**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.

Supervisor: Aidan Mooney & James Lockwood

**Contents**

[Declaration i](#_Toc476122285)

[Acknowledgements ii](#_Toc476122286)

[Abstract iii](#_Toc476122287)

[List of Figures iv](#_Toc476122288)

[List of Tables iv](#_Toc476122289)

[**Chapter one: Introduction** 1](#_Toc476122290)

[Summary 1](#_Toc476122291)

[1.1 Topic addressed in this project 1](#_Toc476122292)

[1.2 Motivation 1](#_Toc476122293)

[1.3 Problem statement 1](#_Toc476122294)

[1.4 Approach 1](#_Toc476122295)

[1.5 Metrics 2](#_Toc476122296)

[1.6 Project 2](#_Toc476122297)

[**Chapter two: Technical Background** 3](#_Toc476122298)

[Summary 3](#_Toc476122299)

[2.1 Topic material 3](#_Toc476122300)

[2.2 Technical material 3](#_Toc476122301)

[**Chapter three: The Problem** 4](#_Toc476122302)

[Summary 4](#_Toc476122303)

[3.1 Project UML documentation 4](#_Toc476122304)

[3.2 Problem analysis 4](#_Toc476122305)

[**Chapter four: The Solution** 5](#_Toc476122306)

[Summary 5](#_Toc476122307)

[Depending on your type of project, you may not need to include all of these: 5](#_Toc476122308)

[4.1 Analytical Work 5](#_Toc476122309)

[4.2 Architectural Level 5](#_Toc476122310)

[4.2 High Level 5](#_Toc476122311)

[E.g. Packages, Class Diagrams, etc. 5](#_Toc476122312)

[4.2 Low Level 5](#_Toc476122313)

[E.g. Method specifications, Algorithms, etc. 5](#_Toc476122314)

[4.2 Implementation 5](#_Toc476122315)

[**Chapter five: Evaluation** 6](#_Toc476122316)

[Summary 6](#_Toc476122317)

[5.1 Solution Verification 6](#_Toc476122318)

[E.g. use your equations to verify the correctness of your solution 6](#_Toc476122319)

[5.2 Software Design Verification 6](#_Toc476122320)

[5.3 Software Verification 6](#_Toc476122321)

[5.3.1 Your test approach (i.e. unit testing, sub-system testing, system testing) 6](#_Toc476122322)

[5.3.2 Your tests (e.g. scenarios, test cases, test data, etc.) 6](#_Toc476122323)

[5.3.3 Your test results 6](#_Toc476122324)

[5.3.4 An interpretation of the results 6](#_Toc476122325)

[5.4 Validation/Measurements 6](#_Toc476122326)

[5.4.1 Results 6](#_Toc476122327)

[5.4.2 Explanation of Results 7](#_Toc476122328)

[5.4.3 Analysis of Results 7](#_Toc476122329)

[5.4.4 Comparison with previous solutions (if relevant) 7](#_Toc476122330)

[**Chapter six: Conclusion** 8](#_Toc476122331)

[**Summary** 8](#_Toc476122332)

[**5.1** **Contribution to the state-of-the-art** 8](#_Toc476122333)

[**5.2** **Results discussion** 8](#_Toc476122334)

[**5.3** **Project Approach** 8](#_Toc476122335)

[**5.3** **Future Work** 8](#_Toc476122336)

[**References** 9](#_Toc476122337)

[**Appendices** 10](#_Toc476122338)

[Appendix 1 Schematic of the hardware associated with this project. 11](#_Toc476122339)

[Appendix 2 Code developed for this project. 12](#_Toc476122340)

[Appendix 3 UML Class, Use Case and sequence diagrams for this project. 13](#_Toc476122341)

[Appendix 4 Screen shots of the project implementation 14](#_Toc476122342)

## 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:

## Acknowledgements

## Abstract

Style

The abstract should be a microcosm of the full report.

The abstract must be self-contained, **without** abbreviations, footnotes, or references.

The abstract must be between 150-250 words.

The abstract must be written as one paragraph, and **should not** contain displayed mathematical equations or tabular material.

The abstract should include three or four different keywords or phrases, as this will help readers to find it.

Ensure that your abstract reads well and is grammatically correct.

## List of Figures

[Figure 3‑1 UML class diagram overview for this project. 4](#_Toc445718606)

## List of Tables

[Table 2‑1 Table of interest: Aspect of your implementation 2](#_Toc445714278)

[Table 2‑2 Data sources used in your implementation 2](#_Toc445714279)

# **Chapter one: Introduction**

## Summary

The goal of this project is to create a computational thinking engine, namely 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 this programme is to introduce students and teachers to the field of Computer Science, from this to improve the skill of Computational Thinking in the participating students. The programme is now in well-established and enrolled in more than 60 schools and to over 1000 students. As part of this programme a Computational Thinking curriculum is being developed by members of the PACT team and this course content is being displayed and used on a third-party platform, namely Schoology [2].

## 1.2 Motivation

eLearning is a powerful tool to engage, educate and inspire students and faculty. The PACT team at Maynooth University has content readily available although using an out-sourced online learning environment to supply the content brings its own issues. There was an obvious need for an in-house computational thinking curriculum engine. Developing an eLearning application provides many benefits, mainly due to it being independently controlled by the Department of Computer Science, this also removes the need to rely on a third-party platform. Due to these concerns, the overall functionality, look, feel and usability can be tweaked and maintained in-house.

## 1.3 Problem statement

There were various technical elements that were encountered during this project, the aim of the project was to develop a secure, user friendly environment for students and teachers to use 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, various tests and survey and from this represent data to teachers and admin users in a data analytic style e.g. the use of graphs.

## 1.4 Approach

This section, provides a summary of the approach taken, although an in-depth (chapter 4) chapter explains in detail each section. Due to the nature of the project the best approach to take would be to use an iterative and incremental development methodology. Due to the majority of the requirements being known during the early stages of the project, certain features were susceptible to change as time evolves. Therefore, this approach would permit a smoother transition if an adjustment to the application was required. Essentially, an iterative process is one that makes progress through successive modification, it is also highly based around planning. During the planning and research phase, research was conducted into which framework to implements the project on and Flask, a Python Micro-framework as well as a MongoDB database was 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 their 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, this lead to deployment being on a regular basis, after being satisfied the code worked on a local environment, it was tested and pushed to a pre-production server called Heroku.
* A user feedback questionnaire will be given to users to see how they found the experience. This metric will also act as a form of ‘user testing’ to find bugs and provide feedback for upgrades.
* Getting the developed work evaluated by individuals at Maynooth University, who have no prior experience using the developed platform to provide positive and negative feedback

## 1.6 Project

This project entailed many achievements, some of these include:

* Developing a fully functioning eLearning platform which is deployed and working flawlessly on a Heroku server.
* Simplifying data visualization process in the form of charts for both students and teachers who use the platform.

# **Chapter two: Technical Background**

## Summary

## 2.1 Topic material

Within the last decade the eLearning market has been thriving at an uncommon rate, most schools and universities now have a functioning Virtual Learning Environment (VLE), at the forefront of their teachings. [3] They are used by all universities around the world and every day, new applications are added to the virtual learning platforms. The main objective overall is to improve the efficiency and the interaction between the students. 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 second level and Moodle and Blackboard are mainly used at third level. Most VLE’s have a lot in common in terms of features, although there are some key differences which differentiate each of them. The Blackboard Learning System allows instructors to post course information and course materials, readings and assignments and provides functionality for basic discussion and other collaborative tools from Blackboard. Moodle is a course management system (CMS), it is a free Open Source software package designed using sound pedagogical principles, to help educators create effective online learning communities. [5]

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 distant education where mobile devices are used. The mLearning trend market is growing staggeringly and is reported to reach $70 billion by 2020. [12] Finally, Microlearning is the process of teaching and delivering content to its users in small, very specific 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 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 are 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 just to simply maintain the system.

### 2.2.2 MongoDB

MongoDB seemed an obvious choice for many reasons as well as being a requirement of the project. Building 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 decision that was needed 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 seemed implementation to be complex in nature. Whereas 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 mentioned above it is easier to implement.

# **Chapter three: The Problem**

## Summary

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

## 3.1 Problem Statement

There exists various VLE systems, these systems are very practical and offer the user a satisfiable user experience. However, these systems aren’t independently controlled by Maynooth University which is their down fall. This makes it difficult to personalise and control such systems. Developing an eLearning application provides many benefits, mainly due to it being independently controlled by the Department of Computer Science as well as 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 use of 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 various other VLE applications.

## 3.2 Project UML Documentation

Documenting via UML can be a very powerful tool that can greatly improve the quality of a 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 an UML behavioural diagram was generated. Use case diagrams aid in identifying system functionalities, and categorising system requirements for an application.

The use case diagram in figure X shows the main model of the problem. It shows the role of the application user and what they should be able to do within the application, as well as the associated necessities of the application.

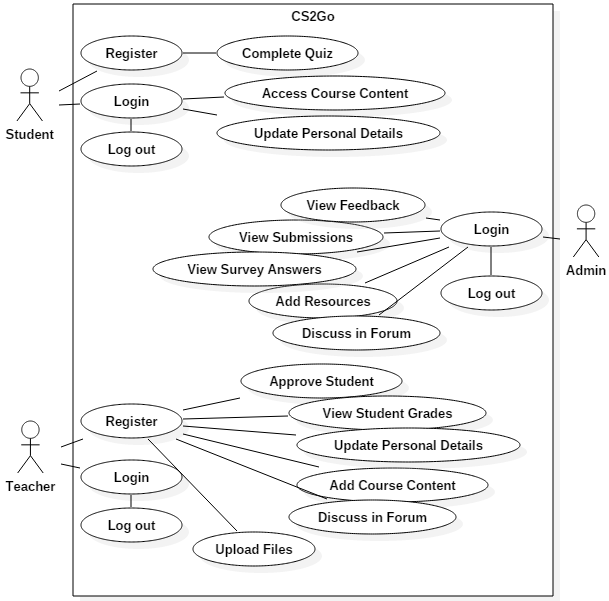
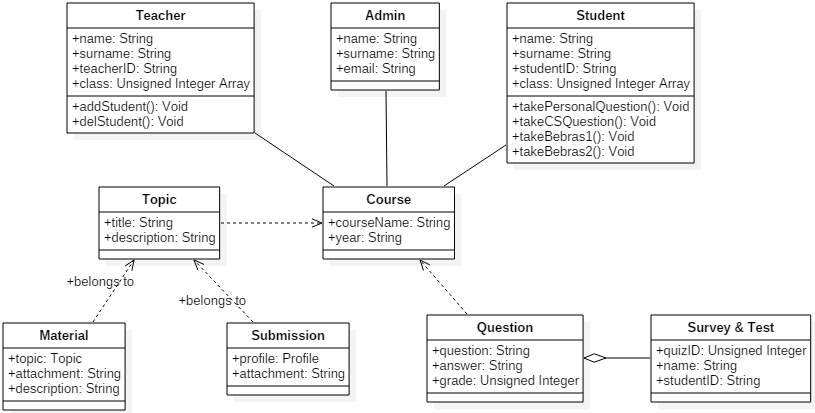


Figure 2: Use Case Diagram

### 3.2.2 Class Diagram

Secondly, a class diagram was created. Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations and relationships between objects. [See figure in chapter 4 – going to talk about it in more detail there]



Finally, a sequence diagram, which helps compliment the use case diagram, shows the interaction of ‘how’ and ‘in what order’ a group of objects works together. Sequence diagrams are a useful tool when trying to understand the requirements of a new system. [To be added].

## 3.3 Problem Analysis

In order to implement the mentioned functionalities in figure [use case diagram], the following user and system requirements must be addressed.

|  |  |  |
| --- | --- | --- |
| **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 and handle 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 practises. | General |
| SR9 | The system should be easy (for someone with a CS background) to add lessons and content to it. |

# **Chapter four: The Solution**

## Summary

## 4.1 Implementation

As mentioned in chapter one, the approach taken throughout the duration of the project was a simplified version of the Agile Scrum development methodology. Using an iterative and incremental development process, was the most suitable to use due to the majority of the requirements being known at the beginning of the project. Although certain features as well as additional features were susceptible to change as time evolved. This approach clearly allowed for a smoother transition if an adjustment to the application was required.

Following the Scrum methodology, standard practice outlined by the structure of the Iterative and incremental development (IID). The first phase of the IID is planning which was fundamental to the design of the system. This allowed for strategically strategizing the necessary and requested technologies which were required 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 estimate time. These tasks which were viewed as iterations were included in the Gantt chart one task per feature. All tasks have a start date and estimate 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 sub sections within the profile page were considered an iteration. These iterations consisted of a planning phase, an implementation phase and a testing phase. These phases where then repeated upon until the requirements had been fulfilled.

Figure 3: Gantt Chart

In figure X the left of the chart is a list of the activities and along the top is a suitable time scale. This is the precise Gantt chart which time the elements took to complete. Being a large project with many unknows and new technologies some iterations took longer than expected, due to additional implementation needed or simply not knowing the correct way to code it.

Maybe talk about jinja template here – index inheritaces all ….

Talk about user authentication bcrypt

## 4.2 Architectural Level

### 4.2.1 System Architecture

The overall architecture of the application was designed and implemented with four main components in mind, these include the user interface, server and the database layer.

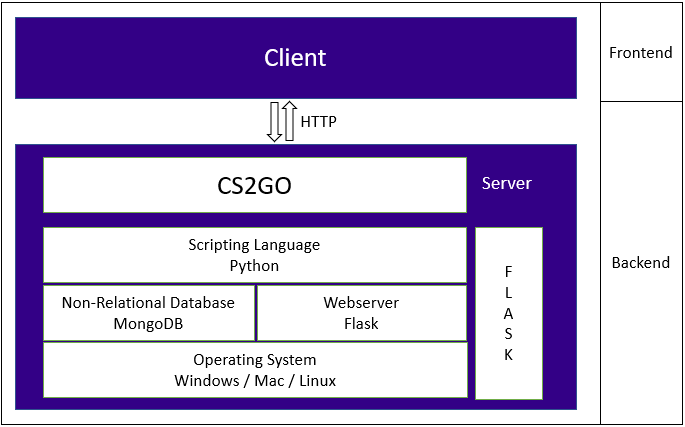


Figure 4: High Level diagram of system architecture

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 exists 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 the Style Guide is to create styles that are open/general; these styles can then be used on several pages as they weren’t meant for just one. This helped speed up the designing process of the pages and allowed for reusability.

The server side consisted of a single module, a single module was suitable due to the need of only a few routes to be served. This primary python file named main.py contains numerous functions.

The database layer is connected using 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.

### 4.2.2 Information Architecture

Another form of architecture used throughout the project was information architecture (IA). IA focuses on the organization 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. Connecting the user to the content they’re looking for was the goal. With IA in use various elements were focused on, such as, the end users and the data that will be presented through the website.

Two main factors were considered, cognitive science and mental models.

*Cognitive psychology*

Cognitive psychology which is 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 load [amount of information that a person can process at any given time] of the end user needed to be considered. Simplicity was highly considered within the design aspect, this resolved from inadvertently overloading a user with too much information all at once.

*Mental Models*

To answer these questions, the information architect must focus on a number of things: the target audience, the technologies related to the website, and the data that will be presented through the website. Because eLearning is 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 easier to discover when it is in a place that matches the users mental model of where it should be.

## 4.2 Visualization Design

Well-designed web applications offer much more than just aesthetics, they attract a user base and help people understand the product and branding through a variety of indicators, encompassing visuals, text, and interactions. There are several stages needed to take into 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 specifically refers to how closely it matches the look-and-feel of the final system. Wireframes are the backbones of a design, its purpose is 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.

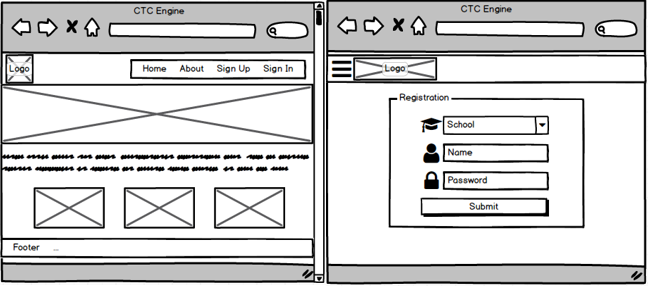


Figure 5: Wireframe of proposed Home page & User authentication page

Iterative development was also used in 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 in order to allow the best one to rise to the surface. Iterations were used to achieve a minimum viable product, until a satisfiable wireframe representation of the final design was achieved.

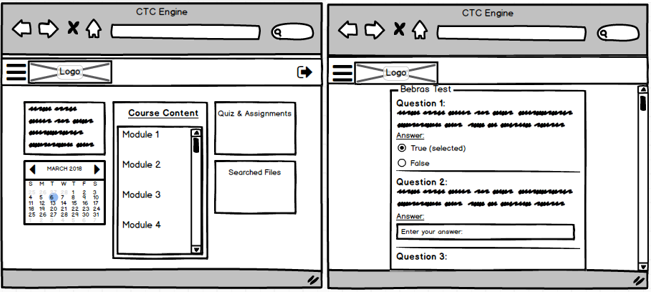


Figure 6: Wireframe of proposed Profile page & Test page

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) representation of the application, although these Hi-Fi representations were not used. These Hi-Fi representations are both time-consuming and expensive in nature, which were factors which needed to be highly considered throughout the duration of the development phase.

# **Chapter five: Evaluation**

## Summary

Evaluation is a necessary and important aspect in web design, faults are easy to overlook. There were three main topics that needed to be evaluated throughout the implementation phase, these include, 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 X.

|  |  |  |
| --- | --- | --- |
| **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 practises. |
| SR9 | Yes | The system should be easy (for someone with a CS background) to add lessons and content to it. |

## 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 it needed to be implemented due to most people nowadays using a wide range of devices. There were two main areas which were tested, Browser compatibility and Mobile Compatibility.

First off it was necessary to assess the website on various major browsers other than the one it was developed on, Google Chrome. This test was carried out and evaluated the visualisation engines performance in three areas as shown in the table below. Different browsers have different configurations and settings that your web page should be compatible with. Your web site coding should be cross browser platform compatible.

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **User Interface** | **Retrieval of Data** | **Functionality** |
| Google Chrome |  |  |  |
| Microsoft Edge (IE) |  |  |  |
| Mozilla Firefox |  |  |  |
| Safari |  |  |  |

[Talk about the results from the table above] - The website in terms of functionality worked flawlessly on all browsers. With regards to Mobile browser compatibility, there were a few locations on the UI were the page overflowed, although this did not affect the functionality or the usability of the application.

|  |  |  |  |
| --- | --- | --- | --- |
| **Phone** | **User Interface** | **Retrieval of Data** | **Functionality** |
| iPhone |  |  |  |
| Samsung |  |  |  |

## 5.3 User Testing

A beta version of the web application was deployed on Heroku’s cloud platform which enables developers to build and engage with the application entirely on the cloud. This version of the application was user tested by several individuals at Maynooth University, these individuals include lectures, a master student and several undergraduates. The tests that were conducted were on a sample of users who had no prior experience interacting with the system before. Their feedback was gathered via a questionnaire which was created on Google Forms which asked the user 5 simple yet effective questions.

[Link Google Forms]

## 5.4 Unit Testing

**Chapter six: Conclusion**

**Summary**

Chapter 5 identifies and discuss the implications of your work.

**5.1 Contribution to the state-of-the-art**

If you made a contribution to the state-of-the-art, clearly identify it here.

**5.2 Results discussion**

Discuss whether your results are general, potentially generalizable, or specific to a particular case. Identify threats to the validity of your results (e.g. limitations, risks introduced by your approach, etc.)

**5.3 Project Approach**

Discuss your project approach

**5.3 Future Work**

Discuss future work, based on what you have done (and not done)

# **References**

[1] PACT: An Initiative to Introduce Computational Thinking in to 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: <https://www.schoology.com/>

[3] <https://elearningindustry.com/online-learning-statistics-and-trends>

[4] <http://www.asbacademy.org/top-6-elearning-trends-watch-2017/>

[5]https://www.moodlebites.com/pluginfile.php/26295/mod\_resource/content/1/Pub4\_ComparisonBetweenMoodleAndBlackboard.pdf

[6] http://flask.pocoo.org/docs/0.12/

[7] https://api.mongodb.com/python/3.6.0/

[8] https://docs.mongodb.com/manual/core/gridfs/

[9] http://www.chartjs.org/docs/latest/

[10] https://d3js.org/

[11] Syed Fazle Rahman, 16 JavaScript Libraries for Creating Beautiful Charts, <https://www.sitepoint.com/15-best-javascript-charting-libraries/>

[12] <https://www.shiftelearning.com/blog/bid/331987/mobile-learning-stats-that-will-make-you-rethink-your-training-strategy> mlearning 2020 70billion

**Appendices**

Include here all extra material, e.g. your source code, project management (optional) including: the task list, Gantt Chart diagrams (or equivalent), discussion of any significant deviations from plan, and how you managed them, discussion of what you would do differently if you repeated the project.

## Appendix 1 Schematic of the hardware associated with this project.

## Appendix 2 Code developed for this project.

## Appendix 3 UML Class, Use Case and sequence diagrams for this project.

|  |
| --- |
|  |
| Appendix 4 Screen shots of the project implementation |
|  |