# How to make Test Case

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**

Creating a **test case** means defining a clear, structured scenario to verify that a piece of software (or feature) works as expected.  
Here’s a complete, practical guide 👇

## 🧩 What Is a Test Case?

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**

A **test case** describes:

* What to test
* How to test it
* What result to expect

It ensures that every feature or function works correctly under specific conditions.

## 🧱 Basic Structure of a Test Case

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**



| **Field** | **Description** | **Example** |
| --- | --- | --- |
| Test Case ID | Unique identifier for tracking | TC001 |
| Test Title / Name | Brief description of what’s being tested | Login with valid credentials |
| Module / Feature | The part of the system under test | Authentication |
| Preconditions | What must exist before running the test | User must be registered |
| Test Data | Data required for the test | username: john\_doe, password: 12345 |
| Test Steps | Step-by-step actions to perform | 1. Go to login page 2. Enter username/password 3. Click login |
| Expected Result | What should happen if the test passes | User is redirected to the dashboard |
| Actual Result | What actually happened during the test | (Filled after execution) |
| Status | Pass / Fail / Blocked / Pending | Pass |
| Remarks | Any notes or bugs found | Error message appears incorrectly |

## 🧭 Procedure Steps for Writing a Complete Test Case

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**

**Step 1: Understand the Requirements**

* Read the **Software Requirements Specification (SRS)** or **User Stories** carefully.
* Identify:
  + What the feature does
  + Inputs and expected outputs
  + Dependencies and constraints
* Ask questions if something is unclear — you can’t test what you don’t understand.

🧠 *Example:*  
Requirement: “User must be able to log in using a valid username and password.”

**Step 2: Define the Scope of Testing**

* Identify **which part** of the system you’ll test.
* Determine **what type** of test case you’re writing:
  + Functional
  + Negative
  + Boundary / Edge
  + UI / Usability
  + Performance (optional for QA teams)

🧠 *Example:*  
Scope: Functional testing of login feature (valid and invalid login).

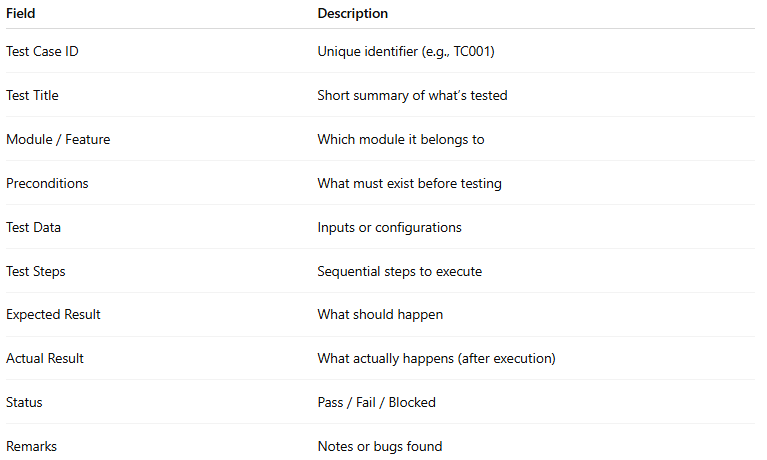
**Step 3: Identify Test Scenarios**

* A **test scenario** is a broad area to test.
* It represents *“What to test”*, while test cases describe *“How to test it.”*

🧠 *Example:*  
Test Scenario: Verify the login functionality with valid and invalid credentials.

**Step 4: Write Detailed Test Cases**

For each test scenario, write multiple **test cases** using a consistent format.  
Include these standard fields:



| **Field** | **Description** |
| --- | --- |
| Test Case ID | Unique identifier (e.g., TC001) |
| Test Title | Short summary of what’s tested |
| Module / Feature | Which module it belongs to |
| Preconditions | What must exist before testing |
| Test Data | Inputs or configurations |
| Test Steps | Sequential steps to execute |
| Expected Result | What should happen |
| Actual Result | What actually happens (after execution) |
| Status | Pass / Fail / Blocked |
| Remarks | Notes or bugs found |

🧠 *Example:*  
**Test Case ID:** TC001  
**Title:** Verify login with valid credentials  
**Steps:**

1. Open the login page
2. Enter valid username and password
3. Click “Login”  
   **Expected Result:** User should successfully log in and see the dashboard.

**Step 5: Review and Validate Test Cases**

* Review your test cases with the QA lead or peers.
* Ensure:
  + Coverage of all requirements
  + Clear and unambiguous language
  + Realistic and reusable test data
* Update test cases based on feedback.

**Step 6: Execute Test Cases**

* Run each test case in the actual application.
* Record the **Actual Result** and **Status (Pass/Fail)**.
* Log any defects found in the bug tracking system (e.g., Jira, Bugzilla).

**Step 7: Maintain and Update**

* Update test cases whenever the system changes.
* Retire obsolete cases, and add new ones for new features.
* Keep your test suite version-controlled and organized.

**🔁 Summary Flow**

1. 📘 Analyze requirements
2. 🎯 Identify test scenarios
3. 🧩 Write test cases
4. 🧾 Review & validate
5. 🧪 Execute test cases
6. 🐞 Log defects
7. 🔄 Maintain & update

## ✅ Example Test Case

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**

**Test Case ID:** TC002  
**Title:** Verify login fails with incorrect password  
**Module:** Login Page  
**Preconditions:** User john\_doe exists  
**Test Data:** username=john\_doe, password=wrongpass  
**Steps:**

1. Open the login page
2. Enter valid username
3. Enter invalid password
4. Click on “Login” button

**Expected Result:**  
System should show “Invalid username or password” message and stay on login page.

**Actual Result:**  
*Observed during test run.*

**Status:** Pending

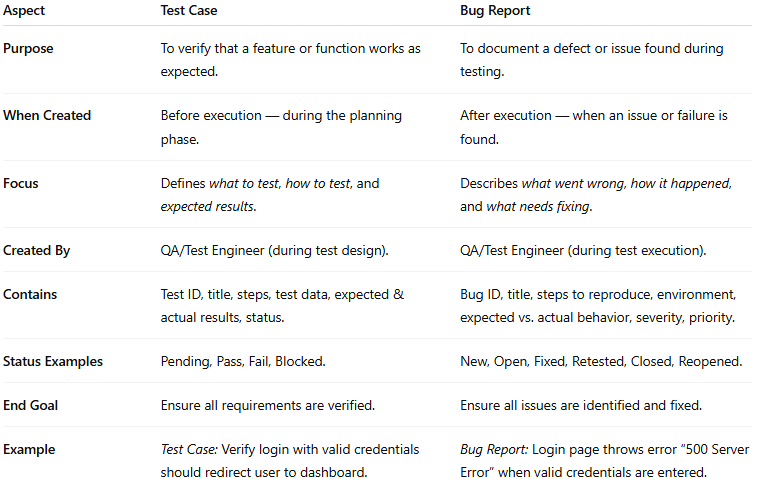
## 🧠 Tips for Writing Good Test Cases

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**

1. ✅ Be clear and concise — anyone should understand it.
2. 🔁 Keep one objective per test case.
3. 🧪 Use meaningful test data.
4. 📊 Include both **positive** and **negative** scenarios.
5. 🔍 Review and update as features change.

## 🧩 Difference Between Test Case and Bug Report

**# Source**: **Chatgpt (GPT-5-Standard) at [10/6/2025]**



| **Aspect** | **Test Case** | **Bug Report** |
| --- | --- | --- |
| **Purpose** | To verify that a feature or function works as expected. | To document a defect or issue found during testing. |
| **When Created** | Before execution — during the planning phase. | After execution — when an issue or failure is found. |
| **Focus** | Defines *what to test*, *how to test*, and *expected results*. | Describes *what went wrong*, *how it happened*, and *what needs fixing*. |
| **Created By** | QA/Test Engineer (during test design). | QA/Test Engineer (during test execution). |
| **Contains** | Test ID, title, steps, test data, expected & actual results, status. | Bug ID, title, steps to reproduce, environment, expected vs. actual behavior, severity, priority. |
| **Status Examples** | Pending, Pass, Fail, Blocked. | New, Open, Fixed, Retested, Closed, Reopened. |
| **End Goal** | Ensure all requirements are verified. | Ensure all issues are identified and fixed. |
| **Example** | *Test Case:* Verify login with valid credentials should redirect user to dashboard. | *Bug Report:* Login page throws error “500 Server Error” when valid credentials are entered. |

**🧠 Easy Way to Remember**

* A **Test Case** is like a **plan** → “What to check.”
* A **Bug Report** is like a **result** → “What went wrong.”