Testing plan

Minimum lengths

Maximum Lengths

Alhapnumerics/numerics/special chars -, . space etc

‘E’ in numbers

Disable / enable controls

Controls/links work as supposed to

Tab order?

Invalid numbers 0.00.00

Negative numbers

Define test objectives

[A test objective](https://istqb-glossary.page/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

Examples of objectives for specific types of testing include:

* Functional testing objectives: Ensure the software works as it should. Examples of goals for this objective include: Validating user workflows, data processing, and verifying input/output parameters.
* Performance testing objectives: Ensure the software is efficient and can handle various loads. Examples of goals for this objective include: Verifying software reaction time, throughput, and scalability.
* Security testing objectives: Uncover program security flaws. Examples of goals for this objective include: Verifying authentication and authorization features and identifying potential threats.
* Usability testing objectives: Concentrate on ease of use and user experience. Examples of goals for this objective include: Validating software accessibility, verifying user flow, and identifying user-related issues.

Measure testing with the right metrics

[Metrics assess the overall quality of a release](https://www.testrail.com/), the progress of your testing, and the effectiveness of your testing (for a particular test cycle or the entirety of your testing).

They provide visibility into your testing process and overall product quality, ultimately helping your team decide if your release is ready to ship. Here are some metric formulas you might consider:

**Defect Density**

* [Defect Density](https://www.geeksforgeeks.org/software-testing-defect-density/) = Defect count/size of the release (lines of code)

Example: If your software has 150 defects and 15,000 lines of code, its defect density is 0.01 defects per line of code.

**Test Coverage**

* [Test Coverage](https://www.testrail.com/blog/test-coverage-traceability/) = (Total number of requirements mapped to test cases / Total number of requirements) x 100.

**Defect Detection Efficiency (DDE)**

* DDE = The percentage of defects detected during a phase / Total number of defects

Time to Market

* TTM = The time it takes for your company to go from idea to product launch

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](https://glossary.istqb.org/en_US/term/test-data-1-3)), 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.

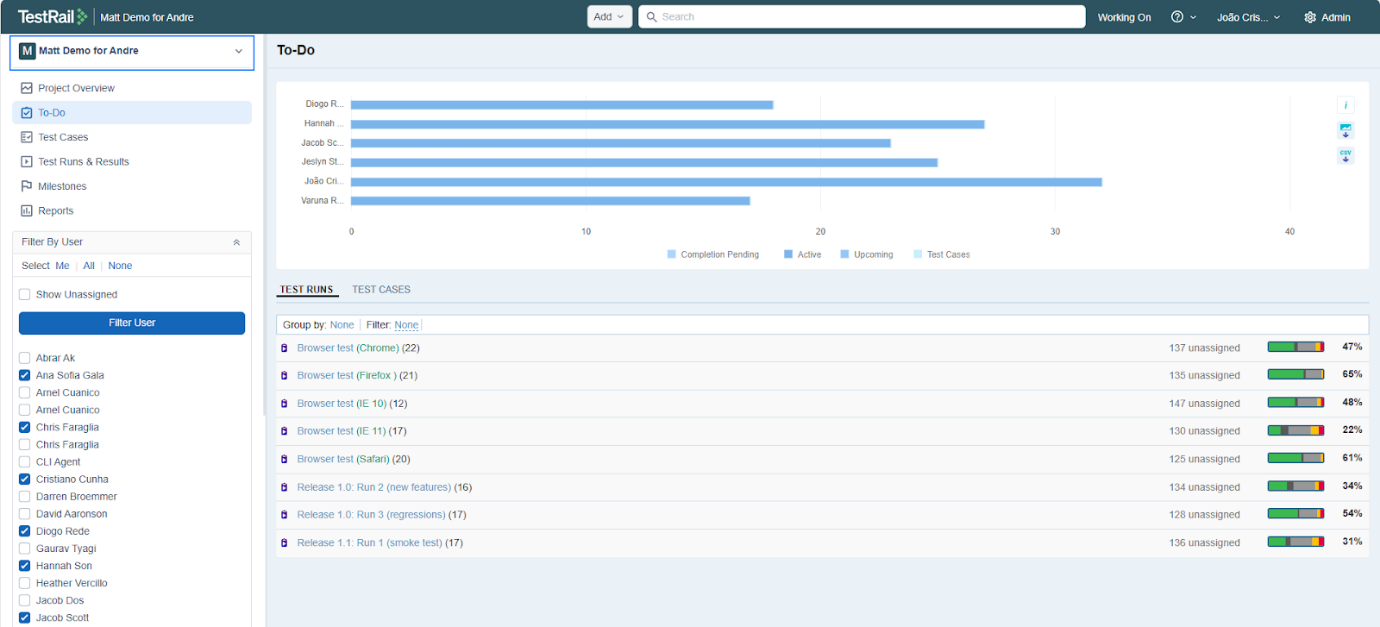
After testing

* **Test completion report:** Covers the testing scope, product quality, and lessons discovered.
* [**User acceptance test**](https://www.testrail.com/blog/user-acceptance-testing/)**(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.

A test plan’s content and structure differ depending on its context. Although there isn’t one cookie-cutter way to write a test plan, following best practices for test plan development can help companies deliver high-quality software.

TestRail is test plan software designed to make it easy to follow best practices for test plan development. In TestRail, you can enter test cases with preconditions, test instructions, expected results, priorities, and effort estimates.

This level of flexibility and visibility into your testing process makes TestRail an easy fit into any organization’s test plan — [Try TestRail for free](https://secure.testrail.com/customers/testrail/trial/?type=signup) to see how it can help with your test planning. Or, learn more about how to build and optimize your testing processes—from test design to [test planning](https://www.testrail.com/blog/test-planning-guide/) and execution with this free TestRail Academy course, [Fundamentals of Testing with TestRail.](https://academy.testrail.com/catalog/info/id:126,cms_featured_course:1)



***Image:****Organize and structure reusable test cases in folders, create agile test plans, and track test execution progress in TestRail.*

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](https://www.testrail.com/blog/automate-testcase/), 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](https://www.geeksforgeeks.org/types-software-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](https://www.testrail.com/blog/perform-exploratory-testing/)  •Usability testing of new features | •[Unit testing](https://www.testrail.com/blog/unit-testing-tdd-bdd/) •[Regression testing](https://www.testrail.com/blog/agile-regression-testing/) for existing features •[Integration testing](https://www.testrail.com/blog/integration-testing/) | •[Performance testing](https://www.testrail.com/blog/performance-testing-types/#:~:text=Performance%20testing%20is%20a%20process,real%2Dworld%20usage%20without%20issues.) •[Security testing](https://www.kiuwan.com/blog/4-best-practices-for-security-testing-in-your-sdlc/) •Accessibility testing |

Document risks and issues

It’s essential to document risks that may occur during testing and the effect of these risks. Risks can include:

* Strict deadlines
* Insufficient or inaccurate budget estimates
* Poor management
* Problems with the code
* Changes in the business environment
* Limited resources for testing
* Unexpected delays during testing

Document test logistics

Test logistics should answer the “Who, what, where, when, and how.” Documenting test logistics ensures that all human and system-related testing resources are available. For example, it may be important that your team identifies who is available to do testing and who will support them if needed during testing. Moreover, when resource planning, it can be helpful to identify alternative resources or build slack into your plan to ensure your project gets completed.

Establish test criteria

Test Criteria is a standard that regulates all activities within a testing project. The two main types of test criteria include suspension and exit criteria.

* Suspension Criteria: Establishes the conditions for suspending all tests.
* [Exit Criteria:](https://www.testrail.com/blog/agile-exit-criteria/) Exit criteria are established items or goals to complete that define the end of a test phase. The exit criteria of a test are the predetermined results that must be achieved to move on to the next testing phase. For example, 92% of all critical test cases must pass before a feature can be deemed suitable for release to your customers.

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:**](https://www.datprof.com/solutions/what-is-test-data/#:~:text=Data%20can%20be%20created%201,from%20an%20existing%20production%20environment.)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.

Key elements of a test plan

1. **Test plan ID and title:** A unique identifier and name for easy reference.
2. **Introduction and objective:** Purpose of the test effort and high-level goals.
3. **Scope of testing:**Clearly defines what’s included and excluded from the test effort.
4. **Test objectives and approach:** Describes specific testing goals and methodologies (manual, automated, risk-based, etc.).
5. **Test schedule and milestones:** Timeline for key phases such as planning, execution, bug triage, and closure.
6. **Test environment setup:** Hardware, software, tools, and configurations required to execute testing.
7. **Resources and responsibilities:**Who is involved, their roles, and who owns what across the test cycle.
8. **Test deliverables:**The testing artifacts to be produced (e.g., test cases, reports, logs).
9. **Entry and exit criteria:** Preconditions to start and end testing phases (e.g., % of test cases passed).
10. Ri**sks and mitigation strategies:**Identifies potential blockers and how to proactively manage them.

One-page test plan template with examples

|  |  |
| --- | --- |
| **Section** | **Details** |
| **Test Plan Title** | [e.g., v2.4 Web Portal Feature Release] |
| **Prepared By** | [Name, Role] |
| **Date** | [MM/DD/YYYY] |

**1. Introduction**

**Purpose/executive summary:** Briefly describe the objective of the test plan.  
**Example:** “To validate the functionality and performance of the new checkout flow before release.”

**2. Scope of Testing**

* **In Scope:** [Modules/features to be tested]
* **Out of Scope:** [Items/features not covered in this test cycle]

**3. Test Objectives**

* List specific goals, e.g., Validate login authentication, Ensure cross-browser compatibility.

**4. Testing Approach**

* **Methodologies:** [e.g., Manual, Automated, Risk-Based, Agile Testing]
* **Types of Testing:** [e.g., Functional, Regression, Usability, Performance]
* **Tools Used:** [e.g., TestRail, Selenium, JMeter]

**5. Test Schedule**

|  |  |  |
| --- | --- | --- |
| **Phase** | **Start Date** | **End Date** |
| Test Planning | [MM/DD] | [MM/DD] |
| Test Case Design | [MM/DD] | [MM/DD] |
| Test Execution | [MM/DD] | [MM/DD] |
| Bug Fix Verification | [MM/DD] | [MM/DD] |
| Test Completion | [MM/DD] | [MM/DD] |

**6. Test Environment**

* **Hardware/Software:** [e.g., Windows 11, Chrome 124, iOS 17]
* **Staging URL or App Version:** [Insert here]
* **Test Data Sources:** [e.g., Mock data, anonymized production data]

**7. Resources & Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Responsibilities** |
| QA Lead |  | Test plan, coordination |
| Test Engineers |  | Test execution, defect reporting |
| Dev Support |  | Bug triage, environment setup support |

**8. Risks & Mitigation**

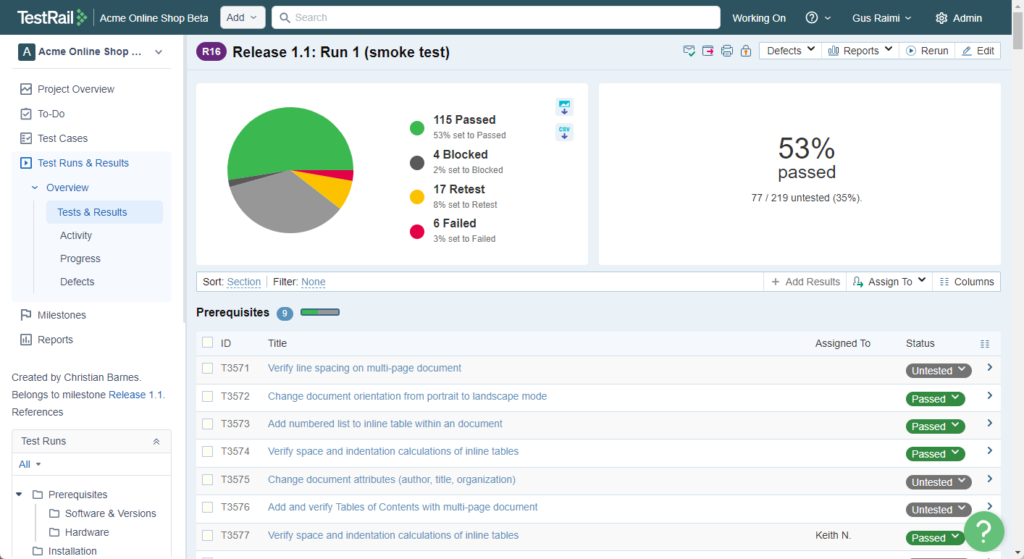
|  |  |
| --- | --- |
| **Risk** | **Mitigation Strategy** |
| Tight release schedule | Prioritize critical test cases |
| Limited device/browser coverage | Use cloud testing platforms |

**9. Test Deliverables**

* List of key artifacts to be created or reviewed throughout the test effort:

**10. Entry & Exit Criteria**

* **Entry:** Code complete, environment stable, test cases reviewed
* **Exit:** 95% test case pass rate, no critical/severe open bugs



Test report in excel???