# 

**TEST PLAN**

<QAProject>

CONFIDENTIAL

18 pages

**Table of Contents**

1. **Introduction**
   1. **Overview**
   2. **Purpose of the Test Plan**
   3. **Objectives of the testing effort**
   4. **Scope of the testing**
   5. **Application description**
2. **Test Items and Features**
   1. **Test Items**
   2. **Features to be tested**
   3. **Features not to be tested**
3. **Approach**
   1. **Testing methodology or approach**
   2. **Testing levels**
   3. **Testing types to be used**
   4. **Testing types not to be used**
   5. **Testing techniques and tools**
   6. **Test Case development and management**
   7. **Defects development and management**
   8. **Test data management**
   9. **Execution and reporting**
4. **Enter/Exit/ Acceptance Criteria**
   1. **Enter Criteria**
   2. **Exit Criteria**
   3. **Acceptance Criteria**
5. **Environments needs**
   1. **Hardware and Software recourses**
   2. **Tools and required for testing**
6. **Responsibilities**
   1. **Rollers, responsibilities and schedule of testing team members**
   2. **Communication and collaboration channels for testing team**
   3. **Escalation and issue resolution procedure**
7. **Risk and Contingencies**
8. **Approvals**
9. **Change history**
10. **Introduction\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**
    1. **Overview**

This Test Plan is designed for Manual project “QAProject” web application.

This document prescribes:

* scope of the testing
* approach of the testing
* testing strategy
* resources (time/human/technique)
* team composition and responsibilities
* schedule of the testing activities
* risk analysis

This Test Plan is designed for:

|  |  |  |
| --- | --- | --- |
| Business representative | make corrections to the process at different needs and can take part in the testing, so that they can be sure that the business is showing the results | Person 1 |
| Testing Team | check the tasks specified in this document, plan to go on testing and be responsible for the result, also guarantee that the test plan is consistent with the design | QA Team |
| Development Team | guarantee that the result is consistent with the design, following the procedures for correcting defects | DevTeam |
| Technical Group | provide the testing environment | TechGroup |

* 1. **Purpose of the Test Plan**

Prioritization of testing tasks Timely planning of resource costs for testing Accounting for the required resources (software, hardware) required for testing Early consideration of risks that may arise during the implementation of the plan and implementation of a preventive strategy

* 1. **Objectives of the testing effort**

The objective of the testing is to check the correctness of the functionality on the different versions of browsers and various mobile devices and to check non-functional requirements.

* 1. **Scope of the testing**

In their work, the team refers to such documentation:

|  |  |
| --- | --- |
| **User Stories:** | |
| QAProject: Main page | <https://jira.ithillel.com/browse/U4QA23022023-125> |
| QAProject: User registration pop up | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-127?filter=allopenissues> |
| QAProject: Registered user sign in | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-128?filter=allopenissues> |
| QAProject: Restore user access | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-129?filter=allopenissues> |
| QAProject: Remove car(s) in garage view | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-137?filter=allopenissues> |
| QAProject: User Settings | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-139?filter=allopenissues> |
| QAProject: User Instructions | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-138?filter=allopenissues> |
| QAProject: Remove expenses | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-134?filter=allopenissues> |
| QAProject: Garage view with/without cars and possibility to add new cars | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-135?filter=allopenissues> |
| QAProject: Add expenses, edit and table view | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-133?filter=allopenissues> |
| QAProject: Guest log in | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-126?filter=allopenissues> |
| QAProject: Fuel expenses view when no car or no expenses added yet | <https://jira.ithillel.com/projects/U4QA23022023/issues/U4QA23022023-132?filter=allopenissues> |
| **Figma design:** | |
| <https://www.figma.com/proto/Wy5XbgpfZpu9SasJO2QhnO/Hillel-Auto-1.2?node-id=52482-621&viewport=697%2C46%2C0.5580483675003052&scaling=scale-down> | |
| **Requirements:** | |
| <https://confluence.ithillel.com/display/HA/Hillel+auto> | |

* 1. **Application description**

“QAProject” web application is created for car owners. So that they can save, control and track the mileage and fuel decodes of their car / their cars, as well as receive useful information about maintenance.

1. **Test Items**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   1. **Test Items**

Within this test plan will be tested web application QAProject (version 1.2), namely:

* User interface (for desktop / for mobile)
* System functional
  1. **Features to be tested :**

- User registration and login

- Guest log in

- Integration of video on the main page of the site

- Garage without cars

- Adding, editing and deleting one car or more than one in the garage

- Ability to choose user’s cars

- Fuel expenses when no car or no expenses added yet

- Ability to add, remove and change mileage and fuel consumption for the car

- Ability to search, view and download the manual for the car

- Editing a personal profile, adding a photo

- Ability to change currency and recalculation of previously paid expenses

- Change mail, password, account deletion

- Site navigation

- Restore user access

- Multiple simultaneous sessions of the same user

* 1. **Features not to be tested**
* content of manuals for car
* relevance exchange rates

1. **Approach**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   1. **Testing methodology or approach**

QA team works with Agile methodology in software development.

Testing is started at the Testing stage, after the product development is completed, then if the Development stage is completed.

The testing for current project is planned with respect to risk-based approach. Each test case is prioritized as High, Medium or Low priority and then scheduled accordingly (highest first). The testing activities are performed among functionality of the web application. Basic metrics are kept for test effort, test cases executed, and incidents.

The testing will use manual testing. Generally, features are checked manually, using black-box method of testing.

* 1. **Testing Levels**

In this project, a ready-made system is provided for testing, then only system testing will be carried out. It is assumed that unit testing and integration testing were carried out on the developer's side.

Acceptance Testing will performs by the actual end users with the assistance of the other testing team.

* 1. **Testing Types to be used:**
  + Functional Testing
  + Non Function Testing:

User Interface Testing

Compatibility Testing

Usability Testing

Cross-browser Testing

* + Testing Related to Changes

Smoke Testing

Sanity Testing

Regression Testing

Retest

* 1. **Testing Types not to be used:**

Performance Testing

Security Testing

These Types of Testing will be carried out by other QA Team

Testing Types Priority

|  |  |  |
| --- | --- | --- |
| Version | Testing types | Priority |
| 1.2 | Functional testing | High |
| User interface testing | High |
| Compatibility Testing | Medium |
| Usability Testing | Medium |
| Cross-browser Testing | High |
| Regression | Functional testing | High |
| User interface testing | High |
| Compatibility Testing | Medium |
| Usability Testing | Medium |
| Cross-browser Testing | High |
| Regression testing | High |
| Retest | High |
| Sanity Testing | High |
| Smoke Testing | High |

* 1. **Testing techniques and tools**:
* Equivalence partitioning
* Boundary values analysis
* Decision table
* State transition testing
* Pairwise Testing
  1. **Test case development and management**
* Test Cases will be written using online resource <https://hillelitschool.testcaselab.com>
* Test case structure include: Summary, Priority (high, medium, low), Suite, Created by, Last Edit by, Type, Tags, Description (including Preconditions), Steps (in this project using only steps), Expected results, Links to User Stories, Attach screenshot with correct and incorrect application work
* Using Positive and Negative Test Cases
* In Test Case can be included not more than 20 steps
  1. **Defect development and management**

Defects found during the Testing will be categorized according to the bug-reporting tool

and the categories are:

|  |  |
| --- | --- |
| Severity | Impact |
| High | Bug should be fixed as soon as possible, as availability is critical for the project. Bug should be fixed |
| Medium | Bug, the presence of which is not critical, but it requires an urgent solution. Bug should be fixed. |
| Low | Bug, the presence of which is not critical, it does not require an urgent solution. Bug should be fixed |

* Bug report will be written using online resource [https://jira.ithillel.com](https://jira.ithillel.com/secure/Dashboard.jspa?selectPageId=12737)
* Bug report structure include: Priority (high, medium, low), Status, Labels, Environment, Epic Link, Sprint, Description (steps to reproduce, expected result, actual result), Attachments, Issue Links, Comments, Assignee (reporter and assignee can’t be the same person)
* Bug reports must be clearly and dispassionately
* Bug report should not contain conclusions, but it must contain an accurate description of the problem
* After create Bug report we need to make sure that: problem is replicate, we isolate it, externalize it and investigate it
* Bug Life Cycle

Tester Found a bug

Assign Bug on developer This Bug is Duplicate other Bug

Developer starts work with bug Found problem is not a Bug

Developer Fixed Bug Bug is Deferred

Bug is reproduce again Tester makes Retest

Tester Closes Bug

* 1. **Test data management**

Tester himself creates Test Data which need for testing QAuto web application:

* E-mails
* Regestrated users
* Instagram account
* Facebook account
* Telegram account
* LinkedIn account
  1. **Execution and reporting:**

On project team is going to use next reporting:

1. Planned and executed test cases

The formula for calculating the ratio of planned test cases to executed test cases would look like this:

*Executed test cases / Planned test cases x 100%*

For example, if 100 test cases were planned but only 80 were executed, the ratio of planned test cases to executed test cases would be:

80 / 100 x 100% = 80%

This way, you can assess the effectiveness of test planning and monitor the progress of test case implementation. If the ratio is less than 100%, it may indicate that the test planning was not precise enough or that the implementation of test cases took longer than planned. If the ratio is close to or greater than 100%, it may indicate good planning and efficient implementation of test cases.

2. Prioritizing bugs

Prioritizing bugs allows determining which bugs should be fixed first. This is especially important when numerous errors are found during testing and resources are limited.

The priority of a bug is determined based on its importance and impact on the application's functionality. More critical errors that can lead to serious problems should have a higher priority and be fixed first.

Prioritizing bugs helps the development and testing team focus on fixing the most important issues, reducing the risk of possible problems in the production environment.

We prioritize bugs by these categories - Low, Medium, High.

3. The ratio of the number of bugs fixed to the number of bugs found

The formula for calculating the ratio of bugs found to bugs fixed would be as follows:

*Bugs Fixed / Bugs Found x 100%*

For example, if 50 bugs were found and 40 were fixed, the ratio of bugs found to bugs fixed would be:

40 / 50 x 100% = 80%

4. The planned time versus actual time spent

The formula for calculating the ratio between planned time and actual time spent on the task will look like this:

*Planned time / Actual time spent on the task x 100%*

For example, if 10 hours were planned for completing a task, but it took 12 hours to complete, the ratio between the planned time and actual time spent on the task will be:

10 / 12 x 100% = 83.33%

This way, you can determine how accurately the time was planned for the task and how efficiently the time was spent during its completion. If the ratio is less than 100%, it may indicate that the task was planned somewhat optimistically and took longer than planned. If the ratio is higher than 100%, it may indicate that the task was completed more efficiently than planned.

Thus, the ratio of bugs found to bugs fixed can help the development and testing team to control the bug fixing process and assess the quality of testing. If the ratio is less than 100%, this may indicate an ineffective bug fixing process or insufficient bug detection during testing. If the ratio is close to 100% or 100%, this may indicate an effective bug fixing process and high bug detection during testing.

5. Coverage of all user-stories by test cases

The percentage of the total number of test cases that cover each user-story.

For example, if a project has 50 user-stories and each user-story has 5 test cases, the total number of test cases is 250 (50 \* 5). If 200 test cases cover all 50 user stories, the coverage metric will be 80% (200/250 \* 100%).

6. The remaining bugs that will be released according to user stories and priority

Identify all the bugs related to specific stories that need to be implemented in the current release.

Evaluate the priority of each bug based on its importance and impact on the application's functionality. More critical errors should have a higher priority and be fixed first.

Determine how many of these bugs have already been fixed and tested.

Calculate the remaining bugs that will be released by subtracting the number of fixed bugs from the total number of bugs for each story.

For each story, you can calculate the percentage of remaining bugs by dividing the number of remaining bugs by the total number of bugs and multiplying by 100%.

Identify the most critical bugs that need to be fixed first.

Thus, calculating the remaining bugs that will be released according to stories and priority allows the development and testing team to control the bug-fixing process and assess the application's readiness for release.

1. **Entry / Exit Criteria**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   1. **Entry Criteria**

* Formed set of requirements
* All system components are ready
* Unit testing was successful (All critical test cases were completed successfully)
* Integration testing was successful;
* Ready Suite cases and Test cases
* Ready hardware and software required for testing
* In the presence of all possessions and tools
  1. **Exit Criteria**
* All Test Cases run
* All Test Cases with priority “high” and “normal” are Pass
* There are no Critical / Blocker defects
* All Product requirement are implemented
* Test Items should be covered for 100% among the functional requirements
* Test item is passed when all the test cases are executed with status ‘Pass’
* Test item is failed when one or more test cases are executed with status ‘Fail’
* Carrying out all types of testing of the application for this document (test plan)
* All project documentation complies with requirements
  1. **Acceptance Criteria**

Acceptance criteria is achieved when all functions work correctly, open critical and major defects are fixed and verified by the test team. The web application may have Minors, Trivial defects are acceptable if Product Owner approves. Prior to final completion of Acceptance testing all open critical defects must be corrected and verified by the Customer test representative.

1. **Environments needs**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Test environment where the application is deployed QAProject app – <https://qauto2.forstudy.space/>

* 1. **Hardware and software recourses**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Device | 7 | Standard configuration | | |
| Tests PC | 1 | Windows 10 | Opera |  |
| Tests Laptop | 3 | Windows 7 | Google Chrome | v. 109.0.5414.120 |
|  |  | Mozila Firefox | v. 111 |
|  | Windows 10 | Google Chrome | v. 112.0.5815.121 |
| Mobile Phone | 3 | IOS (v. 16.2) | Safari | 6.5d / 2688 x 1242 |
|  | IOS (v. 16.3) | Safari | 6.1d /2532 x 1170 |
|  | Android (v. 12) | Google Chrome | 6.5d / 2400 x 1080 |

Testing on different versions is necessary:

Ensure compatibility: Each version of software can have differences in functionality, API support, and other parameters. Testing on different versions ensures that the application works on all these versions without crashes and errors.

Identify compatibility issues: Testing on different versions can also help identify compatibility issues that may arise, for example, when an old application is run on a new operating system or browser version.

Ensure quality: Testing on different versions helps to guarantee the quality of the application, as it must work on different configurations and versions.

Check new features: If a new version of the application contains new features or improvements, testing on different versions ensures that they work correctly and do not cause issues on other versions.

Customer support: Some customers may use old versions of operating systems, browsers, and devices, so testing on different versions ensures that the application will work for all customers, regardless of the versions they use.

Testing on different versions helps to ensure the quality of the application and verify its functionality on various configurations.

* 1. **Tools and resources required for testing:**
* <https://pairwise.teremokgames.com/>
* <https://jira.ithillel.com/projects/U4QA23022023>, <https://confluence.ithillel.com/display/HA/Hillel+auto>, <https://www.figma.com/proto/Wy5XbgpfZpu9SasJO2QhnO/Hillel-Auto-1.2?node-id=52482-621&viewport=697%2C46%2C0.5580483675003052&scaling=scale-down>
* Dashboard (link)

1. **Responsibilities**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   1. **Rollers, responsibilities and schedule of testing team members**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Responsible person** |  | **Estimation** | **Start date** | **End date** |
| Person 2 | **Generate Test Plan** | 7 days | 10.04.2023 | 16.04.2023 |
|  | **Generate Test Suits** | 1 day | 17.04.2023 | 17.04.2023 |
|  | **Create Test Cases** | 2 days | 18.04.2023 | 19.04.2023 |
|  | [Fuel expenses view when no car or no expenses added yet](https://jira.ithillel.com/browse/U4QA23022023-132) |  |  |  |
|  | [Add expenses, edit and table view](https://jira.ithillel.com/browse/U4QA23022023-133) |  |  |  |
|  | [Remove expenses](https://jira.ithillel.com/browse/U4QA23022023-134) |  |  |  |
|  | **Execute Test Cases / Create Bug reports** | 5 days | 20.04.2023 | 24.04.2023 |
|  | [Guest log in](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179314) |  |  |  |
|  | [User registration pop up](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179313) |  |  |  |
|  | [Registered user sign in](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179312) |  |  |  |
|  | **Testing Related to Changes** | 5 days | 25.04.2023 | 29.04.2023 |
|  | **Defect Dashboard** | 1 day | 30.04.2023 | 30.04.2023 |
|  | **Test Summary Report** | 5 days | 01.05.2023 | 05.05.2023 |
| Person 3 | **Generate Test Plan** | 7 days | 10.04.2023 | 16.04.2023 |
|  | **Generate Test Suits** | 1 day | 17.04.2023 | 17.04.2023 |
|  | **Create Test Cases** | 2 days | 18.04.2023 | 19.04.2023 |
|  | [Guest log in](https://jira.ithillel.com/browse/U4QA23022023-126) |  |  |  |
|  | [User registration pop up](https://jira.ithillel.com/browse/U4QA23022023-127) |  |  |  |
|  | [Registered user sign in](https://jira.ithillel.com/browse/U4QA23022023-127) |  |  |  |
|  | **Execute Test Cases / Create Bug reports** | 5 days | 20.04.2023 | 24.04.2023 |
|  | [Fuel expenses view when no car or no expenses added yet](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179308) |  |  |  |
|  | [Add expenses, edit and table view](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179307) |  |  |  |
|  | [Remove expenses](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179306) |  |  |  |
|  | **Testing Related to Changes** | 4 days | 25.04.2023 | 28.04.2023 |
|  | **Defect Dashboard** | 2 days | 29.04.2023 | 30.04.2023 |
|  | **Test Summary Report** | 5 days | 01.05.2023 | 05.05.2023 |
| Sadovats Daria | **Generate Test Plan** | 7 days | 10.04.2023 | 16.04.2023 |
|  | **Generate Test Suits** | 1 day | 17.04.2023 | 17.04.2023 |
|  | **Create Test Cases** | 2 days | 18.04.2023 | 19.04.2023 |
|  | [Main page](https://jira.ithillel.com/browse/U4QA23022023-125) |  |  |  |
|  | [Restore user access](https://jira.ithillel.com/browse/U4QA23022023-129) |  |  |  |
|  | [Garage view with/without cars and possibility to add new cars](https://jira.ithillel.com/browse/U4QA23022023-135) |  |  |  |
|  | **Execute Test Cases / Create Bug reports** | 5 days | 20.04.2023 | 24.04.2023 |
|  | [User profile header > redirections](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179310) |  |  |  |
|  | [User profile settings and edit profile](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179309) |  |  |  |
|  | **Testing Related to Changes** | 4 days | 25.04.2023 | 28.04.2023 |
|  | **Defect Dashboard** | 2 days | 29.04.2023 | 30.04.2023 |
|  | **Test Summary Report** | 5 days | 01.05.2023 | 05.05.2023 |
| Person 4 | **Generate Test Plan** | 4 days | 12.04.2023 | 16.04.2023 |
|  | **Generate Test Suits** | 1 day | 17.04.2023 | 17.04.2023 |
|  | **Create Test Cases** | 3 days | 18.04.2023 | 20.04.2023 |
|  | [User profile header > redirections](https://jira.ithillel.com/browse/U4QA23022023-130) |  |  |  |
|  | [User profile settings and edit profile](https://jira.ithillel.com/browse/U4QA23022023-131) |  |  |  |
|  | **Execute Test Cases / Create Bug reports** | 4 days | 21.04.2023 | 24.04.2023 |
|  | [Main page](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179315) |  |  |  |
|  | [Restore user access](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179311) |  |  |  |
|  | [Garage view with/without cars and possibility to add new cars](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179305) |  |  |  |
|  | **Testing Related to Changes** | 3 days | 25.04.2023 | 27.04.2023 |
|  | **Defect Dashboard** | 3 days | 28.04.2023 | 30.04.2023 |
|  | **Test Summary Report** | 5 days | 01.05.2023 | 05.05.2023 |
| Person 5 | **Generate Test Plan** | 7 days | 10.04.2023 | 16.04.2023 |
|  | **Generate Test Suits** | 1 day | 17.04.2023 | 17.04.2023 |
|  | **Create Test Cases** | 3 days | 18.04.2023 | 20.04.2023 |
|  | [User instructions](https://jira.ithillel.com/browse/U4QA23022023-138) |  |  |  |
|  | [User Settings](https://jira.ithillel.com/browse/U4QA23022023-139) |  |  |  |
|  | **Execute Test Cases / Create Bug reports** | 4 days | 21.04.2023 | 24.04.2023 |
|  | [Edit cars in garage view](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179304) |  |  |  |
|  | [Remove car(s) in garage view](https://hillelitschool.testcaselab.com/projects/QAM/test_cases?sort_dir=desc&sort_attr=created_at&test_case_id=1461783&category_id=179303) |  |  |  |
|  | **Testing Related to Changes** | 4 days | 25.04.2023 | 28.04.2023 |
|  | **Defect Dashboard** | 2 days | 29.04.2023 | 30.04.2023 |
|  | **Test Summary Report** | 5 days | 01.05.2023 | 05.05.2023 |

* 1. **Communication and collaboration channels for testing team**

For the communication QA Team is using:

* telegram channel for chatting <https://t.me/>...
* discord for meetings: https://discord.gg/...
* google docs for data sharing: <https://docs.google.com/...>
  1. **Escalation and issue resolution procedures**

What is the procedure for escalation?

Use these steps to escalate an issue up the chain of command:

Formally inform the decision-makers about the problem (using official communication channel of team)

Analyze the source of the problem and potential project impacts (search any options available)

Suggest options for solving the problem and highlight the advantages and disadvantages of each of them (inform your team about your decision)

|  |  |  |
| --- | --- | --- |
| Problem | Options for solving the problem | Highlight the advantages and disadvantages of each of them |
| blackout | Change work place  Use generators | Opportunity to work/It takes time  Opportunity to work/Not available |
| internet outage | Use mobile internet  Change work place | Opportunity to work/Low speed  Opportunity to work /It takes time |
| software problems | Change device (if it possible)  Solve a problem yourself  Solve a problem with the help of third parties | Opportunity to work/Lost last tests data  Opportunity to work/It takes time  Opportunity to work/It takes time and money |
| hardware problems | Change device (if it possible)  Solve a problem yourself  Solve a problem with the help of third parties | Opportunity to work/Lose last tests data  Opportunity to work/It takes time  Opportunity to work/It takes time and money |
| sick leave care | Redistribute tasks to others | Opportunity to work/It can be too much for a person |

1. **Risk and Contingencies**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Product Risk:
* Missing key functionality
* Critical defects
* Dependence on external systems and their failure
* Product is too difficult to use
* A lot of defects
* Change by the customer of requirements at a later stage
* Complex logic of product functionality
* Project Risk
* Wrong test design (priority, estimation, communication, strategy, tools, techniques, test types etc)
* Delay of developers with fixing defects
* Limited test environment
* Incorrect risk assessment (also included)

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Impact** | **Trigger** | **Mitigation** |
| Missing key functionality | release an application with a critical bug | incomplete requirements analysis | returns and overtime |
| Critical defects | inability to complete testing | error in code | return to development stage, passing test cases with the highest priority, overtime |
| Product is too difficult to use | user is unable (or very difficult) to use the product to its full potential | wrong requirements assessment | collaborate with the customer to resolve the issue |
| A lot of defects | failure deadlines for testing | not enough time for creating bugs | passing test cases with the highest priority and overtime |
| Delay of developers with fixing defects | failure deadlines for testing | too much bugs was found | passing test cases with the highest priority |
| Complex logic of product functionality | a lot of time for design test and execution test | complex business logic | allow more time for testing |
| Change by the customer of requirements at a later stage | failure deadlines | changing market requirements | extend deadlines  new analysis and design  new development  new testing |
| Wrong test design (priority, estimation, communication, strategy, tools, techniques, test types etc) | failure deadlines for testing | not involving testers at the requirement stage | passing test cases with the highest priority |
| Incorrect risk assessment (also included) | failure deadlines for testing or inability to complete testing | paid little attention to risk | work with risks |
| Limited test environment | release an application with a critical/not critical bugs for some hardware/software configuration | small amount of test software and test hardware | use simulators and emulators |

1. **Approvals**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

|  |  |
| --- | --- |
| Signature: |  |
| Name: |  |
| Role: |  |
| Date |  |

1. **Change history**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

To maintain a list of changes being made

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description of Change |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |