Environmental needs**: Infrastructure required for testing

How to Write a Test Plan? Outline of a Test Plan - Description

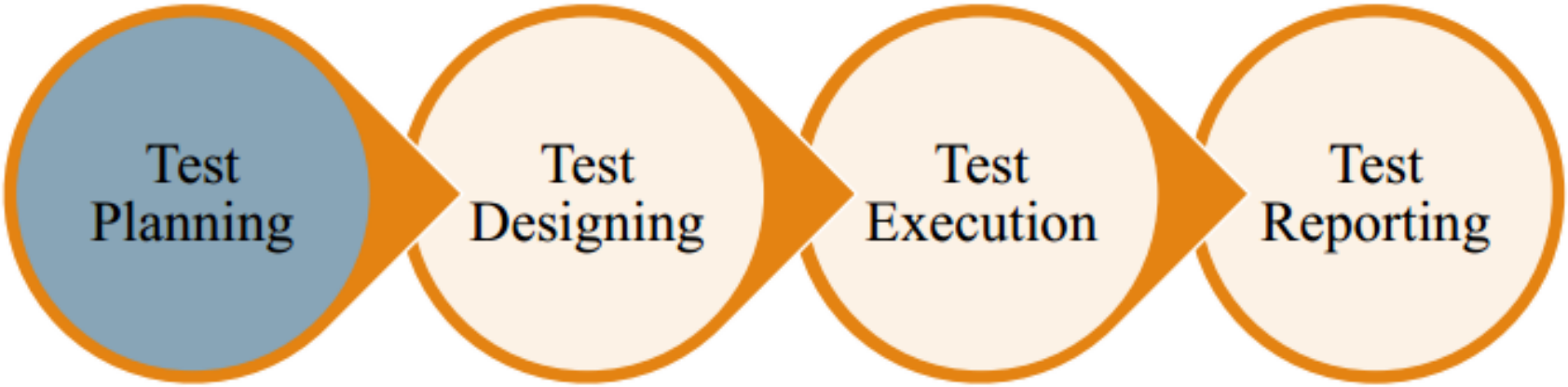
- **Responsibilities**: Roles and responsibilities for various testing / supported activities
- **Staffing and training needs**: Training / hiring needs to bridge the gap of available and expected skill
- **Schedule**: Test estimation (Efforts) and high-level schedule. Schedule should be for key deliverables or important milestones
  - Ideally, all test deliverables included in the test plan should be scheduled
  - Detailed test schedule (at feature or defects or resource level) is prepared at appropriate time during test execution
- **Risks and Mitigation**: Risk identification for applicable items, assumptions, and mitigation plan
- **Approvals**: Approvals and sign of dates.

Test Management

- Test management is a series of planning, execution, monitoring and control activities that help achieve project goals
- Test management is the practice of organizing and controlling the process and artifacts required for the testing effort throughout the project cycle
- The goal of test management is to allow teams to plan, develop, execute and assess all testing activities within the overall software development effort
- Through well-planned and well-managed testing processes, teams can ensure that they produce the best products possible, while making the most of their limited resources
- A test management tool is software used to manage tests (automated or manual) that have been previously specified by a test procedure

Test Management

- **Processes**



Test Management

Test Planning

- Test planning is the processing of preparing a test plan
- A test plan is a detailed document that outlines:
  - The test strategy, testing objectives
  - resources (manpower, software, hardware) required for testing
  - test schedule, test estimation and test deliverables
- It is intended to determine:
  - How the test will be done?
  - Test strategy
  - What test methodologies will be followed?
  - When to stop testing?
  - Resource requirements (HW, SW, Budget & Time)
- **Already covered in the previous session**

Test Management

Test Design

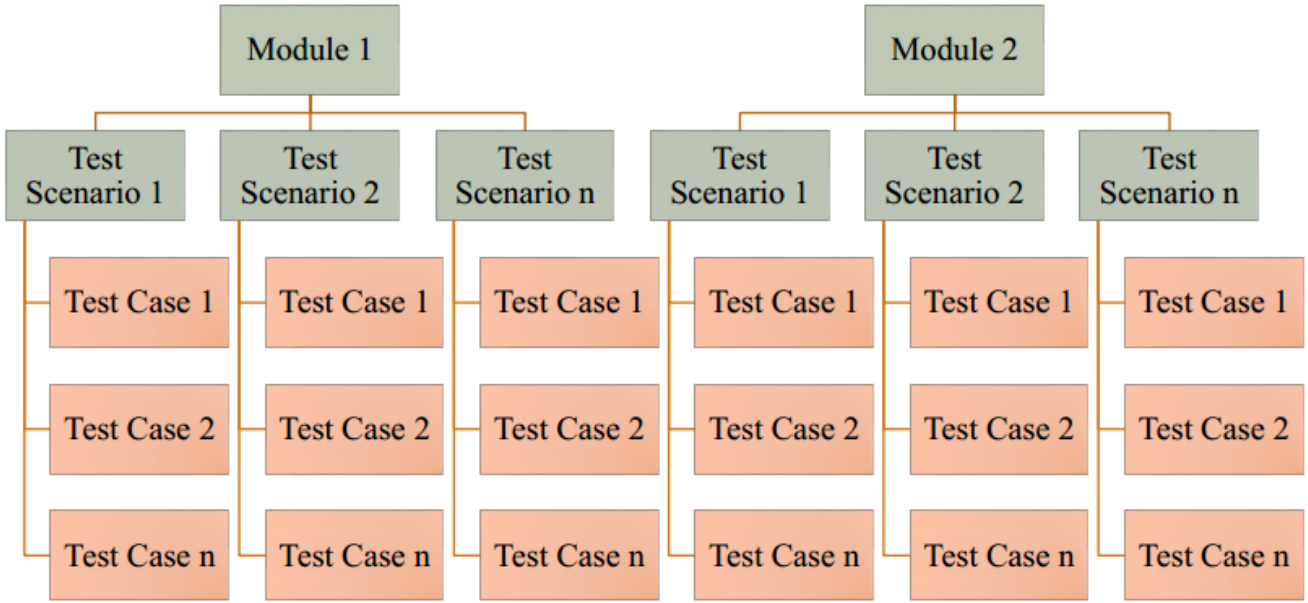
- Test design is the act of creating and writing test artifacts for testing a software
  - Tasks include:
    - ★ Identify test basis
    - ★ Develop test scenarios
    - ★ Identify and describe test cases
    - ★ Develop test suite
    - ★ Identify and structure test scripts
    - ★ Review and access test coverage using traceability Matrix

Test Design: Test Basis

- Test basis is defined as the source for creation of scenarios and test cases
- It can be
  - The application itself or
  - The requirement documents like
    - ★ SRS (Software Requirement Specification)
    - ★ BRS (Business Requirement Specification), etc

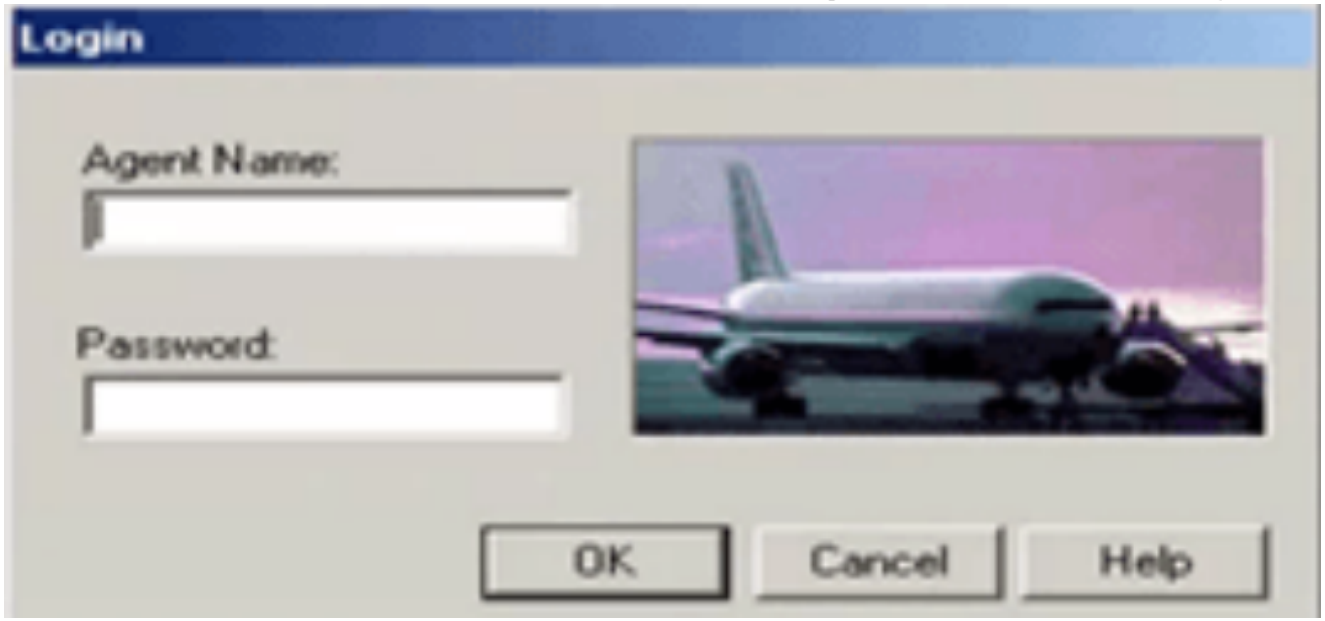
Test Design: Test Scenario

- A test scenario is any functionality that can be tested
  - It is also called test condition or test possibility
- As a tester, you may put yourself in the end users shoes and figure out the real-world scenarios and use cases of the Application Under Test (AUT)
- Test scenario is what to be tested and a test case is how to be tested



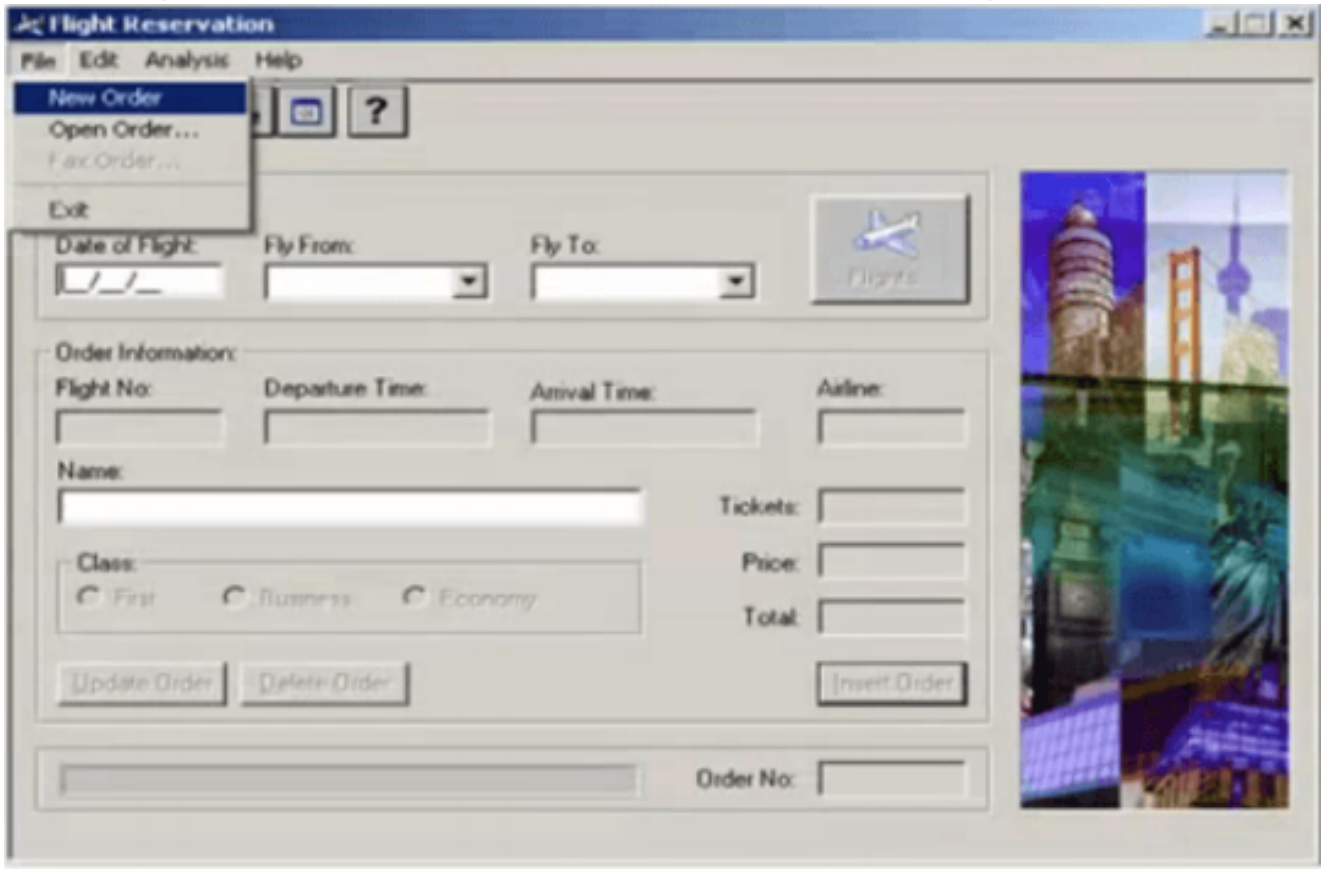
Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 2**: Check the **Login** Functionality



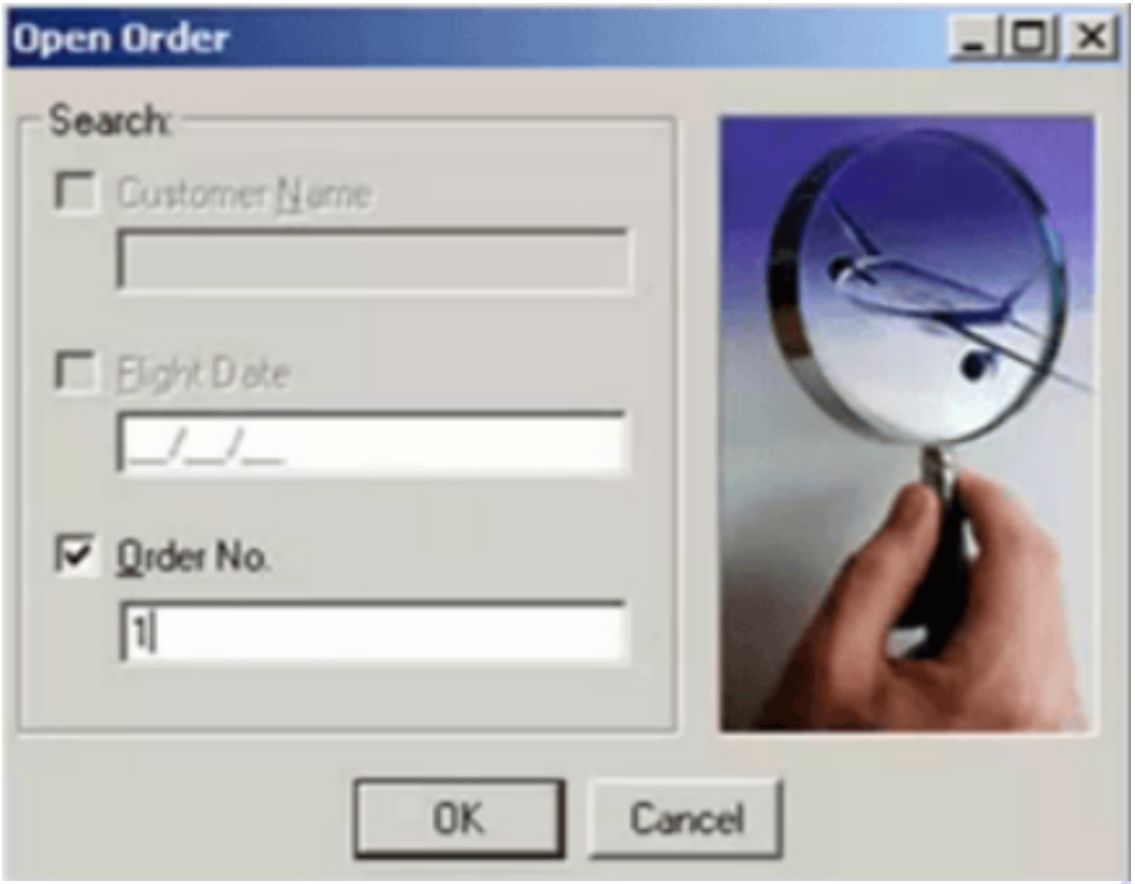
Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 2**: Check that a **New Order** can be created



Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 3**: Check that an **existing Order can be opened**



Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 4**: Check that a user, can **FAX an order**

Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 5**: Check that the information displayed in the **HELP section is correct**

Test Design: Test Scenario

- **Example**: Test Scenario for Flight Reservation
- For the Flight Reservation Application, a few test scenarios would be:
- **Test Scenario 6**: Check that the information displayed in **About** section, like version, programmer name, copy right information is **correct**

Test Design: Test Scenario

- Apart from these six scenarios, here is the list of all other scenarios
  - Update Order
  - Delete Order
  - Check Reports
  - Check Graphs and so on.
- We have already learned **exhaustive testing is not possible**.
- Suppose you have time only to execute 4 out of those 6 scenarios
  - Which two low priority scenarios of these six will you eliminate?
    - ★ I am sure most of you would have guessed scenarios 5 & 6
    - ★ Since they are **not the core functionality** of the application
    - ★ This is nothing but **Test Prioritization**

Test Design: Test Scenario

- **Example**: Test Scenarios for a Banking Site
  - **Test Scenario 1**: Check the **Login** and **Authentication** functionality
  - **Test Scenario 2**: Check Money **Transfer** can be done
  - **Test Scenario 3**: Check Account Statement can be **viewed**
  - **Test Scenario 4**: Check Fixed Deposit/Recurring Deposit can be **created**
  - And so on ...

Test Design: Test Scenario - Why Create Test Scenarios?

- Test scenarios are created for the following reasons:
  - Creating test scenarios ensures complete test coverage
  - Test scenarios can be approved by various stakeholders like business analyst, developers, customers to ensure the application under test is thoroughly tested
    - ★ It ensures that the software is working for the most common use cases
  - They serve as a quick tool to determine the testing work effort and accordingly create a proposal for the client or organize the workforce
  - They help determine the most important end-to-end transactions or the real use of the software applications

Test Design: Test Scenario - When not Create Test Scenarios?

- Test scenarios may not be created when:
  - The application under test is complicated, unstable and there is a time crunch in the project
  - Projects that follow agile methodology like Scrum, Kanban may not create test scenarios
  - Test scenario may not be created for a new bug fix or regression testing
    - ★ In such cases, test scenarios must be already heavily documented in the previous test cycles
    - ★ This is especially true for maintenance projects

Test Design: Test Scenario - How to Create a Test Scenario?

- As a tester, you can follow these five steps to create Test Scenarios
  1. **Step 1**: Read the Requirement Documents like BRS, SRS of the Application Under Test (AUT)
     - ★ You could also refer uses cases, books, manual, etc. of the application to be tested
  2. **Step 2**: For each requirement, figure out possible users actions and objectives
     - ★ Determine the technical aspects of the requirement
     - ★ Ascertain possible scenarios of system abuse and evaluate users with hacker's mindset
  3. **Step 3**: After reading the requirements document and doing your due analysis:
     - ★ List out different test scenarios that verify each feature of the software
  4. **Step 4**: Once you have listed all possible Test Scenarios,
     - ★ A **Traceability Matrix** is created to verify that each & every requirement has a corresponding Test Scenario
  5. **Step 5**: The scenarios created are reviewed by your supervisor
     - ★ Later, they are also reviewed by other stakeholders in the project

Test Design: Test Case

- A test case is a set of actions executed to verify a particular feature or functionality of your software application
- A test case is a document which consists of a set of conditions or actions which are performed on the software application in order to verify the expected functionality of the feature
- Here we describe the end to end logical flow of a specific requirement with test data, prerequisites and expected results
- Now, consider the Test Scenario Check Login functionality there many possible cases like
  - Test Case 1: Check results on entering valid User Id & Password
  - Test Case 2: Check results on entering Invalid User ID & Password
  - Test Case 3: Check response when User ID is Empty & Login Button is pressed, and many more
- Test scenarios are rather vague and cover a wide range of possibilities
  - Testing is all about being very specific

Test Design: Test Case

- **Format of Standard Test Case**

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
TU01	Check Customer Login with valid Data	1. Go to site http://demo.guru99.com 1. Enter UserId 2. Enter Password 3. Click Submit	Userid = guru99 Password = pass99	User should Login into application	As Expected	Pass
TU02	Check Customer Login with invalid Data	1. Go to site http://demo.guru99.com 1. Enter UserId 2. Enter Password 3. Click Submit	Userid = guru99 Password = glass99	User should not Login into application	As Expected	Pass

- Test cases **shall include the following components**

Test Design: Test Case

- Test cases **shall include the following components**

Test Case Field	Description
Test case ID:	•Each test case should be represented by a unique ID. To indicate test types follow some convention like "TC_UI_1" indicating "User Interface Test Case#1."
Test Priority:	•It is useful while executing the test. • Low • Medium • High
Name of the Module:	•Determine the name of the main module or sub-module being tested
Test Designed by:	•Tester's Name
Date of test designed:	•Date when test was designed
Test Executed by:	•Who executed the test- tester
Date of the Test Execution:	•Date when test needs to be executed
Name or Test Title:	•Title of the test case
Description/Summary of Test:	•Determine the summary or test purpose in brief
Pre-condition:	•Any requirement that needs to be done before execution of this test case. To execute this test case list all pre-conditions
Dependencies:	•Determine any dependencies on test requirements or other test cases
Test Steps:	•Mention all the test steps in detail and write in the order in which it requires to be executed. While writing test steps ensure that you provide as much detail as you can
Test Data:	•Use of test data as an input for the test case. Deliver different data sets with precise values to be used as an input
Expected Results:	•Mention the expected result including error or message that should appear on screen
Post-Condition:	•What would be the state of the system after running the test case?
Actual Result:	•After test execution, actual test result should be filled
Status (Fail/Pass):	•Mark this field as failed, if actual result is not as per the estimated result
Notes:	•If there are some special condition which is left in above field

Test Design: Test Case - Characteristics of Good Test Case

1. **Test Cases need to be simple and transparent**
   - Create test cases that are as simple as possible
   - They must be clear and concise as the author of test case may not execute them
   - Use assertive language like go to home page, enter data, click on this and so on
   - This makes the understanding the test steps easy and test execution faster
2. **Create Test Case with End User in Mind**
   - Ultimate goal of any software project is to create test cases that meets customer requirements and is easy to use and operate
   - A tester must create test cases keeping in mind the end user perspective

Test Design: Test Case - Characteristics of Good Test Case

3. **Avoid test case repetition**
   - Do not repeat test cases
   - If a test case is needed for executing some other test case, call the test case by its test case id in the precondition column
4. **Do not Assume**
   - Do not assume functionality and features of your software application while preparing test case
   - Stick to the Specification Documents

Test Design: Test Case - Characteristics of Good Test Case

5. **Ensure 100% Coverage**
   - Make sure you write test cases to check all software requirements mentioned in the specification document
   - Use Traceability Matrix to ensure no functions/conditions is left untested
6. **Test Cases must be identifiable**
   - Name the test case id such that they are identified easily while tracking defects or identifying a software requirement at a later stage
7. **Implement Testing Techniques**
   - It's not possible to check every possible condition in your software application
   - Testing techniques help you select a few test cases with the maximum possibility of finding a defect
     - ⋆ **Example**: BVA, EP, etc.

Test Design: Test Case - Characteristics of Good Test Case

8. **Self cleaning**
   - The test case you create must return the Test Environment to the pre-test state and should not render the test environment unusable
   - This is especially true for configuration testing
9. **Repeatable and self-standing**
   - The test case should generate the same results every time no matter who tests it
10. **Peer Review**
    - After creating test cases, get them reviewed by your colleagues.
    - Your peers can uncover defects in your test case design, which you may easily miss

Test Design: Test Case - Types of Test Cases

- **Two types** of test cases
  1. **Formal test cases**: Formal test cases are those test cases which are authored as per the test case format
     - ⋆ It has all the information like preconditions, input data, output data, post conditions, etc
     - ⋆ It has a defined set of inputs which will provide the expected output
  2. **Informal test cases**: Informal test cases are authored for such requirements where the exact input and output are not known
     - ⋆ In order to test them the formal test cases are not authored but the activities will be done and the outcomes are reported once the tests are run

Test Management

Test Design: Test Suit

- In software development, a test suite is a **collection of test cases** that are intended to be used to test a software program to show that it has some specified set of behaviors
- Test suite is a container that has a set of tests which helps testers in executing and reporting the test execution status
- A test suite is a collection of test cases that are grouped for test execution purposes

Test Design: Test Script

- A test script is a set of instructions (written using a scripting/programming language) that is performed on a system under test to verify that the system performs as expected
- Test scripts are used in automated testing

Test Design: Test Data

- Test data is data which has been specifically identified for use in tests
  - Some data may be used in a confirmatory way, typically to verify that a given set of input to a given function produces some expected result
  - Some other data may be used in order to challenge the ability of the program to respond to unusual, extreme, exceptional, or unexpected input
- Test data may be produced in a focused or systematic way (as is typically the case in domain testing), or
  - By using other, less-focused approaches (as is typically the case in high-volume randomized automated tests)
- Test data may be produced by the tester, or by a program or function that aids the tester
  - Test data may be recorded for re-use, or used once and then forgotten

Test Design: Test Data

- Test data is the documented data that is basically used to test the software program
- Test data is divided into two categories
  - Positive test data which is generally gives to system to generate the expected result
  - Negative test data which is used to test the unhandled conditions, unexpected, exceptional or extreme input conditions
- If the test data is inadequately designed, then such test inputs will not cover all possible test scenarios, which impact the quality of the software application under test

Test Design: Test Data

- Test data can be documented in any manner Excel Sheet, Word Document, Text file and many-more
- The data stored in an Excel Sheet can be entered by hand while running test cases or can be examined automatically from files (XML, Flat Files, Database etc.) using automation tools
- Using test data, you can verify the expected result and the software behavior of invalid input data
- It is also used in order to challenge the ability of the application to respond to unusual, extreme, exceptional, or unexpected input

Test Design: Test Data - Limitations

- It is not always possible to produce enough data for testing
- The amount of data to be tested is determined or limited by considerations such as time, cost and quality.
- Time to produce, cost to produce and quality of the test data, and efficiency

Test Design: Test Data - Guidelines in Generating Test Data

1. **The best test data**: Try to create the best data set that should not be so long and identifies bugs of all kind of applications functions but does not exceed cost and time limitation for preparing test data and running tests
2. **Unlawful data set-up**: Create wrong data set format
   - This invalid or dishonest format of data cannot be accepted by system and generates an error message
   - Check that, it has to generate an error message
3. **Boundary condition data set**: Data set holding out of range data
   - Recognize application boundary cases to organize data set that will cover lower as well as upper boundary conditions

Test Design: Test Data - Guidelines in Generating Test Data

4. **Correct dataset**: Create correct data set to:
   - Ensure that an application is responding as per the requirement or not and
   - Know that the data is correctly saved in a database/file or not
5. **Incorrect data set**: Create incorrect data set to
   - Confirm application behavior for negative values, alphanumeric string inputs
6. **Create large data set for performance, load and stress testing, and regression testing**: Large amount of data set is required for these kinds of testing
   - To do the performance testing for the DB application that fetches/updates data from/to DB table large data set is required

Test Design: Test Data - Guidelines in Generating Test Data

7. **Blank or default data**: Execute your test cases in blank or default data set environment to
   - Check that correct error messages are generated or not
8. **Check the corrupted data**: Fill a bug after correct troubleshooting
   - Before running test case on a particular data, you shall ensure that the data is not corrupted

Test Design: Traceability Matrix

- A traceability matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship
- It is used to track the requirements and to check the current project requirements are met
- Requirement traceability matrix (RTM) captures all requirements proposed by the client or software development team and their traceability in a single document delivered at the conclusion of the life-cycle
- In other words, it is a document that maps and traces user requirement with test cases
  - The main purpose of requirement traceability matrix is to see that all test cases are covered so that no functionality should miss while doing software testing

Test Design: Requirements Traceability Matrix

- Requirement Traceability Matrix; parameters include:
  - Requirement ID
  - Risks
  - Requirement Type and Description
  - Trace to design specification
  - Unit test cases
  - Integration test cases
  - System test cases
  - User acceptance test cases
  - Trace to test script

Test Design: Traceability Test Matrix

- Sample traceability matrix

Requirement Identifiers	Reqs Tested	REQ1 UC 1.1	REQ1 UC 1.2	REQ1 UC 1.3	REQ1 UC 2.1	REQ1 UC 2.2	REQ1 UC 2.3.1	REQ1 UC 2.3.2	REQ1 UC 2.3.3	REQ1 UC 2.4	REQ1 UC 3.1	REQ1 UC 3.2	REQ1 TECH 1.1	REQ1 TECH 1.2	REQ1 TECH 1.3
Test Cases	321	3	2	3	1	1	1	1	1	1	2	3	1	1	1
Tested Implicitly	77														
1.1.1	1	x													
1.1.2	2		x	x											
1.1.3	2	x											x		
1.1.4	1			x											
1.1.5	2	x												x	
1.1.6	1		x												
1.1.7	1			x											
1.2.1	2				x		x								
1.2.2	2					x		x							

Test Design: Types of Traceability Test Matrix

- **Forward traceability**: This matrix is used to check whether the project progresses in the desired direction and for the right product
  - It makes sure that each requirement is applied to the product and that each requirement is tested thoroughly
  - It maps requirements to test cases.
- **Backward or reverse traceability**: It is used to ensure whether the current product remains on the right track
  - The purpose behind this type of traceability is to verify that we are not expanding the scope of the project by adding code, design elements, test or other work that is not specified in the requirements
  - It maps test cases to requirements.
- **Bi-directional traceability (Forward+Backward)**: This traceability matrix ensures that all requirements are covered by test cases
  - It analyzes the impact of a change in requirements affected by the defect in a work product and vice versa

Test Design: Advantage of Traceability Test Matrix

- 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

Test Management

Test Execution

- Once the preparation of test case development and test environment setup is completed, then test execution phase can be kicked off
- Test execution is the process of executing the code and comparing the expected and actual results
- In this phase, testing team start executing test cases based on prepared test planning & prepared test cases in the prior step
- Once the test case is passed then same can be marked as passed
- If any test case is failed then corresponding defect can be reported to developer team via bug tracking system & bug can be linked for corresponding test case for further analysis

Test Management

Test Execution

- Ideally, every failed test case should be associated with at least single bug
- Once the bug fixed by development team then same test case can be executed based on your test planning.
- If any of the test cases are blocked due to any defect then such test cases can be marked as Blocked, so we can get the report based on how many test cases passed, failed, blocked or not run etc.
- Once the defects are fixed, same Failed or Blocked test cases can be executed again to re-test the functionality

Test Execution

- A

Entry Criteria	Activity	Exit Criteria	Deliverables
• Baselined RTM, Test Plan, Test case/scripts are available • Test environment is ready • Test data set up is done	• Execute tests as per plan • Document test results, and log defects for failed cases • Update test plans/test cases, if necessary Map defects to test cases in RTM • Retest the defect fixes • Regression Testing of the application • Track the defects to closure	• All tests planned are executed • Defects logged and tracked to closure	• Completed RTM with execution status • Test cases updated with results • Defect reports

Test Management

Test Reporting and Cycle Closure

- Call out the testing team member meeting & evaluate cycle completion criteria based on test coverage, quality, cost, time, and critical business objectives
- Discuss what all went good, which area needs to be improved & taking the lessons from current STLC as input to upcoming test cycles, which will help to improve bottlenecks in the STLC process.
- Test cases & bug reports will analyze to find out the defect distribution by type and severity .
- Once complete the test cycle then test closure report & test metrics will be prepared.

Test Reporting and Cycle Closure

- A

Entry Criteria	Activity	Exit Criteria	Deliverables
• Testing has been completed • Test results are available • **Defect report**	• Evaluate cycle completion criteria • Prepare test metrics • Document the learning out of the project • Prepare test closure report • Qualitative and quantitative reporting of quality of the work product to the customer. • Test result analysis to find out the defect distribution by type and severity	• Test closure report signed off by clients	• Test closure report • Test metrics

Test Reporting and Cycle Closure - Defect Reporting

- When a tester executes the test cases, he might come across the test result which is contradictory to expected result
- This variation in the test result is referred as a software defect
- While reporting the defect to developer, your defect report should contain the following information:
  - Defect_ID - Unique identification number for the defect.
  - Defect Description - Detailed description of the Defect including information about the module in which Defect was found.
  - Version - Version of the application in which defect was found
  - Steps - Detailed steps along with screenshots with which the developer can reproduce the defects

Test Reporting and Cycle Closure - Defect Reporting

- While reporting the defect to developer, your defect report should contain the following information:
  - Date Raised - Date when the defect is raised
  - Reference- where in you Provide reference to the documents like requirements, design, architecture or maybe even screenshots of the error to help understand the defect
  - Detected By - Name/ID of the tester who raised the defect
  - Status - Status of the defect , more on this later
  - Fixed by - Name/ID of the developer who fixed it
  - Date Closed - Date when the defect is closed
  - Severity which describes the impact of the defect on the application
  - Priority which is related to defect fixing urgency. Priority could be High/Medium/Low based on the impact urgency at which the defect should be fixed respectively

End

- Question
- Many Thanks!

Home Study

- When to not rely on state transition?
- **Wishing you a long age as this lecture!**