Test plan

SoloLearn

Product: SoloLearn/mobil version

Author: Hennadii Prokopenko

Approvals: Kiril Katasonov

Version 1.0

Create: 09/02/2023

Status: Draft

**1. INTRODUCTION**

**1.1 Project Overview**

**SoloLearn** is educational resource, which will help you learn how to program yourself, partially free. Advanced features are available for payment. There are more then 20 programming languages in the project, including Python, JavaScript, HTML, SQL, C++. Each course is designed by experienced programmers and adapted to different levels of participants.

**1.2 Initial data**

* Programming language courses;
* Community of people are working, interested in programming;
* Discussion chat;
* Personal profile (account registration, about the course, achievements, tasks, practice, activity);

**1.3 Testing purpose**

The purpose of testing is to test all functionality according to the initial data.

**1.4 Entrance and exit criteria**

**1.4.1 Entrance criteria**

* According to homework №14.
* Create the test documentations (test cases on [Qase](https://app.qase.io/projects)) according to the selected testing approaches.

**1.4.2 Exit criteria**

* According to deadlines.
* Tests should cover 90% of the basic functionality according to the initial data.
* All tests passed. Created bug reports for all found bugs.
* If no bugs are found, then testing finishes upon the test execution complete.
* According to management decision.

**2. TEST STRATEGY**

**2.1 The target audience**

The target audience is users of different ages with an educational purpose.

The business functionality of the app is education.

Distribution channels are Google Play, Apply Store.

**2.1 Testing strategy steps**

The test strategy will be as follows:

* The first stage consists in drawing up a general test plan and test cases, in the context of the selected testing approaches. The result of the first stage is the coordination of the test plan and test cases with the manager.
* The second stage will consist of functional testing according to the initial data. Functionality will be tested using the critical path testing. The result of the second stage will be the passage of the critical path test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.
* The third stage will consist of usability testing. Usability is the verification of a software product for compliance with the requirements in terms of ease of use of the application. The result of the third stage will be the passage of the usability test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.
* The fourth stage will consist of localization testing. The result of the fourth stage will be the passage of the localization test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.
* The fifth stage will consist of performance testing. The result of the fifth stage will be the passage of the performance test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.
* The sixth stage will consist of security testing. The client provides personal data, credit card number, CVV number during the transaction. It is important to make sure that everything is transmitted after encryption and that the data transmission channel is secure. The result of the sixth stage will be the passage of the performance test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.
* The seventh stage will consist of compatibility testing. The result of the seventh stage will be the passage of the performance test and the creation of bug reports, If the bugs are found. The results must be agreed with the manager.

**2.2 Testing workflow**

**2.2.1 Functional testing**

Functional testing will be performed to check the functions of program. Function testing will be performed using Critical Path Test.

The functional test cycle helps ensure that the application works and performs its essential functions.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.2.2** **Usability testing**

This cycle of tests is designed to evaluate the ease of use of the program for the user.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.2.3 Localization testing**

This cycle of tests checks the adaptation of a mobile application to a specific target audience in accordance with its cultural characteristics.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.2.4 Performance testing**

This cycle of tests is designed to check the speed of procedures, stability, reliability and scalability of the system as a whole.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.2.5 Security testing**

This cycle of tests is armed at finding flaws and gaps in terms of application security.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.2.6 Compatibility testing**

This cycle of tests checks whether the mobile application works correctly on different devices.

The test cases are created in the Qase program at the link: <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

**2.3 List of jobs and deadlines**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Job types** | **Author** | **Due date** |
| 1 | Test plan | Hennadii Prokopenko | 09.02.2023 |
| 2 | Create Critical Path Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 3 | Create Usability Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 4 | Create Localization Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 5 | Create Performance Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 6 | Create Security Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 7 | Create Compatibility Test Cases | Hennadii Prokopenko | 10.02.2023 |
| 8 | Create Bug-Reports in Jira | Hennadii Prokopenko | 11.02.2023 |

**2.4 Software product development methodology**

Scrum was chosen to implement the project.

**2.5 Daily Standup**

Daily Standup lasts 25-40 minutes on the daily.

**2.6 Priority**

Task priorities according to scrum board in Jira:

**2.7 Test execution process**

Create the test cases in Qase

Create Bug-Reports in Jira (if a bug will be found)

Complete the

test execution of

all the test cases

Mark Status as Pass/Fail in Qase

Execute each of the test step in test case

**2.8 Test environment**

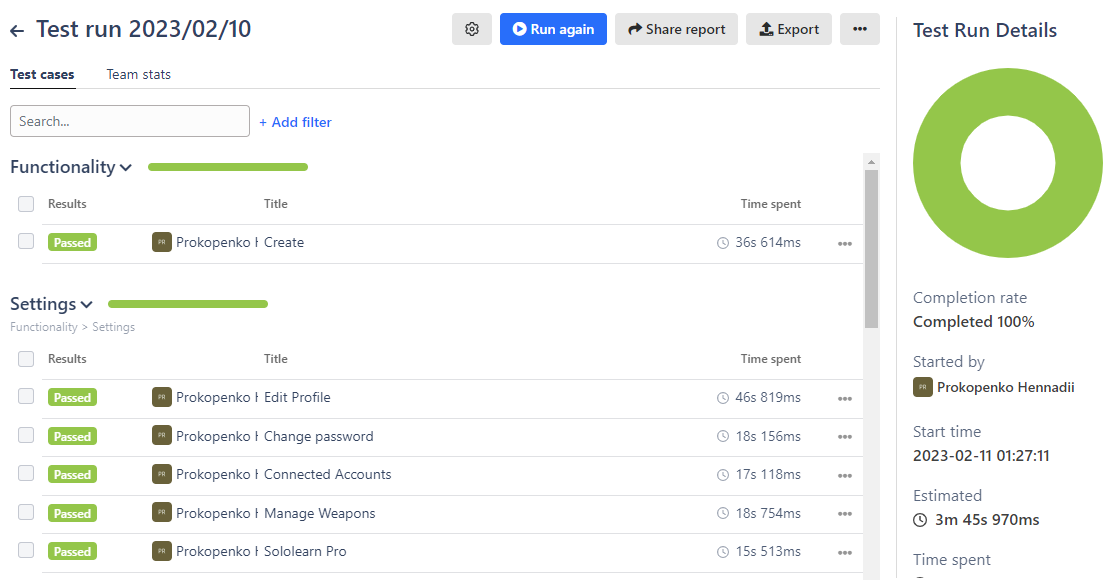
Xiaomi Mi 10T Pro

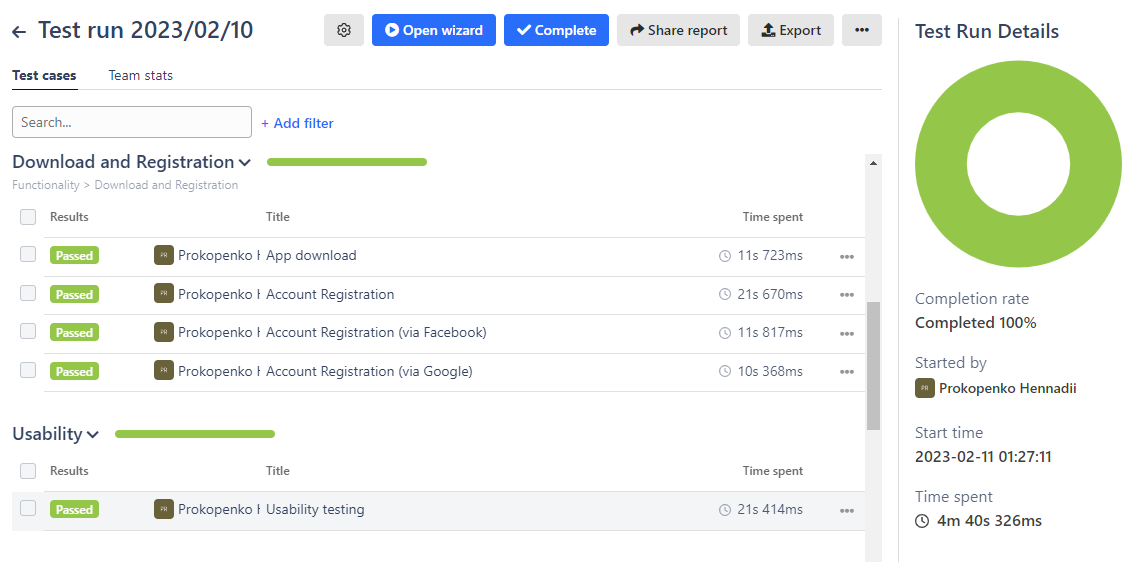
OS Android version 12 SKQ1.211006.001

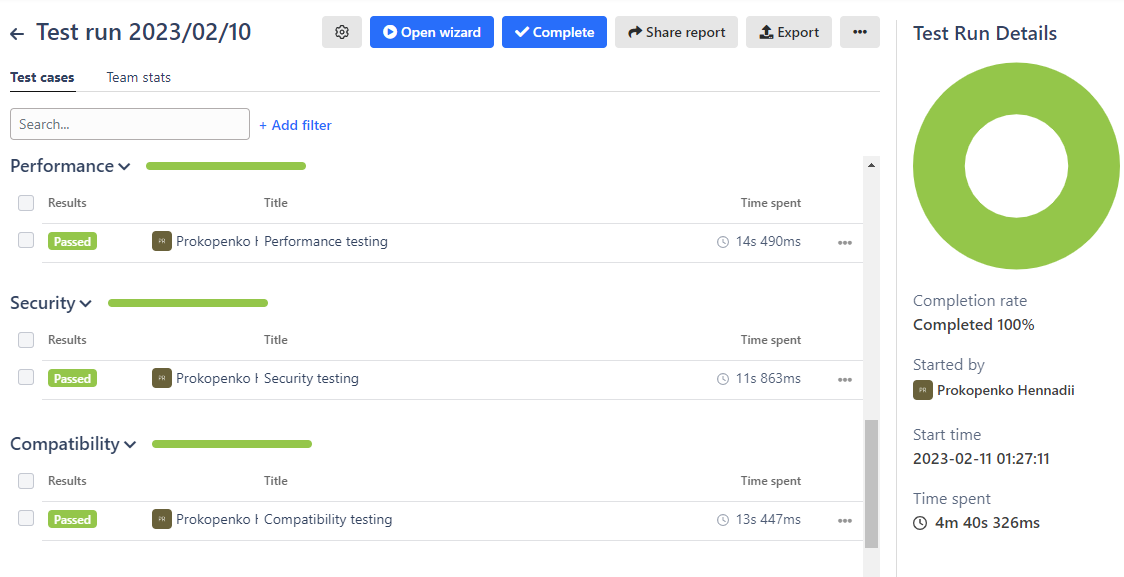
Apple iPhone 13 mini 256Gb

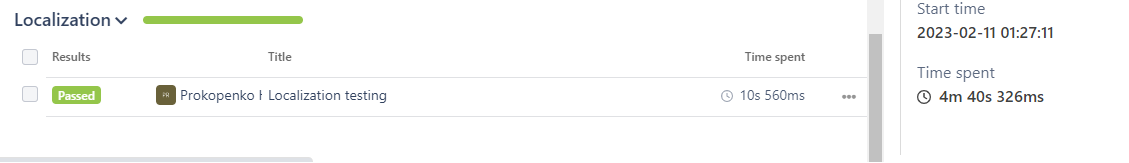
OS IOS version 15.5

**Results of testing:** <https://app.qase.io/public/report/579c2231ce63467cfd70b011e23b2887d33a4d28>

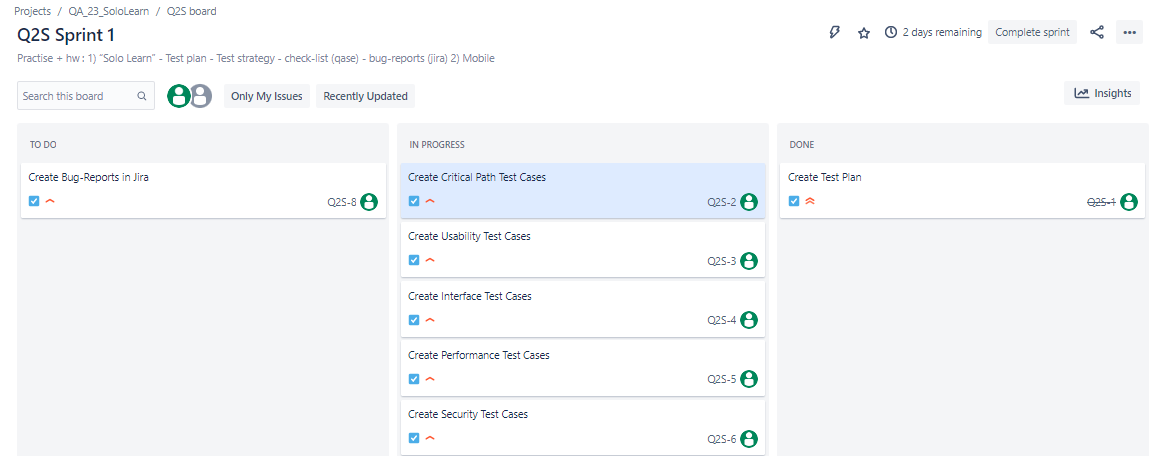
****

****



****

**Project in Jira:** [**https://prokopenkoga.atlassian.net/jira/software/c/projects/Q2S/boards/10?selectedIssue=Q2S-8&atlOrigin=eyJpIjoiZTBlOWE2ZTM4YTMxNDEyNjg1NWMxZmIzZjRkN2U3MzYiLCJwIjoiaiJ9**](https://prokopenkoga.atlassian.net/jira/software/c/projects/Q2S/boards/10?selectedIssue=Q2S-8&atlOrigin=eyJpIjoiZTBlOWE2ZTM4YTMxNDEyNjg1NWMxZmIzZjRkN2U3MzYiLCJwIjoiaiJ9)

****

**3. Test design**

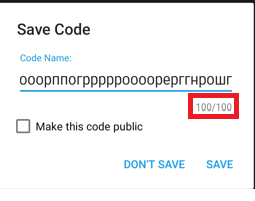
**3.1 Equivalence Partitioning**

Create your account:

|  |  |  |  |
| --- | --- | --- | --- |
| Valid | Comments | Not valid | Comments |
|  | The account was created because it has not yet been registered with this email |  | The account was created earlier via Google with this email, so when registering via email, this option is not valid |
|  | The account was created because it has not yet been registered with this email |  | Email without @ |
|  |  |  | Missing dot before com |

**3.2 Boundary Values**

Save Code:



Code Name must be from 1 to 100 characters.

Testing:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Not valid | Valid | Not valid |
| Number of characters in code name | 0 | 1 100 | 101 |

**3.3 Decision Table**

Create your account:

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Option** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **E-mail** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** |
| **Name** | **1** | **1** | **0** | **0** | **1** | **1** | **0** | **0** |
| **Password** | **1** | **0** | **1** | **0** | **1** | 0 | **1** | **0** |
| **Result** | **Pass** | **Failed** | **Failed** | **Failed** | **Failed** | **Failed** | **Failed** | **Failed** |

**1 -True**

**0 - False**

**3.4 Pairwise Testing**

Filters can be tested with pairwise testing.

|  |
| --- |
|  |

**3.5 State-transition Diagram**

Learn:

Finish (Your certificate is close)

Practice Lesson 2

Correct

Correct

Correct

Lesson 2 (theory)

Practice Lesson 1

Lesson 1 (theory)

Start

Incorrect

Incorrect

**3.6 Use Case Diagram**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Roles** | **Register** | **Edit profile** | **Change Password** | **Connected Account** | **Delete Account** | **Add New Question** | **Add Comments** | **Delete Comments** | **Help Service** | **Choose a course** | **Course Learn** | **Provide Feedback** | **Feedback** | **Ban user** |
| **User** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Administrator of SoloLearn** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Software Developer** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**3.7 Error Guessing**

Typical conditions to try to pass guessing errors:

3.7.1 Put video files, music, text files on an avatar;

3.7.2 Enter an empty value in the field;

3.7.3. Enter a space at the beginning, in the middle, at the end of the field;

**3.8 Cause/Effect**

Connected Accounts

|  |
| --- |
|  |

Cause – Add

Effect – Connected Facebook, Linkedln, GitHub, StackOverflow