Design Document

ExamIT

AUTOMATIC TEST CORRECTION PLATFORM

4th year, Software Development project
Institute Of Technology Carlow

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY
CARLOW
At the heart of South Leinster

Roger Marciniak

Student Number: C00169733
Supervisor: Paul Barry

9th January, 2017

Table Of Contents

Introduction

This document will describe the various elements of the system like the backend technologies used, the structure of the database, design class diagram, sequence diagrams and the graphical user interface of the system.

In the case of the Database Structure section, it will be presented in an unusual way because there is no one agreed method of showing a document database structure diagram.

The last section of the document will focus on the changes applied to the design of this project over the three iterations.

Backend

The server hosting the ExamIT system is an Arch Linux server, although any Linux server would suffice. All of the necessary packages and libraries were available in the Linux distribution package repository. Because of this, setting up the necessary dependencies and packages is straightforward.

OpenCV needs to be installed from the package repository. Any existing version will be working as the code is version proof with version checks and alternative steps.

MongoDB needs to be installed from the package repository. Version 3.4 is used with this project.

Python 3.6 was chosen as the development language because of its improvements like the print function, integer division, unicode strings by default. Python 3.6 comes pre-installed with the Arch Linux distribution.

For the simplicity of installing Python packages necessary, pip is installed from the package repository. Afterwards, all the Python packages can be installed using the pip install command.

The packages which need to be installed using pip are:

- Flask - Python web micro-framework
- Cv2 (OpenCV binding)
- numPy - scientific computing package for Python
- Wand - Image manipulation binding to ImageMagick (preinstalled on Linux systems)
- pyFPDF - A popular PDF manipulation library ported from PHP

The remaining packages used come preinstalled with Python and are listed with the code in the Technical Manual.

Frontend

For the purpose of the frontend the following technologies are used:

1. Bower was used to fetch all the necessary dependencies.
2. The user interface is designed using Bootstrap elements, CSS and HTML code.
3. The technologies used for dynamic content are:
    a. jQuery (dynamic tables, onclick events ie.)
    b. Jinja2 templating language

Some third party plugin solutions are used:

- Datatables - visually attractive table plugin for jQuery [J1]
- Fontawesome - used for attractive fonts [F1]

Database Structure

Because the project does not use a standard SQL database design, but instead has the document form, there is no way of displaying the database structure in the same format.

Therefore, the author will present the database design by explaining it in detail.

Firstly, the database consists of the following collections:

1. Categories - categories for the questions
2. Questions:
   a. Most important collection
   b. Contain a list of answers, answer key
   c. Tests are made up of questions
3. Tests:
   a. Test metadata
   b. Questions
4. Results:
   a. Which test is the result from
   b. Score

Each collection consists of documents, if a collection is ‘Questions’ then each document inside of it is a ‘Question’ document.

Each document has an automatically generated id element, ObjectID.

Example: "_id" : ObjectId("58a5ac9735f23f071e50d14e"

Category

```
{
  "_id": "ObjectId(\"58a5ac9735f23f071e50d14e\")",
  "CATEGORY": "physics",
  "CREATED": "Thu, 16 Feb 2017 13:43:51 GMT"
}
```

Categories are used to draw questions for a test. For example, if the user selects the ‘Physics’ category, and a number of questions (5, 10, 15…), that amount of questions of the ‘Physics’ category will be randomly drawn from the available questions.

Fields: _id, CATEGORY, CREATED

Question

```
{
  "_id": "ObjectId(\"58a5be0335f23f071e50d542\")",
  "CATEGORY": "electronics",
  "QUESTION": "Radix of binary number system is _____?",
  "ANSWERS": [
    "0",
    "1",
    "2",
    "A & B",
    "8"
  ],
  "KEY": "C",
  "CREATED": "Thu, 16 Feb 2017 14:58:11 GMT"
}
```

Questions are the core element of a test. Each test is made out of questions. Each question has a question category to which it belongs, question body and a list of answers along with the answer key.

FIelds: _id, CATEGORY, QUESTION, ANSWERS, KEY, CREATED

Test

```
{
  "_id": "ObjectId(\"58aeb93df2afa3ae1495df4b\")",
  "CATEGORY": "electronics",
  "CREATED": "Thu, 23 Feb 2017 10:27:49 GMT",
  "LECTURER": "John Marks",
  "MODULE": "Advanced Electronics",
  "QUESTCNT": 5,
  "QUESTIONS": [
    {
      "ANSWERS": [
        "AB + CD",
        "AB(CD)",
        "(A + B)(C + D)",
        "(A)B(CD)",
        "A^2"
      ],
      "CATEGORY": "electronics",
      "CREATED": "Thu, 16 Feb 2017 14:59:25 GMT",
      "KEY": "A",
      "QUESTION": "Which of the following expressions is in the sum-of-products (SOP) form?",
      "_id": "58a5be4d35f23f071e50d554"
    }, //followed by multiples of ‘Question’
  ],
  "TIME_ALLOWED": "25 minutes",
  "TITLE": "Electronics Basics Assessment"
}
```

Each test contains its category, which is the same document used for categorising questions as all the questions in the test must be of the same category in the current design. It also contains test metadata like the lecturer, module, amount of questions and the question documents.

The question documents are not referred to from the test, they are directly copied in their full form. This is because, if the questions were referred to from the test, and if a question was deleted from the collection of questions that is being used in a test, then the test would be broken.

This causes the database to essentially store the same information multiple times but is necessary for the correct functioning of the system. Also, because the data is of BSON form, it does not in fact use a lot of storage space.

Fields: _id, CATEGORY, CREATED, LECTURER, MODULE, QUESTCNT, QUESTIONS, TIME_ALLOWED, TITLE

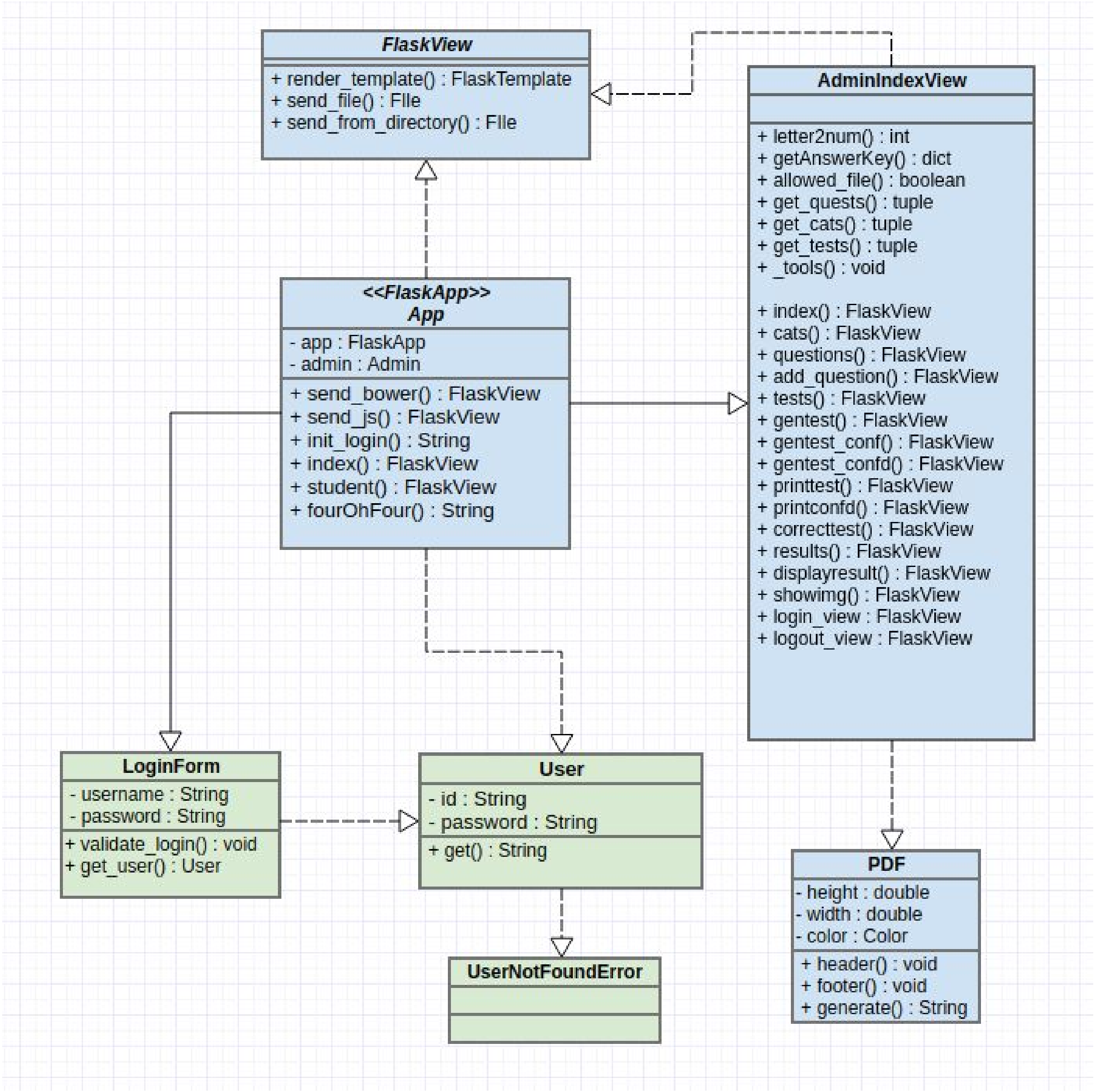
Result

```
{
  "_id": "ObjectId(\"58e160753669e80543cda562\")",
  "TEST": "Electronics Christmas Assessment",
  "SCORE": 100,
  "CORRECT": 15,
  "AMOUNT": 15,
  "FLAG": false,
  "CREATED": "Sun, 02 Apr 2017 21:35:01 GMT",
  "HREF": "58e160753669e80543cda562.png"
}
```

The result document is created when a test is corrected. It contains information about the test it is a result for, the score achieved by the student, number of correct answers and the number of questions. It also contains a flag boolean field which informs if the test was void or corrected properly. The test becomes void if a wrong answer sheet has been supplied or if the student broke the correction algorithm by marking the test sheet in places not meant for that purpose. The 'HREF' field is a filename of the annotated by the algorithm form of the test scanned, which serves the purpose of the lecturer being able to view and correct the test personally if the test is void or if the lecturer suspects the unlikely algorithm mistakes.

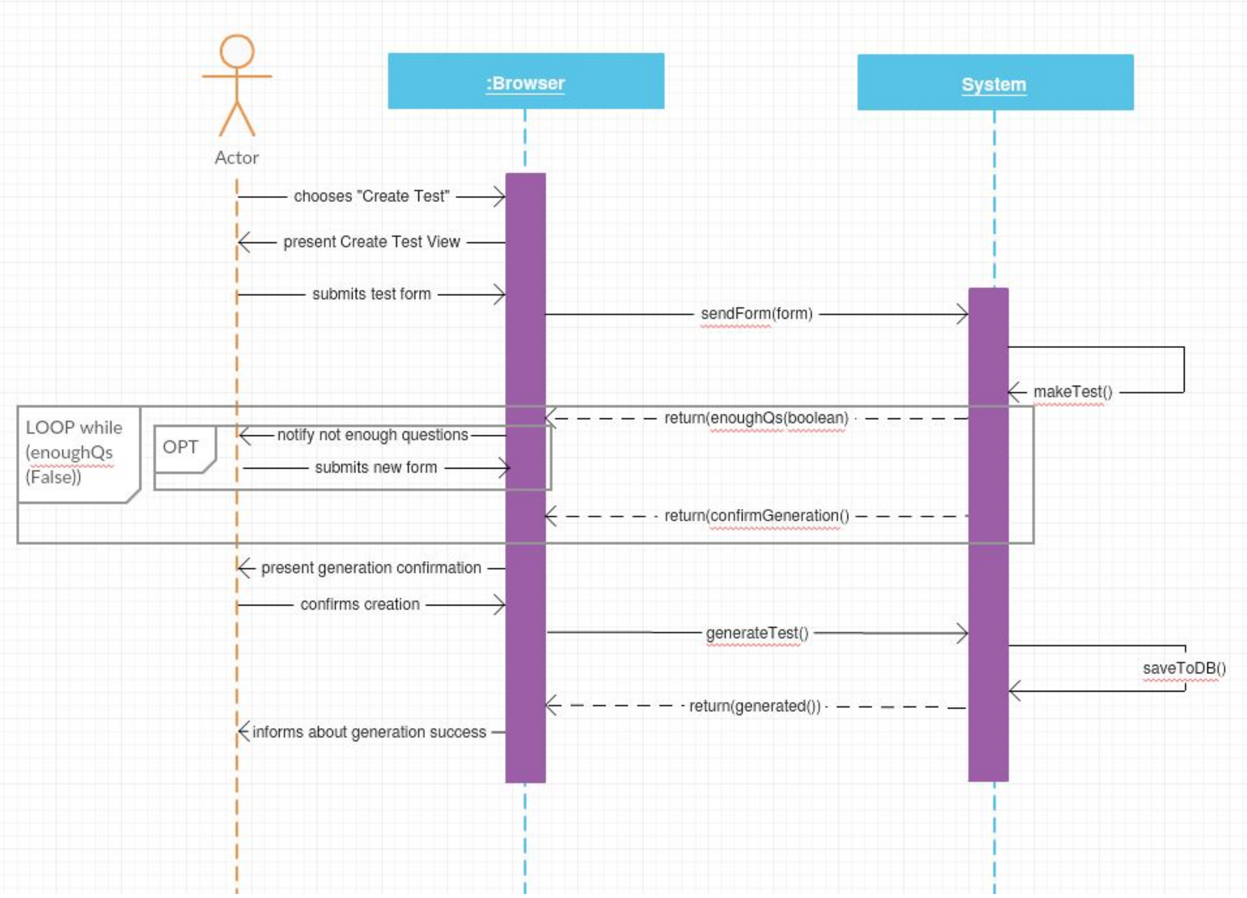
Fields: _id, TEST, SCORE, CORRECT, AMOUNT, FLAG, CREATED, HREF

Design Class Diagram

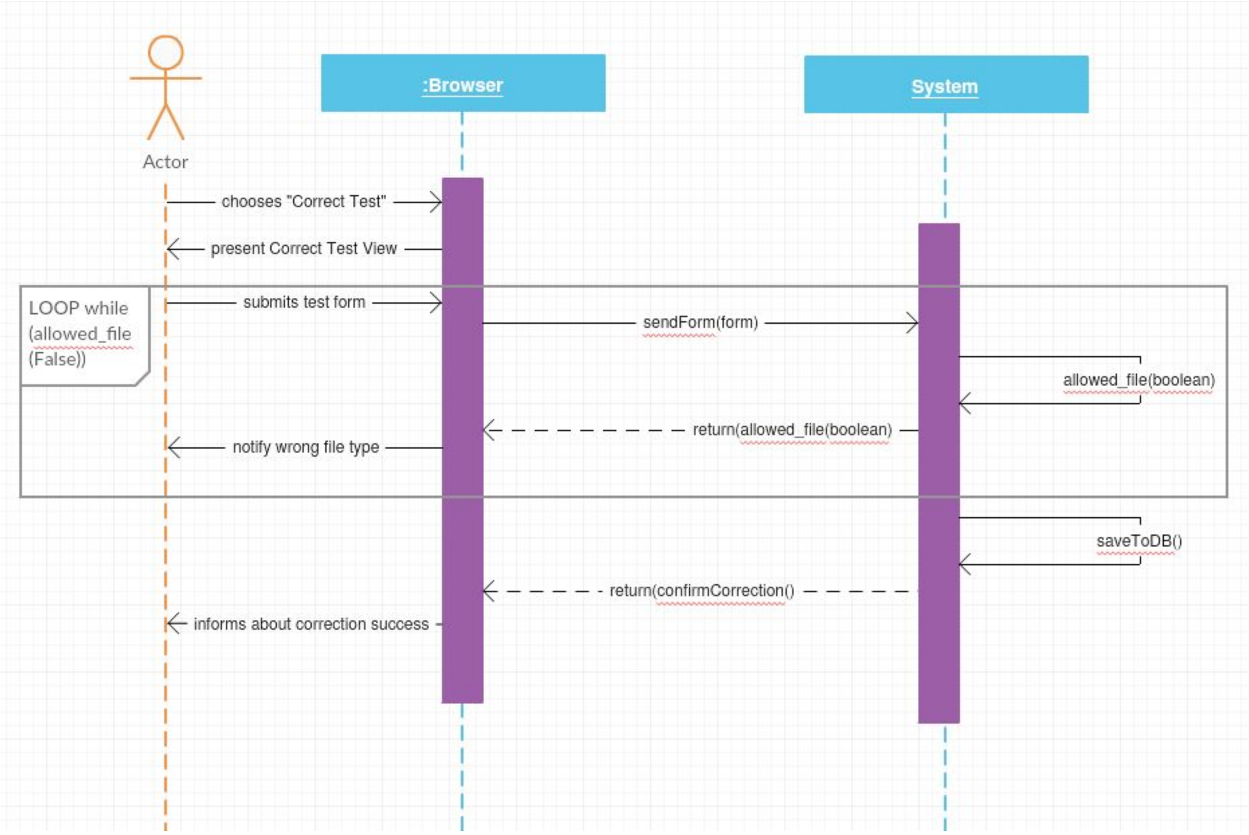


Sequence Diagrams

Test Creation



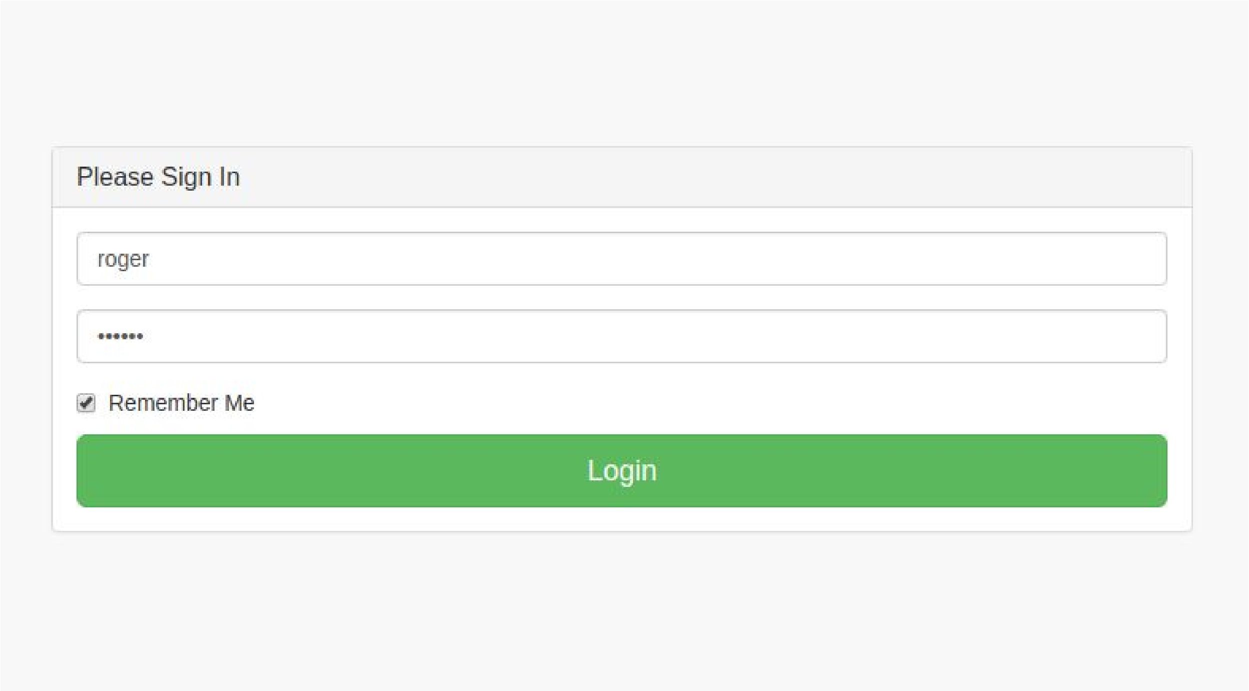
Test Correction


:Browser
System
Actor
chooses "Correct Test"
present Correct Test View
LOOP while (allowed_file (False))
submits test form
sendForm(form)
allowed_file(boolean)
return(allowed_file(boolean)
notify wrong file type
saveToDB()
return(confirmCorrection()
informs about correction success

GUI

Login Screen

A simple login screen, allowing lecturers to login and use the system



Please Sign In

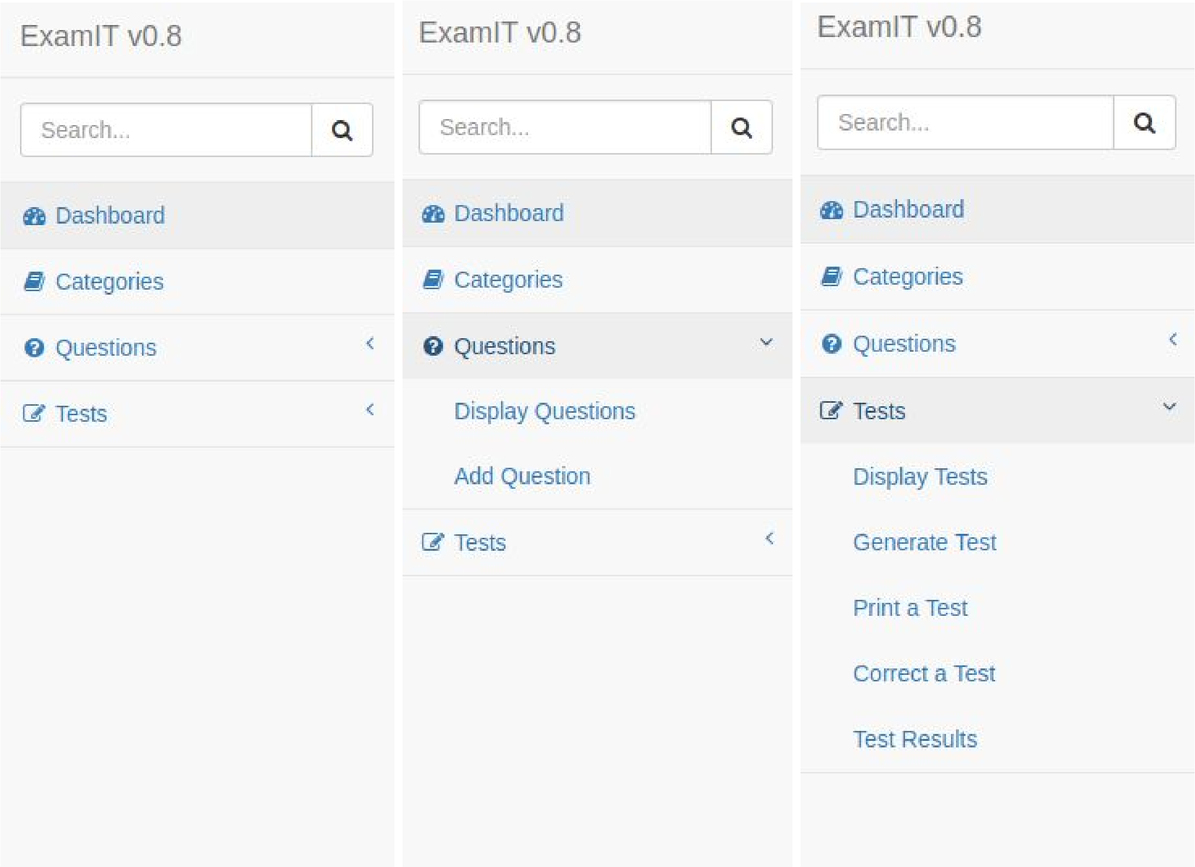
roger

••••••

☑ Remember Me

Menu

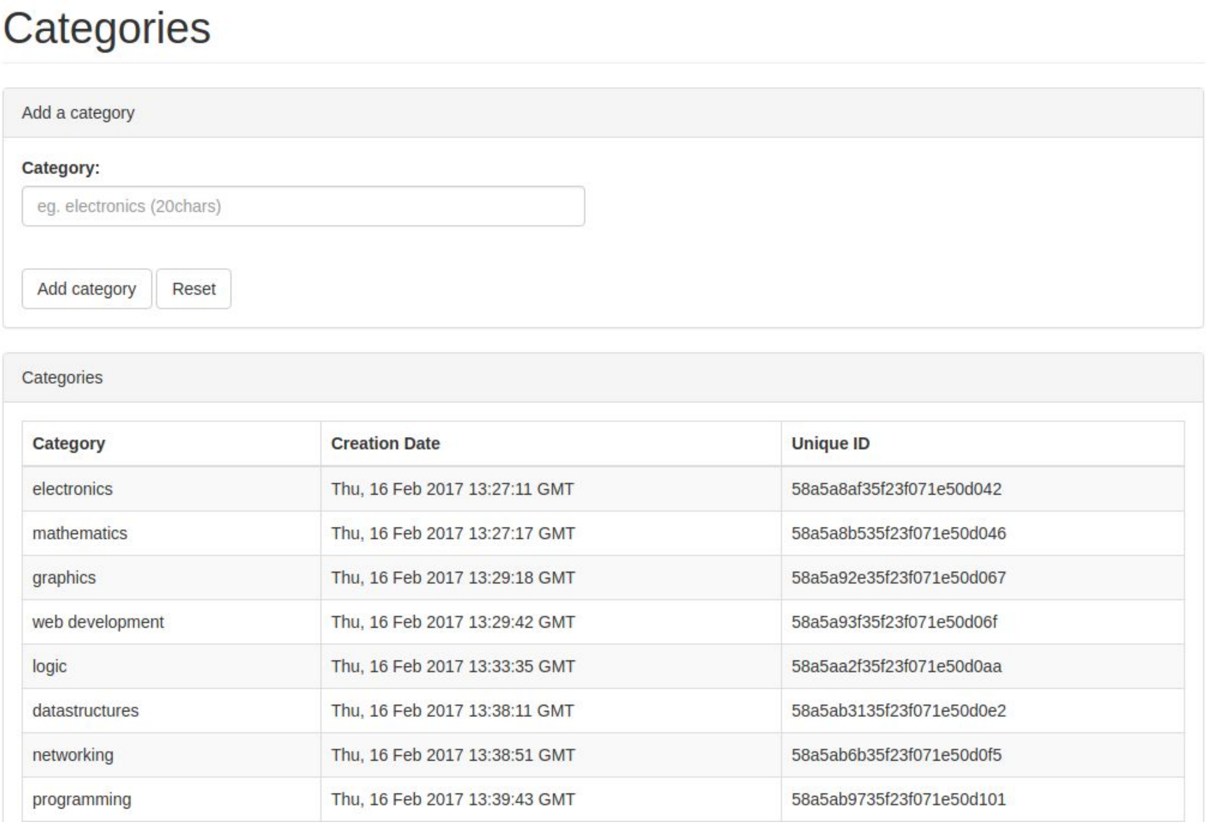
The menu is always available in the sidebar and allows for the navigation of the web application



Everything nested | Questions exposed | Tests Exposed

Categories

Allow for creating categories which are later used for categorising questions and tests



Categories

Add a category

Category:

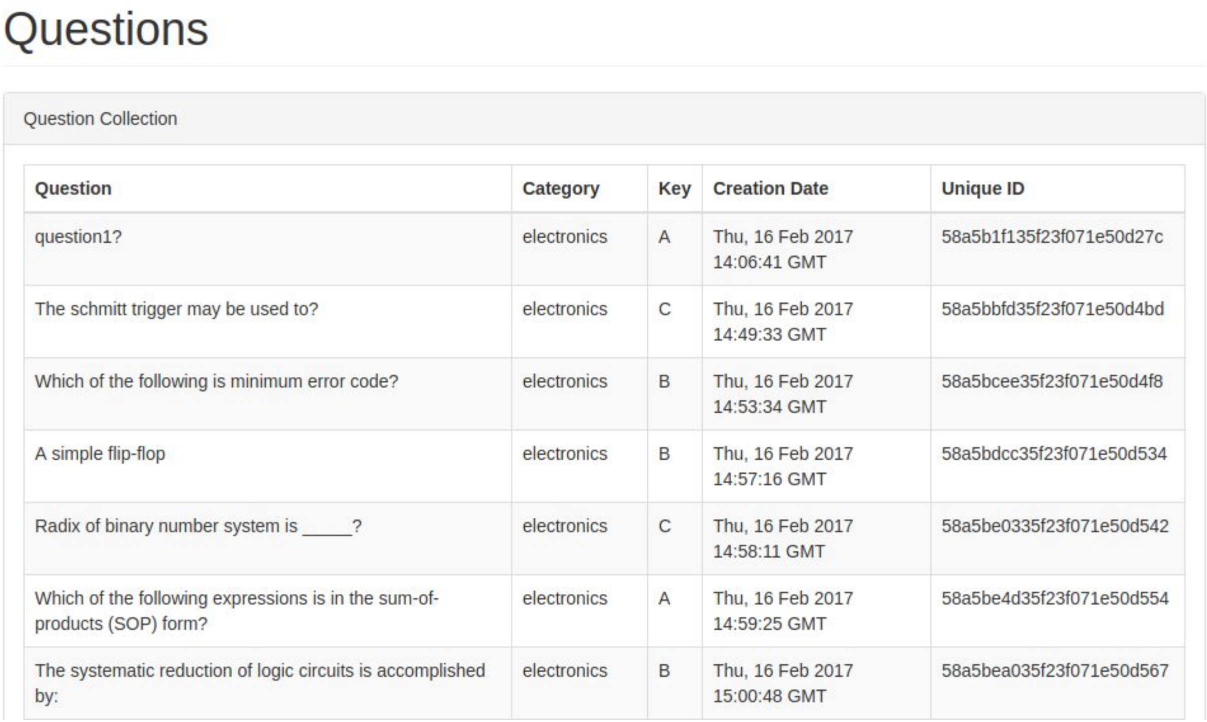
eg. electronics (20chars)

Categories

Category	Creation Date	Unique ID
electronics	Thu, 16 Feb 2017 13:27:11 GMT	58a5a8af35f23f071e50d042
mathematics	Thu, 16 Feb 2017 13:27:17 GMT	58a5a8b535f23f071e50d046
graphics	Thu, 16 Feb 2017 13:29:18 GMT	58a5a92e35f23f071e50d067
web development	Thu, 16 Feb 2017 13:29:42 GMT	58a5a93f35f23f071e50d06f
logic	Thu, 16 Feb 2017 13:33:35 GMT	58a5aa2f35f23f071e50d0aa
datastructures	Thu, 16 Feb 2017 13:38:11 GMT	58a5ab3135f23f071e50d0e2
networking	Thu, 16 Feb 2017 13:38:51 GMT	58a5ab6b35f23f071e50d0f5
programming	Thu, 16 Feb 2017 13:39:43 GMT	58a5ab9735f23f071e50d101

Display Questions

Displays a table of all the questions previously generated in the system



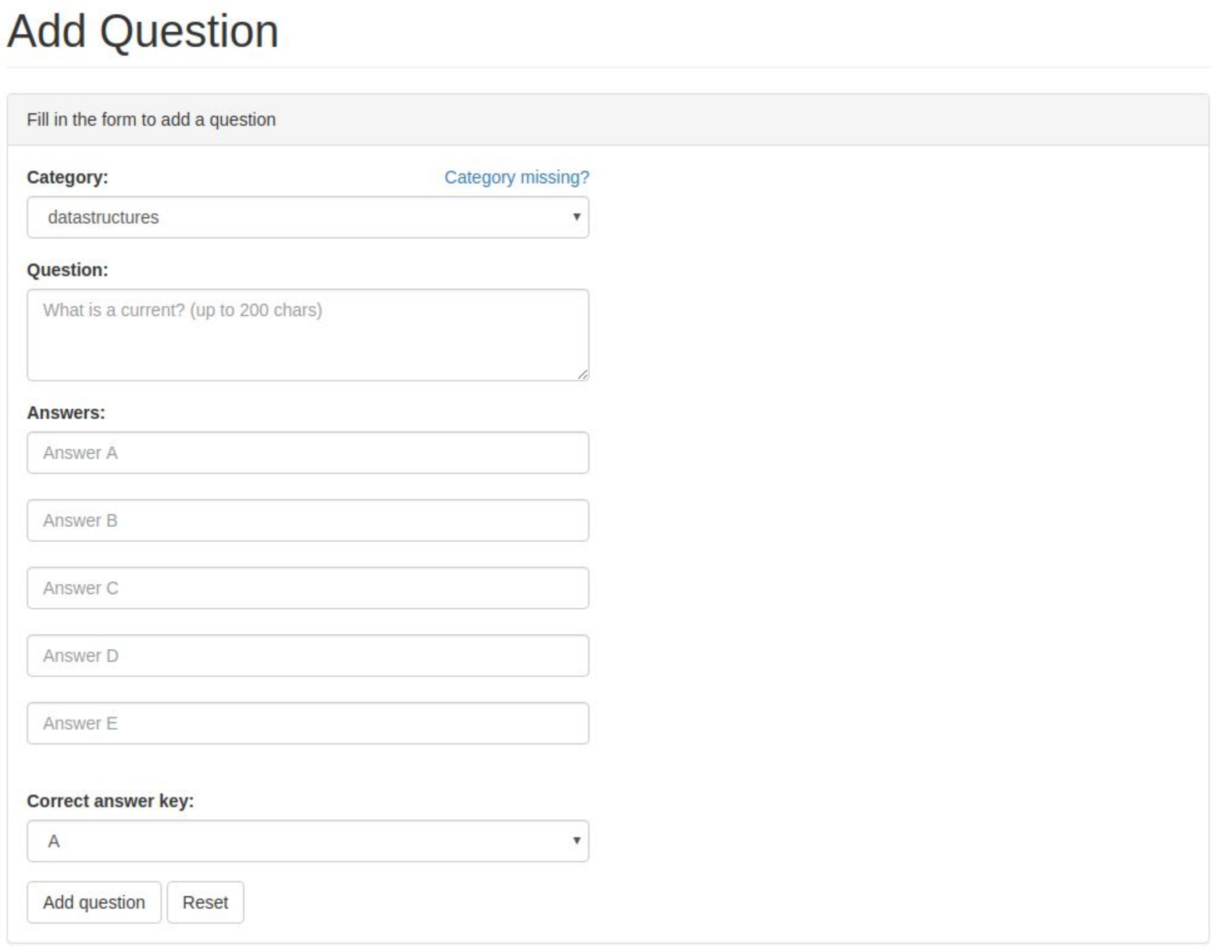
Questions

Question Collection

Question	Category	Key	Creation Date	Unique ID
question1?	electronics	A	Thu, 16 Feb 2017 14:06:41 GMT	58a5b1f135f23f071e50d27c
The schmitt trigger may be used to?	electronics	C	Thu, 16 Feb 2017 14:49:33 GMT	58a5bbfd35f23f071e50d4bd
Which of the following is minimum error code?	electronics	B	Thu, 16 Feb 2017 14:53:34 GMT	58a5bcee35f23f071e50d4f8
A simple flip-flop	electronics	B	Thu, 16 Feb 2017 14:57:16 GMT	58a5bdcc35f23f071e50d534
Radix of binary number system is _____?	electronics	C	Thu, 16 Feb 2017 14:58:11 GMT	58a5be0335f23f071e50d542
Which of the following expressions is in the sum-of-products (SOP) form?	electronics	A	Thu, 16 Feb 2017 14:59:25 GMT	58a5be4d35f23f071e50d554
The systematic reduction of logic circuits is accomplished by:	electronics	B	Thu, 16 Feb 2017 15:00:48 GMT	58a5bea035f23f071e50d567

Add Question

Allow for adding questions to the collection of questions



Add Question

Fill in the form to add a question

Category: datastructures

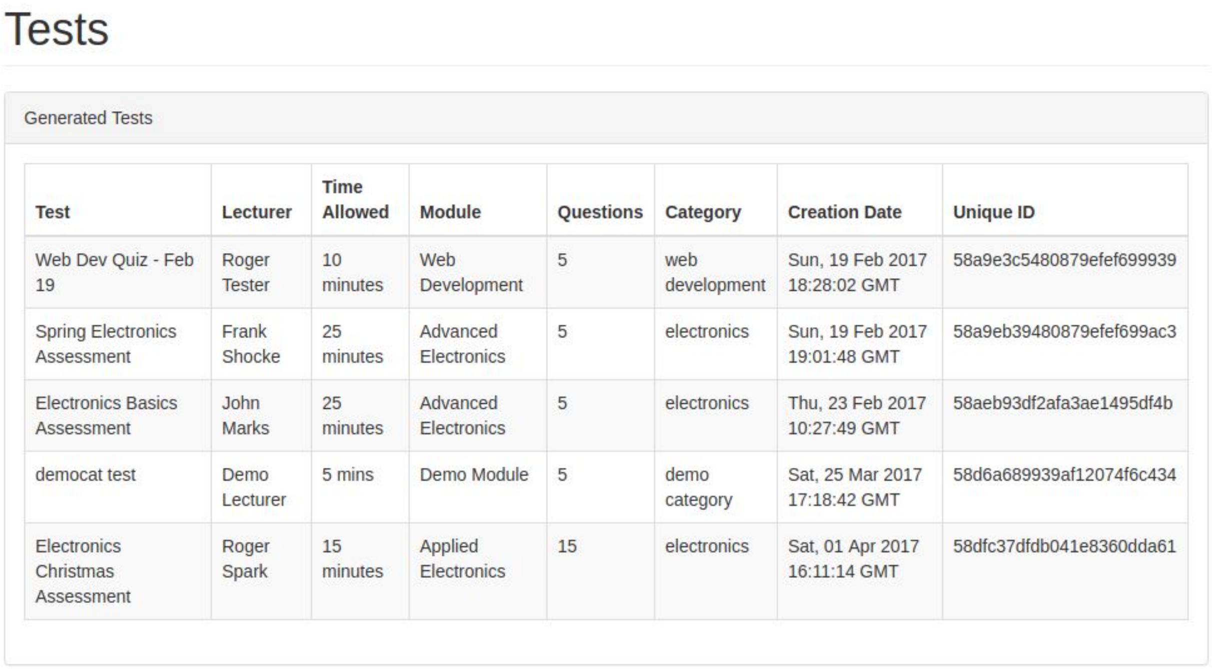
Question:

Answers:

Correct answer key: A

Display Tests

Displays all the available tests previously generated in the system



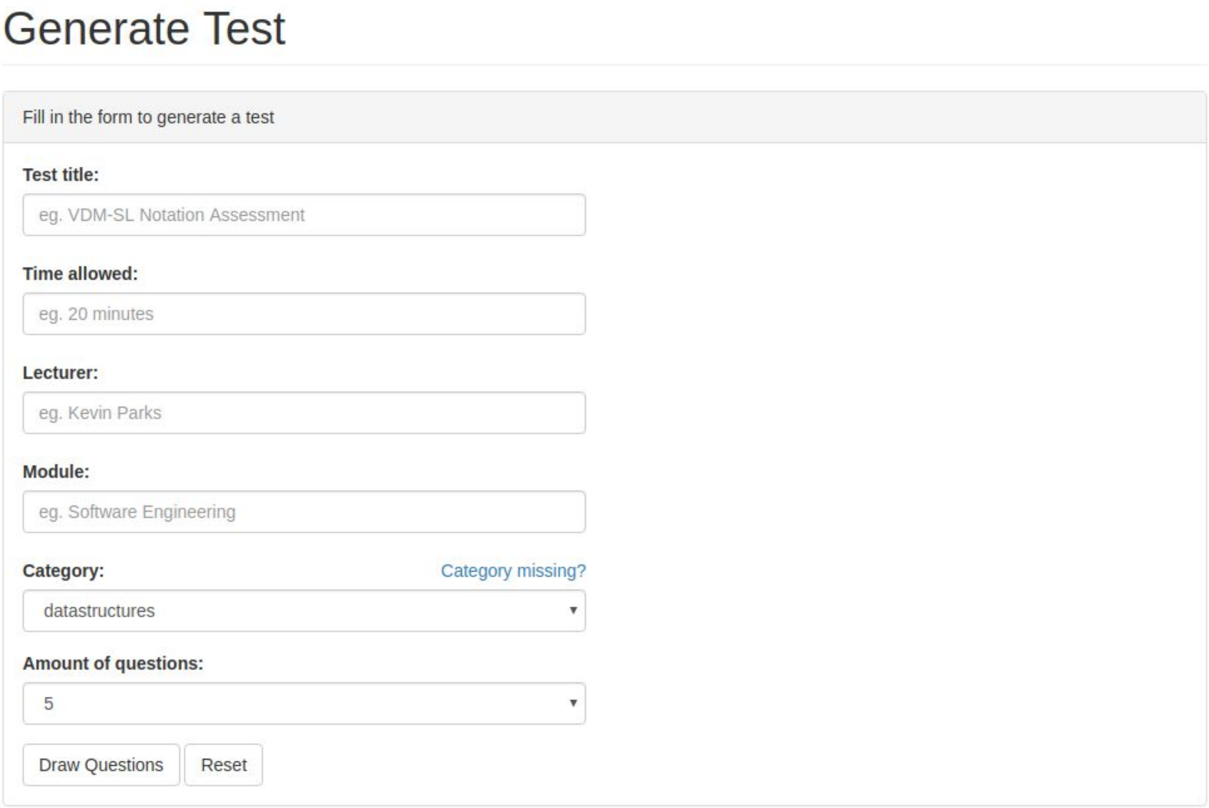
Tests

Generated Tests

Test	Lecturer	Time Allowed	Module	Questions	Category	Creation Date	Unique ID
Web Dev Quiz - Feb 19	Roger Tester	10 minutes	Web Development	5	web development	Sun, 19 Feb 2017 18:28:02 GMT	58a9e3c5480879efef699939
Spring Electronics Assessment	Frank Shocke	25 minutes	Advanced Electronics	5	electronics	Sun, 19 Feb 2017 19:01:48 GMT	58a9eb39480879efef699ac3
Electronics Basics Assessment	John Marks	25 minutes	Advanced Electronics	5	electronics	Thu, 23 Feb 2017 10:27:49 GMT	58aeb93df2afa3ae1495df4b
democat test	Demo Lecturer	5 mins	Demo Module	5	demo category	Sat, 25 Mar 2017 17:18:42 GMT	58d6a689939af12074f6c434
Electronics Christmas Assessment	Roger Spark	15 minutes	Applied Electronics	15	electronics	Sat, 01 Apr 2017 16:11:14 GMT	58dfc37dfdb041e8360dda61

Generate Test

Allows for the generation of tests



Generate Test

Fill in the form to generate a test

Test title:

eg. VDM-SL Notation Assessment

Time allowed:

eg. 20 minutes

Lecturer:

eg. Kevin Parks

Module:

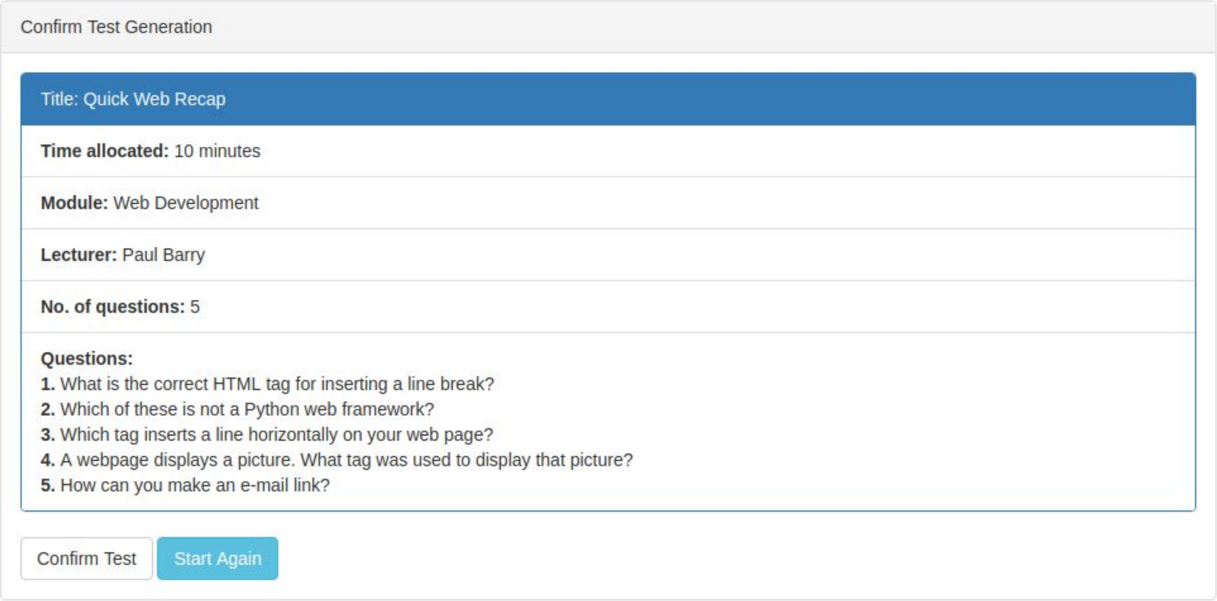
eg. Software Engineering

Category: Category missing?

datastructures ▾

Amount of questions:

5 ▾



Confirm Test Generation

Title: Quick Web Recap

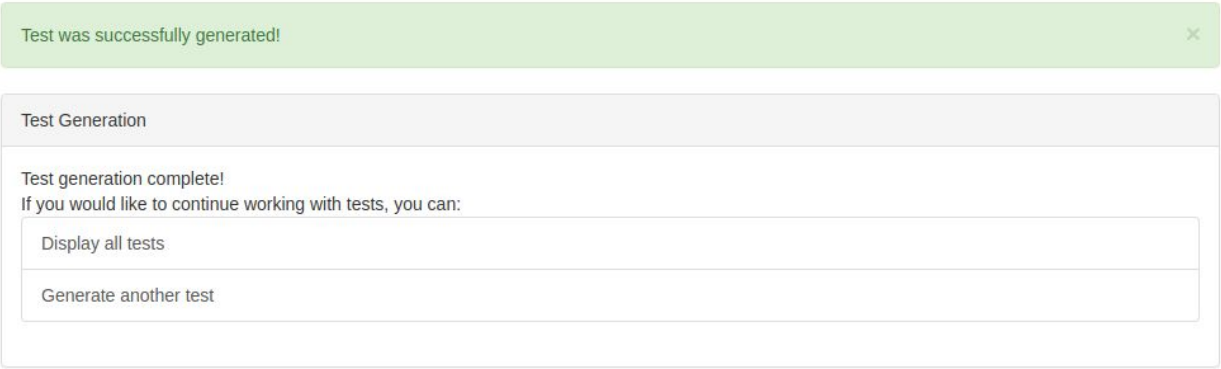
Time allocated: 10 minutes

Module: Web Development

Lecturer: Paul Barry

No. of questions: 5

Questions:
1. What is the correct HTML tag for inserting a line break?
2. Which of these is not a Python web framework?
3. Which tag inserts a line horizontally on your web page?
4. A webpage displays a picture. What tag was used to display that picture?
5. How can you make an e-mail link?



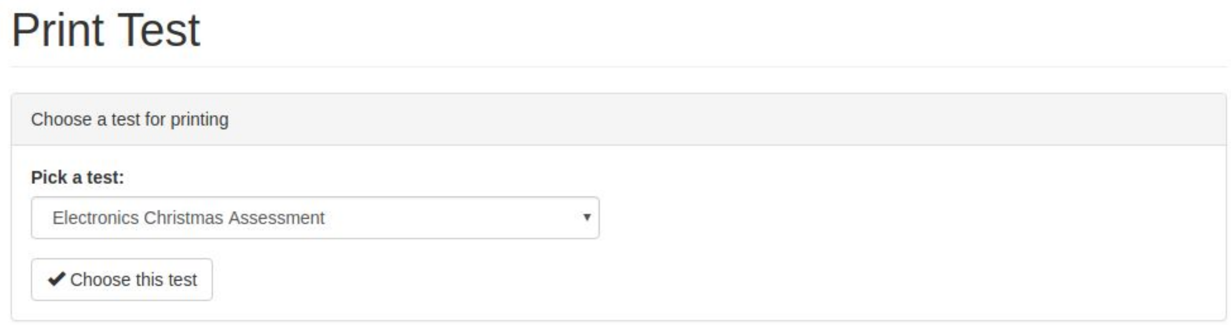
Test was successfully generated!

Test Generation

Test generation complete!
If you would like to continue working with tests, you can:

Print Test

Used for printing the existing tests

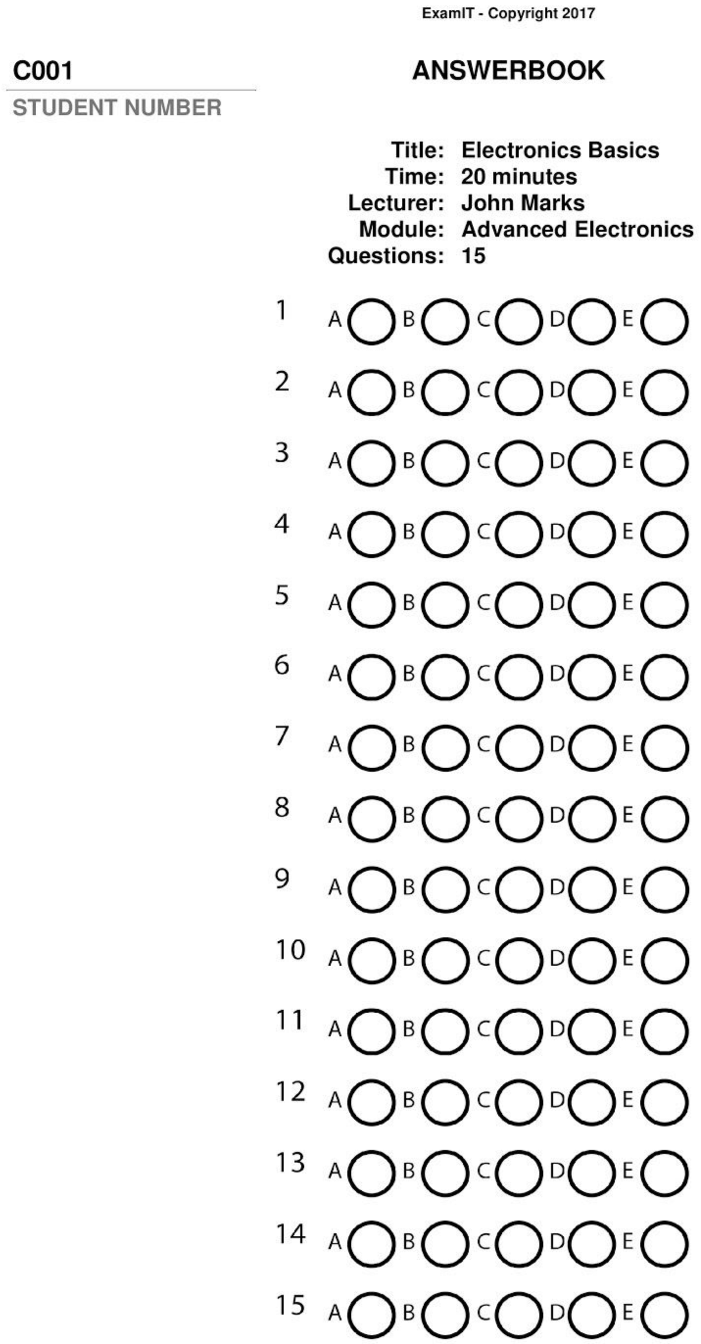


Print Test

Choose a test for printing

Pick a test: Electronics Christmas Assessment

Printed Test Example



ExamIT - Copyright 2017

C001

STUDENT NUMBER

ANSWERBOOK

Title:	Electronics Basics
Time:	20 minutes
Lecturer:	John Marks
Module:	Advanced Electronics
Questions:	15

1 A ○ B ○ C ○ D ○ E ○

2 A ○ B ○ C ○ D ○ E ○

3 A ○ B ○ C ○ D ○ E ○

4 A ○ B ○ C ○ D ○ E ○

5 A ○ B ○ C ○ D ○ E ○

6 A ○ B ○ C ○ D ○ E ○

7 A ○ B ○ C ○ D ○ E ○

8 A ○ B ○ C ○ D ○ E ○

9 A ○ B ○ C ○ D ○ E ○

10 A ○ B ○ C ○ D ○ E ○

11 A ○ B ○ C ○ D ○ E ○

12 A ○ B ○ C ○ D ○ E ○

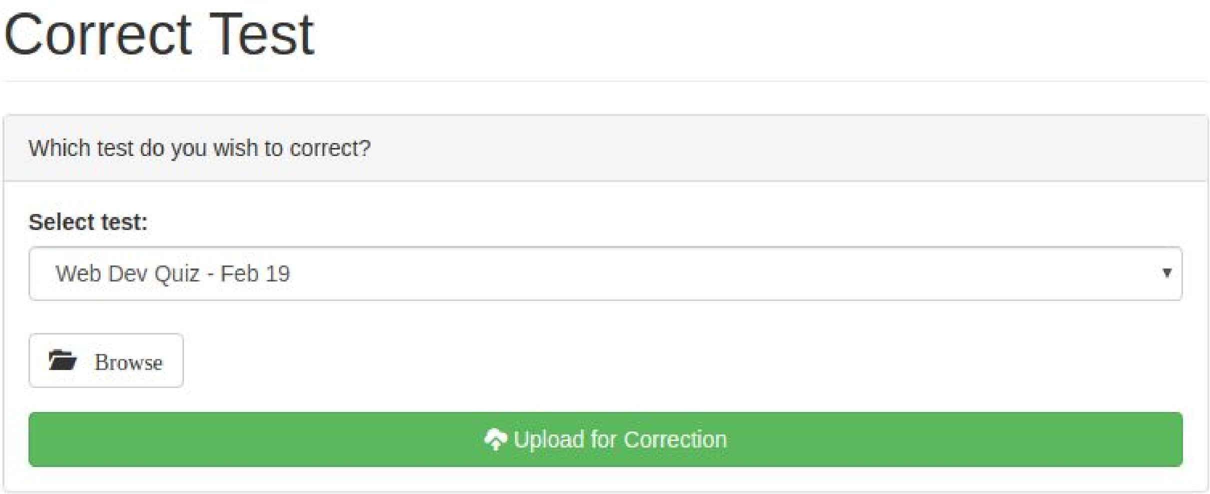
13 A ○ B ○ C ○ D ○ E ○

14 A ○ B ○ C ○ D ○ E ○

15 A ○ B ○ C ○ D ○ E ○

Correct Test

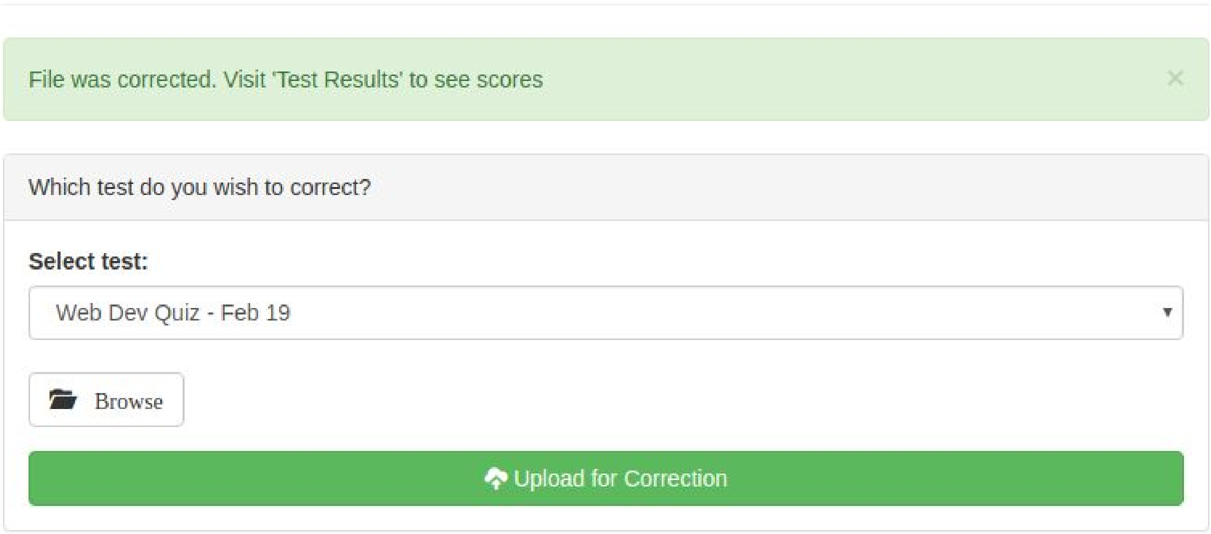
Used for correcting scanned tests



Correct Test

Which test do you wish to correct?

Select test: Web Dev Quiz - Feb 19



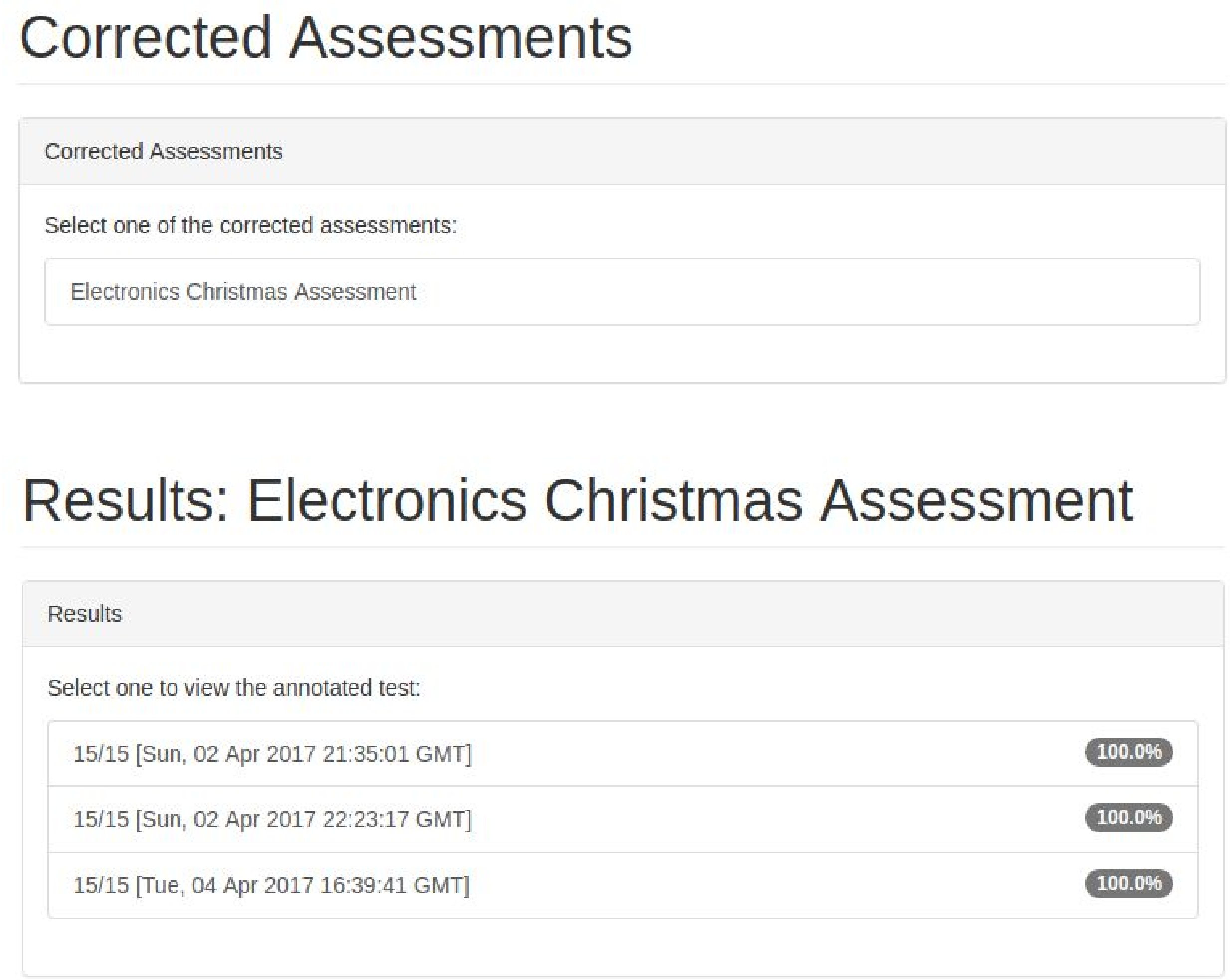
File was corrected. Visit 'Test Results' to see scores ×

Which test do you wish to correct?

Select test: Web Dev Quiz - Feb 19

Test Results

Display the test results



Corrected Assessments

Corrected Assessments

Select one of the corrected assessments:

Electronics Christmas Assessment

Results: Electronics Christmas Assessment

Results

Select one to view the annotated test:

15/15 [Sun, 02 Apr 2017 21:35:01 GMT] 100.0%

15/15 [Sun, 02 Apr 2017 22:23:17 GMT] 100.0%

15/15 [Tue, 04 Apr 2017 16:39:41 GMT] 100.0%

Design Over Iterations

Iteration 1 (24 Oct - 12 Dec)

- During the first iteration, the design of the project was completely conceptual
- Most of the iteration was spent on research and learning the OpenCV and numPy libraries in order to be able to create the correction algorithm
- The author had decided on the Python development language and Flask web framework towards the end of the iteration
- Initial version of the correction algorithm was created and proved working on simple test cases

Iteration 2 (9 Jan - 20 Feb)

- The design was essentially only created in this iteration
- The database technologies were compared and MongoDB chosen as the document schema fitted well with the data the project would generate
- The database design was created
- PDF manipulation libraries were tested and prototype tests were generated with them, pyPDF library was chosen
- The test layout was created
- Wand library was selected for transforming .pdf files to .png for use with OpenCV
- The Flask app was built with part of he views

Iteration 3 (27 Feb - 5 Apr)

- The PDF generation, test correction and Flask app were all functional but working separately
- The three parts of the system were put together
- Remaining missing views were added for the three parts
- Bootstrap was used to give the application an attractive UI
- The design from iteration 2 proved good enough, only slight adaptations were necessary:
  - The author had to alter the database model for tests (copying of the question documents as is instead of referring to them, mentioned in detail earlier in the document)
  - Some views had to be split into multiple views for better functionality and clarity:
    - Test Results were split into 2 views
    - Test Generation was split into 3 views to accommodate confirmation dialogues to ensure that they were generated correctly

References

[J1] Datatables (2017), Datatables Homepage, [online], available:
https://datatables.net/

[accessed 9 February, 2017]

[F1] Fontawesome (2017), Fontawesome Homepage, [online], available:
https://datatables.net/

[accessed 9 February, 2017]