Final Report May 2018

Mark Kenny | BSHCE (Part-time) | May 2018

Document Control

Revision History

Date	Version	Scope of Activity	Prepared	Reviewed	Approved

Distribution List

Name	Title	Version
Dominic Carr	Supervisor	
Dominic Carr	Lecturer	
Mark Kenny	Developer	

Related Documents

Title	Comments
Class Diagram	Contained In System Architecture
Use Case Model	Contained in Functional Requirements
Mockups	Containted in GUI

Executive Summary

The purpose of this project is to develop and online e-learning application that is used to train professional staff to that respective company's standards. The application is called Developmint. The application consists of a number of different pages that are designed to help staff develop in order to aid their progression through a business. The application allows an administrator to create and design courses in an easy to use and minimalist interface. Deveopmint provides administrators and trainers the ability to create tests after the course materials which can show if the trainee has an understanding of the modules. And allows them to put the theory into practice.

Developmint allows administrators to create course materials in a manner that keeps all aspects of formatting and tidying up to a minimum. It makes the building of these modules straightforward and uncomplicated. Administrators can upload all materials, tests and assign them both together to create the course. Anyone who attempts the tests will be presented with results straight afterwards, which get stored.

Contents

1 Introduction

1.1 Dictionary

- AWS – Amazon Web Services, the site were the application is hosted
- HTML – Hypter Text Markup Language, the basic language that websites are designed in
- CSS – Cascading Style Sheets. This a language used to style your website
- Javascript – Another language that assists in making interactive websites
- Bootstrap – A framework used to style your website
- PHP – A hypertext processor that allows commands to be embedded into HTML.
- LAMP – An acronym for Linux, Apache MySQL and PHP. It is a stack that is installed as your web server and is used to maintain your website
- Linux – Operating system, for this project specifically, it will be mentioned in the context of a Linux Ubuntu server
- UML – Unified Modelling Manluage
- DOM – Document Object Model

1.2 Background

My whole career path thus far has been in sales and marketing. I started working as a door to door sales rep and since then have moved into an office environment. I am now a team leader of a small sales and customer service team in a field marketing company. Our company works on behalf of a lot of bigger firms and because of this, extensive product knowledge is needed to work on behalf of these clients.

Upon joining the company, depending on the department, you can spend anywhere between one day two a week in training and there can be a lot to take in. I imagine it is the same within other firms.

What I have noticed, especially since I have begun managing teams, is that certain information gets forgotten. This is especially true if a particular scenario hasn't occurred in a long time. After a while I started to issue random tests to keep the sales and customer service teams' product knowledge fresh in their heads.

It was at that point that I began to wonder if a centralised hub for everyone to log on where they can take part in quizzes would be easier than handing out paper tests.

For career progression, sessions with the training manager would have to be arranged where you would sit with him for an hour and go through different management principles. With one training manager conducting sessions with potentially seventy head office staff, the times between training and coaching sessions could be consistently planned out.

I decided it would be easier, again, to have a centralised log in for employees who are on a development program where they could access all the materials relevant to them.

1.3 Technologies

As the project is a web application, it will be developed using a combination of different languages. As stated at the start of this final year, I used Cloud9 for the development of the site, however it was AWS's version of the IDE, which I have explained in more detail in the Implementation feature of this document.
There will be a database to store user data as well as material uploaded. The data base was a MYSQL data based that was installed as part of the LAMP service installed on the Ubuntu server. As mentioned at the start of the year, I used Phpmyadmin to main the database.
I also implemented use of AWS' EC2 instance to host my application, this is again described in more detail in the implementation section of this report.
I the languages used to develop the application were:

- HTML5
- PHP
- Javascript
- MySQL
- Linux commands (server side)

1.4 Purpose

The purpose of this document is to set out the requirements for the development of Developmint, a web application for professional development.
The intended customers are business who are looking for additional resources to develop their staff with online content and progression tracking.

1.5 Project Scope

The scope of the project is to develop an online web application that allows professionals to access training material so they can develop. The system shall have a specific course structure with exams and quizzes. Course content will be blocked to certain users depending on their professional ranking.
The courses will be designed by the company trainer and management using a drag and drop style function.
The application will only be accessible based on the user's current IP address range, which will be predetermined by the administrator. Users will not register their account; it will be created through a form filled in by the administrator and the user will be emailed their details.

1.6 Aims

The aims of this project are to create an online web application that allows trainers to create E-Learning platforms in a simple and easy to use manner. The interface is minimalist which allows for the easiest flow possible.

A key portion of this website is that it is not specific to any one area of learning and development, all materials can be created on the site.

In this section, the different requirements shall be described for Developmint. The aim of the application will be to provide an uncomplicated interface for trainers and trainees to use. The objective of the course design portion of the application is to allow trainees to easily create course work through a 'drag and drop' style interface. The trainees will be able to view their coursework in a clean interface.

1.7 Core Functional Component – Course & Quiz Design

The main component of the application that will prove to be the most complicated will be the course design functionality. The aim of the course design is for it to be uncomplicated and user friendly. Essentially, with little to no training, the administrator should have a decent idea of how he or she will be able to design a course.
The plan is to implement a drag and drop style design user interface, where the administrator can drag an object into an area and plot out the points of the course without the constant need to fill out text fields and forms. The problem lies in creating some sort of constant. So, for example, if there is a text document that needs to be read, there will have to be some sort of object or shape that will contain the document, can then be dragged into the course design interface and then send this document to be stored on the database once the course has been created. The solution can be to use a JQuery library called Draggables. This allows you to create shapes and objects that can be dragged and dropped on any DOM element, then use PHP to send the data to the database.
I have a similar plan for the quiz creation. However, this will have an easier solution. There is a Javascript UX code package of the same name Draggables. This allows a user to take a prepopulated form, and can drag it into another area which will then fill out another form, this will save the administrator having to retype the same text into different fields.

The above will be my main focus for the application as I feel it will take the most time, and will take priority in terms of testing.

The above is in italics as it was written at the time of the technical report upload in November. Since then and with different ideas, I decided not to use any drag and drop feature. One of the main reasons for this is because, for the sake of security I did not

want any files to be uploaded directly onto the database itself. I implemented a text editor that would allow users to copy and paste any materials they had and it would keep the same format that they are used to viewing it in. This text editor also prevented any cross-site scripting or injection into the application, meaning it was safer.
Also, aesthetically, it is a nicer experience for the user than using an upload function and it assisted with the overall flow of the course creation process.

2 System

2.1 System Architecture

The Developmint application is hosted on a LAMP server. All information is stored on a MySQL database. This database is maintained by PhpMyadmin, which allows me to create tables within the database without having to use SQL statements.
Developmint's databse is central in the architecture of the application. All user, administrator and sign-in information is stored in in it. This also remains true for all functions such as creating users, courses and tests. All user input and attempts to these tests are also stored through queries to it. This allows for the results to be called from the databases and be presented to the user. This architecture was decided up as it is the best fit for the requirements of the application. As mentioned in the below 'System Evoltion' section, this architecture could be subject to change slightly as the application grows.

2.4 User requirements Definition

The main design goal of Developmint is to give the ability to learn new content and upskill in your job, as well as course creation, in a clear and easy to use. To keep things as simple as possible, there will only be two types of users on the application:

- **Administrators / Trainers:** The administrators will oversee the functionality of the application from account creation to maintenance. They will have the following privileges:
    - Account creation – Administrators / Trainers will be responsible for creating accounts of all future users
    - Design course - Administrators and Trainers will have the ability to design courses and exams, and then assign them to the appropriate user. They will also have the ability to update those courses and, if needed, remove them
    - View course – Administrators / Trainers will also have the ability of viewing the course from the perspective of a user, this is for testing reasons
    - Attempt Quiz – Also for testing reasons, administrators / trainers will have the ability to attempt quizzes

    - View Results – Administrators / Trainers will need to view results of the user to feedback reports to the relevant management

- **Trainee (User):** The trainee will have had their account created by the administrator / trainer. The will now have access to the relevant course material that has been assigned to them. They will have the following privileges:
    - View course content
    - Attempt course exam / test / quiz after course work

2.5 Functional Requirements

The point of this section is to give a breakdown of each function that can be actioned by the different users. The main functions of the application come to a total of eleven. I will be giving the breakdown from the perspective of the Administrator / Trainee as he or she will have the same privileges of the user as well as their own.

2.7 Non-Functional Requirements

Performance / Response time requirement

Like any other application, how Developmint will perform will be a major concern. This is because it will be used by businesses in potential high-pressure environments so it will need to function without issue. Users will need to know how they are progressing through whatever material is being covered in real time, so a progression bar will be included.

The course creation should uploaded and be saved to the database without issue, some sort of loading bar will be implemented so the designer will know their course is being created.

Upon setting up the EC2 instance server, I deployed the server in the Irish region, as Developmint will be originally targeting Irish businesses, response times and latency need to be addressed early.

Availability Requirement

The application will only need to be available during the company's business hours. If needed, the site may need to go down if updates are required. This is ideal as updates can then be planned around those office hours so enough notice can be given for downtime.

Operating System Requirement

As this is a webn application, a specific OS is not required, just a web browser. However, all steps will be taken to ensure that the application is cross browser compatible.

Reliability Requirement

Reliability of the application will be constantly tested during the development process. This will be done with regular testing by myself and potential users of the application. As mentioned in the product proposal, different methods of testing will be carried out such as Unit Testing and UI Testing.

Maintainability Requirement

I want to allow other developers the ability to easily understand my code. To do that, I provide comments and documentation to allow ease of use. As I will be implementing different features of the website, I will do my best to keep them separated and labelled so developers will know where to look if they want to update anything/

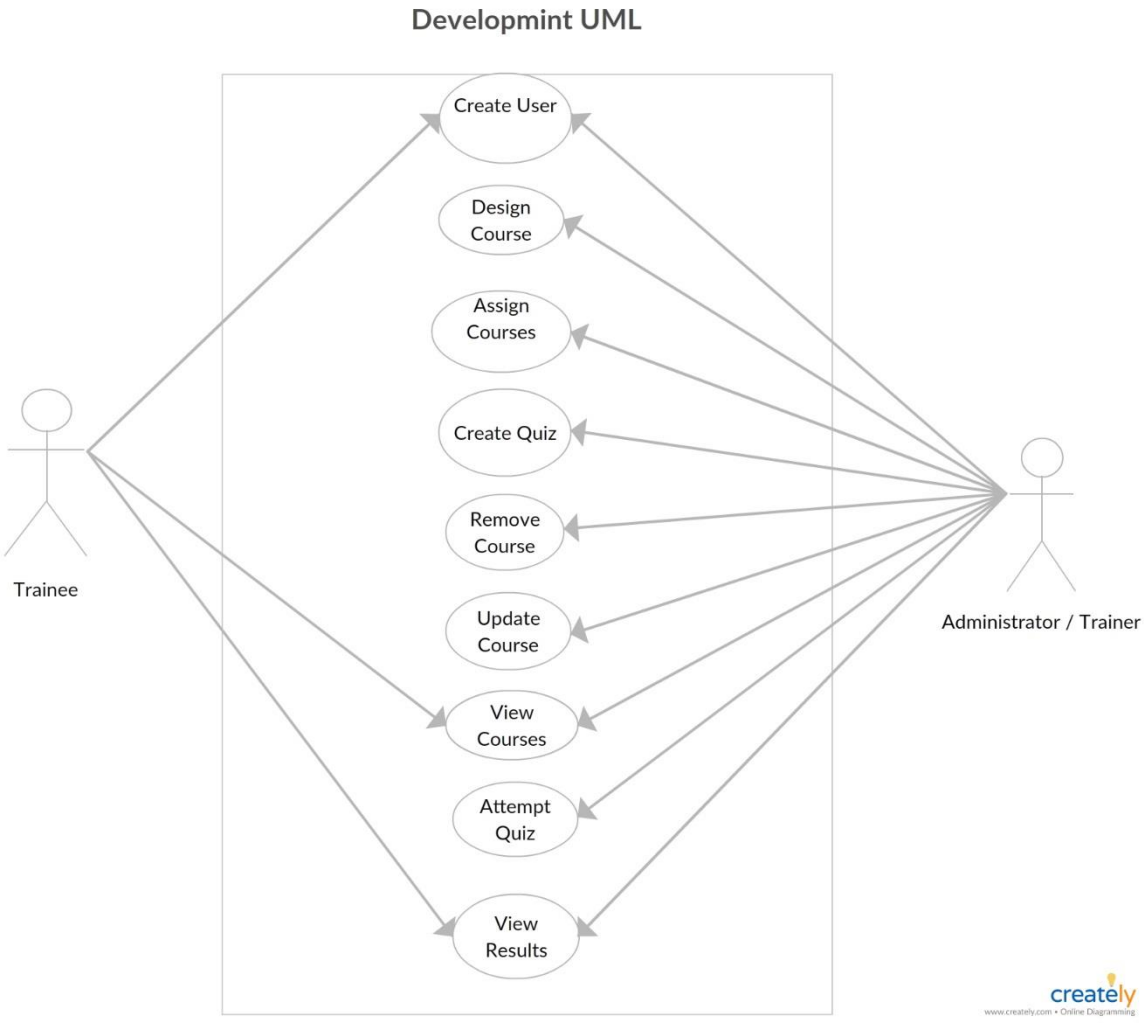
Reusability Requirement

As mentioned above, code structure will be well laid out so it is easy to understand. I will provide a user manual to provide a better understanding of how the application works. Because the application is also an e-learning platform, anyone can use the basis of the application to host whatever educational topics they want on the site.

Security

Certain security configurations have been implemented onto the server and the application itself. These will be covered off in a more in-depth description in the security testing section of this document.With Developmint being a web application, certain web security models will be implemented on the server side and client side.
Same origin policy will be implemented to stop outside origins accessing files or documents on Developmint's domain. This will also prevent cookies being read and written from a different origin than Developmint. Sessions will be logged out after a certain amount of inactivity time. An SSL certificate from Open SSL was implemented and installed through the Ubuntu console.
I have set up certain security groups on my EC2 instance on AWS. This allows on traffic on ports 22, 80 and 443. This is to allow for SSH, HTTP and HTTPS.

2.6 Use Case Model



Use Case 1: Create User

Scope
The scope of this use case is to highlight to an administrator how a user can be created.

Description
This describes how a user is created by administrator so they can use Developmint

Flow Description

Precondition
The 'Create User' page and form is loaded waiting to be filled by an administrator.

Activation
This use case starts when administrator fills out the form and clicks 'Create User'

Main Flow

1. The system allows the administrator to enter the details of the new user being created
2. The administrator fills in the required fields (Name, Email Address, Company Position, Password etc)
3. The administrator clicks the 'Create User' button.
4. The system saves all relevant details to the database and redirects to a page asking if they would like to edit or create another user.
5. The newly created user can now access Developmint.

Alternate Flow

A1: The user is already registered

1. The administrator enters the details of the user
2. The administrator clicks 'Create User'
3. The application displays an error to say the user is already created
4. The application redirects to part one of the Main Flow

Exceptional Flow

E1: The administrator enters incorrect or invalid data into the form

1. The administrator enters invalid data into the form
2. The application displays an error describing as such
3. The administrator re-enters the correct data
4. The use case reverts to point three of the Main Flow

Termination

This flow terminates when the administrator selects 'Create User'
Post Condition

The system enters a wait state for the next interaction

Use Case 2: Design Course

Scope

The scope of this use case is to highlight to an administrator how to design a course

Description

This user case describes how a course is created within the application
Flow Description

Precondition

The administrator is signed in and the application is in a wait state on the home page.

Activation

The use case is started by clicking on the 'Design Course' button.

Main Flow

1. The system recognises the credentials of the administrator and allows him / her access to the course design interface.
2. The user is presented with a UI where they can lay out what their course looks like
3. The administrator chooses what content they would like in their course
4. The administrator sets access rights on the course allowing only users who have been assigned this course to see it
5. The administrator gives the course a title.
6. The administrator clicks 'Create Course'

7. The course data is saved to the database.

Exceptional Flow

E1: The course the administrator is trying to create has the same name as previously created course.

1. The administrator gives the course a title
2. The application presents an error saying the same title exists
3. The use case reverts to point 5 of the main flow.

E2: The administrator tries creating a course with no content

1. The administrator enters the create course section
2. The administrator does not choose any content
3. The administrator chooses a course title
4. The administrator clicks 'Create Course'
5. The application shows an error saying there is no content to save
6. The use case reverts to point 2 of the main flow

Termination

The use case is terminated when the administrator saves the course

Post Condition

The application enters a wait state for the next interaction

Use Case 3: Assign Course

Scope

The scope of this use case is to highlight to an administrator how a course is assigned to a user

Description

This user case describes how a course is assigned to a user

Flow Description

Precondition

The application is awaiting interaction on the user details section

Activation

The use case is started by clicking on the 'Assign Course' button.

Main Flow

1. The administrator is on the user details section
2. The administrator clicks on the ‘Assign Course’ button
3. The administrator then chooses which course he or she wants to assign to that user
4. The administrator then clicks on the ‘Assign Course to User’ button.
5. The application will then show a confirmation that the course has been assigned.
6. The application updates the details of that user’s new assignment and saves it to the database

Alternate Flow

A1: The administrator assigns a course that has already been assigned.

1. The administrator is on the user details section
2. The administrator clicks on the ‘Assign Course’ button
3. The administrator then chooses which course he or she wants to assign to that user
4. The administrator then clicks on the ‘Assign Course to User’ button.
5. The application shows an error saying that this user has already been assigned this course
6. The use case reverts to point 3 of the main flow.

Exceptional Flow

E1: The user does not have the right privileges to have a course assigned.

1. The administrator is on the user details section
2. The administrator clicks on the ‘Assign Course’ button
3. The administrator then chooses which course he or she wants to assign to that user
4. The administrator then clicks on the ‘Assign Course to User’ button.
5. The application shows an error saying that this user does not have the correct credentials to have this course assigned to them
6. The use case reverts to point 3 of the main flow

Termination

The use case is terminated when the administrator clicks on ‘Assign Course to User’ button

Post Condition

The application enters a wait state for the next interaction

Use Case 4: Create Quiz

Scope

The scope of this use case is to highlight to an administrator how a quiz is created

Description

This user case describes how a course is created by the administrator
Flow Description

Precondition

The application is awaiting interaction on the courses section

Activation

The use case is started by clicking on the 'Create Quiz' button.

Main Flow

1. The administrator clicks the 'Create Quiz' button
2. The administrator is presented with a form where they can enter in the questions they want to store in the quiz.
3. The administrator assigns a score weighting to each quiz
4. The administrator gives the quiz a title
5. The administrator then clicks the 'Save Quiz' button.
6. The quiz data is then saved to the database.

Exceptional Flow

E1: The course the administrator is trying to create has the same name as previously created quiz.

1. The administrator gives the quiz a title
2. The application presents an error saying the same title exists
3. The use case reverts to point 4 of the main flow.

E2: The administrator tries creating a quiz with no content

1. The administrator enters the create quiz section

2. The administrator does not choose any questions
3. The administrator chooses a quiz title
4. The application shows an error saying there is no content in the quiz
5. The use case reverts to point 2 of the main flow.

E3: The course the administrator does not give the quiz questions any weighted score

1. The administrator enters the create quiz section
2. The administrator enters the questions
3. The administrator does not give the quiz any weighting
4. The administrator chooses a quiz title
5. The application shows an error saying there is no score weighting to the quiz
6. The use case reverts to point 2 of the main flow.

Termination

The use case is terminated when the administrator clicks on 'Save Quiz' button.

Post Condition

The application enters a wait state for the next interaction

Use Case 5: Update Course

Scope

The scope of this use case is to highlight to an administrator how they can update the created courses

Description

This user case describes how a course is updated and modified by an administrator

Flow Description

Precondition

The application is awaiting interaction on the courses section

Activation

The use case is started by clicking on the 'Edit Course' button beside the chosen course

Main Flow

1. The administrator enters the course editing section by clicking 'Edit Course'
2. The administrator can then update, add or remove content on the course.
3. The administrator clicks on the 'Update Course' button.
4. The updates to the course are saved to the database.

Alternate Flow

A1: The additions or updates to the course have already been done.

1. The administrator discovers the content as already been updated.
2. The administrator presses 'Back' on the browser.
3. The administrator returns to the Courses section.

Exceptional Flow

E1: The administrator tries to add the same documentation to the course.

1. The administrator adds the documentation to the course update section.
2. The administrator clicks on the 'Update Course' button.
3. The application displays an error saying that the content already exists.
4. The use case reverts to point 1 of the main flow.

Termination

The use case is terminated when the administrator clicks on 'Update Course' button.

Post Condition

The application enters a wait state for the next interaction

Use Case 6: Remove Course

Scope

The scope of this use case is to highlight to an administrator how they can remove courses they have created.

Description

This user case describes how a course is deleted from the application by an administrator

Flow Description

Precondition

The application is awaiting interaction on the courses section

Activation

The use case is started by clicking on the ‘Edit Course’ button beside the chosen course

Main Flow

1. The administrator enters the course editing section by clicking ‘Edit Course’
2. The administrator then clicks the ‘Remove Course’ button.
3. The application asks you if the administrator is sure they want to delete the course.
4. The course is removed from the application and the information is removed from the database.

Termination

The use case is terminated when the course has been removed from the application.

Post Condition

The application enters a wait state for the next interaction

Use Case 7: View Course

Scope

The scope of this use case is to highlight to a user how they can access their course.

Description

This user case describes how a course is viewed and accessed by a user.

Flow Description

Precondition

The user is logged in and on the home page.

Activation

The use case is started by the user clicking on the ‘My Courses’ button.

Main Flow

1. The user clicks on the ‘My Courses’ button
2. The user chooses which course they want to view from the list that is assigned to them.
3. The course assigned to them opens and the user begins to look through the material.
4. Once the user has finished reading through the material, the user can return back to the ‘My Courses’ section.

Alternate Flow

A1: The user chooses the wrong course

1. The user clicks on the wrong course by accident.
2. The user returns to the ‘My Courses’ section
3. The user then chooses the course they meant to.

Exceptional Flow

Termination

The use case is terminated when the user clicks on the ‘Return to my courses’ button.

Post Condition

The application enters a wait state for the next interaction.

Use Case 8: Attempt Quiz

Scope

The scope of this use case is to highlight to a user how they can access tests and quizzes based on the courses assigned to them

Description

This user case describes how a test and quiz is attempted by a user
Flow Description

Precondition

The user is logged in and on the home page.

Activation

The use case is started by the user clicking on the ‘Attempt Quiz’ button in the ‘My Courses’ section.

Main Flow

1. The user clicks the ‘My Courses’ button.
2. In the ‘My Courses’ section, the user clicks the ‘Attempt Quiz’ button.
3. The user attempts each question.
4. The user clicks the ‘Submit Answers’ button.
5. The quiz data is saved to the database.
6. The application returns to the ‘My Courses’ section.

Alternate Flow

A1: The user doesn’t answer one or multiple questions.

1. The user leaves an answer field or fields blank.
2. The user indicates they want to finish the quiz by clicking the ‘Submit Answers’ button
3. The application displays a message indicating that there hasn’t been some questions answered and checks with the user if they want to submit their answers
4. The user can choose to either attempt the questions that were not answered or submit regardless

5. The use case returns to point 4 of the main flow.

Exceptional Flow

Termination

The use case is terminated when the user clicks on the ‘Submit Answers’ button.

Post Condition

The application enters a wait state for the next interaction

Use Case 9: View Results

Scope

The scope of this use case is to highlight to a user how they can view the progress they've made with their development.

Description

This user case describes how a user can check their progress.

Flow Description

Precondition

The user is logged in and on the home page.

Activation

The use case is started by the user clicking on the 'View Results' button in the 'User Details' section.

Main Flow

1. The user goes to the 'User Details' section of the application.
2. The user clicks on the 'View Results' button.
3. The application loads all results from quizzes attempted from the database.
4. The user can view the results of all quizzes they have attempted.

Alternate Flow

A1: The user attempts to view results when they haven't attempted any quizzes.

1. The user clicks on 'View Results' button.
2. The application attempts to pull results from the database but there is no data.
3. The application displays a message to indicate there aren't any results to show
4. The use case reverts to point 1 at the Main Flow.

Exceptional Flow

Termination

The use case is terminated when the user returns to the ‘User Details’ section,

Post Condition

The application enters a wait state for the next interaction

2.8 Data Requirements

This section will provide information on the database of the application. This is an important part of the document as it will describe how user data will be stored.
I will be using a MySQL database to store all user information. There will be three main tables within the Developmint database; Users, Courses and Tests.

2.9 Implementation

AWS

Developminttraining.com is hosted on an Amazon Web Services EC2 instances. This is an Ubuntu v 16.04.4 server. I downloaded a .pem authentication key file. This allows me to SSH onto the server from the application PUTTY. This meant I had a Linux command line to allow me to make configurations to my server. I also downloaded and installed an SFTP service, FIlezilla, to update the site with new files.

Once this had been set up, I added my domain name mentioned above to the AWS Route 53 service as an A(IPv4) record. This allows users to access the website through the domain.

Now that I had access to the domain, I could use AWS’ version of Cloud9, which I could create an SSH session to, AWS provided me with another rsa key to allow this session, it also required versions of PHP and Node.js. I also had to install a LAMP stack, and set up Apache. I also no longer needed Filezilla because of, as I could now update the site in real-time.

TinyMCE Text Editor

In the original product proposal, I stated that I was going to provide a drag and drop style framework for users who were designing courses. That was based on the idea of having a user upload their training documents, and then having them presented to them to be dragged and dropped into an area, along with a test that was created.

I decided against that idea. The main reason being that, in terms of security, it would be very difficult to monitor what gets uploaded to the database.

I did some research and discovered TinyMCE, which is an online WYSIWYG HTML editor. WYSIWYG is an acronym for *what you see is what you get*. In terms of the Developmint application, it means that whatever text format you enter into the website, it gives an output into the form in the same format.

If you copy and paste a format from another text document, for example a .docx file, it will prompt you asking if you would like to keep the same format.

I felt would this be the best option, as a lot of trainers and companies would already have training materials created and stored locally. This means that all they would have to do is copy and paste their documents into the editor and upload them to the database.

TinyMCE stores the inputs in a HTML format, allowing the ouput of the information to appear as you entered it.

```
<html>
<head>
</head>
<body>
<p style="margin: 0cm 0cm 0.0001pt; text-align: center; line-height: 115%; font-
size: 11pt; font-family: Calibri, sans-serif;" align="center"><strong><span
style="font-size: 20.0pt; line-height: 115%; font-family: Arial, sans-
serif;">Showcase Poster</span></strong></p>
<p style="margin: 0cm 0cm 0.0001pt; line-height: 115%; font-size: 11pt; font-
family: Calibri, sans-serif;"><span style="font-size: 12.0pt; line-height: 115%;
font-family: Arial, sans-serif;"> </span></p>
<p style="margin: 0cm 0cm 0.0001pt; line-height: 115%; font-size: 11pt; font-
family: Calibri, sans-serif;"><strong><span style="font-size: 12.0pt; line-
height: 115%; font-family: Arial, sans-serif;"> </span></strong></p>
<table class="MsoTableGrid" style="width: 443.5pt; margin-left: 26.7pt; border-
collapse: collapse; border: none;" border="1" width="591" cellspacing="0"
cellpadding="0">
<tbody>
<tr style="height: 25.65pt;">
<td style="width: 319.0pt; border: solid windowtext 1.0pt; padding: 0cm 5.4pt
0cm 5.4pt; height: 25.65pt;" width="425">
<p style="margin: 0cm 0cm 0.0001pt; line-height: normal; font-size: 11pt; font-
family: Calibri, sans-serif;"><span style="font-size: 12.0pt; font-family:
Arial, sans-serif;">Size</span></p>
</td>
<td style="width: 124.5pt; border: solid windowtext 1.0pt; border-left: none;
padding: 0cm 5.4pt 0cm 5.4pt; height: 25.65pt;" width="166">
<p style="margin: 0cm 0cm 0.0001pt; text-align: center; line-height: normal;
font-size: 11pt; font-family: Calibri, sans-serif;" align="center"><span
style="font-size: 12.0pt; font-family: Arial, sans-serif;">A1</span></p>
```

The above is a sample insert of course-content into the database, notice the HTML script formatting the information.

Like Bootstrap, or Javascript, TinyMCE provides you with a link to place in your header of your application. As it is a paid service, you can sign up for the free version, and the company provides you with an API key to include in the link.

TinyMCE is initialised in the section of the site you require, it asks you to declare a text area, which will transform into the text editor.

```
<script>
    tinymce.init({
        selector: '#mytextarea',
        height: 500,
        theme: 'modern',
        plugins: 'print preview fullpage powerpaste searchreplace autolink directionality advcode
        toolbar1: 'formatselect | bold italic strikethrough forecolor backcolor | link | alignleft
    });
</script>
<form id="upload-materials" action='db/dbuploadmaterials.php' method = "POST">
    <div class='form-group'>
    <input type='text' name='title' class='form-control' placeholder='Material Title'>
    </div>
    <textarea id="mytextarea" name='content'></textarea>
```

phpMyAdmin

As stated above, I decided to use phpMyAdmin to provide me with the ability to manage my database. phpMyAdmin comes as part of the LAMP service that was installed in my server. It allows me to create, update, alter and remove tables from the database without having to insert SQL code.

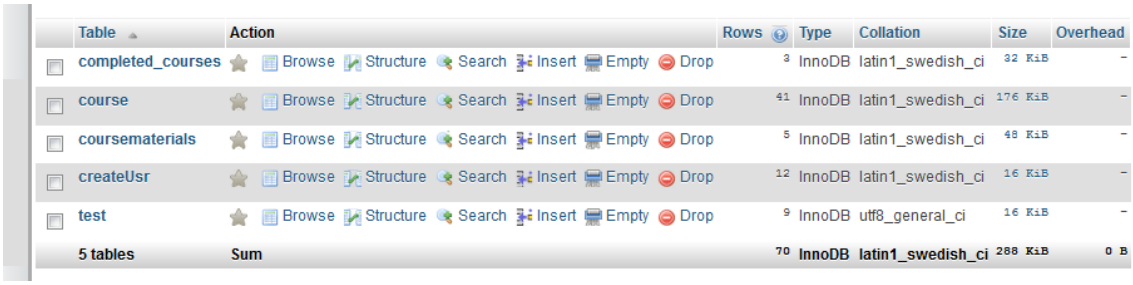
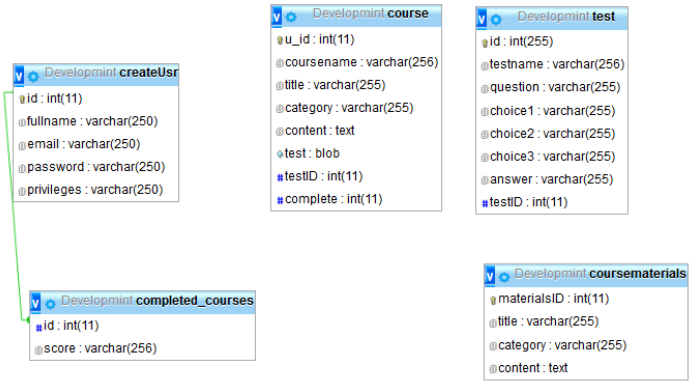


Table	Action	Rows	Type	Collation	Size	Overhead
completed_courses	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	32 KiB	-
course	Browse Structure Search Insert Empty Drop	41	InnoDB	latin1_swedish_ci	176 KiB	-
coursematerials	Browse Structure Search Insert Empty Drop	5	InnoDB	latin1_swedish_ci	48 KiB	-
createUsr	Browse Structure Search Insert Empty Drop	12	InnoDB	latin1_swedish_ci	16 KiB	-
test	Browse Structure Search Insert Empty Drop	9	InnoDB	utf8_general_ci	16 KiB	-
5 tables	Sum	70	InnoDB	latin1_swedish_ci	288 KiB	0 B

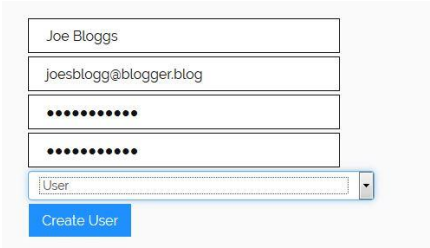


Features

This section will cover all features of the site. Including their name, how the steps of each site works with technical information and Unit testing to show how I made sure everything worked.

Features – Create User

The Create User form allows an administrator to enter new user details into the form, this giving this new user access to the site. It is accessed on the 'User Management' dropdown under the link 'Create User' The form takes the user's full name, email address, and allows the administrator to give them either admin or user privileges.



Feature steps:

- The administrator is presented with details of a new user to create.
- The administrator enters the details of the user.
- The administrator selects what privileges to give the user.
- The administrator clicks 'Create User'
- The URL bar displays "createUsr.php?submit=create success" if the process has been completed properly.

1. The administrator presses the "Create User" button

This triggers a "POST" method in the form and calls on the action file "userCreate.php"

2. userCreate.php checks all fields to ensure none are blank:

If any field are blank, the URL returns an error to alert the administrator.

```
if(empty($fullname) || empty($email) || empty($password) || empty($confirmpassword) || empty($privileges)){
header("Location: ../createUsr.php?submit=empty");
 exit();
```

3. userCreate.php checks the Full Name field:

The script checks the field to make sure only specified characters are entered in the field with the statement in the image below. If any of the characters are outside this limit, the form will not submit the information, giving an error in the URL bar '?=invald'.

```
if(!preg_match("/^[a-z A-Z]*$/", $fullname)){
    header("Location: ../createUsr.php?submit=invalid");
    exit();
```

4. userCreate.php checks the email field:

The script checks that the email that has been submitted is a valid email with a validation filter, if the email is not correct, the URL gives an error to say '?submit=invalid email'

```
if(!filter_var($email, FILTER_VALIDATE_EMAIL)){
  header("Location: ../createUsr.php?submit=invalid email");
```

The script then checks the database to make sure that there isn't an existing email address within the user's table. If the email has been used before, the form will not submit the information, giving an error '?submit=email taken'.

5. userCreate.php checks the password field:

The script takes the password and hashes it, sending the hashed password to the database.

```
//Hashing passwords
$hashedPwd = password_hash($password, PASSWORD_DEFAULT);
```

6. Once all the above checks are complete:

The script accepts all of the information submitted, and inserts them into the 'createUsr' table in the database.

Unit Testing

Unit testing was implemented on the create user forms to ensure the 'userCreate.php' file worked correctly with the all input values. The steps taken were:

a. Click 'Create User' with all fields empty

/createUsr.php?submit=empty

b. Click 'Create User' with no email address or one currently in use. Note the red border around the field, indicating the incorrect email format.

/createUsr.php?submit=email taken

Features - Login

The login function gives a user the ability to access the main page of the application. To be able to log in, the user must have their correct credentials. For Developmint, this is their email address and password. Once the user has provided the correct credentials, they will get access to their main page.

Feature steps

1. **The user is on the landing page and clicks 'Login'.**
   The user is presented with a login form, prompting them to input their username and password.

2. **The user inserts their credentials and clicks the login button:**
   The login button triggers a 'POST' method and loads an action on /db/dblogin.php file.

3. **The 'dblogin.php' file runs a script to check the credentials:**
   The script checks if an email and or password has been placed into the input fields, if no email or password has been entered, it produces an error message in the URL to advise the user.

```php
if (empty($email) || empty($password)) {
    header('Location: ../login.php?login=empty');
    exit();
```

4. **The script checks to see if an email exists on the database:**
   If an email does not exist on the database, and error is produced to advise the user.

```
}else{
    $sql = "SELECT * FROM createUsr WHERE email='$email'";
    $result = mysqli_query($connection, $sql);
    $resultCheck = mysqli_num_rows($result);
    if ($resultCheck < 1) {
        //error message for empty DB
        header('Location: ../login.php?=error');
        exit();
```

5. **The script then checks the password verification:**
   Firstly, the script should check if the password that is provided matches the hashed password on the database. If not, the script returns an error to say an error has occurred.

```
if ($row = mysqli_fetch_assoc($result)) {
    //Retrieving secure password
        $hashedPwdCheck = password_verify($password, $row['password']);
    if ($hashedPwdCheck == false) {
        header('Location: ../index.php?login=error');
        exit();
```

   If the provided password and the hashed password match, the script redirects the user to their profile page and starts a session based on the user's data on the user table.

```
}elseif($hashedPwdCheck == true){
    //Logs in user
    $_SESSION['id'] = $row['id'];
    $_SESSION['fullname'] = $row['fullname'];
    $_SESSION['email'] = $row['email'];
    $_SESSION['privileges'] = $row['privileges'];
    header('Location: ../profile.php');
    exit();
}
```

Unit Testing

Unit testing was implemented on the login form to ensure the 'dblogin.php' file worked correctly with the all input values. The steps taken were:

a. Click 'Login' with all fields empty

   /login.php?login=empty

b. Click 'Login' with an unregistered email

   /login.php?=error

c. Click 'Login' with wrong password

   /index.php?login=error

d. Click 'Login' with correct credentials.

   'profile.php

Upload Materials

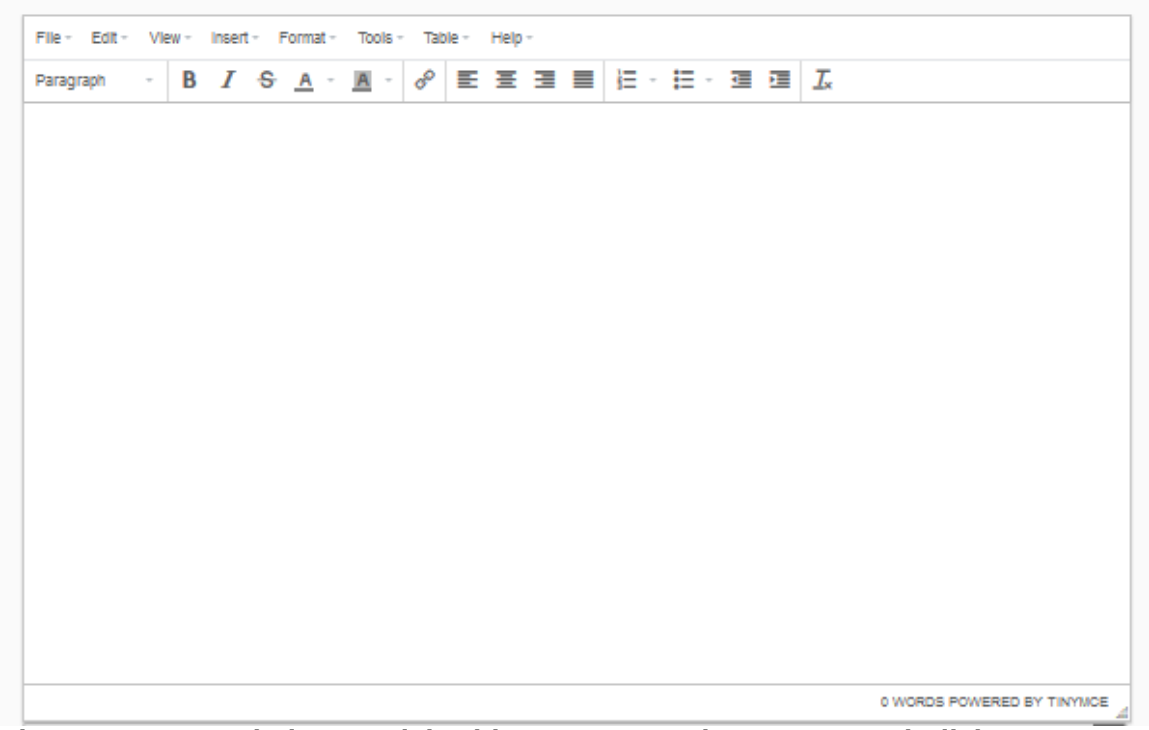
This section is accessed by clicking on the dropdown ‘Courses’ and then ‘Upload Materials’. It provides a trainer the ability to insert, edit and format their training materials to create the foundation of the course. This section asks the trainer to name their course, insert the materials and then give a category.

Feature steps

1. **The user is presented with a page to upload materials:**
   This includes an input field to name the materials, the TinyMCE text editor, where the user can either format their course materials, or paste a format in from a different text document such as .docx. They are also asked to select a category.
2. **The page that is loaded includes the script to initiate the text editor:**

```
<script>
    tinymce.init({
        selector: '#mytextarea',
        height: 500,
        theme: 'modern',
        plugins: 'print preview fullpage powerpaste searchreplace autolink directionality advcode v
        toolbar1: 'formatselect | bold italic strikethrough forecolor backcolor | link | alignleft
    });
</script>
<form id="upload-materials" action='db/dbuploadmaterials.php' method = "POST">
    <div class='form-group'>
    <input type='text' name='title' class='form-control' placeholder='Material Title'>
    </div>
    <textarea id="mytextarea" name='content'></textarea>
```

   The plugin and toolbar strings are lengthy, but provide a wide range of text editing functionality.



3. **The user enters their materials title, content and category and clicks 'Upload Materials':**
   This tiggers a 'POST' method and loads an action file called 'dbuploadmaterials.php'. This file runs a script to check the inputs of the user.

4. **The script checks all inputs to make sure they contain text:**
   The script checks the title, text editing fields and the dropdown bar to make sure they all contain an input. If they don't, the script produces an error to say one of them is empty.

5. **The script also gives an error if any fields are empty:**

```
else{
    header('Location: ../uploadmaterials.php?=Nothing selected');
    exit();
}
```

6. **Once all entries have been made, the script inserts the materials title, content and category into the materials table:**
   It also alerts the user that the upload has been successful in the URL

```
$sql = "INSERT INTO coursematerials (title, category, content) VALUES ('$title', '$category', '$content');";
mysqli_query($connection, $sql);
header("Location: ../uploadmaterials.php?submit=success");
```

a. User clicks ‘Upload Materials’ with either the Title

'uploadmaterials.php?submit=emptytitle

Upload Test

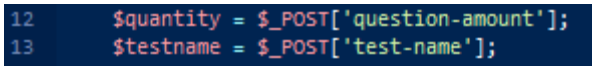
This section of the application is divided into two pages. The first is accessed by clicking the 'Create Test' button in the sidebar dropdown. This navigates you to a text input field and a dropdown list. Here the user names the test, and chooses the number of questions will be in the test.

Feature steps

1. **On clicking 'Create Test', the user is presented with the 'nametest.php. page:**
   The user inputs the name of the test into the text field.

2. **The user selects how many questions the test will have:**
   The user selects between one and ten questions and clicks 'Name Test'. This triggers a 'POST' method and loads the 'uploadtest.php' page.

3. **The 'uploadtest.php' page loads:**
   The file in this page takes certain parameters from the previous 'nametest.php' and sets them as variables. These set up what variables can be posted to the database. It takes:
   - The id of the test name input field "id='test-name'" and sets it as the $testname variable "$testname=$_POST['test-name'];".
   - The id of the select field input "id='question-amount'" and sets it as the $quantity variable "$quantity = $_POST['question-amount'];".
   - These are both stored in the <?php ?> section at the head of the page



4. **The ‘uploadtest.php’ file’s script creates a loop:**

   The script runs a for loop to check how many question amounts there are, and prints them to the page. It takes the $quantity variable for the loop’s iteration amount and uses the variable “$numberofloops = 0;” to count out each question amount. The iteration $i is echoed against the ‘test-question’ and ‘anschoice’ id’s of the select options for that input field, as well as the actual answer id ‘actanswer’. It then increments every loop at the end of the input field with “$numberofloops++;”.

```
    <?php
    $numberOfLoops = 0;
    for ($i = 1; $i <= $quantity; $i++){
    ?>
<label for='test-question'>Question <?php echo $i; ?></label>
<input type="text" class='form-control' id='test-question1' name="<?php echo $i; ?>test-question"><br>
<label for='anschoice1'>Answer Choice 1</label>
<input type="text" class='form-control' id='anschoice1' name="<?php echo $i; ?>anschoice1"><br>
<label for='anschoice2'>Answer Choice 2</label>
<input type="text" class='form-control' id='anschoice2' name="<?php echo $i; ?>anschoice2"><br>
<label for='anschoice3'>Answer Choice 3</label>
<input type="text" class='form-control' id='anschoice3' name="<?php echo $i; ?>anschoice3"><br>
<label for='actanswer'>Actual Answer</label>
<input type="text" class='form-control' id='actanswer' name="<?php echo $i; ?>actanswer">
<div class="line"></div>
<?php
  $numberOfLoops++;
}
echo "<input type='hidden' name='numberOfLoops' value='".$numberOfLoops."'>";
?>
<button type="submit" class="btn btn-primary btn-lg" name='upload-test'>Upload Test</button>
</form>
```

   This “$numberofloops;” variable is stored in a hidden input field, so it can be posted to the upload test function page.

5. **The user enters in their test questions and possible answers and clicks ‘Upload Test’:**

   This triggers a ‘POST’ method and loads the action file ‘dbuploadtest.php’. This file takes the posted form and prepares to enter the information into the test table on the database.

   It takes the all the test parameters (question, answer choices one to three, and the actual answer), as well as the $numberofloops variable, all of which were posted in the form.

   The script then loops through the amount of questions and answers and stores them so it knows the quantity to send to the database.

   As previously mentioned, the number of loops is posted and declared as variable $qty. This $qty variable is then placed in the for loop as the iteration which is taking all of the input question fields and counts them as they are posted.

   The variable $num is declared as a random number which is generated between 1 and 1,000. This $num will be generated and entered into the database as the test’s ID.

```php
if (isset($_POST['upload-test'])) {

    require 'init.php';

    $qty = $_POST['numberOfLoops'];//mysqli_real_escape_string($connection, $_POST['numberOfLoops']);

    //echo "***".$qty;
//}

    $num = rand (1, 1000);
    $tn = mysqli_real_escape_string($connection, $_POST['test-name']);
   // exit();
    for ($i = 1; $i <= $qty; $i++) {
        $tq =  mysqli_real_escape_string($connection, $_POST[$i.'test-question']);
        $o1 = mysqli_real_escape_string($connection, $_POST[$i.'anschoice1']);
        $o2 = mysqli_real_escape_string($connection, $_POST[$i.'anschoice2']);
        $o3 = mysqli_real_escape_string($connection, $_POST[$i.'anschoice3']);
        $a = mysqli_real_escape_string($connection, $_POST[$i.'actanswer']);
```

6. **The script then checks all fields are not empty:**
   If any of the fields are empty, the script returns an error in the URL to advise the user.

```php
//Checks for empty question and answer fields
if (empty($tq) || empty($o1) || empty($o2) || empty($o3) || empty($a)) {
    header('Location: ../nametest.php?fields=empty');
    exit();
```

7. **The script then checks if the questions contain any unwanted characters:**
   If any characters that are outside the parameters of the form are submitted, the script returns an error in the URL.

```php
//Checks input validation on question and answers
if (!preg_match("/^[a-z A-Z0-9]*$/", $tq) && !preg_match("/^[a-z A-Z0-9]*$/", $o1) && !preg_match("/^[a-z A-Z0-9]*$/", $o2)
&& !preg_match("/^[a-z A-Z0-9]*$/", $o3) && !preg_match("/^[a-z A-Z0-9]*$/", $a)) {
    header('Location: ../nametest.php?qas=invalid');
    exit();
```

8. **Once all the checks have been completed, the test name, questions, answer options and the test ID get inserted into the test table:**

```php
else {
  $sql = "INSERT INTO test (testname, question, choice1, choice2, choice3, answer, testID) VALUES ('$tn', '$tq', '$o1', '$o2', '$o3', '$a', '$num');";
  if (mysqli_query($connection, $sql)) {
  }
```

9. **The user is returned to the name test page with a success notification in the URL:**

```php
} header('Location: ../nametest.php?=success');
exit();
```

Unit Testing

a. User clicks ‘Upload Test’ leaving a question empty

/nametest.php?fields=empty

b. User clicks ‘Upload Test’ leaving an answer choice empty

/nametest.php?fields=empty

Create Course

This section of the page allows a user to combine the test they created with the course materials they uploaded. It is access by clicking the ‘Create Course’ link on the ‘Courses’ dropdown on the side navbar. This loads the ‘createcourse.php’ page.

Feature Steps

1. The two dropdowns:

The two dropdowns appear, one containing the course materials and one containing the test. They are both identified by their unique identifier and name. At the top of the ‘createcourse.php’ file, is a php script that says “include ‘db/dbcreatecourse.php’;”. This file is includes because it contains two functions:

i. One to display all the uploaded materials
ii. One to display all the tests

These two functions are sql queries, to pull all materials and tests from the database, and insert them into the two dropdowns.

```
function optionMaterials($connection){
    $sql = "SELECT * FROM coursematerials";

    $result = mysqli_query($connection, $sql);

    if (!$result) {
        printf("Error: %s\n", mysqli_error($connection));

    }

    while($row = mysqli_fetch_assoc($result)){
        echo "<option>".$row['title']."</option>";
    }
}
```

```
function optionTests($connection){
    $sql = "SELECT * FROM test";
    $result = mysqli_query($connection, $sql);

    if (!$result) {
        printf("Error: %s\n", mysqli_error($connection));

    }

    while($row = mysqli_fetch_assoc($result)){
        echo "<option value='".$row['testID']."'>".$row['testname']."</option>";
    }
}
```

2. **The two functions are echoed into the two dropdowns:**

   In the 'createcourse.php' file, the two functions are seen in the select dropdown forms.

```
<form action='db/dbuploadcourse.php' method='POST'>
    <div class='form-group'>
        <label for='course-name'>Course name.</label>
        <input type='text' id='course-name' name='course-name' class='form-control'>
    </div>
    <div class='form-group'>
        <label for='course-materials'>Choose the materials.</label>
        <select class='form-control' name='course-materials'>
            <option name='course-materials'><?php optionMaterials($connection);?></option>
        </select>
    </div>
    <div class='form-group'>
        <label for='course'>Choose the test.</label>
        <select class='form-control' name='course-tests'>
            <option name='course-tests'><?php optionTests($connection);?></option><br>
        </select>
    </div>
    <button type="submit" class="btn btn-primary btn-lg" name='create-course'>Create Course</button>
</form>
```

3. **The user selects the two options they want to merge:**

   The user then clicks 'Create Course', this triggers a 'POST' method and calls on the action file 'dbuploadcourse.php'.

4. **'dbuploadcourse.php':**

   This file takes the inputs posted from both dropdowns. It declares the variable $coursename as the posted dropdown name 'course-materials'. The script then performs a query on the coursematerials table, while declaring the variables $content, $title and $category.

```
$coursename = mysqli_real_escape_string($connection, $_POST['course-materials']);
$title = "";
$category = "";
$content = "";
$sql = "SELECT * FROM coursematerials
        WHERE title = '$coursename' ";
$result = mysqli_query($connection, $sql);
//  $materialsArray = array();
//  $matCount = 1;
while($row = mysqli_fetch_assoc($result)){
    $title = $row['title'];
    $category = $row['category'];
    $content = $row['content'];
    // $materialsArray['row'.$matCount]['title'] = $row['title'];
    // $materialsArray['row'.$matCount]['category'] = $row['category'];
    // $materialsArray['row'.$matCount]['content'] = $row['content'];
    // $matcount++;

}
```

The script also declares the variable $coursetests as the posted dropdown name ‘course-tests’. It also runs an sql query on the test table inside the database while declaring the variable $testID. This portion of the script has to take all of the possible questions and answer options, and place them inside an array, as they are stored in the database as a blob.

The test array is declared as “$questionArray = array();” These are placed inside the while loop that declares all of the options of the test. While also assigning a value to the $testID variable.

Due to the test array being placed inside a blob, it has to be serialised. So the variable “$tarray” was declared as “$tarray = serialize($questionarray);”

```
$coursetests = mysqli_real_escape_string($connection, $_POST['course-tests']);
$sql2 = "SELECT * FROM test
        WHERE testID = '$coursetests' ";
$result2 = mysqli_query($connection, $sql2);
$questuonArray = array();
// name
// q
// c1
// c2
// c3
// a
$testID = "";
$count = 1;
while($row2 = mysqli_fetch_assoc($result2)){
    $questuonArray['row'.$count]['id'] = $row2['id'];
    $questuonArray['row'.$count]['testname'] = $row2['testname'];
    $questuonArray['row'.$count]['question'] = $row2['question'];
    $questuonArray['row'.$count]['choice1'] = $row2['choice1'];
    $questuonArray['row'.$count]['choice2'] = $row2['choice2'];
    $questuonArray['row'.$count]['choice3'] = $row2['choice3'];
    $questuonArray['row'.$count]['answer'] = $row2['answer'];
    $testID = $row2['testID'];
    $count++;
// }
// // echo"<pre>";
// // print_r($questuonArray);



$tarray = serialize($questuonArray);
```

Finally, all the variables from the course materials, and the entire test are queried to the course table inside the database.

```
$query = "INSERT INTO course (coursename, title, category, content, test, testID) VALUES ('$coursename', '$title', '$category', '$content', '$tarray', '$testID')";

$result = mysqli_query($connection, $query);
```

The user is then redirected to the create course page with an updated URL saying the creation was a success.

```
header('Location: ../createcourse.php?=creationsuccess');
```

From the screenshots above, you will see snippets of code that are now comments. They were included to ensure that information was being printed, so I would know that said information would be sent to the new table.

```
Array
(
    [row1] => Array
        (
            [id] => 68
            [testname] => Steps of Sales
            [question] => What does the letter S stand for?
            [choice1] => See
            [choice2] => Smell
            [choice3] => Smile
            [answer] => Smile
        )

    [row2] => Array
        (
            [id] => 69
            [testname] => Steps of Sales
            [question] => What does the first E stand for?
            [choice1] => Eye contact
            [choice2] => Excitement
            [choice3] => Enema
            [answer] => Eye contact
        )

    [row3] => Array
        (
            [id] => 70
            [testname] => Steps of Sales
            [question] => What does the second E stand for?
            [choice1] => Entrance
            [choice2] => Enthusiasm
            [choice3] => Elephant
            [answer] => Enthusiasm
        )

)
```

Take Course

This section of the course allows the user to take a course. It is accessible in the 'Take a Course' link under the Courses dropdown on the navbar. By clicking the link, it loads the 'choosematerials.php' file.

Feature Steps

1. **Inlude 'db/dbpairusrsandcourses':**

   This included file contains the function that queries the courses table in the Developmint database. It takes everything from the course table and echoes the course u_id and course title row into a dropdown.

```php
function optionCourse($connection){
    $sql = "SELECT * FROM course";
    $result = mysqli_query($connection, $sql);

    if (!$result) {
        printf("Error: %s\n", mysqli_error($connection));

    }

    while($row = mysqli_fetch_assoc($result)){
        echo "<option value='".$row['u_id']."'>".$row['u_id'].". ".$row['coursename']."</option>";
    }
}
```

2. **'choosematerials.php':**

   The page itself asks the user to select a course they would like to take from the dropdown menu.

```php
<div class='content'>
    <h2>Choose the course you would like to view.</h2>
    <form action='viewmaterials.php' method="POST">
        <div class='form-group'>
            <label for='course-name'>Choose the course.</label>
                <select class='form-control' name='course-name'>
                    <option name='course-name'><?php optionCourse($connection); ?></option><br>
                </select>
        </div>
        <button type="submit" name='choose-course' id='choose-course' class="btn btn-primary btn-lg">Take Course</button>
    </form>
            <div class="line"></div>
</div>
```

3. **Selecting course:**

   The user selects the course they want to take, and clicks 'Take Course'

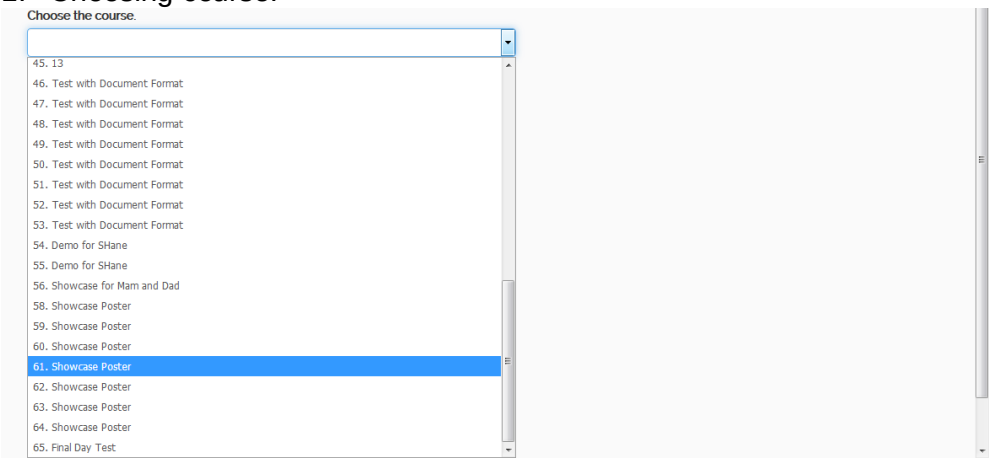
4. **Viewing materials:**

   The user is now presented with the course materials from that course. They will appear in the format they were uploaded to the database with.

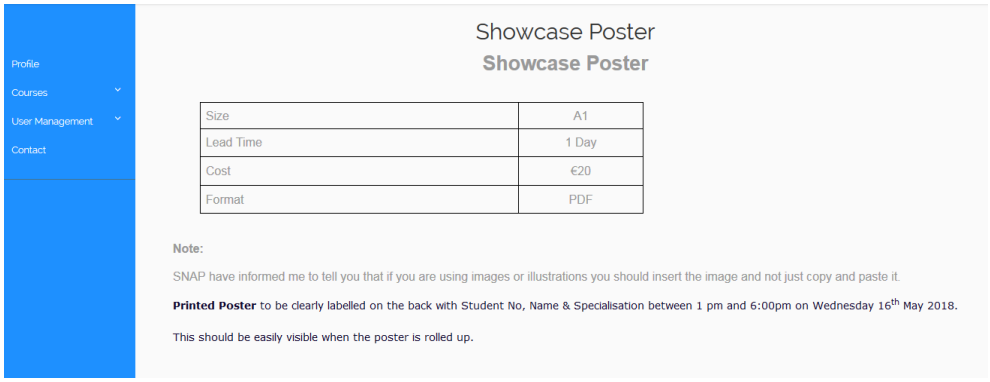
Unit Testing

There were only two methods I wanted to test here. Namely, did all course ID's and titles appear, and when I clicked 'Take Course' did the course materials appear, and in what format.

1. Choosing course.



2.



Attempt Test

This section of the application is a continuation from the above feature and falls under the 'Take Course' feature. Once the user has read through the materials, they click the 'Attempt Test' button at the bottom of the page. This loads the 'viewtest.php' file.

Feature Steps

1. **SQL Query:**

   At the top of this file is a php script to query the course database, requesting any test with the TestID that was specified in the hidden input on the previous page. The variable is declared as "$testID = $_POST['attempt-test'];". This is to ensure the corresponding test to the previous course materials is selected.

```php
$testID = $_POST['attempt-test'];
$sql = "SELECT DISTINCT test FROM course WHERE testID = $testID";
```

2. **Test Array:**

   The script then takes the test stored as a blob from the course table. In order to do this, it has to unserialize the array.

```php
$testArray = "";
$count = 0;
$result = mysqli_query($connection, $sql);
while($row = mysqli_fetch_assoc($result)){
   $temp  = $row['test'];
    $testArray = unserialize($temp);
    //  echo "<pre>";
    //   print_r($testArray);
     $count =  count($testArray);
}
```

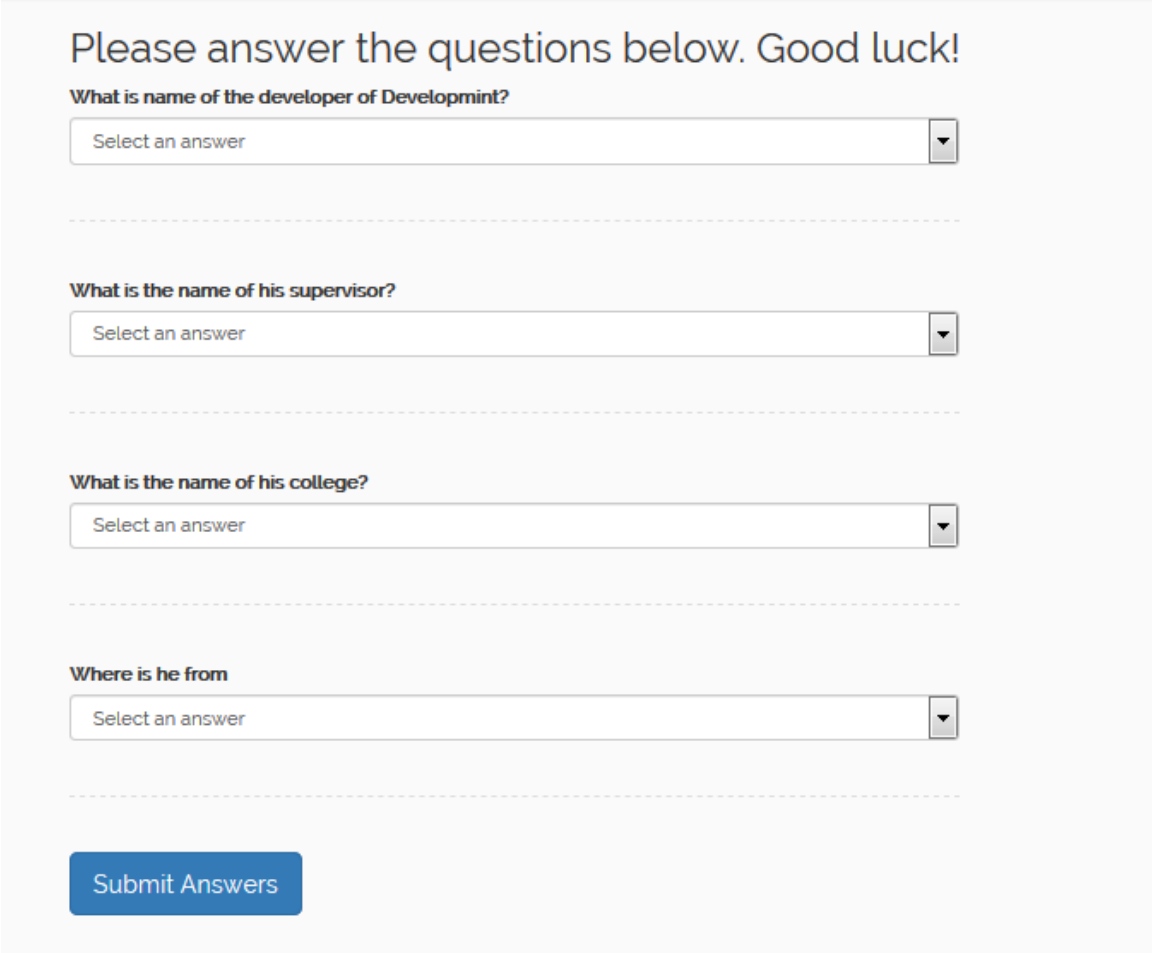
3. **Loop:**

   At the bottom of the above code snippet is "$count = count(testArray);". Count is declared because, as in the test creation section, it needs to loop through how many iterations of questions and answer options there are.

```php
<?php
$numberOfLoops = 0;
foreach ($testArray as $testArray['row']) {
    //echo  $testArray['row']['choice1'];

//}
//for ($i = 1; $i <= $quantity; $i++){
?>
    <label for='test-question'><?php echo $testArray['row']['question']; ?></label>

    <select class = 'form-control' name="Question<?php echo $numberOfLoops; ?>">
      <option value="Select an answer" disable selected hidden>Select an answer</option>
      <option value="<?php echo $testArray['row']['choice1']; ?>"><?php echo $testArray['row']['choice1']; ?></option>
      <option value="<?php echo $testArray['row']['choice2']; ?>"><?php echo $testArray['row']['choice2']; ?></option>
      <option value="<?php echo $testArray['row']['choice3']; ?>"><?php echo $testArray['row']['choice3']; ?></option>
    </select>

    <input type="hidden" class='form-control' id='actanswer' name="actanswer<?php echo $numberOfLoops; ?>"  value="<?php echo $testArray['row']['answer']; ?>">
    <div class="line"></div>
    <?php
        $numberOfLoops++;
    }

    echo "<input type='hidden' name='numberOfLoops' value='".$numberOfLoops."'>";
?>
    <button type="submit" class="btn btn-primary btn-lg" name='submit-answers'>Submit Answers</button>
</form>
```

4. **The test is produced:**

The test is shown, prompting the user to answer the questions.



Unit Testing

The main action that needed to be tested was that the test questions printed out in correct order and with all the correct answer options.

You will see more comments that are pieces of code to print the array.

```
Array
(
    [row1] => Array
        (
            [id] => 84
            [testname] => Showcase Poster Test
            [question] => What is the name of the print shop mentioned in the document?
            [choice1] => SNAP
            [choice2] => CRACKLE
            [choice3] => POP
            [answer] => SNAP
        )

    [row2] => Array
        (
            [id] => 85
            [testname] => Showcase Poster Test
            [question] => How much is the poster going to cost to print?
            [choice1] => 20 Euro
            [choice2] => 10 Euro
            [choice3] => 1000 Euro
            [answer] => 20 Euro
        )

)
```

Submitting Answers and Grading

This section of the application is a continuation from the above feature and falls under the 'Take Course' feature. Once the user has answered all question, they click the 'Attempt Test' button at the bottom of the page. This loads the 'grade.php' file.

Feature Steps

1. **PHP:**
   At the top of this file is a PHP script, it takes the number of loops, the question, the response and the actual answer, which was hidden in an input in the previous file.

```php
$correct = 0;
$incorrect = 0;
// $count = "";
// $answerCount = array($choice, $choice2, $choice3, $answer);
// echo "<pre>";
// print_r($_POST);


$id = $_SESSION['id'];
$num = $_POST['numberOfLoops'];
// echo"<pre>";
// echo $num;
```

2. **Loop:**
   The script loops through all of the answers, taking the posted number of loops as the iterations.

3. **Calculations:**
   Within the loop, the script then uses an if statement to check if the response is the same as the actual answer, it counts it. There is also an else statement to count and incorrect answer.

```php
for ($x = 1; $x <= $num; $x++) {
    $response = $_POST['Question'.$x];
    // echo $response."<br>";
    $answer = $_POST['actanswer'.$x];
    // echo $answer."<br>";
    if($answer == $response){
        $correct++;

    }elseif($response !== $answer){
        $incorrect++;
    }


} //echo "<p>You got ".$correct." questi
```

4. **Result:**
   The variable $score is declared as the number of correct iterations (correct answers), divided by the number of total loops (the amount of questions) multiplied by 100. This produces a % figure, which is given to the user upon completion of the test.

```
$score = ($correct / $num)*100;
```

Unit Testing

The main portions of this feature I needed to test were that all of the questions in the loop printed correctly, as well as the answers.

```
Array
(
    [Question1] => CRACKLE
    [actanswer1] => SNAP
    [Question2] => 1000 Euro
    [actanswer2] => 20 Euro
    [numberOfLoops] => 3
    [submit-answers] =>
)

CRACKLE
SNAP1000 Euro
20 Euro
```

3. Testing

Security Testing

OWASP Checklist

Injection:
On any input form within Deveopmint, the functional files that query the database scrub all information that has been sent. The function mysqli_real_escape_string is used on any input field or option dropdown list to prevent injection into the database. A snippet below shows an example from the create user form.

```
$fullname = mysqli_real_escape_string($connection, $_POST['fullname']);
$email = mysqli_real_escape_string($connection, $_POST['email']);
```

Broken authentication:
Developmint uses a login feature so users cannot access past the landing page without providing their correct credentials. If an attacker somehow figures out a directory of part of the application, if they try and access it, the site will redirect them back to the login form.

```
if (isset($_SESSION['id'])) {

}else{
    header('Location: login.php');
}
```

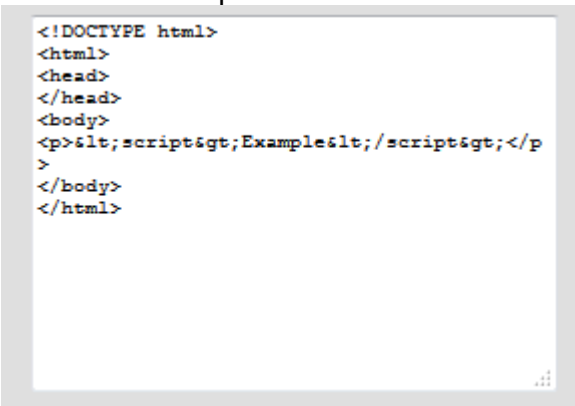
The above snippet is stored in the header, and checks if a user access a page without a session id, they will be redirected to the login page.

Sensitive Data Exposure:
As mentioned below, the apache server configuration file has been changed to stop sensitive information being displayed. This also applies to the PHP errors.

XML External Entities:
XML was not implemented within this project so was not a concern, but will be revised if it is implemented at a later date.

Security Misconfiguration:
As mentioned below as well, the header were updated within the Apache configuration file.

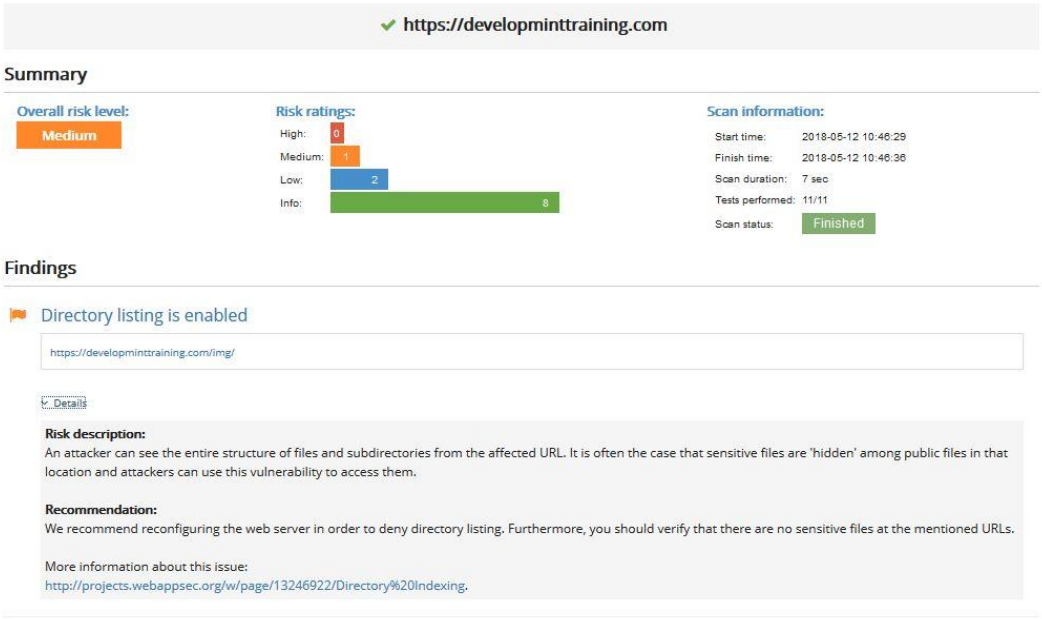
Cross-site Scripting (XSS):
As mentioned above the apache configurations have mitigated against this. Also, the TinyMCE text editor stores any document in a HTML format, however, when inputting the function <script></script>, the editor scrubs those tags to unreadable text to prevent against this. The below snipped shows the entry to the database, notice the brackets around both 'scripts' have been altered.



Components with known vulnerabilities:
Not many components were used within Developmint. The one main one being the text editor, however upon research shows there was a XSS vulnerability back in 2014, but as mentioned above, this has been updated.

Automated Penetration Testing

To test the robustness of the security to my application, I used an automated web vulnerability scanner to check Developminttraining.com to see what results were produced. I was pleasantly surprised to find that the website's security scored quite well, with only one medium priority issue and two low priority issues.



The medium issue was the directory listings. This meant that anyone could sign on to the page in the browser and add /images/ as an exampled. The attacker would then be privy to my image directory folder. To fix this, it was just some simple Apache configurations that needed to be adjusted. To access the configuration file, I opened the Putty application and got to the directory through the Linux console.

I just had to access my web root directory and set the configuration file to the below settings.

```
<Directory /var/www/>
        #Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
```

The other two low priority issues were that versions were displayed and the headers needed to be updated. The below additions were made to the config file to update the secure headers.

```
<IfModule mod_headers.c>
Header unset Server
Header always unset X-Powered-By
Header unset X-Powered-By
Header unset X-CF-Powered-By
Header unset X-Mod-Pagespeed
Header unset X-Pingback
Header always set Strict-Transport-Security "max-age=31536000; includeSubdomain$
#Header always set Public-Key-Pins 'pin-sha256="<primary>"; pin-sha256="<backup$
Header always set X-Frame-Options "deny"
```

Once these settings were applied I restarted the server to allow the changes to take place.

PHP Display Errors

For development purposes, I also had configured the PHP.ini file to display errors. This was set up to allow me to debug code. However, if something were to break, the errors expose exactly what is going on behind the application, making recon for a potential attacker much easier.

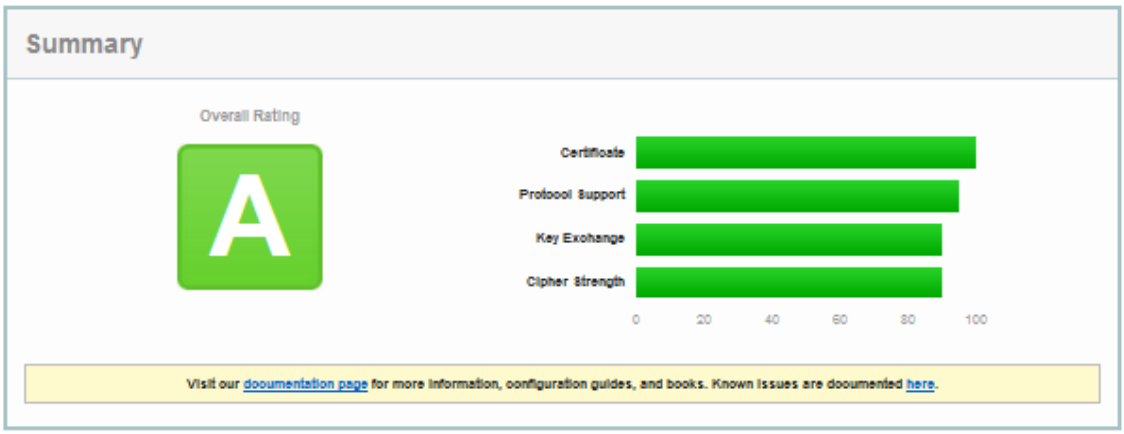
```
display_startup_errors
;   Default Value: On
;   Development Value: On
;   Production Value: Off

;error_reporting
;   Default Value: E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED
;   Development Value: E_ALL
;   Production Value: E_ALL & ~E_DEPRECATED & ~E_STRICT
```

I changed back those settings in the same Putty application as I did for the Apache settings. The semi colon means that part of the file is commented out, so it doesn't affect the file. Once again, I restarted the server to allow the changes to take place.

SSL Cert Scan

I also used a scan from the website ssllabs.com to scan my SSL certificate. The scan checks for common http issues and checks the validity of your certificate.

Developminttraining.com has a grade A certificate security grading. Which can be seen in the image below.

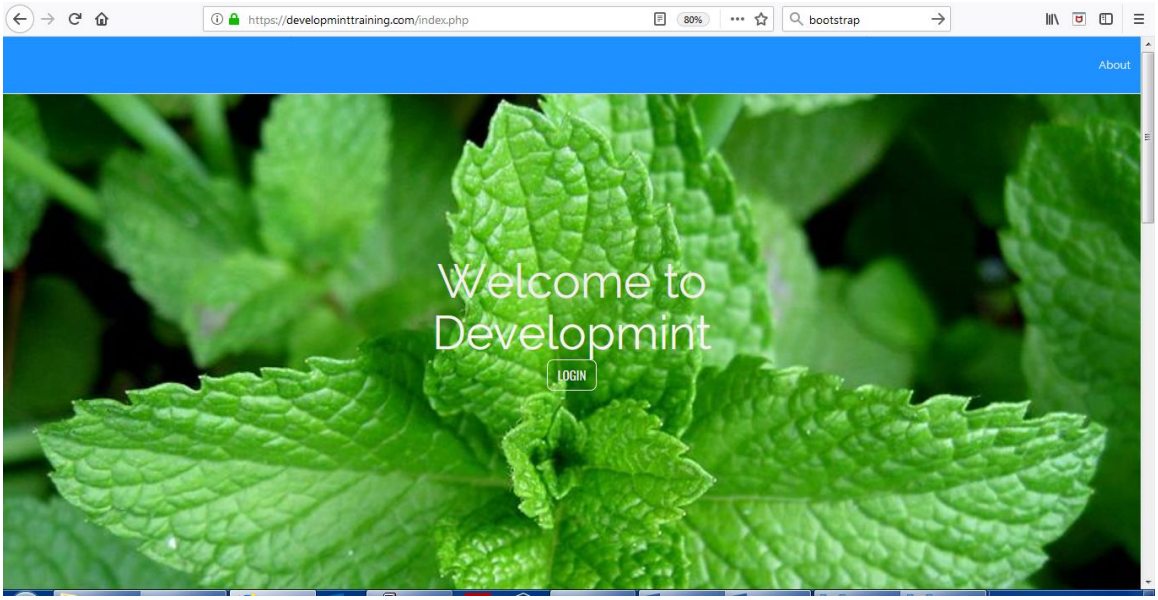


The main two vulnerabilities the scan looks for are Heartbleed and Ticketbleed. Developmint is secure against both.

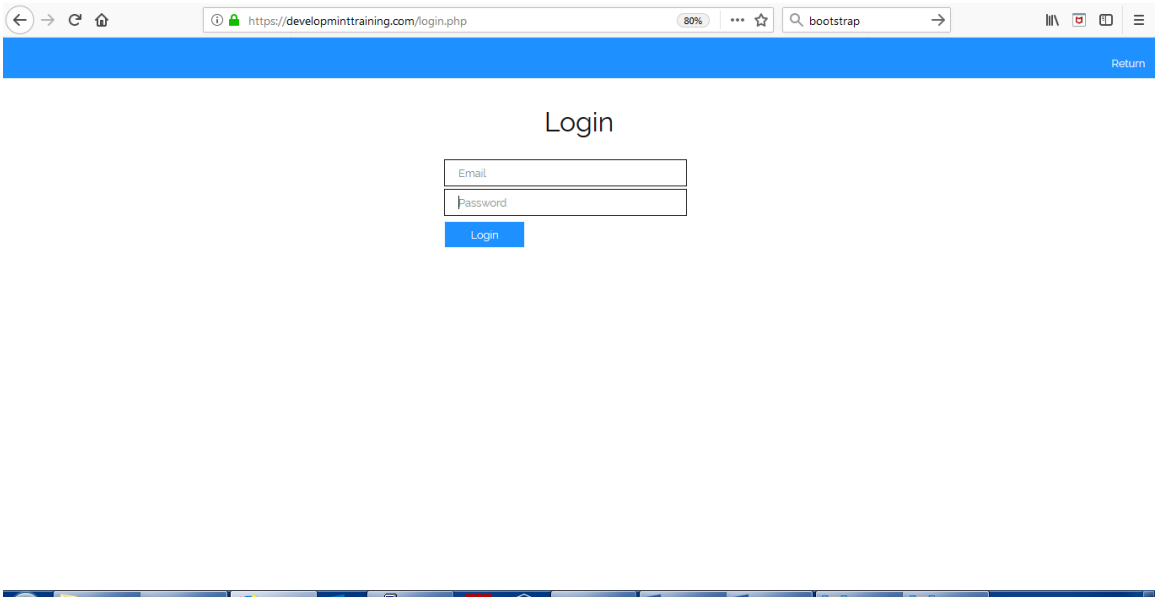
3 Interface requirements

In this section I have provided screenshots of the various pages of the application. These screenshots are included to highlight the minimalist design and user friendly interface.

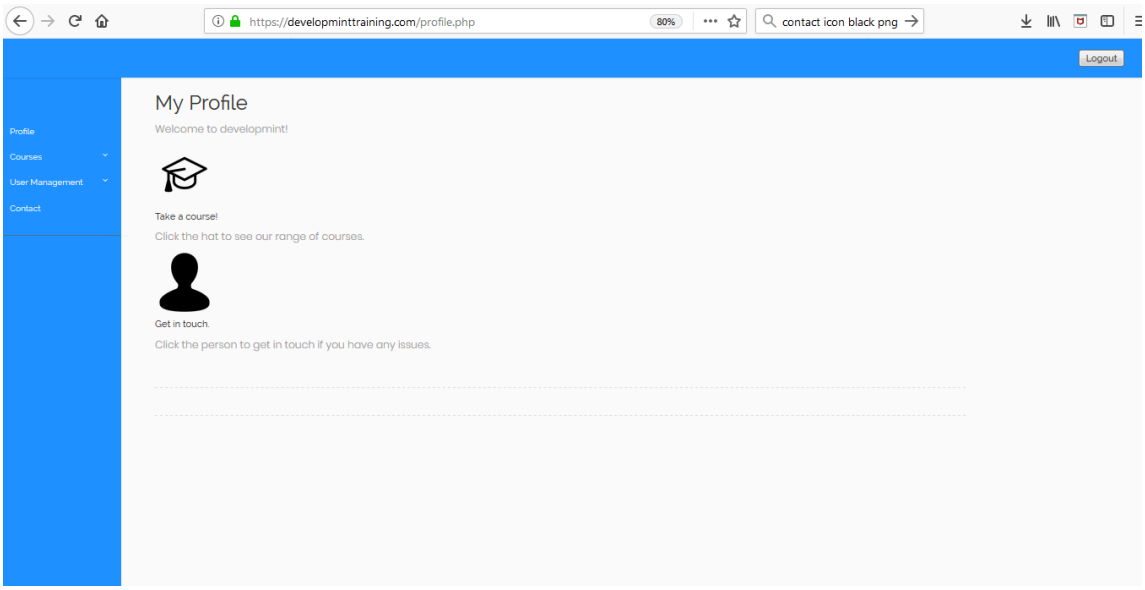
3.1 GUI



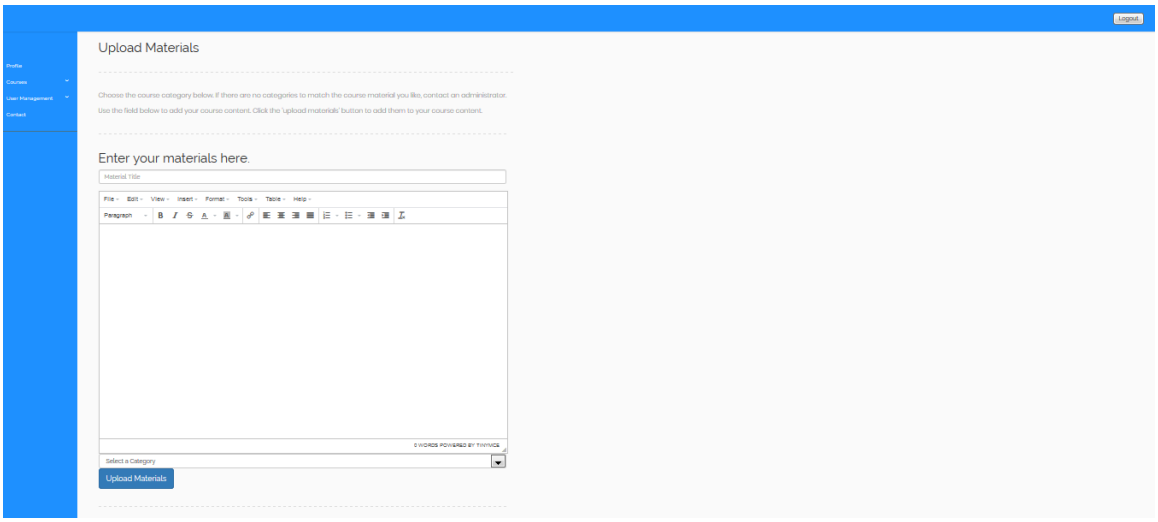
Landing page (index.php)



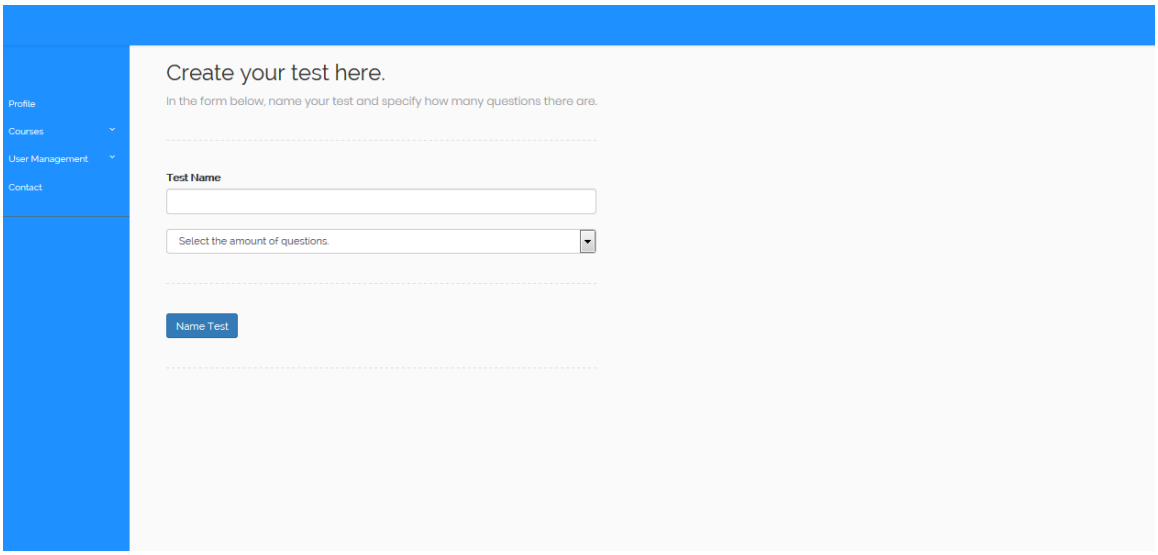
Login page.



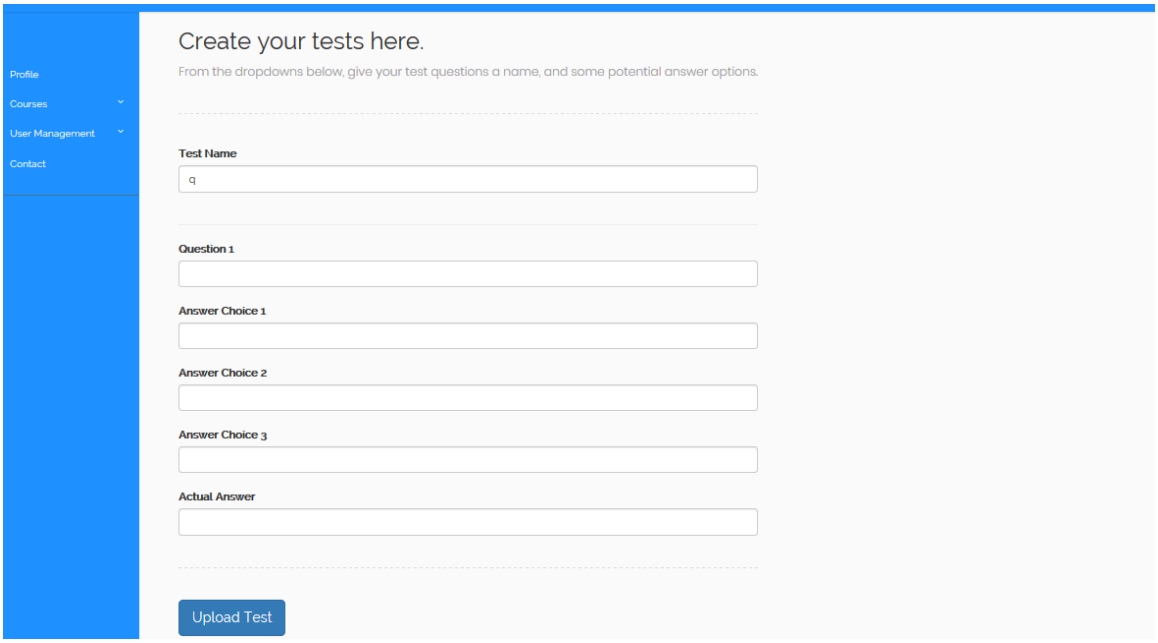
Profile page.



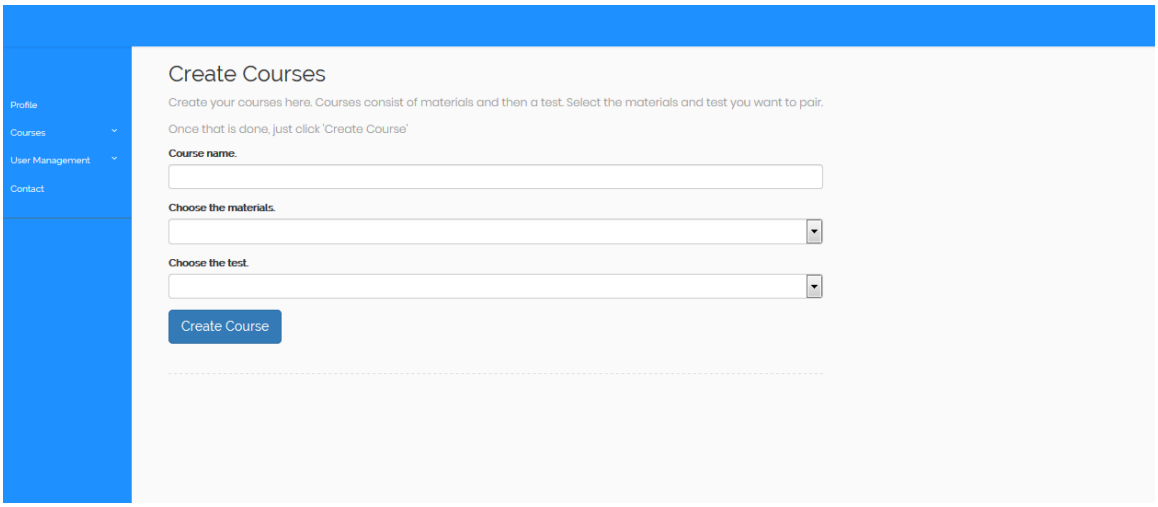
Upload Materials



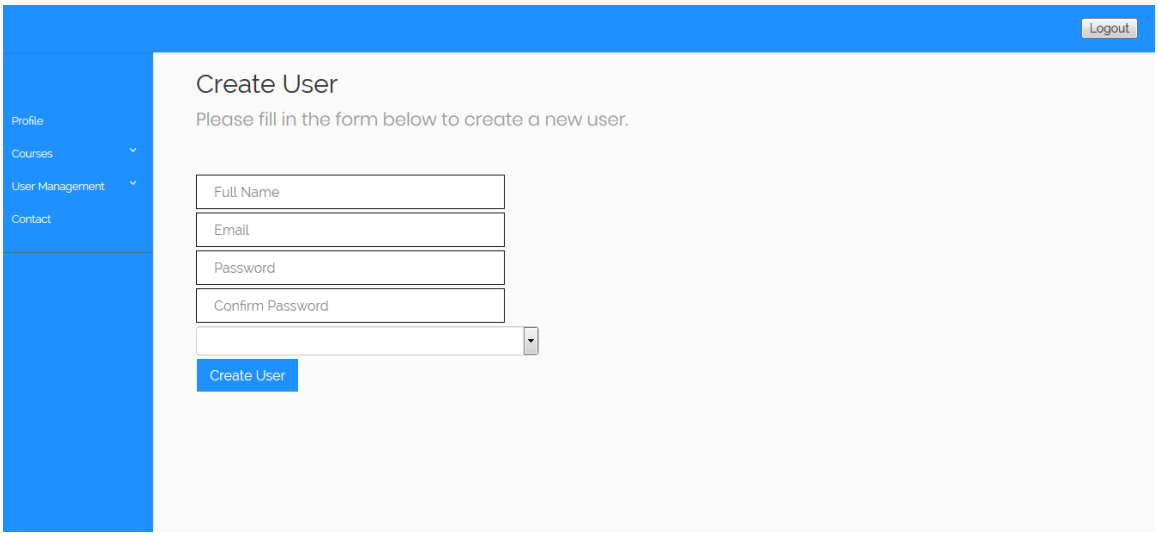
Create test part 1.



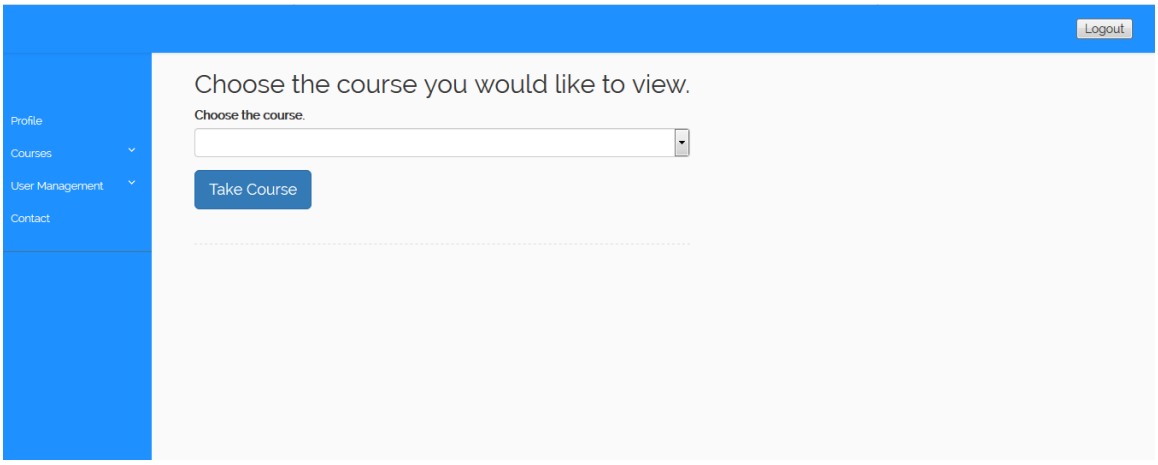
Create test part 2.



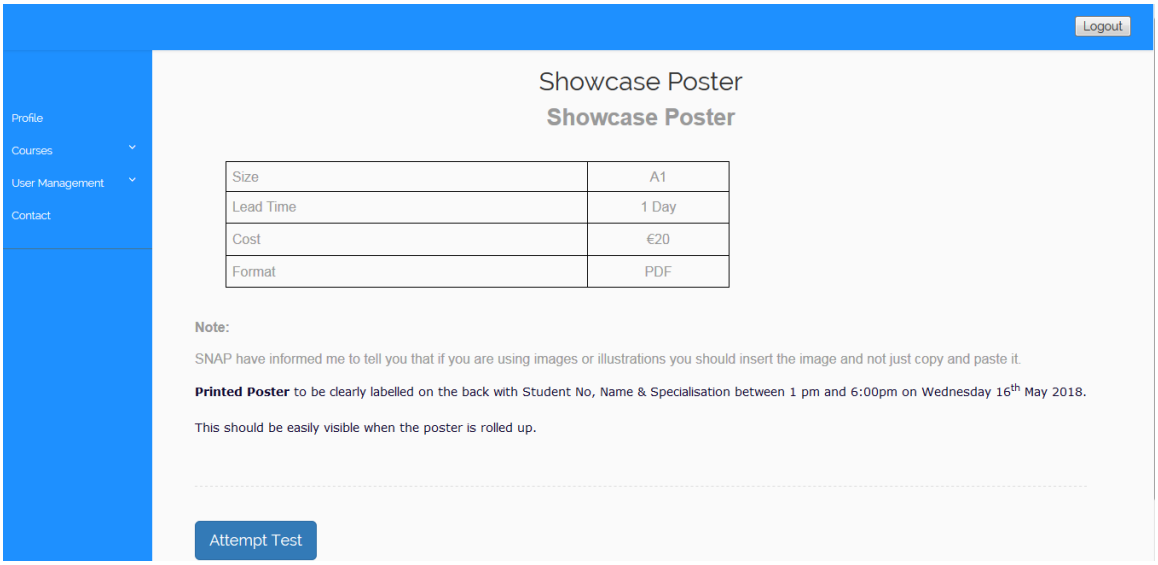
Merge the courses and tests together.



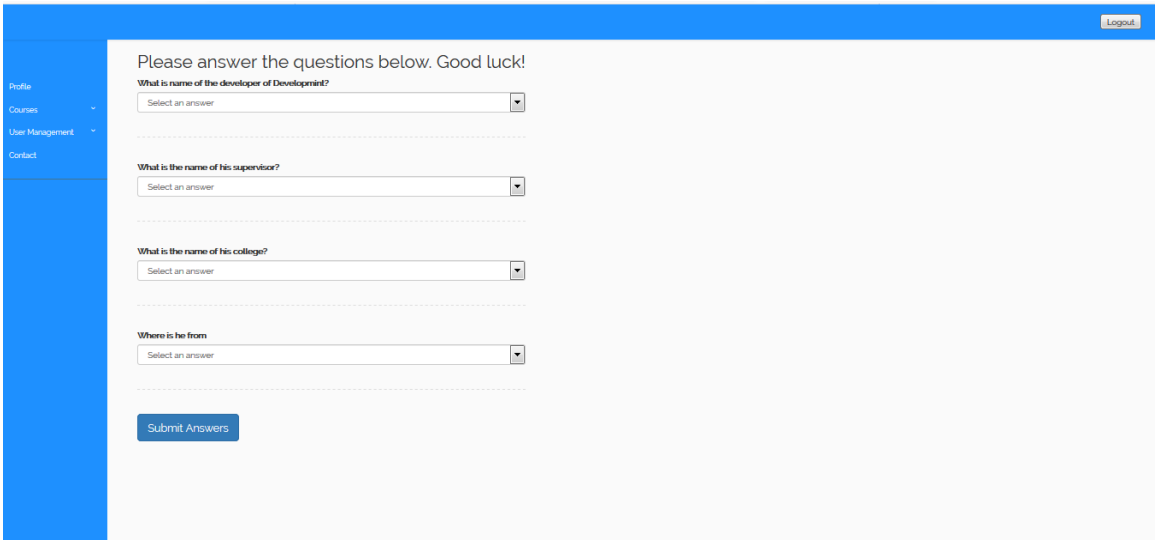
Create User



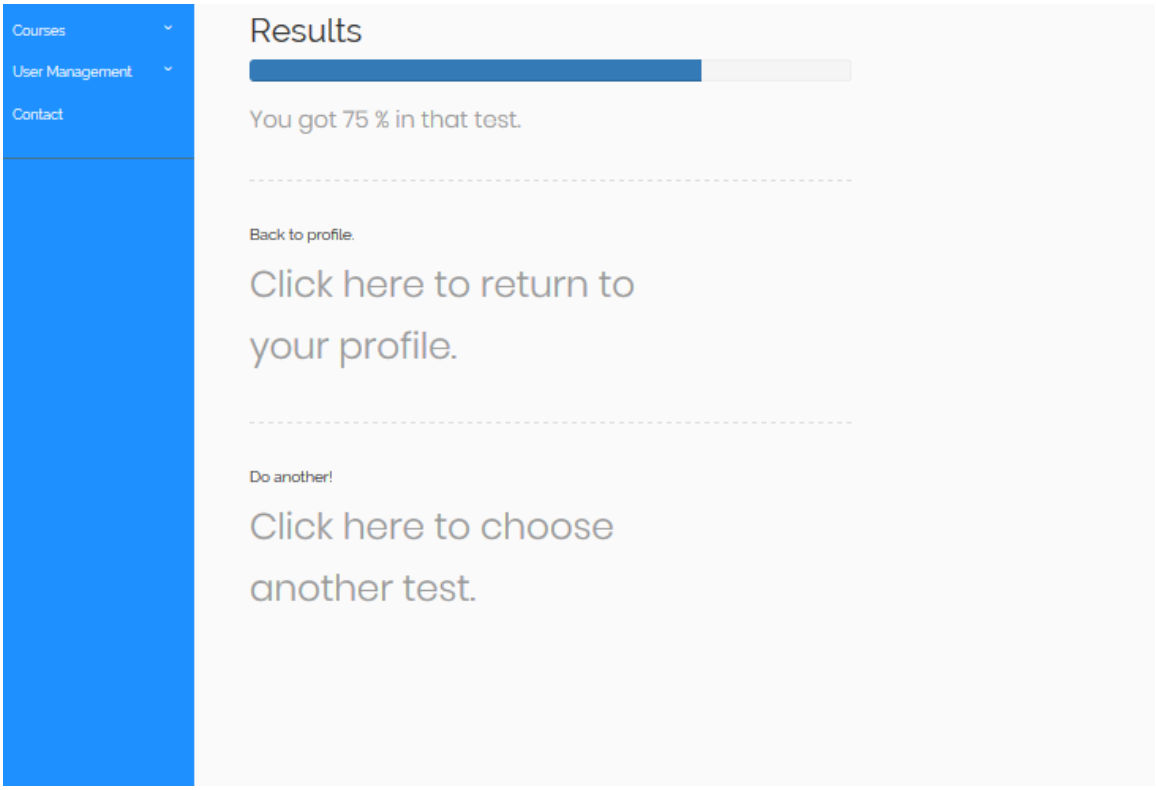
Choose the course to take.



View the materials.



Attempt the test


Courses
User Management
Contact
Results
You got 75 % in that test.
Back to profile.
Click here to return to your profile.
Do another!
Click here to choose another test.

Get results

4 System Evolution

As mentioned in my product proposal, one of the more important aspects of the application will be scalability. As more and more users from one company use the application to when multiple users from multiple users begin to use the application's service, the servers and databases will need to be able to handle these increases.

The application's hosting and database architecture may change as the growth of users increases as it will need to be more scalable.

Future Implementation

In this section I will described features that will be added into the application in its next iteration. These were features that were described and discussed as ones that will be implemented.

Assign Course

Currently, all users are able to view the courses that are uploaded. The plan is the give and administrator or trainer the ability to assign these courses to a specific user. This isn't a big deal as it does not affect security, but it would aid in tracking progress.

Update Course

The administrator will have the ability to take an already uploaded course, and change its contents and tests.

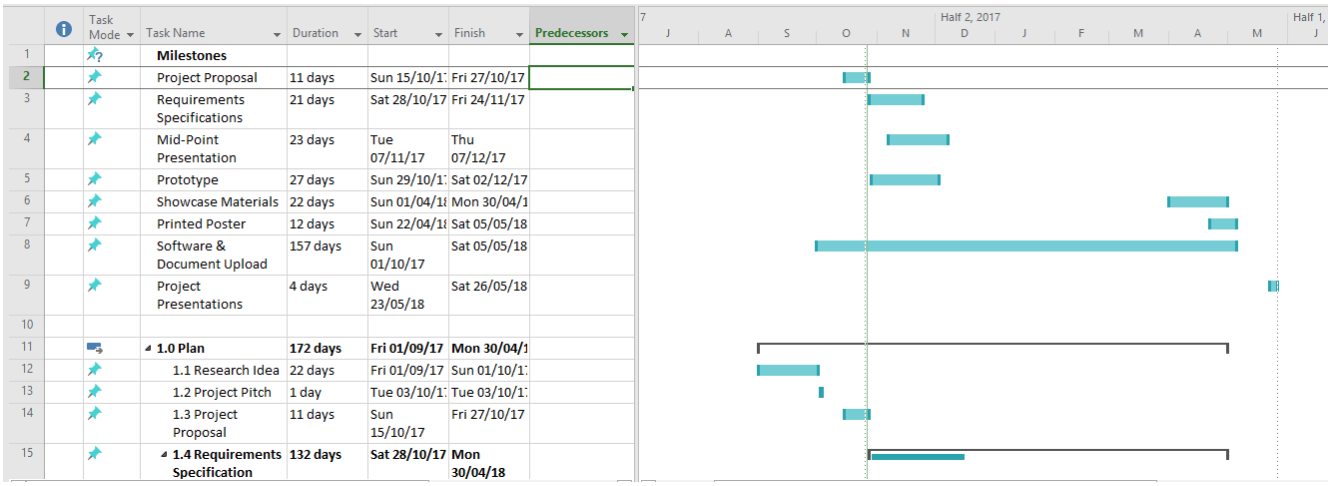
Remove Course

This will give the administrator the ability to remove a course entirely from the application.

5 Appendix

5.1 Project Plan

	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		**Milestones**				
2		Project Proposal	11 days	Sun 15/10/17	Fri 27/10/17	
3		Requirements Specifications	21 days	Sat 28/10/17	Fri 24/11/17	
4		Mid-Point Presentation	23 days	Tue 07/11/17	Thu 07/12/17	
5		Prototype	27 days	Sun 29/10/17	Sat 02/12/17	
6		Showcase Materials	22 days	Sun 01/04/18	Mon 30/04/18	
7		Printed Poster	12 days	Sun 22/04/18	Sat 05/05/18	
8		Software & Document Upload	157 days	Sun 01/10/17	Sat 05/05/18	
9		Project Presentations	4 days	Wed 23/05/18	Sat 26/05/18	
10						
11		**1.0 Plan**	**172 days**	**Fri 01/09/17**	**Mon 30/04/18**	
12		1.1 Research Idea	22 days	Fri 01/09/17	Sun 01/10/17	
13		1.2 Project Pitch	1 day	Tue 03/10/17	Tue 03/10/17	
14		1.3 Project Proposal	11 days	Sun 15/10/17	Fri 27/10/17	
15		**1.4 Requirements Specification**	**132 days**	**Sat 28/10/17**	**Mon 30/04/18**	



ProjectPlan.mpp

The plan allows time to send out a survey for feedback after testing is completed in early 2018. Free time has been allowed after feedback has been gathered to fix any issues and make modifications to the application.

5.2 Project Proposal

5.2.1 Objectives

To develop an application for any company looking to train / upskill / coach their staff via a personalised homepage. The staff being trained will have exposure to the relevant material of the training they are undertaking.

Users will be able to log on, check the training material, study it and attempt a quiz on it. Once an employee has completed the training stage and has passed the quiz, or the employee gets promoted within the company, the trainer will grant access to the next relevant material.

Trainers will have the ability to upload material and grant access to individual users based on what stage of the training they are attempting to complete. The UI for trainers

and managers will be fluid. This feature is essential as it will be used in fast paced environments so ease of use is a must.

The user will have to go through a two-factor authentication login in order to access the material. This is because the training material may contain information that is unique to that company only. The user accounts will also be locked to an IP range, meaning they will only be able to access their own homepage within the company's IP address.

This application will be available to all employees in the business, no matter the level, and will be suited to the role of that employee. For example, if the company employs people who are on the road, they can access the application site via their phone. This will allow for training to be done efficiently without the need to call be with employees face to face.

If a user finishes a course. A new course can be suggested to them by the application. This will be done on the amount of similar material covered between two courses.

I have decided to call the application "Developmint". I have called it this because the application is intended to create a fresh approach to developing and training staff.

6.1.2 Background

My whole career path thus far has been in sales and marketing. I started working as a door to door sales rep and since then have moved into an office environment. I am now a team leader of a small sales and customer service team in a field marketing company. Our company works on behalf of a lot of bigger firms and because of this, extensive product knowledge is needed to work on behalf of these clients.

Upon joining the company, depending on the department, you can spend anywhere between one day two a week in training and there can be a lot to take in. I imagine it is the same within other firms.

What I have noticed, especially since I have begun managing teams, is that certain information gets forgotten. This is especially true if a particular scenario hasn't occurred in a long time. After a while I started to issue random tests to keep the sales and customer service teams' product knowledge fresh in their heads.

It was at that point that I began to wonder if a centralised hub for everyone to log on where they can take part in quizzes would be easier than handing out paper tests.

For career progression, sessions with the training manager would have to be arranged where you would sit with him for an hour and go through different management principles. With one training manager conducting sessions with potentially seventy head office staff, the times between training and coaching sessions could be consistently planned out.

I decided it would be easier, again, to have a centralised log in for employees who are on a development program where they could access all the materials relevant to them.

6.1.3 Technical Approach

The project will be to create a web application that is designed around training, developing and upskilling employees. Proper planning will need to be done before the development will take place. This is because it will be used in a professional environment, so it will need to suit the needs of management. Continuous liaising with management and developing staff will be essential to the success of the application.

- Create mockups and wireframes
- Research the creation of two-part authentication
- All user information will be encrypted on the database
- User privileges will be given on the database
- Researching and creating multiple quizzes that are stored on the database
- When the user logs in, they will see the course material available to them based on what privileges they are given. The application should block the ability of the user to reach other material.
- Research and implement IP range blocking
    - This will be predetermined by the administrator, who will enter the IP address in a text field.
- Research 'drag and drop' uploading so a manager can plan and create a course by dragging files and items onto the screen

- Research and implement making the application 'intelligent' so it knows what material has been covered and can suggest new courses / material based on what has been read
    - Will check to see if similar material is covered in another course. If true, it will tell the user to maybe take a look at that course.
- Research and implement a function that only allows a user to attempt a test or quiz only after the material has been read. E.g., if 10% of a course's material has been covered in a previous course, that new course will be suggested.
- Implement a PHPMVC framework

6.1.4 Special resources required

As this is a software project. I don't need any specific or special resources. Most of the code will be stored on Cloud9 and GitHub. And coding will be done on my laptop and college computers.

5.3 Monthly Reflective Journals

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: September 2017
After having nine months of work placement that I was exempt from, it was a bit of a shock to the system having to give up my Tuesday and Thursday evenings again. However, I got back into the swing of things relatively quickly.
Within our first couple of weeks we were numerous assignments deep. On the second week, we had to pitch our project ideas to three lecturers. To be quite honest, I was not very confident in my idea in terms of complexity, but I pitched the idea in terms of the niche area of staff development. Having said all that, my idea got its approval with revisions.

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: October 2017
I met with my supervisor, Dominic, who gave great insight into what was expected for my project. I was lacking in confidence with what my idea was, and after some suggestions from him, I revised my proposal and met with him again. These revisions were cleared and I went back to my proposal to revise it.
I added in some extra technical details and included ideas on some of the functionality that I couldn't think of before so it made the proposal look more professional and ready. I uploaded the document on the due date.

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: November 2017
The last two weeks of November have been a struggle. The number of assignments have not dropped and it has just been deadline after deadline. For my project, I had to prepare a Requirements Specification. It took a lot of research and effort to plan out how my application was going to look. Luckily, I was able to get some of it done during my work hours as I skipped lunch breaks and stayed back as it was more feasible to do college work from there.
After uploading the document, I received an email to say we had to work on our technical document. Which was uploaded on the 30th November.

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: December 2017
With Semester One winding down and having finished the midpoint, I decided to put my focus on the remaining assessment I had for the other modules. I received my grade for the midpoint presentation, it wasn't the highest, but I felt I did a lot worse. So I was pleasantly surprised.

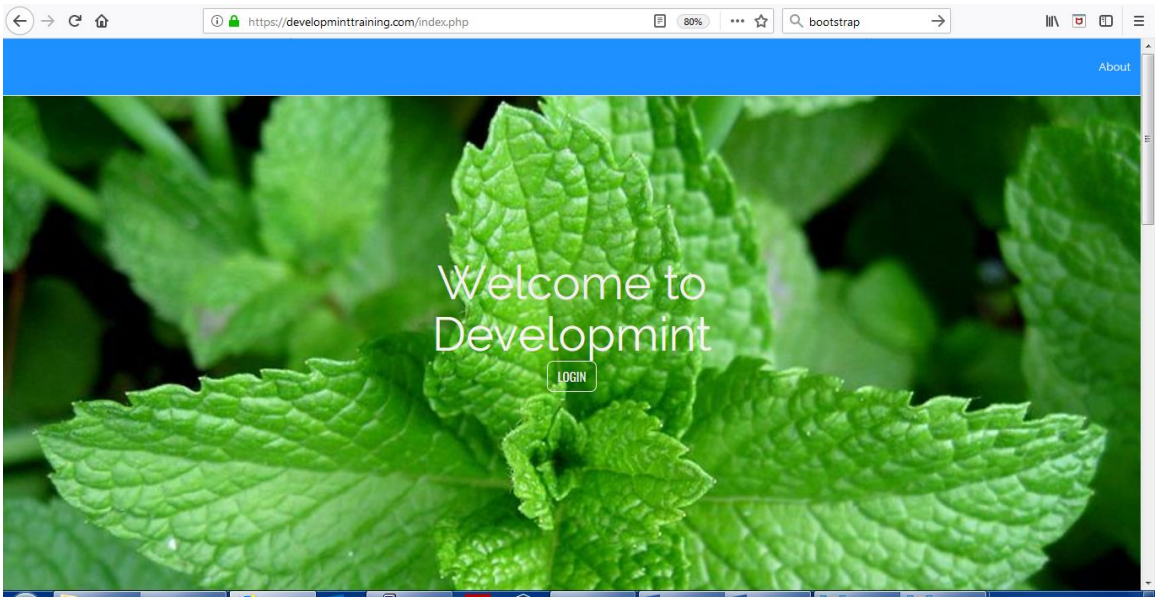
Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: January 2018
Straight after the Christmas break and it was head-first into exams. Obviously, with a combination of a lot of things, inside of college and out, I was feeling the pressure. We only had three exams and they all went very well. I had a decent grade average from the first semester so that perked me up starting the second.

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: February 2018
February was a quiet month in terms of any assessments for the other modules to I took advantage of this time to narrow down the scope of my project, it was around this time that I decided against the drag and drop feature of the application.

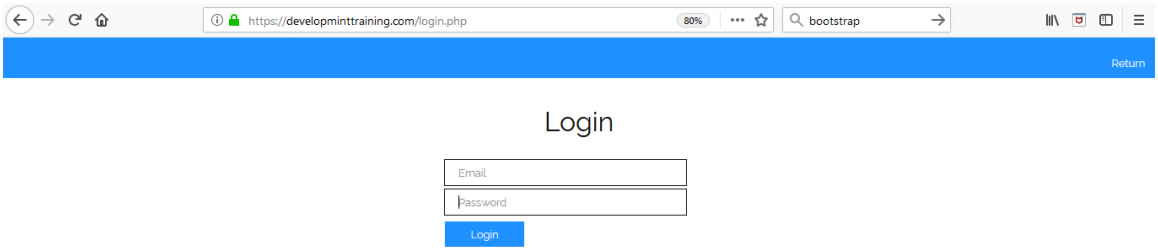
Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: March 2018
The quietness of February was more than compensated for in March, with multiple assessments due that were quite complicated. I had to put the project on hold again to accomplish these.

Reflective Journal
Student name: Mark Kenny
Programme: Bsc Computing
Month: April 2018
Because we are fourth years we had our exams this month instead of in May like usual. This was to give us time to finalise out projects afterwards. The exams were tougher than the ones in January but I think I did okay. I planned out the next few weeks in terms of annual leave from work to give me the biggest advantage possible heading into the final month…

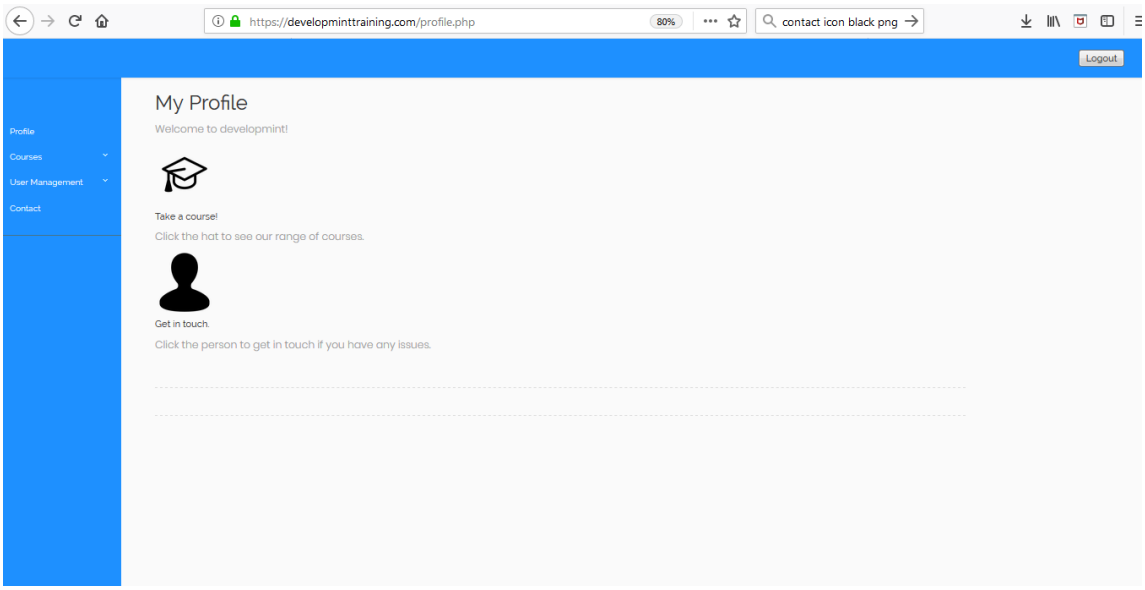
5.4 User Manual



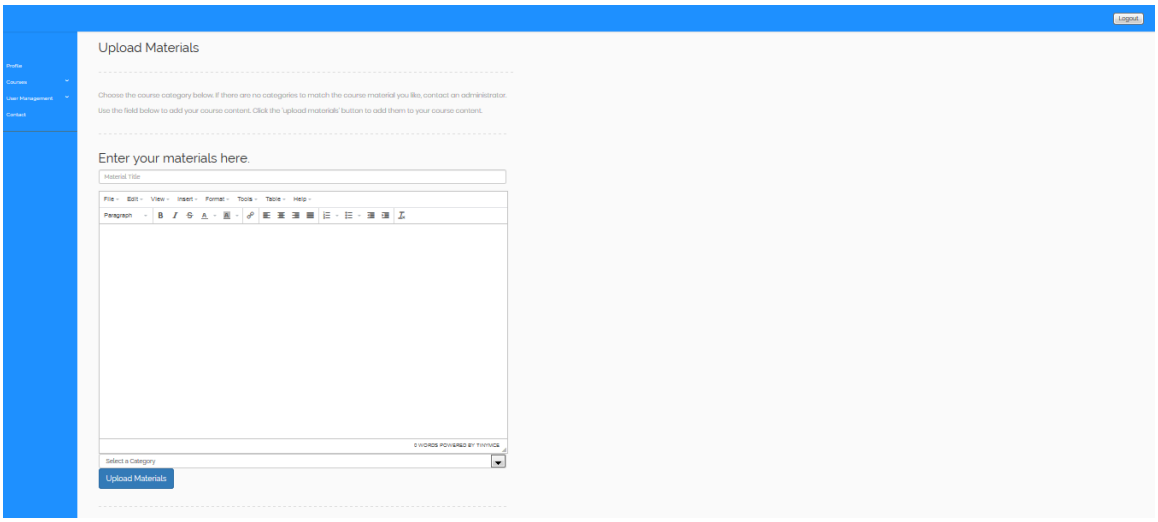
Landing page (index.php)



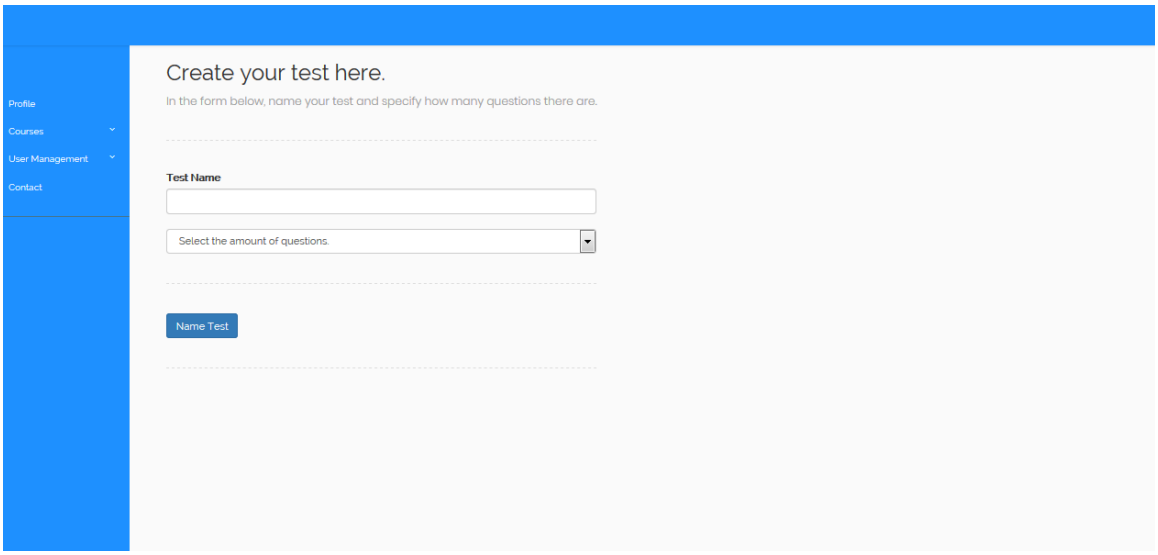
Login page.



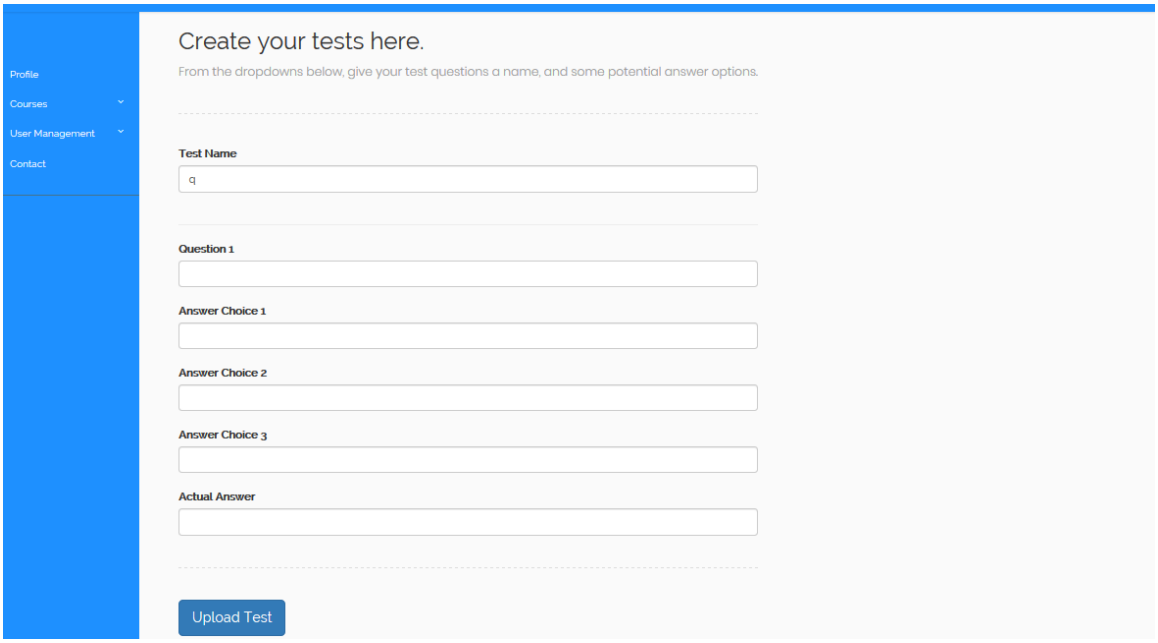
Profile page.



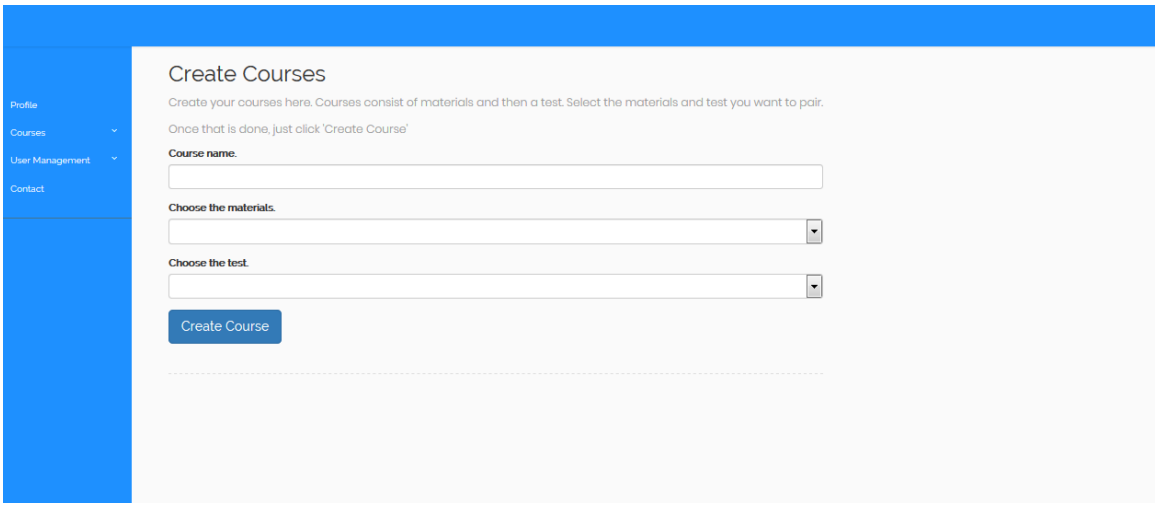
Upload Materials



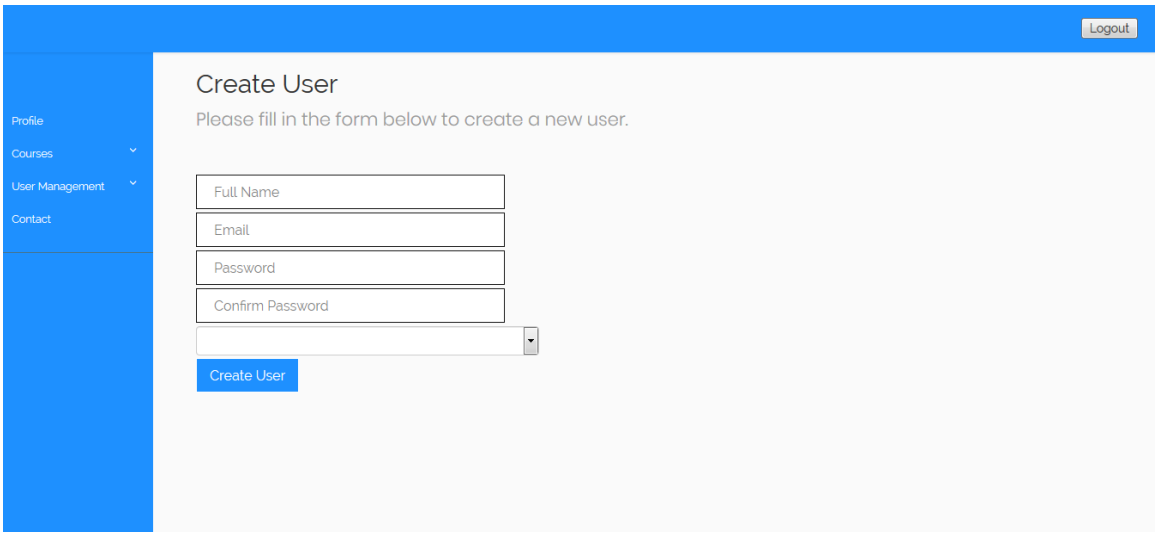
Create test part 1.



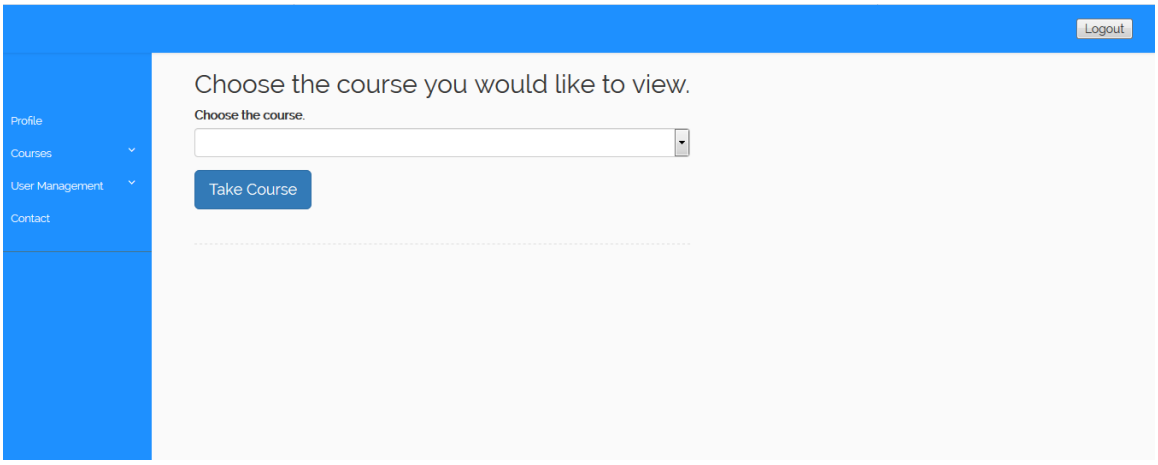
Create test part 2.



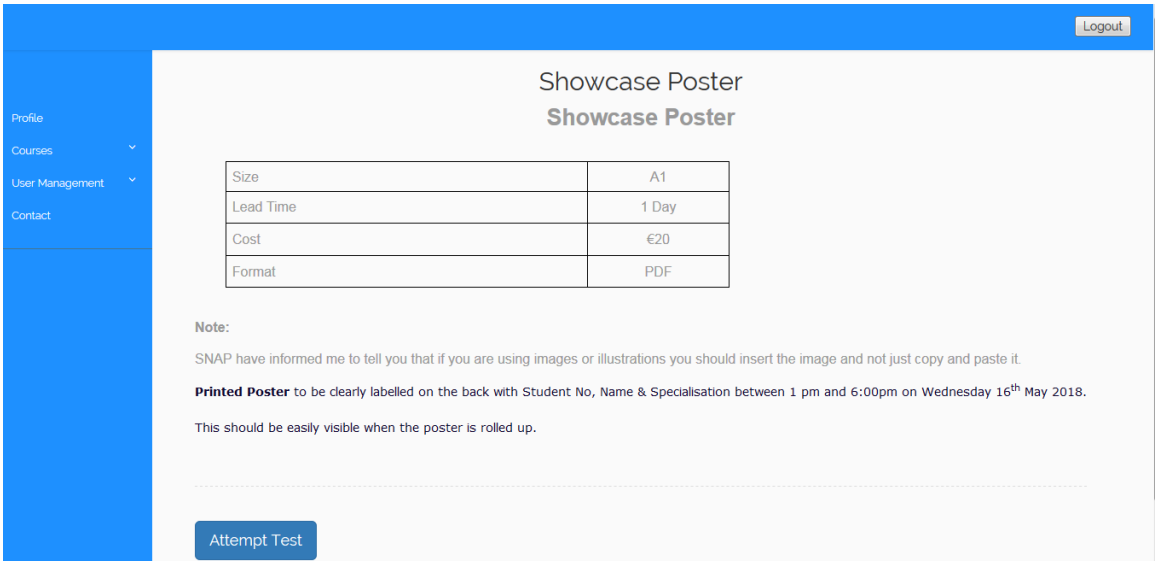
Merge the courses and tests together.



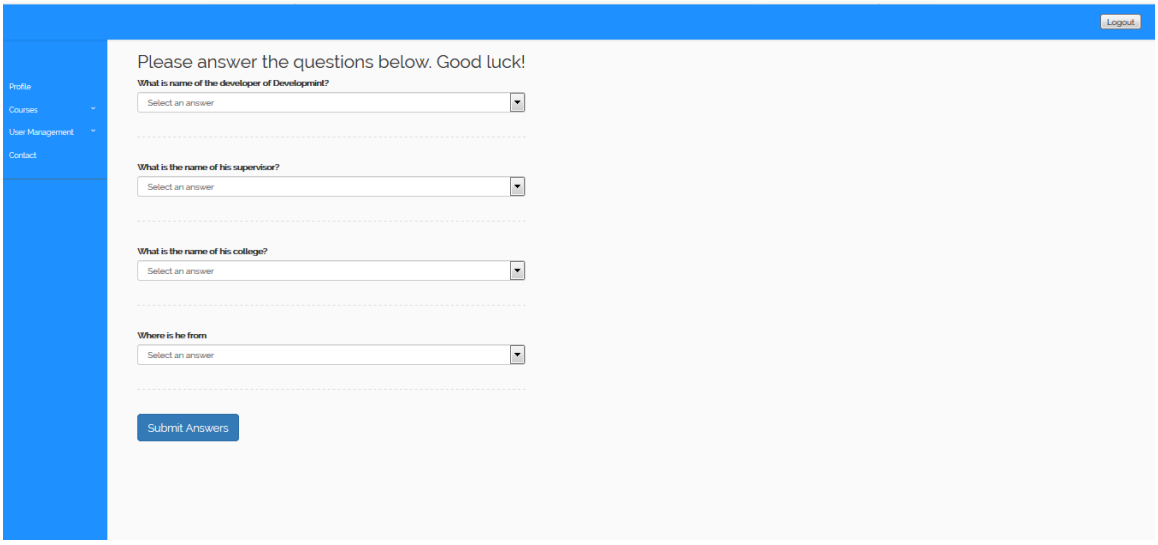
Create User



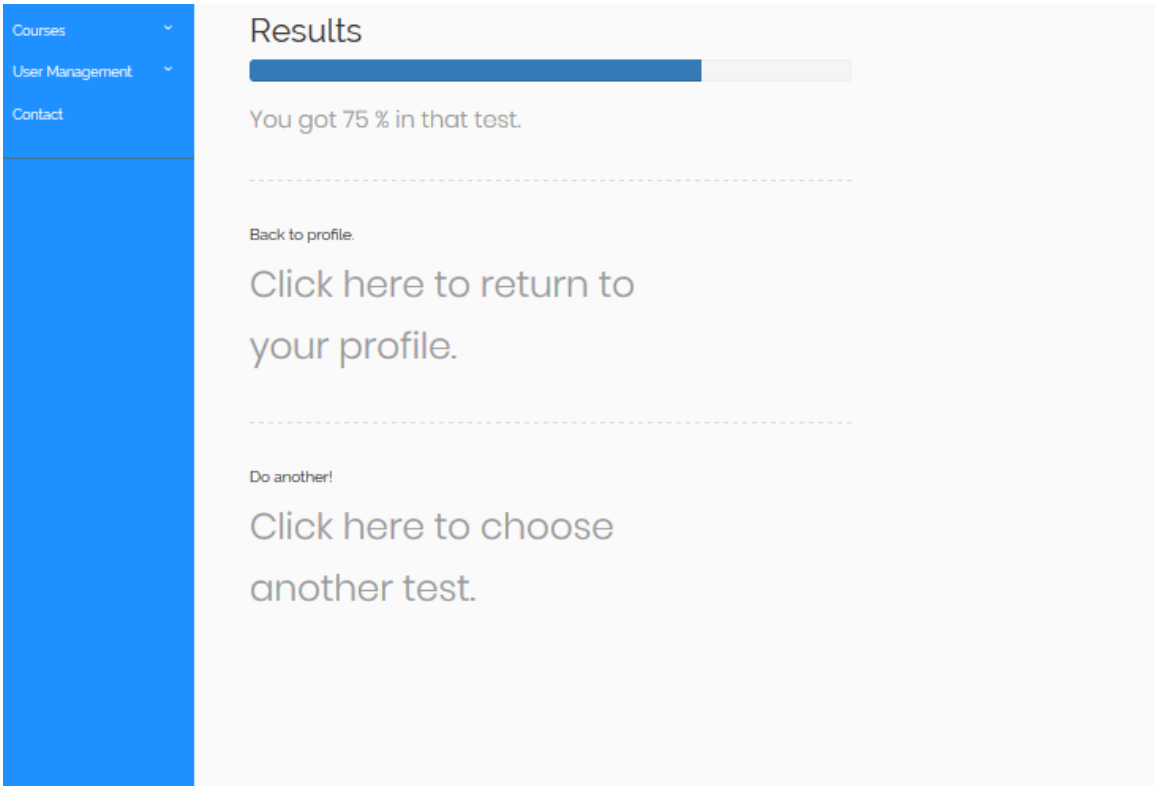
Choose the course to take.



View the materials.



Attempt the test


Courses
User Management
Contact
Results
You got 75 % in that test.
Back to profile.
Click here to return to your profile.
Do another!
Click here to choose another test.

Get results

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: **Mark Kenny**
Student ID: **X14107791**
Supervisor: **Dominic Carr**

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature:_____Mark Kenny_____
Date:__13
52018_____

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,