Test Plan for Testomat.io

version 1.0

Revision and Signoff Sheet

Document History:

Version #	Date	Changed By	Reason for change
0.1	25.07.2023	Shubina Nataliia	Creation

Approvers List:

Name	Role	Approved/ Reviewer	Approval/Review Data

Reference Document:

#	Document Name	Link
1	User Guide	https://docs.testomat.io/getting-started/start-from-scratch

Content

1. Introduction

1.1. Purpose

This document clearly defines the testing approach and procedures that will be used in the testing process of the Testomat.io project.

The test plan includes:

Test Strategy:

Identify approach for testing specify the type of testing that we will use; rules the test will be based on, including the givens of the project; description of the process to set up a valid test (entry and exit criterias, specific creation of test cases, scheduling).

Execution Strategy:

Formulating workflow processes to align with strategic objectives (e.g. describe performing tests; process to identify and report defects; re-test bugs).

Test Management:

Process helps closely organize, control, analyse, and trace the testing process while managing testing resources (communications, risk and mitigation, identify responsibilities).

1.2. Project Overview

Testomat.io is a test management system for automated tests. It links manual test cases to test auto. This testing tool allows managing BA, Dev, QA and non-tech stakeholders in one flow for maximizing the synergy of testing efforts in real-time.

In the ‘Testomatio’ website user can do following:

- create test documentations (test cases; checklist) and organize them to suites; create test plans
- create test runs
- import tests

- get analytics
- work with different branches
- Testomatio has integration with Jira (have: link tests to issues; list tests attached to issues view test coverage (all tests to issues); view run status for tests; create a test from an issue; edit a test; built-in Gherkin editor for BDD projects)
- integrations with modern js automated test frameworks, especially CodeceptJS, Cypress+BDD, WebdriverIO, Playwright, JUnit, TestCafe, Jest, Mocha, Chai, Jasmin, etc.
- JUnit interaction, through the JUnit XML file you are able to generate reports for any language or other test framework.
- Build-in CI\CD integrations (GitHub, GitLab, Bamboo, Jenkins, CircleCI)
- Advanced bi-directional Jira Plugin, Living Documentation, Confluence integrations

2. Test strategy

2.1. Test objectives

1. Test suites
2. Test folders
3. Tests
4. Search
    a. Tests
    b. Suites
    c. Folders
5. Filters
    a. Tests
    b. Suites
    c. Folders
6. Run manual tests

2.2. Test assumptions

General:

- The project will provide test planning, test design and test execution support.
- Exploratory Testing would be carried out once the build is ready for testing.
- Test case design activities will be performed by QA Group;
- Test team will manage the testing effort with close coordination with Project Manager and Business analyst
- Project team has the knowledge and experience necessary, or has received adequate training in the system, the project and the testing processes.
- Performance testing is not considered for this estimation
- The defects will be tracked through Jira Software only

2.3. Scope and Levels of Testing

2.3.1. Exploratory Testing

This level helps testers avoid the limitations of test cases, allows them to further explore certain features, and ensures that they don't miss cases that could lead to critical failures.

Tools: Use Google Spreadsheets or Chrome extension (Exploratory testing)

Testers: QA Team

Method: Testing should be done without test scripts

Use: Executed at the beginning of each sprint. The time for this type of testing depends on the estimate for each sprint

Template for Exploratory testing:

Test Environment URL:	Convert to PDF
Start:	23.07.2023 16:40:00
Tester:	**Shubina Nataliia**
Duration:	40 minutes
Testing Notes:	Table below

Areas:
Choose file from explorer
Drop file
Import from Google Drive / Dropbox
Add several files
Download PDF
Restart
Erase PDF
Save PDF to Google Drive / Dropbox

Notes	Status	Bugs
Convert to PDF valid file format	Pass	
Convert to PDF different file sizes (weight)	Pass	
Convert to PDF with invalid formats	Fail	BUG #1 https://prnt.sc/E6qHKmvwfZGx
Drop files to file area	Pass	
Import from Google Drive/ Dropbox	Fail	BUG #2 https://prnt.sc/iP1jB4SS6dCp
Upload file using all available ways (drop file,import, upload)	Pass	
Convert two and more files	Pass	
Show imported/uploaded image enlarged	Fail	BUG #3 https://prnt.sc/hCHs_-e77z-X https://prnt.sc/5lh6o4YLM3d5
Delete uploaded /imported file	Pass	
Download pdf	Fail	BUG #4 https://prnt.sc/UIZhvzAVuHg7
Preview (pdf with image)	Pass	
Erase	Pass	
Restart	Pass	
Save to Google Drive/Dropbox	Pass	

2.3.2. Functionality Testing

Functionality testing is based on testing each feature of the software application by providing the appropriate input and checking the output against the functional requirements.

Tools: TMS Testomatio.

Testers: QA Team

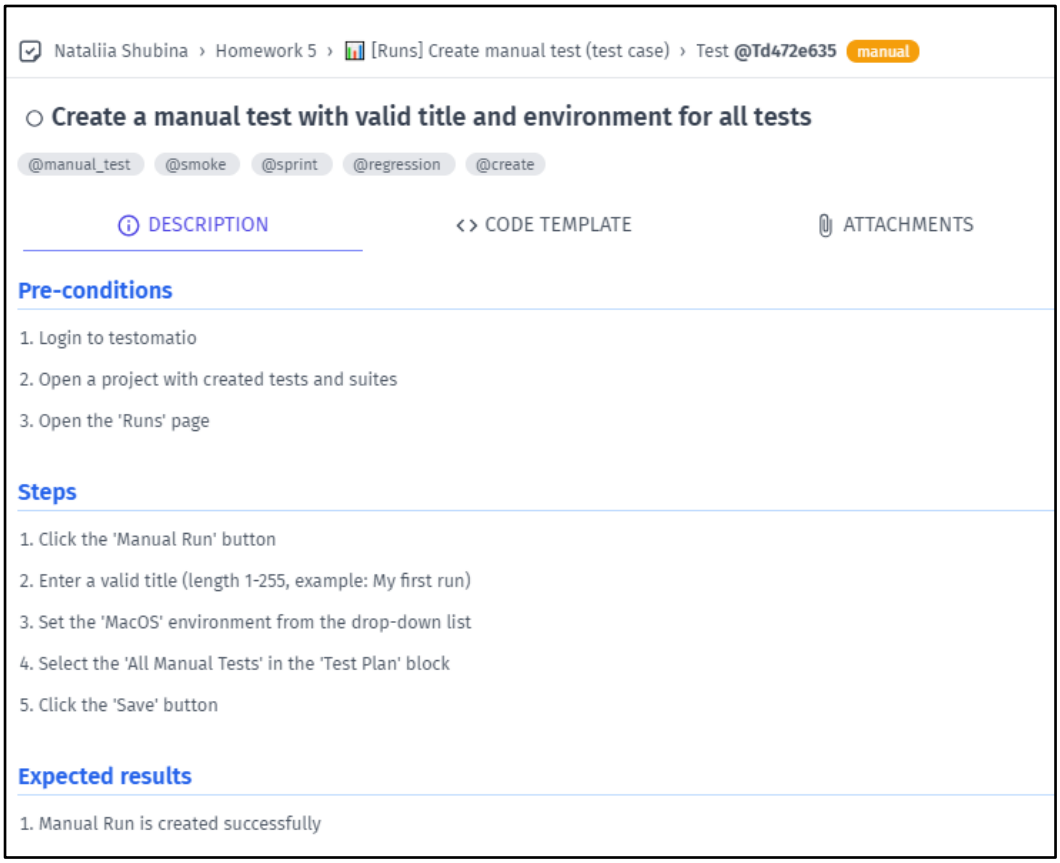
Method: The test will be performed according to the Test Cases, Checklists, which are stored in testomat.io.

Use: Functionality testing user after exploratory testing

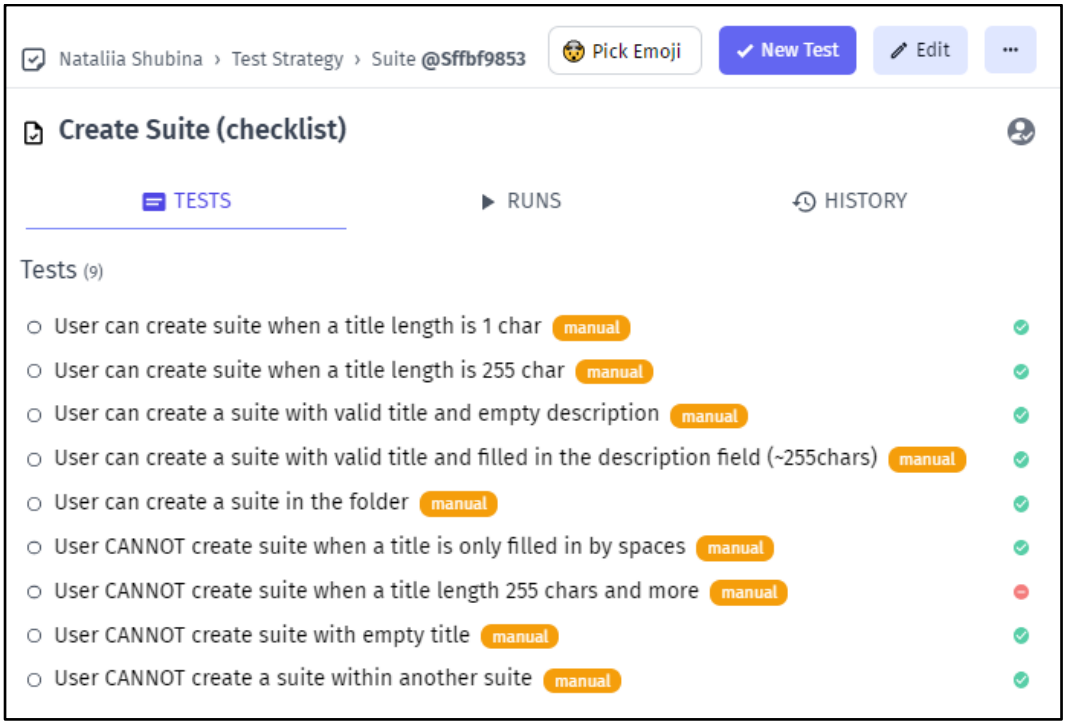
2.3.2.1. Test Design

As part of the test design, the test team creates test cases, checklists and mind maps. QA team uses various test design techniques (equivalent classes, boundary values, decision table, pairwise). Use positive and negative scenarios.

Test Case example:



Checklist example:



2.3.3. User Acceptance Test (UAT)

This testing focuses on validating the business logic. It allows the end users to complete one final review of the system prior to deployment.

Testers: The UAT is performed by the end users.

Method: Since the business users are the most indicated to provide input around business needs and how the system adapts to them, it may happen that the users do some validation not contained in the scripts. Test team write the UAT test cases based on the inputs from End user and Business Analyst's.

Use: After all other levels of testing are done. Only after this test is completed the product can be released to production.

2.3.4. Stages of testing

Planned stages of the testing process:

1. Analysis of requirements for a software product and design for the project; drafting a test plan.
2. Smoke testing
3. Make Functional/ Non Functional testing performed during to sprint (with the detection and description of defects)
4. Regression testing

2.3.4. Test environments

The environment for QA testing is QA: {url}

The environment for developers is Staging: {url}

The environment for beta publish on prod is Pre-prod: {url}

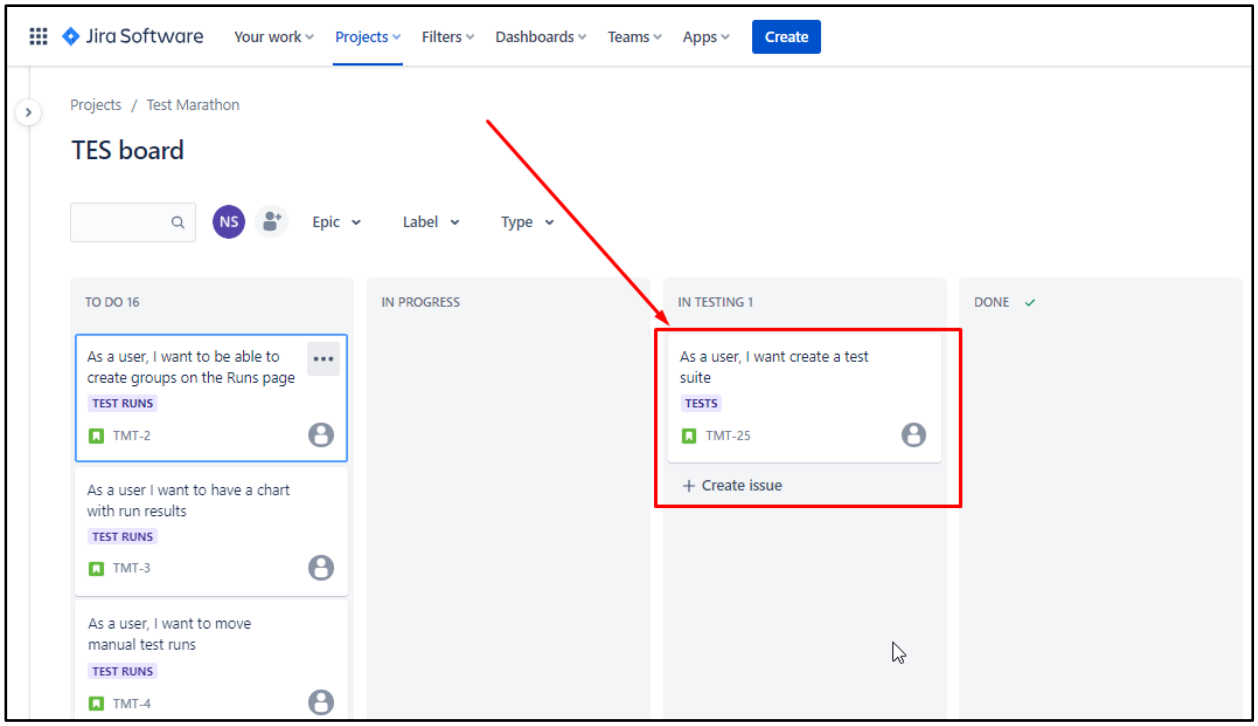
The environment for customer is Prod: https://app.testomat.io/

2.4. Tools

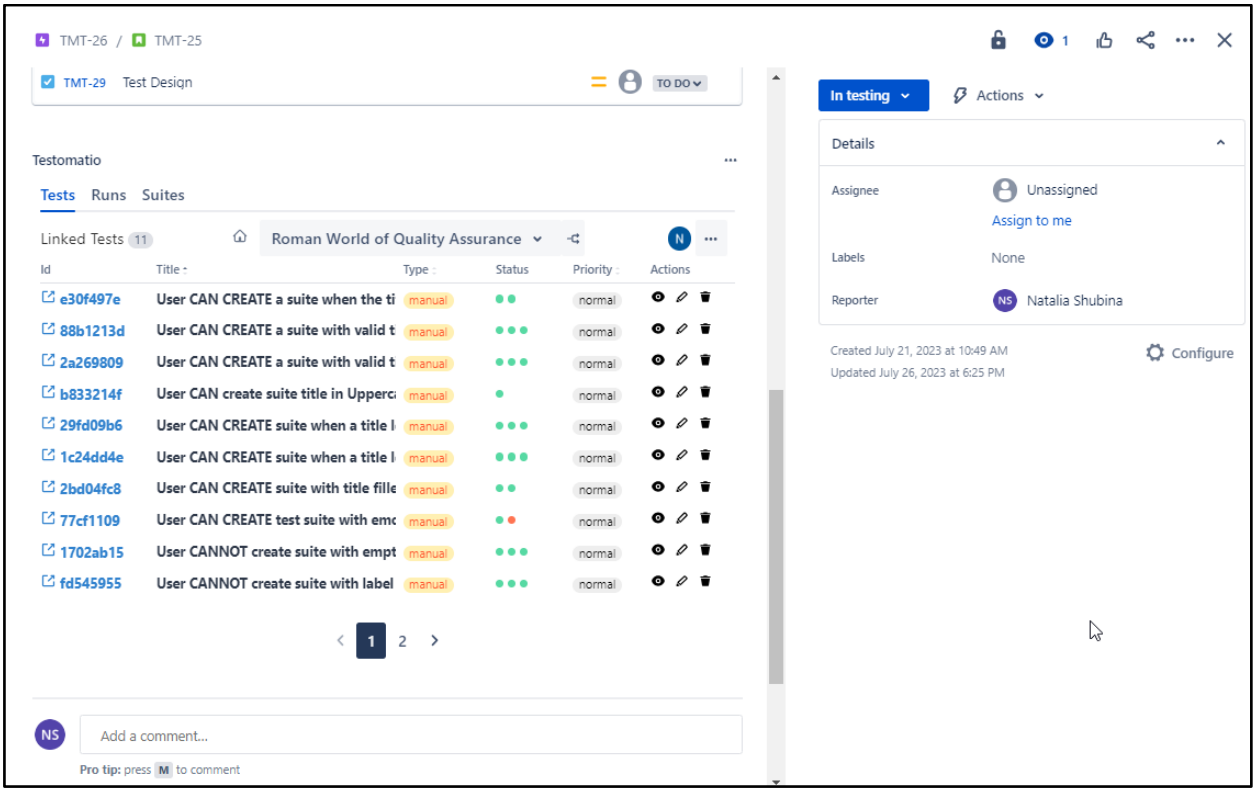
- Testomat.io
- Mind Map (miro.com)
- Jira
- SQLite
- Postman
- Chrome Devtools
- Google word and Google Sheets
- ShareX (for screen capture, gifs and videos)

3. Test execution

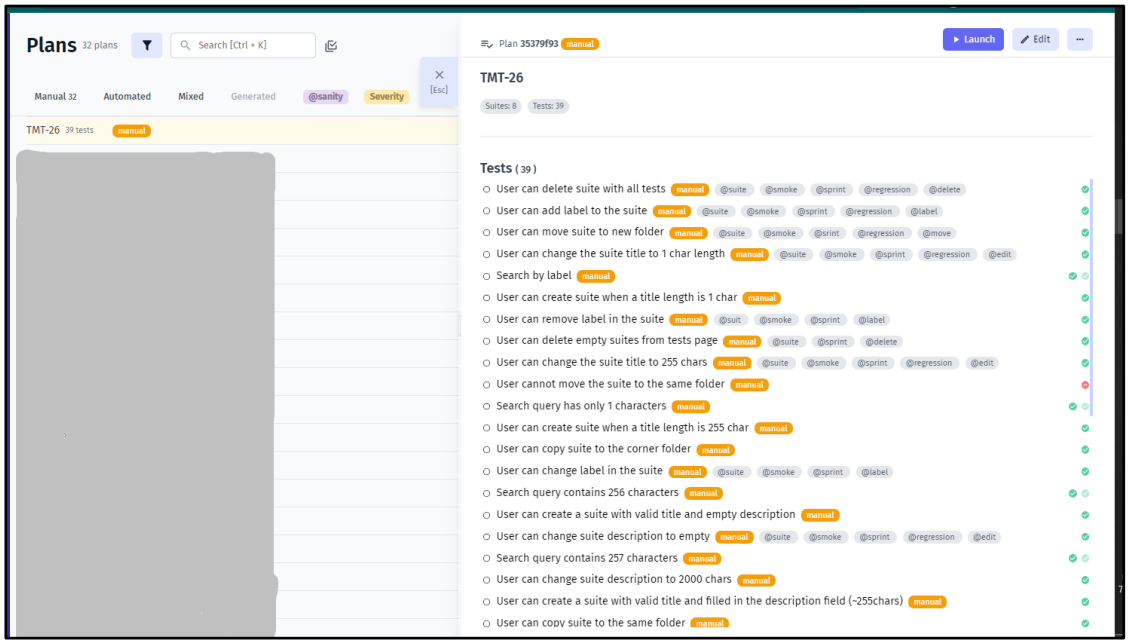
We start execution when User story moved to In Testing column



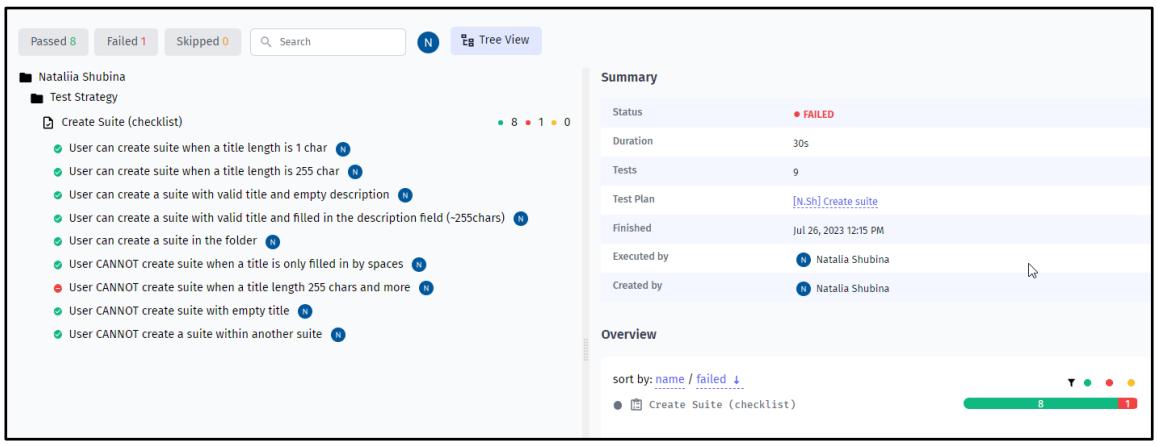
Link checklists with User Stories (link testomat.io and JIRA)



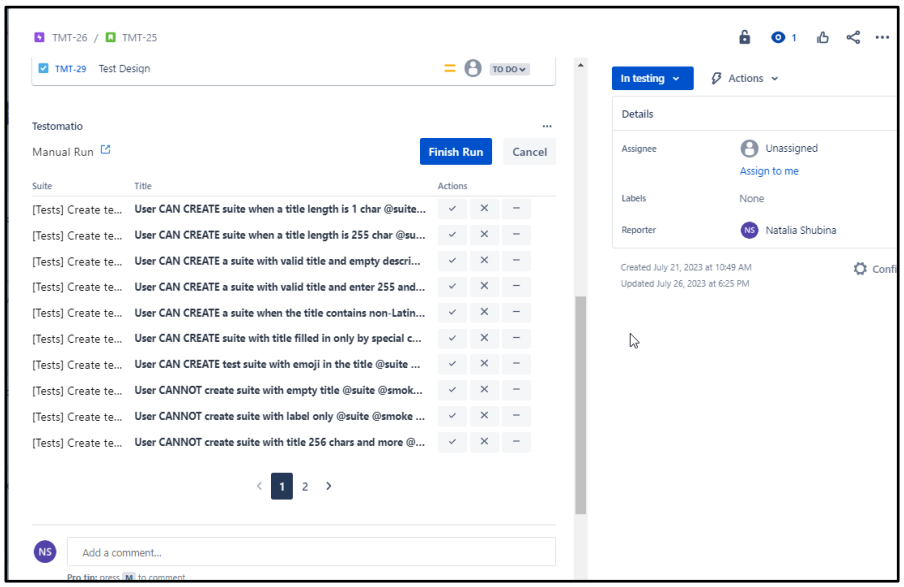
Create Test Plan in testomat.io



Create test run in Testomatio:



Create test run in Jira:



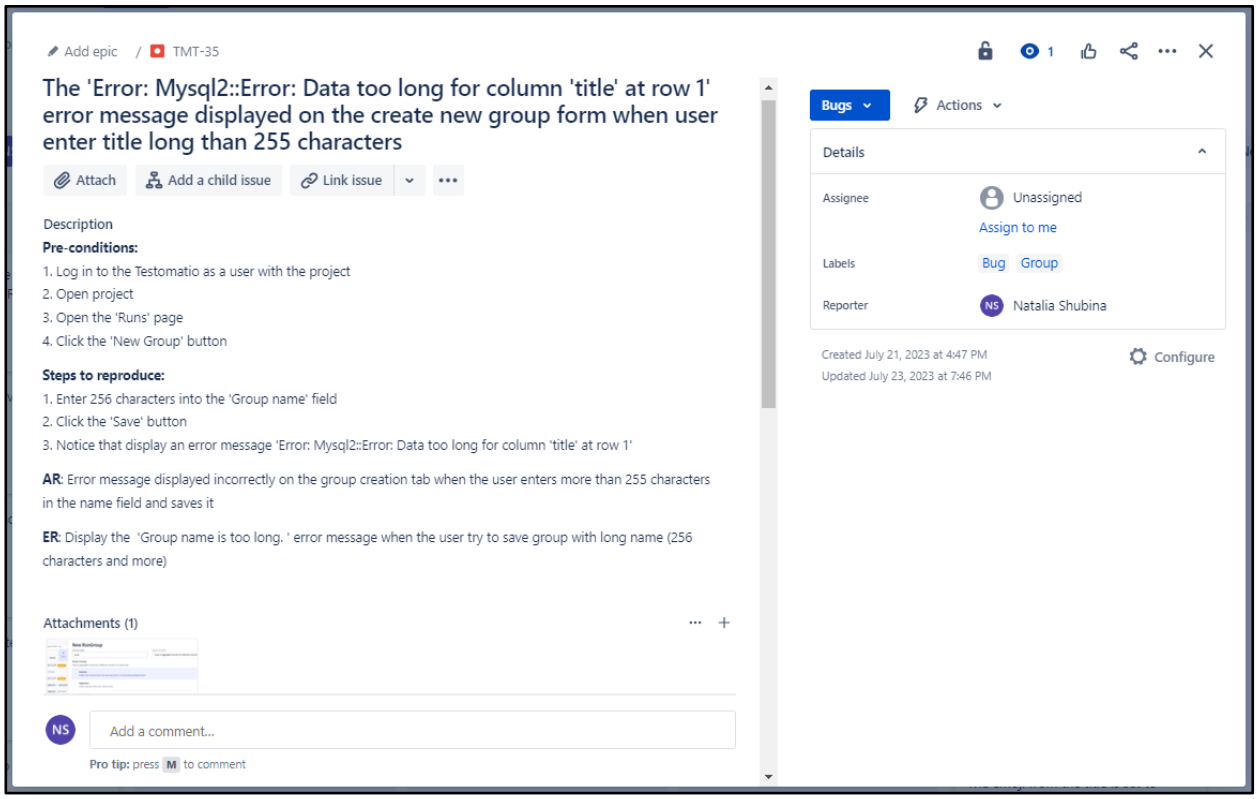
4. Defect tracking and Reporting

4.1. Basic requirements to bug-report

Bug report should contain the following information:

- Summary
- Pre-conditions
- Steps to reproduce
- Actual result
- Expected result
- Attachments
- Link to the test case and task (in jira)
- Label
- Priority

Bug report example:



5. Test risks and mitigation factors

#	Risk	Mitigation Plan
1	Customer can change the basic requirements to the project of one of testing features	Some buffer has been added to the schedule for contingencies
2	The customer has reduced costs for develop the project	Review all tasks. Remove some functional and change priority for tasks
3	More time spent on review than expected	Some buffer has been added to the schedule for contingencies
4	Delay testing due to design tasks	Some buffer has been added to the schedule for contingencies

6. Entry and Exit Criteria

6.1. Entry criteria

Testing can be started if the following entry criteria are met:

1. The necessary documentation is ready, approved and allowed;
2. Scope of work defined, described and prioritized;
3. Acceptance criteria defined.
4. Test cases/checklists are developed;
5. The feature is developed.

6.2. Exit criteria

Testing is completed if the following exit criteria are met:

1. All detected bugs are reported;
2. Ensuring all critical test cases are passed successfully;
3. Deadlines meet or budget depleted;
4. Desired and sufficient coverage of the requirements and functionalities under the test.
5. Identifying and fixing all the high-priority defects;
6. Re-testing and closing all the high-priority defects.

7. Project Team and responsibilities

#	Role	Name	Responsibilities
1	PM		- Monitor the progress of the project; - Identify and resolve issues that arise.
2	BA		- Create Prototype and Specification.
3	Designer		- Creating images and layouts by hand or using design software
4	Frontend developer		- Create a part of the web page that is visible to the user; - Implement user logic. - Assure that all user input is validated before submitting to back-end
5	Backend developer		- Building and maintaining web applications; - Writing high-quality code for project functional; - Managing database; - Troubleshooting and debugging
6	QA Engineer		- Review software requirements; - Test Software to detect bugs; - Create test documentations (test plan, test reports, checklists, test cases, bug reports); - Execute tests; - Analyzing test results

8. Test environment

OS systems:

- Windows
- MacOS

Mobile:

- iOS
- Android

Devices:

- Models
- Resolution
- OS

Browsers:

- Chrome
- Safari
- Opera

9. Testing strategy

[Test Strategy]

	A	B	C	D	E	F	G	H	I	J	K	L
1	System object and actions	Priority	Strategy	Test Analysis		Test Execution			Non-functional testing			
2				Readliness	Link	Readliness	Link	Date of the testing	Usability	Performance	Load	Availability
3	Suite	1	One of the main objects of the Testomatio project. Test suites are a logical grouping of tests for different test scenarios. **For testing (techniques and approaches to testing)**: Create *Checklists; make Positive and negative testing; use the Equivalence classes or boundary values* techniques	100%		90%		26/7/2023	100%	100%	100%	100%
4	Create	1	One of the main actions for suite - creation. Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	100%	Link	89%	Link	26/7/2023				
5	Open	1	Use a checklist and exploratory testing	100%	Link	100%	Link	26/7/2023				
6	Edit	1	Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	100%	Link	89%	Link	26/7/2023				
7	Copy	2	Use *Exploratory testing*	100%	Link	100%	Link	26/7/2023				
8	Move	2	Use *Exploratory testing*	100%	Link	50%	Link	26/7/2023				
9	Add label	3	Cover bu checklist. Use *equivalent classes and Boundary values* techniques	100%	Link	100%	Link	26/7/2023				
10	Delete suite with all tests	1	Cover by checklists. Use *exploratory testing*	100%	Link	100%	Link	26/7/2023				
11	Tests	1	One of the main objectives of the Testomatio project is for businesses and users. Can create checklists or test cases and be in a suite. This object is important for the system, and users will often interact with it **For testing (techniques and approaches to testing):** *Create a S&T diagram; cover by checklist and test cases; Make Positive and negative testing; use test design techniques*	0%		0%			0%	0%	0%	0%
12	Create single test	1	Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	0%		0%						
13	Create bulk tests	2	Cover by checklists. Use the equivalent classes and Boundary values test design techniques	0%		0%						
14	Edit test	1	Cover by checklists. Use the equivalent classes and Boundary values test design techniques; pairwise	0%		0%						
15	Bulk edit test	2	Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	0%		0%						

	A	B	C	D	E	F	G	H	I	J	K	L
			For testing (techniques and approaches to testing): *Create a S&T diagram; cover by checklist and test cases; Make Positive and negative testing; use test design techniques*									
12	Create single test	1	Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	0%		0%						
13	Create bulk tests	2	Cover by checklists. Use the equivalent classes and Boundary values test design techniques	0%		0%						
14	Edit test	1	Cover by checklists. Use the equivalent classes and Boundary values test design techniques; pairwise	0%		0%						
15	Bulk edit test	2	Cover by checklists. Use the *equivalent classes and Boundary values* test design techniques	0%		0%						
16	Copy test	3	Covered by test case and checklist. Use the *Exploratory testing and Pairwise testing (Must agree with the team)* test design techniques	0%		0%						
17	Duplicate tests	3	Use *Exploratory testing*	0%		0%						
18	Move test	3	Use *Exploratory testing*	0%		0%						
19	Add label to the test	3	Use *Exploratory testing*	0%		0%						
20	Delete test	1	Cover by checklists. Use *exploratory* testing	0%		0%						
21	Search	2	The main purpose is to help the user find information according to the user's request. Cover by checklists. Use the *equivalent classes and Boundary values test design* techniques	0%		0%			0%	0%	0%	0%
22	Write request	1	Cover by checklists. Use the equivalent classes and Boundary values test design techniques	0%		0%						
23	Update request	2	Cover by checklists. Use the equivalent classes and Boundary values test design techniques	0%		0%						
24	Clear request	1	Use exploratory testing. Cover in checklist	0%		0%						
25	Filter	3	The main purpose is to help with filtration tests/suites/folders on the Tests page. Cover by checklists. Use the equivalent classes and Boundary values test design techniques	100%		100%		26/7/2023	0%	0%	0%	0%
26	Select filter	1	Use pairwise test design techniques and cover by checklists	100%	Link	100%	Link	26/7/2023				
27	Clear filter	1	Use exploratory testing	100%	Link	100%	Link	27/7/2023				
28			Total:	44%		41%			25%	25%	25%	25%

10. Approvals

The Names and Titles of all persons who must approve this plan.

Signature	Name	Role	Date