[Add images as they appear in [Best Practices for All Documentation document](https://docs.google.com/document/d/1GkFT3AYeMTforxrGPXCd8KDfP4AS8Ad-mcICTguCN-g/edit)]

**Best Practices for User Acceptance Testing**

Recommendations from the Scalable Data Use Workgroup  
On Standards for UAT

Created August 2024

As part of the FY23 SLDS Scalable Data Use   
supplemental grant for the CEDS Collaborative Exchange

<https://github.com/CEDStandards>

**Table of Contents**

[Table of Contents 2](#yu75qmkfpbfl)

[User Acceptance Testing (UAT) Plan 3](#_6ey3ax20hoiv)

[1. Introduction 3](#_ogxqlbgpqnfi)

[1.1 Purpose 3](#_8ndzg9pz1ag4)

[1.2 Scope 3](#_knw8kem5xb3g)

[2. UAT Objectives 3](#_fajpyihkfej6)

[2.1. Verify that the software solution meets the agreed business requirements. 3](#_7gctx87jbutb)

[2.2 Identify and resolve any defects or issues before deployment. 4](#_oyb01zkato4e)

[2.3 Ensure that the software performs as expected in real-world scenarios. 4](#_6i8jlqs2x2im)

[2.4 Gain formal approval from stakeholders to proceed to production. 4](#_vvgm4yedm1wd)

[3. UAT Approach 4](#_hikije3frfbv)

[3.1 Testing Types 4](#_zi4g8npbrv8h)

[3.2 Test Environment 5](#_k1eqome2ptbe)

[4. Roles and Responsibilities 5](#_g01hhl15sq2c)

[4.1 Project Manager: 5](#_1no1ls51fnwx)

[4.2 UAT Manager: 5](#_3t47b4p5r5b0)

[4.3 Business Analysts: 5](#_7gzkp3wtvea3)

[4.4 Developers: 6](#_yvuo0nop54wc)

[4.5 Testers (End Users): 6](#_xat3j2gbs93f)

[5. Use Case 6](#_4pgdr82sl9vh)

[6. Test Plan 6](#_v4vk1xye76a2)

[6.1 Test Plan Definition 6](#_49dlqebu5f0l)

[6.2 Test Cases 6](#_4jpwdgo4zsbm)

[6.3 Test Data 6](#_ticidyz4d27y)

[6.4 Test Schedule 7](#_gpqdrlxdpskd)

[7. Entry and Exit Criteria 7](#_j024pbviv7mo)

[7.1 Entry Criteria 7](#_2905ro2ex0da)

[7.2 Exit Criteria 7](#_rn2d1l7va4nn)

[8. Defect Management 7](#_wyhlct1s6lnr)

[8.1 Defect Reporting 7](#_mi7thotzq07d)

[8.2 Defect Triage 8](#_uy0q8g5qw1ws)

[9. Risks and Mitigations 8](#_z3wr6v2i5fbe)

[10. Sign-Off 8](#_vvraklypby8c)

[☑️Link to UAT Template Draft](https://drive.google.com/file/d/1sPOWXovjgjoNeIaILWsV7G_NzUw8D6NI/view)

[☑️Link to UAT Template Example](https://drive.google.com/drive/folders/1-trhkz5UutgigXi-mCVtdVSilfJAL0el)

NOTE: This template can be customized to suit the specific needs of your project and may include best practices document for User Acceptance testing and/or other fillable documentation.

# **User Acceptance Testing (UAT) Plan**

## **1.** **Introduction**

### **1.1 Purpose**

The purpose of this User Acceptance Testing (UAT) plan is to define the scope, approach, resources, and schedule of the testing activities. The primary goal of UAT is to ensure that the software meets the business requirements and is ready for deployment. It encompasses the following key aspects:

**Scope** – clearly defines what functionalities or features that will be tested.

**Approach** – specify the testing methodology, including test scenarios, test cases, and acceptance testing.

**Resources** – identifies team members responsible for UAT, roles, and any required tools or environments.

**Schedule** – sets the timeline for UAT activities, including start and end dates.

**Execution** – execution of test cases, validation of software behavior, and reporting and mitigation of defects.

### **1.2 Scope**

This document outlines the UAT process, including test objectives, criteria for acceptance, and roles and responsibilities. It covers all functionalities and features that need to be tested to validate the software against user requirements.

## **2.** **UAT Objectives**

### **2.1. Verify the software solution meets the agreed business requirements.**

1. Ensure the software aligns with the business goals and user needs as documented in the requirements specification.
2. Validate that all functional requirements, use cases, and user stories are correctly implemented.
3. Confirm that non-functional requirements such as performance, usability, and security are addressed.

### **2.2 Identify and resolve any defects or issues before deployment.**

1. Perform thorough testing to uncover any bugs, inconsistencies, or discrepancies in the software.
2. Document all identified issues, prioritize them based on severity, and track them to resolution.
3. Ensure that fixes and updates do not introduce new issues or regressions.

### **2.3 Ensure the software performs as expected in real-world scenarios.**

1. Conduct tests that mimic actual usage conditions, including various user roles, data loads, and interaction patterns.
2. Validate software behavior under normal and peak loads to ensure stability and performance.
3. Test integration with other systems, interfaces, and third-party services to ensure seamless operation.

### **2.4 Gain formal approval from stakeholders to proceed to production.**

1. Present test results, including any unresolved issues, to stakeholders for review and feedback.
2. Demonstrate the software meets all acceptance criteria and is ready for production use.
3. Obtain sign-off from key stakeholders, including business owners, project managers, and end-users, to confirm their satisfaction and readiness to deploy the software.

## **3.** **UAT Approach**

### **3.1 Testing Types**

* **Functional Testing:** Validate that the software functions according to the specified requirements.
* **Usability Testing:** Ensure the software is user-friendly and intuitive.
* **Performance Testing:** Verify that the software performs well under expected load conditions.
* **Regression Testing:** Ensure that new changes have not adversely affected existing functionality.

### **3.2** **Test Environment**

Specify the environment in which UAT will be conducted, including hardware, software, network configurations, and any other relevant details.

## **4. Roles and Responsibilities**

All roles may not be applicable or possible in all institutions, but responsibilities should be shared between available staff.

### **4.1 Project Manager:**

1. **Ensure alignment with project schedule and objectives:** Make sure that UAT activities are integrated into the overall project plan and timeline.
2. **Resource allocation:** Ensure that the necessary resources (e.g., time, personnel, tools) are available for successful UAT execution.
3. **Monitor progress:** Track the progress of UAT activities and address any deviations from the plan.
4. **Stakeholder communication**: Maintain regular communication with all stakeholders, providing updates on UAT status and addressing any concerns.
5. **Manage scope creep** by not allowing complete changes in process or new additions that may come up during testing.

### **4.2** **UAT Manager:**

1. **Oversee the UAT process**: Ensure the entire UAT process is properly managed from planning to execution and closure.
2. **Coordinate activities**: Organize and facilitate UAT meetings, ensuring effective communication among all team members.
3. **Testing Plan**: Monitor adherence to the UAT plan and schedule, making adjustments as necessary to stay on track.
4. **Issue Resolution**: Assist in the prioritization and resolution of any issues or defects identified during testing.
5. **Reporting**: Provide regular status updates and comprehensive reports on UAT progress and outcomes to stakeholders.

### **4.3** **Business Analysts:**

1. **Provide clarification on requirements:** Serve as the primary point of contact for any questions or clarifications regarding business requirements.
2. **Assist with test case creation**: Collaborate with testers to develop comprehensive test cases that accurately identify software behavior, functionality, and identify any gaps or issues.
3. **Support during testing**: Offer ongoing support to testers during the execution of UAT, ensuring they understand the expected outcomes.

### **4.4** **Developers:**

1. **Fix defects identified during UAT**: Address and resolve any defects.
2. **Implement Feedback**: Make necessary adjustments and improvements based on user feedback and testing results.
3. **Support testing**: Provide technical assistance and support to the UAT team as needed, ensuring a smooth testing process.

### **4.5** **Testers (End Users):**

1. **Execute test cases**: Perform the testing tasks as per the defined test cases and scenarios.
2. **Report defects**: Document and report any defects, issues, or unexpected behaviors observed during testing.
3. **Provide feedback**: Offer feedback on the usability, functionality, and overall performance of the software.
4. **Verify Fixes**: Retest resolved issues to confirm that defects have been fixed and no new issues have been introduced.

## **5. Use Case**

An analysis that outlines the user problem, including the identification, clarification, and categorization of software or user requirements. It also details plausible sequences or steps to achieve a specified objective, ensuring a specific resolution or verifying that a feature functions as intended.

## **6.** **Test Plan**

### **6.1 Test Plan Definition**

A representative of cases whose outcomes are likely to serve as precedent. Detail the test cases to be executed, including test scenarios, test steps, expected results, user roles, and acceptance criteria.

### **6.2 Test Cases**

Test cases represent the final opportunity for end users to confirm requirements are met. An effective test case will target specific features, rules and data and ultimately yield a binary determination (e.g., the feature passed or failed testing). Time investment in building effective test cases will foster confidence during testing.

### **6.3** **Test Data**

Describe the data to be used for testing, including any specific data sets required to validate the functionality.

### **6.4** **Test Schedule**

Provide a timeline for UAT activities, including test case preparation, execution, defect resolution, and retesting.

## **7. Entry and Exit Criteria**

### **7.1 Entry Criteria**

* All functional requirements have been implemented and unit tested.
* Test environment is set up and ready.
* Test data is prepared.
* UAT test cases have been reviewed and approved.

### **7.2** **Exit Criteria**

* All critical and high-priority defects have been resolved.
* Test cases have been executed with a pass rate of [specific percentage].
* Formal sign-off has been received from stakeholders.

## **8. Defect Management**

### **8.1 Defect Reporting**

Describe the process for logging and tracking defects, including the use of any defect tracking tools.

1. **Date defect reported** - Include the date the defect is reported in the software, tracking sheet, etc.
2. **Subject/topic** of issue - Short and concise.
3. **Description** of issue - Defect information should be clear and concise.
4. **Requested priority** for the defect - Urgent, High, Medium, Low, None.
5. **Location within the product** where the defect resides (path).
6. Include the **steps to reproduce** - Spell out each step in order that was taken that produced the defect.
7. **Expected behavior** - What is the software expected to do?
8. **Actual behavior** - What is the software doing instead of what is expected (defect .itself)?
9. **Relevant logs or screenshots** - Specific information relevant to an error or issue that will help the development team.
10. **Who is impacted** by the defect - Explain the scope of who is impacted.
11. **Issue began** - Provide the date the issue either started or was discovered.
12. **Contact or person reporting** the defect - This allows development to reach out with further questions regarding the defect or process that produced the defect. It also allows for communication when the defect will be corrected and the version/release the fix will appear in.
13. **Resolution** - Resolution to the defect by the development team.
14. **Date resolved** - Date resolved by development team.
15. **Fix version** - In what version/release will the fix be released?

### **8.2** **Defect Triage**

Outline the process for prioritizing and addressing defects, including roles involved in triage meetings.

1. **Identification:** Test team identifies and logs defects.
2. **Categorization:** Defects are categorized by the triage team/group (SCRUM participants and possible stakeholders) based on impact and nature of the defect.
   1. Immediate Action - Considered urgent to send to development for a fix.
   2. Future Consideration - Defect/Error does not require immediate action but will need to be addressed in the future due to potential impact later on.
   3. Safe to Ignore - Does not require action; not a defect.
3. **Prioritization:** Defects should be prioritized based on the severity, business impact, frequency, and risk, etc.
4. **Assignment:** Defects are then assigned to the proper group within the development team for resolution.
5. **Stakeholder Involvement:** Dependent on the project and level of stakeholder involvement.

## **9.** **Risks and Mitigations**

Identify potential risks to the UAT process and describe mitigation strategies for each risk. Clear testing documentation and process.

When testing, the user needs to be able to clearly track and identify the steps taken which led to the failure both to ensure it was accurate and to assist the developer in correction. Identifying key pieces for regression testing after updates can also be crucial.

## **10. Sign-Off**

Include a section for formal sign-off by key stakeholders, confirming that UAT has been completed successfully and the software is ready for deployment.

|  |
| --- |

**Signatures:**

| **UAT Manager:** |  |
| --- | --- |
| **Project Manager:** |  |
| **Business Analyst:** |  |
| **Key Stakeholders:** |  |

|  |
| --- |