

CS2Go: A Computational Thinking Curriculum Engine
Conor O’Keeffe
14385046

Final Year Project – **2018**
B.Sc. Single Honours in
Computer Science and Software Engineering



Department of Computer Science
Maynooth University
Maynooth, Co. Kildare
Ireland

A thesis submitted in partial fulfilment of the requirements for the B.Sc.
Single Honours in Computer Science and Software Engineering.
Supervisors: Dr. Aidan Mooney & James Lockwood

Contents

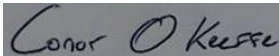
Declaration.....	i
Acknowledgements.....	ii
Abstract.....	iii
List of Figures	iv
List of Tables	iv
Chapter one: Introduction	1
Summary	1
1.1 Topic addressed in this project.....	1
1.2 Motivation.....	1
1.3 Problem statement	1
1.4 Approach.....	1
1.5 Metrics	2
1.6 Project.....	2
Chapter two: Technical Background	3
Summary	3
2.1 Topic material	3
2.2 Technical material.....	3
2.2.1 Python Flask.....	4
2.2.2 MongoDB	4
2.2.3 GridFS.....	4
2.2.4 Data Visualization – ChartJS vs D3.....	4
Chapter three: The Problem.....	5
Summary	5
3.1 Problem Statement.....	5
3.2 Project UML Documentation	5
3.2.1 Use Case Diagram.....	5
3.2.2 Class Diagram.....	6
3.2.3 Sequence Diagram	6
3.3 Problem Analysis.....	6
Chapter four: The Solution.....	8
4.1 Implementation	8
4.1.1 Retrieval of Data	8
4.2 User authentication	9

4.3	Architectural Level	10
4.3.1	System Architecture.....	10
4.3.2	Information Architecture.....	10
4.4	Visualisation Design	11
4.5	Network Level Design	13
Chapter five: Evaluation.....		14
	Summary	14
5.1	Evaluation of System Requirements	14
5.2	Compatibility Verification	14
5.3	Functionality Testing.....	15
5.3.1	Selenium Testing.....	15
5.3.2	User Testing	15
5.4	Unit Testing	16
Chapter Six: Conclusion		17
6.1	Results discussion	17
6.2	Project Approach	17
6.3	Future Work	17
References		18
Appendices		19
Appendix 1	Code developed for this project.	19
Appendix 2	Screenshots of the project implementation	19

Declaration

I hereby certify that this material, which I now submit for assessment on the program of study as part of Single Honours in Computer Science & Software Engineering qualification, is *entirely* my own work and has not been taken from the work of others - save and to the extent that such work has been cited and acknowledged within the text of my work.

I hereby acknowledge and accept that this thesis may be distributed to future final year students, as an example of the standard expected of final year projects.

Signed: 

Date: 27/03/2018

Acknowledgements

I would like to thank my supervisors Dr Aidan Mooney and James Lockwood of Maynooth University for giving me the opportunity to undertake and complete this project, and also for their valuable guidance and support throughout. Thanks also to my girlfriend and her parents for reviewing and advising throughout the duration of this project. Lastly, thanks to my friends who have made my college experience so enjoyable and memorable.

Abstract

Over the past two decades, technology has restructured how we live, how we communicate, but most of all, how we learn. As a result of the increasing use of computers, the internet, and the advancement in technologies, eLearning has become widely recognised as a valuable tool for learning and education practices. eLearning is a form of teaching in which educational material is delivered electronically to the learner via the internet. This project will aim to develop an entirely functioning eLearning system, which will provide a configurable infrastructure that integrates the learning material into a single solution, in order to deliver the educational content most effectively and cost-effectively. The educational content will be supplied by the PACT team in the Department of Computer Science at Maynooth University, which is currently being used on a third-party eLearning website. For this reason, an in-house eLearning application was needed. The goal is to provide the application user with an enjoyable and satisfying experience while using and navigating through the application. Evaluation will be carried out in the form of a survey as well as various software testing practices to determine the outcome and success of the application.

List of Figures

Figure 1: No. of users from different eLearning sites.....	3
Figure 2: Use Case Diagram	5
Figure 3: Class Diagram	6
Figure 4: Gantt Chart representing the time each activity took to complete.	8
Figure 5: Insert method, storing user information	9
Figure 6: Find_one method to check for a particular instance.....	9
Figure 7: Update function to approve a student to class	9
Figure 8: Hash password algorithm	9
Figure 9: Algorithm used for decrypting the password and storing users unique ID in the session	9
Figure 10: High-Level diagram of a system architecture	10
Figure 11: Wireframe of proposed Homepage & User authentication page	11
Figure 12: Wireframe of proposed Profile page & Test page.....	12
Figure 13: Profile page of the final design.....	12
Figure 14: Sequence diagram of the network level of various pages.	13
Figure 15: Evaluating the outcome of an invalid username input	15
Figure 16: Evaluating the outcome of unapproved students on the profile page.....	15
Figure 17: Unit testing the validate of various URLs	16

List of Tables

Table 1: Overall system requirements	7
Table 2: System requirement evaluation.....	14
Table 3: Cross-browser evaluation outcome.....	14
Table 4: Represents results of Mobile results	15

Chapter one: Introduction

Summary

The goal of this project is to create a computational thinking engine, named CS2Go. CS2Go is a Virtual Learning Environment, with the content supplied by the PACT team at Maynooth University.

1.1 Topic addressed in this project

The PACT programme at Maynooth University is a partnership between researchers in the Department of Computer Science and teachers at various primary and post-primary schools around Ireland [1]. The goal of the programme is to introduce students and teachers to the field of Computer Science, with a view to improving Computational Thinking skills among the participating students. The programme is now well-established and enrolled in over 60 schools and to over 10,000 students. As part of this programme, a Computational Thinking curriculum is being developed by members of the PACT team and this course content is currently being presented and used on a third-party platform, namely Schoology [2].

1.2 Motivation

eLearning is a powerful tool to engage, educate and inspire both students and faculty. The PACT team at Maynooth University has content readily available, although the use of an out-sourced online learning environment to supply the content brings its own issues. There was a distinct need for an in-house computational thinking curriculum engine. The development of an eLearning application would provide many benefits, including independent control by the Department of Computer Science, but would also remove the need to rely on a third-party platform. The end result is that the overall functionality, look, feel and usability could be tweaked and maintained in-house.

1.3 Problem statement

There were various technical elements encountered during this project, as the main aim was to develop a secure and user-friendly environment for students and teachers to access the relevant course content. The platform needed to be developed using Python Flask and a MongoDB database. There were several requirements given when developing the application; some include the ability for users to register on the system, the completion of various tests and survey and from this the ability to represent data to teachers and admin users in a visually pleasing manner e.g. the use of graphs.

1.4 Approach

This section provides a summary of the approach taken, although Chapter 4 provides the in-depth detail of each section. Due to the nature of the project, it was determined that the best approach was to take an iterative and incremental development (IID) methodology. The majority of the requirements were outlined at the outset, but there was an understanding that specific features were liable to change over the duration of the project, and this approach would facilitate a smoother transition should any adjustments be required to the application. An iterative process is one that makes progress through successive modification but is also highly structured around planning. During the initial planning and research phase, research was conducted to determine the preferred framework to implement the project on and Flask, a Python Micro-framework, as well as a MongoDB database were chosen. During each increment, each feature will be tested and evaluated. Finally, the application will be user tested to provide feedback on various elements such as functionality, the user interface, and the user experience.

1.5 Metrics

The metrics used to evaluate the undertaken work will be:

- As mentioned in section 1.4, the approach taken was an iterative one, which requires deployment on a regular basis. Once satisfied that the code worked on a local environment, it could be tested and pushed to a pre-production server called Heroku.
- A user feedback questionnaire was given to users to assess how they found the experience. This metric will also act as a form of ‘user testing’ to find bugs and provide feedback for upgrades.
- An evaluation, in the form of positive and negative feedback, from individuals at Maynooth University, with no prior experience using the developed platform.

1.6 Project

This project entailed many achievements, some of these include:

- Developing an entirely functioning eLearning platform for the use of students and faculty.
- Deploying the system on a Heroku server.
- Simplifying the data visualisation process in the form of charts for both students and teachers who use the platform.

Chapter two: Technical Background

Summary

2.1 Topic material

Over the last decade, the eLearning market has been thriving at a phenomenal rate, with most schools and universities now having a Virtual Learning Environment (VLE), at the forefront of their teachings. [3] They are used by universities globally, and every day, new applications are added to the virtual learning platforms. The primary overall objective is to improve the efficiency and the interaction experience for the student. A VLE is an online learning platform that allows teachers to share educational material with their students via the world wide web (www). Some examples of the most popular VLE's include Moodle, Blackboard and Schoology [4], although Moodle and Blackboard currently have significantly more users than Schoology, see Figure 1. Schoology tends to be aimed towards the second level education market, and Moodle and Blackboard are mainly used at third level. Most VLE's have a lot in common regarding features, although there are some fundamental differences which distinguish them from each other. Blackboard's Learning System allows faculty to upload course content, assignments and grants the functionality elementary dialogue and cooperative tools. Moodle slightly differs as it is a course management system (CMS), designed using pedagogical principles, to aid in the education advancement in educators.

In recent years, noticeable trends have emerged within the eLearning field. These trends include Gamification, Mobile Learning (mLearning) and Microlearning. Gamification is an educational approach to learn by using video game design and elements in a learning environment. As its name states, mLearning is a form of distance education where mobile devices are used. The mLearning market is growing staggeringly and is reported to reach \$70 billion by 2020 [12]. Microlearning is the process of teaching and delivering content to its users in small, precise bursts.

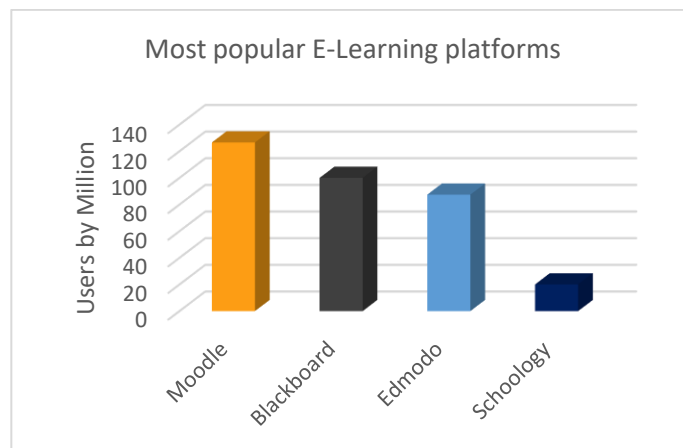


Figure 1: No. of users from different eLearning sites

2.2 Technical material

Designing a web-based application requires a vast amount of technical material. The main components used throughout the developed application were Flask [6] (a Python Microframework) and a NoSQL database (MongoDB) [7] to store the necessary data.

2.2.1 Python Flask

There were several reasons why Flask was chosen, firstly due to its simplicity and its controllability. The reason for this is that Flask implements a bare-minimum skeleton of the application and leaves the bells and whistles to add-ons or to the developer. It is possible to install necessary and relevant packages using the keyword “pip install” followed by the package name, the package ‘bcrypt’ will be necessary to encrypt the user personal information such as passwords. Secondly, since the Computer Science department in Maynooth University is migrating their web services to Flask, there will always be faculty and individuals within the Computer Science Department that will be familiar with this technology to make any relevant changes or to simply maintain the system.

2.2.2 MongoDB

As well as being a requirement of the project, MongoDB seemed an obvious choice for many reasons. Due to the scalability of an E-Learning application, it was evident that year on year the application and database schemes would change. MongoDB, due to its flexibility and scalability for querying and indexing data was the ideal system. MongoDB is a nonrelational database that stores data in a collection of JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time. The difference between the likes of MongoDB (nonrelational database) and MySQL (relational database) is the representation of data in the two databases, MySQL stores data in a table format (rows and columns) whereas MongoDB stores data in a collection of JSON-like documents.

2.2.3 GridFS

Another topic researched as part of this project was GridFS [8]. MongoDB allows for files smaller than 16MB to be pushed to the database, although the scenario where, as an example, a video clip (which exceeds the 16MB limit) is pushed to the database would have to be considered. GridFS solves this issue. GridFS is MongoDB specification for storing and retrieving large files. The way GridFS works is instead of storing a file in a single document, GridFS divides the file into chunks, and stores each chunk as a separate document.

2.2.4 Data Visualization – ChartJS vs D3

As the task of creating a visualised environment for online learning visualising the data would be an important aspect - the visualisation of data, such as test results would be relevant. Therefore, a charting library would become very relevant. A key decision to be made was which JavaScript charting library would be the most suited for the project. There were two obvious choices, D3 [10] (Data-Driven Documents) and ChartJS [9]. From researching both, each had their pros and cons. D3 offered a variety of different charts although when assessed its implementation was determined to be complex in nature. ChartJS offered basic geometric charts although a lot simpler to implement than D3 [11]. ChartJS was chosen simply due to its simplicity and elegance, and as outlined above was easier to implement.

Chapter three: The Problem

Summary

This chapter identifies the requirements of the project in several Unified Modelling Language (UML) diagrams, namely a use case diagram, a class diagram and a sequence diagram. Also, several system requirements needed to be addressed.

3.1 Problem Statement

Various VLE systems exist, these systems are efficient a satisfactory experience to the user. However, these systems are not fully controllable by Maynooth University, making them difficult to personalise and control, which is why an independently controlled system needs to be built. Developing an eLearning application provides many benefits. It supports the independent control by the Department of Computer Science. It facilitates the tailoring of the overall functionality, look, feel and usability being able to be tweaked and maintained in-house.

This project aims to develop a VLE system for the students and teachers, called CS2Go. The goal can be achieved by designing and implementing the system, as well as adding multiple functionalities which are currently not available within other various VLE applications.

3.2 Project UML Documentation

Documenting via UML can be a potent tool that can significantly improve the quality of system analysis and design. There were several UML diagrams used in the process of implementing the system.

3.2.1 Use Case Diagram

Firstly, a UML use case diagram, which is a UML behavioural diagram was generated. Use case diagrams aid in identifying system functionalities, and categorising system requirements for an application.

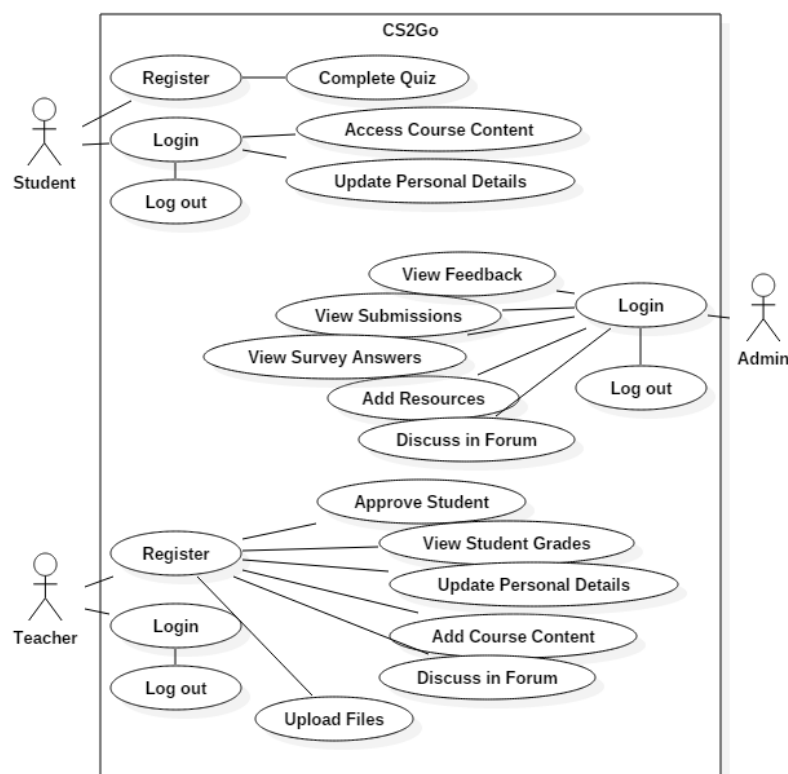


Figure 2: Use Case Diagram

The use case diagram in Figure 2 shows the primary model of the problem. It shows the role of the application user and demonstrates a brief run-through of how a typical user would interact with the CS2Go system.

3.2.2 Class Diagram

Secondly, a class diagram was created (see Figure 3). Class diagrams are one of the most useful types of diagrams in UML as they precisely map out the structure of a particular system by modelling its classes, attributes, operations and relationships between objects.

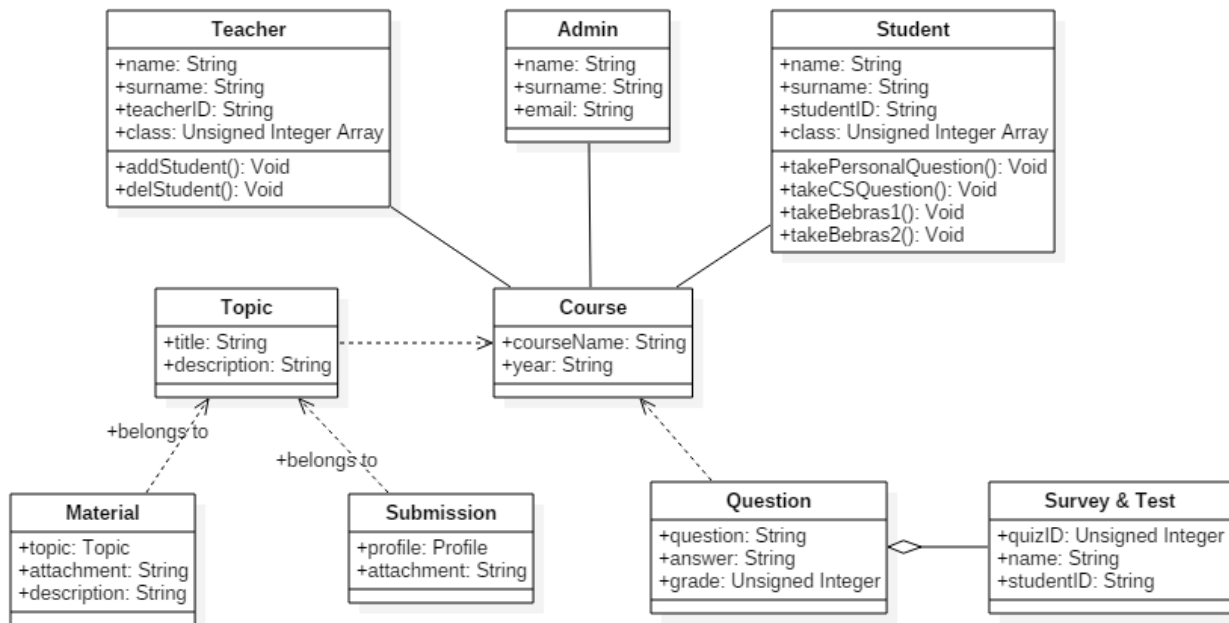


Figure 3: Class Diagram

3.2.3 Sequence Diagram

Finally, a sequence diagram was used, which helps compliment the use case diagram, shows the interaction of ‘how’ and ‘in what order’ a group of objects works together. The sequence diagram used, see Figure 8 in Section 4.5, is a low-level network design of the interaction flow between the different layers within the system. Further analysis is explained in Section 4.3.

3.3 Problem Analysis

In order to implement the functionalities referred to in Figure 2, the following user and system requirements must be addressed. Time was taken to meet with clients to discuss possible system requirements. These requirements are documented in the table below.

No.	System Requirements	
SR1	The frontend should be built using HTML5, CSS3 and JavaScript. The use of outside frameworks and libraries such as jQuery and Bootstrap are permitted.	Frontend
SR2	The system should allow all the content to be uploaded and downloaded from it. This includes file types such as PDF’s, PowerPoint etc.	
SR3	Test and survey data should include questions and multiple-choice answers. The results should be stored in a database.	
SR4	Test and survey data should be visually represented in a user-friendly and informative format, via charts or a tabular layout.	

SR5	The system should be developed using a python microframework called Flask, due to Maynooth University migrating all their web services to python, Flask.	Backend
SR6	Due to the possibility of front-end information being visible, user data should be stored in the back-end. This will improve security concerns around the accessibility of data.	
SR7	The database layer should be a MongoDB database. Additionally, due to Maynooth University moving towards a non-relational database, using MongoDB will allow for more flexibility will assist in handling the ever-increasing demand for data without any issues at all.	Database
SR8	The code base must be future-proof; well designed, testable and efficient by using software development practices.	General
SR9	The system should be easy (for someone with a CS background) to add lessons and content to it.	

Table 1: Overall system requirements

Chapter four: The Solution

4.1 Implementation

As discussed in Chapter 1, the approach taken throughout the project was a simplified version of the Agile Scrum development methodology. The use of this iterative and incremental development process was the most suitable due to the majority of the requirements being known at the beginning of the project. However, specific features, as well as additional features, were susceptible to change as time evolved. This approach allowed for a smoother transition should an adjustment to the application be required.

Following the standard practice of the iterative and incremental development (IID), several phases were conducted. The first phase of the IID is planning, which was fundamental to the design of the system. This planning allowed for strategic management of the requested technologies to develop the system and interface. Due to the nature of the project and most of the requirements being evident, an initial Gantt chart was constructed which mapped the tasks displayed against an estimated timeline. These tasks which were viewed as iterations were included in the Gantt chart one task per feature. All tasks had a start date and estimated time of completion (in days). The main pages of the website such as the user authentication pages, profile page and course content page, as well as subsections within the profile page were considered an iteration. These iterations consisted of a planning phase, an implementation phase and a testing phase. These phases were then repeated until the requirements had been fulfilled.

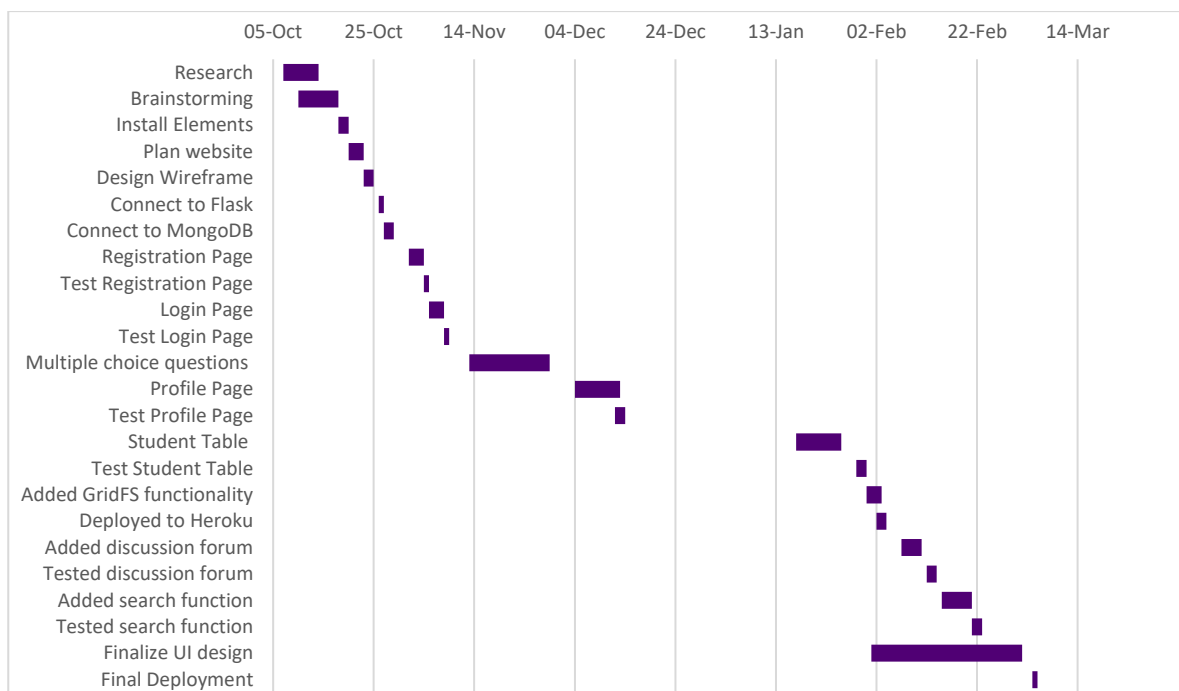


Figure 4: Gantt Chart representing the time each activity took to complete.

In Figure 4, the X-axis outlines the time scale, and the Y-axis represents the list of activities. Being a massive project with many unknowns and new technologies some iterations took longer than expected, due to additional implementation needed or simply not knowing the correct way to code it.

4.1.1 Retrieval of Data

The retrieval of data from the MongoDB database was done via PyMongo which is a python library for working with MongoDB. Connection to the MongoDB database was implemented using a URI, from this it was possible to fetch any document from a collection in the database. Users, tests results and course content were stored in the database each within its own collection. This data was inserted via the

Mongo's 'insert()' method. Figure 5 shows a teacher's details being inserted into the user's collection in the database as well as the attributes are shown.

```
users.insert(
{
    'user_type': user_type,
    'school': school,
    'title': title,
    'name': name,
    'surname': surname,
    'email': email,
    'password': hashpass
})
```

Figure 5: Insert method, storing user information

```
users.find_one({'email': session['email']})['name']
users.find_one({'userid': session['userid']})['name']
```

Figure 6: Find_one method to check for a particular instance

With Mongo, it was also possible to get a single document within a collection using the 'find_one()' method. This method was useful when there was only one matching document, but also to determine the type of user who is logged in. Figure 6 shows an implementation checking the session variable to see the type of user who is logged in and is returning that users name. An ObjectId was also used from the request URL to find the matching document in the collection; this was used when approving a student to the class. Essentially the teacher finds all users in the class, gets there ObjectId and approves them via the update function in Figure 7.

```
def update():
    users = mongo.db.users
    userid = request.args.get("_id")
    users.update_one({'_id': ObjectId(userid)}, {'$set': {'approved': 1}})
    return redirect(url_for('UsersPage'))
```

Figure 7: Update function to approve a student to class

4.2 User authentication

With regard to user authentication, measures had to be taken to protect the user's data securely. Bcrypt which is a python package was used, it is a modern hashing algorithm which is capable of importing into Flask. User authentication was developed on the backend to ensure all confidential data was secure.

Upon registration, both students and teachers are prompted to enter various fields, which include their first and last name as well as a password of their choosing. The algorithm used, hashes the inputted password which encrypts it, and stores the user's details in the user's collection in the database.

```
hashpass = bcrypt.hashpw(request.form['pass'].encode('utf-8'), bcrypt.gensalt())
```

Figure 8: Hash password algorithm

In terms of logging in as an existing user, the user needs to input their username and password. In Figure 9 this method begins by searching the user's collection to check if the user exists if the user exists the bcrypt function is called and decrypts the user's password and checks if they are the same, when this is complete the user's unique ID is stored in the session [13]. If the user or password do not exist or match, the user is sent an error message to retry. Flask's session data was used to store the user's unique ID on the server. By default, this session variable lasts until the user closes their browser.

```
login = users.find_one({'email': request.form['email']})
if login:
    if bcrypt.hashpw(request.form['password'].encode('utf-8'), login['password']) == login['password'] :
        session['email'] = request.form['email']
        return redirect(url_for('signIn'))
```

Figure 9: Algorithm used for decrypting the password and storing users unique ID in the session

4.3 Architectural Level

4.3.1 System Architecture

The overall architecture of the application was designed and implemented with three main components in mind; these include the user interface, server and the database layer. These principal components can be seen in Figure 10.

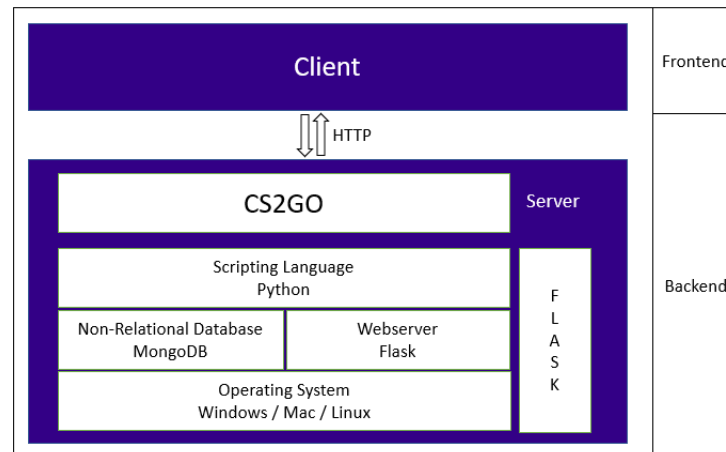


Figure 10: High-Level diagram of a system architecture

The user interface consists of various elements; these include HTML, JavaScript and CSS files. The JavaScript and CSS files are located in the static folder within the main directory. The HTML files are stored in the templates directory. With regards to the styling of the application, bootstrap was mostly used, although there also exist two CSS files in the application, one being main.css which is the general styling of the home page of the application and overall.css which is a central CSS file that contains styling that all pages use as a style guide. The main idea of these general style guides is to create styles that are open and common. These styles can then be used on several pages, due to them being multifunctional. This helped speed up the designing process of the pages and allowed for reusability throughout.

The server-side consisted of a single module. A single module was suitable due to the need of only a few routes needed to be served. This primary python file named main.py contains numerous functions which handles routing and several implementation actions.

The database layer is connected using a Uniform Resource Identifier (URI) via the server-side layer. The URI is MongoDB's standard format used to connect to a MongoDB database server. Using mLab which is a cloud database service that hosts MongoDB databases, made it possible to manage and view all collections in the database.

4.3.2 Information Architecture

Another form of architecture used throughout the project was information architecture (IA). IA focuses on the organisation and structure of content in a manner which a user can navigate through it. The structure and format of the content needed to be considered as CS2Go is an application highly focused on the end user. The initial design was implemented as a wireframe which represents the connections between different screens and identifies how the site will work from a practical perspective. The goal was to connect the user to the content they are looking for. With IA in use, the focus was on a number of elements, such as the end users and the data that will be presented on the website. Two main factors were considered in relation to IA; cognitive psychology and mental models.

Cognitive psychology - Cognitive psychology, the study of how the mind works, influences both the interactions we design and the way we architect information. Without overflowing any given page of the application, the cognitive load¹ of the end user needed to be considered. The primary aspect considered in the design was its simplicity of use, to avoid overloading a user with too much information all at once.

Mental Models - As eLearning is a widely used feature, it is more efficient to use the standard template. Mental models are the assumptions people carry in their minds before interacting with the website or application. Information is more accessible to discover when it is in a place that matches the users mental model of where it should be. With the concept of mental models in mind a grid type layout was used where data displayed is in the form of a hierarchy.

4.4 Visualisation Design

Well-designed web applications offer much more than just aesthetics, they attract a user base and help people understand the product through a variety of indicators, encompassing visuals, text, and interactions. There are several stages which require consideration when designing a web application. The first stage of the web design process is wireframing which is a low fidelity (Lo-Fi) representation of the design. The fidelity of the prototype accurately refers to how closely it matches the look-and-feel of the final system. Wireframes are the backbones of design, their purpose being to work in a minimalistic manner to start organising the information and start figuring out the flow of a page rather than worrying about things like fonts and colours and making everything pixel perfect from the get-go.

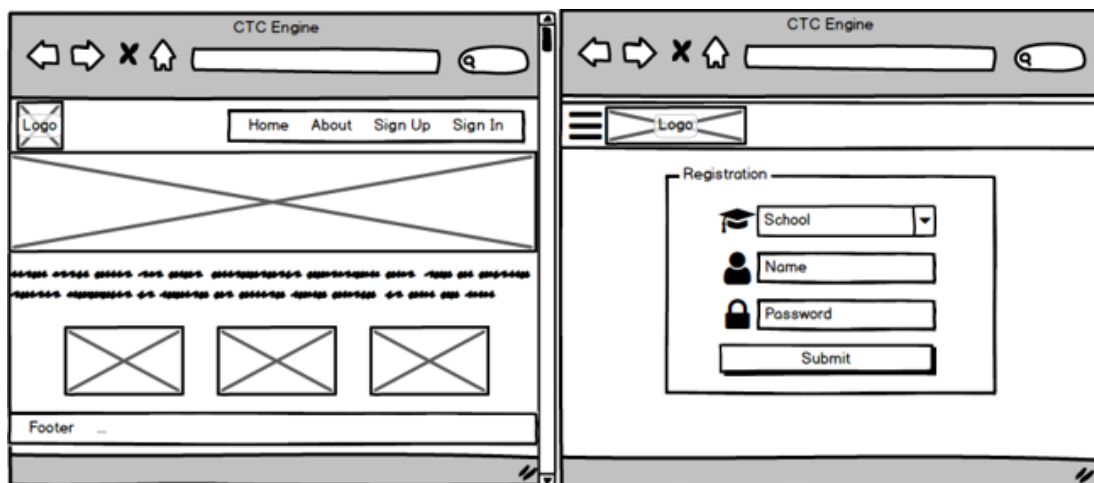


Figure 11: Wireframe of proposed Homepage & User authentication page

Iterative development was also used in the design phase of the application. An initial sketch was developed of various pages of the application with some of the required elements. While designing the wireframe elements needed to be added and removed several times. Wireframing was a suitable option as not one specific design was tested, several were used to explore many concepts for the most suitable option to be determined. Iterations were used to achieve a minimum viable product until an adequate wireframe representation of the final design was achieved.

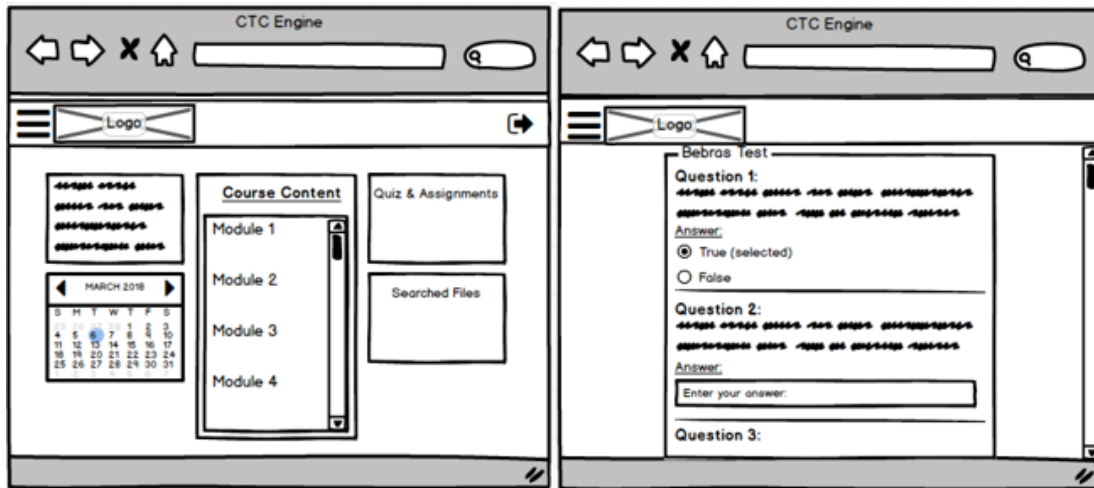


Figure 12: Wireframe of proposed Profile page & Test page

The next stage after designing a wireframe would be to develop a Mockup and/or a Prototype which are a medium to high fidelity (Hi-Fi) representations of the application, although these Hi-Fi representations were not used. These Hi-Fi representations are both time-consuming and expensive in nature, which was factors which needed significant consideration throughout the development phase.

The final design for various pages closely represents the design based on the sketch on the wireframe. This can be seen in Figure 13 of the profile page which includes the chosen colour scheme and icons which.

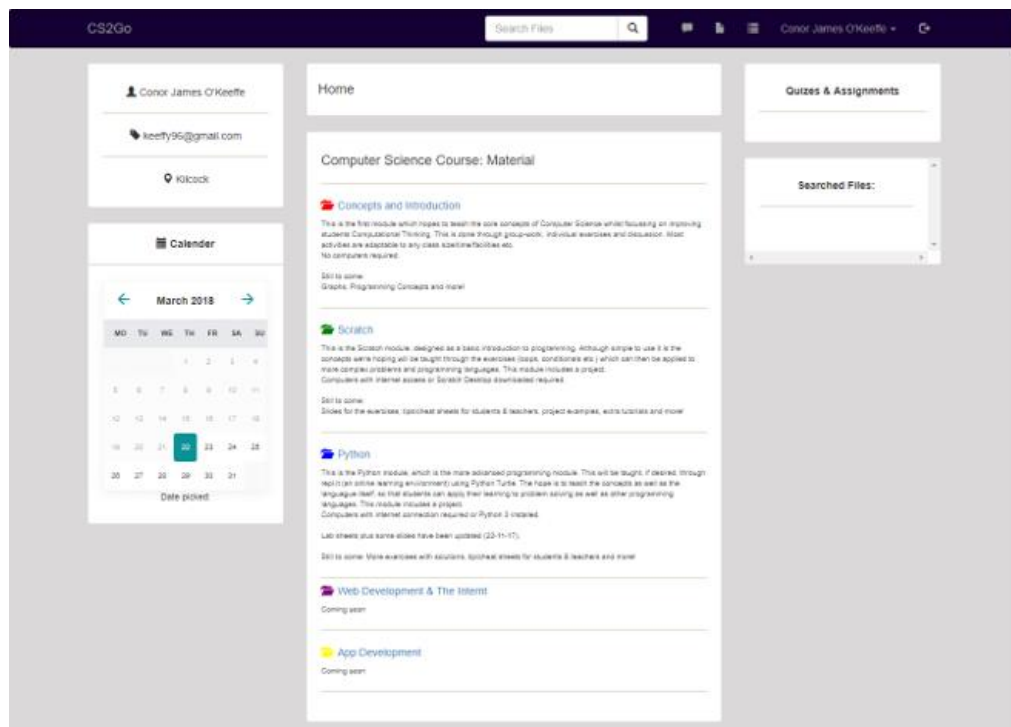


Figure 13: Profile page of the final design

4.5 Network Level Design

Figure 14 represents a network low-level design of the interaction flow between the various components over the network. HTTP requests were sent between layers sending the relevant and necessary data.

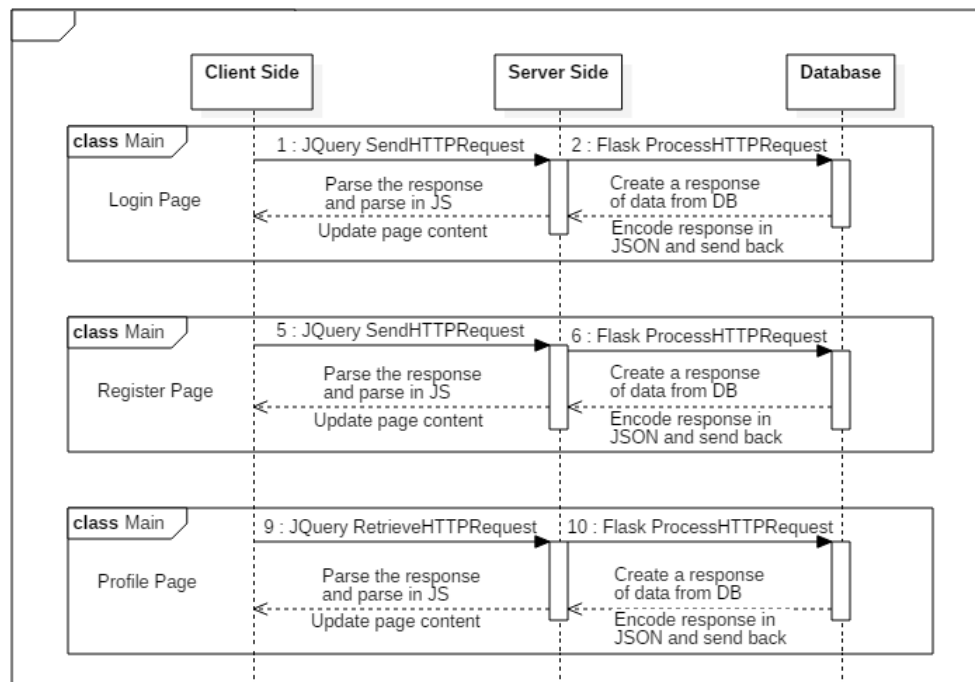


Figure 14: Sequence diagram of the network level of various pages.

Chapter five: Evaluation

Summary

Evaluation is a necessary and an essential aspect of web design, as faults are easy to overlook. Three main evaluation approaches needed to be considered throughout the implementation phase, these included, compatibility testing, functionality testing and unit testing.

5.1 Evaluation of System Requirements

In Section 3.3 the system requirements were presented for this project, the outcome of these requirements can be observed in Table 2. Table 2 depicts that all the requirements have been achieved.

Requirement	Achieved	Summary
SR1	Yes	The frontend should be built using HTML5, CSS3 and JavaScript.
SR2	Yes	The system should allow all the content to be uploaded and downloaded from it.
SR3	Yes	Test and survey data should include questions and multiple-choice answers. The results should be stored in a database.
SR4	Yes	Test and survey data should be visually represented in a user friendly and informative format, via charts or a tabular layout.
SR5	Yes	The system should be developed using Flask.
SR6	Yes	Due to the possibility of front-end information being visible, user data should be stored in the back-end.
SR7	Yes	The database layer should be a MongoDB database.
SR8	Yes	The code base must be future-proof; well designed, testable and efficient by using software development practices.
SR9	Yes	The system should be easy (for someone with a CS background) to add lessons and content to it.

Table 2: System requirement evaluation

5.2 Compatibility Verification

Although it was not a requirement to have the web application compatible with different browsers, operating systems and devices, it was evident that this needed to be considered because there is now a wide range of user devices available. There were two main areas which were tested, Browser compatibility and Mobile Compatibility.

Browser	User Interface	Retrieval of Data	Functionality
Google Chrome	As designed	Works as intended	Works as intended
Microsoft Edge (IE)	As designed	Works as intended	Works as intended
Mozilla Firefox	As designed	Works as intended	Works as intended
Safari	As designed	Works as intended	Works as intended

Table 3: Cross-browser evaluation outcome

Firstly, it was necessary to assess the website on various significant browsers other than the one it was developed on, Google Chrome. Due to different browsers having different configurations and settings CS2Go needed to be examined.

From the cross-browser evaluation, it is evident that the browser is, in fact, cross-browser compatible. Regarding functionality, the website worked flawlessly on all browsers.

With regards to Mobile browser compatibility, there were a few locations on the UI where the page overflowed, although this did not affect the functionality or the usability of the application.

Phone	User Interface	Retrieval of Data	Functionality
iPhone	Slight overflow on the user table	Works as intended	Works as intended
Samsung	Slight overflow on the user table	Works as intended	Works as intended

Table 4: Represents results of Mobile results

5.3 Functionality Testing

Functionality testing was split into two sections.

5.3.1 Selenium Testing

Selenium is a web testing plugin which uses simple scripts to run tests directly within a browser. It provides a convenient interface for developing automated tests. The selenium plugin includes a recording feature, which records a typical users actions as they are performed, these actions can include commands such as open, click, type and many others. Verification commands are also possible which allows the testing suite to specify expected values or behaviours. This recording is then converted into reusable test scripts that can be later executed or manually alternated. Passing test cases turn green and failing test cases turn red as the commands are run in real time.

Test cases were run at the end of every iteration which tested and evaluated the outcome of a given objective. Figure 15 and 16 represents a selection of the test cases implemented.

Command	Target	Value
1. click at	//a[contains(text(),'Sign in')]	31,17
2. click at	id=login-username	166,15
3. type	id=login-username	helloWorld
4. type	id=login-password	password
5. submit	id=loginform	
6. verify text	css=div.panel-title > div	Invalid login, please try again

Figure 15: Evaluating the outcome of an invalid username input

Figure 15 represents a test case where a user enters either of their details incorrectly, the selenium test case iterates through each command and evaluates the outcome. This test case passes successfully returning each command green.

Command	Target	Value
1. verify element present	css=div.alert.alert-danger	
2. verify text	css=div.alert.alert-danger	Attention! Please wait to be approved to this class.

Figure 16: Evaluating the outcome of unapproved students on the profile page

Figure 16 represents a test case where a user completes the sign-up process; the user is then placed in the profile page without permissions to access any of the content until a teacher approves of the student. Again, this test passes.

The 'Selenium_TestingCS2Go.side' file contains the remaining selenium test cases.

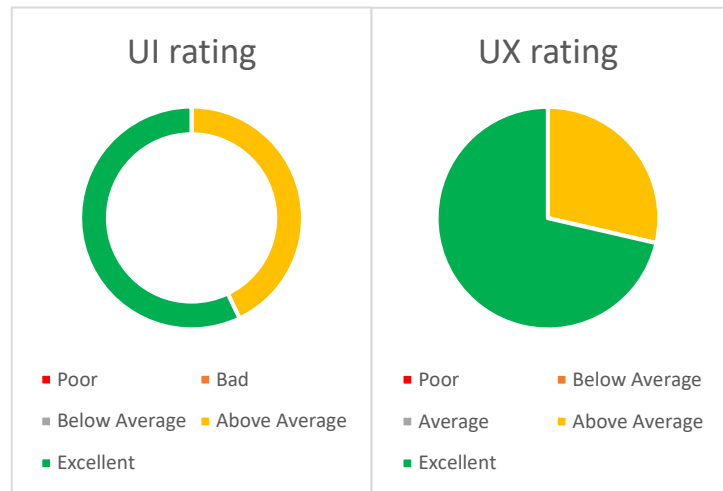
5.3.2 User Testing

A beta version of the web application was deployed on Heroku's cloud platform which enabled any user to sign up and view the application.

A survey was created via Google Forms to gather feedback from those who tested and interacted with CS2Go. In this survey, the participants were asked their opinion on some questions such as ‘What do you like about the UI’, ‘How would you rate the UI’, ‘What do you feed could be improved within the UI’. The same questions were asked based on their UX as well as any comments or suggestions regarding the application.

To validate the outcome of the survey results limited information was given to testers on how to interact with the system. Had there been any negative feedback about any element, then it would have been considered that the design aspect was not fit for purpose. However, this was not the case.

The results indicated that the majority of the participants found that the UI and UX were excellent.



5.4 Unit Testing

Unit testing is a form of software testing where individual units of the software are tested. The purpose is to validate that each unit of the software performs as designed. In Flask unit tests are created using the unittest library by creating a class in the test file that inherits from the unittest’s test case [14]. Debugging the file uses the unittest library and automatically finds all test that is written and runs them and returns the output. Figure 17 represents test case which checks the validate of several URLs. The homepage returns a HTTP status code of 200 (status “ok”) due to no user authentication being needed. The profile page returns a status code of 302 (status “found”) although nothing loads due to the need of a user being logged in. Finally, if an invalid URL is inputted, the status code 404 (status “error”) will be returned.

```
#Checks if home page loads correctly
def test_homePage_loads_correctly(self):
    tester = app.test_client(self)
    response = tester.get('/', content_type='html/text')
    self.assertEqual(response.status_code, 200)

#Checks if profile page returns code 302 - due to no user been signed in
def test_profilePage_loads_correctly(self):
    tester = app.test_client(self)
    response = tester.get('/profile', content_type='html/text')
    self.assertEqual(response.status_code, 302)

#Checks if invalid url returns error 404
def test_invalid_url(self):
    tester = app.test_client(self)
    response = tester.get('/notVaidURL', content_type='html/text')
    self.assertEqual(response.status_code, 404)
```

Figure 17: Unit testing the validate of various URLs

Unit testing, alongside the testing of the functionality of the system, reinforced that CS2Go was functioning as intended and without fault.

Chapter Six: Conclusion

6.1 Results discussion

The purpose, and motivation for this project was to develop a virtual learning environment for the use of students and teachers who are currently enrolled in the Computational thinking curriculum programme. The system, named CS2Go, adheres to this purpose, and successfully achieved the client's system requirements. The overall structure and functionality of CS2Go provides a visually engaging platform for the user, as well as a positive and beneficial user experience.

Success is determined by a number of key factors; the delivery of all system requirements, and the positive feedback provided from those who participated in the user feedback survey.

6.2 Project Approach

An iterative and incremental development methodology worked exceptionally well throughout the development phase. Research and planning were essential elements here and allowed for the swift handling of challenges and unforeseen circumstances. Testing was also fundamental in the approach, ensuring that iteration errors were found early to avoid unnecessary time-wasting searching for faults. If asked to design a similar application, this approach and methodology would be recommended.

6.3 Future Work

Website development is a never-ending process, and CS2Go is an application which can be continuously updated and improved to add new features and to keep up with recent trends. Web design trends are ever changing in line with evolving consumer preferences. Had there been no project timescale constraint, other additional features could have been considered/implemented such as:

- Data analytics based on a user's time spent on any given page.
- Optimise loading time of forms through the use of AJAX which handles asynchronous loading.
- Adding gamification which allows the learner to experience "fun" during the tests in the form of a game and still learn if the level of engagement is high. Although this task is easier said than done.
- More functionality to the calendar feature.
- Add more methods of registering, possibly the implementation of adding a google sign in or a similar feature.

Hence, the continuous development of CS2Go is required to ensure and deliver the maximum user satisfaction experience.

References

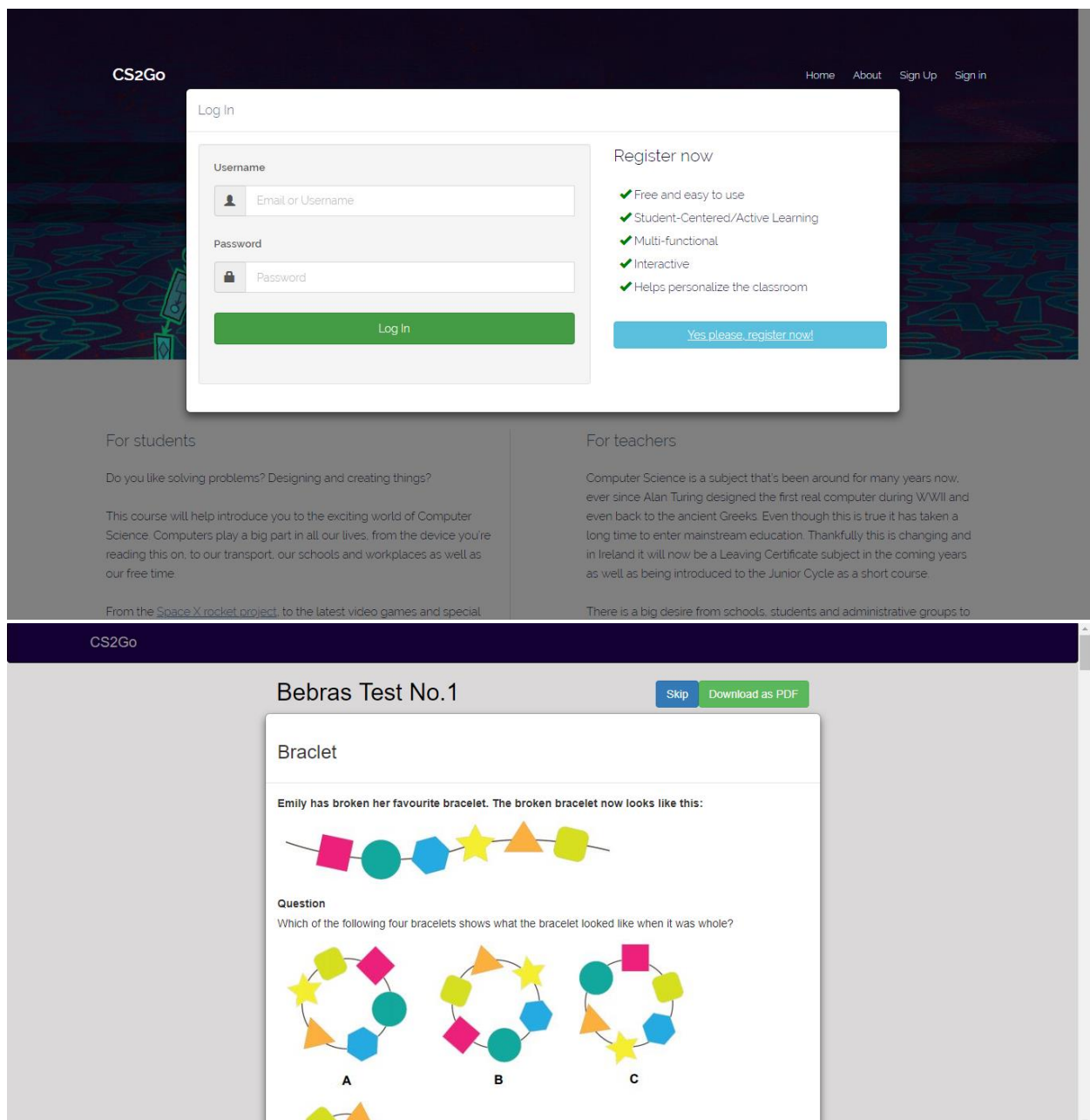
- [1] PACT: An Initiative to Introduce Computational Thinking into second level education Aidan Mooney, J. Duffin, T. Naughton, R. Monahan, J. Power and P. Maguire (2014) PACT: An Initiative to Introduce Computational Thinking in to second level education *International Conference on Engaging Pedagogy (ICEP)*
- [2] Schoology, Learning Management System, Retrieved November 5th, 2017, <<https://www.schoology.com/>>.
- [3] Jasmini V 2017, *Online Learning Statistics And Trends*, Retrieved February 8th, 2018 <<https://elearningindustry.com/online-learning-statistics-and-trends>>
- [4] ASB Academy, *Top 6 eLearning Trends to Watch Out For 2017*, Retrieved February 8th, 2018 <<http://www.asbacademy.org/top-6-elearning-trends-watch-2017/>>
- [5] Jisri. *A Study of Comparison between Moodle and Blackboard based on Case Studies for Better LMS*. Retrieved 20 March 2018, <https://www.moodlebites.com/pluginfile.php/26295/mod_resource/content/1/Pub4_ComparisonBetweenMoodleAndBlackboard.pdf>
- [6] Armin Ronacher 2018, Flask Micro-Framework, Retrieved October 29th, 2017, <<http://flask.pocoo.org/docs/0.12/>>
- [7] MongoDB, Inc. 2008, MongoDB, Retrieved October 28th, 2017, <<https://api.mongodb.com/python/3.6.0/>>
- [8] MongoDB, Inc. 2008, MongoDB, Retrieved January 29th, 2018, <<https://docs.mongodb.com/manual/core/gridfs/>>
- [9] ChartJS 2018, Retrieved November 8th, 2017, <<http://www.chartjs.org/docs/latest/>>
- [10] Mike Bostock 2017, *Data-Driven Documents*, Retrieved November 8th, 2017, <<https://d3js.org/>>
- [11] Syed Fazle Rahman, *JavaScript Libraries for Creating Beautiful Charts*, Retrieved November 12th, 2018, <<https://www.sitepoint.com/15-best-javascript-charting-libraries/>>
- [12] Karla Gutierrez 2017, *Mobile Learning Stats that Will Make You Rethink Your Training Strategy*, Retrieved February 20th, 2018, <<https://www.shiftelearning.com/blog/bid/331987/mobile-learning-stats-that-will-make-you-rethink-your-training-strategy>>
- [13] Creating a Login Page in Flask Using Sessions. (2018). YouTube. Retrieved 27 March 2018, <<https://www.youtube.com/watch?v=eBwhBrNbrNI>>
- [14] Discover Flask, Part 7 - Unit Tests. (2018). YouTube. Retrieved 27 March 2018, <<https://www.youtube.com/watch?v=1aHNs1aEATg&t>>

Appendices

Appendix 1 Code developed for this project.

Due to the nature of this project the code base is vast. The code developed for this project can be viewed via the submission on Moodle.

Appendix 2 Screenshots of the project implementation



CS2Go

Conor James O'Keeffe

Bebras 1 Test - Conor O'Keeffe's results:

Correct Answers

Wrong Answers

Correct Answers

Wrong Answers

Question	Answers	
Braclet	a	
Animation	gt	
Animal Competition	5	
Cross Country	B	
Stack Computer	sdf	
Throw the Dice	4	
Drawing Stars	10.4	✓
Beaver Lunch	B	
You Won't Find It	FLOOD	✓
Bowl Factory	dsf	
Fireworks	3	
Kangaroo	E	
Spies	2	

Close

Student Feedback:

To be implemented

CS2Go

Skip Test

View of Computer Science

3/5

Would you/have you considered studying Computer Science in college/university? *

☐ Yes
 ☐ No
 ☐ Maybe

What do you think of when you hear Computer Science? *

Your answer

Is Computer Science interesting to you? *

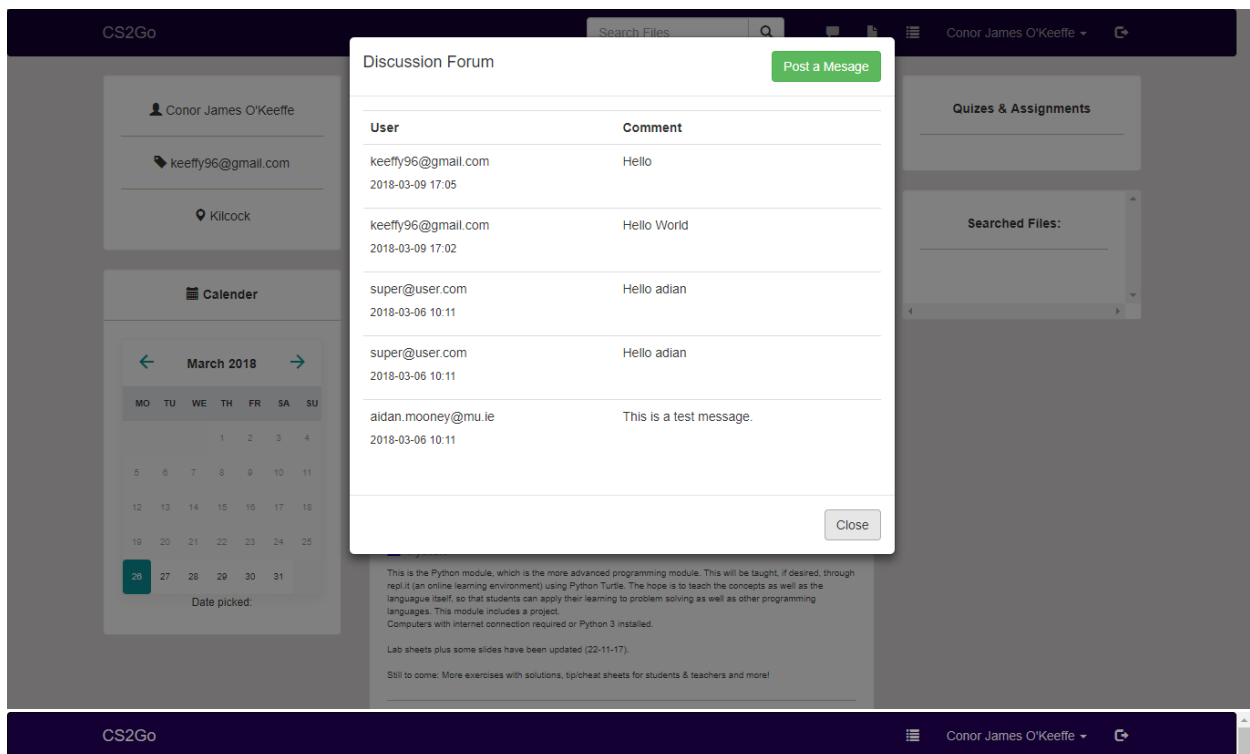
☐ Yes
 ☐ No
 ☐ Maybe

If yes/no, why?

Your answer

Is Computer Science challenging? *

☐ Yes
 ☐ No
 ☐ Maybe



Users table

School	Name	Surname	UserID	Approve student
Kilcock	Niall	Walsh	Kilcock212	
Kilcock	Aoife	Burke	Kilcock645	<input checked="" type="checkbox"/>
Kilcock	Richard	Fox	Kilcock531	<input checked="" type="checkbox"/>
Kilcock	James	Bould	Kilcock714	
Kilcock	Ray	Gibson	Kilcock830	<input checked="" type="checkbox"/>
Kilcock	Test	Test	Kilcock359	<input checked="" type="checkbox"/>
Kilcock	,,ijn	kgku	Kilcock504	
Kilcock	test	check	Kilcock732	<input checked="" type="checkbox"/>
Kilcock	asdf	asdf	Kilcock614	<input checked="" type="checkbox"/>
Kilcock	Richard	Nolan	Kilcock80	<input checked="" type="checkbox"/>
Kilcock	sdfads	asdf	Kilcock989	<input checked="" type="checkbox"/>
Kilcock	gmfd	dfhg	Kilcock341	<input checked="" type="checkbox"/>
Kilcock	Helga	Burke	Kilcock466	<input checked="" type="checkbox"/>
Kilcock	James	Lockwood	Kilcock179	<input checked="" type="checkbox"/>
Kilcock	Conor	O'Keeffe	Kilcock214	<input type="checkbox"/>