Université Grenoble Alpes

M1 MIASHS IC

Test plan

TER Project

2021 - 2022

Descotils Juliette
Gattaciecca Bastien

1. Introduction

1.1. Objectives and methods

The purpose of this document is to list the tests that have been performed to ensure that the software produced corresponds to the specifications.

1.2. Reference documents

This document is based on the project's source code and therefore on the development plan written during the design phase of the project. The internal documentation of the software also allows to understand the functioning and the relations of each module.
This document is also largely based on the request described in the specifications.
The scenarios described in the functional tests section are those present in the task model diagrammed during the design phase.

2. Reading Guide

2.1. Project Management

It is strongly recommended to redo the unit and integration tests after each modification of the code to ensure that the new code does not produce any side effects with the existing code.
Unit tests are more for developers because they require understanding and studying the source code to execute them. Functional tests to be repeated once all unit tests have been performed.

2.2. Project Owner

This document lists the tests necessary for the proper functioning of the application. Passing all the tests will validate the functional objectives of the application.
Functional tests describe the functionalities described in the specifications and can be easily reproduced by the client. Unit tests are more complex and require an understanding of the source code to execute, so it is wiser to have them tested by someone experienced in development.

3. Basic concepts

The application is used to annotate the relevance of words, documents and snippets regarding a topic. It is intended to be used by experts in the field of the topic and by administrators who need the results of annotations, usually to test the performance of search engines.
The type of annotation requested (word annotation, document annotation or both) is defined for the entire annotation campaign. The type of document to annotate (document, snippet or both) is also the same through the whole campaign.
Several experts and administrators must be able to access the application simultaneously and privately, so there are several sessions and an authentication system. Experts have access to the assessment functionality which is used to annotate the relevance of documents. Administrators can download the completed and uncompleted assessment files as well as adding new experts and assessments to the database (via files).

4. Functional tests

4.1. Functional test 1

4.1.1. Description

Connect to the application as an expert.

4.1.2. Constraints

Modify the access parameters to the resource folders in the settings file.
Load the campaign and the files containing the expert and administrator lists.

4.1.3. Dependencies

None.

4.1.4. Test procedure

Click on 'as expert' on the login page.
Enter an existing username and password from the previously loaded expert file. Click on login.
The expert's dashboard should be displayed.

4.2. Functional test 2

4.2.1. Description

Connect to the application as an admin.

4.2.2. Constraints

Modify the access parameters to the resource folders in the settings file.
Load the campaign and the files containing the expert and administrator lists.

4.2.3. Dependencies

None.

4.2.4. Test procedure

Click on “as administrator” on the login page.
Enter an existing username and password from the previously loaded expert file. Click on login.
The administrator’s dashboard should be displayed.

4.3. Functional test 3

4.3.1. Description

Complete assessment.

4.3.2. Constraints

Load the campaign and the administrators, experts, assessments (snippet, document) and assignments files beforehand.
Install a database viewer extension (SQLite Viewer for example).

4.3.3. Dependencies

None.

4.3.4. Test procedure

Connect to the application as an expert, click on “start assessment”, check one of the global relevance radio buttons then click on one of the save buttons at the end of the page.
The values of the “annotation” column in the “global_assessment” table should all be set at -2 except for the one that has been annotated.

4.4. Functional test 4

4.4.1. Description

Set word relevance on a word type campaign.

4.4.2. Constraints

Load the campaign and the administrators, experts, assessments (topics, snippets and/or documents) and global tags and/or word tags files beforehand. Load an assignations file linking at least one snippet or document to an expert.
Install a database viewer extension (SQLite Viewer for example).

4.4.3. Dependencies

None.

4.4.4. Test procedure

Connect to the application as an expert, click on “start assessment”, click on one of the words of the abstract. Pressing the right arrow key on the keyboard should move the black frame around the word, pressing the space bar should change the background color of the word. Check one of the global relevance radio buttons then click on one of the save buttons at the end of the page.
The values corresponding to the clicked words should have an index different from -2 in the annotation column in the "words_assessments" table.

4.5. Functional test 5

4.5.1. Description

Validate assessment and go to the next one.

4.5.2. Constraints

Load the campaign and the administrators and global tags and/or word tags files beforehand. Load a document file containing two document’s uri/url, an expert file containing one expert and an assignation file linking the expert's id to the ids of the two documents.
Install a database viewer extension (SQLite Viewer for example).

4.5.3. Dependencies

None.

4.5.4. Test procedure

Connect to the application as the expert, click on “start assessment”. Check one of the global relevance radio buttons then click on the “Save & Next assessment” button at the bottom of the page.
The second assessment should be displayed. Open the database. The “global_assessment” table should contain two lines. The “annotation” column should contain a value different from -2 on the first line and -2 on

the second line. Then check one of the global relevance radio buttons again. The “Save & Next assessment” button should not be clickable and the “annotation” value of the second line should have been changed.

4.6. Functional test 6

4.6.1. Description

Abandon assessment.

4.6.2. Constraints

Load the campaign and the administrators, experts, assessments (topics, snippets and/or documents) and global tags and/or word tags files beforehand. Load an assignations file linking at least one snippet or document to an expert.
Install a database viewer extension (SQLite Viewer for example).

4.6.3. Dependencies

None.

4.6.4. Test procedure

View the database. The values of all rows in the "annotation" column of the "global_assessment" table and the "word_assessment" table should be -2. Click on one of the global relevance radio buttons and on some words in the abstract. Click on "Abandon & Back to dashboard". The expert’s dashboard should be displayed. Refresh the database. None of the values in the "annotation" columns should have changed,they should all remain set at -2.

4.7. Functional test 7

4.7.1. Description

Read detailed instructions.

4.7.2. Constraints

Load the campaign file containing a valid link to an instructions page in the "detailed_instructions_URL" tag. Load an expert file, an amins file, a topics file, a documents file or snippets file. Load an assignment file linking an expert to a document and a topic. Load global_tags and words_tags files.

4.7.3. Dependencies

None.

4.7.4. Test procedure

Log in as an expert and click on "detailed instructions". A new web page corresponding to the link specified in the campaign file should appear.

4.8. Functional test 8

4.8.1. Description

Export the completed assessment file.

4.8.2. Constraints

Load the campaign and the administrators, experts, topics, assessments (snippets and/or documents) and global tags and/or word tags files beforehand. Load an assignations file linking at least one snippet or document to an expert and a topic.
Log in as an expert and annotate at least one document or snippet by clicking on at least one global relevance radio button on the assessment page.

4.8.3. Dependencies

None.

4.8.4. Test procedure

Log in as administrator and click on export completed assessment. A .zip folder containing two files (one for global annotations, one for word annotations) should download if the assessment type defined in the campaign is "global/words". A single file should download if the assessment type defined in the campaign is simply "global" or simply "words". The files must contain the date, the assessment type, and the assignment type in the title. They must contain as many lines as there are annotated words/documents/snippets.

4.9. Functional test 9

4.9.1. Description

Export the remaining assessment file.

4.9.2. Constraints

Load the campaign and the administrators, experts, topics, assessments (snippets and/or documents) and global tags and/or word tags files beforehand. Load an assignations file linking at least one snippet or document to an expert and a topic.
Log in as an expert and annotate at least one document or snippet by clicking on at least one global relevance radio button on the assessment page.

4.9.3. Dependencies

None.

4.9.4. Test procedure

Log in as administrator and click on export remaining assessment. A .zip folder containing two files (one for global annotations, one for word annotations) should download if the assessment type defined in the campaign is "global/words". A single file should download if the assessment type defined in the campaign is simply "global" or simply "words". The files must contain the date, the assessment type, and the assignment type in the title. They must contain as many lines as there are annotated words/documents/snippets.

5. Integration tests

5.1. Integration test 1

5.1.1. Description

Reading the Campaign's file which is campaign.xml. Checks if the application can successfully parse the campaign xml file and extract 9 variables from it.

5.1.2. Constraints

As it is a Campaign file, only one record is expected. If the campaign's file contains more than one <campaign> element, only the first one is kept. A campaign.xml file is linked to a DTD that validates it.

5.1.3. Dependencies

None.

5.1.4. Test procedure

Calling parseFile method from CampaignReader class. Expected result is an array containing the 9 key-values element. A key is a campaign's table column name and a value is the value we associate to this key.

5.2. Integration test 2

5.2.1. Description

Loading the all files needed to launch the campaign (campaign.xml**,** administrator.xml, topics.xml, documents.xml, snippets.xml, tags_global.xmltags_words.xml) and seeing the information in the database.

5.2.2. Constraints

Install a database viewer extension (SQLite Viewer for example).

5.2.3. Dependencies

None.

5.2.4. Test procedure

The lines contained in the loaded file should be added to the database. Columns corresponding to the information contained in the files should be created in each table and each line of the file should be parsed and added to the table.

5.3. Integration test 3

5.3.1. Description

Successfully connect as an Expert to the dashboard view.

5.3.2. Constraints

Have valid expert credentials.

5.3.3. Dependencies

None.

5.3.4. Test procedure

Once on the welcome page, select that you want to connect as an Expert. Once redirected to the login view, enter the credentials and press Login. If no error occurs, you should be redirected to the expert dashboard view.

5.4. Integration test 4

5.4.1. Description

Successfully connect as an Administrator to the dashboard view.

5.4.2. Constraints

Have valid administrator credentials.

5.4.3. Dependencies

None.

5.4.4. Test procedure

Once on the welcome page, select that you want to connect as an Administrator. Once redirected to the login view, enter the credentials and press Login. If no error occurs, you should be redirected to the admin dashboard view.

5.5. Integration test 5

5.5.1. Description

Successfully disconnect.

5.5.2. Constraints

Must be already connected (as an expert or as an admin, it doesn't matter).

5.5.3. Dependencies

None.

5.5.4. Test procedure

Once disconnected, you should be redirected to the welcome page. To check that you are actually disconnected, try to reconnect with the same role you had when you were connected. If you are not really disconnected (if the dashboard view has been simply closed) you should be directly redirected to the dashboard view. If you were actually disconnected, you should be redirected to the login view.

6. Tests unitaires

Only the unit tests of the methods of the model part of the MVC are written in the rest of this document because the methods of the controller are difficult to isolate and the methods of the view are tested multiple times in the integration tests and the functional tests.

6.1. Unit test 1

6.1.1. Description

The parseFile method (defined in the readers classes : AdminReader.php, AssignaionReader.php, CampaignReader.php, DocumentReader.php, ExpertReader.php, GlobalTagReader.php, SnippetReader.php, TopicReader.php, WordTagReader.php) parses the file given in parameter and returns an associative array containing the information written in the file in each column.

6.1.2. Constraints

Create an xml admins file in the “ressources” directory of the application writing one line per administrator as follow :

```
<?xml?>
<admins>
   <admin id="Alexandra" password="01378"/>
   <admin id="Natasha" password="456378"/>
</admins>
```

6.1.3. Dependencies

None.

6.1.4. Test procedure

Call the parseFille method by passing as a parameter the path of the admins file created. The method should return an array of two columns (“id” and “password”) and two rows (one for each expert) containing the file information.

6.2. Unit test 2

6.2.1. Description

The parseFile method (defined in the readers classes : AdminReader.php, AssignaionReader.php, CampaignReader.php, DocumentReader.php, ExpertReader.php, GlobalTagReader.php, SnippetReader.php, TopicReader.php, WordTagReader.php) throws an error if the path given in parameters does not correspond to any file.

6.2.2. Constraints

None.

6.2.3. Dependencies

Unit test 1.

6.2.4. Test procedure

Call the parseFile method by passing as a parameter a wrong path. The method should write "Assignment file doesn't exist here:" and the path passed in parameter in the logs.

6.3. Unit test 3

6.3.1. Description

The parseFile method (as defined in the readers classes : AssignationReader.php and ExpertReader.php) throws an error if the path given in parameters leads to a text file that does not respect the document format expected.

6.3.2. Constraints

Create an assignations file with the wrong format for example :

```
Alexandra
Natasha
```

6.3.3. Dependencies

Unit test 1.

6.3.4. Test procedure

Call the parseFile method by passing as a parameter the admins file's path. The method should return "Assignation file at " (the path passed in parameters) "could not find 3 elements at line 1".

6.4. Unit test 4

6.4.1. Description

The parseFile method (as defined in the readers classes :AssignaionReader.php, CampaignReader.php, DocumentReader.php, GlobalTagReader.php, SnippetReader.php, TopicReader.php and WordTagReader.php) throws an error if the path given in parameters leads to an xml file that is impossible to load.

6.4.2. Constraints

None.

6.4.3. Dependencies

Unit test 1.

6.4.4. Test procedure

Call the parseFile method by passing as a parameter a path to a text file. The method should return "Impossible to load snippet file from" and the path passed in parameter.

6.5. Unit test 5

6.5.1. Description

The parseFileFromContent method (defined in the AssignationReader and ExpertReader classes) parses a file given as a string and returns an array containing the information in each column.

6.5.2. Constraints

None.

6.5.3. Dependencies

None.

6.5.4. Test procedure

Call parseFileFromContent method from the AssignationReader class by passing this string in parameter :

"Farid 002 582
Juliette 005 582
Farid 009 002
Farid 012 456".

Expected result is an array containing the 3 key-values element. A key is a assignation's table column name and a value is the value associated with this key. Here the result should be a 4 rows, 3 columns table containing the experts ids, the topics ids and the documents ids.

6.6. Unit test 6

6.6.1. Description

The fileExists method as defined in FileReader returns true if the file path in the parameter is valid.

6.6.2. Constraints

None.

6.6.3. Dependencies

None.

6.6.4. Test procedure

Call fileExists method from the FileReader class by passing a valid path file in parameter such as : D:\Users\...\ressources\admins.xml

6.7. Unit test 7

6.7.1. Description

The fileExists method as defined in FileReader returns false if the file path in the parameter is invalid.

6.7.2. Constraints

None.

6.7.3. Dependencies

None.

6.7.4. Test procedure

Call fileExists method from the FileReader class by passing an invalid path file in parameter such as : D:\Users\...\resso/urces\

6.8. Unit test 8

6.8.1. Description

The createTable method (defined in the models classes : Admin.php, Assignation.php, Campaign.php, Document.php, Expert.php, GlobalAssessment.php, GlobalTag.php, Settings.php, Snippet.php, Topic.php, WordAssessment.php, WordTag.php) creates a table in the database with a valid associative array as parameter.

6.8.2. Constraints

Unit test 1. Unit test 2.

6.8.3. Dependencies

Must have a database viewer.

6.8.4. Test procedure

Call createTable method from the Admin class (example) and then open the database and then refresh. Expected result is a new table called ‘admins’ with 2 columns named “id” and “password”.

6.9. Unit test 9

6.9.1. Description

The fillTable method (defined in the models classes : Admin.php, Assignation.php, Campaign.php, Document.php, Expert.php, GlobalAssessment.php, GlobalTag.php, Settings.php, Snippet.php, Topic.php, WordAssessment.php, WordTag.php) fills a table in the database with a valid associative array as parameter.

6.9.2. Constraints

Unit test 1. Unit test 2. Unit 8.

6.9.3. Dependencies

None.

6.9.4. Test procedure

Call fillTable method from the Admin class (example) with this associative array as parameter :

```
Array
(
    [0] => Array
        (
            [id] => Farid
            [password] => 48952
        )

    [1] => Array
        (
            [id] => Juliette
            [password] => 5896
        )

    [2] => Array
        (
            [id] => Dimitri
            [password] => 6588
        )

    [3] => Array
        (
            [id] => Nathaniel
            [password] => 01235
        )

    [4] => Array
        (
            [id] => Bastien
            [password] => 02088
        )

    [5] => Array
        (
            [id] => Francis
            [password] => 9648
        )
```

)
Go to the database and open the admins table. You should see the following records added to the table :

id	password
Farid	48952
Juliette	5896
Dimitri	6588
Nathaniel	01235
Bastien	02088
Francis	9648

6.10. Unit test 10

6.10.1. Description

The annotableAbstract method from Assessment returns the abstract without any additional blank spaces and separators.

6.10.2. Constraints

None.

6.10.3. Dependencies

None.

6.10.4. Test procedure

Call annotableAbstract method from Assessment with
“Le terme tsunami est un mot japonais composé de tsu, (port), (gué), et de nami, (vague)”
as a String parameter. It should return this String :
“Le terme tsunami est un mot japonais composé de tsu port gué et de nami vague”.

6.11. Unit test 11

6.11.1. Description

The parseAbstract method from Assessment returns the abstract where every clickable word is surrounded by a span block.

6.11.2. Constraints

None.

6.11.3. Dependencies

None.

6.11.4. Test procedure

Call parseAbstract method from Assessment with
“Bonjour, je m’appelle Juliette”
It should returns:
“Bonjour, jem’appelle Juliette"

6.12. Unit test 12

6.12.1. Description

The exportAssessment method from Assessment returns an array of String containing several paths. exportAssessment takes as parameter a string indicating if we are looking for the “exported” or “remaining” assessments. Both work in the same way so we'll do an example with “completed”.

6.12.2. Constraints

Set the campaign’s type as “globalwords”.

6.12.3. Dependencies

None.

6.12.4. Test procedure

Call exportAssessment method from Assessment with “completed” as a string parameter. Expected result is an array containing the two file paths created by the method. In the exported folder at the root of the project, a compressed file named with the datetime containing two txt files should appear.

6.13. Unit test 13

6.13.1. Description

The exportAssessment method from Assessment returns a String containing one path. exportAssessment takes as parameter a string indicating if we are looking for the “exported” or “remaining” assessments. Both work in the same way so we'll do an example with “completed”.

6.13.2. Constraints

Set the campaign’s type as “global”.

6.13.3. Dependencies

None.

6.13.4. Test procedure

Call exportAssessment method from Assessment with “completed” as a string parameter. Expected result is a String containing the file path created by the method. In the exported folder at the root of the project, a file named with the datetime should appear.

6.14. Unit test 14

6.14.1. Description

The exportAssessment method from Assessment returns false because the exported folder is not found.

6.14.2. Constraints

Set the EXPORT_PATH attribute from Settings.php to “”.

6.14.3. Dependencies

None.

6.14.4. Test procedure

Call exportAssessment method from Assessment with “completed” as a string parameter. Expected result is false.

6.15. Unit test 15

6.15.1. Description

Call assessmentOf method from Assessment returns an Assessment object.

6.15.2. Constraints

Unit test 9.

6.15.3. Dependencies

None.

6.15.4. Test procedure

The assessmentOf method from Assessment returns the Assessment object linked to the assignation as parameter.

6.16. Unit test 16

6.16.1. Description

Call the getNbCompletedAssessments method from Expert.

6.16.2. Constraints

Unit test 9.

6.16.3. Dependencies

None.

6.16.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Check the assignations tables. Calling getNbCompletedAssessments on Farid expert should return 2.

6.17. Unit test 17

6.17.1. Description

Call the getNbRemainingAssessments method from Expert.

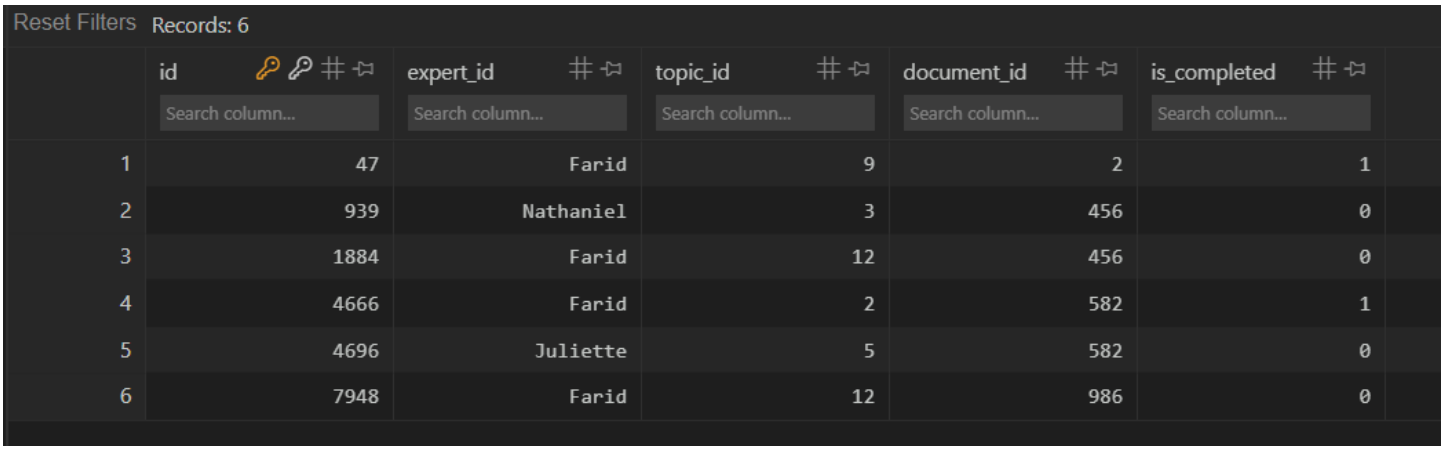
6.17.2. Constraints

Unit test 9.

6.17.3. Dependencies

None.

6.17.4. Test procedure


Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Check the assignations tables. Calling getNbRemainingAssessments on Farid expert should return 2.

6.18. Unit test 18

6.18.1. Description

Call the getNbAssessments method from Expert.

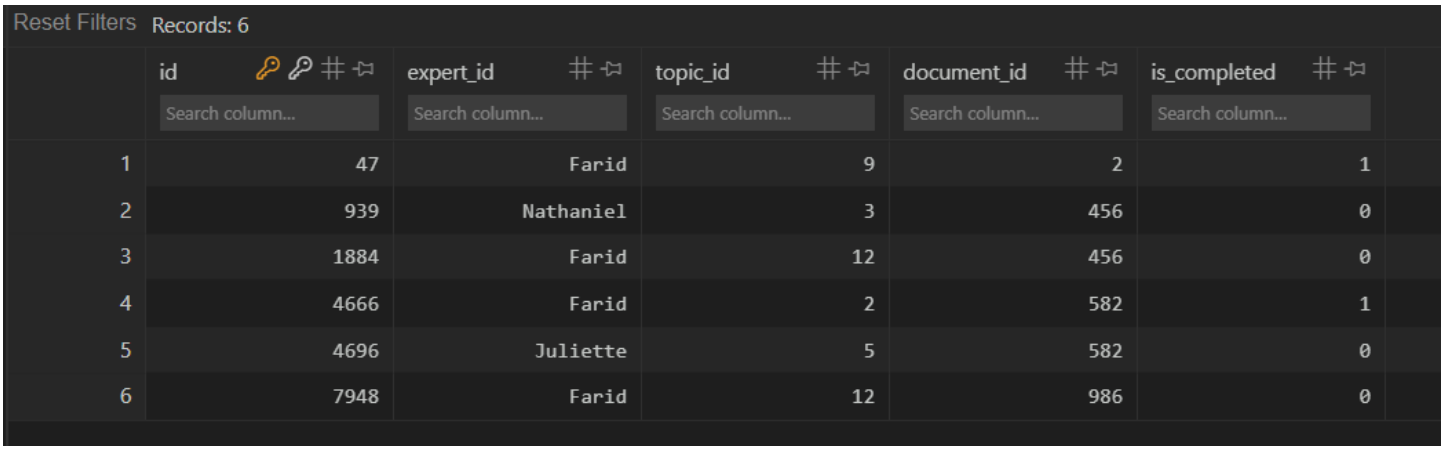
6.18.2. Constraints

Unit test 9.

6.18.3. Dependencies

None.

6.18.4. Test procedure


Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Check the assignations tables. Calling getNbAssessments on Farid expert should return 4.

6.19. Unit test 19

6.19.1. Description

Call the getTopicsId method from Expert.

6.19.2. Constraints

Unit test 9.

6.19.3. Dependencies

None.

6.19.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Check the assignations tables. Calling getTopicsId on Farid expert should return an array of 3 elements containing [9, 12, 2].

6.20. Unit test 20

6.20.1. Description

Call the getAdvancement method from Expert.

6.20.2. Constraints

Unit test 9.

6.20.3. Dependencies

None.

6.20.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Check the assignations tables. Calling getAdvancement on Farid expert should return an array like this :

```
[
  [id] => Farid,
  [nbTotal] => 4,
  [nbCompleted] => 2,
  [nbRemaining] => 2,
  [isDone] => false,
]
```

6.21. Unit test 21

6.21.1. Description

Call the getNbCompletedTopics method from Expert.

6.21.2. Constraints

Unit test 9.

6.21.3. Dependencies

None.

6.21.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Call getNbCompletedTopics method from Expert on Farid should return 2.

6.22. Unit test 22

6.22.1. Description

Call the getNbRemainingTopics method from Expert.

6.22.2. Constraints

Unit test 9.

6.22.3. Dependencies

None.

6.22.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Call getNbRemainingTopics method from Expert on Farid should return 1.

6.23. Unit test 23

6.23.1. Description

Call the getNbTopics method from Expert.

6.23.2. Constraints

Unit test 9.

6.23.3. Dependencies

None.

6.23.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	0
6	7948	Farid	12	986	0

Call getNbTopics method from Expert on Farid should return 3.

6.24. Unit test 24

6.24.1. Description

Call the hasFinished method from Expert.

6.24.2. Constraints

Unit test 9.

6.24.3. Dependencies

None.

6.24.4. Test procedure

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call hasFinished method from Expert on Farid should return false.
Call hasFinished method from Expert on Juliette should return true.

6.25. Unit test 25

6.25.1. Description

Call the getNbCompletedTasks method from Expert.

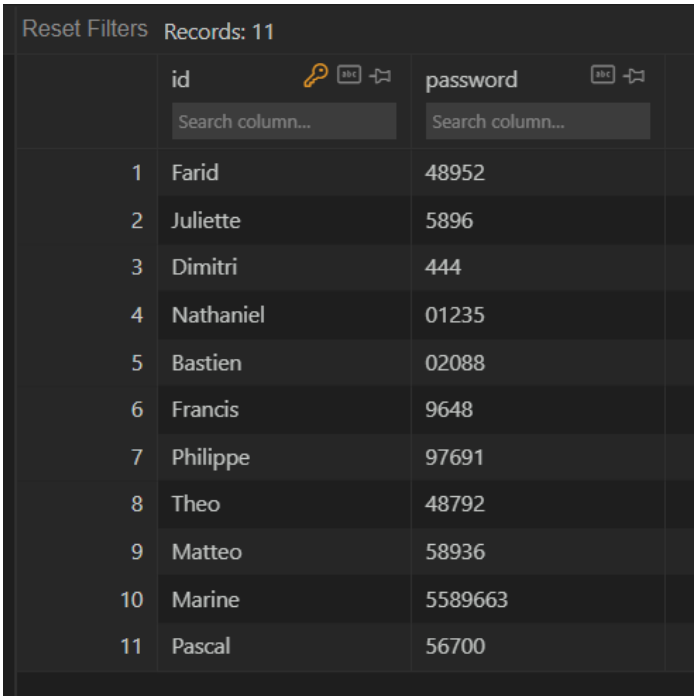
6.25.2. Constraints

Unit test 9.

6.25.3. Dependencies

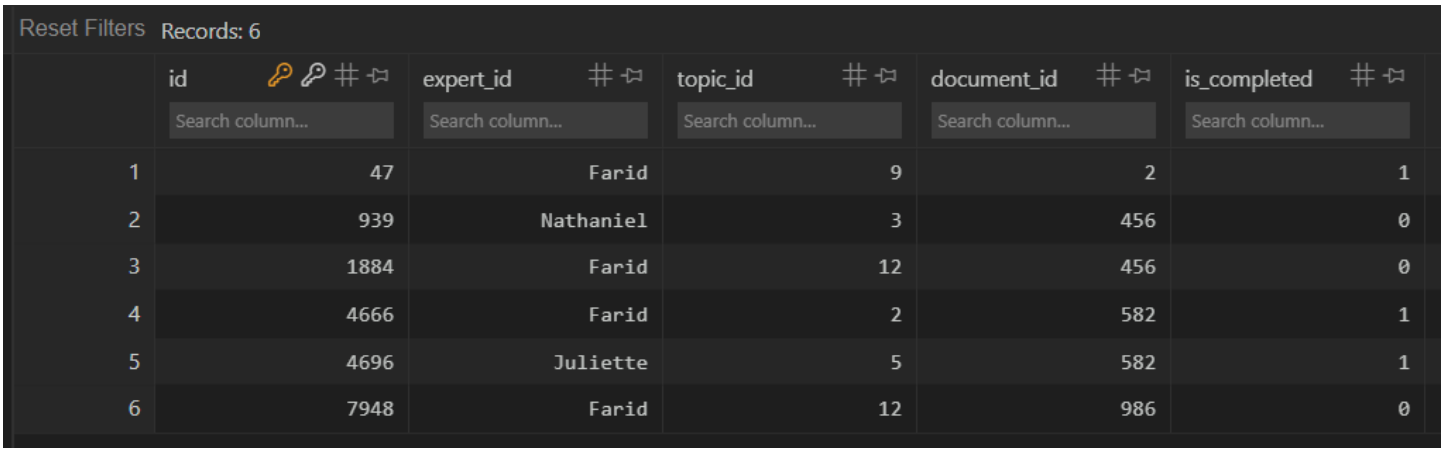
None.

6.25.4. Test procedure



Reset Filters Records: 11

	id	password
	Search column...	Search column...
1	Farid	48952
2	Juliette	5896
3	Dimitri	444
4	Nathaniel	01235
5	Bastien	02088
6	Francis	9648
7	Philippe	97691
8	Theo	48792
9	Matteo	58936
10	Marine	5589663
11	Pascal	56700



Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call getNbCompletedTasks method from Expert should return 9 because out of the 11 experts, 9 have no more assignments to do.

6.26. Unit test 26

6.26.1. Description

Call the getNbRemainingTasks method from Expert.

6.26.2. Constraints

Unit test 9.

6.26.3. Dependencies

None.

6.26.4. Test procedure

Reset Filters Records: 11

	id	password
	Search column...	Search column...
1	Farid	48952
2	Juliette	5896
3	Dimitri	444
4	Nathaniel	01235
5	Bastien	02088
6	Francis	9648
7	Philippe	97691
8	Theo	48792
9	Matteo	58936
10	Marine	5589663
11	Pascal	56700

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call getNbRemainingTasks method from Expert should return 2 because out of the 11 experts, 2 still have assignments to do.

6.27. Unit test 27

6.27.1. Description

Call the getNbTasks method from Expert.

6.27.2. Constraints

Unit test 9.

6.27.3. Dependencies

None.

6.27.4. Test procedure

Reset Filters Records: 11

	id	password
	Search column...	Search column...
1	Farid	48952
2	Juliette	5896
3	Dimitri	444
4	Nathaniel	01235
5	Bastien	02088
6	Francis	9648
7	Philippe	97691
8	Theo	48792
9	Matteo	58936
10	Marine	5589663
11	Pascal	56700

Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call getNbTasks method from Expert should return 11 because there are 11 experts in the expert list.

6.28. Unit test 28

6.28.1. Description

Call the isLastAssignation method from Expert.

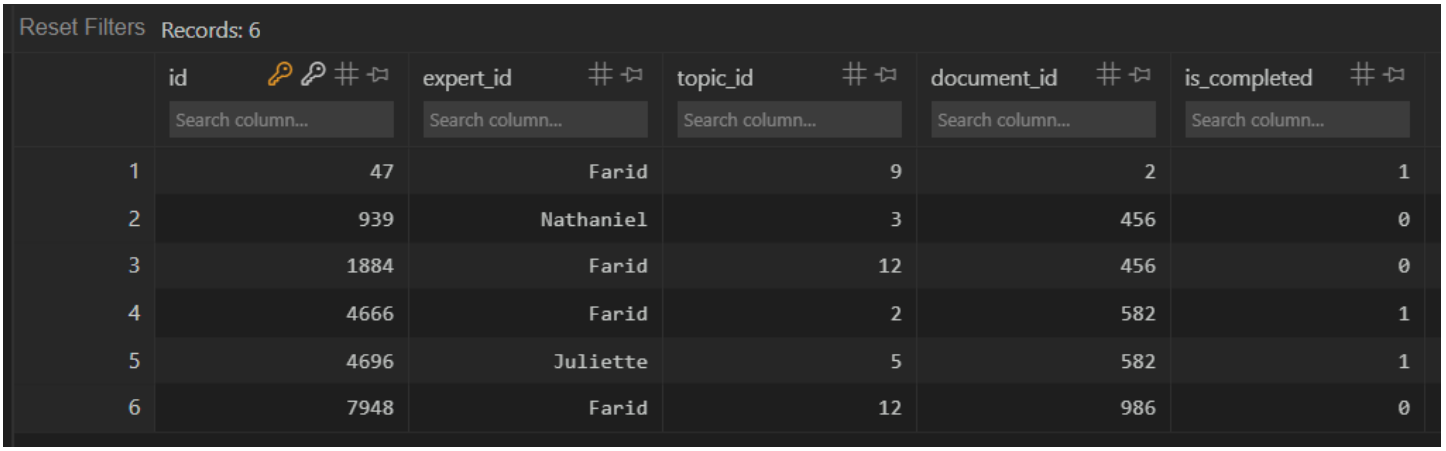
6.28.2. Constraints

Unit test 9.

6.28.3. Dependencies

None.

6.28.4. Test procedure


Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call isLastAssignation method from Expert on Farid should return false because there are 2 assignments left for Farid.
Call isLastAssignation method from Expert on Nathaniel should return true because there is only 1 assignment left for Nathaniel.

6.29. Unit test 29

6.29.1. Description

Call the getNextAssignation method from Expert.

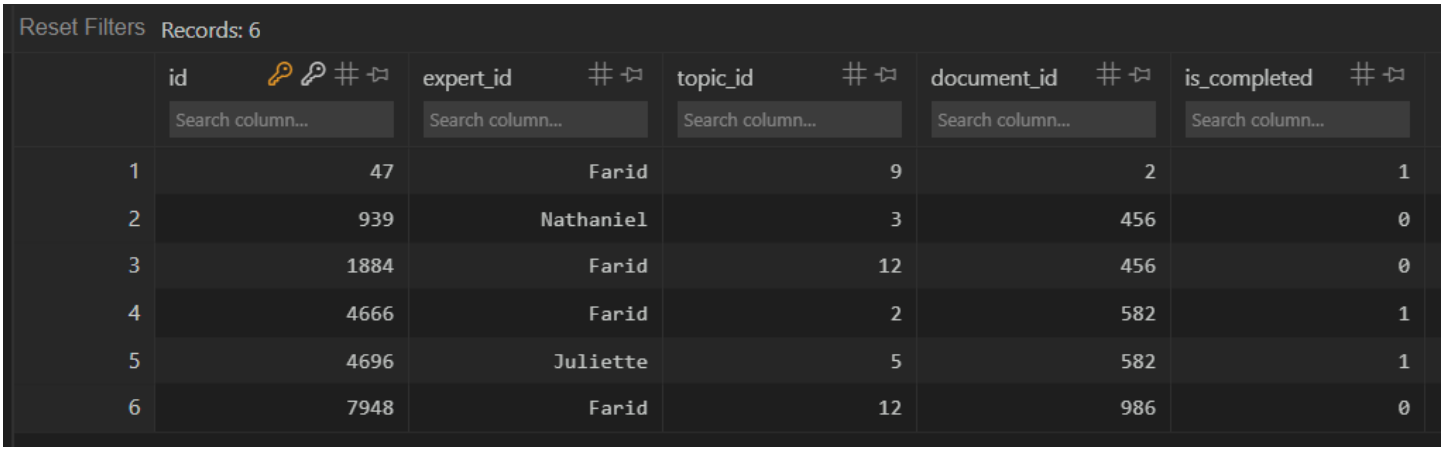
6.29.2. Constraints

Unit test 9.

6.29.3. Dependencies

None.

6.29.4. Test procedure


Reset Filters Records: 6

	id	expert_id	topic_id	document_id	is_completed
	Search column...	Search column...	Search column...	Search column...	Search column...
1	47	Farid	9	2	1
2	939	Nathaniel	3	456	0
3	1884	Farid	12	456	0
4	4666	Farid	2	582	1
5	4696	Juliette	5	582	1
6	7948	Farid	12	986	0

Call getNextAssignation method from Expert on Farid should return the Assignation object with id 4666 because assignments are called by order of topic id.
Call getNextAssignation method from Expert on Juliette should return false because Juliette has no more uncompleted assignments.

6.30. Unit test 30

6.30.1. Description

Call the changePassword method from Expert.

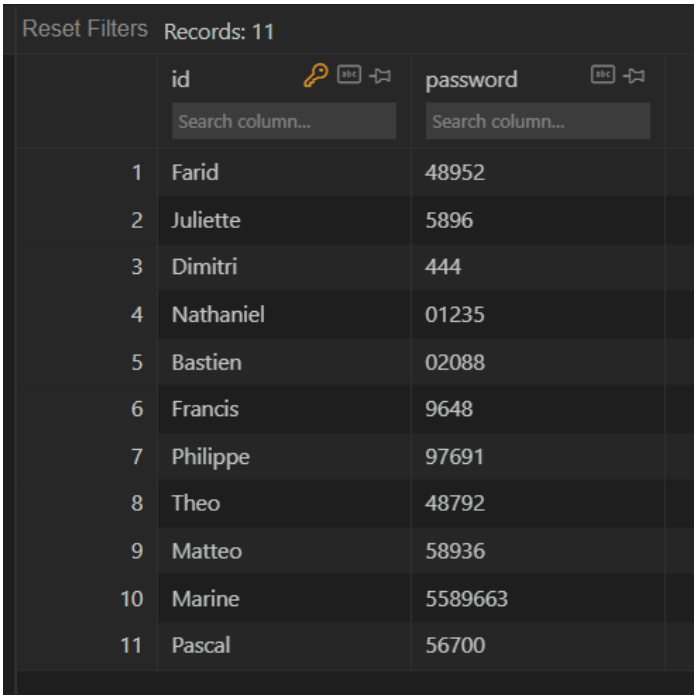
6.30.2. Constraints

Unit test 9.

6.30.3. Dependencies

None.

6.30.4. Test procedure



Call changePassword method from Expert with Farid and “azeaze” as parameters should update Farid’s password.
Call changePassword method from Expert with Damien and “azeaze” as parameters should write an error into the logs file and return false because Damien is not in the expert list.

7. Verification of the documentation

The verification between the procedures described in the documentation and the implementation of these procedures was done by cross reading. That is, one of the members of the pair verified the feasibility of the tests and the accuracy of the explanations while the other wrote them.

8. Glossary

Abstract: the annotatable part of a **document** or **snippet**.
Administrator: the person in charge of the management of the overall **assessment**.
Annotation: one annotation consists in setting a degree of **relevance** to one word (word annotation), **snippet** or **document** (global annotation).
Assessment: one assessment consists, for an expert, in annotating one **snippet**/**document** for a given **topic**.
Corpus: set of **documents** on which the relevance is evaluated.
Campaign: the organization of the rating campaign. It is defined by an ID, its activation state, the type of target (**snippets** or **documents**), the type of assessment (**global** and/or **words**), a description and rating instructions.
Document: a document is a long string defined by an ID and an internal URI or external URL.
Expert: the person who has knowledge about the **topic**. He uses the application to do the **assessment**.

Global relevance: the relevance degree of a document or snippet in its entirety. It has to be set by the **expert.**
Relevance: the degree of global relevance and word relevance is evaluated on a scale defined in the **campaign**.
SERP: search engine result page.
Snippet: one snippet refers to one **document**. It is displayed in a frame, gives the title and a short preview of this document.
Topic: the query against which the **experts** must annotate the **relevance** of the **documents** and/or **snippets**. It is a character string.
Word relevance: the relevance degree of a word of the **abstract**. It has to be set by the **expert.**

9. References

Laravel documentation https://laravel.com/docs/9.x
Pages and Files Specifications link