

E-Learning Web App for Second-Level Students

Interim Report

TU857

BSc in Computer Science Infrastructure

**Maya O’Connell**

**D20125203**

**Patricia O’Byrne**

School of Computer Science

Technological University, Dublin

**Date**

Abstract

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name

Date

Acknowledgments

Body text

Table of Contents

[1. Introduction 8](#_Toc120625667)

[1.1. Project Background 8](#_Toc120625668)

[1.2. Project Description 9](#_Toc120625669)

[1.3. Project Aims and Objectives 10](#_Toc120625670)

[1.4. Project Scope 10](#_Toc120625671)

[1.5. Thesis Roadmap 10](#_Toc120625672)

[2. Literature Review 11](#_Toc120625673)

[2.1. Introduction 11](#_Toc120625674)

[2.2. Alternative Existing Solutions to Your Problem 11](#_Toc120625675)

[2.2.1 Moodle 11](#_Toc120625676)

[2.2.2 Blackboard 12](#_Toc120625677)

[2.2.3 Vevox 13](#_Toc120625678)

[2.3. Technologies you’ve researched 15](#_Toc120625679)

[2.3.1 ReactJS 15](#_Toc120625680)

[2.3.2 JSX 16](#_Toc120625681)

[2.3.X Vue.js 16](#_Toc120625682)

[2.3.X Angular.js 17](#_Toc120625683)

[2.3.x Node.js 17](#_Toc120625684)

[2.3.x Express 17](#_Toc120625685)

[2.3.X PHP 17](#_Toc120625686)

[2.3.X Go 17](#_Toc120625687)

[2.3.x MongoDB 18](#_Toc120625688)

[2.3.x PostgreSQL 18](#_Toc120625689)

[2.4. Other Research you’ve done 18](#_Toc120625690)

[2.4.1 Profanity Detectors 18](#_Toc120625691)

[2.4.2 Impact of Student Response Systems 19](#_Toc120625692)

[2.4.3 Visual Studio Code 19](#_Toc120625693)

[2.4.4 GitHub 19](#_Toc120625694)

[2.5. Existing Final Year Projects 19](#_Toc120625695)

[2.5.1 An eLearning system for anonymous feedback sessions 19](#_Toc120625696)

[2.5.2 Edu-Play 20](#_Toc120625697)

[2.6. Conclusions 20](#_Toc120625698)

[3. System Design 20](#_Toc120625699)

[3.1. Introduction 20](#_Toc120625700)

[3.2. Software Methodology 20](#_Toc120625701)

[3.2.1 Waterfall Methodology 20](#_Toc120625702)

[3.2.2 Agile Development 21](#_Toc120625703)

[3.2.3 Scrum Methodology 22](#_Toc120625704)

[3.2.4 Feature Driven Development 22](#_Toc120625705)

[3.3. Overview of System 23](#_Toc120625706)

[3.2. Front End 25](#_Toc120625707)

[3.2.1 Use Case Diagrams 25](#_Toc120625708)

[3.2.2 Class Diagram 29](#_Toc120625709)

[3.2.2 Middle Tier 31](#_Toc120625710)

[3.2.3 Back End 33](#_Toc120625711)

[3.X. Conclusions 35](#_Toc120625712)

[4. Testing and Evaluation 36](#_Toc120625713)

[4.1. Introduction 36](#_Toc120625714)

[4.2. Plan for Testing 36](#_Toc120625715)

[4.2.1 User Experience Testing 36](#_Toc120625716)

[4.2.2 Software Testing (Black Box) 36](#_Toc120625717)

[4.2.3 Software Testing (White Box) 36](#_Toc120625718)

[4.3. Plan for Evaluation 36](#_Toc120625719)

[4.3.1 Usability Evaluation 36](#_Toc120625720)

[4.3.2 Responsive Design Evaluation 37](#_Toc120625721)

[4.4. Conclusions 37](#_Toc120625722)

[5. Prototype Development 38](#_Toc120625723)

[5.1. Introduction 38](#_Toc120625724)

[5.2. Prototype Development 38](#_Toc120625725)

[5.2.1 File Structure 38](#_Toc120625726)

[5.2.2 Front End Development 39](#_Toc120625727)

[5.3. Conclusions 41](#_Toc120625728)

[6. Issues and Future Work 42](#_Toc120625729)

[6.1. Introduction 42](#_Toc120625730)

[6.2. Issues and Risks 42](#_Toc120625731)

[6.2.1 Time Constraints 42](#_Toc120625732)

[6.2.2 Implementation of File Uploads 42](#_Toc120625733)

[6.2.3 Architecture Development 42](#_Toc120625734)

[6.2.4 Performance Risks 42](#_Toc120625735)

[6.3. Plans and Future Work 42](#_Toc120625736)

[Deliverable 43](#_Toc120625737)

[Date 43](#_Toc120625738)

[6.3.1. GANTT Chart 43](#_Toc120625739)

[Bibliography 45](#_Toc120625740)

Table of Figures

[Figure 1 Moodle Environment Course Overview (Mount Orange School, 2022) 13](#_Toc119660200)

[Figure 2 Moodle Environment Course Page (Mount Orange School, 2022) 13](#_Toc119660201)

[Figure 3 Blackboard Learn Courses (Try Blackboard | Blackboard, no date) 14](#_Toc119660202)

[Figure 4 Vevox Poll Question with displayed answers as session creator(Vevox, 2022) 15](#_Toc119660203)

[Figure 5 Vevox Q&A Board as session creator (Vevox, 2022) 15](#_Toc119660204)

[Figure 6 Vevox Joined Session (Vevox, 2022) 16](#_Toc119660205)

[Figure 7 Stack Overflow Most Popular Web Frameworks (Stack Overflow Developer Survey 2021, 2021) 17](#_Toc119660206)

[Figure 8 Waterfall Method (Sami, 2018) 22](#_Toc119660207)

[Figure 9 Agile Development Cycle (Santos, 2021) 22](#_Toc119660208)

[Figure 10 Scrum Methodology (Premium Vector | Agile development methodology. software developments sprint, develop process management and scrum sprints illustration, no date) 23](#_Toc119660209)

[Figure 11 Feature Driven Development (Mirzoyan, 2020) 24](#_Toc119660210)

[Figure 12 Three-tier Architecture(Pandey, 2018) 24](#_Toc119660211)

[Figure 13 System architecture 25](#_Toc119660212)

[Figure 14 Generic User Use Case Diagram 26](#_Toc119660213)

[Figure 15 Student User Use Case Diagram 27](#_Toc119660214)

[Figure 16 Teacher User Use Case Diagram 28](#_Toc119660215)

[Figure 17 UML Class Diagram 29](#_Toc119660216)

# 1. Introduction

## Project Background

In today’s world, eLearning platforms are a fundamental part of third-level education, with a study showing 99% of institutions have at least one Learning Management System in place (Dahlstrom, Brooks and Bichsel, 2014). However, during the initial school closures of the Covid pandemic, second-level teachers were tasked with implementing an online learning environment to replace the in-person classroom environment, with a study showing 90% of teachers had no or very limited experience with online learning environments (Li *et al.*, 2022). Teachers described how many schools did not have an e-learning platform in place pre-pandemic(McGuire, 2021), resulting in a scramble for them to find the most suitable solution.

Two of the most popular eLearning platforms are Blackboard(*Learning Management Systems (LMS) & Software | Blackboard*, no date) and Moodle(*Moodle - Open-source learning platform | Moodle.org*, no date). While these are both suitable solutions for second-level education, providing an online space for content uploading, submitting work, and participating in online quiz assessments. A feature students don’t have, is a way they can post questions during lessons. This feature which could benefit classroom interaction, by improving students’ comfortability with interaction in the classroom. Research concluded that about 95% of students who have questions do not ask their questions in the classroom (Dillon, 2010), due to factors such as fear of judgement, shyness, or social anxiety. Solutions to alleviate this issue may include question boxes, in which questions are left and answered during the next class. While this reduces the anxiety of asking questions, such a method may not be effective due to students receiving answers, when the information is no longer fresh in their minds (Chu, Li and Hsia, 2007).

This project will hopefully provide an eLearning platform to improve the e-readiness of secondary schools during times of unforeseen closures, as well as improve students’ comfort and engagement in the classroom.

## Project Description

Include a diagram

This project aims to create a reliable, easy-to-use eLearning platform to be integrated with the day-to-day routine of second-level education, following a similar use as that of third-level education, with various features for teacher/student to use, to aid them in their studies. The implementation of this eLearning platform within second-level education could prevent the pressure felt on both teachers and students in the case of future school closures, as it could provide ease with online learning, rather than either party feeling unprepared.

This project will be an eLearning web application where users can sign up as either a student or a teacher and create an account in the system. Teacher users will have more controls than a student users, such as having the ability to create a group, which they can be edit or delete, create polls and quizzes (which will have auto-correction), after which they can view the results, and broadcast them to the group. They can also upload teaching content, broadcasting it to the group. Signing up as a student user will automatically set the account to delete one year from its creation to minimize the storage of minors’ data. Students can view all this content, partaking in quizzes and polls.

In hopes of improving engagement in the classroom, a questions feature will be implemented, in which students can post questions they have throughout or after the class, reducing the anxiety of having all eyes on them while aiding the students to be able to learn on a deeper level. Alongside this, a private chat feature will also be implemented for students to directly message their teacher, with questions they may have outside of the classroom. To prevent bullying or unsociable behaviour on the app, a profanity detector will be implemented, to filter out any posts or messages which users try to post, with a three-strike warning in place before the user will be blocked from the application. Teacher users will be able to view the blocked users, allowing them to discuss further and hopefully resolve the issue with the student.

## Project Aims and Objectives

The aims and objectives of this project are:

* Research various literature sources, related to my project.
* Research modern eLearning platforms.
* Investigate and determine which technologies would be most suitable for this project.
* Design a three-tier architecture with a presentation layer, application layer, and database layer, and implement it.
* Develop a web application using the chosen technologies.
* Test the developed web application.

## Project Scope

This project will create an eLearning web application for secondary school use and will focus solely on making it suitable for second-level education, to increase engagement in the classroom while aiding students in comfortability with blended learning. This application is not intended to be used as any form of social media platform or any source of potential bullying. To target this, a profanity detector will be implemented, which will monitor any users trying to send inappropriate messages, which will inform the teacher to be dealt with. There was consideration taken as to whether to implement a feature to allow users to post anonymously, however it was ultimately decided for the safety of both students and teachers to extract this feature.

## Thesis Roadmap

One sentence explaining what each of the following chapters is about.

# 2. Literature Review

## 2.1. Introduction

In this chapter …

## 2.2. Alternative Existing Solutions to Your Problem

### 2.2.1 Moodle

Moodle is a free, online learning management system, which enables educators to create a private website, filled with dynamic courses. It enables the use of interactive features such as question forums, feedback surveys, and quizzes, allows for lesson content pages, and assignment uploads, as well as progress tracking and notifications, to aid in organization and ensure students are on track. (*Features - MoodleDocs*, no date)

Figures 1 and 2 display the web app on a laptop, however, the web app has a responsive layout, meaning it can be used on other devices, such as a tablet or smartphone. It has a clean layout, using neutral colours, with minimal distractions. This is highly suitable for an educational environment, to ensure users are engaged in their work.Graphical user interface, website

Description automatically generated

Figure 1 Moodle Environment Course Overview (Mount Orange School, 2022)

Graphical user interface, application

Description automatically generated

Figure 2 Moodle Environment Course Page (Mount Orange School, 2022)

### 2.2.2 Blackboard

Blackboard is a virtual learning environment, similar to Moodle, however, it is a paid software. The web app provides various features similar to Moodle. Teachers can use features such as creating and uploading content, previewing created content, broadcasting announcements, grading assignments, partaking in discussion forums, while students can use features such as viewing course content and announcements, partaking in discussion forums, viewing and uploading assignments, and taking tests. (*Feature Guide for the Blackboard App*, no date)

Figure 3 demonstrates the users' courses page, it follows a very similar layout to Moodle, except with a sidebar for user navigation, maintaining a clean layout. Like Moodle, this web app is responsive and can be used on other devices.

Graphical user interface, website

Description automatically generated

Figure 3 Blackboard Learn Courses (Try Blackboard | Blackboard, no date)

### 2.2.3 Vevox

Vevox is a live, online polling and Q&A system used to increase engagement in education and business environments. It allows users to anonymously vote in polls or ask questions, which can then be discussed during or after the meeting / class. The goal of this web app is to improve engagement, which it has been shown to do so according to (Waugh, 2018). This study completed in the University of Southampton found that 95% of lecturers who implemented Vevox during the lecture experienced and improvement in engagement, and showed whether the students had actually grasped the topic being discussed. They found that student participation with Vevox was nearly one hundred percent, as opposed to one or two hands being raised when asked a question verbally.

Vevox is easy to use, with a clean layout. As shown in figure 5 and figure 6 users who join a session have minimal distractions to ensure maximum engagement. It has a responsive design and can therefore be used with various devices such as tablets and smartphones.

Graphical user interface, text, application, website

Description automatically generated

Figure 4 Vevox Poll Question with displayed answers as session creator(Vevox, 2022)

Graphical user interface, text, application, email

Description automatically generated

Figure 5 Vevox Q&A Board as session creator (Vevox, 2022)

Graphical user interface

Description automatically generated

Figure 6 Vevox Joined Session (Vevox, 2022)

## 2.3. Technologies you’ve researched

### 2.3.1 ReactJS

ReactJS is an open-source JavaScript library for building user interfaces which was developed by Meta (then Facebook) in 2011. React is component-based, meaning that components built manage their state and can then be composed to make complex UIs (*React – A JavaScript library for building user interfaces*, no date). React allows users to easily create reusable components, which will reduce the development process. It also has a range of useful libraries open to the public such as Material UI, Bootstrap, and Formik.

Although there are many other JavaScript framework options such as Vue.js and Angular, React is the most commonly used web framework, according to (*Stack Overflow Developer Survey 2021*, 2021) 40.14% of respondents, therefore expanding knowledge of this technology is hugely beneficial. It is also the framework that I have the most experience in but wish to expand my knowledge further.

Chart, funnel chart

Description automatically generated

Figure 7 Stack Overflow Most Popular Web Frameworks (Stack Overflow Developer Survey 2021, 2021)

### 2.3.2 JSX

JSX is a syntax extension to JavaScript, which is used in developing a UI with React. It appears like a template language but has the power of JavaScript, combining the technologies into components(*Introducing JSX – React*, no date). This efficiently combines the development of JavaScript and HTML front end, which will improve my development process.

### 2.3.X Vue.js

Vue.js was developed in 2014 by Evan You (*Frequently Asked Questions | Vue.js*, no date). It is a JavaScript framework that provides a declarative and component based method alongside standard HTML, CSS, and JavaScript to develop user interfaces efficiently (*Introduction | Vue.js*, no date). Vue.js has two core features; declarative rendering and reactivity. Declarative rendering is when data can be rendered to the DOM with simple template syntax (Hunter, 2021). Reactivity is when a page updates automatically when changes occur in the DOM.

However, as Vue.js is relatively new, it does not have as large of a community – which could make it more difficult to learn as I have no experience with Vue.

### 2.3.X Angular.js

Angular.js was released by Google in 2010 (Hartman, 2020a). It is an open-source, JavaScript framework for developing the front end of dynamic web applications. It uses HTML and JavaScript, allowing users to extend the ability of HTML attributes using Angular built in attributes known as directives. It also uses data binding and dependency injections to reduce a lot of the code you would otherwise have to write (*AngularJS: Developer Guide: Introduction*, no date).

While Angular would work well with this application, the learning curve is steeper for Angular than React. While it would be good to expand my technology knowledge, in terms of time management, it may result in delays in other project development areas.

### 2.3.x Node.js

Node.js is an open-source, cross-platform JavaScript runtime environment designed to build scalable network applications, providing asynchronous event-driven execution (Node.js, no date). Node.js was developed and released by Ryan Dahl, designed to tackle the issues of the popular web servers at the time. Its non-blocking, single-threaded programming makes it lightweight and highly efficient. I have previously used Node.js but wish to dive deeper to further my understanding and improve.

### 2.3.x Express

Express is a minimal and flexible Node.js framework that provides a robust set of features for web and mobile applications, facilitating the rapid development of Node-based web applications(*Node.js - Express Framework*, no date). Express provides middleware to simplify making database queries, API calls, or preparing a response (Hombergs, 2022).

As my application requires a backend API the combined use of Node.js and Express will make implementing once easier.

### 2.3.X PHP

PHP is an open-source, server-side programming language which was launched in 1994 (‘What is PHP? Uses & Introduction’, 2014). PHP is embedded into HTML, aiding in implementing dynamic content, databases and session tracking (*PHP - Introduction*, no date). PHP allows developers to implement various functions, such as CRUD operations in the database, handling forms, page authorization, cookie implementation and data encryption (*PHP - Introduction*, no date).

While PHP would meet the functional needs of this project, the decrease of popularity in the technology in modern day, means its libraries are limited, and cannot compare to other modern technologies. As well as this, PHP’s execution time is slower than that of its competitors such as Node.js. This is due to it performing synchronous execution as opposed to modern technologies using asynchronous execution.

### 2.3.X Go

Go is a programming language with syntax similar to C, that was announced to the public in 2009. It was developed by Google cloud engineers Robert Griesemer, Rob Pike and Ken Thompson (Kolade, 2021). Go can be implemented as the middle tier of a web application easily, and is a great option as it is easy to learn due to it’s similarities between itself and the C++ syntax. As well as this, it slightly outperforms other technologies such as Node.js in speed. It also has a large community and libraries which hugely aid in the development process. However, while this is a great alternative, the benefits of this language are not great enough reason to take on a new language. The code would also be easier to maintain during the development process if JavaScript was implemented on the back end, as it will be on the front end using React.js.

### 2.3.x MongoDB

MongoDB is a free NoSQL database, released by MongoDB in 2009. It is a free, document database that stores data in flexible, JSON-like documents, within which fields can vary. It is a distributed database with high availability and horizontal scaling (*What Is MongoDB?*, no date). A MongoDB document is a structure made of fields and values – like key-value pairs, formatted similarly to JSON, BSON (Binary JSON). A set of documents is known as a collection – the equivalent of a table in a relational database. This database would be suitable for the application as it will require large amounts of file storage, and MongoDB is much faster for retrieval.

### 2.3.x PostgreSQL

PostgreSQL is an open-source, relational database (Group, 2022). It stores data as structured objects, following syntax and schemas, similar to SQL. Schemas contain tables, made up of columns and keys. The structure of these tables must be predefined, and the data must be formatted accordingly to be successfully stored (Smallcombe, 2022). PostgreSQL is suitable for applications which would require more complex queries, however this project will not require such. Also while the strictly structured data can often be beneficial as it gives clarity of your database, however should there be a need to scale, or need for flexibility, it can be a very time consuming process.

## 2.4. Other Research you’ve done

### 2.4.1 Profanity Detectors

Profanity is the use of offensive, vulgar or hateful language and is something that a study has found on up to 80% of online blogs. Social media platforms are also a breeding ground for cyberbullying to occur, with 19% of teenagers reporting that something mean or embarrassing had been written about them online (Chen *et al.*, 2012).

A profanity detector detects unwanted, hateful, sexual, and toxic content in user-generated text (*Profanity Detection and Moderation*, no date), which can be used to filter content of this type. Python libraries provide a library called profanity-check which implements a profanity detector using a dataset made up of both offensive and non-offensive tweets from Twitter and Wikipedia pieces. This dataset is then vectorized using sckit-learn CountVectorize class, counting how many times each word appears, resulting in what is known as a Bag of Words. It is then trained using the Linear Support Vector Machine model which determines the offensive language, and how offensive they are.(Zhou, 2022)

### 2.4.2 Impact of Student Response Systems

A Student Response System(SRS) is a method in which a lecturer can post questions to gather students’ responses during a lecture (*Student Response System (SRS): Information Technology - Northwestern University*, 2021). A study found that the use of an SRS system greatly improved class engagement, reporting increased participation, livelier classroom discussions, and improved view of their studies (Ulbig and Notman, 2012). Another study found that 94% of teachers felt that student engagement was increased and 87% found student participation to increase, while 69% of students felt the SRS systems increased their engagement and 70% believed it increased their participation (Kaleta and Joosten, 2007). These results determine that implementing another medium for students to respond in classrooms has a positive impact. This project will provide a medium that will thus improve student engagement and participation.

### 2.4.3 Visual Studio Code

Visual Studio Code is a free, open-source text editor created by Microsoft. It comes with built-in support for JavaScript, TypeScript, and Node.js, but has extensions for a huge variety of other programming languages, runtimes, environments, and clouds(Heller, 2022). It has many beneficial features including IntelliSense, which provides smart completions based on variable types, function definitions, and imported modules, as well as being integrated with Git, which I will discuss more below (*Visual Studio Code - Code Editing. Redefined*, no date).

### 2.4.4 GitHub

Founded in 2008, GitHub is a code hosting platform for version control and collaboration, letting users collaborate on projects from anywhere (*Hello World*, no date). It aids in project management as it allows users to manage and track changes in their code, while securely storing it in the cloud in a Git repository. As it is integrated into Visual Studio Code it will allow for repository creation, regular commits, pushes, and rollbacks easily, using the source control extension. Using GitHub will be beneficial for this project as any bugs which may occur, can easily be reversed by rolling back to a previous, working version of the application.

## 2.5. Existing Final Year Projects

### 2.5.1 An eLearning system for anonymous feedback sessions

Daniel Hogan, C10701531

This project is an eLearning system for lecturers and students in which students can anonymously ask questions during a lecture, as well as provide other useful lecture tools. A lecturer can create a session, in which they can create a quiz and view overall quiz results, post lecture slides, and monitor disruptive behaviour, by blocking disruptive users. Students can vote in polls, take quizzes, view their quiz results, and ask anonymous questions. It was implanted using AngularJS, Node.js, Express, Websockets, JSON, and MongoDB.

The complexity of this project comes from the creation of sessions using WebSockets, implemented using the 'socket.io' library. This allows for communication within this session (group of sockets) without notifying sockets that are not in this session.

### 2.5.2 Edu-Play

Richard J Power, 2015

This is an application designed for children in the 8-12 age range, to aid in their learning. It does this using various games which would return the score to the user instantly. The games included maths games, spelling games, and trivia games. A web application was also created for teachers or parents to update the contents of spelling or trivia games. It was implemented using HTML, PHP SQLite, CSS, and Java.

The complexity of this project comes from the incorporation of gamification of the design was a focus of the project, to ensure it was accessible to the target audience as possible. The user applied this, as well as gained feedback from the target demographic to ensure the best possible design.

## 2.6. Conclusions

This chapter presented the literature for the eLearning web app. It discussed research of existing solutions similar or relating to the web app, technologies required for the development of the web app, past final year projects similar or relating to the web app, as well as a range of other research relevant to the web app including academic papers and web information.

# 3. System Design

## 3.1. Introduction

## 3.2. Software Methodology

### 3.2.1 Waterfall Methodology

The Waterfall Methodology was the first process model to be introduced, established by Winston W. Royce in 1970 (Hoorey and Bottorff, 2022). It follows a sequential process of six steps, where each step flows from one to the next – similar to the flow of a waterfall. The next step of the process cannot begin until the previous step has been completed, so are no overlapping steps.

Diagram

Description automatically generated

Figure 8 Waterfall Method (Sami, 2018)

The Waterfall Model is beneficial as it is easy to understand and implement, with each stage clearly defined(*SDLC - Waterfall Model*, no date). This method is suited for smaller project, with clearly defined, fixed requirements. However due to the lack of flexibility of this methodology, I will not be using it in the development of this project as it may require room to evolve and grow.

### 3.2.2 Agile Development

Agile Development was introduced in 2001 when seventeen software developers and defined four principles for agile project management including; individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan (Sacolick, 2022). This development process was created as an alternative to the commonly used, inflexible Waterfall method.Chart, diagram

Description automatically generated

Figure 9 Agile Development Cycle (Santos, 2021)

Agile development follows an iterative process, encouraging continuous development and testing throughout the project’s development cycle (Nanda, 2021). This would be most suitable for the development of this project as it allows flexibility to adapt to changes which may occur in the process. The use of continuous testing ensures less risk of failure as you work bit by bit and makes it easier to locate what is causing any bugs or errors.

Agile development is not a methodology itself, but rather a base from which a range of methodologies follow, including the SCRUM methodology and Feature Driven Development methodology which will be discussed below.

### 3.2.3 Scrum Methodology

Scrum is one of the methodologies of Agile Development. It was first implemented in 1993 by Jeff Sutherland, John Scumniotales and Jeff McKenna, two of whom went on to create the Agile Manifesto(Kneafsey, 2015).

Logo

Description automatically generated

Figure 10 Scrum Methodology (Premium Vector | Agile development methodology. software developments sprint, develop process management and scrum sprints illustration, no date)

This methodology breaks a project down into smaller units of work known as sprints, which focus firstly on developing the most important features. These sprints may occur over two to four weeks, each resulting in a potentially deliverable product, which have been thoroughly tested. During the course of a sprint, daily meetings occur where the Scrum team update each other on their progress or issues encountered which could delay the development of the deliverables of that sprint. On the completion of a sprint, a sprint review occurs, where the deliverable is demonstrated and discussed with the client (*JIRA Scrum | Scrum Methodology - Javatpoint*, no date).

The Scrum methodology has many benefits such as dividing the project into smaller, more manageable tasks and the production of deliverable products at each sprint, however a Scrum tends to be implemented by a small team, consisting of a Scrum Master, a product owner and a developer team and would have to be adapted to be used for an individual project, therefore it will not be used for this project.

### 3.2.4 Feature Driven Development

Feature Driven Development is another of the methodologies of Agile Development. It was first applied in 1997 by Jeff De Luca (Lynn, no date). This methodology – as the name suggests, has a feature focused approach. It is a model-driven, short-iteration process developed around software engineering best practices such as domain object modelling, developing by feature, and code ownership (Mirzoyan, 2020).

Diagram

Description automatically generated

Figure 11 Feature Driven Development (Mirzoyan, 2020)

FDD has a five step process as shown in figure 11. Firstly an overall model is designed and developed. The model is then broken down into a list of the features which are required for the project. A plan will then be made of the order in which the features will be developed, considering factors such as difficulty, value or urgency of the feature, or estimated development times. The first feature is then designed and built, continuing the design and build steps for each feature in the list. The design, build phase occurs in bi-weekly cycles.

The feature focused approach of this methodology would be well-suited for this projects’ implementation, as it provides clear goals for each iteration. Breaking down the project into features will improve the efficiency of planning out the project timeline. It will also ensure flexibility in the development process due to its agile nature.

## 3.3. Overview of System

The project will implement a three-tier architecture. A three tier architecture breaks down an application into three layers; the presentation tier, the application tier and the database tier. Dividing the application into three parts, means each section can be altered or updated, without having an impact on the other tiers. Implementing a three-tier architecture also improves horizontal scalability, performance and availability (Contributor, 2021).

Diagram

Description automatically generated

Figure 12 Three-tier Architecture(Pandey, 2018)

The presentation tier is the tier in which an end user communicates with the application, using the user interface. Information is displayed from the application to the user on the user interface or gathered from the user through input from a mouse or keyboard and passed to the application. The front end of this application will be developed using React.js.

The application tier is the middle tier of the application. This tier is the core of the application as it contains the business logic to support the applications functions (Contributor, 2021). It will provide the functionality for users to register, login, upload files etc. This tier will be implemented using Node.js server with Express server-side framework running inside of it.

Finally, the database tier is the tier in which data taken from the presentation tier and using application tier methods, is stored in a database. This application will implement CRUD (Create, Read, Update, Delete) operations to add, retrieve, alter, and delete data from the database table. This application will use a MongoDB database for the database tier.

The architecture of this application is demonstrated in the diagram in figure 13 below, displaying the technologies used in each tier.

Diagram

Description automatically generated

Figure 13 System architecture

## 3.2. Front End

### 3.2.1 Use Case Diagrams

Use case diagrams are an effective method of demonstrating the ways a user interacts with the system. For this project, I created a use case diagram for three actors: a generic user, a student user and a teacher user.

Diagram

Description automatically generated

Figure Generic User Use Case Diagram

Figure 14 shows the use case for the generic user. The generic user demonstrates the interactions that both student and teacher users share.

The use case covers the following interactions:

* **Register**: This would occur on a user’s first interaction with the system, determining which user type they are, and storing their information into the database to create their account.
* **Login**: This is where a user would enter their details to access their account. The system will authenticate the user in order to log them in successfully.
* **View Content**: Once logged in, users can view the content page with available content.
* **View Poll**: Users can view the poll page, with available polls. They can also choose to view available poll results from this section.
* **View Quiz**: Users can view the quiz page, with available quizzes. They can also choose to view available quiz results from this section.
* **View Question**: Users can view questions posted on the questions page. They can also choose to respond to questions posted, which will be checked for profanity before posting successfully.
* **View Private Message**: Users can view their private messages. They can also choose to send a private message, which will be checked for profanity before being sent successfully.

Diagram

Description automatically generated

Figure Student User Use Case Diagram

Figure 15 shows the use case of a student user.

The use case of the student user covers the following interactions:

* **Join Group:** Students can join a group using the uniquely generated group id and associated password.
* **Vote in Poll:** Students can vote in polls made available to them.
* **Post Question:** Students can post a question on the questions page. Posts will be checked for profanity before being successfully posted.
* **Take Quiz:** Students can take quizzes made available to them.

Diagram

Description automatically generated

Figure Teacher User Use Case Diagram

Figure 16 demonstrates the use case for a teacher user.

The use case for the teacher user covers the following interactions:

* **Create Group**: A teacher user can create a group for their class, creating a name and password, which generates a unique id for students to use to join.
* **Manipulate Group:** Teacher users can edit a groups details and delete groups.
* **View Group users:** Teacher users can view all the members of the group. They can also choose to remove members.
* **Create Poll:** Teacher users can create a poll.
* **Close Poll:** Teacher users can close polls, to get the final results of the poll.
* **Post Poll results:** Teacher users can post the results of the polls to the classroom.
* **Create quiz:** Teacher users can create a quiz, adding questions and their answers.
* **Correct quiz:** Teacher users can review, and correct the quiz attempts made by their students, returning a final result.
* **Post quiz results:** Teacher users can post the results of the quizzes to the classroom.
* **Create content:** Teacher users can create content and post it to the classroom.
* **Manipulate content:** Teacher users can edit and delete content they created.

### 3.2.2 Class Diagram

Class diagrams are used to visualise the class structure of the system. A class is defined as a blueprint or set of instructions to build a specific type of object, determining how an object will behave and what the object with contain (Hartman, 2020). A class diagram demonstrates this, as well as how the classes relate to each other.

Timeline

Description automatically generated

Figure 17 UML Class Diagram

Figure 17 demonstrates the class structure of this application. The users’ class is a parent class, as it holds features that both user types will have, such as the various information required to create an account in the system. Both user types will also login. Both student and teacher classes are then child classes, which inherit these features and methods. Neither of these users will have unique features from each other, but they each have different methods, such as the various create element methods a teacher will have access to that the student will not. The student has its various methods, such as joining a group. The group class is represented in the diagram with a one to zero or many relationship, as without the teacher user, a group cannot exist. However, a teacher can also create many groups, but each group can only have one teacher creator. This relationship is also implemented for the content, messages, polls, quiz and questions classes, as without a group, none of these can exist, however a group can have many of each of these classes. This diagram will aid in the implementation of the system during the development phase.

## 3.2.2 Middle Tier

As discussed previously, the middle tier is what adds the functionality to connect the three tiers, completing tasks such as handling front end requests and manipulating data from the database. Sequence diagrams are used to demonstrate these operations, describing how they occur, and in what order (Tarika, 2017). These diagrams further flesh out the requirements of the systems features and will aid greatly in the development cycle.

Diagram

Description automatically generated

Figure 18 Sequence diagram for system login

Figure 18 above demonstrates the login process of a user. The diagram clearly shows the interactions from one layer of the application to another. Once the user enters their login details, a HTTP POST request will be made from the front end, which the middle ware will handle, making the appropriate query to the database to search through the collection to check for valid login details. Upon successful retrieval, the database will return the data to the middleware, which will send a success response to the front end. This finally results in the front end displaying the user homepage to the validated user.

Diagram

Description automatically generated

Figure 19 Sequence diagram for file upload

Figure 19 portrays the process of a user uploading a file on the web application, again clearly showing the way that each layer interacts with each other. The user selects the files to be uploaded, which are then sent in a HTTP POST request from the front end. This request is then handled by the middeware which performs an insert of the files into a mongodb collection. This results in a success message being passed from the back end to the front end and displaying to the user. This sequence diagram would work similarly for many other of the systems use cases, such as registration, posting questions, and creating quizzes and polls.

Diagram

Description automatically generated

Figure 20 Sequence diagram for sending private message

Figure 20 above displays the sequence diagram for sending a private message. This scenario is assuming the user has no previous chat history with the recipient. It demonstrates the user selects the message recipient, which causes the front end to make a HTTP POST request to create a new chat. The middleware then handles this request, by inserting a new document into the Chat collection, containing details about the chat, such as the chats unique id, the chat participants and the id of each message which is sent in the chat. This returns a successful response to the front end which then displays the chat to the user. Then the user can add a message to be sent, which causes the front end to sent a HTTP POST request again, along with the message data. The middleware then handles the request by inserting the message data and a unique message id into the message collection. This unique message id is also added to its associated chat document, as mentioned previously. A success response is finally returned to the front end.

## 3.2.3 Back End

MongoDB was the chosen database for this project. As Mongo is a NoSQL document database, the design process of a schema differs greatly from that of a relational database schema. While relational schema design is very structured and tends to be normalized – meaning there is no duplicated data, Mongo is a lot more flexible, without a formal design process or strict rules. Thus, rather than displaying the schema design in an Entity Relational Diagram, the general format of each collection will be described in the following paragraphs.

**Users Collection**

The user collection will store the data for each user registered on the system. Every user will have the following data to create an account:

* User id, of object id data type. This is a unique 12-byte value generated by Mongo.
* Username, of a string data type.
* Password, of a string data type.
* Email, of a string data type.
* User type, of a string data type.

However, documents then may differ depending on the user type, for example:

* A teacher user will create one or many groups, therefore they would require another attribute called groups of an array data type, to store the id of each group they have created.
* A student user has an automatic deletion date set to one year from account creation, therefore a deletion date attribute of date type would be needed.

**Groups Collection**

The groups collection will store the data for each group created on the system. Every group will have the following data to be created:

* Group id, of object id data type.
* Group name, of string data type.
* Group password, of string data type.
* Group members, of array data type. This will store the id of each user who has joined the group.
* Content, of array data type. This will store the id of every content upload.
* Quiz, of array data type. This will store the id of each quiz created.
* Poll, of array data type. This will store the id of each poll created.
* Question, of array data type. This will store the id of each question created.
* Chat, of array data type. This will store the id of each private chat within the group.

**Content Collection**

The content collection will store the data of any uploaded content. Each upload will have the following attributes:

* Content id, of object id data type.
* Content name, of string data type.
* Date of creation, of date type.
* Files, of array data type. This will store the data of each file uploaded.

**Quiz Collection**

This collection will store the data of quizzes created. Each quiz will have the following attributes:

* Quiz id, of object is data type.
* Quiz name, of string data type.
* Start date time, of string data type.
* End date time, of string data type.
* Questions, of array data type to store embedded data. Each question will have an id of integer type, along with the question.
* Automatic correction, of Boolean type.
* Quiz results, of array data type. This will store the user id of each user who took the quiz, and the result they received.

More attributes may be added depending on the quiz type. If the quiz is an MCQ, and attribute to hold an array of the answer options would be needed. If the quiz was to be autocorrected, an array to store the answers would be required.

**Poll Collection**

This collection will store data of polls created, which will have the following attributes:

* Poll id, of object id type.
* Poll question, of string type.
* Poll answer options, of array type. Each option will have an id and the option body.
* Poll results, of array type. Each option id will have its total result.

**Question Collection**This collection will store data of questions posted by users. Each question will have:

* Question id, of object id type.

Store questions as its own collection or embedded in user??

**Chat collection**

The data of each chat will be stored in the chat collection. Each chat will have:

* Chat id, of object id type.
* Participants, of array type. This will store the id of users in the chat.
* Messages, of array type. This will store the message id of each message within the chat.

**Messages Collection**

* Message id, of object id type.
* Message body, of string type.
* Sender, referencing the user id of the sender.
* Message date time, of string type.

## 3.X. Conclusions

# 4. Testing and Evaluation

## 4.1. Introduction

## 

## 4.2. Plan for Testing

### 4.2.1 User Experience Testing

Ensuring a positive user experience is a key component in creating a successful application. User experience testing is a method used to determine how a user feels about an application, whether it is easy to use, user friendly, whether it does what it is meant to do – things which determine whether they would likely return to use the application (Hamilton, 2020). To perform user experience testing, a testing environment of the web app will be set up to be tested by users. Following this, a survey will have been preprepared for them to partake in, asking them various questions on their experience with the application.

### 4.2.2 Software Testing (Black Box)

The ability of an application to do what it is meant to do is crucial to an application success. Software testing allows for applications to be tested for bugs, it’s performance and to ensure it is working as it is intended to. I intend to perform software testing using a manual black box technique. Black box testing is where the testing user has no knowledge of the internal code, focusing mainly on the input and output of the application (Hamilton, 2020a). To perform this test, test cases will be created, determining the requirement to be tested with their expected outcome. These will then be executed by test users, with the actual outcome to be recorded.

### 4.2.3 Software Testing (White Box)

A white box method of software testing will also be performed on the application. White box testing is where the internal code and structure of a program are evaluated (Vijay, 2015). The method of white box testing which will be performed is unit testing. Unit testing is a method which entails testing individual units or components of the application, such as a function, methods or module (Hamilton, 2022). This testing is done continually through the development cycle, to ensure that the code produced is performing as it is meant to. Implementing unit testing minimises the risk of bugs in later stages of the development cycle.

## 4.3. Plan for Evaluation

### 4.3.1 Usability Evaluation

Usability is defined as ‘a quality attribute that assess how easy user interfaces are to use’ according to (Nielsen, 2012) and is further described to be essential for the survival of a web application. If an application is difficult to use, does not do what it is meant to do, or is unclear of its purpose to the user, they will stop using this application. Therefore, ensuring the usability of a web application is essential. This evaluation can be done using Nielson’s Usability Heuristics, which are 10 general principles which define a usable user interface. This evaluation will occur towards the end of the lifecycle.

### 4.3.2 Responsive Design Evaluation

A responsive web application is hugely important in today’s world of various smart devices. Ensuring an application can work on any device means greater user reach, as not every person may have a laptop or PC – this is also highly important to consider as this application is intended for secondary school students. To perform responsivity evaluation, the application will be evaluated on a variety of browsers to ensure the user interface display is consistent and correct throughout. As well as this, the Google Chrome developer tool will be used to demonstrate the application user interface on various mobile and tablet devices.

## 4.4. Conclusions

# 5. Prototype Development

## 5.1. Introduction

Most of this project thus far has been research to develop a well-structured system design to ensure an efficient development cycle. Alongside this research, a prototype has been developed. The prototype development focused on the integration of the technologies required for this web application – React, Node, Express and MongoDB. It also tackles the feature of file uploading and display, as it is an area which was least familiar. The implementation of these areas will be discussed further in the following sections.

## 5.2. Prototype Development

As discussed in previous sections, the technologies which will be used to implement this web application are MongoDB as the database, Express and Node as the back end and React as the front-end. The combined use of these technologies forms what is known as a MERN stack, easily implementing the three-tier architecture, as demonstrated in Figure 13 of section 3.3. The React front end makes up the presentation layer where the user will interact with the system. The Node and Express back-end act as the middle tier, connecting the three tiers, and adding functionality to the application. Finally, the MongoDB database acts as the database tier where data is stored to be used for CRUD operations.

### 5.2.1 File Structure

The front end of the prototype was developed using React. As previously mentioned, React is a component based method for creating single page application user interfaces. The application was implemented used Visual Studio’s code editor. Creating a web application is enabled by the use of Reacts’ Create React App, which sets up a development environment for the user(*Create a New React App – React*, no date). It creates a hierarchy of various files and folders, which can be used in developing your application. This method was used in getting started with the React front end.

Text

Description automatically generated

Figure Front end file structure

Upon the creation of the web application by Create React App, various files which were not required were deleted and new files required for the prototype development were created, resulting in the file structure show in Figure 18. The src file holds the main components of the front end, such as the App.js file which is the main component of the application, and will contain the various other app components to be rendered. The index.js file imports and renders the App.js file, and the various CSS files style their associated files. Three sub folders were also added to the src folder – components folder, pages folder and styles folder. The components folder stores components that which have created, in this case the Navbar component. The pages folder contains as the name suggests, the various pages the web application has – at this point it only has a file upload page, a home page and a view file page. Finally, the styles folder contain the CSS files for styling each page – at this point there is only a CSS for the file upload page, as the layout design has not been a key feature of the prototype development. Other files and folders which were created by the React application and are still required include the node modules file which contains the various dependencies required to run a React web application, the public folder which contains the HTML file which serves the application, as well as the manifest JSON file which stores metadata.

Text

Description automatically generatedThe development of the back end of the web application included the use of a MongoDB database with Mongoose, combined with Node.js and Express as the middleware. The backend file structure is as shown in figure 19 – a main folder called server, with multiple sub folders and files. The models folder contains what are known as mongoose models. These are files which define the structure of a document and aid in creating and reading from a MongoDB database. There is a node\_modules folder containing the various dependencies required for the backend. The uploads folder stores files which are uploaded from the front end. The last file in the server folder is the index.js file – this is where the functionality that brings the front end and back end together occurs. This file deals with requests from the front end, connecting with the database and manipulating database data.

Figure Back end file structure

### 5.2.2 Front End Development

As the focus of the prototype development was on functionality – specifically focusing on the integration of the technologies and the file uploading feature, the front-end is very limited in appearance. Using various frameworks, such as React Bootstrap, simple components were added such as buttons, navbars and file selectors as show in figure 20. The file selector and save button allow a user to match a HTTP POST request to send their selected files to be stored, and the file data sent to the database. The show button then allows a user to make a HTTP GET request, to retrieve the names of the stored files from the database, which are displayed on the front end, also shown in figure x.

Graphical user interface, text, application, email

Description automatically generated

Figure Contents page

The user selects the file(s) they wish to upload with the file selector, which triggers the onChange event handler to call the saveFile function. In this function, the object which contains the files is looped through, to return each individual file into an array. This is done using React’s State Hook setting function. This is shown in figure 21 below.

A screenshot of a computer

Description automatically generated with medium confidence

Figure Save files function

Figure 22 below, then shows the function called by the onClick event of the Save button. This creates a form data object, which can be used to send key, value pairs in a HTTP request. Each of the files from the previously set file array state, is then appended to the form data object to be sent. A HTTP POST request is then made using the Axios library, along with the form data to the back end – which will be discussed in section 5.3.3. Once completed the user can display the name of each uploaded file, by selecting the show button. This calls the function shown in Figure 26, making a HTTP GET request to the back end, and adds the returned data to an array. Each item in the array is then returned as a link element as shown in Figure 23, using the JavaScript map function.

Text

Description automatically generated

Figure 25 Upload file function

A screenshot of a computer

Description automatically generated with medium confidence

Figure Show file function

Selecting a filename will route the user to a page to view that file, implementing using React routers useParam hook, allowing you to pass parameters over a URL. This parameter is retrieved on this page, and used to access the file from the /uploads folder to be displayed to the user, as shown in Figure 27 below.

Graphical user interface, application

Description automatically generated

Figure View file page

### 5.2.3 Database Creation

Creating a MongoDB database takes only a few steps, using MongoDB Atlas. This is a fully-managed cloud database which allows you to deploy, run and scale MongoDB in the cloud (MongoDB, no date).

### 5.2.3 Back End Development

## 5.3. Conclusions

# 6. Issues and Future Work

## 6.1. Introduction

## 6.2. Issues and Risks

### 6.2.1 Time Constraints

This project is due to be completed at the end of March, and there remains a large chunk of research, development, testing and presentation preparation to be completed. Ensuring to stick to the project plan defined in Figures 26 and 27 is essential to meeting this deadline. Effective time management is key to meeting deadlines, which can be achieved through the use of daily routines, and prioritizing tasks efficiently. This will aid in ensuring the projects successful outcome.

### 6.2.2 Implementation of File Uploads

Uploading content is a key feature of the eLearning web app, and something which I had not had previous experience with. Often areas which are unknown cause more stress or anxious feelings, and can often lead to avoiding the task, only to have to tackle it later in the cycle, likely in a higher stress period. To avoid this, the implementation of the feature began in the prototype development. This allowed for plenty of time to overcome the challenges faces with the development of this feature.

### 6.2.3 Architecture Development

Implementing the architecture was challenging during the beginning of the prototype development. Integrating the ReactJS front end, Nodejs middleware and MongoDB backend took some time. As I had previously worked with React and Node, I jumped straight into implementation, however quickly realized a deeper understanding of the technologies was required. Through research into the implementation of each of the technologies, a seamless integration of them was achieved.

### 6.2.4 Performance Risks

There are many areas which could results in performance risks in the application. An increase in the data stored in the database could result in a decrease in database performance. To alleviate this risk, implementing a schema design using MongoDB best practices is important. The implementation of other features of the application also potentially cause performance risk, therefore monitoring the performance of the application during the development cycle is essential.

## 6.3. Plans and Future Work

During the coming months, the application will be further developed to incorporate the remaining features. The order in which each feature will be implemented is outlined below:

1. **Register/Login system:** This is a fundamental feature in the application, as the application is not useful without its users. This phase of development will include the development of authorization, using JWT, as well as hashing of passwords to be securely stored in the database. It will include user management features such as editing user profile and deleting the account.
2. **Group creation:** Another fundamental application feature, the groups are required to create classrooms which can be safely accessed by users. This is where all the content, quizzes, polls etc will be available, each of which will require a group that it belongs to. This development phase will include the management features of the group, such as editing the group, deleting the group and group user management.
3. **Questions feature:** The next feature which will be implemented will be the questions feature. This feature will be implemented next as it requires the integration of a Python library – profanity-check, which was discussed in previous sections. As I have never integrated a python library with a web application before, it will be implemented earlier on in the development cycle, allowing plenty of time to overcome possible issues.
4. **Quiz feature:** The next feature to be developed will the quiz feature. This will include the creation of the quiz, the taking of the quiz, the correction of the quiz and the quiz results display.
5. **Polling feature:** The polling feature is placed at the stage of the development cycle for similar reasons as the quiz feature – it is an important feature but is not anticipated to be difficult to implement. This development phase will include the creation of the poll, the poll voting and the polling results display.
6. **Private chat:** The private chat feature is at a later stage of the development plan as it is not quite as essential as the other features.
7. **Blocking users:** The final feature to be implemented will be the blocking of users attempting to send inappropriate messages. This feature will be implemented last as while it is a good feature, it is not essential for the application – it is more of an add on.

Below I will outline the key deliverables of this project:

|  |  |
| --- | --- |
| Deliverable | Date |
| Finalised Project Proposal Submission | 23rd October 2022 |
| Interim Report Submission | 09th December 2022 |
| Interim Prototype Submission | 09th December 2022 |
| Interim Report Presentation | TBA |
| Web Application Submission | 31st March 2022 |
| Project Dissertation Submission | 31st March 2022 |
| Project Demonstration and Presentation | April 2022 |

### 6.3.1. GANTT Chart

The GANTT chart shown in figure 23 outlines the project plan for the lifecycle of this project. It displays the time required to be spent on each part of the cycle for the application to be successfully completed for the end of March.

Chart

Description automatically generated

Figure GANTT Chart Semester 1

Chart

Description automatically generated

Figure GANTT Chart Semester 2

# Bibliography

*AngularJS: Developer Guide: Introduction* (no date). Available at: https://docs.angularjs.org/guide/introduction (Accessed: 13 November 2022).

Chen, Y. *et al.* (2012) ‘Detecting Offensive Language in Social Media to Protect Adolescent Online Safety’, in *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*. *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pp. 71–80. Available at: https://doi.org/10.1109/SocialCom-PASSAT.2012.55.

Chu, K.-K., Li, M.-C. and Hsia, Y.-T. (2007) ‘Not Afraid to Ask’, in *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*. *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, pp. 600–604. Available at: https://doi.org/10.1109/ICALT.2007.194.

Contributor, T. (2021) *What is a 3-Tier Application Architecture? Definition from SearchSoftwareQuality*, *SearchSoftwareQuality*. Available at: https://www.techtarget.com/searchsoftwarequality/definition/3-tier-application (Accessed: 12 November 2022).

*Create a New React App – React* (no date). Available at: https://reactjs.org/docs/create-a-new-react-app.html (Accessed: 22 November 2022).

Dahlstrom, E., Brooks, D.C. and Bichsel, J. (2014) ‘The Current Ecosystem of Learning Management Systems in Higher Education: Student, Faculty, and IT Perspectives’, p. 27.

Dillon, D.J.T. (2010) ‘A Norm against Student Questions’, *The Clearing House: A Journal of Educational Strategies, Issues and Ideas* [Preprint]. Available at: https://www.tandfonline.com/doi/abs/10.1080/00098655.1981.10113676 (Accessed: 23 October 2022).

*Feature Guide for the Blackboard App* (no date). Available at: https://help.blackboard.com/Blackboard\_App/Feature\_Guide (Accessed: 4 November 2022).

*Features - MoodleDocs* (no date). Available at: https://docs.moodle.org/311/en/Features (Accessed: 4 November 2022).

*Frequently Asked Questions | Vue.js* (no date). Available at: https://vuejs.org/about/faq.html#who-maintains-vue (Accessed: 13 November 2022).

Group, P.G.D. (2022) *PostgreSQL*, *PostgreSQL*. Available at: https://www.postgresql.org/ (Accessed: 15 November 2022).

Hamilton, T. (2020) *What is Usability Testing? Software UX*. Available at: https://www.guru99.com/usability-testing-tutorial.html (Accessed: 25 November 2022).

Hamilton, T. (2022) *Unit Testing Tutorial – What is, Types & Test Example*. Available at: https://www.guru99.com/unit-testing-guide.html (Accessed: 29 November 2022).

Hartman, J. (2020a) *Angular Version List & History – Angular 2,4,5,6,7,8*. Available at: https://www.guru99.com/angularjs-1-vs-2-vs-4-vs-5-difference.html (Accessed: 13 November 2022).

Hartman, J. (2020b) *What is Class and Object in Java OOPS? Learn with Example*. Available at: https://www.guru99.com/java-oops-class-objects.html (Accessed: 15 November 2022).

Heller, M. (2022) *What is Visual Studio Code? Microsoft’s extensible code editor*, *InfoWorld*. Available at: https://www.infoworld.com/article/3666488/what-is-visual-studio-code-microsofts-extensible-code-editor.html (Accessed: 8 November 2022).

*Hello World* (no date) *GitHub Docs*. Available at: https://ghdocs-prod.azurewebsites.net/en/get-started/quickstart/hello-world (Accessed: 8 November 2022).

Hombergs, T. (2022) *Complete Guide to Express Middleware*. Available at: https://reflectoring.io/express-middleware/ (Accessed: 6 November 2022).

Hoorey, L. and Bottorff, C. (2022) *What is Waterfall Methodology? – Forbes Advisor*. Available at: https://www.forbes.com/advisor/business/what-is-waterfall-methodology/ (Accessed: 9 November 2022).

Hunter, J. (2021) ‘Vue.js Declarative Rendering’, *GeeksforGeeks*, 9 February. Available at: https://www.geeksforgeeks.org/vue-js-declarative-rendering/ (Accessed: 13 November 2022).

*Introducing JSX – React* (no date). Available at: https://reactjs.org/docs/introducing-jsx.html (Accessed: 6 November 2022).

*Introduction | Vue.js* (no date). Available at: https://vuejs.org/guide/introduction.html (Accessed: 13 November 2022).

*JIRA Scrum | Scrum Methodology - Javatpoint* (no date) *www.javatpoint.com*. Available at: https://www.javatpoint.com/jira-scrum (Accessed: 9 November 2022).

Kaleta, R. and Joosten, T. (2007) ‘Student Response Systems: A University of Wisconsin System Study of Clickers’, p. 12.

Kneafsey, S. (2015) *A Short History Of Scrum*, *TheScrumMaster.co.uk*. Available at: https://www.thescrummaster.co.uk/scrum/short-history-scrum/ (Accessed: 9 November 2022).

Kolade, C. (2021) *What is Go? Golang Programming Language Meaning Explained*, *freeCodeCamp.org*. Available at: https://www.freecodecamp.org/news/what-is-go-programming-language/ (Accessed: 15 November 2022).

*Learning Management Systems (LMS) & Software | Blackboard* (no date). Available at: https://www.blackboard.com/en-eu/teaching-learning/learning-management (Accessed: 23 October 2022).

Li, S. *et al.* (2022) ‘E-Learning during COVID-19: perspectives and experiences of the faculty and students’, *BMC Medical Education*, 22(1), p. 328. Available at: https://doi.org/10.1186/s12909-022-03383-x.

Lynn, R. (no date) *What is FDD in Agile?*, *Planview*. Available at: https://www.planview.com/resources/articles/fdd-agile/ (Accessed: 12 November 2022).

McGuire, P. (2021) *How online learning affected secondary schooling*, *The Irish Times*. Available at: https://www.irishtimes.com/news/education/how-online-learning-affected-secondary-schooling-1.4563307 (Accessed: 23 October 2022).

Mirzoyan, V. (2020) *Why should you use a feature-driven development?* Available at: https://aist.global/en/use-a-feature-driven-development (Accessed: 9 November 2022).

MongoDB (no date) *MongoDB Atlas Tutorial | MongoDB*. Available at: https://www.mongodb.com/basics/mongodb-atlas-tutorial (Accessed: 29 November 2022).

*Moodle - Open-source learning platform | Moodle.org* (no date). Available at: https://moodle.org/ (Accessed: 23 October 2022).

*Mount Orange School* (2022). Available at: https://school.moodledemo.net/ (Accessed: 12 November 2022).

Nanda, V. (2021) *Agile Methodology and Model*. Available at: https://www.tutorialspoint.com/agile-methodology-and-model (Accessed: 9 November 2022).

Nielsen, J. (2012) *Usability 101: Introduction to Usability*, *Nielsen Norman Group*. Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/ (Accessed: 29 November 2022).

Node.js (no date) *Node.js*, *Node.js*. Available at: https://nodejs.org/en/ (Accessed: 6 November 2022).

*Node.js - Express Framework* (no date). Available at: https://www.tutorialspoint.com/nodejs/nodejs\_express\_framework.htm (Accessed: 6 November 2022).

Pandey, S. (2018) ‘THREE TIER ARCHITECTURE : THE BEGINNING’, *CoffeeTechAndMe*, 11 February. Available at: https://medium.com/coffeetechandme/three-tier-architecture-the-beginning-2d2f6063fa1e (Accessed: 12 November 2022).

*PHP - Introduction* (no date). Available at: https://www.tutorialspoint.com/php/php\_introduction.htm (Accessed: 15 November 2022).

*Premium Vector | Agile development methodology. software developments sprint, develop process management and scrum sprints illustration* (no date) *Freepik*. Available at: https://www.freepik.com/premium-vector/agile-development-methodology-software-developments-sprint-develop-process-management-scrum-sprints-illustration\_8636934.htm (Accessed: 9 November 2022).

*Profanity Detection and Moderation* (no date) *Sightengine*. Available at: http://sightengine.com/docs/profanity-detection-hate-offensive-text-moderation (Accessed: 8 November 2022).

*React – A JavaScript library for building user interfaces* (no date). Available at: https://reactjs.org/ (Accessed: 4 November 2022).

Sacolick, I. (2022) *A brief history of the agile methodology*, *InfoWorld*. Available at: https://www.infoworld.com/article/3655646/a-brief-history-of-the-agile-methodology.html (Accessed: 9 November 2022).

Sami, M. (2018) *The Waterfall Model, a different perspective*. Available at: https://melsatar.blog/2018/02/16/the-waterfall-model-a-different-perspective/ (Accessed: 9 November 2022).

Santos, J.M.D. (2021) *Best Agile Development Methodology & Principles for 2022*. Available at: https://project-management.com/10-key-principles-of-agile-software-development/ (Accessed: 9 November 2022).

*SDLC - Waterfall Model* (no date). Available at: https://www.tutorialspoint.com/sdlc/sdlc\_waterfall\_model.htm (Accessed: 9 November 2022).

Smallcombe, M. (2022) *Mongodb vs. PostgresSQL: Compare Database Structure*, *Integrate.io*. Available at: https://www.integrate.io/blog/mongodb-vs-postgresql/ (Accessed: 15 November 2022).

*Stack Overflow Developer Survey 2021* (2021) *Stack Overflow*. Available at: https://insights.stackoverflow.com/survey/2021/?utm\_source=social-share&utm\_medium=social&utm\_campaign=dev-survey-2021 (Accessed: 4 November 2022).

*Student Response System (SRS): Information Technology - Northwestern University* (2021). Available at: https://www.it.northwestern.edu/srs/ (Accessed: 8 November 2022).

Tarika, S. (2017) ‘Unified Modeling Language (UML) | Sequence Diagrams’, *GeeksforGeeks*, 27 October. Available at: https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/ (Accessed: 26 November 2022).

*Try Blackboard | Blackboard* (no date). Available at: https://www.blackboard.com/en-uk/try-blackboard (Accessed: 12 November 2022).

Ulbig, S.G. and Notman, F. (2012) ‘Is Class Appreciation Just a Click Away?: Using Student Response System Technology to Enhance Shy Students’ Introductory American Government Experience’, *Journal of Political Science Education*, 8(4), pp. 352–371. Available at: https://doi.org/10.1080/15512169.2012.729450.

*Vevox* (2022). Available at: https://vevox.app/#/ (Accessed: 12 November 2022).

Vijay (2015) *White Box Testing: A Complete Guide with Techniques, Examples, & Tools*, *Software Testing Help*. Available at: https://www.softwaretestinghelp.com/white-box-testing-techniques-with-example/ (Accessed: 29 November 2022).

*Visual Studio Code - Code Editing. Redefined* (no date). Available at: https://code.visualstudio.com/ (Accessed: 8 November 2022).

Waugh, B. (2018) *4 statistics on student response systems*. Available at: https://www.vevox.com/blog/4-statistics-that-show-the-impact-of-student-response-systems (Accessed: 4 November 2022).

*What Is MongoDB?* (no date) *MongoDB*. Available at: https://www.mongodb.com/what-is-mongodb (Accessed: 6 November 2022).

‘What is PHP? Uses & Introduction’ (2014) *Code Institute IE*, 10 June. Available at: https://codeinstitute.net/ie/blog/what-is-php-programming/ (Accessed: 15 November 2022).

Zhou, V. (2022) ‘profanity-check’. Available at: https://github.com/vzhou842/profanity-check (Accessed: 8 November 2022).