**What is Test Plan:**

In software development, a test plan is a document which defines your testing team’s test strategy, goals, and scope, which ultimately work together to ensure that all your software components are tested sufficiently before a release.

**How to create a Test Plan:**

Follow these six steps to create an efficient test plan:

1. Define the release scope.
2. Schedule timelines.
3. Define test objectives.
4. Determine test deliverables.
5. Design the test strategy.
6. Plan test environment and test data.

### **1. Define the release scope**

Before any test activity occurs, it’s important to define the scope of testing for your release. This means defining the features or functions that need to be included in the release, considering any constraints and dependencies that can affect the release, and determining what type of release it is.

### **2. Schedule timelines**

Specify release deadlines to help you decide your testing time and routine. Here are some pointers for determining timelines:

* Consult your project manager to understand the current release timeline.
* Look at past release times and schedules.
* Consider extraneous elements: Does the release need to coincide with outside variables, such as conferences or events? Factor those into your release date prediction.
* Consider the timeframes for development: Your development team might have a set schedule for finishing development work. Make sure you comprehend that timeframe so you can adjust the testing schedule.
* Add some extra wiggle room: It’s common to encounter unexpected delays. Including extra time for unforeseen events can help you stick to your plan.
* Review and update the schedule frequently to ensure the test timetable is attainable.

### **3. Define test objectives**

A test objective is a reason or purpose for designing and executing a test. These objectives ultimately help guide and define the scope of testing activities.

Examples of general test objectives include:

* Identifying and reporting defects
* Testing new features
* A certain level of test coverage

### **4. Determine test deliverables**

Test deliverables are the products of testing that help track testing progress. Deliverables should meet your project’s and client’s needs, be identified early enough to be included in the test plan, and be scheduled accordingly. There are different test deliverables at every phase of the software development lifecycle. Here are important deliverables to focus on before, during, and after testing:

#### **Before testing**

* **Test plan document:** The scope, objectives, and approach of the testing endeavor are all outlined in the test plan.
* **Test suite:**Test cases illustrate how to run a test, including input data, expected output, and pass/fail criteria.
* **Test design and environment specifications:** The test environment outlines the hardware and software configurations used for testing.

#### **During testing**

* **Test log:** The test log records each test case’s results, including issues and resolutions.
* **Defect report:** A defect report lists testing issues by severity, priority, and reproducibility.
* **Test data:** According to the International Software Testing Qualifications Board (ISTQB), test data is data created or selected to satisfy the execution preconditions and input content required to execute one or more test cases.
* **Test summary report:**The test summary report lists the number of tests run, passed, and failed, as well as open defects.

#### **After testing**

* **Test completion report:** Covers the testing scope, product quality, and lessons discovered.
* **User acceptance test (UAT) report:**Points to any issues found and fixed.
* **Release notes:** List information about what the release includes. Examples include any new features for development, advancements, or fixes.

### **5. Design the test strategy**

Test strategy helps determine test cost, test effort, and which features will be in-scope (planned to be tested) versus out-of-scope (not planned to be tested).

#### **Identify testing types.**

It is critical to identify when to perform what type of testing, what should be tested manually vs automated, the scope of automated tests, how much work will be required to create new test cases, and who will be doing that work.

Depending on several factors, there may be various types of testing to include in your test plan.

Examples of factors to consider when choosing the right testing type to perform include:

* Test objectives
* Your project’s feature requirements
* The complexity of your product
* Your team’s experience levels.
* Regulatory requirements
* Time and budget

Here are commonly used types of testing to consider including in your test plan:

|  |  |  |
| --- | --- | --- |
| Manual Testing | Automated Testing | Other |
| •Smoke testing •Exploratory testing  •Usability testing of new features | •Unit testing  •Regression testing for existing features •Integration testing | •Performance testing  •Security testing •Accessibility testing |

### **6. Plan the test environment and test data**

Planning a test environment guarantees precise and robust testing. The test environment includes hardware, software, and network configurations for software testing. Follow these procedures to set up the test environment:

* **Determine your hardware and program requirements:**Select test environment devices and software, including operating systems, browsers, databases, and testing tools.
* **Install the required software:**Once prerequisites are established, install the necessary tools on the test environment. This may require setting up a separate server to host the application and installing a database management system or other tools.
* **Configure the network:**Make sure that firewall protocols, IP addresses, and DNS settings, among other network configurations, are identical between the test and production environments.
* **Create the test data:** Prepare the test material for the application’s testing. Test data can be created manually with data from the production environment, retrieved from an existing production environment and database, or created via automated Data Generation Tools.
* **Access the builds:**Ensure that the builds that the testers will be testing are accessible. One example is setting up a file-sharing or version control system to allow testers access to the most current builds.
* **Verify the test environment:**After setting it up, check that your test environment fulfills the requirements.

One-page test plan template

|  |
| --- |
| **Test Plan** **Title** Prepared by: John Doe |
| 1. **Introduction** •Executive summary (This should be kept brief) |
| 2. **Testing Resources** •Tester’s name and role |
| 3. **Scope of Testing** •In scope: Modules that are to be tested •Out of scope: Modules that are not to be tested |
| 4. **Testing approaches** •Testing approach and methodology •Types of testing to be performed (e.g., functional, performance, security, usability) |
| 5. **Test Schedule** •Timeline for each testing phase |
| 6. **Risks & Issues** •Risks associated with the testing process •Mitigation strategies for identified risks |

If your test plan doesn’t fit onto one page, don’t worry. The intention is to minimize extraneous information and capture the necessary information that your stakeholders and testers need to execute the plan.