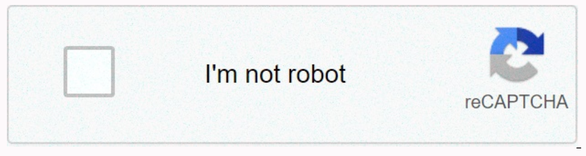

Continue

End user software testing template

Testing user acceptance is an important step – but often overlooked – in every software development project. The UAT principle is simple: It allows you to check if a solution/software/application works for the user. However, its implementation in real life software development teams and processes is something a lot of companies struggle with. In this article, we guide you through a practical example of user acceptance testing, illustrated by testing a Trello feature. When and start with user acceptance testing? UAT tests are usually designed to verify that the customer's needs are met with the solution developed. This is usually done by checking the contract between the customer and the supplier. Therefore, user acceptance tests are necessarily performed at the end of the software development cycle. This means that the developed software (for example, the web application or mobile app) must be largely complete features. While the project team defines a UAT plan right at the beginning of a project (usually during the requirement definition), it executes UAT test cases after development. (source: sfsu.edu) An example of user acceptance test with Trello Let's say I was hired by Atlassian to do a UAT test in Trello (yay!). Their development team approached us to check if the archiving feature developed for Trello cards actually works for the user. Testing the functionality of such a simple feature sounds easy, right? However, we want to follow a proper UAT test setup here. So, we're doing this? UAT Test Case Template For our purpose to test a particular feature of Trello, we make use of the available template case test uAT test by San Francisco State University. The SCSU template helps testers identify, define, and execute UAT test cases based on defined requirements. The template ensures that all relevant information is available to the person performing the UAT test cases. Let's start by filling out this template for our particular testcase. 1. UAT Scope First, we need to define the UAT scope of our particular testing case. Since we are responsible for testing the new Trello Card Archive feature, we define the list of features that we want to test as well as those that are not tested. After you see below, this example of case testing focuses only on desktop tests, while mobile tests are not included in this UAT test case. 2. UAT Hypotheses and Constraints After defining our UAT scope, we must be aware of the testing assumptions and constraints. These assumptions and constraints include the timing and resources available, and test documentation processes. For our example of testing the Trello archiving feature on the desktop, the constraints could be the operating system to be used and certain browser versions. In addition, we must also state clearly the defined assumptions, such as how our test environment looks or how a tester should manage error reporting. 3. UAT Risks As UAT is a very crucial part of software software cycle, we need to look at the potential risks of UAT while planning, executing and analyzing our UAT testcase. In our example we may face some of the following risks during UAT: Properly trained UAT Testers: Our UAT Testers may not be properly trained and may not have complete knowledge of business and user needs. Incomplete UAT environment: Due to lack of resources and time constraints, tests may be incomplete by the deadline. Error handling: Testers may not know how to correctly report errors and errors during testing. UAT Test Error: Sometimes developed features are incomplete and lack too many bugs that do not allow a test case to be completed at all. 4. UAT Team Roles & Responsibilities Participants in a UAT team may vary from project to project. The entire UAT team will be responsible for coordinating the preparation of all test cases and its execution. Our UAT team will ensure ... that defined test cases are planned and carried out accordingly so that test results are documented and shared between project team 5. Input Criteria UAT Entry criteria ensure that everything is in place, which allows you to conduct a user acceptance test. To start a UAT test for the archiving feature of Trello, the following main criteria must be available: The development of the archiving feature is fully completed All reported bugs are fixed The UAT test environment is available & testers are trained to start the UAT 6 test. UAT Requirements-based test cases While all previous steps aim to ensure a well executable test case, now is the time to describe the actual test case itself to be performed successfully by UAT testers. As user accepting tests purpose to verify that business and user requirements are met, a case test procedure could look like this: Please visit: Trello.com Connect with the following credentials: Mike 123 Open available Trello board Archive first Trello card on the expected result board: This Trello card will be removed from the board This Trello card will be moved to the Archive To execute a test case , testers follow the test cases described step by step. Ideally, the tester can successfully run the test. However, an important part of UAT testing is to report unexpected behaviors of the software, or even actual bugs. UAT test tool, such as Usersnap, support UAT testers to report errors, track bugs, and other flaws while running a UAT test case without even leaving the test context. Usersnap is present in the app and can be invoked every time the tester experiences errors or With Usersnap in place, you can run a test case in an effective way, while all bugs and errors are reported on the flight without leaving the test scenario. Because all created error reports are sent directly to the project dashboard, project managers and developers can easily replicate identified errors and fix them in a timely manner. 7. Results of UAT tests After successful completion of UAT, UAT, the tester must provide the test results. A well-documented UAT test case allows the product or project team to complete the following steps and define the next actions based on the test results. 8. The signatures of the Last document, but not least, both the service owner and the project manager must sign the test cases carried out. After disconnection, the tested feature is good to be available for production. By wrapping everything in everything, we've shown you all the relevant steps for performing a UAT test case. You can find fully performed UAT example case test for Trello here (and as a PDF version here). From identifying the scope of the UAT and the risks to describing and executing certain test cases, such an example of user acceptance testing shows you the necessary and practical tasks. Bonus: Get your Usersnap UAT solution To get started with UAT, I'd love to recommend our own solution use acceptance test from Usersnap. Used by small and large companies, such as Microsoft, Hawaiian Airlines and others, helps you manage your UAT efforts. Find out what user acceptance test (UAT) is, along with its definition, types, steps, and examples: my number one rule when trying to understand a new concept is that: the name is always going to be relevant and mostly a literal meaning (in the technical context). Find out what it is, will give you an initial understanding of it and help me start with.=> Click Here for the full Tutorial Series test plan puts us this concept to test.=> Read all the tutorials in our acceptance test series. What is user acceptance testing? We know what testing is, acceptance means approval or agreement. The user in the context of a software product is either the consumer of the software or the person who requested that it be built for him/her (customer). So following my rule – the definition will be: User Acceptance Test (UAT), also known as beta or end user testing, is defined as testing the software by the user or client to determine whether it can be accepted or not. This is the final testing performed once functional, system and regression testing is completed. The main purpose of this testing is to validate the software in relation to business requirements. This validation is performed by end users who are familiar with business requirements. UAT, alpha and beta testing are different types of acceptance testing. Since the user acceptance test is the last test that is performed before the software is live, obviously this is the last chance for the customer to test the software and measure whether it is suitable for this purpose. When it is This is usually the last step before the product go live or before the delivery of the product is accepted. This is done after the product itself is thoroughly tested (i.e. after testing the system). Who performs the UAT? Users or client - This could be either someone who is buying a product (in the case of commercial software) or someone who had a custom-built software through a service provider or end user, if the software is made available to them ahead of time and when searching for their feedback. The team can be made up of beta testers or the client should select internal UAT members from each group of the organization so that each user role can be tested accordingly. Need For User Acceptance TestingDevelopers and functional testers are technical people who validate the software in relation to functional specifications. They interpret the requirements according to their knowledge and develop/test the software (here is the importance of domain knowledge). This software is completely in accordance with functional specifications, but there are some business requirements and processes that are known only by end users are either missed to communicate or misinterpreted. This testing plays an important role in validating whether or not all business requirements are met before the software is released for use on the market. The use of live data and cases of actual use make this test an important part of the launch cycle. Many companies that have suffered large losses due to post-release issues know the importance of a successful user acceptance test. The cost of fixing defects after release is often higher than fixing forward. Is UAT really necessary? After performing system tasks, integration and regression testing would ask about the necessity of this test. Actually speaking, this is the most important phase of the project because this is the time when users who are actually going to use the system would validate the system for its fit to the end. The UAT is a testing phase that depends to a large extent on the perspective of end-users and the domain knowledge of a department representing end-users. As a matter of fact, it would really be useful for business teams, if they were involved in the project early enough so that they could offer their opinions and contributions that would help the efficient use of the system in the real world. The process of testing user acceptanceThe easiest way to understand this process is to think of it as an autonomous testing project – which means it will have the plan, design and execution phases. The following are the prerequisites before the start of the planning phase:#1) Gather the key acceptance criteriaIn simple terms, the Acceptance Criteria is a list of things that will be evaluated before accepting the product. These could be of 2 types:(i) Functionality of the application or Business RelatedIdeally, all key business functionality should get validated, but for various reasons, including time, it is not practical to do everything. Therefore, a meeting or two with or users who will be involved in this testing can give us an idea of how much testing will be involved and what aspects will be tested. (ii) Contractually - We will not go into this and the involvement of the QA team in all this is almost nothing. The initial value of the which is drawn up just before the start of the SDLC is reviewed and an agreement is reached as to whether all aspects of the contract have been delivered or not. We will focus only on the functionality of the application.#2) We define the scope of QA involvement. QA team role is one of the following: (i) No involvement - This is very rare. (ii) Assist this test – The most common. In this case, our involvement could be training UAT users on how to use the app and be on standby during this test to make sure that we can help users in case of any difficulty. Or in some cases, in addition to being waiting and assisting, we could share their responses and record results or log bugs, etc., while users perform actual testing. (iii) UAT performance and present results – If applicable, users will indicate the aut areas they wish to assess and the assessment itself is carried out by the QA team. Once achieved, the results are presented to customers/users and they will make a decision as to whether or not the results in their hands are sufficient and in line with their expectations to accept AUT. The decision is never that of the QA team. Depending on the case here, we decide which approach is best. Main objectives and expectations: Usually, UAT is undertaken by a subject expert (IMM) and/or a business user who could be the owner or client of a system being tested. Similar to the system testing phase, the UAT phase also comprises religious phases before being brought to closure. The key activities of each UAT phase are defined below:UAT GovernanceSimilar to system testing, effective governance is required for the UAT to ensure that the high quality gates, together with the defined input and output criteria (provided below **).** Please note that it is only an orientation. This could be modified according to the needs and requirements of the project. The planning of the UATS process tests is almost the same as with the regular test plan in the system phase. The most common approach followed in most projects is to plan for both the UAT system and testing phases together. For more information about the UAT test plan, along with a sample, please refer to the UAT sections of the attached test plan document. User Acceptance Test Plan (This is the same that we would find on our website for the QA training series as well). Click the image below and scroll down to find the sample of the test plan document in different formats. In this template, check the UAT section. Data, environment, actors (who), communication protocols, roles and templates, their results and analysis process, entry-exit criteria – all this and anything else relevant will be found in the UAT test plan. Whether the QA team participates, participates partially or not at all in this test, it is our task to plan this phase and ensure that everything is taken into account.=> Here is a sample of the user's user acceptance test plan Accept Test DesignFollowing acceptance criteria from users are used in this step. The samples might look like this is what it looks like below. (These are excerpts from CSTE CBOK. This is one of the best available references about this test.) User Acceptance Test Template: Based on the criteria, we (the QA team) provide users with a list of UAT test cases. These test cases are no different from our regular system testing cases. They are just a subset as I test all applications, unlike, only to key functional areas. In addition to these, the data, templates for recording test results, administrative procedures, fault recording mechanism, etc., must be in force before moving on to the next phase. Test ExecutionUsulally, when possible, this test happens in a conference or some kind of war room kind of set up where users, PM, representatives of the QA team all sit together for a day or two and work through all the acceptance test cases. Or if the QA team conducts the tests, we run the test cases on the AUT. Once all the tests are carried out and the results are in hand, the acceptance decision is made. This is also called the Go/No-Go decision. If users are satisfied it's a Go, or something else is a No-go.Reaching acceptance decision is usually at the end of this phase. Typically, the type of software tools that are used during this test ing phase is similar to the tools used during functional testing. Tools: Because this phase involves validating complete flows from one end of the application to the other, it may be difficult to have a single tool to fully automate this validation. However, to some extent, we would be able to leverage automated scripts developed during system testing. Similar to system testing, users would also use test management and fault management tool, such as QC, JIRA, etc. These tools can be configured to accumulate data for the user acceptance phase. Methodologies: Although conventional methodologies would be specific business users performing The UAT of the product is still relevant, in a truly global world like today, testing user acceptance sometimes has to involve varied customers across the country based on the product. For example, an e-commerce website would be used by customers around the world. In such scenarios, crowd testing would be the best viable option. Crowd testing is a methodology in which people from all over the world can participate and validate the use of the product and give suggestions and recommendations. Crowd testing platforms are built and are organisations now. A website or product that needs to be tested in the crowd is hosted in the platform, and customers can nominate to do the validation. The feedback provided is then analysed and prioritised. The crowd testing methodology proves to be more effective, as the customer's pulse around the world can be easy to understand. UAT In Agile environmentThe agile environment is more dynamic in nature. In an agile world, business users will be involved in project sprints and the project would be improved based on feedback loops from them. At the beginning of the project, business users would be the main stakeholders to provide the requirement, thereby updating the product arrears. During each sprint, business users would participate in the demo sprint and would be available for providing any feedback. In addition, a UAT phase would be planned before the sprint completion if business users would make their validations. Feedbacks that are received during sprint demo and sprint UAT, are collected and added back to the outstanding product, which is constantly reviewed and prioritized. Thus, in an agile world, business users are closer to the project and evaluate the same for its use more frequently, as opposed to traditional waterfall projects. The UAT – Roles & ResponsibilitiesA typical UAT organization would have the following roles and responsibilities. The UAT team would be supported by the project manager, Development and testing teams based on their needs.7 UAT challenges and mitigation planIt matters whether you are a part of a billion-dollar version or a starting team, should overcome all these challenges for providing successful software for the ultimate-user.#1) Environment configuration and implementation process: Performing this test in the same environment used by the functional test team will certainly end up overlooking real-world use cases. Also, crucial testing activities, such as performance testing, cannot be performed on a test environment with incomplete test data. A separate production medium should be established for this test. Once the UAT environment is separated from the test environment, you must effectively control the release cycle. The uncontrolled release cycle can lead to different versions of software on the test environment and UAT. Valuable test ing acceptance time is lost when the software is not tested on the latest version. In the meantime, the time it takes to track problems on the incorrect software version is high.#2) Test planning: This test should be planned with a clear acceptance test plan in the analysis of requirements and the design phase. In strategy planning, the set of real-world usage cases should be identified for execution. It is very important to define the test ing objectives for this test, since full execution of the test is not possible for large applications at this test stage. Testing should be carried out first by prioritizing critical business objectives. This test shall be carried out at the end of the test cycle. Obviously, it's the most critical time for launching the software. Delay in any of the previous stages of development and testing eat up uAT time. Inadequate test planning, in the worst cases, leads to an overlap between system testing and UAT. Due to less time and pressure to meet deadlines, the software is implemented in this environment, even if functional testing is not completed. The basic objectives of this test cannot be achieved in such The UAT test plan must be prepared and communicated to the team well before the start of this test. This will help them to plan tests, write case tests & test scripts and create a UAT.#3) Handling new business requirements as incidents/flaws:Ambiguities in requirements are caught in the UAT phase. UAT testers find problems that arise because of ambiguous requirements (looking at the full interface that was not available during the requirement collection phase) and record it as a defect. The client expects these to be fixed in the current version without taking into account the time for change requests. If a timely decision is not made by project management on these last-minute changes, then this could lead to launch failure.#4) unqualified testers or testers without business knowledge:When there is no permanent team, the company selects UAT staff from different internal departments. Even if the staff is well acquainted with business needs, or if they are not trained for the new requirements that are developed, they cannot perform effective UATs. A non-technical business team may also face many technical difficulties in the execution of test cases. Meanwhile, the attribution of testers at the end of the UAT cycle does not add any value to the project. Little time to train UAT staff can significantly increase the chances of success UAT.#5) Inadequate communication channel: Communication between remote development, testing and UAT team is more difficult. Email communication is often very difficult when you have an offshore technology team. A little ambiguity in incident reports may delay its remediation for a day. Proper planning and effective communication are essential for effective teamwork. Project teams should use a web tool to record flaws and questions. This will help distribute workload equally and avoid reporting duplicate problems.#6) Asking functional test team to perform this test:There is no worse situation than requiring the functional test team to perform UAT. Customers discharge their responsibility to the test team due to lack of resources. The whole purpose of this test is compromised in such cases. Once the software goes live, end users will quickly spot problems that are not considered as real-world scenarios by functional testers. One solution to this is to assign this test to dedicated and qualified testers who have knowledge.#7) Blame GameSometimes business users just try to find reasons to reject the software. Could it be their autodom to show how are either blame the development and testing team to gain respect in the business team. This is very rare, but it happens in teams with domestic politics. It's very difficult to deal with situations like this. However, building a positive relationship with the business team would definitely help to avoid the game's fault. I hope these guidelines will certainly help you execute a successful user acceptance plan by overcoming various challenges. Corresponding communication, execution and motivated team are the keys to successful user acceptance testing. Testing of the system Vs Testing of user acceptanceThe test team's application starts early on in the requirements analysis phase. Throughout the life cycle of the project, a kind of validation is carried out for the project, namely static testing, unit testing, system testing, integration testing, end-to-end testing or regression testing. This leaves us to better understand about the testing carried out in the UAT phase and how different it is from the other tests performed earlier. While we see the differences between SIT and UAT, it is important to build on synergies, but to maintain independence between the two phases, which would allow for a shorter marketing time. Conclusion #1) UAT is not about pages, fields or buttons. The underlying assumption just before this test starts is that all that basic stuff is tested and is working fine. God forbid, users find a bug as basic as this - it's a piece of really bad news for the QA team. :(#2) This test refers to the entity that is the main element in the business. Let me give you an example: If AUT is a ticketing system, UAT will not be about, searching for the menu that opens a page, etc. It's about tickets and their reservation, the states you can take, travel through the system, etc. Another example, if the site is a car dealer site, then the focus is on the car and its sales and not really the site. Therefore, basic business is what is verified and validated and which is better to do it than business owners. That's why this testing makes the most sense when the client is involved to a major extent.#3) UAT is also a form of basic testing, which means there is a good chance to identify some bugs in this phase too. Sometimes it happens. Aside from the fact that there is a major escalation on the QA team, UAT bugs usually mean a meeting to sit and discuss to deal with them as following this test there is usually no time to fix and retest. The decision would be either to:Push go-live date, fix the problem first and then move on. Leave the cockroach like that. Consider as part of the demand for change for future versions.#4) UAT is classified as Alpha and Beta testing, but this classification is not so important in the context of typical software development projects in a service-based industry. Alpha testing is when UAT is performed in the software manufacturer's environment and is more significant in the context of commercial software on the shelf. Beta testing is when UAT is performed in the production environment or in the customer environment. This is more for customer-oriented applications. Users here are real customers like you and me in this context.#5) Most of the time in a regular software development project, UAT is performed in the QA environment where there is no staging or uAT environment. In short, briefly, The way to find out if your product is acceptable and fit for purpose is to actually put in front of users.Organizations are getting into agile delivery mode, business users get more involved and projects are being improved and delivered through feedback loops. All being done, the user acceptance phase is considered as the gateway to get into implementation and production. What was your UAT experience? Have you been on hold or tested for your users? Did users find problems? If so, would you deal with them?=> Read also all the tutorials in this series here = > Visit here for the full Tutorial Series test ing plan