

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: Shane O'Mahony
Student ID: 10359651
Supervisor: Josephine Andrews

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: Shane O'Mahony

Date: 13 – 5 - 2018

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,

What constitutes plagiarism or cheating?

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks

and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism

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. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and

- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

2018

Edyoucate.ie final report

X10359651

NATIONAL COLLEGE OF IRELAND

TABLE OF CONTENTS

Table of Contents	7
Executive Summary	9
Introduction	10
Background	10
Aims 10	
Technologies	11
High level Structure overview	11
System	13
User Requirements Definition	13
Requirements Specification.....	14
Functional requirements	14
Use Case Model	14
Requirement 1: user registration.....	15
Requirement 2 Upload materials	17
Requirement 3 download materials	19
Requirement 4: user registration.....	21
Requirement 5: Creating forum posts	23
Requirement 6: Viewing and commenting on forum posts.....	25
Non-Functional Requirements	27
Performance/Response time requirement	27
Availability requirement	27
Recover requirement	27
Security requirement	28
Maintainability requirement	29
Extendibility requirement.....	29
Reusability requirement	29
Interface requirements	30
GUI 30	
Implementation and Feature development	38
AWS environment development.....	38

Secure login and session management	40
Unit test	42
Material upload and download	43
Unit test	44
Forums	45
Unit test	47
System Architecture Class Diagram	48
Penetration testing	49
Further development or research.....	52
Appendix	52
Project Proposal	52
Table of Contents	54
Introduction and Background.....	54
Objectives	55
Technical Approach.....	56
Special Resources Required.....	58
Project Plan.....	59
Technical details	62
Evaluation	63
Monthly Journals	64
September	65
October	65
November	66
December.....	66
January	67
February	67
February	68
March	68
April	69
May	69
User manual.....	70
References	79

EXECUTIVE SUMMARY

The scope of this project pertains the development of a fully scalable and secure means of content sharing for college students and curious individuals alike. The application has a simple user interface which primarily provides the capability for users to securely upload and download college materials. There is a forum page for discussion on each category where individuals can ask questions and engage in conversation.

To minimize cost for development I developed the application on a LAMP () stack which is open source and free to use.

INTRODUCTION

BACKGROUND

The inspiration for this concept came from my own experience of being in 3rd level education for 7 years. I found the material we are provided with in college to be extremely beneficial, however on further self-study and research of a given topic, I found it hard to find related material on a similar academic level to give me some perspective. This led me to reaching out to friends and family in similar courses to find additional material from their courses which I could use to enhance my own understanding of a given topic. I found the combination of these different materials and approaches to be very helpful and thought if I could widen the pool of contributors it would be mutually beneficial for all involved.

After conducting some research in the area, I found nothing to specifically facilitate this kind of content sharing apart from the colleges own in house content sharing systems such as Moodle or Blackboard which are only accessible by students of that particular college and only allowing students to access their own courses.

Having a keen interest in the open source community I felt I could develop an application that would be available to all who are interested and all who care to contribute that could become an extremely useful tool in every students arsenal.

I also feel that an application like this will not just benefit current students but also prospective college students who are unsure of what course they would like to start. Again, taking from my own experiences; when I completed my leaving certificate in 2010 I was very unsure of what path I wanted to take and what course I wanted to commit 4 years of my life to. If I had access to an application such as this, I feel I could have made a much more educated decision and also taken away some of the stress and fear of the unknown involved in the transition from 2nd to 3rd education.

AIMS

Given this problem I have decided to create a web application for students around the country to share their course materials. First and foremost, the application should have a signup form where the user can input their username and create a password in order to create an account. Once the user has created an account, it should have a secure login once they return. The application will offer a clean interface which will be divided into categories and courses which will be easily identifiable and accessible. It should provide a section for students to upload their course materials such as slides, videos, past exam papers, past projects etc. All materials should be easily downloadable or viewable from this point. The application should scan all uploads for malicious software to protect users when downloading materials to their local machines.

I have also incorporated a forum. This will give the users more space to ask questions and have open discussions about different topics.

I will primarily target the Irish market of over 17,000 students and countless prospective students, but can see no reason why it could not be pushed out on a global scale.

TECHNOLOGIES

First and foremost, I started by clearly defining the user requirements, requirements specifications, functional requirements and non-functional requirements in the requirements specification document. This granted a clear path and allow me to meet the goals and deadlines laid out in the Gantt chart. I wanted to use as many open source resources as possible, so I started by creating a LAMP stack in AWS. To do this I launched an EC2 and instance and installed Apache, MySQL and PHP. Once the LAMP stack was set up, I then began working on logins, registrations and. At this point I will carried out some penetration testing on the logins using the Burp suit to validate the security of the application. Once the session management was under control and penetration testing has complete, I will began creating and defining the structure of the application by developing the PHP page templates for each section such as course type, content type etc. Within each section I added an upload, so users can upload their materials.

Technologies and services utilized:

HTML, CSS and JavaScript – for the frontend development

PHP – server-side scripting language

MySQL – database language

PHPMyAdmin – This is the localhost database

AWS – Amazon Web Services

EC2 – Elastic Cloud Compute, used for hosting the site

Route53 – used to handle the traffic routing from the domain to the site

VirusTotal public API – a service for scanning files for known malicious code

Cloud 9 – this is the cloud based IDE I used to develop the project

HIGH LEVEL STRUCTURE OVERVIEW

Below I will outline a high level over of the overall structure of the application. I will go into further detail on this further down the document.

Hosting and Deployment:

I started by creating a LAMP stack on the AWS platform for hosting the application. This pertains installing Apache, MySQL and PHP on a Linux server. The browser will send a request to the Apache server which will then pass the request to the PHP scripts to retrieve the data from the database to generate the webpage which is then sent back to the browser via the Apache server. I used Route53 DNS which connects the domain name to the LAMP stack. I am utilizing an EC2 t2.micro instance and a localhost database in the form of PHPMyAdmin. I have 2 separate databases, 1 for the main application and one for the forum. I have installed and tested an SSL certificate and made traffic over 443 mandatory.

Frontend:

For the frontend development, I will use HTML5, CSS3 and JavaScript.

Backend:

The server-side scripting language I have decided to use is PHP. I chose this language because there is a ton of resources and useful libraries I can call upon and it is compatible with Apache. When a user uploads a file, it calls a function which adds an entry to the database and gives it a default value of 0. Another function uploads the file to VirusTotal for scanning. I then created a script which periodically checks for a result. Once the scan is complete, the function returns an array which determines the state of the file. If the file is clean the default value in the database is switched to 1 and the file remains on the server. If the file is found to be malicious the entry is removed from the database and the file is removed from the server.

API's:

As mentioned above I will be leveraging the VirusTotal public API. This free API allows for 4 requests per minute on the public version, however it is possible to request an unthrottled private API with a good enough use case.

SYSTEM

USER REQUIREMENTS DEFINITION

Edyoucate.ie will be designed to make it as easy and as clear as possible for users to locate specific topics/categories so they can upload or download the necessary materials. There are 2 types of user access for the application; Non-registered users and registered users:

1. **Non-registered users:** these users have not signed up to the website through the signup form and thus will have restricted access. These users will have the following privileges:
 - Option to signup – These users will have the options to set up a free account using the signup for which will then convert them to a registered user.
 - Register for forum – The user can register of the forum.
 - View forum posts – These users can view the forum posts.
2. **Registered users:** These users have signed up through the signup form and now have access to the main functionalities of the application. Registered users will have the following privileges:
 - View lists of categories and content – These users can view all content on the application.
 - Upload and download materials – They can upload and download materials to any category or topic.
 - View, create and comment on forum posts – Registered users will have full access to the forums for viewing and creating posts and also for commenting on existing posts.

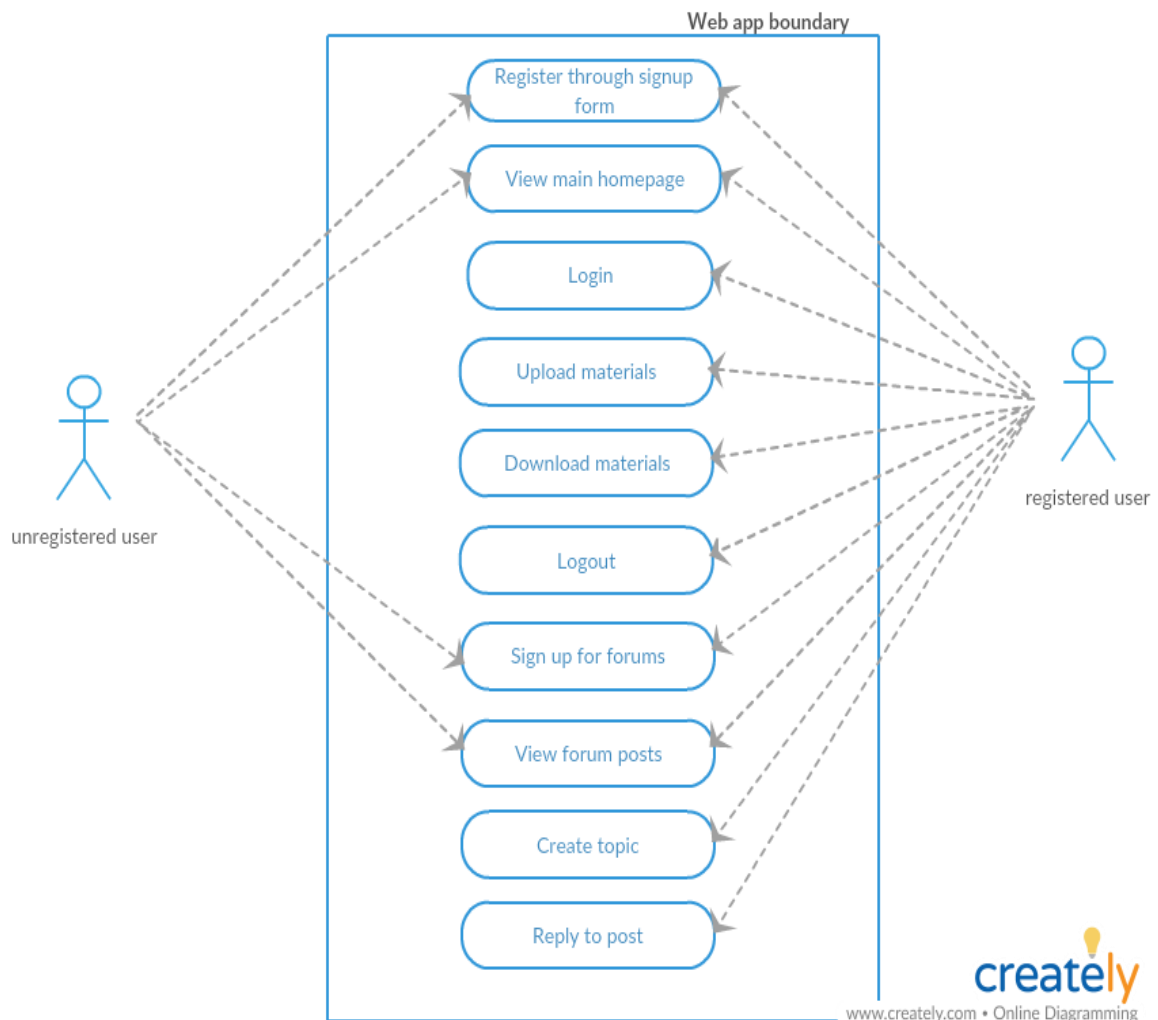
REQUIREMENTS SPECIFICATION

Edyoucate.ie has been designed and developed with ease of use being a number 1 priority. It has a clean and minimal interface with no cluttering of elements. On visiting the site, users will not be asked to sign up, however once they attempt to navigate to the categories section to view the material, they will be redirected to the register page. I have developed the forum as a separate entity so users do not need to sign up in order to view forum posts. As the forum is a separate entity, users do not need to be registered for the main application before signing up. There will be no training required to use the web application. If users do have any questions or issues with navigating the site or using the functionalities, once registered, they can reach out to other users via the communications channel, that being the forum.

FUNCTIONAL REQUIREMENTS

In this section we I will break down each function which can be performed by each type of user.

USE CASE MODEL



REQUIREMENT 1: USER REGISTRATION

DESCRIPTION & PRIORITY

This requirement is ranked 1. The user needs to register for the web application in order to utilize the functionalities of the web application.

USE CASE

Scope

The scope of this use case is to provide users with a means of signing up via a GUI in order to use the functionalities of the application.

Description

This use case describes what a non-registered user can do so they can become a registered user.

Flow Description

Precondition

The web application is idle on the home page awaiting user interaction.

Activation

This use case starts when a non-registered user clicks on the register button.

Main flow

1. The system identifies the user interaction and displays an option to register or login. See A1.
2. The non-registered user clicks the register option
3. The system displays the register form.
4. The non-registered user fills in the required fields (name, email, password, username) and clicks the save button. See A2.
5. The system then saves the user details to the database and redirects the now registered user back to the home page. See E1.
6. The now registered use can utilize all functionalities of the system and the system awaits interaction.

Alternate flow

A1 : User is already registered

1. The user clicks the login option.
2. The system displays the login portal.

3. The user inputs their credentials and clicks login. See E2.
4. The main flow then continues from point 1.

A2: Non-registered does not register

1. The non-registered user clicks cancel
2. The system then reverts to the home page where the non-registered user can navigate site without functionality.
3. The non-registered user then clicks a function and main flow continues from point 1.

Exceptional flow

E1: Non-registered user inputs invalid data in form

1. The system throws an error asking user to input valid characters/username/password.
2. The user inputs valid characters/username/password and clicks save.
3. The use case continues at position 5 of the main flow.

E2: Registered user inputs incorrect login credentials

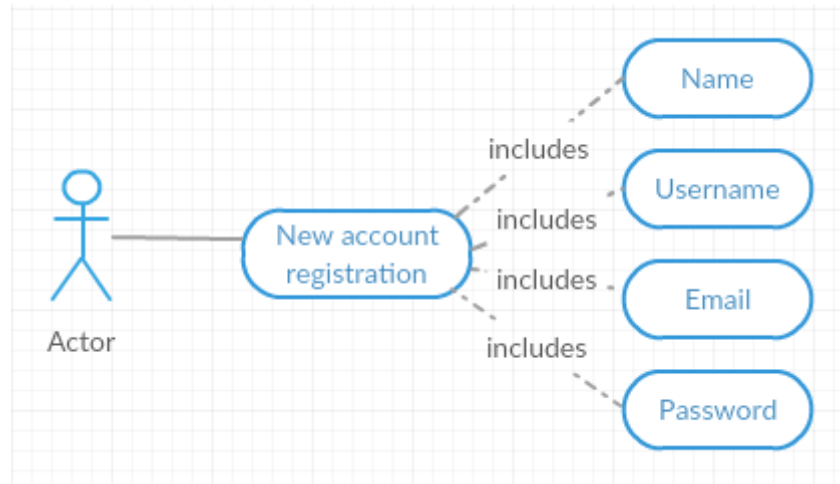
1. The systems checks database and returns error “invalid credentials”.
2. The registered user inputs the correct credentials and the main flow continues from point 6.

Termination

This flow is terminated when the user is successfully registered and returned to the home page.

Post condition

The system then awaits the next user interaction.



REQUIREMENT 2 UPLOAD MATERIALS

DESCRIPTION & PRIORITY

This is ranked 2 as it is one of the primary functions of the application. This allows registered users to upload materials for others to download and utilize.

USE CASE

Scope

The scope of this use case is to allow registered users to upload materials in a secure manner.

Description

This use case describes the process of how a registered user can upload materials to specific topics/categories.

Flow Description

Precondition

The user is logged in as a registered user and the system is awaiting interaction on the home page.

Activation

This use case starts when a registered user selects the upload option from the navigation bar.

Main flow

1. The system identifies the registered user interaction and displays the upload form. See E1.
2. The registered user selects the category and topic they would like to upload to.
3. The system then displays the upload box.
4. The registered user the drags the desired file to the upload box and clicks upload.
5. The system calls the TotalVirus API and scans the file being uploaded and once scanned and cleared the system displays a message to confirm upload. See E2.
6. The registered user clicks save. See A1
7. The system then uploads the file and displays a upload successful message.
8. The registered user clicks OK.
9. The system then redirects the user to the location for the file upload.

Alternate flow

A1 : Registered user clicks cancel on the upload

1. The system then displays a message asking if they are sure they want to cancel.
2. The registered user clicks yes.
3. The system returns the registered user to the home page and the use case continues at position 6 of use case 1 of the main flow

Exceptional flow

E1 : A non-registered user clicks the upload.

1. The system displays a message saying you are not a registered user and redirects to the register form.
2. The use case continues at position 3 of use case 1 of the main flow

E2 : The upload contains malicious software identified by the TotalVirus API.

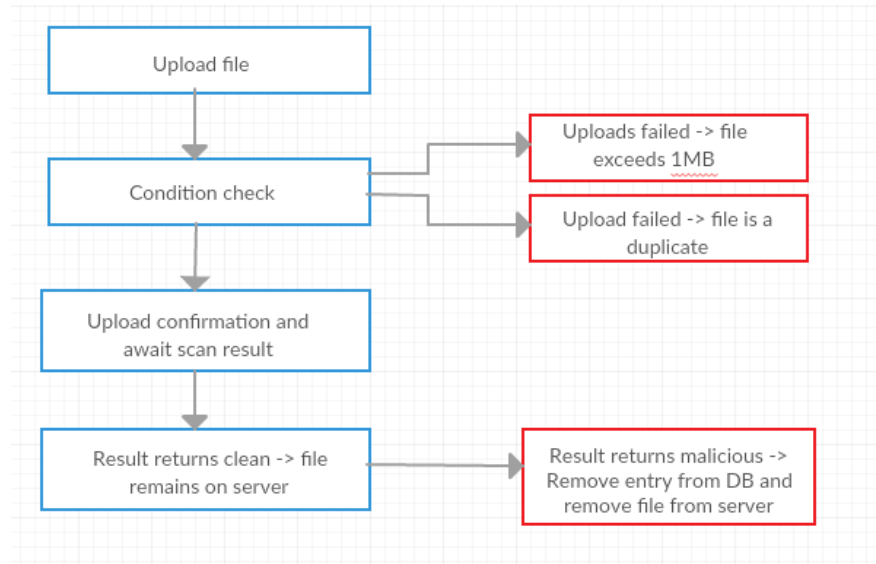
1. The system displays a message saying this upload contains malicious software and cannot be uploaded.
2. The user clicks ok.
3. The main flow continues from point 1.

Termination

The system redirects the user to the upload location.

Post condition

The system then awaits the next user interaction.



REQUIREMENT 3 DOWNLOAD MATERIALS

DESCRIPTION & PRIORITY

This is ranked 3 and just as important as the previous requirement of uploading. This allows registered users to download materials to utilize.

USE CASE

Scope

The scope of this use case is to allow registered users to download materials to their local machines.

Description

This use case describes the process of how a registered user can download materials from specific topics/categories.

Flow Description

Precondition

The user is logged in as a registered user and the system is awaiting interaction on the home page.

Activation

This use case starts when a registered user navigates to a given topic.

Main flow

1. The system identifies the registered user interaction and displays the all materials for the given topic.
2. The user views the description of the topic and then clicks download. See E1.
3. The system displays a message asking if they are sure they want to download.
4. The user clicks yes. See A1.
5. The system then initiates the download to the users local machine.

Alternate flow

A1 : Registered user clicks cancel on the download.

1. The system then displays a message asking if they are sure they want to cancel.
2. The registered user clicks yes.
3. The system then cancels the awaiting download and returns the user to the same topic page.

Exceptional flow

E1 : A non-registered user clicks the download link.

1. The system displays a message saying you are not a registered user and redirects to the register form.
2. The use case continues at position 3 of use case 1 of the main flow

Termination

The system returns the user to the topic page for further browsing.

Post condition

The system then awaits the next user interaction.

REQUIREMENT 4: USER REGISTRATION

DESCRIPTION & PRIORITY

This requirement is ranked 4. The user needs to register for the web application forum in order to create a post and/or comment on an existing post.

USE CASE

Scope

The scope of this use case is to provide users with a means of signing up via a GUI in order to create a post and/or comment on an existing post.

Description

This use case describes what a non-registered user can do so they can become a registered user on the forum.

Flow Description

Precondition

The forum is idle on the home page awaiting user interaction.

Activation

This use case starts when a non-registered user clicks on the register button.

Main flow

7. The system identifies the user interaction and displays an option to register or login. See A1.
8. The non-registered user clicks the register option
9. The system displays the register form.
10. The non-registered user fills in the required fields (name, email, password, username) and clicks the save button. See A2.
11. The system then saves the user details to the database and redirects the now registered user back to the home page of the forum. See E1.
12. The now registered user can now post on the forum and/or comment on an existing post.

Alternate flow

A1 : User is already registered

5. The user clicks the login option.

6. The system displays the login portal.
7. The user inputs their credentials and clicks login. See E2.
8. The main flow then continues from point 1.

A2: Non-registered does not register

4. The non-registered user clicks cancel
5. The system then reverts to the home page where the non-registered user can navigate site without functionality.
6. The non-registered user then clicks a function and main flow continues from point 1.

Exceptional flow

E1: Non-registered user inputs invalid data in form

4. The system throws an error asking user to input valid characters/username/password.
5. The user inputs valid characters/username/password and clicks save.
6. The use case continues at position 5 of the main flow.

E2: Registered user inputs incorrect login credentials

3. The systems checks database and returns error “invalid credentials”.
4. The registered user inputs the correct credentials and the main flow continues from point 6.

Termination

This flow is terminated when the user is successfully registered and returned to the home page.

Post condition

The system then awaits the next user interaction.

REQUIREMENT 5: CREATING FORUM POSTS

DESCRIPTION & PRIORITY

This is ranked 4. This allows registered users to create forum posts.

USE CASE

Scope

The scope of this use case is to allow registered users to create forum posts to interact with other users.

Description

This use case describes the process of how a registered user can create a forum posts.

Flow Description

Precondition

The user is logged in as a registered user and the system is awaiting interaction on the home page.

Activation

This use case starts when a registered user navigates to the Forums section.

Main flow

1. The system identifies the registered user interaction and displays all forum posts and option to create post. See E1.
2. The user clicks create post
3. The system displays the form requesting a title and a body.
4. The user fills in both fields clicks submit. See E2.
5. Systems displays a message asking if they would like to submit the post to the forum.
6. User clicks yes. See A1.
7. System redirects the registered user to the newly created forum post.

Alternate flow

A1 : The user clicks no on the submission confirmation prompt.

1. The user is returned to the still populated form and as the option to edit or cancel.
2. If the user edits they can again click submit and the main flow will continue from 4. If the user clicks cancel the main flow will continue from point 1.

Exceptional flow

E1 : A non-registered user clicks the create post option.

1. The system displays a message saying you are not a registered user and redirects to the register form.
2. The use case continues at position 3 of use case 1 of the main flow.

E2 : The user does not fill in one of the fields.

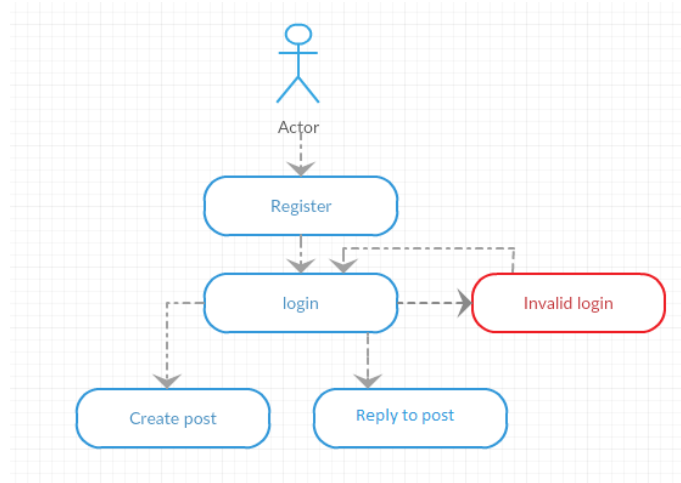
1. The system will display throw an error advising that all fields have not been filled in.
2. The user then fills in the required field and clicks submit.
3. The main flow continues from point 5 of the main flow.

Termination

The system idles on the forum page.

Post condition

The system then awaits the next user interaction.



REQUIREMENT 6: VIEWING AND COMMENTING ON FORUM POSTS

DESCRIPTION & PRIORITY

This is ranked 5. This allows registered users to view and comment on forum posts.

USE CASE

Scope

The scope of this use case is to allow registered users to view and comment on existing forums posts.

Description

This use case describes the process of how a registered user can view and comment on forum posts.

Flow Description

Precondition

The user is logged in as a registered user and the system is awaiting interaction on the home page.

Activation

This use case starts when a registered user navigates to the Forums section.

Main flow

1. The system identifies the registered user interaction and displays all forum posts.
2. The user clicks an existing post.

3. The system displays the post and all comments on the post and an empty comment box.
4. The user adds their comment in the comment box and clicks submit. See E1.
5. Systems displays a message asking if they would like to submit the comment to the post.
6. User clicks yes. See A1.
7. System then submits the comment and reloads the page with the newly added comment.

Alternate flow

A1 : The user clicks no on the comment submission confirmation prompt.

1. The user is returned to the still populated comment box and has the option to edit or cancel.
2. If the user edits they can again click submit and the main flow will continue from 7. If the user clicks cancel the main flow will continue from point 3.

Exceptional flow

E1 : A non-registered user clicks the submit comment option.

1. The system displays a message saying you are not a registered user and redirects to the register form.
2. The use case continues at position 3 of use case 1 of the main flow.

Termination

The system idles on the forum post page.

Post condition

The system then awaits the next user interaction.

NON-FUNCTIONAL REQUIREMENTS

In this section I will discuss the non-functional attributes which are required for the success and longevity of the Edyoucate.ie web application.

PERFORMANCE/RESPONSE TIME REQUIREMENT

The optimal response time for general navigation for the site (meaning page to page) is 20 milliseconds. For pages such as the topics which include material that will be pulled from the database and also the forums page which will be pulling existing posts from the database I will be aiming for a response time of no more than 40 milliseconds. The login response times will be a little longer but should be no more than 90 milliseconds, the reason for this extended response time is due to a security requirement which I will go into more detail about in the Security Requirement further down. Upload and download performance will be very much reliant on the users internet connection however taking an average download speed of 20Mbits/s and an average file size (a document) 1MB and assuming the user is on a 24Mbit/s ADSL connection the download should take no more than 1 second. The upload on the same connection should take no more than 3 seconds. To optimise these times, I will be installing Googles PageSpeed tool on my Apache server to monitor response times over time in order to identify any potential bottle necks. When making calls to the database for logins I have added a limit of 1 to the prepared statement which returns the email once found rather than looping through all email address until it gets to the end.

AVAILABILITY REQUIREMENT

The Edyoucate.ie application is accessible 24/7 from any location. To enforce this promise, the application will employ a traffic handling service on AWS called Elastic Load Balancing. This allows me to add multiple EC2 instances to the environment, the traffic is then routed to and balanced across each instance. If one of the instances fails, the traffic is automatically rerouted to the operational instances hence eliminating unscheduled downtime. I will be using this load balancer across 2 AWS availability zones, this means that even in the unlikely event that an AWS datacentre goes down, the traffic is rerouted to the running instances in the second availability zone further decreasing the risk of unscheduled downtime.

RECOVER REQUIREMENT

The structure of my application consists of an EC2 instance and an EBS volume. The instance is responsible for the computing and the EBS volume is responsible for the storage. With this in mind, backups of the EBS volume are vital. If the application does do down or is compromised in anyway a backup will allow me to restore the environment in minutes. To add more consistency to the backup process I am leveraging AWS CloudWatch events to create a snapshot (a compressed image) of the storage volume every 2 days:

Summary

ARN  arn:aws:events:eu-west-1:225803119069:rule/AutoEBSbackup

Schedule Fixed rate of 2 days

Status **Enabled**

Description

Monitoring [Show metrics for the rule](#)

Targets

Type	Resource name	Input	Role	Additional parameters
Built-in target	create-snapshot	Constant: "vol-099e5278bc119358f"	AWS_Events_Invoke_Action_On_EBS_Volume_1702401811	

As my database is stored locally to the server I have created a bash script to which connects to the database and creates a back. I then created a cronjob to run the script again every 2 days and store the backup in a backups folder outside the web directory.

```


1 #!/bin/sh
2 ADMIN_URL="https://edyoucate.ie/phpmyadmin"
3 USERNAME='root'
4 PASSWORD='Edyoucate11prod'
5 COOKIEJAR='/tmp/my_cookiejar'
6 token=$(curl -s -o - -d "pma_username=$USERNAME" -d "pma_password=$PASSWORD" -c "$COOKIEJAR" "$ADMIN_URL/" | grep -o -m 1 "token=.*" | sed "s/://'")
7 curl -o - -b "$COOKIEJAR" -d "what-sql" -d "export_type=server" -d "sql_structure_or_data=structure_and_data" -d "token" "$ADMIN_URL/export.php"
8 rm "$COOKIEJAR"

```

```

#db backup cron job
0 0 */2 * * /var/www/html/dbScript.sh > /var/www/backups/db.sql

```



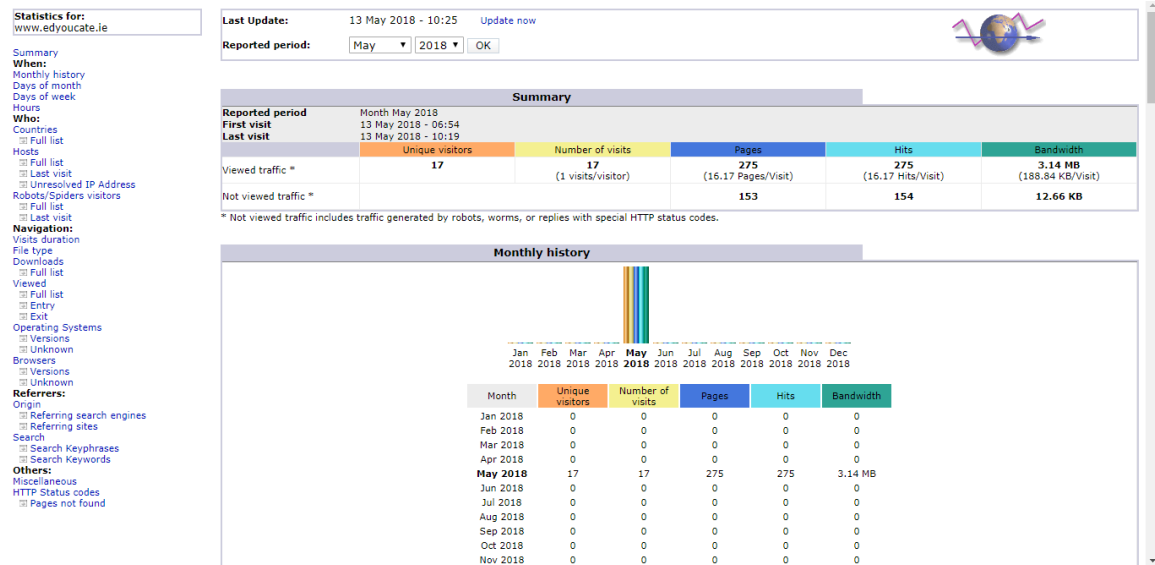
SECURITY REQUIREMENT

As the nature of this project is the uploading and download of files, user security is of utmost importance. To expand on the login response times mentioned in the Performance/Response time section, these response times will be a little longer as I have used SHA512 password hashing with Salt as PHP is my main server side language. Salt essentially adds a random string of letters to the end of the password to disguise it which are assigned to that user and then the full string is passed through the SHA512 encryption. To prevent against SQL injection, I will be calling the `mysql_real_escape_string` PHP function, which adds a backlash before potentially malicious characters. I used prepared statements throughout project. Further down the document I will outline the penetration testing carried out along with the results.

In order to prevent users from downloading files with malicious contents, all uploads are ran through VirusTotal which scans the file and runs it through 60 different antivirus engines. Once the file has been passed the check, it is uploaded to Edyoucate.ie, however if the file fails the check the file is removed from the server and the entry is removed from the database.

MAINTAINABILITY REQUIREMENT

The web application itself will be relatively maintenance free as it is in a cloud environment allowing for seamless interchanging of infrastructure. To accommodate any frontend development however there may need to be some scheduled downtime. To best identify the optimal time for this downtime I have installed AWStat on the server which monitors traffic to the site. From the information recorded through this, I can identify at what time of the week and day/night has the least amount of traffic and schedule the downtime accordingly. I can then send a mass email to all users notifying them of the scheduled time. I can view the web app statistics at any time by going to <http://edyoucate.ie/cgi-bin/awstat/awstats.pl?config=linux>



EXTENDIBILITY REQUIREMENT

To keep the current scope of the project realistic, I will primarily be focusing on the Irish market attempting to build a library of resources, however once the application has matured I could then look at creating "By Country" categories and also implementing multilingual support.

REUSABILITY REQUIREMENT

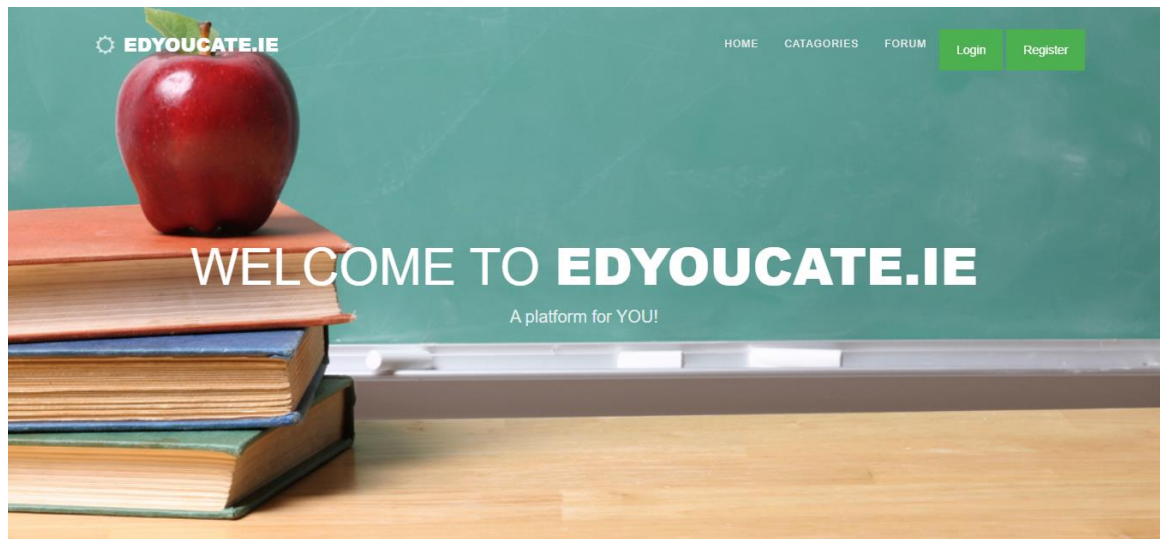
As the content of the application is user generated, code reusability will be a prominent focus. In my forum section I have used an MVC like framework which Each input will have a specified template.php file that will be populated by the user through the UI form and then stored in the database. When a user revisits the page, the include() PHP function will be called to pull all required information from the database. Using these templates allows for future changes with minimal work.

INTERFACE REQUIREMENTS

In this section I will describe how the software interfaces with the user and give a run through of how the user navigates the application. I have provided screenshots from the working application.

GUI

Edyoucate.ie is a clean, bright, minimal and inviting interface. It is extremely important that the first impressions of the interface are good as this will set the tone for the rest of the user experience. When a non-registered user first visits the site, they met by the home page which contains a main header with the title, a login or register option, a simple but straight forward menu bar, a carousel containing the main topics and a short paragraph about what we offer.



What is EdYOUcate.ie and how can it benefit you?



What is edyoucate.ie?

A content sharing platform for students!

A place for learning and exploring!

A place for questions answers and discussion!

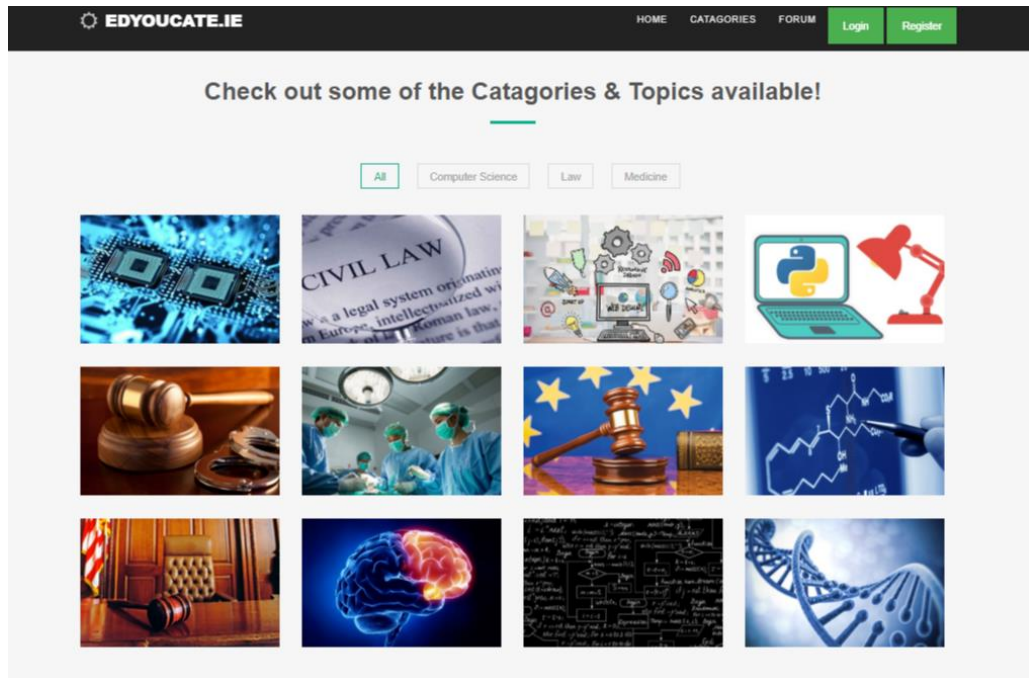
Why should I use it?

Access content relevant to your studies,

Communicate with other students via the forum,

Safely download and review materials,

and best of all, its completely FREE!!



As per the main flow a non-registered user can freely navigate the main homepage, however if they click on the “Categories” tab they will be redirected to the registration page where they will be asked to register an account using an username, email address and password:

EDYOUCATE.IE HOME CATAGORIES FORUM Login Register

Register with us

Usernames may contain only digits, upper and lowercase letters and underscores
Emails must have a valid email format
Passwords must be at least 6 characters long
Passwords must contain
At least one uppercase letter (A-Z)
At least one lowercase letter (a-z)
At least one number (0-9)
Your password and confirmation must match exactly

Username:

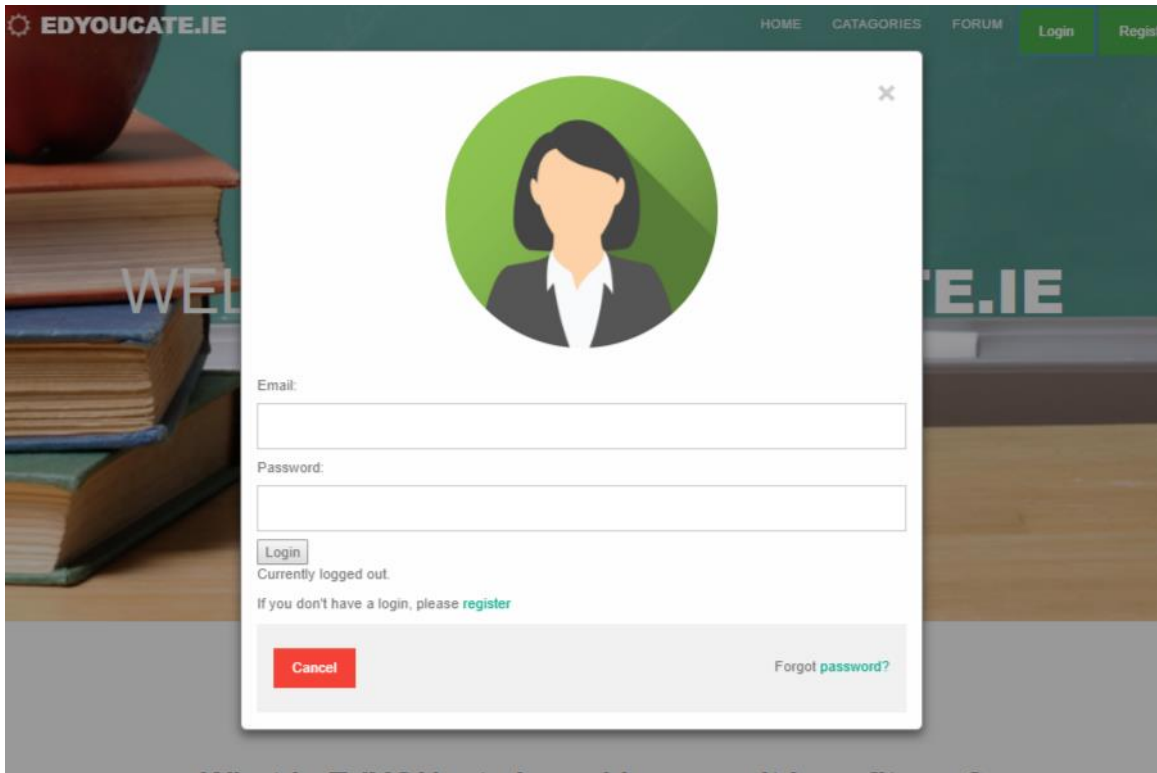
Email:

Password:

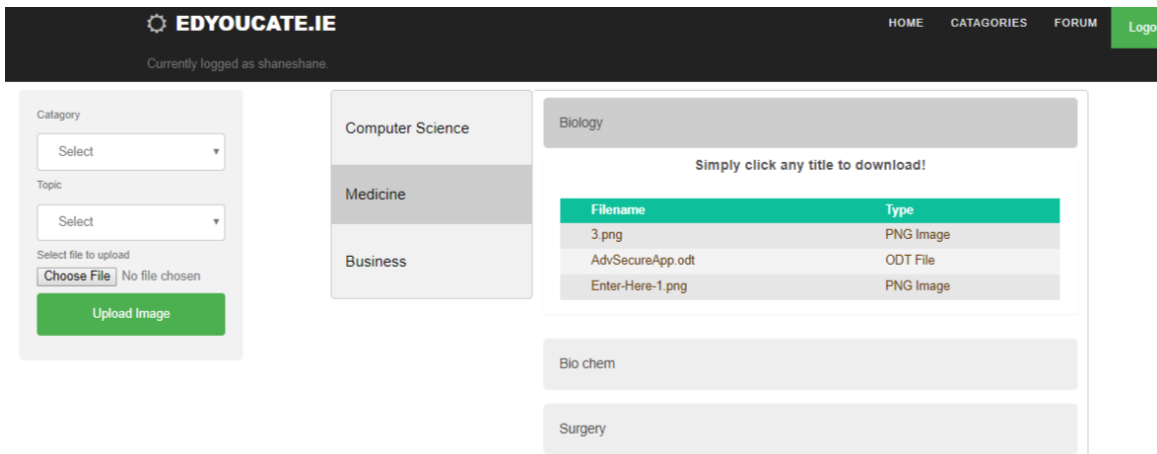
Confirm password:

[Return to the login page.](#)

Once the user has registered they will then be redirected back to the homepage where they can sign in with their new credentials using the Login button in the top right corner of the page:



When the user logs in they can then navigate to the “Categories” where they can view materials. To view the materials the user simply needs to click the desired category and then the desired topic:



In order to download a file, the user simply needs to click on the desired file. If the file is an image it will open in the browser, any other file type will begin download.

If you user would like to upload material they can do so by using the upload form on the left of the screen, selecting the desired category, the desired topic and then using the file chooser to choose a file from their local machine:

EDYOUcate.IE
Currently logged as shaneshane.

Category
Computer Science

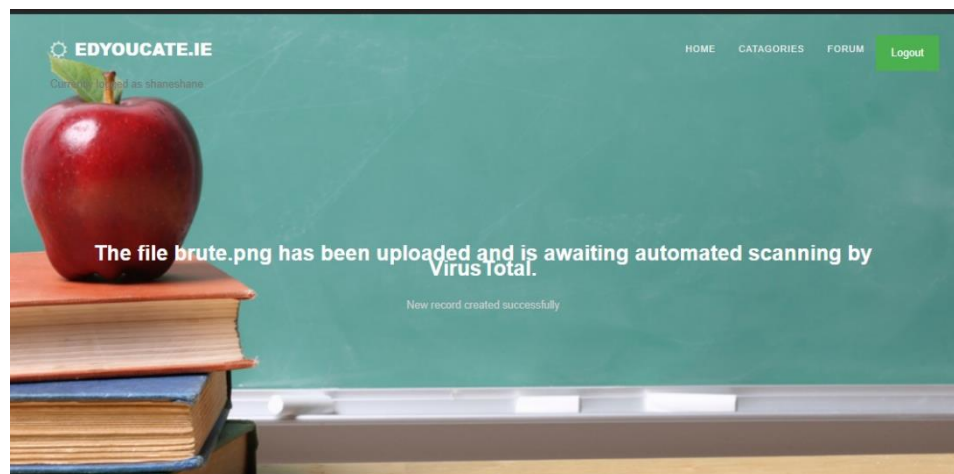
Topic
Programming

Select file to upload
Choose File brute.png

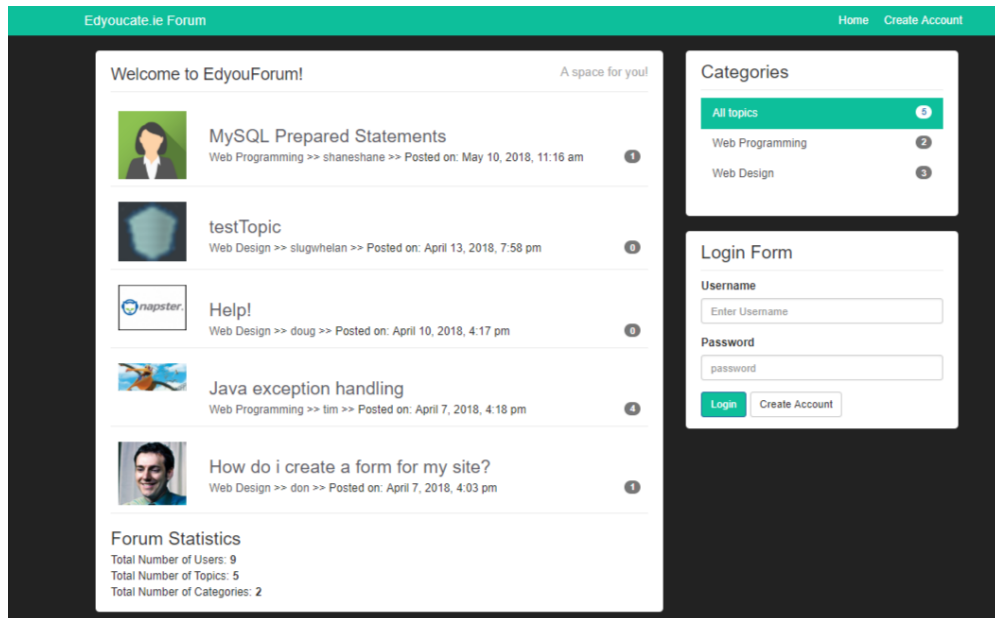
Upload Image

Computer Sci
Medicine
Business

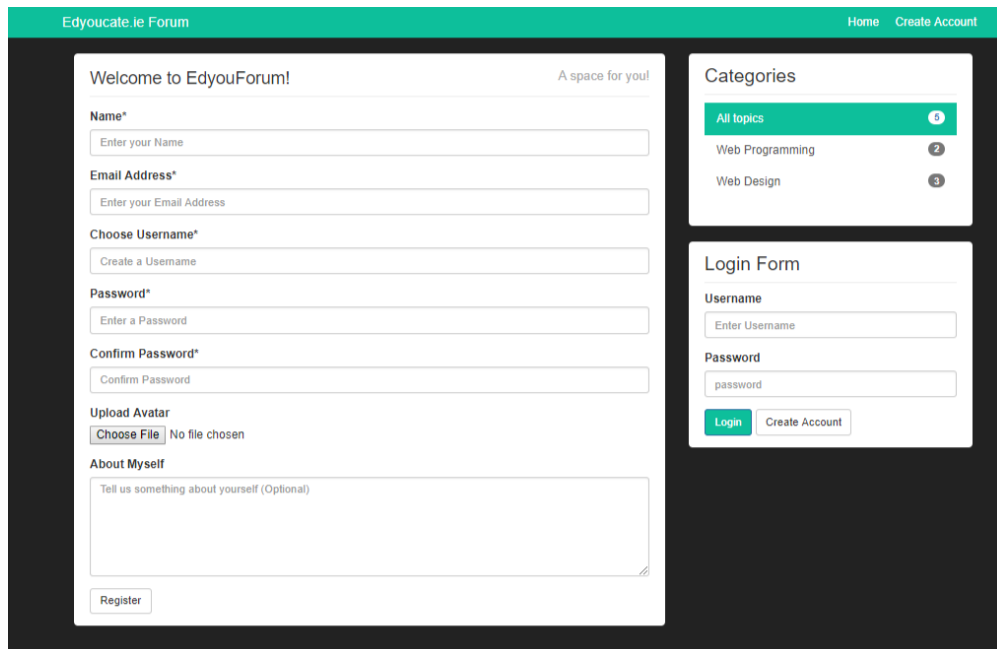
If they are happy to upload, they can click the Upload button which will submit the file to the chosen position and also queue the file for scanning by VirusTotal:



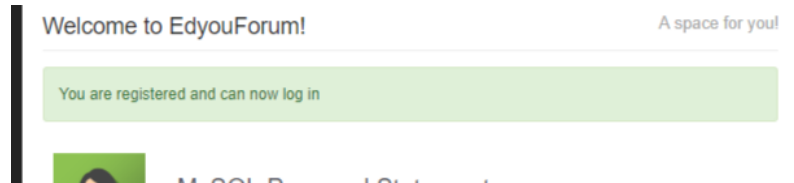
Registered and non-registered users alike can both visit the forum where they can view all forum posts.



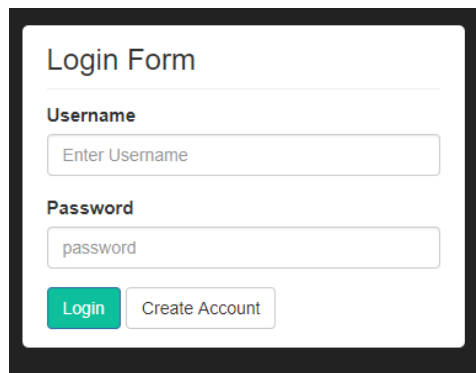
Before a user can create post or comment on a post they first need to register an account. They can do this by clicking the register button on the frontpage. The register requests the name, username, email, password, avatar and has an optional about section.



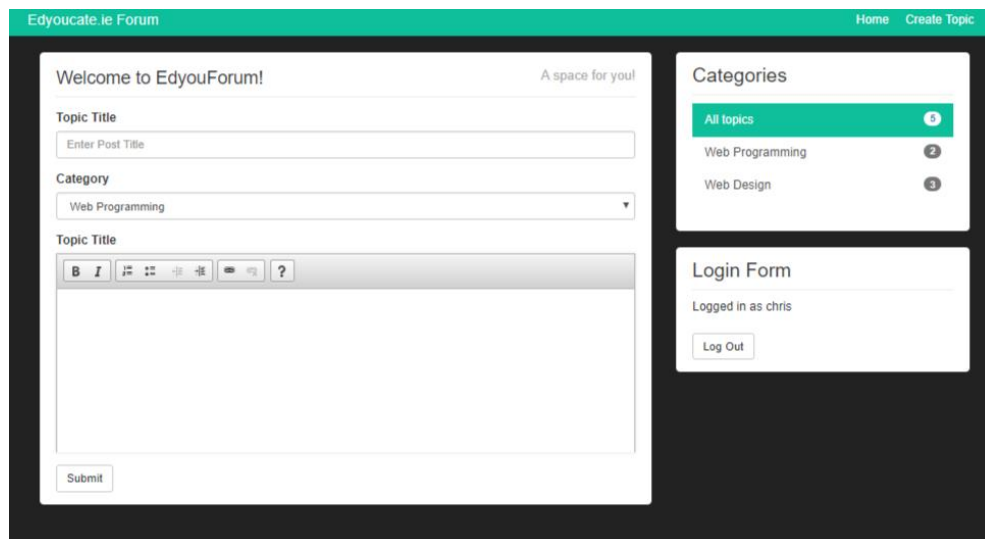
Once the user has added the required fields they can then click the register button which will redirect them back to the frontpage of the forum and display a registration successful message:



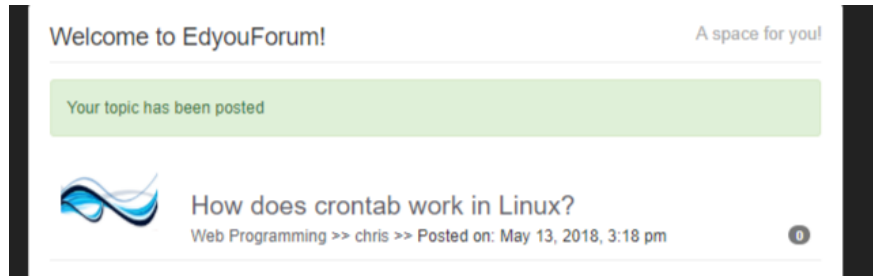
Now that the user is fully registered they can login using the login form:



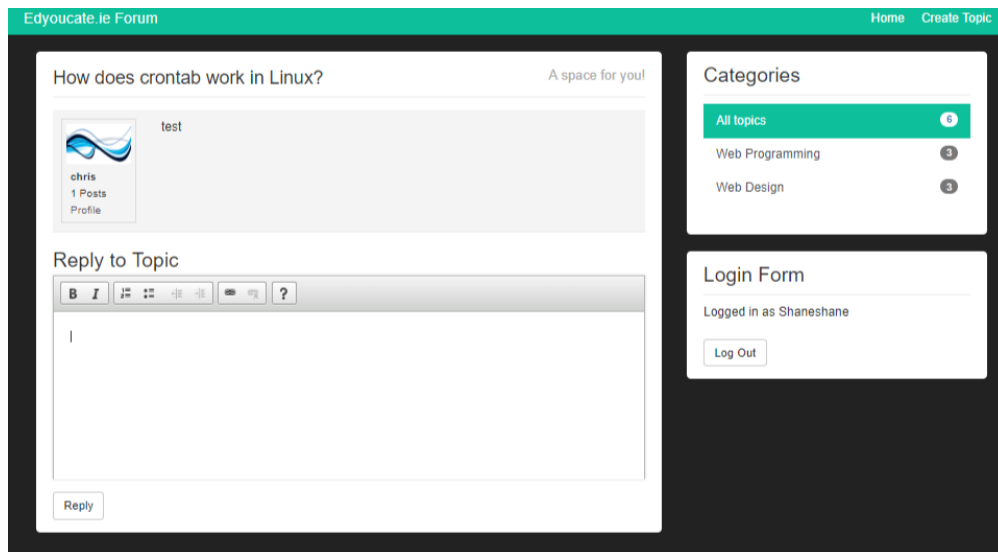
Now that the user is logged in, they can create a forum post by clicking the create button in the top right corner of the page. This will bring the user to a form where they can choose a title, choose a category and add further details of the question:



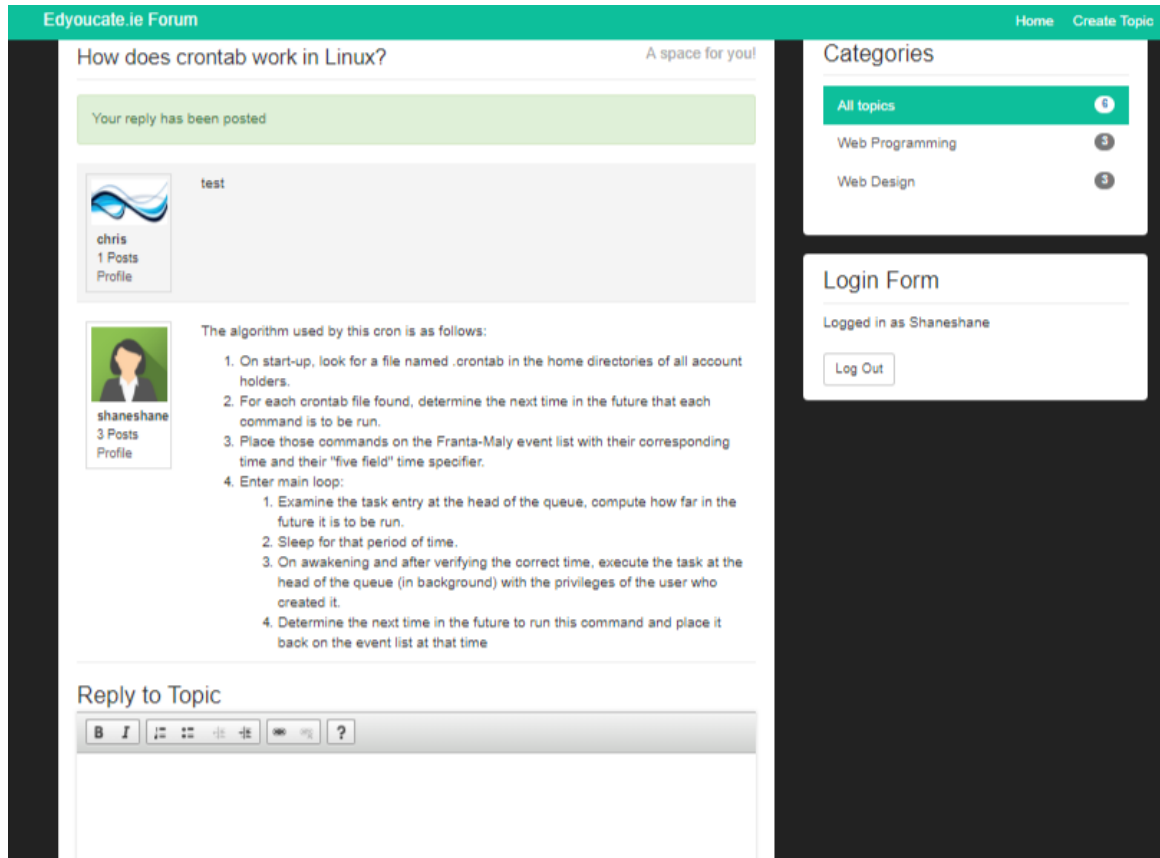
Once the user has filled in all fields, they can hit the submit button which will submit the post and redirect the user back to the forum frontpage where they will see a confirmation message and their post will be at the top of the list:



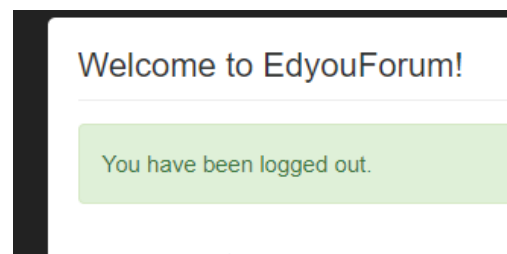
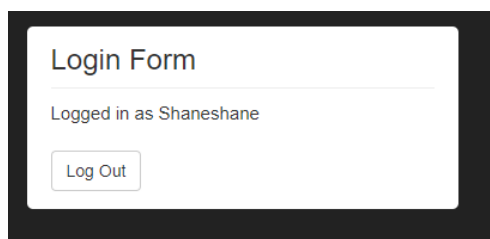
Users can also comment on other post by clicking on the desired post title. This will take the user to the post where they will find a blank text area:



Once the user fills in their reply they can click the reply button which will instantly post the reply to the page and they will also receive a reply confirmation:



When the user has finished reviewing the forums they can use the logout button to log themselves out of the application. Once successfully logged out they will also receive a logout confirmation:



IMPLEMENTATION AND FEATURE DEVELOPMENT

AWS ENVIRONMENT DEVELOPMENT

I developed my application on an ubuntu Linux t2.micro instance which I backed with an 8GB EBS volume. As the application primarily targets Irish residents, I launched the instance in the EU-West-1(Ireland) region, this cuts down on any potential latency.

Once I launched my instance I installed a LAMP stack to support the application. A LAMP stack consists of Linux, Apache, MySQL and PHP. In order to secure the instance, I configured the security groups to only allow traffic on port 22 (SSH), port 80 (HTTP) and port 443 (HTTPS). In a production environment it would not be ideal to leave port 22 open, however as I was using Cloud9 IDE to develop the application it was necessary to open this port so I could SSH into the instance.

The screenshot shows the 'Edit inbound rules' interface for an AWS Security Group. It contains a table with the following data:

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom ::/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom ::/0	e.g. SSH for Admin Desktop
Custom TCP f	TCP	443	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP f	TCP	443	Custom ::/0	e.g. SSH for Admin Desktop

Below the table is an 'Add Rule' button and a note: 'NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.' At the bottom right are 'Cancel' and 'Save' buttons.

To further secure the inbound traffic I installed an SSL certificate and configured to server to force all inbound connections onto port 443 (HTTPS). I tested my SSL certificate on SSL labs which returned an A rating.

The screenshot shows a 'Certificate' dialog box with the following information:

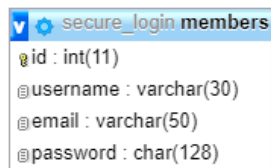
- Certificate Information**
- This certificate is intended for the following purpose(s):**
 - Ensures the identity of a remote computer
 - Proves your identity to a remote computer
 - 2.23.140.1.2.1
 - 1.3.6.1.4.1.44947.1.1.1
- * Refer to the certification authority's statement for details.
- Issued to:** www.edyoucate.ie
- Issued by:** Let's Encrypt Authority X3
- Valid from:** 28/03/2018 to 26/06/2018
- Buttons: 'Issuer Statement' and 'OK'

The screenshot shows a Qualys SSL Labs report for the domain www.edyoucate.ie. The report includes the following details:

- Overall Rating:** A
- Assessed on:** Wed, 09 May 2018 20:13:46 UTC
- Summary:**
 - Certificate: 100%
 - Protocol Support: 100%
 - Key Exchange: 100%
 - Cipher Strength: 100%
- Buttons: 'Scan Another', 'Home', 'Projects', 'Qualys.com', 'Contact'

I created my databases using phpmyadmin. I have 2 separate databases for the project, one for the web application to record uploads and one for the forums. I decided to use 2 separate databases because all data on the main uploads database would already be sanitized by the VirusTotal API and by the very nature of the forums there would be many different code segments being uploaded. Even though the forum post uploads are well sanitized using secure application development practices, I still felt it better to separate the data. I had initially planned to only use the phpmyadmin localhost database for testing and then migrate to an independent RDS instance, however due to time constraints I was unable to complete this.

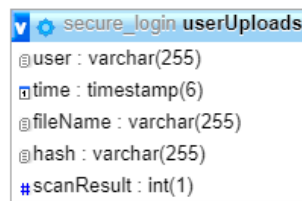
Main application database tables:



Field	Type	Attributes
id	int(11)	PK
username	varchar(30)	
email	varchar(50)	
password	char(128)	



Field	Type	Attributes
user_id	int(11)	FK
time	varchar(30)	

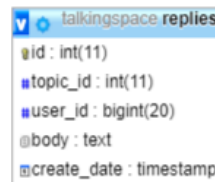


Field	Type	Attributes
user	varchar(255)	
time	timestamp(6)	
fileName	varchar(255)	
hash	varchar(255)	
scanResult	int(1)	

Forums database tables:



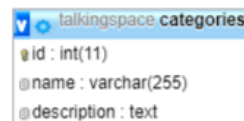
Field	Type	Attributes
id	int(11)	PK
name	varchar(100)	
email	varchar(100)	
avatar	varchar(100)	
username	varchar(20)	
password	varchar(64)	
about	text	
last_activity	datetime	
join_date	timestamp	



Field	Type	Attributes
id	int(11)	PK
topic_id	int(11)	FK
user_id	bigint(20)	FK
body	text	
create_date	timestamp	



Field	Type	Attributes
id	int(11)	PK
category_id	int(11)	FK
user_id	int(11)	FK
title	varchar(200)	
body	text	
last_activity	datetime	
create_date	timestamp	



Field	Type	Attributes
id	int(11)	PK
name	varchar(255)	
description	text	

SECURE LOGIN AND SESSION MANAGEMENT

I put a lot of time into the login system and session management as Broken Authentication and Session Management from the 2013 OWASP top 10 was broken into 2 separate points in the OWASP 2017 revision due to its severity.

I started by creating my database in PHPmyadmin named secure_login and created a user with restricted privileges to access this database. In the event that an attacker does find a hole, the use of a user with restricted privileges will deny them the ability to DROP a table or database. I then created 2 tables, "members" and "login_attempts". The members table stores the general user data such id, name, email and encrypted password (I will develop further on the latter later). The login_attempts table stores all user login attempts for the last 2 hours.



Field	Type
id	int(11)
username	varchar(30)
email	varchar(50)
password	char(128)



Field	Type
user_id	int(11)
time	varchar(30)

On the registration page I used FILTER_SANITIZE to sanitize all user input on each input field. When the users sets a password it is hashed using the SHA512 hashing algorithm and then stored in the database. When the user logs in using the password, their input is hashed and compared with the stored password hash, if the hashes match then the user is logged in. This is an extremely important process because even if the database was compromised the attacker could not utilize the credentials to access the user accounts. I am utilizing prepared statements throughout the web application, for example when a user is logging in I am using a prepared statement with set parameters and limiting the selection to 1 email. This prevents potential attacker from inputting their own queries and it prevents them from manipulating the given query to select more than 1 email address.

```
$prep_stmt = "SELECT id FROM members WHERE email = ? LIMIT 1";
$stmt = $mysqli->prepare($prep_stmt);

// check existing email
if ($stmt) {
    $stmt->bind_param('s', $email);
    $stmt->execute();
    $stmt->store_result();
}
```

To promote say password use have set conditions for the chosen password, it must be a least 6 characters long, it must contain at least one capital letter, at least one lowercase letter and at least one number.

Once my database structure was complete I created a PHP function called `sec_session_start()` rather than using the insecure default `session_start()`. This forces the sessions to use only cookies which stops potential attackers from gaining access to the session id cookie through an XSS attack. This function also calls a secondary function called `session_regenerate_id()` which forces a session id regeneration with every page reload. This prevents attackers from hijacking the session id.

```
3
4 function sec_session_start() {
5     $session_name = 'sec_session_id'; // Set a custom session name
6     $secure = SECURE; //SECURE
7     // This stops JavaScript being able to access the session id.
8     $httponly = true;
9     // Forces sessions to only use cookies.
10    if (ini_set('session.use_only_cookies', 1) === TRUE) {
11        header("Location: ../error.php?err=Could not initiate a safe session (ini_set)");
12        exit();
13    }
14    // Gets current cookies params.
15    $cookieParams = session_get_cookie_params();
16    session_set_cookie_params($cookieParams["lifetime"], $cookieParams["path"], $cookieParams["domain"], $secure, $httponly);
17    // Sets the session name to the one set above.
18    session_name($session_name);
19    session_start(); // Start the PHP session
20    session_regenerate_id(); // regenerated the session, delete the old one.
21 }
22
```

To eliminate the risk of a brute force attack on the login, I have implemented a timeout function which queries the `login_attempts` table. If the user has 5 failed attempts in the past 2 hours, the account will be locked out for 2 hours from the first login attempt.

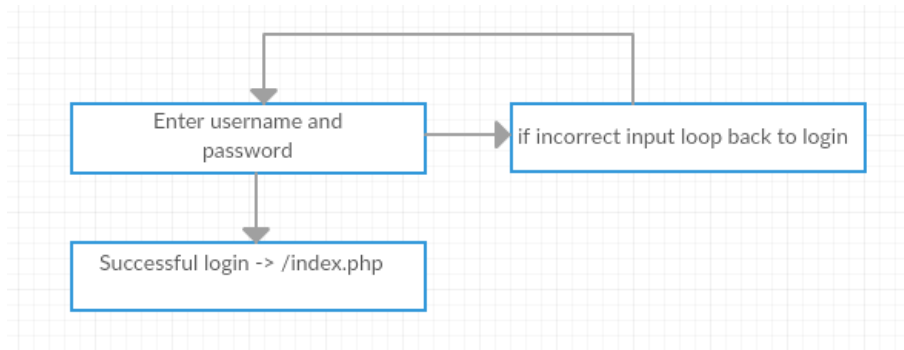
```
82 function checkbrute($user_id, $mysqli) {
83     // Get timestamp of current time
84     $now = time();
85
86     // All login attempts are counted from the past 2 hours.
87     $valid_attempts = $now - (2 * 60 * 60);
88
89     if ($stmt = $mysqli->prepare("SELECT time
90                                 FROM login_attempts
91                                 WHERE user_id = ?
92                                 AND time > '$valid_attempts'")) {
93         $stmt->bind_param('i', $user_id);
94
95         // Execute the prepared query.
96         $stmt->execute();
97         $stmt->store_result();
98
99         // If there have been more than 5 failed logins
100        if ($stmt->num_rows > 5) {
101            return true;
102        } else {
103            return false;
104        }
105    }
106 }
107
```

I am also using a function to sanitize the URL to prevent against Reflective XSS. This function scrubs the URL of nefarious characters and calls `htmlspecialchars()` to prevent against PHP_SELF exploits.

UNIT TEST

I tested the basic functionality of the login system and tested the brute force function for account lockout.

Expected login behaviour:



Observed login behaviour:

First, I inputted the correct username and password and clicked login, this passed me back to the index page and displayed my username as logged in. Then I input incorrect password which failed to log me in as expected. I also tested the lockout feature by inputting the incorrect password 5 times and then inputting the correct password which again resulted in a login error signifying the account is now locked out. I then waited 2 hours and attempted the login again with the correct password which successfully login me in.

MATERIAL UPLOAD AND DOWNLOAD

The Categories section allows registered users to upload and download materials to specified categories and topics within those categories. The section is comprised of a tab structure which contains the category and an accordion which contains the topics for the corresponding category both of which are built using JavaScript. The file upload has 2 conditions, it must not already exist on the server and it must be small than 1MB.

To mitigate against the risk of users uploading malicious files (intentionally or non-intentionally) I am leveraging the VirusTotal API which scans files for malicious content and runs the result against 60 different anti-virus databases.

I first created a VirusTotal account to get my API key. I then used the used the class and functions provided on the VirusTotal website to lay the basis for the API calls I would be making. I created a class called `virustotalClass.php` and a functions file called `virustotalFunctions.php`. The Class is first initialized and then uses its `checkFile()` method to send the file for scanning and then uses its `getResponse()` to grab the output. I then used a function `VirusTotalScan()` to take the output and put it in an array. I then used another function `VTCheckHashOfFileSubmitted()` to populate the array with the details of the file that is being scanned including the file hash and a link to the VirusTotal scan.

```
Array
(
    [scan_id] => 2281d76df18b93cb9a8d973d0e5af7b5349ba84d8fb7531fead7a3871e9214b2-1526134906
    [sha1] => eb4b8570828a9d10597d886caed61e0feae92cfa
    [resource] => 2281d76df18b93cb9a8d973d0e5af7b5349ba84d8fb7531fead7a3871e9214b2
    [response_code] => 1
    [sha256] => 2281d76df18b93cb9a8d973d0e5af7b5349ba84d8fb7531fead7a3871e9214b2
    [permalink] => https://www.virustotal.com/file/2281d76df18b93cb9a8d973d0e5af7b5349ba84d8fb7531fead7a3871e9214b2/analysis/1526134906/
    [md5] => e722ddda4771de33564e47337e3fa9c
    [verbose_msg] => Scan request successfully queued, come back later for the report
)
```

When the file is uploaded it also creates an entry in the `userUploads` table in the database. The entry consists of the user who uploaded, a timestamp, the file path (including the name) and the hash of the file. I also have a column called `scanResult` which has an initial default value of 0. When uploaded, the file is not scanned instantly, it enters a queue system and can take a number of minutes to return a scan result. To get the result I created a cronjob to run the `virustotalCronScan.php` script, this pulls the result and checks it. If the file is clean the `scanResult` value is changed to 1 and the file stays on the webserver. If the file is malicious, the `scanResult` value stays as 0, the entry is removed from the database and the file is removed from the webserver using `php unlink`.

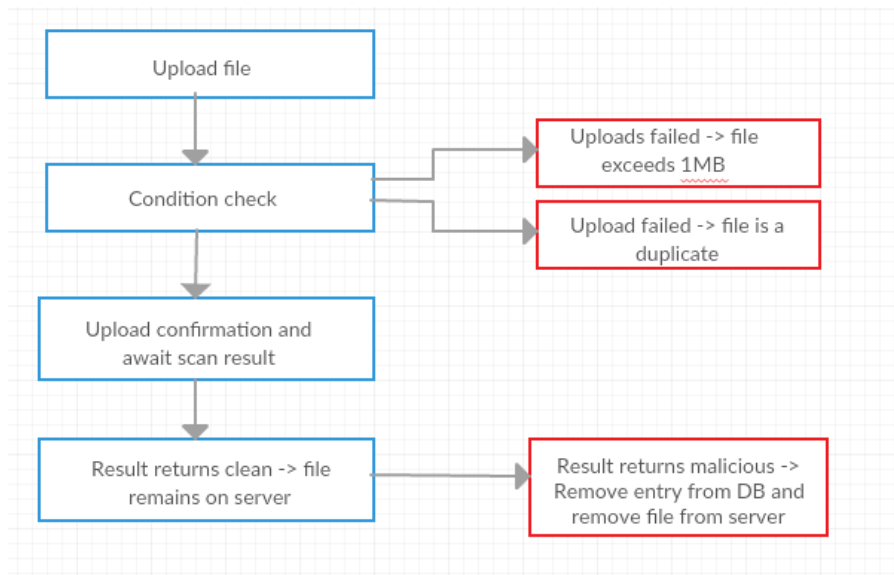
The user chooses the category and topic and upload file chooser using the form on the left of the page. Once the file is uploaded, it is displayed under the chosen heading. I achieved this by using a directory viewer to pull all files in the given directory and display their types.

UNIT TEST

I conducted 4 tests on the upload feature:

- Upload clean file
- Upload malicious file
- Upload duplicate file
- Upload file that is too large

Expected behaviour:



Observed behavior:

I first uploaded a clean file and received a confirmation that the file was submitted and awaiting the VirusTotal scan. After the scan was complete I checked the scanResult in the database which had correctly changed to 1 and observed that file remained on the server. I then submitted a malicious reverse shell php script and received confirmation that the file had been submitted and was awaiting the VirusTotal scan. After the scan was complete I checked the scanResult in the database which had correctly remained as 0. Once the cronjob ran the virustotalCronScan.php script, the database entry for the file was removed and the file was correctly removed from the server. Next, I submitted a file that was 1.2MB and received an error advising the file was too big and thus had not been submitted as expected. I then submitted a duplicate file and received an error advising that the file was a duplicate and had not been submitted as expected.

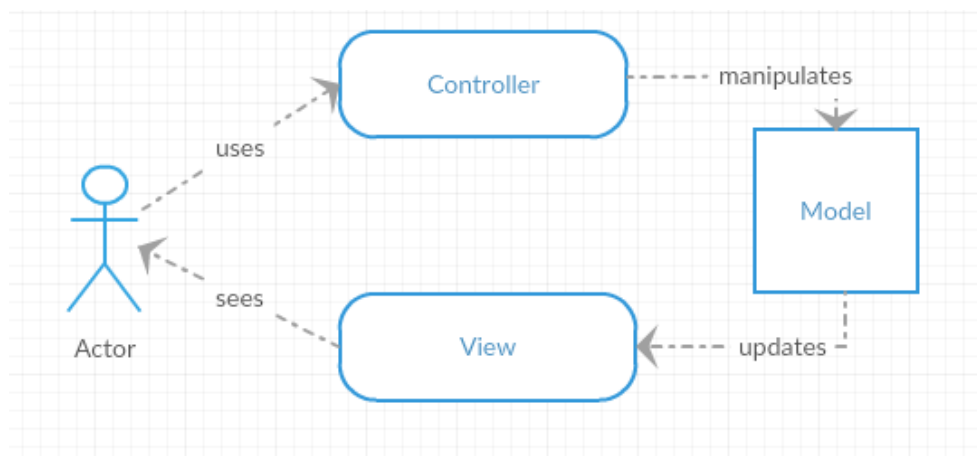
FORUMS

To develop the forums, I used an MVC (Model-View controller) like framework and PDO (PHP Data Objects) to connect to the database. PDO uses secure practices which works on the premise of creating prepared statements with placeholders. So first we create the query and add placeholders for our variables, we then prepare the statement and define the variables. This prevents potential attackers from adding their own query to perform an SQL injection.

PDO is now the preferred method of developing PHP with SQL due its security and portability. PDO queries are structured so they can be incorporated with multiple other database types.

```
public function __construct() {
    // Set DSN
    $dsn = 'mysql:host=' . $this->host . ';dbname=' . $this->dbname;
    // Set options
    $options = array (
        PDO::ATTR_PERSISTENT => true,
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    );
    // Create a new PDO instance
    try {
        $this->dbh = new PDO ($dsn, $this->user, $this->pass, $options);
    } // Catch any errors
    catch ( PDOException $e ) {
        $this->error = $e->getMessage();
    }
}
```

MVC or Model-View-Controller strongly supports code reusability. The user uses the controllers which are the main functionalities of the application to the manipulate the models which are the database queries and then outputs them in different views which are then templates for the data to sit which is what the user sees.



The Forum2 root directory contains the 5 Controllers which are the main functions of the forum:

- index.php is the main page/front page controller
- register.php is the controller for user registration
- create.php is the controller for creating posts once a user has successfully registered
- topics.php is the controller for displaying all posts on the page
- topic.php is the controller for displaying replies and creating replies on a forum post

The 5 Models are contained in forum2/libraries directory. These are used by the controller to query the database and pass the data through the views. These models are as follows:

- database.php is the model that connects to the database
- template.php is the model that is used for defining and managing the views
- topic.php is the model that is used to define the forum topics page
- user.php is the model that defines the user which handles the sessions and registration
- validator.php is that model that is used to define the validation for all user input fields.

The 5 views (or templates) which are contained in the forum2/templates folder takes the output from the model that was manipulated by the controller and displays it in a meaningful way. The 5 views are as follows:

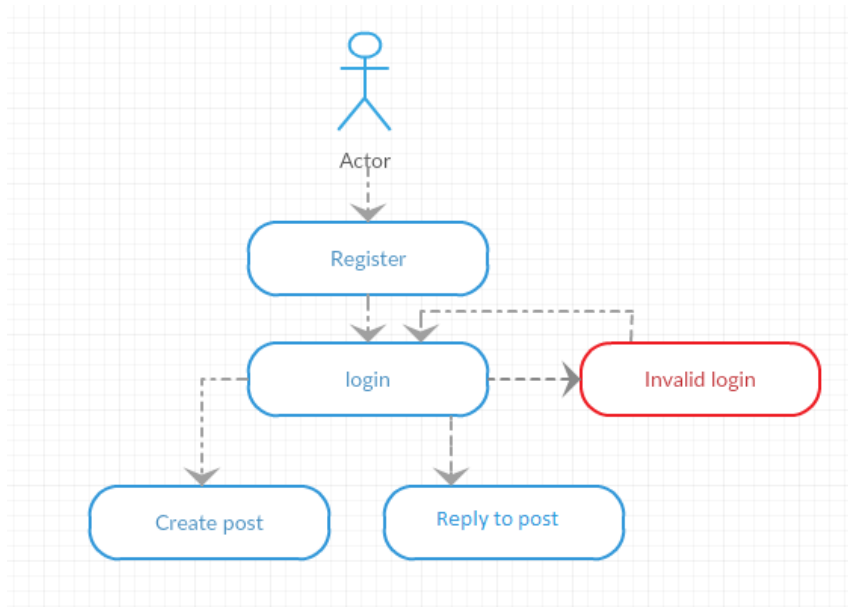
- frontpage.php is the view that contains all forum posts and takes data provided by the index.php controller.
- Register.php is the view which handles the user registration.
- create.php is the view that is used to create a forum post. This takes data provided by the create.php controller.
- topic.php is the view which handles the topic.php controller data to display replies on a forum post.
- topics.php is the view that takes the data from the topics.php controller and displays all forum posts for that given topic.

UNIT TEST

I conducted 2 unit tests on the forums:

- Registration
- Login
- Create post
- Reply to post

Expected behaviour:



Observed behaviour:

I first tested the registration form by inputting all details correctly. This worked as expected registering the account. I then came back and attempted the registration again this time leaving out required fields which resulted in an error as expected.

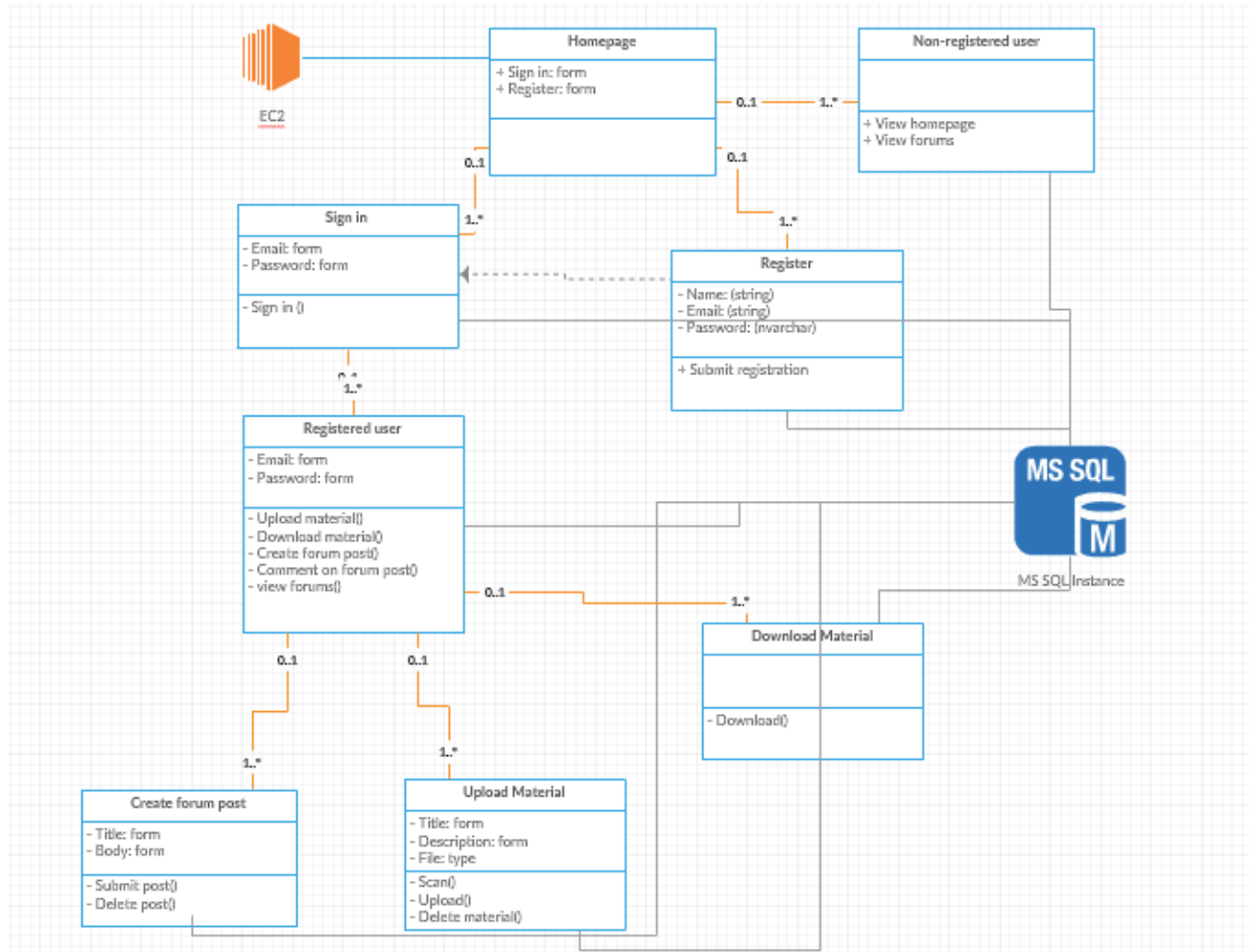
Once registered I then attempted login with the correct details, the app logged me in as expected. I then attempted the login with the incorrect details which resulted in an error as expected.

I then followed the expected behaviour flow and created a created a forum post inputting all required information correctly. This action performed as expected. I then came back and left out the title and body which resulted in the app throwing an error as expected.

Lastly, I tested the reply to post feature by adding a reply and hitting the reply button, this worked as expected. I then attempted to reply without adding a body to the reply, this through an error as expected.

SYSTEM ARCHITECTURE CLASS DIAGRAM

Below is the class diagram which includes all of the above outlined:

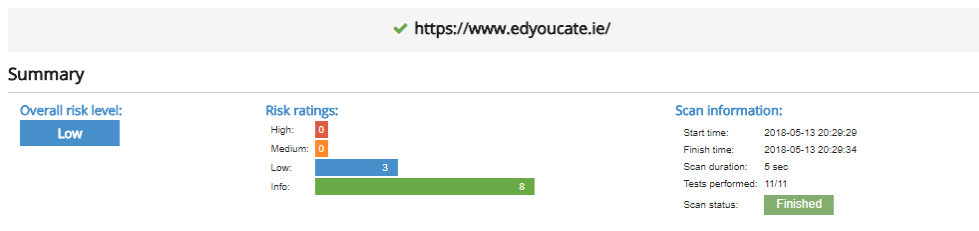


PENETRATION TESTING

I conducted 3 separate penetration testing phases on the web application.

Phase 1:

The first was a light automated scan using pentest-tools.com which returned no medium or high risks, however it did return 3 low risks, 2 of which were false positives and the other being a positive.



The positive was auto complete was not switched off.

🚩 Password auto-complete is enabled

```
<input id="password" name="password" type="password"/>
```

Details

Risk description:
When password auto-complete is enabled, the browser will remember the password entered into the login form, such that it will automatically fill it next time the user tries to login.
However, if an attacker gains physical access to the victim's computer, he can retrieve the saved password from the browser's memory and use it to gain access to the victim's account in the application.
Furthermore, if the application is also vulnerable to Cross-Site Scripting, the attacker could steal the saved password remotely.

Recommendation:
We recommend you to disable the password auto-complete feature on the login forms by setting the attribute `autocomplete="off"` on all password fields.

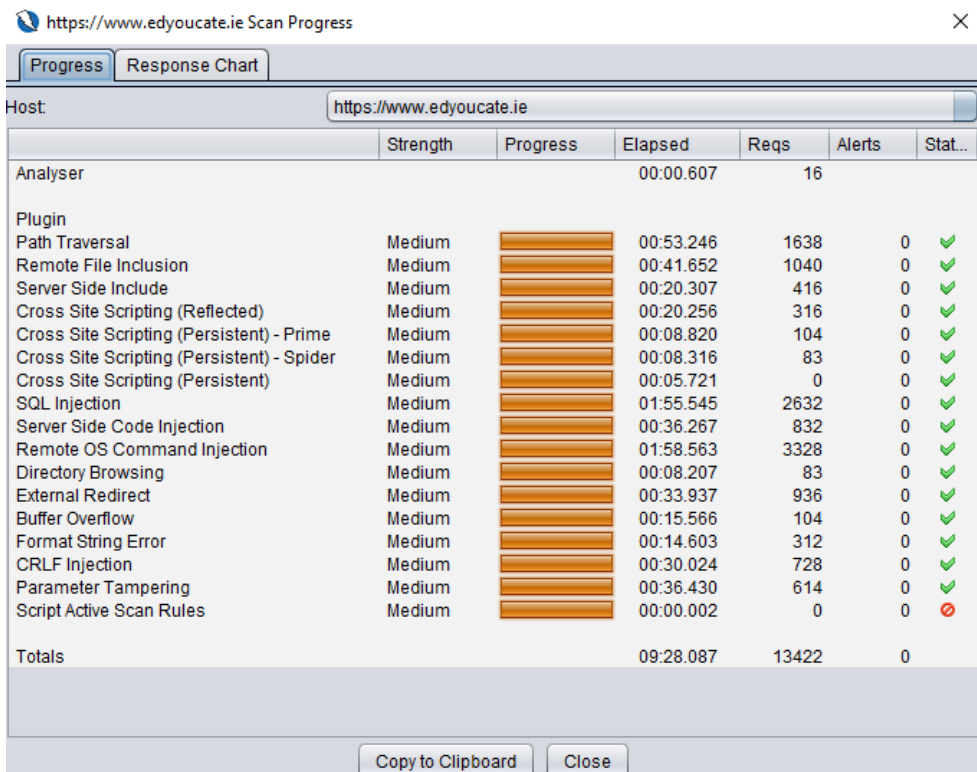
More information about this issue:
[https://www.owasp.org/index.php/Testing_for_Vulnerable_Remember_Password_\(OTG-AUTHN-005\)](https://www.owasp.org/index.php/Testing_for_Vulnerable_Remember_Password_(OTG-AUTHN-005)).

I fixed this issue by going back to my forms input and adding the "autocomplete=off" attribute to the input tags:

```
Email: <input type="text" name="email" autocomplete="off"/>  
Password: <input type="password" name="password" id="password" autocomplete="off"/>
```

Phase 2:

For the second stage of penetration testing I used another automated tool called OWASP Zed Attack Proxy Project (or ZAP). ZAP is a much more intense scan which tests the application against the OWASP top 10 vulnerabilities. The results of the scan came back clear on all counts:



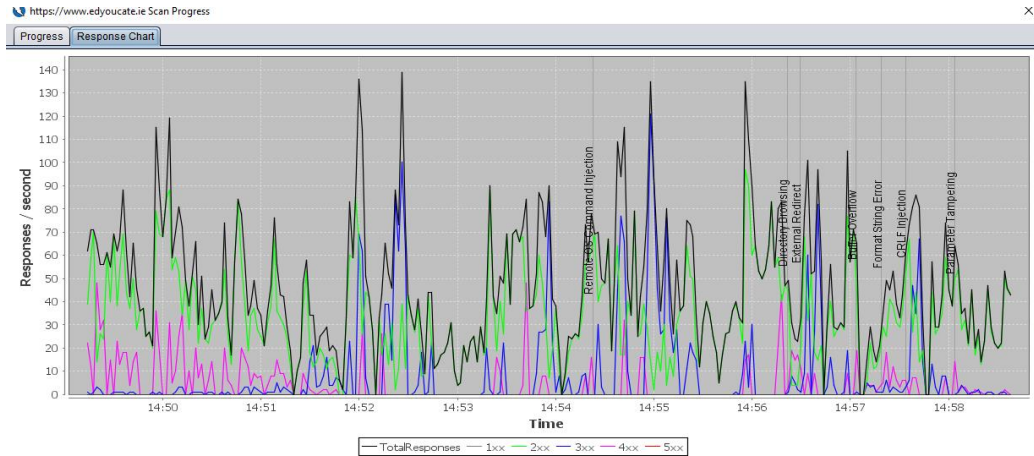
https://www.edyoucate.ie Scan Progress

Progress Response Chart

Host: https://www.edyoucate.ie

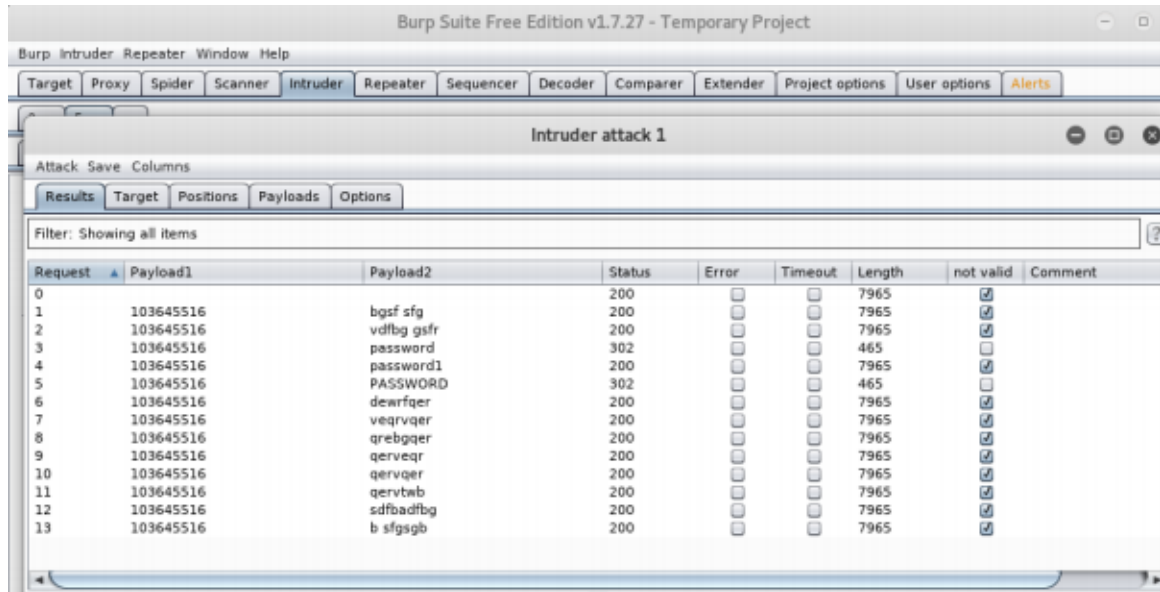
	Strength	Progress	Elapsed	Reqs	Alerts	Stat...
Analyser			00:00.607	16		
Plugin						
Path Traversal	Medium	<div style="width: 100%;"></div>	00:53.246	1638	0	✓
Remote File Inclusion	Medium	<div style="width: 100%;"></div>	00:41.652	1040	0	✓
Server Side Include	Medium	<div style="width: 100%;"></div>	00:20.307	416	0	✓
Cross Site Scripting (Reflected)	Medium	<div style="width: 100%;"></div>	00:20.256	316	0	✓
Cross Site Scripting (Persistent) - Prime	Medium	<div style="width: 100%;"></div>	00:08.820	104	0	✓
Cross Site Scripting (Persistent) - Spider	Medium	<div style="width: 100%;"></div>	00:08.316	83	0	✓
Cross Site Scripting (Persistent)	Medium	<div style="width: 100%;"></div>	00:05.721	0	0	✓
SQL Injection	Medium	<div style="width: 100%;"></div>	01:55.545	2632	0	✓
Server Side Code Injection	Medium	<div style="width: 100%;"></div>	00:36.267	832	0	✓
Remote OS Command Injection	Medium	<div style="width: 100%;"></div>	01:58.563	3328	0	✓
Directory Browsing	Medium	<div style="width: 100%;"></div>	00:08.207	83	0	✓
External Redirect	Medium	<div style="width: 100%;"></div>	00:33.937	936	0	✓
Buffer Overflow	Medium	<div style="width: 100%;"></div>	00:15.566	104	0	✓
Format String Error	Medium	<div style="width: 100%;"></div>	00:14.603	312	0	✓
CRLF Injection	Medium	<div style="width: 100%;"></div>	00:30.024	728	0	✓
Parameter Tampering	Medium	<div style="width: 100%;"></div>	00:36.430	614	0	✓
Script Active Scan Rules	Medium	<div style="width: 100%;"></div>	00:00.002	0	0	✗
Totals			09:28.087	13422	0	

Copy to Clipboard Close



Phase 3:

The third phase of the testing was carried out on the login system as the unit test brought to light a brute force vulnerability. I used the Intruder tool of the Burp suite to confirm that there was indeed a major flaw:



To fix this issue I implemented a lockout feature which would lock the account if there was more than 5 failed attempts in the past 2 hours:

```
82 function checkbrute($user_id, $mysqli) {
83     // Get timestamp of current time
84     $now = time();
85
86     // All login attempts are counted from the past 2 hours.
87     $valid_attempts = $now - (2 * 60 * 60);
88
89     if ($stmt = $mysqli->prepare("SELECT time
90                                 FROM login_attempts
91                                 WHERE user_id = ?
92                                 AND time > '$valid_attempts'")) {
93         $stmt->bind_param('i', $user_id);
94
95         // Execute the prepared query.
96         $stmt->execute();
97         $stmt->store_result();
98
99         // If there have been more than 5 failed logins
100        if ($stmt->num_rows > 5) {
101            return true;
102        } else {
103            return false;
104        }
105    }
106 }
107 }
```

FURTHER DEVELOPMENT OR RESEARCH

Given the AWS cloud environment which Edyoucate.ie will be developed in, backend infrastructure can easily be changed with little to no interruption to service. This makes it easier to keep the technology current and up to date. Also, given the elastic nature of EC2 and ELB coupled with the traffic monitoring tools that will be employed I can easily scale the application as traffic numbers grow.

Going forward it will be important to maintain the scope of the project by solidifying the current functional requirements, however there is always room for adding new features in the future such as;

- Private meeting rooms (or group chats) with screen share capabilities,
- Cloud storage for materials,
- Multilingual support.

With the current technologies being utilized and the aforementioned future feature implementation possibilities, Edyoucate.ie will be an application that will meet user needs and scale seamlessly with growth promising a prosperous future.

APPENDIX

PROJECT PROPOSAL

EDYOUNCATE.IE

Shane O'Mahony, 10359651, x10359651@student.ncirl.ie

BSc (Hons) in Computing

Specialisation: Cyber Security

Date : September 18th 2017

TABLE OF CONTENTS

Project Proposal	1
Table of Contents	2
Introduction and Background.....	3
Objectives	4
Technical Approach	5
Special Resources Required	6
Project Plan	7
Technical details.....	10
Evaluation	11

INTRODUCTION AND BACKGROUND

The inspiration for this concept came from my own experience of being in 3rd level education for 7 years. I found the material we are provided with in college to be extremely beneficial, however on further self-study and research of a given topic, I found it hard to find related material on a similar academic level to give me some perspective. This lead me to reaching out to friends and family in similar courses to find additional material from their courses which I could use to enhance my own understanding of a given topic. I found the combination of these different

materials and approaches to be very helpful and thought if I could widen the pool of contributors it would be mutually beneficial for all involved.

After conducting some research in the area, I found nothing to specifically facilitate this kind of content sharing apart from the colleges own in house content sharing systems such as Moodle or Blackboard which are only accessible by students of that particular college and only allowing students to access their own courses.

Having a keen interest in the open source community I felt I could develop an application that would be available to all who are interested and all who care to contribute that could become an extremely useful tool in every students arsenal.

I also feel that an application like this will not just benefit current students but also prospective college students who are unsure of what course they would like to start. Again, taking from my own experiences; when I completed my leaving certificate in 2010 I was very unsure of what path I wanted to take and what course I wanted to commit 4 years of my life to. If I had access to an application such as this, I feel I could have made a much more educated decision and also taken away some of the stress and fear of the unknown involved in the transition from 2nd to 3rd education.

OBJECTIVES

Given this problem I have decided to create a web application for students around the country to share their course materials. First and foremost, the application should have a signup form where the user can input their username and create a password in order to create an account. On this form the user should be asked to choose from a number of different preferred areas, in order to better define and propose their content. Once the user has created an account, it should have a secure login once they return. The application will offer a clean interface which will be divided into categories and courses which will be easily identifiable and accessible. From here, the user should be able to choose which year of the course they want to view. It should provide a section for students to upload their course materials such as slides, videos, past exam papers, past projects etc. All materials should be easily downloadable or viewable from this point. The application should scan all uploads for malicious software to protect users when downloads materials to their local machines. It should also scan all picture uploads to prevent explicit pictures.

Upon first use, the application should suggest content based on the preferences chosen I the signup form. Over time, it should suggest new material which the user has not yet viewed that

they may find useful based on their past activity and courses and topics browsed. It should also suggest content based on the materials view by others with the same interests. This will help users discover more topics and materials they may not have previously considered.

I will also include a global chat for quick questions and answers and for general interaction between users. To moderate this, I will develop an auto-moderator bot to scrape the chat to delete any key words I define and also give the user 3 chances before being banned from the chat. This will insure a secure and friendly environment for users to chat.

Like the above I would also like to incorporate a forum. This will give the users more space to ask questions and have open discussions about different topics. To moderate this I will use a similar bot to scrape the forum posts for any malicious or offensive words or phrases again securing a friendly and thought provoking environment.

I will primarily target the Irish market of over 17,000 students and countless prospective students, but can see no reason why it could not be pushed out on a global scale.

TECHNICAL APPROACH

First and foremost, I will start by clearly defining the user requirements, requirements specifications, functional requirements and non-functional requirements in the requirements specification document. This will grant a clear path and allow me to meet the goals and deadlines laid out in the Gantt chart. Ideally, I want to use as many open source resources as possible, so I will start by creating a LAMP stack in AWS. To do this I will need to carry out extensive research on what size EC2 instance and what size MySQL database instance will be most optimal for the project at hand. Once the LAMP stack has been set up, I will then begin working on logins and sessions and attempt to implement 2factor authentication using a PHP library and also a CAPTCHA on the sign-up form. At this point I will carry out some penetration testing on the logins using the Burp suit to validate the security of the application. Once the session management is under control and penetration testing has complete, I will begin creating and defining the structure of the application by developing the PHP page templates for each section such as course type, content type etc. Within each section I will add an upload, so users can upload their materials, I will also be utilizing the TotalVirus and Rekognition APIs to prevent malicious uploads.

Once the above is complete, I will have a comprehensive prototype to demonstrate at the Midpoint presentation.

From here I will begin the implementation of the machine learning algorithm. I will be using the Linear regression model to suggest content. This model will be implemented in a number of stages.






































- Construct algorithm
- Data injection and validation Stage 1 – This is the baseline for the machine learning model. The results of this first iteration will be quite far out and inconsistent at roughly 5% - 10%.
- Data injection and validation Stage 2 – This is a second iteration with a different dataset. This iteration will be closer to the desired accuracy at 60% - 70%.
- Data injection and Testing (final stage) – This is the final stage of injection. The results of this final phase will be manually altered and fed back in to improve the accuracy hit rate. This should lead to accuracy of over 90%.

I will then create the live global chat using PHP, jQuery and MySQL. The MySQL will not be used to store/record chats but to keep track of active users and also to keep track of the chatrooms. I will also be using PHP and MySQL to create the user forums.

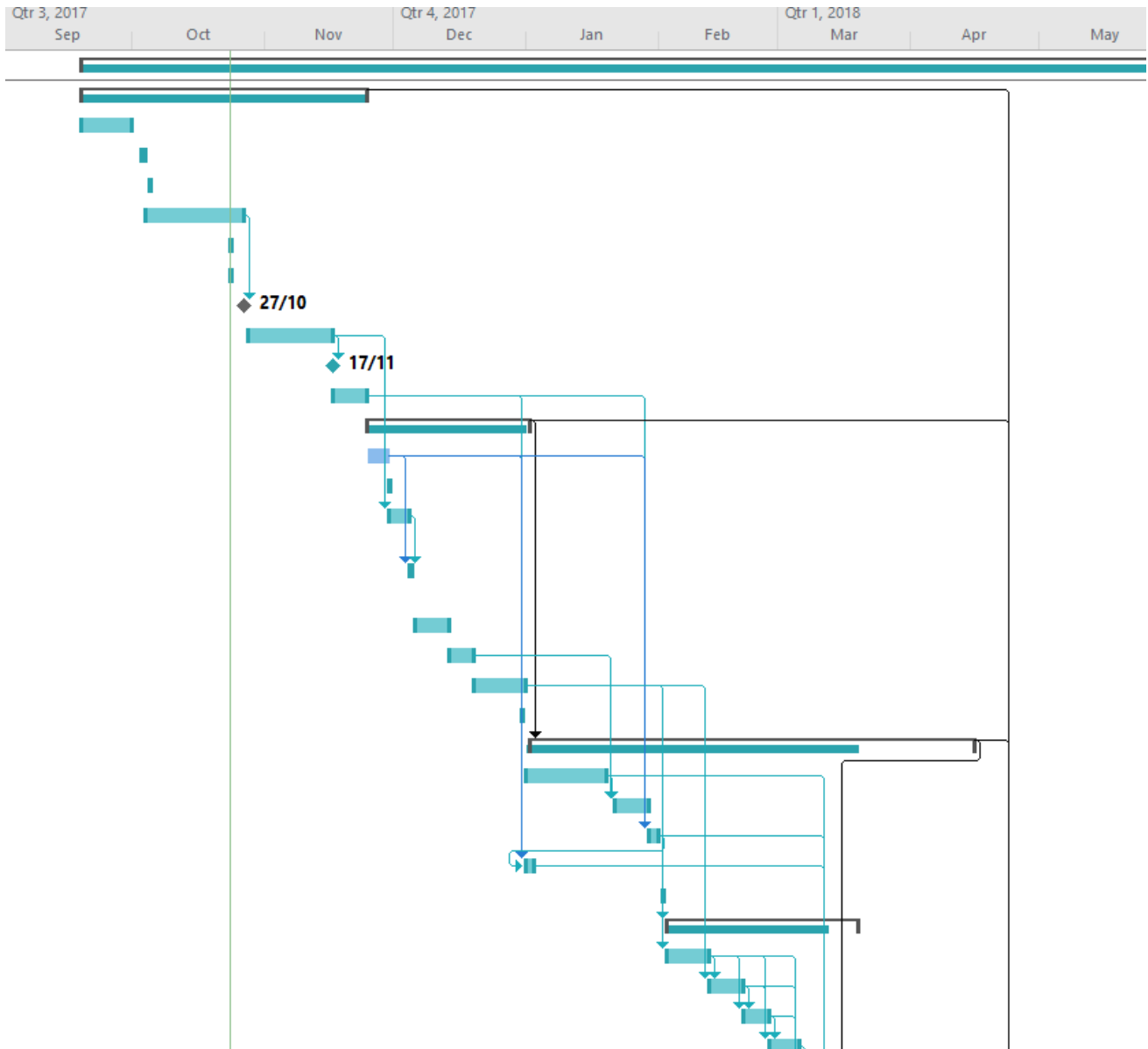
SPECIAL RESOURCES REQUIRED

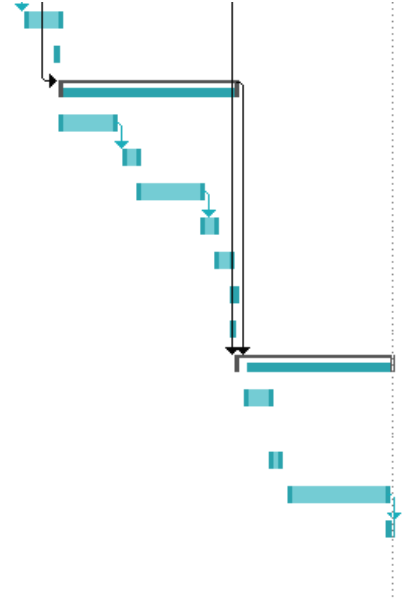
- I will be deploying the application in AWS so firstly I will need an AWS account. I will initially be testing my application on a t2.micro EC2 instance and also an RDS db.t2.micro MySQL instance. For my production environment I will be moving to a t2.medium EC2 instance and multiple corresponding RDS MySQL instances.
- For the machine learning element, I will need to source relevant datasets from Kaggle.com.
- Towards the end of the project I will need 20 individuals to test the application features both simultaneously and individually.
- I will need to download the Burp suite for penetration testing purposes.

PROJECT PLAN

		Task Mode ▾	Task Name ▾	Duration ▾	Start ▾	Finish ▾	Predecessors
1			▾ 1.0 Edyoucate.ie	181 days	Tue 19/09/17	Sat 26/05/18	
2			▾ 1.1 Initiation and Documentation gathering	49 days	Tue 19/09/17	Fri 24/11/17	
3			1.1.1 Research project possibilities	10 days	Tue 19/09/17	Sat 30/09/17	
4			1.1.2 Project Proposal Ratification	1 day	Tue 03/10/17	Tue 03/10/17	
5			1.1.3 Supervisor meet and great	30 mins	Thu 05/10/17	Thu 05/10/17	
6			1.1.4 Proposal information gathering and creation	17 days	Wed 04/10/17	Thu 26/10/17	
7			1.1.5 Supervisor meeting Proposal revision	30 mins	Tue 24/10/17	Tue 24/10/17	
8			1.1.6 October Journal	1 hr	Tue 24/10/17	Tue 24/10/17	
9			1.1.7 Proposal submission	0 days	Fri 27/10/17	Fri 27/10/17	6
10			1.1.8 Define Requirements Specification	15 days	Sat 28/10/17	Thu 16/11/17	
11			1.1.9 Requirements Specification submission	0 days	Fri 17/11/17	Fri 17/11/17	10
12			1.1.10 Research LAMP stack requirements	6 days	Fri 17/11/17	Fri 24/11/17	
13			▾ 1.2 Planning and Prototyping	27 days	Sat 25/11/17	Mon 01/01/18	
14			1.2.1 Deploy prototype LAMP stack on AWS	4 days	Sat 25/11/17	Wed 29/11/17	
15			1.2.2 November Journal	30 mins	Thu 30/11/17	Thu 30/11/17	
16			1.2.3 Add skelaton web frame do demonstrate basic functionality	3 days	Thu 30/11/17	Mon 04/12/17	10
17			1.2.4 Mid Point presentation and prototype showcase	1 hr	Tue 05/12/17	Tue 05/12/17	16,14
18			1.2.5 Login and session creation and management	6 days	Wed 06/12/17	Wed 13/12/17	
19			1.2.6 2factor auth research	4 days	Thu 14/12/17	Tue 19/12/17	
20			1.2.7 Machine learning model research	9 days	Wed 20/12/17	Sun 31/12/17	
21			1.2.8 December Journal	30 mins	Sun 31/12/17	Sun 31/12/17	
22			▾ 1.3 Execution and Deployment	75 days	Tue 02/01/18	Sun 15/04/18	13
23			1.3.1 Develop PHP templates	15 days	Mon 01/01/18	Fri 19/01/18	
24			1.3.2 Implement 2Factor Auth to login	6 days	Mon 22/01/18	Mon 29/01/18	19,23
25			1.3.3 LAMP development (EC2 instance)	2 days	Tue 30/01/18	Wed 31/01/18	12,14
26			1.3.5 LAMP development (MySQL instance)	2 days	Mon 01/01/18	Tue 02/01/18	12,14,25
27			1.3.6 January Journal	30 mins	Fri 02/02/18	Fri 02/02/18	
28			▾ 1.3.7 Machine learning Implementation	32 days	Sat 03/02/18	Mon 19/03/18	20
29			1.3.7.1 Linear Regression research	7 days	Sat 03/02/18	Mon 12/02/18	20
30			1.3.7.2 Construct algorithm	6 days	Tue 13/02/18	Tue 20/02/18	20,29
31			1.3.7.3 Data injection and validation Stage 1	4 days	Wed 21/02/18	Mon 26/02/18	30,29
32			1.3.7.4 Data injection and validation Stage 2	5 days	Tue 27/02/18	Mon 05/03/18	31,30,29
33			1.3.7.5 February Journal	30 mins	Mon 05/03/18	Mon 05/03/18	
34			1.3.7.6 Final Validation and Testing	5 days	Tue 06/03/18	Mon 12/03/18	32,31,30,29

35	★	1.3.8 Deploy web application on LAMP stack	5 days	Tue 13/03/18	Mon 19/03/18	25,26,23
36	★	1.3.9 March Journal	30 mins	Mon 19/03/18	Mon 19/03/18	
37	★	▸ 1.4 Monitoring & Testing	26 days	Tue 20/03/18	Tue 24/04/18	22
38	★	1.4.1 Burp suite penetration testing	9 days	Tue 20/03/18	Fri 30/03/18	
39	★	1.4.2 Code refactor	3 days	Mon 02/04/18	Wed 04/04/18	38
40	★	1.4.3 User testing with 20 testers	9 days	Thu 05/04/18	Tue 17/04/18	
41	★	1.4.4 Code refactor	3 days	Wed 18/04/18	Fri 20/04/18	40
42	★	1.4.5 Apache Jmeter stress test	2 days	Sat 21/04/18	Mon 23/04/18	
43	★	1.4.6 Code refactor	1 day	Tue 24/04/18	Tue 24/04/18	
44	★	1.4.7 April Journal	30 mins	Tue 24/04/18	Tue 24/04/18	
45	★	▸ 1.5 Closing	24 days	Wed 25/04/18	Sat 26/05/18	2,13,22,37
46	★	1.5.1 Showcase Materials preparation and submission	3 days	Fri 27/04/18	Tue 01/05/18	
47	★	1.5.2 Showcase poster creation	2 days	Wed 02/05/18	Thu 03/05/18	
48	★	1.5.3 Preperation for final showcase	16 days	Sun 06/05/18	Fri 25/05/18	
49	★	1.5.4 Final Showcase	1 day	Sat 26/05/18	Sat 26/05/18	48





Link to file:



edyoucate.mpp

TECHNICAL DETAILS

To develop this application, I will be utilizing several technologies, those being:

Hosting and Deployment:

I will start by creating a LAMP stack on the AWS platform for hosting the application. The browser will send a request to the Apache server which will then pass the request to the PHP scripts to retrieve the data from the database to generate the webpage which is then sent back to the browser via the Apache server. I will be using Route53 DNS which will connect to the LAMP stack. I will be utilizing an EC2 t2.micro and RDS MySQL db.t2micro for initial testing and will then be scaling up to an EC2 t2.medium and 2 RDS MySQL db.t2.medium instances. For the machine learning algorithm I will need an instance with more GPU power so I will be utilizing a p2.xlarge, however this is pending further research which I have allotted time for in my project plan.

Frontend:

For the frontend development, I will be utilizing HTML5, CSS3 and JavaScript. Depending on time constraints I may utilize WordPress.

Backend:

The server side scripting language I have decided to use is PHP. I chose this language as there is a ton of resources and useful libraries I can call upon and it is compatible with Apache. During the sign up process I will be integrating a CAPTCHA using PHP. If the CAPTCHA code is validated, the secure session will start, however if it is not validated an error message will be displayed and the script will terminate with exit(). As one of the most important features of the application is file uploading, I will be using the Stash library to speed up database queries and API calls by caching the results of expensive/taxing functions. I will be using the AWS PHP SDK version 3, this is Amazon's official PHP library for working with AWS. For the 2Factor authentication I will be using the Google authenticator PHP class and to secure this I will be limiting the number of verifications from a single IP address to 10 tries over the space of 10 minutes.

API's:

I will be using 2 different APIs to secure the uploads, those being the TotalVirus API which will be used to scan all file uploads for malicious scripts and the AWS Rekognition API to remove illicit pictures.

EVALUATION

My main evaluation priority for this application is to make sure it is as secure as possible. To ensure this I have allocated time in my project plan to carry out a number of tests. The first stage of this testing will be on the code integrity, I will be testing this with a penetration testing application called Burp: a suite of tools used to “hack” a website or application. The main tool from this suite I will use is Intruder, this is used to automatically scan and attack the application using malicious HTTP requests to perform an SQL injection, cross site scripting and pinpoint vulnerabilities for brute force attacks. If a vulnerability is found, I will refactor the code and perform the tests again. I will then test the application across multiple browser such as Chrome, Firefox and Edge to insure compatibility.

Once the code integrity has been thoroughly tested, I will then begin the second stage which is the user evaluation. This stage will be split into 2 phases, the first phase will entail 20 separate single user tests at different times. I will provide each user with a set of tasks to carry out and once finished they will fill out a survey detailing their thoughts of the application and any issues encountered. The second phase will entail all 20 users log in and using the application simultaneously to perform a real-world stress test. If any is feedback I provided and/or recommendations made, I will refactor the code and ask the test users to conduct a follow up evaluation. If any issues arise from the stress test, I will reconfigure the auto scaling of the environment, however as it may not be feasible to gather the group of 20 users gain, I will use a semi-automated software stress testing application called Apache JMeter.

Monthly Journals

SEPTEMBER

GOALS

This month I need to:

- Solidify the idea,
- Research technologies to use,
- Draft a proposal.

ACHIEVEMENTS

I achieved all goals I set out to accomplish this month. I decided on a web application that will help college students from all 3rd level institutions across Ireland to share their materials, much like a nationwide moodle. I researched different technologies incorporate such as upload/download tech and also decided to implement a machine learning model. I decided on AWS for the backend infrastructure.

OCTOBER

GOALS

This month I need to:

- Define exactly what I would like the app to do,
- Define the different user access levels

ACHIEVEMENTS

I did not fully achieve the goals I set out to do this month due to other module deadlines, however I did solidify the user access level and I also put some time into how I want the end result to look. Further more I decided on deploying the app on a LAMP stack back by an ELB.

NOVEMBER

GOALS

This month I need to:

- Define exactly what I would like the app to do,
- Research non-functional requirements,
- Create class diagram
- Draft a Requirements specification

ACHIEVEMENTS

This month I put a lot of work into the application on the conceptual side and the documentation side.

As last month I did not fully decide on the functionalities, I began the month by completing this goal and decided on the core functionalities of upload/download, forums, machine learning model and global chat. From this information I then drafted the functional requirements section of my Req Spec. As I had put some research into the backend infrastructure last month, I was also able to draft the non-functional requirements and create a class diagram. I then combined all sections to create the final Requirements Specification document.

DECEMBER

GOALS

This month I need to:

- Prepare for mid-point presentation
- Research most suitable IDE for development
- Research deployment options

ACHIEVEMENTS

Most of my time this month was split between the preparing for the midpoint presentation and researching deployment options. After my presentation was complete I put all my time into deployment options. Decided to utilize Amazon Web Services EC2, EBS and Route53. This was also reinforced by the fact that cloud9 was acquired by AWS which meant I could use cloud9 on top of AWS.

JANUARY

GOALS

This month I need to:

- Launch my AWS architecture
- Purchase a Domain

ACHIEVEMENTS

This month I research what size EC2 instance I should utilize. I was also initially going to use an RDS instance for my database, however I decided I would first build it locally and then if I had time I could migrate the local phpmyadmin database over to the RDS instance. I decided to launch a t2.micro instance and an 8GB EBS volume.

I also purchased the domain name edyoucate.ie from Irish domains.

FEBRUARY

GOALS

This month I need to:

- Research good secure practices for login systems
- Database development
- Start login system
- Start registration system

ACHIEVEMENTS

This month I began the development of the login system in my tester cloud9 account rather pushing straight onto my live environment. Before I started I researched good secure practices for login systems.

I also build the database in phpmyadmin. Once I was satisfied that the login system was working correctly I the test account, I moved it to the production account.

I also developed the registration system.

FEBRUARY

GOALS

This month I need to:

- Test login system
- Research upload methods
- Research machine learning model implementations

ACHIEVEMENTS

This month I tested the login in system using the burp suite. I test it by inputting Javascript to try get some feedback which failed due to the implementation of escape strings. I then tested for SQL injection by inputting an SQL query however this also failed. The last test I carried out was a brute force attack on a weak password protected account, this however was successful. I then researched and implemented a brute force protection function which locks the account after 5 failed attempts.

This unforeseen challenge meant I did not have to time focus on the upload and machine learning elements.

MARCH

GOALS

This month I need to:

- Research upload and VirusTotal scanning
- Research machine learning

ACHIEVEMENTS

I knew what way I was going to upload the files, however the structure was quite challenging. I needed to have a tab for each category and an accordion for each corresponding topic. I was able to achieve this using JavaScript for the most part. I then used a simple directory viewer which echo's the files with in their dedicated directories. Once I had decided on the structure I then began work on implementing the Virustotal API. Having researched the API and the work involved and also having researched machine learning and the worked involved in the model training I came to the conclusion the it would not be realistic to continue with the machine learning.

I used the classes provided by VirusTotal to shape the structure of the upload and schedule the scan.

APRIL

GOALS

This month I need to:

- Continue the VirusTotal scanning
- Complete the Forum section

ACHIEVEMENTS

This month I put more work into forming the database table to handle the uploads and virustotal input.

I experienced issues when I was running the cronjob that pulls the data from virustotal. The script was running fine when I was running it manually through the CLI, however when I tried to run it via cron it was not removing the malicious file from the server but it was removing the entry from the database.

After some time and research I discovered this was due to the fact that cron does not run from within the web directory, so it was in easy fix in that I just needed to add `/var/www/html` to the file path.

After some research on Udemy I developed a php ODP forum. The forum took shorter than expected to build as I had the help of a follow along course.

For the first time in a few months I met the outlined goals.

MAY

GOALS

This month I need to:

- Pen Testing
- Unit Testing

ACHIEVEMENTS

This month I worked solely on testing the application from all aspects.

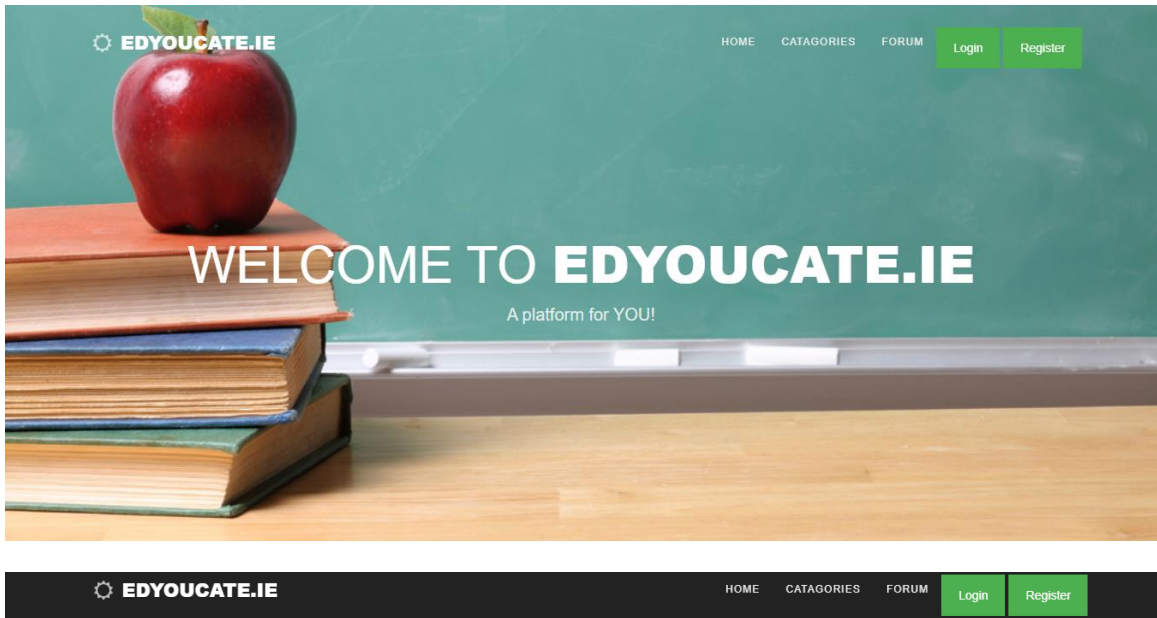
I started by unit testing each feature and was happy to discover all unit tests passed successfully.

I then used the burp suite to manually test all user input and URL and privilege escalation vulnerabilities.

Edyoucate.ie

User Manual

When a user first visits the site they are met by the homepage for which they can navigate and read about the site:



What is EdYOUcate.ie and how can it benefit you?



What is edyoucate.ie?

A content sharing platform for students!

A place for learning and exploring!

A place for questions answers and discussion!

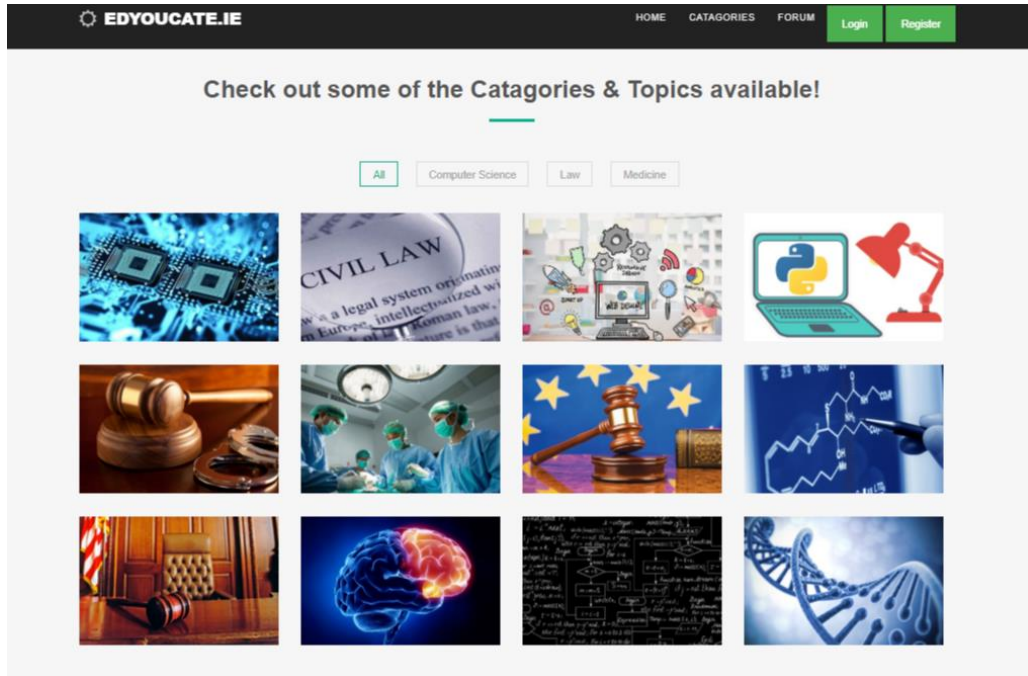
Why should I use it?

Access content relevant to your studies,

Communicate with other students via the forum,

Safely download and review materials,

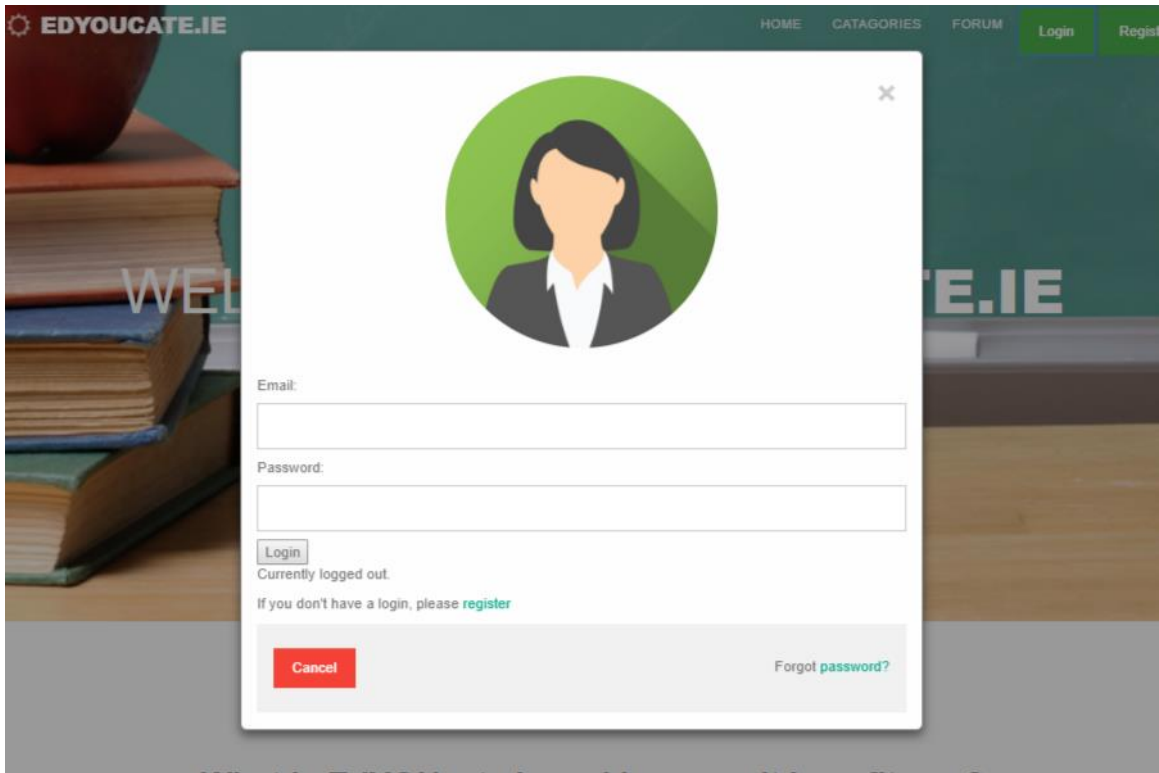
and best of all, its completely FREE!!



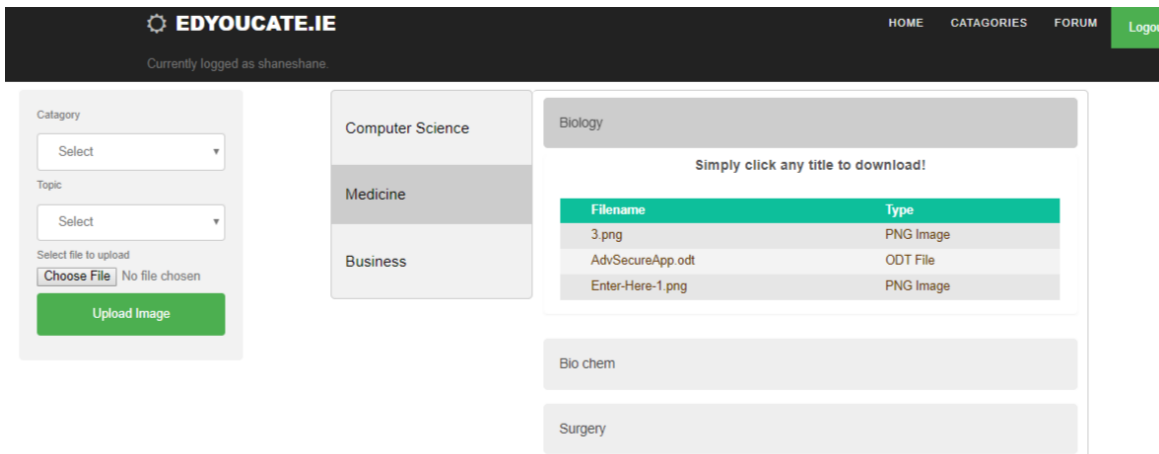
As per the main flow a non-registered user can freely navigate the main homepage, however if they click on the “Categories” tab they will be redirected to the registration page where they will be asked to register an account using an username, email address and password:

The screenshot shows the registration page of Edyucate.ie. The top navigation bar is identical to the homepage. The main heading is 'Register with us'. Below the heading, there are several lines of text providing registration rules: 'Usernames may contain only digits, upper and lowercase letters and underscores', 'Emails must have a valid email format', 'Passwords must be at least 6 characters long', 'Passwords must contain', 'At least one uppercase letter (A-Z)', 'At least one lowercase letter (a-z)', 'At least one number (0-9)', and 'Your password and confirmation must match exactly'. Below these rules are four input fields: 'Username:', 'Email:', 'Password:', and 'Confirm password:'. At the bottom of the form is a 'Register' button and a link that says 'Return to the login page.'

Once the user has registered they will then be redirected back to the homepage where they can sign in with their new credentials using the Login button in the top right corner of the page:

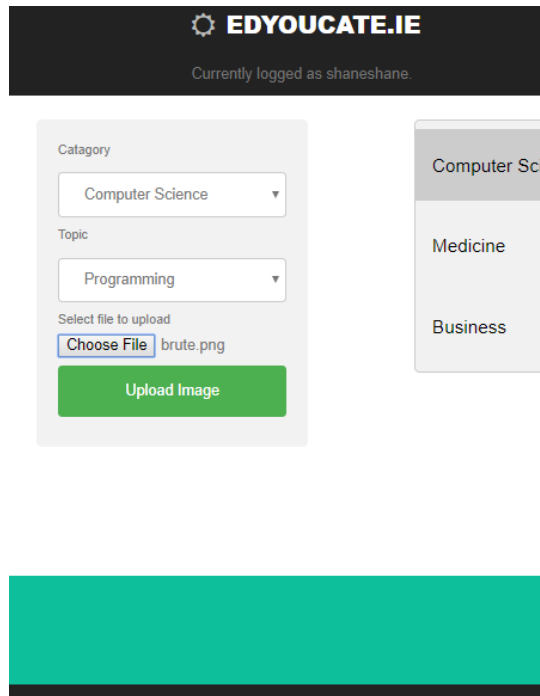


When the user logs in they can then navigate to the “Categories” where they can view materials. To view the materials the user simply needs to click the desired category and then the desired topic:

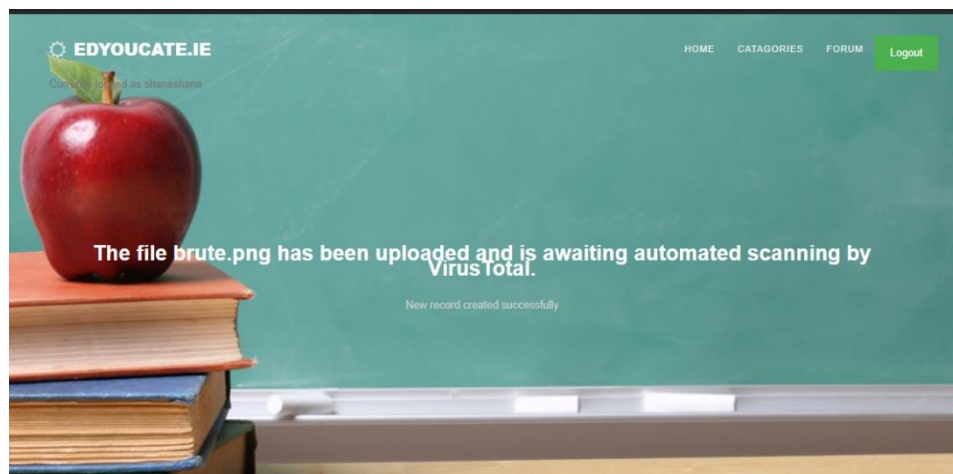


In order to download a file, the user simply needs to click on the desired file. If the file is an image it will open in the browser, any other file type will begin download.

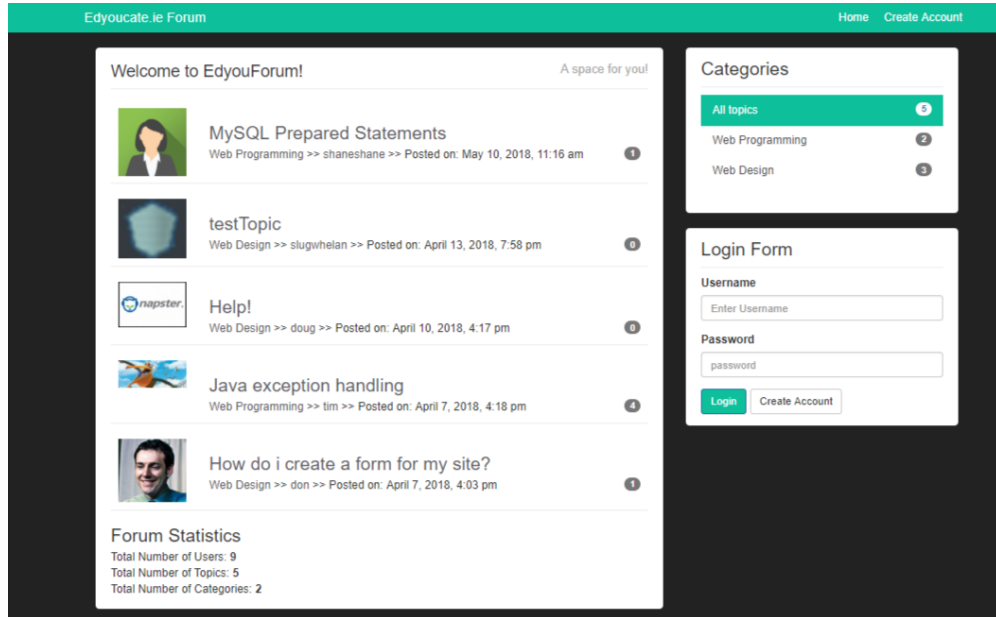
If you user would like to upload material they can do so by using the upload form on the left of the screen, selecting the desired category, the desired topic and then using the file chooser to choose a file from their local machine:



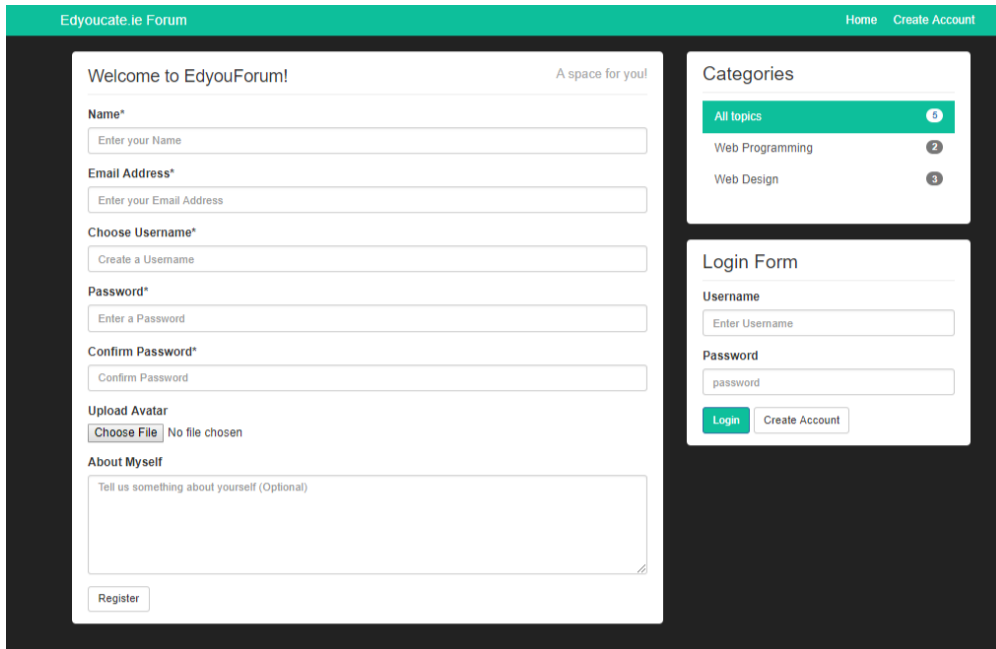
If they are happy to upload, they can click the Upload button which will submit the file to the chosen position and also queue the file for scanning by VirusTotal:



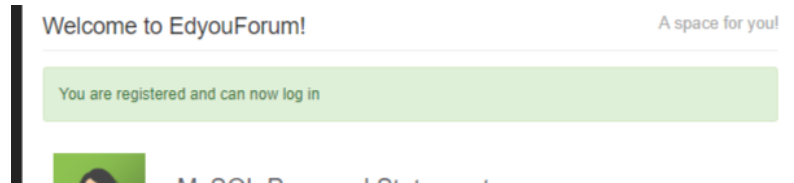
Registered and non-registered users alike can both visit the forum where they can view all forum posts.



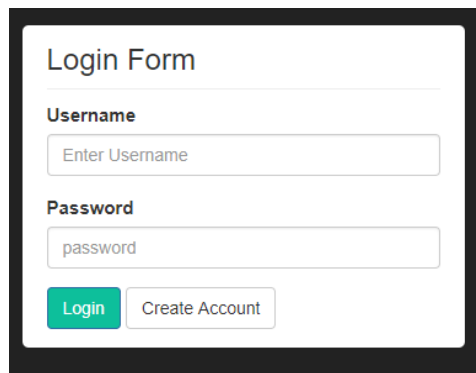
Before a user can create post or comment on a post they first need to register an account. They can do this by clicking the register button on the frontpage. The register requests the name, username, email, password, avatar and has an optional about section.



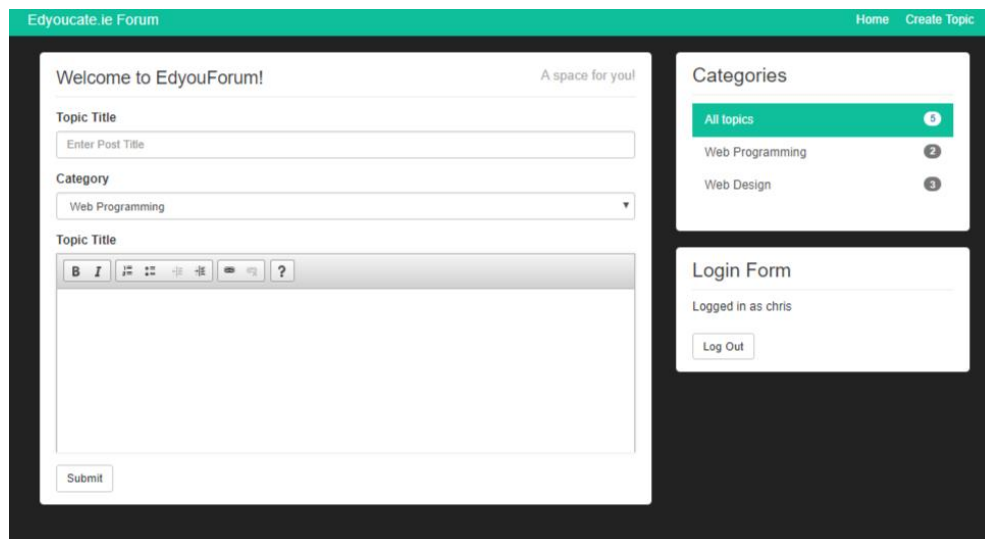
Once the user has added the required fields they can then click the register button which will redirect them back to the frontpage of the forum and display a registration successful message:



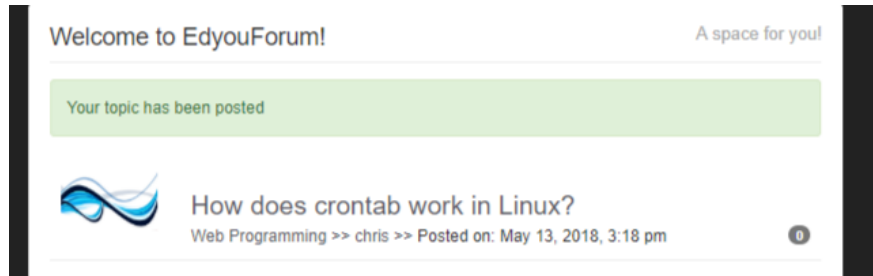
Now that the user is fully registered they can login using the login form:



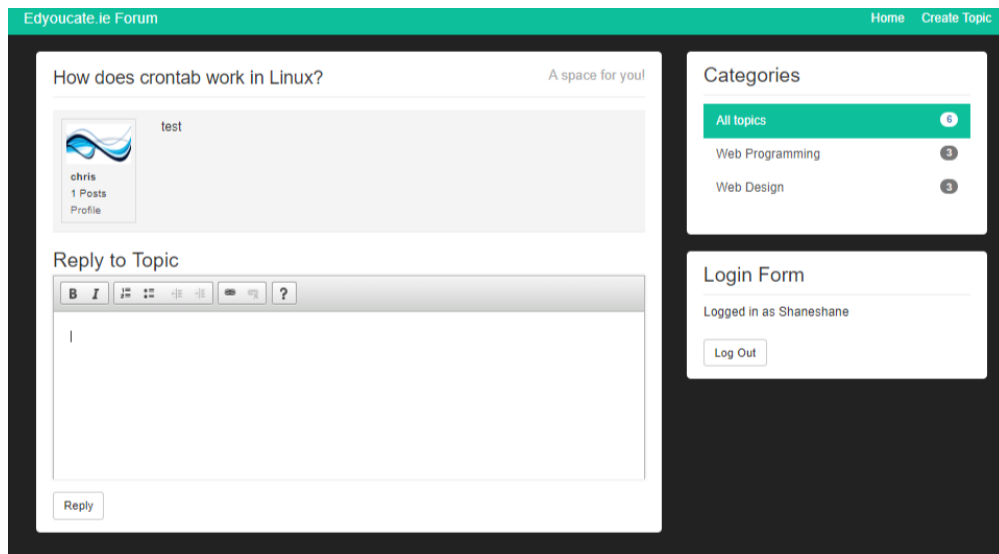
Now that the user is logged in, they can create a forum post by clicking the create button in the top right corner of the page. This will bring the user to a form where they can choose a title, choose a category and add further details of the question:



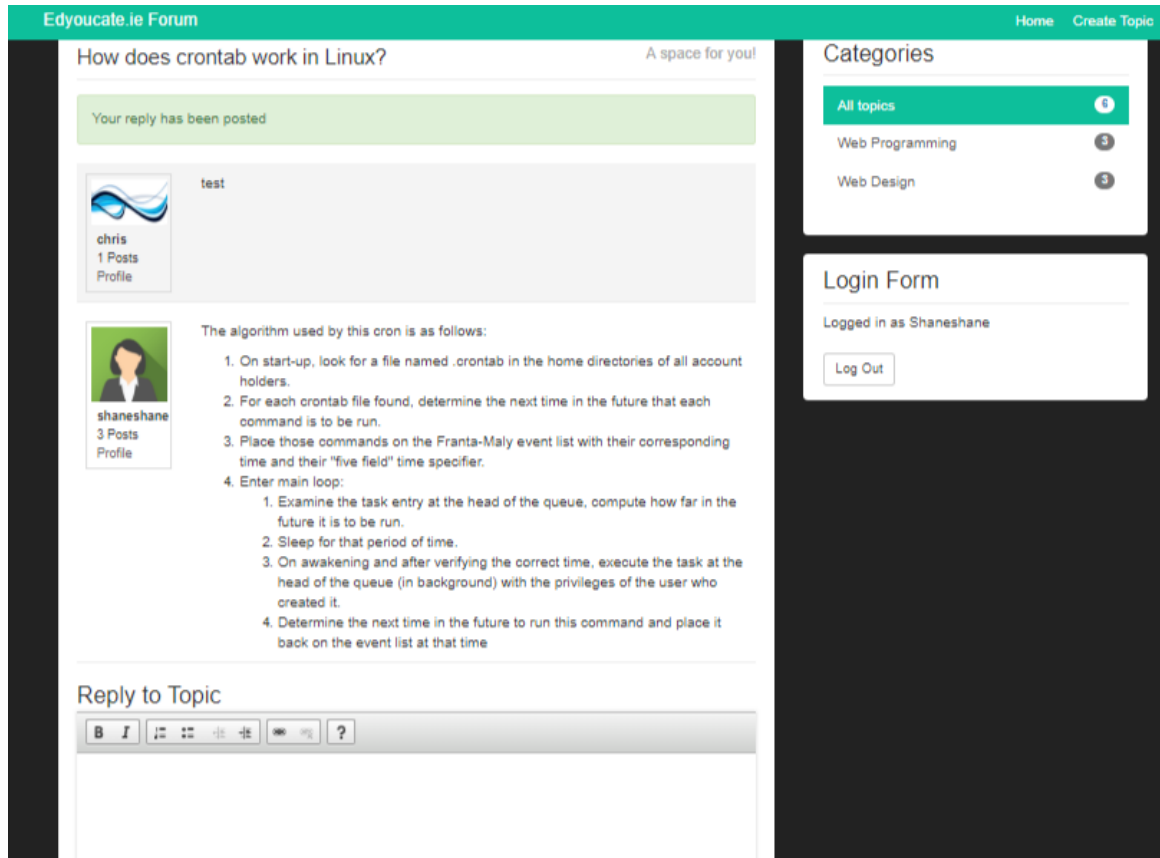
Once the user has filled in all fields, they can hit the submit button which will submit the post and redirect the user back to the forum frontpage where they will see a confirmation message and their post will be at the top of the list:



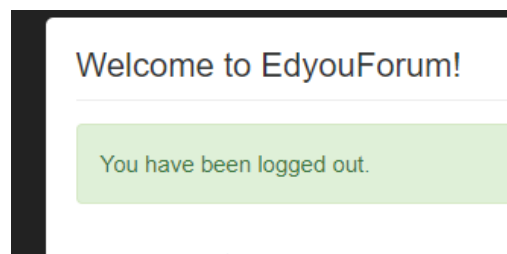
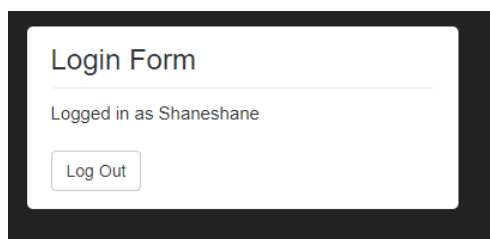
Users can also comment on other post by clicking on the desired post title. This will take the user to the post where they will find a blank text area:



Once the user fills in their reply they can click the reply button which will instantly post the reply to the page and they will also receive a reply confirmation:



When the user has finished reviewing the forums they can use the logout button to log themselves out of the application. Once successfully logged out they will also receive a logout confirmation:



REFERENCES

Amazon Web Services, Inc. (2017). *Amazon Relational Database Service (RDS) – AWS*. [online] Available at: <https://aws.amazon.com/rds/> [Accessed 2 Nov. 2017].

Amazon Web Services, Inc. (2017). *Amazon Simple Storage Service (S3) — Cloud Storage — AWS*. [online] Available at: <https://aws.amazon.com/s3/> [Accessed 6 Nov. 2017].

Amazon Web Services, Inc. (2017). *AWS | Elastic Load Balancing - Cloud Network Load Balancer*. [online] Available at: <https://aws.amazon.com/elasticloadbalancing/> [Accessed 3 Nov. 2017].

Amazon Web Services, Inc. (2017). *Elastic Compute Cloud (EC2) – Cloud Server & Hosting – AWS*. [online] Available at: <https://aws.amazon.com/ec2/> [Accessed 3 Nov. 2017].

Chris Veness, 2. (2017). *SHA-256 Cryptographic Hash Algorithm implemented in JavaScript | Movable Type Scripts*. [online] Movable-type.co.uk. Available at: <https://www.movable-type.co.uk/scripts/sha256.html> [Accessed 24 Oct. 2017].

Portswigger.net. (2017). *Burp Suite Scanner | PortSwigger*. [online] Available at: <https://portswigger.net/burp> [Accessed 10 Nov. 2017].