QA Module Documentation

Overview:

The QA module will test all of the specified functionalities of the application and will ensure that every component is ready to handle any type of exception.

Objectives:

- Establish testing strategies according to the specified use-cases
- Create testing logs in order to keep track of progress
- Signal any faulty components as soon as possible with details on the problem
- Ensure that all expected functionalities work as intended
- Create a user manual

Test parameters:

- response time: the time in which the application will provide the result to the user must meet the user's expectations
- functionality: each button or option will do what the user expects it to do
- stability: all exceptions will be handled by the application so that it does not stop working
- relevant name or description: each button or option will have an intuitive name that will correspond to the functionality

Functionalities to be tested:

- Register

- Log In

- Import

- Translate

- Edit

- Search

- The interface

Module	Initial testing	Intermediary tests(Junit)	Final tests*
Register	Yes	Yes	Yes
Log in	Yes	Yes	Yes
Import	Yes	Yes	Yes
Translate	No	Yes	Yes
Edit	Yes	Yes	Yes
Search	No	Yes	Yes
The Interface	No	No	Yes

List of tests for:

1.Register:

1.1- username and password length constraints

1.2- the correct encryption of passwords

1.3- the appropriate handling of the possible exceptions

Possible issues:

1.1.1- The application will allow a username of length 0 to be registered alongside a length 0 password. This can lead to security issues.

Possible causes:

- the application does not check length constraints for the credentials

1.1.2- The application will allow a length 0 username and a correct password to register. This can lead to database inconsistencies: more passwords than usernames.

Possible causes:

- the application does not check length constraints for the username

1.1.3- The application will allow a correct username and a length 0 password to register. This can lead to security issues.

Possible causes:

- the application does not check length constraints for the password

1.2.1 - The application will not correctly encrypt the password. This can lead to inconsistencies between the database and the user.

Possible causes:

- flawed encryption algorithm

1.2.2- The application will not encrypt the password. This can lead to security issues.

Possible causes:

- flawed encryption algorithm

1.3.1- An exception arises, but the application either crashes or proceeds to the next step in a flawed state. This can lead to security issues, inconsistencies with the database, denial of services which require the user to be logged in while the user thinks he has logged in.

Possible causes:

- the application does not check for or handle the exception before proceeding

2.Log In:

2.1- the validation of the entered credentials

2.2- the correct encryption of passwords

2.3- the appropriate handling of possible exceptions

Possible issues

2.1.1- The application allows the user to log in with an unregistered username. This can lead to inconsistencies with the database and also security issues if the username is supposed to have administrator privilages.

Possible causes:

- the application does not check if the username exists in the database

- the application proceeds to function normally in spite of the fact that the username does not exist in the database instead of throwing an exception

2.1.2- The application allows the user to log in using a incorrect password. This can lead to security issues.

Possible causes:

- the application does not check if the password exists in the database

- the application does not check if the password corresponds to the username

- the application preceeds to function normally in spite of the fact that the password is incorrect instead of throwing an exception

2.2.1- The application will not correctly encrypt the password. This will render the user unable to log in.

Possible causes:

- flawed encryption algorithm

2.2.2- The application will not encrypt the password. This can lead to security issues.

Possible causes:

- flawed encryption algorithm

3.**Import:**

We will first interrogate the WordNet in English to give us the meaning of a chosen word. The received information will then be stored in a separate file. Then, we will test if the method implemented by the developers returns the same result as the one stored in the file. If so, then the test passes, else the test fails.

Possible issues

3.1-The application might not import anything or may import only partially correct, in the sense that it doesn't import a chosen word, instead it imports a different word that begins with the same letter ot something similar or doesn't get the full gloss and synset.

Possible causes:

-flawed importing method

3.2-The application might crack if the user enters more than 1 word in the "Import" column.

Possible causes:

3.3 -The application lacks a proper alert which warns the user that he cannot enter more than 1 word in the "Import" column.

4.Translate:

Here we will test if the result of the English to Romanian translation method created by the developers matches the Romanian translation we give to a common English word. For example we check if the translation of the word "dog" returns the Romanian word "caine". If so, then the test passes, else the test fails.

Possible issues

4.1-The application might not return the correct translation

Possible causes:

-flawed translate method

5.Edit:

Possible issues

5.1- The application will not find the word in the database.

Possible causes:

- The application does not have a record of the word.

- The search function doesn't work properly.

5.2- The application will not save the modifications done.

Possible causes:

5.3- The function for saving the modifications failed to update the database.

6.**Search:**

We will first interrogate the application database to give us the word we are searching for and store the result in a separate file. Afterwards, we test whether the searching method implemented by the developers returns the same result as the one stored in the file. If so, then the test passes, else the test fails.

Possible issues

6.1-The application might not search for the correct word in the sense that it might search for a word that begins with the same letter instead of a full word. Also it might not get the full synset.

Possible causes:

-flawed searching method

6.2-The application might crash if the user enters more than one word in the "Search" column.

Possible causes:

- The application lacks a proper alert which warns the user that he cannot enter more than 1 word in the "Search" column.
- The application does not throw and handle an exception

6.3-The application might crash if a chosen word doesn't exist in the database.

Possible causes:

- Lack of an alert which signals that the word does not exist.

- The application does not throw and handle an exception

7.**The Interface:**

7.1- user friendly

7.2- the buttons have functionalities that do what they are inteded to do

7.3- the appropriate handling for the possible exceptions

Possible issues

7.1.1- The application is hard to use

Possible causes:

- hidden menus

- inappropiate button combinations (ex: "search" with "edit profile" in the "import menu")

- the buttons have the wrong functionalities attached to them

7.2.1- A button does not do what it says (ex. "search" will allow you to edit your profile)

Possible causes:

- the button has the wrong functionality attached to it

7.2.2- A button does not do anything

- the button has no functionality attached to it

7.3.1- The application will allows the user to edit his profile even though he is not logged in

Possible causes:

- the edit functionality does not check whether the user is logged in or not

7.3.2- An exception arises, but the application does not notify the user and continues to run

Possible causes:

- the application does not handle the exception in the interface before proceeding

The application will also be given to a set of first time users in order to receive feedback so that we can further assess whether the application is user friendly or not. The feedback, along with some of our own, will be passed down to the developers.

Instruments used

- Eclipse IDE
- JUnit

Team Composition

- Alexandru Rusu (B4)
- George Stafie (A3)
- Daneliuc Mihai (A4)
- Nechita Bogdan-Radu (A4)
- Bety Hapau