KING'S College LONDON
University of London

6CCS3PRJ Final Year Orchestrator

Final Project Report

Author: Leonardo Ciocan

Supervisor: Jeroen Keppens

Student ID: 1308123

April 20, 2016

Abstract

In university settings, lecturers often teach classes of up to 200 students. Often they distribute papers to gauge the studentâĂŹs understanding of the current material being taught. The distribution, collection and analysis of these materials makes it hard for the lecturer to evaluate which parts of the material the students need help with. I have created a digital solution that helps the teacher get a better understanding of his students progress in real time.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Leonardo Ciocan

April 20, 2016

Acknowledgements

It is usual to thank those individuals who have provided particularly useful assistance, technical or otherwise, during
your project. Your supervisor will obviously be pleased to be acknowledged as he or she will have invested quite a lot of time overseeing your progress.

Contents

Chapter 1

Introduction

1.0.1 Summary

There is quite a sizeable market that would benefit from this application, there are 23729 universities around the world; a high percentage of them have Computer Science departments. Competitors Some companies have products that aim to facilitate teacher âĂŞ student interaction but I believe they are rather incomplete and further more none appear to provide tools specifically for computer science students. In other words, they provide generic tools for letting students answer questions but that is not tailor for computer science specifically. Motivation The main scope of my project was to create an easily extensible platform that may be useful for teachers specifically teaching computer science as this subject in particular requires some types of question that are more complex than simple text or choices. In other words, letting users write and execute code on my platform was a core goal. Executing arbitrary code is a dangerous so security is an important part of this system. This project is a great way to experiment with visualisation of data and how digital solutions can enrich previous analogue methods , it also allowed me to work with technologies I had not worked before and build a full product which enriched my knowledge of security , containers , server and backend technologies.

1.0.2 Platform

I have decided to build this project for the web instead of a mobile application for a number of reasons. Since this is meant to be used by a lot of students, it is a requirement that it should be as accessible as possible. Statistically speaking, a web app can reach the most people, especially account for the fact that computing students are the main target audience and they are likely to have laptops with them. The main disadvantage of making mobile apps is that to build

a good native app would mean to focus on one platform (as cross platform solutions are not up to the task for the scope of the project) which would exclude students from the process. The web is a free, universal platform and so it is the perfect medium for this project. Ideally, fully native mobile apps are further down the roadmap as they provide a better experience and further expand the pool of users that are able to use the application. The backend part of this project has been built in such a way to allow for easy expansion to other platforms. In fact, the web app could be considered just a consumer of the backend and not being renderer by the server such that another front-end could be used instead without any friction.

1.0.3 Objectives

Because the main target audience is made up of computer savvy student and lecturers, it was possible to take a few liberties with the platform. For example the questions can be written in Markdown, which is something computer scientists are already familiar with from websites such as StackOverflow and Github. Nevertheless, I aim to make a project that requires minimal guidance as the user interface will be built to be easy and intuitive to operate.

The basic requirements for this project are:

1. Let teacher create sheets made of a variety of question types
2. Allow question types to include rich formatting such as tables and code
3. Allow students to subscribe to a lecture such that they may have access to those sheets
4. Allow a teacher to create a class and invite students with a link
5. Let the teacher decide when a sheet should go âĂIJliveâĂİ and be visible by the students
6. Allow the teacher to attach model answers to each question
7. Let the teacher release model answers to the students so they may review their answers
8. Monitor the students progress and present it to the teacher in an anonymous way Allow users to write code to be executed for code questions

1.0.4 Report structure

The next section will be the background which will begin by exploring the keywords or concepts that are related to the project. It will also analyse and contrast current implementation that attempt to solve the problem described with the project's implementation.

Chapter 2

Background

2.0.1 Relevant concepts

Question Currently there are 3 types of questions:

- Multiple choice: The user is presented some choice and selects one
- Input: The user may input any piece of text, multiple solutions can exist
- Code: Users can write code to solve a posed question

The system is built in such a way that adding further questions is trivial and does not require breaking backwards compatibility or redesigning data structures used on the server.

Sheet The application revolves around Sheets. Analogue to a paper sheet handed in class, a sheet is a collection of questions of various types. Each sheet belongs to a lecture which has a teacher.

A teacher can create sheets by mix and matching any number of question types which enable quite a varied way to test their students.

Lecture A lecture is a collection of sheets. It has a teacher and students can subscribe to a lecture so they can have access to all the sheets. Dashboard An important part of this project is for the teacher to be able to monitor the student progress so that they may respond accordingly (for example, write some hints on the board). Each sheet has a dashboard which only the teacher can access. For each type of question, there is a different specialised user interface that is tailored to convey the progress of the students. A code question's dashboard widget will show the percentage of students who completed the question. An input questionâĂŹs

dashboard shows the percentage of completions, as well as a word cloud of popular words in the questions, this is a specialised control that visually conveys to the teacher common words that are being used. A choice questionâĂŹs dashboard will show the userâĂŹs completion, a bar chart for top first choices (which could help identify misleading questions amongst other things) and a transition matrix table, which shows the way students move from one answer to another.

The dashboard is also meant to be extensible, so any future question types could provide their own way of visualising student progress.

2.0.2 Differentiation from competitors

The project is meant to be an extensible platform that can be enriched with more types of question along the development process to increase the variety of sheets the teachers can create. This gives it more potential than competitorâĂŹs who over inflexible solutions that are tailor made for very specific types of questions. Furthermore, by focusing on computer science students we allow users to run arbitrary code on the platform, which while some websites allow that âĂŞ they provide that in a different setting such as code competitions âĂŞ which means they do not compete with us , as this project enabled the teacher to write a question that can be answered with code as part of a sheet along with other questions.

2.0.3 Security

Because the project allows users to input and run arbitrary code it is important to have proper security in place. There are three layers of security for running code:

- Users can only run non-native code , currently Python and Java
- All user code runs in secure Docker containers
- All containers are contained in a separate server

Because the code execution server and the database/main server are separate âĂŞ even if a malicious agent used a vulnerability to break out of the container, he may not access any information or maliciously disrupt server operations.

//security stats about safe code

Chapter 3

Report Body

The central part of the report usually consists of three or four chapters detailing the technical work undertaken during the project. **The structure of these chapters is highly project dependent**. They can reflect the chronological development of the project, e.g. design, implementation, experimentation, optimisation, evaluation, etc (although this is not always the best approach). However you choose to structure this part of the report, you should make it clear how you arrived at your chosen approach in preference to other alternatives. In terms of the software that you produce, you should describe and justify the design of your programs at some high level, e.g. using OMT, Z, VDL, etc., and you should document any interesting problems with, or features of, your implementation. Integration and testing are also important to discuss in some cases. You may include fragments of your source code in the main body of the report to illustrate points; the full source code is included in an appendix to your written report.

3.1 Section Heading

3.1.1 Subsection Heading

Chapter 4

Design & Specification

This section will present an abstract view of how the system works.

4.0.1 Use cases

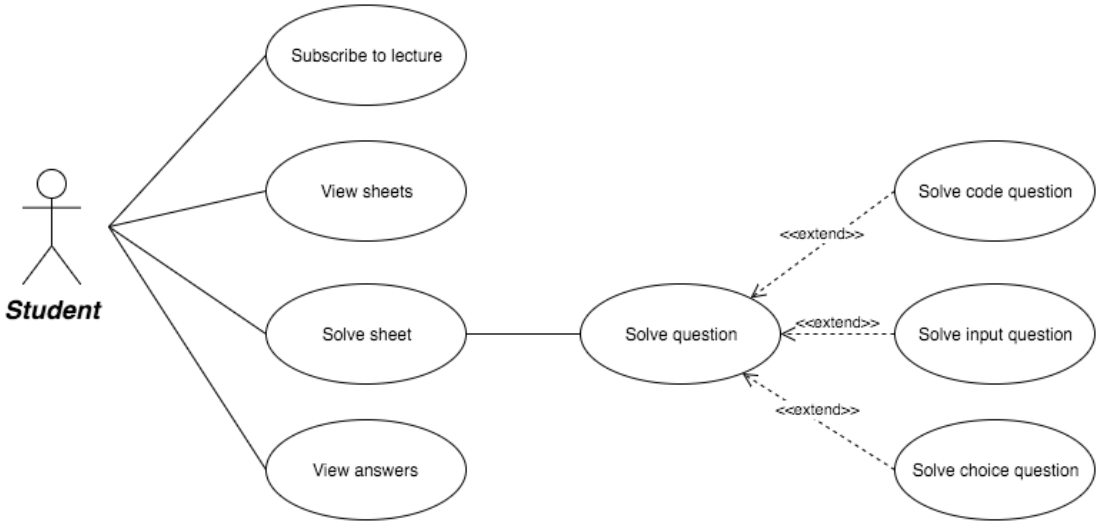

Figure 4.1: Use cases for students

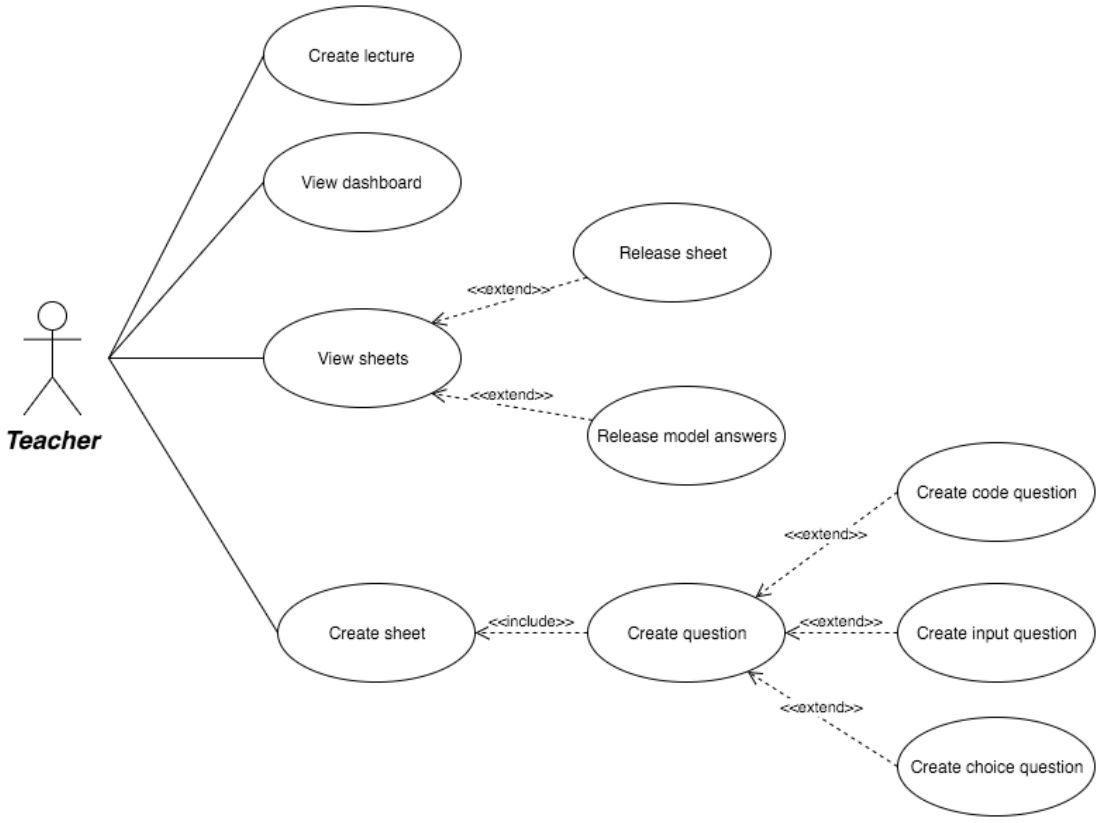

Figure 4.2: Use cases for teachers

4.0.2 System architecture

User interface Since the user's are assumed to be somewhat technically savvy , the application takes some liberties with having a limited amount of help and information banners since it works similarly to other website the target audience is familiar with. Overall the application is meant to have a flat design that looks good on any resolution. Each lecture has a colour chosen by the teacher and shows up throughout the user interface so the user knows which lecture they are looking at.

4.0.3 Third party libraries

React

A javascript library that allows the project to have isolated , reusable components that build up all of the user interface

Chromath

A javascript library that can manipulate colours. It is used within the dashboard to compute darker shades of the lecture color - this is useful for both the charts (each section of the chart is a different colour) and for the transition matrix (the higher the transition count , the darker the shade).

It is statically included in the project.

CodeMirror

A javascript library that provides an embeddable code editor. Used to let user's write their own code.

Markdown-it

A javascript library used to render Markdown , in both the title preview on the sheet creation page and the actual sheet that the user sees.

Highlight

A javascript library to highlight code syntax , used internally by Markdown-it

JQCloud

A javascript library used to render a word cloud in the dashboard for input questions.

Levenshtein-ffi

A ruby gem that allows fast calculation of Levenshtein distances for answer's in the input question's dashboard.

pg

A ruby gem for Postgres support

Devise

A ruby gem that simplifies authentication

ReactRails

Support for React in Rails applications

Typescript Rails

Automatic compilation of typescript files as part of the rails asset pipeline.

Chapter 5

Implementation

5.0.1 Unit testing

Unit testing was used to make sure changes and additions to the system did not break or alter any previous functionality.

Purpose:

1. To make sure existing functionality was not compromised
2. To ensure that the database settings worked as intended (defaults , uniqueness etc)
3. Components of the system all complied with security , such as preventing user's from editing other people's date or gaining access to data they are not allowed to see

5.1 Professional Issues

5.2 Evaluation

In this section we will explore whether the project has met the requirements that were set in the requirements section of this report as well as other general aspects of the software that can be evaluated.

5.2.1 Limitations

There were some things that were simplified or postponed for the sake of keeping in line with deadlines and achieving all the compulsory requirements that I have set. Thus the system does have some limitation (solutions to those limitations will be explored in the next section)

- Code question can be either Java or Python based. This is a limitation imposed to decrease the surface area for bugs that would arise from allowing more languages.
- Sheets cannot be edited after they are created. This is because changing the questions as the sheet is live could invalidate existing statistical data retrieved from users completing the question.
- Teacher cannot kick out or manage users. This is because currently the system aims to anonymise users so they may not feel discouraged from completing the questions
- The website is glitchy and may not work at all on some mobile devices. This is especially true for pages where the user may enter a core

5.2.2 Security

While the system is not meant to contain any sensitive data , it is important that the users of the system can be assured that their data is private. There are safeguards in place to ensure that a malicious agent may not successfully acquire information they are not entitled to , this functionality is unit tested to ensure it stays this way across releases of the software. Another layer of security is in the system that executes the student's arbitrary code. The code runs in a docker container and can only run within either the java virtual machine or the python interpreter. Not allowing low level code to run discourages any traditional security holes such as buffer overflows. Furthermore , even if someone manages to execute code such that they break out of the language runtime and the docker container - the containers run on a server separate from the main server. Thus they cannot access the database or compromise the system.

5.2.3 Overall

The main purpose of the system was to provide teacher's a way to handle handing out sheets digitally and analyse the result easily in real time. It was built in such a way that not only fulfilled this requirement but also provides a platform that is easily extendable such that more question types can be added without having to modify the existing infrastructure.

Chapter 6

Professional and Ethical Issues

Either in a seperate section or throughout the report demonstrate that you are aware of the **Code of Conduct & Code of Good Practice** issued by the British Computer Society and have applied their principles, where appropriate, as you carried out your project.

6.1 Section Heading

Chapter 7

Results/Evaluation

7.1 Software Testing

7.2 Section Heading

Chapter 8

Conclusion and Future Work

Future work

There are a number of ways to improve the project , the system is implemented in such a way that adding more functionality can be done seamlessly without breaking or having to alter any major components of the system.

Ideas for improvements:

- More question types. For example questions that revolve around web development (such as asking the user to create a html/css layout from an image) or unix command line tools (asking the user to complete some command line workflow). This should be easy to implement as the database schema for questions is very flexible and does not make any assumptions about the type of data a question expresses
- Better user management for the teacher , allowing them to kick , filter and analyse student progress. This would involve adding a lot more UI components and possibly other pages
- Allowing teacher to edit the title/body of questions after the sheet is created. Easy to implement the technical aspect of it , however it would not fit within any current place in the UI and would require an additional page
- Password enabled lectures , or alternatively geo-locked lectures. Fairly simple to implement but needs to have extra logic in place to deal with changing the password and so on
- Mobile apps for Android and iOS to expand the pool of students who have a device that can use the project. Takes time but overall the backend would not have to be changed

much as it already exposes most functionality via a simple REST API

References

Appendix A

Extra Information

A.1 Tables, proofs, graphs, test cases, ...

The appendices contain information that is peripheral to the main body of the report. Information typically included in the Appendix are things like tables, proofs, graphs, test cases or any other material that would break up the theme of the text if it appeared in the body of the report. It is necessary to include your source code listings in an appendix that is separate from the body of your written report (see the information on Program Listings below).

Appendix B

User Guide

B.1 Instructions

You must provide an adequate user guide for your software. The guide should provide easily understood instructions on how to use your software. A particularly useful approach is to treat the user guide as a walk-through of a typical session, or set of sessions, which collectively display all of the features of your package. Technical details of how the package works are rarely required. Keep the guide concise and simple. The extensive use of diagrams, illustrating the package in action, can often be particularly helpful. The user guide is sometimes included as a chapter in the main body of the report, but is often better included in an appendix to the main report.

Appendix C

Source Code

C.1 Instructions

Complete source code listings must be submitted as an appendix to the report. The project source codes are usually spread out over several files/units. You should try to help the reader to navigate through your source code by providing a "table of contents" (titles of these files/units and one line descriptions). The first page of the program listings folder must contain the following statement certifying the work as your own: "I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary". Your (typed) signature and the date should follow this statement.

All work on programs must stop once the code is submitted to KEATS. You are required to keep safely several copies of this version of the program and you must use one of these copies in the project examination. Your examiners may ask to see the last-modified dates of your program files, and may ask you to demonstrate that the program files you use in the project examination are identical to the program files you have uploaded to KEATS. Any attempt to demonstrate code that is not included in your submitted source listings is an attempt to cheat; any such attempt will be reported to the KCL Misconduct Committee.

You may find it easier to firstly generate a PDF of your source code using a text editor and then merge it to the end of your report. There are many free tools available that allow you to merge PDF files.