**Test Plan** **Lexiconer**

**for**

CATesters team

|  |  |
| --- | --- |
| **Abbreviations and Acronyms** | |
| App | Application |
|  |  |
|  |  |

**Contents**

|  |  |
| --- | --- |
| **1 Introduction** | **4** |
| **2 Scope of Work** | **4** |
| **2.1 Components and Functions to be Tested** | **4** |
| **2.2 Components and Functions Not to be Tested** | **5** |
| **3 Quality and Acceptance Criteria** | **6** |
| **4 Critical Success Factors** | **6** |
| **5 Risk Assessment** | **6** |
| **6 Resources** | **7** |
| **6.1 Key Project Resources** | **7** |
| **6.2 Test Team** | **7** |
| **6.3 Test Environments** | **7** |
| **6.4 Test Hardware** | **8** |
| **6.5 Test Tools** | **8** |
| **7 Test Documentation and Deliverables** | **9** |
| **8 Test Strategy** | **9** |
| **8.1 Entry Criteria** | **9** |
| **8.2 Test Methods** | **9** |
| **8.3 Test Types** | **10** |
| **8.4 Test Levels** | **10** |
| **8.5 Bug and Documentation Tracking** | **10** |
| **8.5.1 Bug Severity Definitions** | **10** |
| **8.6 Labels** | **10** |
| **9 Testing Schedule** | **11** |

**1. Introduction**

There is desktop application called Lexiconer for learning foreign languages. It was created in the early 2010s. This application has several tasks:

- A learning task for users.

- Earnings for developers.

The target audience of the Lexiconer app is people who want to improve their foreign language skills.

This document contains all the information about the approaches and methodologies, resources and testing team needed to achieve the testing objectives.

**2. Scope of work**

**2.1 Components and functions to be tested**

|  |  |  |  |
| --- | --- | --- | --- |
| # | Application/ component name | Function name | Reference |
| 1 | Lexiconer / File | New dictionary | Allows you to create a new dictionary |
| 2 | Lexiconer / File | Open the dictionary | Allows you to open a loaded dictionary |
| 3 | Lexiconer / File | Download dictionary | Allows you to download dictionaries directly from the program |
| 4 | Lexiconer / View | Toolbar (function) | Displays the buttons of the main interface as an additional panel with buttons (New Dictionary, Open Dictionary, Dictionary Editor, "This word is learned", "Reset word statuses", Progress, Help, About) |
| 5 | Lexiconer / View | Search bar (function) | Allows you to find a word through the search bar in the app quickly |
| 6 | Lexiconer / View | Status bar (function) | Displays (permanently) the status of the word study |
| 7 | Lexiconer / View | On top of all the windows (function) | Allows an application window be displayed on top of other user windows |
| 8 | Lexiconer / Tools | Dictionary Editor | - Full work with the dictionary: search, add, edit, delete words.  - Ability to choose the translation when words are adding  - Ability to add word combinations  - Build dictionaries on the basis of your texts.  - Dictionary comparison  - In the word selection menu, you can override words by clicking on the word itself.  - Ability to mark a word as learned/unlearned by highlighting one or more words and clicking the "Learn/Unlearn" button (check the box marked with already learned words).  - Ability to change the order of words in the dictionary using the "Shuffle" button.  - Ability to delete words using the "Delete" button. |
| 9 | Lexiconer / Tools | This word is learned (button) | Allows you to mark a word as learned (automatically marks it in the dictionary) |
| 10 | Lexiconer / Tools | Reset word statuses (button) | Allows you to reset the status of words (all in the dictionary). All words in the dictionary become unstudied. |
| 11 | Lexiconer / Tools | Progress (button) | Allows you to view the status of the study |
| 12 | Lexiconer / Settings | Learning Model | - In the training model you can choose existing models or create your own training model.  - A variety of exercises: mosaic, translation choice, card, word choice, spelling will help to firmly fix the studied word in memory. |
| 13 | Lexiconer / Settings | Language (localization testing) | The application supports several language interfaces (Russian and English) |
| 14 | Lexiconer / Settings | Fonts | Ability to customize the application font |
| 15 | Lexiconer / Settings | Design | Changed program design (3 variants to choose from) |
| 16 | Lexiconer / Settings | Transcription (function) | Transcription helps you pronounce the word that you are learning correctly. |
| 17 | Lexiconer / Settings | Voicing (function) | The learning process is accompanied by the voicing of words, which contributes to the development of listening skills. |

**2.2 Components and functions not to be tested**

|  |  |  |  |
| --- | --- | --- | --- |
| # | Application/ component name | Function name | Reference/Comment |
| 1 | Lexiconer / Help | Reference | Program information |
| 2 | Lexiconer / Help | Support the project | Allows you to make a donation |
| 3 | Lexiconer / Help | About the program | Program information |
| 4 | Performance testing | - | The process of opening the application is accelerated |
| 5 | Ability to learn other languages, not only English | - | Ability to learn 35 languages |

**3. Quality and acceptance criteria**

- The product must perform the basic tasks;

- The product must perform in accordance with the requirements and functional specification;

- The number of errors in the product must meet the acceptance criteria defined in the contract/requirements or, unless otherwise specified, the product must have no errors of critical and serious severity for release to production.

- The product must be approved by the testing team after testing the product on at least two operating systems (Windows 7 and Windows 10).

**4. Critical success factors**

- Meet a schedule and complete development and testing of all functionality in term.

- Support multi-language interface.

- Application shouldn’t have bugs with severity Critical and Major at the time of Final Release.

- Functional requirements does not have last minute changes.

**5. Risk management**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability** | **Impact** | **Actions** |
| 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. | L | M | 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. | VH | VH | Experienced people: Team Lead or other tester will help testing team resolve any problems in urgent cases. |
| 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. | H | M | Team Lead can provide to project for an additional temporary person to work during 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 | L | M | The presence of another room in the absence of electricity at the permanent workplace |

*Legend: VH – Very High; H – High; M – Medium; L – Low*

**6. Resources**

**6.1 Key project resources**

|  |  |  |
| --- | --- | --- |
| # | Project Role | Name, e-mail, location |
| 1 | Test Leader | Nikita Kotov, [nikita.kotov.1404@gmail.com](mailto:nikita.kotov.1404@gmail.com), Minsk, Belarus |
| 2 | Tester | Nikita Kotovich, nikita.kotovich28@gmail.com, Minsk, Belarus |
| 3 | Tester | Artsiom Hur, logistik.profi.1021@gmail.com, Minsk, Belarus |
| 4 | Tester | Alesya Nichiporchik, nichiporchik.alesy@gmail.com, Minsk, Belarus |
| 5 | Tester | Pavel Bialiauski, bialiauski.paul@gmail.com, Minsk, Belarus |

**6.2 Test team**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Project Role | Name | Location | Responsibilities |
| 1 | Test Leader | Nikita Kotov | Minsk, Belarus | Preparation of test cases, test runs, bug reports. Writing a TRR, control of testing processes |
| 2 | Tester | Nikita Kotovich | Minsk, Belarus | Preparation of test cases, test runs, bug reports. |
| 3 | Tester | Artsiom Hur | Minsk, Belarus | Preparation of test cases, test runs, bug reports. |
| 4 | Tester | Alesya Nichiporchik | Minsk, Belarus | Preparation of test cases, test runs, bug reports. |
| 5 | Tester | Pavel Bialiauski | Minsk, Belarus | Preparation of test cases, test runs, bug reports. |

**6.3** **Test Environments**

The app is desktop and designed for PCs, so we do not need to use mobile devices and browsers to test. PCs with Windows 7 and Windows 10 (at least 2 OS) will be required for testing

**6.4 Test hardware**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Role | Name | Device Features | OS Features |
| 1 | Tester | Nikita Kotov | Processor: AMD Ryzen 7 2700X Eight-Core Processor 3.70 GHz  MM: 32.0 GB  Type of system: 64-bit OS, processor x64 | Edition: Windows 10 for educational institutions |
| 2 | Tester | Nikita Kotovich | Processor: Intel ® Core (™) i5-7200U CPU @ 2.50 GHz 2.70 GHz  MM: 8.00 GB (Available 7.85 GB)  Type of system: 64-bit OS, processor x64 | Edition: Windows 10 Pro  Version: 20H2  OS build: 19042.1526 |
| 3 | Tester | Artsiom Hur | Processor: AMD Athlon (™) II x4 640 processor 3.00 GHz  MM: 2.00 GB  Type of system: 32-bit OS | Edition: Windows 7 Max |
| 4 | Tester | Alesya Nichiporchik | Processor: Intel ® Celeron ® CPU N3050 @ 1.60 GHz 1.60 GHz  MM: 4.00 GB  Type of system: 64-bit OS, processor x64 | Edition: Windows 10 Pro  Version: 1909  OS build: 18363.657 |
| 5 | Tester |  | Processor: AMD Phenom (™) II N930 Quad-Core Processor 2.00 GHz  MM: 4.00 GB (3,74 GB available)  Type of system: 64-bit OS | Edition: Windows 7 Ultimate |

**6.5 Test tools**

|  |  |  |
| --- | --- | --- |
| # | Tool | Comment |
| 1 | Google tables | Creating checklists |
| 2 | Jitsi | Team communication |
| 3 | Telegram | Team communication |
| 4 | Jira | Bug reports |
| 5 | Qase, Jira | Creating test cases |
| 5 | MS Offise Word | Documentation writing |

**7. Test documentation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Title | Responsible person(s) | Frequency (delivery time) | Method of delivery |
| 1 | Lexiconer TestPlan | Nikita Kotov, Artsiom Hur | Once before the testing start | Mailing. |
| 2 | Lexiconer TestCases | All team | Before the testing start | Writing in a special tule(s). |
| 3 | Bug reports | All team | Upon finding a bug | Writing in Jira. |
| 4 | Test Result Reports | Artsiom Hur | Weekly | Writing in Word. Preparing a presentation on the results of the work. |

**8. Test strategy**

The Lexiconer applications will be tested using a “black box” approach, which is based on the requirements and functional specification without knowledge of the internal structure or program source code.

**8.1 Entry criteria**

The Testing Team may suspend partial or full-testing activities on a given build if any of the following occurs:

- It is impossible to install the new build following the readme successfully.

- There is a fault with a feature that prevents testing the feature.

- Application does not contain the specified change(s).

- New claimed functionality doesn’t work or works improperly.

- A severe problem has occurred that does not allow testing to continue.

- Development has not corrected the problem(s) that previously suspended testing.

- A new version of the software is available to test.

**8.2 Test methods**

Testing is the process of attempting to find discrepancies between the program and its functional specification/ requirements. The goal is to make sure that all functions of “Lexiconer” applications work correctly.

Manual functional testing is considered the primary method of application testing. Automated testing will not be used because it requires a large financial outlay and special skills of the team.

**8.3 Test types**

**Functional testing** is a type of testing that verifies that each function of a software application works in accordance with the requirements specification.

**Compatibility testing** - compatibility testing determines whether the product works correctly in configurations with different operating systems, browsers.

**Localization testing** - checks whether the application is translated and works correctly in the specified languages (English and Russian).

**Configuration testing** - testing to verify that the software works in different system configurations

**New Feature Testing** - checks that the new functionality works correctly.

**8.4 Test levels**

**Smoke testing**is conducted to quickly assess the readiness of the product for further in-depth and thorough testing. It involves testing the basic functions of the Lexiconer application on the most frequently used and therefore most important configuration.

If Smoke Test fails, the testing team sends a notification and suspends testing until a corrected version of the product is available.

**Critical Path Test** will be performed after Smoke Test is passed. The goal of the Critical Path Test is to find bugs that could affect the major functionality of the application that is most important for the product users. Critical Path Test will be performed manually.

**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.

**8.5 Bug and documentation tracking**

Tools described in the section Test Tools will be used for bug reporting and documentation tracking. The bug metrics and statistics will be included in the test results reports.

**8.5.1 Bug severity definition**

Critical – Application, component or module crash or are not accessible

Major – Data corruption/loss, a problem in major functionality, no workaround is known.

Medium – A problem with workaround, secondary features do not work properly

Minor – Cosmetic flaw

**8.6 Labels**

The **"CATestersTEAM"** label will be used in Jira to mark bugs on the Lexiconer project. This label will be used in the future to obtain statistics on the Lexiconer project bugs.

We will also use the labels **"Windows7"** and **"Windows10"** to indicate the operating system in which the bug was detected

**9. Testing schedule**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Activity | Begin Date | End Date | Location | Work content |
| 1 | Creating a test plan | 17.03.2022 | 20.03.2022 | Minsk, Belarus | Jitsi |
| 2 | Test cases’ creation | 21.03.2022 | 26.03.2022 | Minsk, Belarus | Jitsi |
| 3 | Smoke Test execution | 28.03.2022 | 01.04.2022 | Minsk, Belarus | Jitsi |
| 4 | Critical path Test execution | 28.03.2022 | 01.03.2022 | Minsk, Belarus | Jitsi |

| Revision history | | | | | |
| --- | --- | --- | --- | --- | --- |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.1. | All New | Artsiom Hur, Nikita Kotov | 20.03.2022 | - | 20.03.2022 |
| 1.2. | New 1. Introduction  New 2. Scope of work  New 3. Quality and acceptance criteria  New 6. Resources  New 8.2 Test methods  New 8.3 Test types | Nikita Kotov | 27.03.2022 | - | 27.03.2022 |