TESTMONITOR

The Complete Guide to

Next-Level User Acceptance Testing

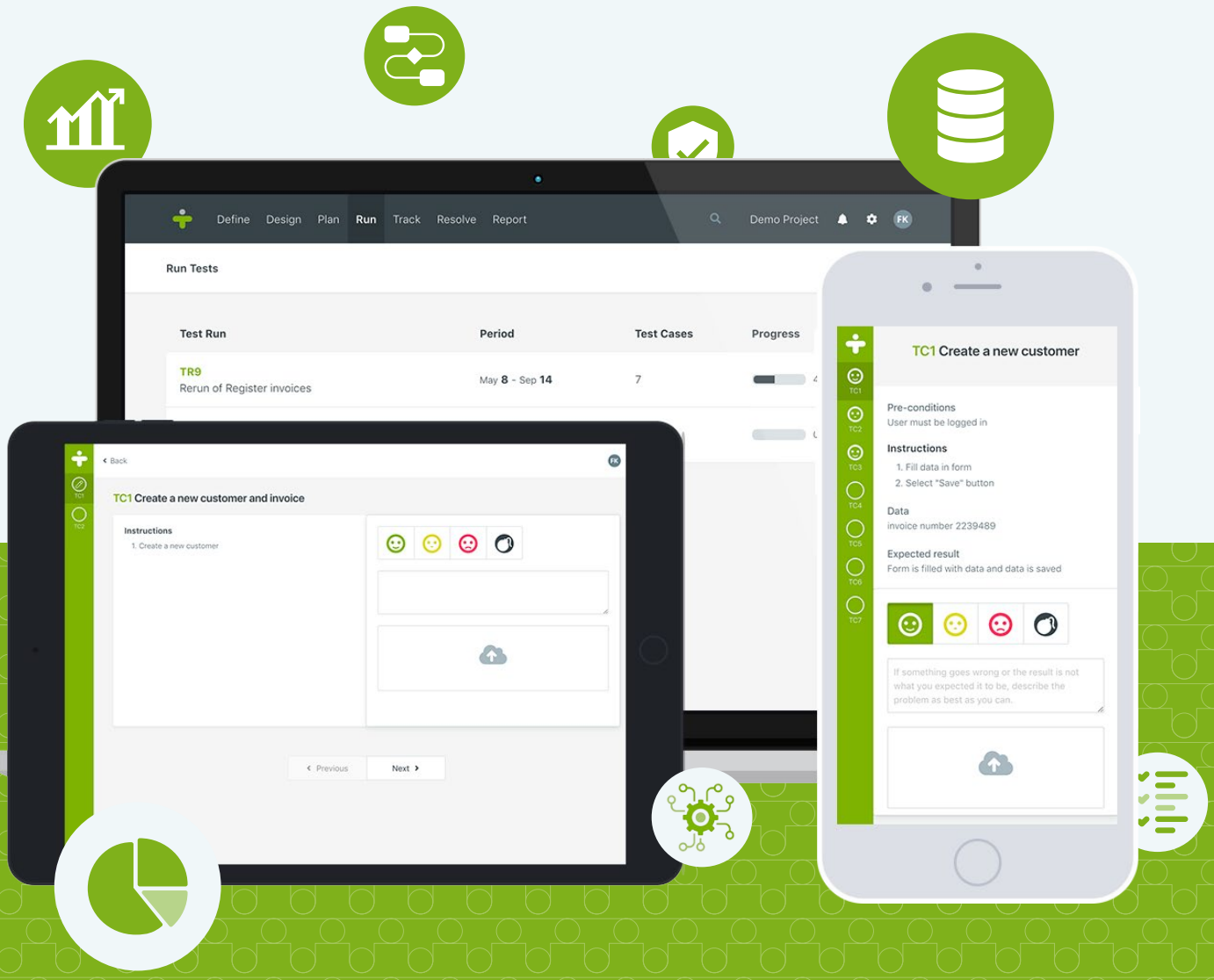


Table of Contents

CHAPTER 1
UAT: The Journey Begins 3

CHAPTER 2
Plan to Succeed 6

CHAPTER 3
Choose the Best UAT Tools 11

CHAPTER 4
Designing a Winning UAT 18

CHAPTER 5
Time to Launch: UAT Test Runs 22

CHAPTER 6
UAT Results: The Right Stuff 24

CHAPTER 7
UAT Sign-Off Sheet 26

CHAPTER 8
Using UAT Templates 27

CHAPTER 9
The Home Stretch 28

Chapter 1

UAT: The Journey Begins

When it comes to a major software project—including a new system, a retooling of a failing system, or the next big thing in app development—every step counts. Whether it's conception, engineering, design, quality assurance, testing, or release, no single process is more important than the others. However, software testing is one process for which your team can't afford an inch of slack. Without proper testing, every step beforehand is useless. How will you know if the product meets the objectives and targets from engineering and design if you can't verify its overall usability and functionality.

OK, we all get it: Software testing is the key ingredient for a well-prepared project. But what kind of testing provides the optimal outcome? Automated testing can have its uses, but when it comes to accuracy, user-friendliness, and overall functionality, manual testing can't be beaten by a robot tester that lacks human intuition.
In this guide, we dive into the world of user acceptance testing (UAT). We cover why this unique process fuels a world-class software build and ensures end user satisfaction in a way that can't be replicated by other testing types. UAT software and testing tools put the user and his or her unique perspective into the driver's seat.

Without proper testing, every step beforehand is useless

Let's begin the journey to creating a UAT environment by asking the key questions:
What and Why?

What is UAT?

The concept of user acceptance testing (UAT) is simple. In a UAT environment, living, breathing (human!) end users test the project in question, whether an accounting system website, app, or other software product. The process deploys several users and assigns rudimentary tasks to make sure all key software functions work properly and flow smoothly. UAT testing weeds out defects and bugs while empowering the user to make suggestions and tweaks to the software.

By scaling UAT testing across a diversity of users, a project manager can determine if the product is ready for prime time (i.e., ready for launch and production). With UAT, users complete the process on an individual basis—a procedure known as black box testing—to provide more accurate and less biased analysis. UAT is not just one single thing. It is a set of tools that encompass several steps and benchmarks.

A word of caution: Although the processes work in tandem, UAT testing is not QA. In general, QA focuses on discovering bugs and more technical software mistakes. The UAT process takes a human-centric approach. The user enters the UAT environment wanting answers to questions like: *What will this software do for me? How will it help me be more productive? Will it be too difficult to learn?*

It may sound a bit philosophical but: UAT testing is ultimately about bringing happiness to the user. And isn't that the ultimate goal of software development? Happy, productive and empowered clients?

Why UAT?

While we're on the topic of happiness, let's figure out why UAT is so vital. Put simply, UAT testing saves time, money, and frustration by ensuring the end user will be satisfied. Quality assurance teams will find UAT perfect for their needs because it helps QA managers and development teams understand the product's purpose from the user's point of view.

UAT testing can (and should) be performed by your team rather than an outside provider. Doing so enables your team to create, monitor, and change ever-evolving requirements, test-case development, and data collection/analysis. UAT gives control to users, as well as to your team.

Designers benefit from UAT testing, leveraging the power of user experience to inform design. Market research and user interviews are useful and often create the foundation for design, but, at the end of the day, designers create the product based on assumptions instead of more actionable data. UAT provides designers with actionable data.

We now know the what and why of UAT. With that in mind, it's time to plan and succeed!

Designers benefit from UAT testing – leveraging the power of user experience informs design.

Chapter 2

Plan to Succeed

Ask any sailor: A ship without a rudder will never reach port. Ask an informed test manager: Any UAT project without a plan is destined to sink.

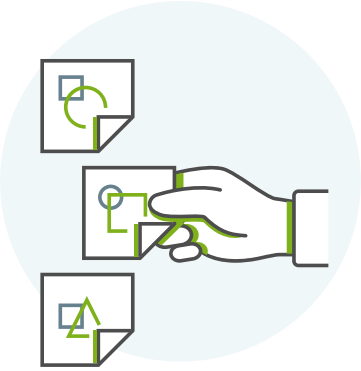
A test plan directs your testing approach, details the testing practices, and defines the responsibilities of team members. It's a step-by-step journey.

Step 1: Determine the Need and Objective

The objective and expected outcome of any UAT project will, of course, be guided by your organization's needs. This seems obvious but many project teams stray from the main course as the project grows and becomes more complex. Keeping an eye on the specific needs of the project will maintain your team's direction toward true north.
Of course, your UAT objective will depend on what type of testing is needed.

UAT needs will depend on the test type. Examples may include testing a website, testing a mobile app, or testing an enterprise resource planning system (ERP). Each test will encompass a diversity of requirements, risks, test cases, and dependencies.

Perhaps you need to execute acceptance testing to validate a system is both operational and acceptable to all team members involved in terms of usability. A modern example would be a test to ensure a company's sudden need to increase remote workers can be properly migrated and deployed using existing software. As you plan, ask yourself and team: Can we clearly describe what a successful test looks like? What does success look like after this UAT?



Step 2: Define the Approach

In order to understand what success looks like for your UAT, you must have an accurate map—The "How?" of the project. What path will best achieve the desired goal?

This process will include defining the team—not only the makeup but the roles involved—as well as choosing the right tools (more on that in Chapter 3).

A winning UAT project will involve creating test cases, defining requirements and risks, planning test runs, analyzing test results, managing related issues, and designing tests. Don't forget, you will also need to make sure the entire team is on board with your testing plan and toolkit.

A winning UAT project will involve creating test cases, defining requirements and risks, planning test runs, analyzing test results, managing related issues, and designing tests.

Step 3: Map the Timeline

Your UAT toolkit should give you a bird's eye view of all test runs across all timelines and should account for regression testing in addition to other legacy test cases. If Step 2 is the map, then think of Step 3 as the schedule or itinerary. It should go without saying but **communicate** both the approach and timeline clearly. Let's repeat that: **communicate!**

Step 4: To Test or Not to Test...

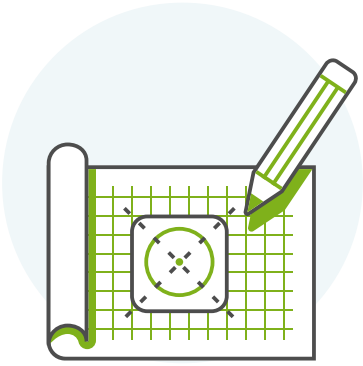
It's sometimes the case that a UAT plan looks amazing on paper—all the objectives are described; every team member role is detailed, etc. The problem? The UAT may be too far-reaching. It may attempt to bite off more than it's designed to chew. In short, you must design what you're going to test and also define what you'll NOT be testing. It's unfortunately easy to lose sight of the UAT forest for the trees—especially "trees" that don't belong in the first place. For example, perhaps your UAT seeks to test the functionality of an accounting software system. Such a UAT may need to test tasks such as paycheck creation and payroll tax calculation. That means, there's no reason to bog the process down into adjacent categories such as corporate cost analysis or business processes.

Step 5: Ask the Right Questions

No planning process would be complete without a time of questioning. Not all questions are equally important, but they should still be asked along the way.

Often, asking the right questions will help map out the best procedure. Best procedural practices flow from answering the key aspects of the overall UAT design. Sometimes, the answer may be “we don’t know,” but even unanswered questions will reveal the strategic next action for your project. Encourage your team to ask, ask, and then ask again. Questions may include:

- Do we know how much time to devote to creating the best test design?
- Does the plan include outcome-based test cases with an actionable end goal?
- Are users included in the approval of test cases? How about planning test runs?
- How will we hold meetings to give users an idea of how testing will work?
- What’s the plan for training users on how to actually use the product?
- What’s the strategy?
- What’s the testing schedule?
- What are both entry and acceptance criteria?



Step 6: Know the Risks

Unless your UAT testing plan includes well-defined risk metrics, it's bound to fail before launch. Defining risk metrics gives you a solid understanding of the project's eventual outcome. Risk metric tactics must identify test results with the highest risk factors. These could include things like operational or financial risks. Your risk metrics must also be directly linked to test cases. They should empower the project such that defined risks can be informed via dynamic filtering capability. This enhances the analysis of test cases, test runs, test results, and other issues based on your defined risks.

Risk metric tactics must identify test results with the highest risk factors. These could include things like operational or financial risks. Your risk metrics must also be directly linked to test cases.

Chapter 3

Choose the Best UAT Tools

Finally, as already noted, your plan must define the most comprehensive, intuitive UAT tools available. Let's dive deeper into what those tools might be and how you can leverage them for UAT testing success!

UAT is not just one single thing. It is a set of tools that encompass several steps and benchmarks.

Risk/Requirement Management

As you continue your journey to the Land of the Perfect UAT, you must have a way of knowing if your direction is right and if there are any potholes, traffic, or dangers down the road (as noted in Chapter 2). You need a compass or GPS, i.e., risk/requirement management. UAT tools help define requirements and risks vital to your UAT testing journey, allowing you to navigate the testing project across the bumpiest of roadways.

UAT tools like TestMonitor empower you to easily deal with large amounts of requirements and risks by organizing them into groups. Users classify requirements by using different requirement types and can easily prioritize risks with provided classifications.

In addition, a champion UAT tool makes it easy to assign one or multiple requirements or risks to test cases. The result? Relationships can be automatically adjusted and connect to test runs, test results, and issues.

As you travel down this roadway, you want the ability to filter and analyze test cases, test runs, test results, and issues based on these defined requirements and risks. That ability allows you to focus on the test results that represent the highest project risk. In addition, you then have a superior view for the risks that have the greatest impact on vital project requirements.

Test Case Management

A key tool in your UAT kit is Test Case Management (more on the details of test cases will be featured in the next chapter).

The best UAT tools relate test cases to reusable objects while test registration tools can organize relationships of tests in a more intuitive way.

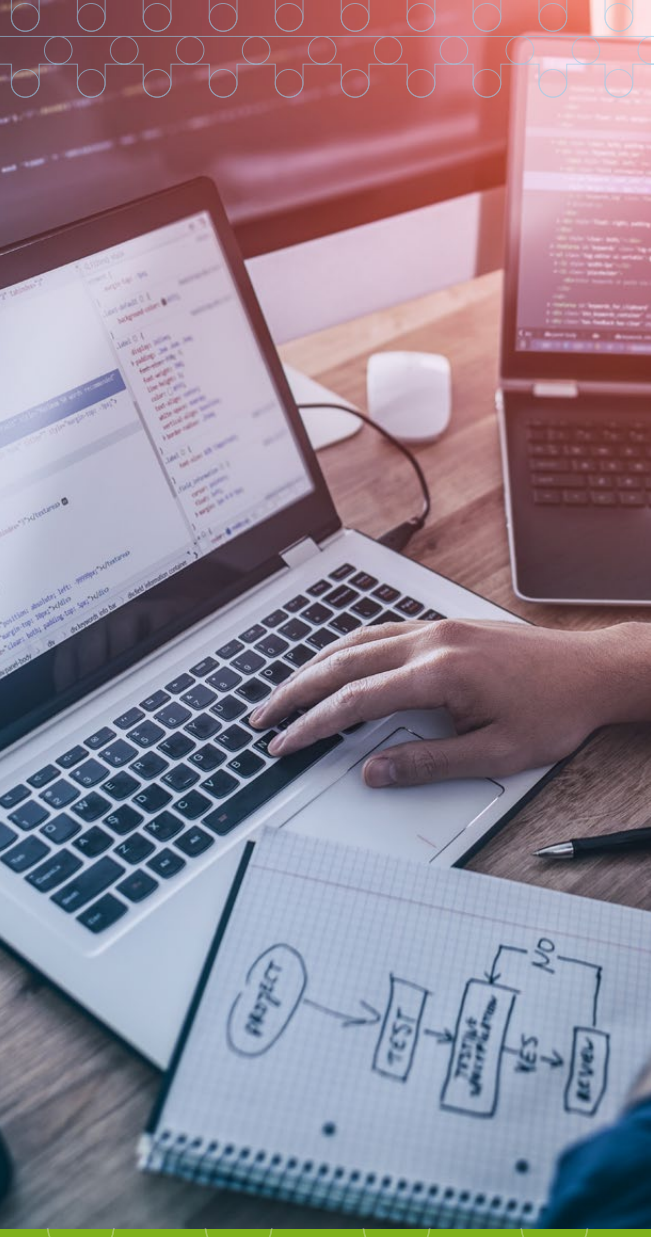
UAT tools like TestMonitor empower you to easily deal with large amounts of requirements and risks by organizing them into groups.

As you plan your test case, ask: “What is the goal? What data do we want to find out as a result? What are the expected outcomes?”

Some common goals for test cases include:

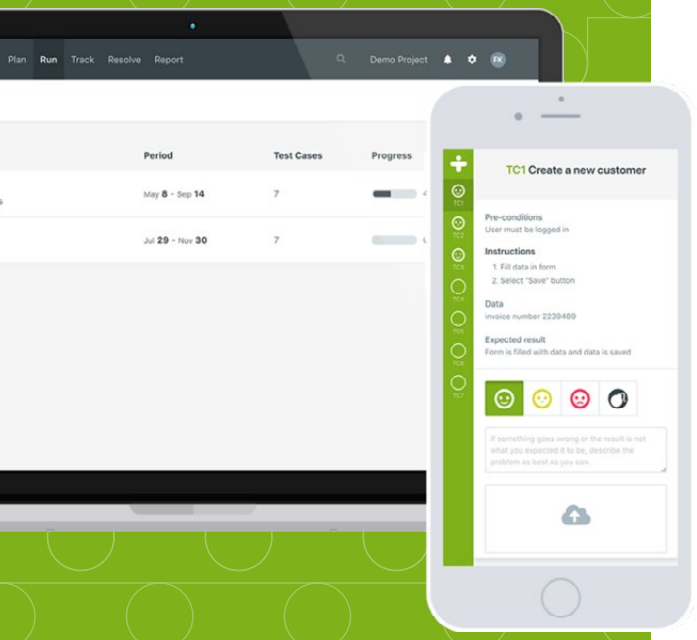
- **Identifying defects**—often seen as the primary reason to test case.
- **Conformance assessment**—for example, are the expected specifications operating within acceptable parameters?
- **Discovering** the greatest number of bugs early to avoid deeper issues down the road.
- **Mitigation of risks** for support managers (especially for “go/no-go” decisions).

Your UAT tool should provide a clear description of the test case purpose, simplifying the activity’s expected outcome. In addition, our tool allows users to define labels or tags that can be linked to test cases based on criteria such as business process, risk, requirement, or application.



Test Runs

Test runs empower a user to leverage correct cases from a test vault while also avoiding unnecessary tests. Your UAT tool kit optimizes milestones to mark important project events. Test run management tools give you a bird's eye view of all test runs across all timelines and account for regression testing in addition to other legacy test cases. Test runs must be scalable across any relevant devices, Windows, Mac, iOS and Android. Finally, test runs should be able to be customized so that a user can duplicate any runs with a single click.



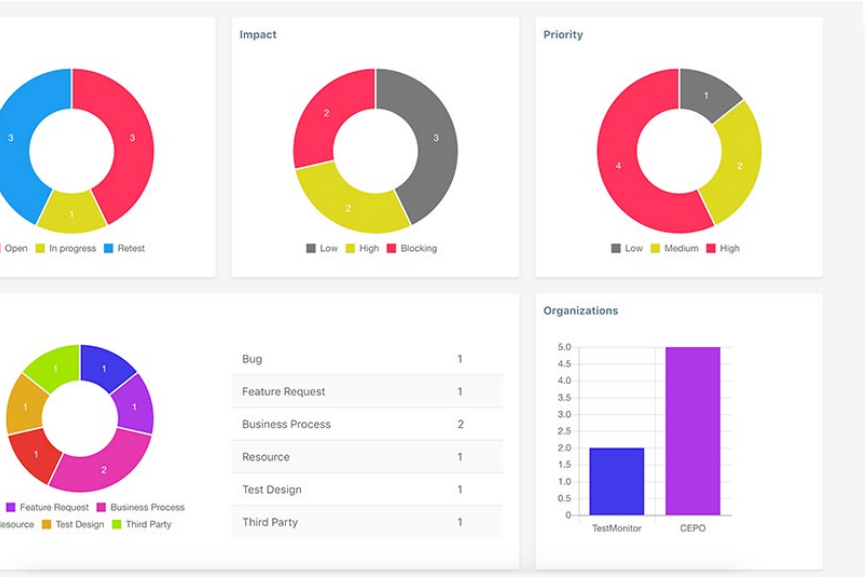
Test run management tools give you a bird's eye view of all test runs across all timelines and account for regression testing in addition to other legacy test cases.

Results

As with any long journey, it's important to know if the trip proved successful—did you arrive down the correct route in the proper time frame? UAT tools provide a detailed overview of test results covering every test run. Test managers can focus on specific details within each test case, as well as monitor results over time for improvement, stability, or decline.

With the proper results-based UAT tools, you will be able to view latest outcomes per test case and test run. As noted above, TestMonitor gives you powerful filters to view results per milestone, requirement, or any other metric.

An unstable, declining result can morph into an issue and must be addressed. UAT tools convert problematic results into issues (or link them to existing issues). This puts you in the driver's seat when it comes to fixing the issues and planning new tests for verification. We'll discuss this issue more in Chapter 6.



TestMonitor gives you powerful filters to view results per milestone, requirement, or any other metric.

Issues

As already noted, declining results can quickly grow into full-blown issues. However, UAT tools, such as TestMonitor, have you covered. Such tools include a simple, yet powerful, integrated issue tracker with filters, prioritization, a full audit trail, attachment handling, commenting, and task management. In short, everything you need to deal with issues.

A super-powered issues management tool resolves issues by breaking them down into manageable tasks for different users. In addition, the team is notified when tasks are completed or assigned. An added bonus is the inclusion of test-result attachments related to issues. With TestMonitor, issues can also be uploaded as attachments using drag-and-drop.

No issues management solution would be complete without a commenting function that notifies team members when a user comment populates.

Reports

UAT management tools must deliver real-time insight into testing status and progress. That includes tracking the team-wide workloads with instant status and progress reports for test runs, test cases, and arising issues.

When reviewing a potential test management tool, ask the questions: Can we view traceability, progress, and coverage reports? Can we view issue reports per status, impact, category, priority, or organization?

World-class reporting options provide key insights across the project: strengths, weaknesses, and growth areas. Smart reporting provides real-time insight into testing status and progress. It also allows management to track the workload of the entire team with real-time status and progress reports for test runs, test cases, and issues.
UAT tools like TestMonitor use integrated reports that provide output for the whole package—requirements, risks, test runs, test results, and issues. Reports also include the ability to view traceability, progress, and coverage reports. For more on reporting, see Chapter 6.

Chapter 4

Designing a Winning UAT

Congratulations! You've created a dynamic, actionable plan for a game-winning UAT process. Now, the project is ready to launch into the design phase. Any attempts to take shortcuts in properly designing these modules will result in an epic, crash-and-burn catastrophe for your UAT project. Let's take a deeper dive.

Your team's UAT design will optimize every aspect of manual testing—defining requirements and risks, planning test runs, analyzing test results, managing related issues and, of course, test design.

What defines a rock-star level design?

Best design practices empower the team to easily reproduce the test and optimize future testing.

Best design practices provide testers with the ability to run tests after the design phase in real-world conditions

Best design practices leverage a UAT management tool with a clear, user-friendly organizational structure while also offering an intuitive interface.

Best design practices are not built upon outdated methods such as Excel spreadsheets. This can clog the process with a host of problems.

Best design practices yield objective results about the usability and functionality of the product.

Best design practices fuel a streamlined testing process, leveraging both intuitive interfaces and a user-friendly process. Whether they're an end user or a testing professional, your team members know immediately how to record results and understand whether or not to support their tasks with attachments and comments.

In addition, best UAT design practices generate an actionable task list of all planned test runs assigned.

Best design practices fuel a streamlined testing process, leveraging both intuitive interfaces and a user-friendly process.

Understanding Test Cases

A test case is a collection of test instructions. The final outcome of a test case is either pass or fail.

For example, a test case may confirm that an accounting database search properly delivers the correct form or data. After a test case is created, it is organized into test runs. This involves assigning one or more testers a number of test cases within a certain time period.

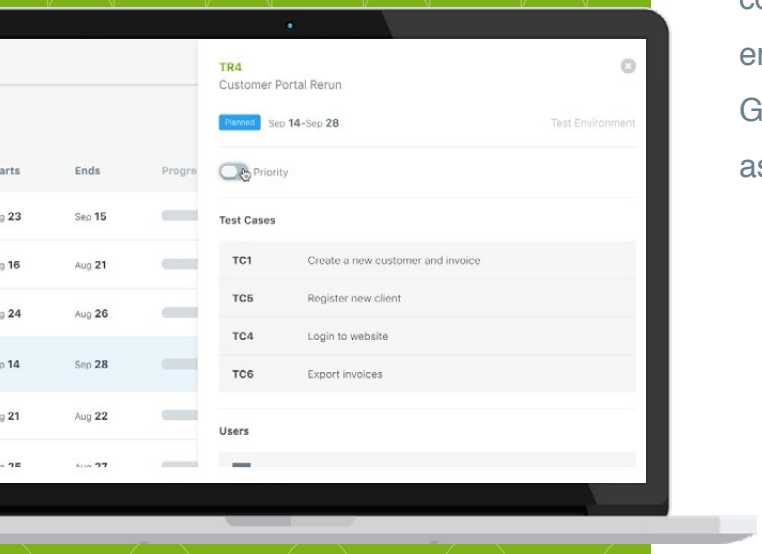
Test cases define input data values, as well as lay out the strategic next actions. This keeps laser focus on expected results. Test cases determine if expectations are on target and help identify defects or errors.

As noted in the previous chapter, test cases must be envisioned in the planning process. You need to define and understand expectations for the test case before launching. Understand what precisely must be tested and know why that's important.
In UAT testing, case names matter. They should connect cases to reusable objects without creating confusing or unclear names. Next, organize preconditions, attachments, and test data within the input stage.

Check and then double-check that test steps and instructions are concise and descriptive. Test managers must also recognize that testers come from different backgrounds. Each has a unique perspective and may not understand the jargon of the UAT world. Education is key.

In case you haven't noticed, organization is the key word when dealing with test cases. Provide a central repository to create additional test cases. Many test case teams make the mistake of using Excel or even Word to consolidate test results. That's a road to failure. Excel is not up to the task of managing test cases for a variety of reasons. Avoid using it!

Test cases must also leverage production data to create an environment that reflects the real world—a reproduction of what a real user might experience. Users should be empowered to add comments, attachments (e.g., screenshots), and employ other methods of actionable feedback. Give users as many conduits for feedback as possible.



Chapter 5

Time to Launch: UAT Test Runs

If a test case is the vehicle that drives a UAT project, then a test run would be the roadway. Test runs determine how test cases are tested, as well as when and by whom. Test runs represent the opportunity to ask questions such as: Who will test? Which cases will be driving down this specific road?

To continue the automotive metaphor, a test run not only paves the road for the test case, but it also provides road signs and directs the case. It means your testers are ready to step into the vehicle and travel a smooth road to the final destination: UAT success. Grouping test runs into milestones allows the UAT team to acknowledge vital events in the life of the project. Your team will then be equipped to schedule sprints, releases, or iterations with organized test runs.

As your team considers both test-case creation and test-run operation, keep the following tips in mind:

- Make sure the UAT provides a bird's-eye view of all test runs, past and present. A UAT tool such as TestMonitor will allow you to view progress of each test run and keep track of all active and upcoming test runs, along with the assigned testers.

- If your UAT tool includes a notifications function, you'll be able to provide constant communication to all project members, informing them of upcoming test activities and planned test runs. TestMonitor offers this feature with one click of a link.

UAT tools must also allow the effortless duplication or rerunning of test runs. TestMonitor boasts single-click functionality to duplicate test runs and keep both testers and test cases on the right track. Test runs should be scalable across any relevant device operating system, including Windows, Mac, iOS, and Android.

Editing a huge quantity of test runs should be a breeze. A world-class UAT tool will allow you to select multiple test runs and easily change scheduling by week or month, or schedule runs to expire on a specific date.

Chapter 6

UAT Results: The Right Stuff

Now that you've taken the time to execute amazing UAT planning and design strategies and complete the process with razor-sharp test runs, it's time to "take the cake out of the oven (in baking parlance) and see how it tastes! In other words, it's time to examine the results.

Next-level UAT tools empower your team to track results with a superlative reporting function that captures "the good, the bad and the ugly (or buggy)." The platform should allow managers to track the workload of your entire team's progress reports for test runs, test cases and issues.

With the right tool, users discover it's easy to report findings with an intuitive test registration function. That, in turn, allows them to report results within a few minutes. A warning: unless the UAT tool includes powerful filter options for reports, your project is in danger of collapsing under the weight of poor communication. For example, your team should be able to filter based on defined requirements and risks or planned milestones.

With the right tool, users discover it's easy to report findings with an intuitive test registration function.

Your design should deploy integrated reports that allow your team to easily view traceability, progress and coverage reports, as well as issue reports per status, impact, category, priority or organization.

Bugs are inevitable and the best UAT tools transform your management process into bug tracking superheroes. Your team then unearths those pesky bugs before the end user. Your UAT tool, then, should relentlessly track problems and inform your team of status impact at a glance.

Solutions that offer an integrated bug tracking solution will equip managers to handle all problems immediately. In addition, such a solution links with external players: Zapier, Jira, Topdesk, DoneDone, Asana, DevOps or Mantis.

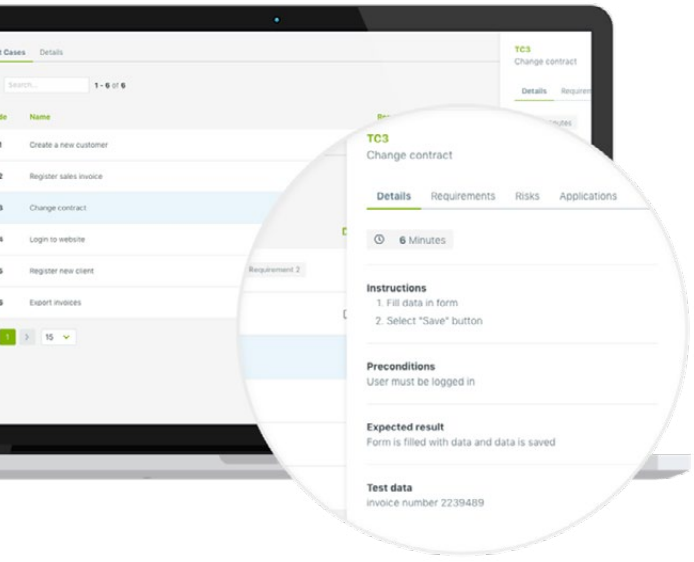
Chapter 7

UAT Sign Off Sheet

Once your team has finalized plans, cases, and design, and deployed all aspects of the UAT toolkit, it's time to deploy a sign-off sheet. This literally keeps all team members on the same page and ensures that each step has been executed, checked, and approved by managers.

A complete sign-off sheet will let the team know the project's status is "fit for purpose for the business area." It also means all defects are resolved and the test manager recommends acceptance of the product so that it can be implemented by the project manager.

There is no secret recipe for the perfect UAT sign-off sheet—you have a variety of layouts to choose from. However, a quality UAT tool will help populate a data-driven, visual sheet that provides a detailed yet strategic view of the process to the entire team.



A proper sign-off sheet should include:

- A way to accept the new product as it is
- Documentation of responsibility for the changes introduced with the new product
- Documentation of responsibility showing all test cases that have been passed or failed by users
- Acknowledgment that any functions may not work as expected, but that workarounds are sufficient
- Acknowledgment of any defects not finished before product deployment

Chapter 8

Using UAT Templates

As important as a superstar UAT tool may be, creating and managing an actionable UAT testing template is also vital. Default UAT testing templates should reflect the real-world environment that is dependent on the product's context. Once test cases have been created, your team will need an understandable way to present the project to the testers so that the results are readily registered.

Templates allow your team to create and manage reusable project blueprints. Templates may be deployed as many times as needed.

A quality UAT tool includes the ability to build an extensive library of project templates that populate requirements, risks, test suites, and test cases. Test templates provide blueprints for the test strategies, objectives, schedules, estimations, and deliverables, as well as needed UAT resources.

A template could include a single requirement working with an entire, ready-to-use project setup (test cases and test runs included). UAT solutions, such as TestMonitor, provide easy-to-use templates that allow you to focus on tester assignment and successful test runs with little to no fuss.

A well-designed template provides a test manager with a monitored and controlled method that can be scaled for collaboration with QA managers and test users.

Chapter 9

The Home Stretch

Test management tools, such as TestMonitor, will provide next-level competency across the entire UAT project. Leveraging your toolkit in the best manner possible requires careful consideration of the overall UAT plan, as well as open communication across the team.

What are the benchmarks of a top-quality UAT tool? Tools that optimize the entire UAT process will deploy test cases, requirement/risk management, test runs, results analysis, and issues management. The best UAT tools offer several test runs and milestone cloning, as well as comprehensive result tracking, smart reporting, filter and visualization options, intuitive user integration, and third-party integration for formats such as Jira, DevOps, and Slack.

A final or supplementary consideration in executing UAT testing for your new project, app, or renovation includes understanding Agile and DevOps.

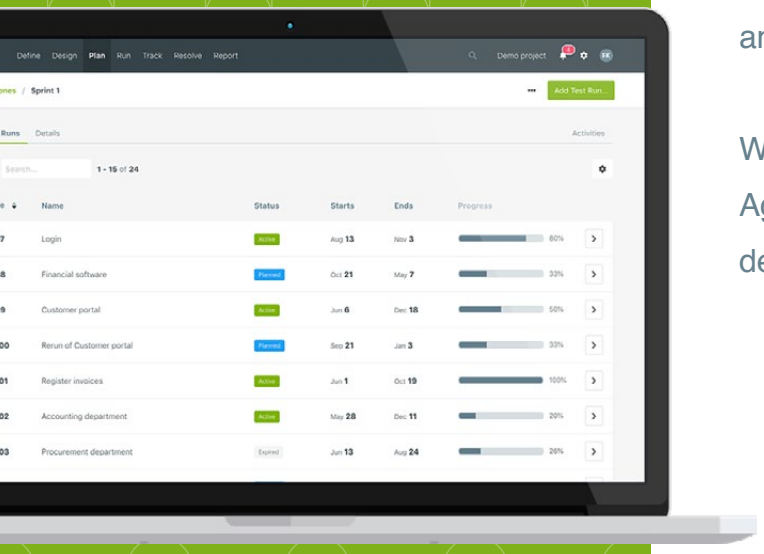
Tools that optimize the entire UAT process will deploy test cases, requirement/risk management, test runs, results analysis, and issues management.

DevOps and Agile

The best UAT tools deploy a testing approach that incorporates every key step of the process, from defining, designing, planning, and executing to tracking test results, resolving issues, and reporting. TestMonitor equips test and QA managers with tools that allow them to intentionally coach teams, while focusing on strategic goals within an Agile environment. As one client noted, our platform is an “easy-to-use test tool enhancing collaboration between end users and IT technicians.”

TestMonitor can easily connect your project to any DevOps project. At its best, keeps processes operational—fixing issues, collecting product wishes and planning regression tests. A UAT tool with superior integrated issue management, like TestMonitor, integrates perfectly into the goals of DevOps—issue creation, new test case design and assurance of “green” regression tests.

When it comes to Azure DevOps and Jira Agile projects, TestMonitor’s milestones can be deployed to plan tests at the end of every sprint.



End of the Journey!

And that's the UAT journey! It's really simple: Realize you and your team can do it. Plan your work and work your plan. Choose industry-leading UAT tools. Deploy proper templates and sign-off sheets. Sound easy? Of course, it's not quite that simple, but with a partner like TestMonitor, your UAT will be poised for amazing success.

TestMonitor is recognized as an industry leader in manual software testing, offering clients a peak experience in designing, planning, creating, and monitoring every vital aspect of the testing process.

TestMonitor's UAT toolkit embraces next-level design principles that result in a consistent, streamlined testing process. That, in turn, produces actionable analysis, results management, and reporting, while intuitively tracking bugs and problems.

Resources

- Checklists
- Knowledge Base
- Webinars
- Video tutorials
- Free trial
- Integrations
- Software Comparisons

Are you ready to make the journey? TestMonitor offers a variety of resources for those considering the fulfilling and useful journey to UAT Testing Perfection.

Try for Free

Try for Free