**Test plan**

**by Team #3**

**(Group M-ST1-259-21)**

| **Revision History** | | | |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Action** |
| 24.08.2021 | 1.0 | Ilya Dyuzhev | Creature |
| 25.08.2021 | 1.0 | Anton Rapinchuk | Review |
| 29.08.2021 | 2.0 | Ilya Dyuzhev | Making changes |
| 1.09.2021 | 2.0 | Anton Rapinchuk | Review |
| 7.09.2021 | 2.1 | Ilya Dyuzhev | Making changes |
| 8.09.2021 | 2.1 | Anton Rapinchuk | Review |
| 22.09.2021 | 3.0 | Ilya Dyuzhev | A complete revision of the strategy |
| 23.09.2021 | 3.1 | Anton Rapinchuk | Review |
| 27.09.2021 | 3.2 | Ilya Dyuzhev | Making changes |
| 28.09.2021 | 3.2 | Anton Rapinchuk | Approved |

**Contents:**

1. **Introduction**

**1.1 Purpose**

**1.2 Background**

1. **Scope of work**
2. **Test team**
3. **Test documentation**
4. **Test strategy**

**5.1 Test methods**

**5.2 Test types**

**5.3 Test levels**

**5.4 Test design techniques**

**5.5 Bug and documentation tracking**

1. **Resources**

**6.1 Test environment**

**6.2 Test tools**

1. **Test schedule**
2. **Risks**
3. **Quality criteria**

**9.1 Testing can be completed when**

**9.2 Test coverage**

1. **Introduction**

**1.1 Purpose**

The purpose of this test plan is to describe the process of testing the TravelWorse application.

**1.2 Background**

The TravelWorse application is a trip planner. The application allows you to create travel plans, where cities, places, attractions, a place to stay and accommodation are added, in general, everything that will allow you to plan your trip perfectly. The functionality of the application allows you to share the created trips with friends, so that everyone can choose the right option together.

This document describes all information about approach and methodologies, resources and the testing team necessary for reach mark goals.

1. **Scope of Work**

The TravelWorse application will be tested. This trip planner allows users to create trip plans, compare them with each other by criteria and choose the best option. We need to test all the basic functionality. Such as:

- creating a trip plan

- adding sections to a trip

- editing

- assigning attributes

- deleting

- commenting

- filling in application fields

- the ability to share a created trip for viewing and making changes.

The project requirements will be presented in the form of a "user story"

Features not to be tested:

* login form
* registration form
* calendar for selecting travel dates
* counting button

1. **Test Team**

| **Team composition** | **Roles** | **Responsibilities** |
| --- | --- | --- |
| Anton Rapinchuk | Test Manager | 1. Coordination of the team's work on the project:  - collection and formalization of project conditions  - control a work execution plan  - setting targetsand monitoring deadlines  2. Planning of activities, risks, and time  3. Team motivation and conflict resolution  4. Test leads cosulting |
| Ilya Dyuzhev | Team Lead | 1. Coordinator of the test team:  - Distribution and delegation of tasks  - Monitoring the project status  - Distribution of the sphere of responsibility  - Planning priorities of tasks and subtasks  - Motivation of the team for success  2. Communication with the customer, management and all team members  3. Choosing technologies for the project |
| Ilona Victorchyk | Testing Engineer | Product testing, including the following steps:  - Study and clarification requirements from the customer  - Writing and subsequent revision of test scripts  - Working with test documentation  - Testing the software functionality  - Search for bugs and entering the found errors into the documentation or tracking system |
| Ekaterina Drabskaya | Testing Engineer |
| Irina Milushkina | Testing Engineer |

1. **Test documentation**

| **Title** | **Responsible** | **Storage/Delivery method** |
| --- | --- | --- |
| Test Result Reports | Ilya Dyuzhev | Google docs, Gmail |
| Checklists | All team | Google docs, Gmail |
| Test Cases | All team | Google docs, TestRail |
| Bug Reports | All team | Google docs, Jira |
| Test Plan | Ilya Dyuzhev | Google docs, Gmail |
| Requirements | Ilya Dyuzhev | Google docs, Jira |

1. **Test Strategy**
2. **Test methods**

Testing is the process of finding inconsistencies between a program and its functional specification/ requirements. The goal is to make sure that all the functions of the application are working correctly.

Manual functional testing is considered as the main method of testing applications.

Also, we will use such a testing method as a Black box.

Only, with the condition that we will use the browser console to view the content of the page displayed by the browser, as well as existing errors.

1. **Test types**

Functional testing will be used as the main type of testing in order to check the application for what it does and whether it meets the customer's requirements.

The following types of non-functional testing will be applied:

* GUI Testing is performed by interacting with the software under test via the graphical user interface. Verifies that user interface meets design guidelines, ensures UI controls, input fields work as expected
* Compatibility Testing - determines whether the product operates correctly in configurations with various operating systems, browsers. Several combinations of browsers and operating systems will be tested in the project.
* Ad hoc the type of testing that is performed without preparation for testing the product, without determining the expected results, designing test scenarios. This is an informal, improvisational testing. It does not require any documentation, planning, or processes that should be followed when performing testing.
* Exploratory testing is a simultaneous study of a software product, the design of tests and their execution. This is an informal method of designing tests, in which the tester actively controls the design of tests while these tests are being performed, and uses the information obtained during testing to design new tests.

If each subsequent test that the tester performs is selected based on the results of the previous test, this means that we are using exploratory testing

Types of testing based on the principle of working with the application:

* Positive testing is aimed at making sure that the main functionality works. That, all scenarios of using our system are feasible and lead to the expected result, and not to errors.
* Negative testing is the process of checking for incorrect behavior. During such testing, we can find out that the system will cope with unforeseen situations.

Mobile testing. We will use mobile testing in order to analyze how the application works in a web environment and on mobile devices. Functional testing and specification testing will be carried out.

1. **Functional testing levels**

* The Smoke test is carried out to quickly assess the readiness of the product for further deeper and more thorough testing. This is testing only the most important functionality, without the successful completion of which it makes no sense to move to the next level of testing
* Critical Path Test - the main type of test, during which the relevant elements and functions of the application are checked for proper operation with their standard use. Critical Path Testing is aimed at exploring the functionality used by typical users in typical daily activities and will be performed after Smoke Test is passed
* Extended Test’s goal to find bugs related to the non-typical but still possible and likely usage scenarios (e.g. entering the incorrect data into the fields, boundary testing and so on). Extended Test will be performed according to test cases

1. **Testing levels according to the level of detail of the system components**

* Component testing checks the functionality and looks for defects in the parts of the application that are available and can be tested separately (program modules, objects, classes, functions, etc.)
* Integration testing it is intended for checking the communication between components, as well as interaction with various parts of the system (operating system, hardware, or communication between different systems)

1. **Test design techniques**

During the testing of the application, there is a high probability of using domain testing. The following types may be applicable:

* Equivalence Class Testing.

The equivalent partitioning method allows you to minimize the number of tests without creating a script for each possible value, but choosing only one value from the whole class

* Boundary Value Testing.

The boundary value technique is based on the assumption that most errors can occur at the boundaries of equivalent classes.

* Pairwise testing.

This is a technique for generating test data sets from a complete set of input data in the system, which allows you to significantly reduce the number of test cases.

1. **Bugs**

The tools described in the "Test Tools" section will be used for error reporting and documentation tracking. Indicators and error statistics will be included in the test results reports.

Bug severity definition:

Blocked - Blocks development and/or testing work, production could not run.

Critical - Crashes, loss of data, severe memory leak.

Major - Major loss of function. The existence of a defect brings tangible inconveniences

Medium - Minor loss of function, or other problem where easy workaround is present.

Minor - A defect that does not affect the operation of the application in most cases

Cosmetic problems, such as misspelled words or uneven text. Somewhere, a very minor typo

1. **Resources**

**1. Test environment**

| **Team Member** | **Device type** | **Operating system** | **Browser** |
| --- | --- | --- | --- |
| Ilona Victorchyk | Smartphone | Android 11 | Chrome 94.0.4606.61 |
| Ilya Dyuzhev | Laptop | Windows 10 Pro x64  Version: 20H2  Assembling: 19042.631 | Opera GX 78.0.4093.214 |
|
| Ekaterina Drabskaya | Laptop | Windows 10 Home x64  Version: 20H2  Assembling: 19042.1165 | Chrome 94.0.4606.61 |
|
| Irina Milushkina | Laptop | Windows 10 Enterprise x64  Version: 1909  Assembling: 18363.1500 | Mozilla 92.0.1 |
|

**2. Test tools**

| **Tools** | **Comment** |
| --- | --- |
| [TestRail](http://178.124.206.46:8000/index.php?/dashboard) | Сreating a project, preparing test cases, performing test runs and generating detailed testing reports |
| Microsoft Office | Creating text documentation |
| Google docs | Creating text documentation |
| [Jira](https://jr.it-academy.by/projects/MST25921/summary) | Maintaining bug reports on the project, the main repository of requirements and user stories |

1. **Test schedule**

| **#** | **Activity** | **Begin date** | **End date** | **Assignment** |
| --- | --- | --- | --- | --- |
| 1 | Test strategy creation | 24.08.2021 | 3.09.2021 | Team Lead |
| 2 | Requirements Analysis | 4.09.2021 | 10.09.2021 | All team |
| 3 | Testing requirements | 11.09.2021 | 15.09.2021 | All team |
| 4 | Сreation of check-lists | 16.09.2021 | 18.09.2021 | All team |
| 5 | Writing test cases | 19.09.2021 | 22.09.2021 | All team |
| 6 | Testing and writing bug reports | 23.09.2021 | 27.09.2021 | All team |
| 7 | Тhe creation of a Final TRR, preparation of presentations | 28.09.2021 | 30.09.2021 | All team |

1. **Risks**

| **Risk** | **Probability** | **Impact** | **Actions** |
| --- | --- | --- | --- |
| Problems when working with requirements | High | High | There may be a misunderstanding with the customer in certain points of the requirements and in general. Ambiguity in the requirements will complicate and slow down the process of forming test documentation |
| Problems in the test equipment. Breakdown of computer hardware or failures on the server side of the project. This can lead to a significant slowdown in project activities | Low | Medium | Availability of spare equipment, which will be provided if necessary |
| The testing team consists of inexperienced workers, this can lead to the omission of a different kinds of bugs | High | High | Help of experienced testers in critical situations |
| Different types of leave (sickness or vacation) of any Member of the testing team, can lead to additional workload on other participants of the project | High | Medium | Providing the project with an additional temporary employee to work in the absence of a team member |
| Force majeure circumstances that may lead to the de-energization of the workplace, which will lead to the suspension of work on the project | Low | Medium | The presence of another room in the absence of electricity at the permanent workplace |

1. **Quality criteria**
2. Requirements

* The final coverage of the requirements by test cases should be at least 90%
* The requirements must be:

- Correct

- Unambiguous

- Completed

- Consistent

- Verifiable

- Modifiable

- Traceable

- Ranked

1. Test case creation

* Create at least one test cases for one unit of requirements (where the unit of requirements is one point of the acceptance criterion)
* Each test case must correspond to one item of the checklist
* 100% of the requirements are taken into account when writing test cases

1. Test case execution

* The transition to testing of the Critical path level of tests is allowed only if 100% of the Smoke level test cases are successfully completed
* The main criterion for starting testing is the build output
* Successful completion of all smoke level tests will be the result - passed 100%
* All tests with the Critical Path level must have a passed priority of at least 90%
* There are no test cases with the status "untested" in the test run

1. Bugs

* Test cases with the Failed status are no more than 10%
* The percentage of test cases with the Blocked status should not exceed 5%
* The total number of defects should not exceed 10% of the completed test cases