Lecture Feedback System

CS39440 Major Project Report

Author: Morgan Jones ([mwj7@aber.ac.uk](mailto:mwj7@aber.ac.uk))

Supervisor: Mr. Chris Loftus (cwl@aber.ac.uk)

22nd April 2019

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a MEng degree in Software Engineering (With Integrated Year In Industry) (G601)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

Declaration of originality

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Registry (AR) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work, I understand and agree to abide by the University’s regulations governing these issues.

Name Morgan Jones

Date 22/04/2019

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Morgan Jones

Date 22/04/2019

Acknowledgements

I’d like to thank my supervisor Chris Loftus for his guidance and patience and those who helped me by trying out the system.

Abstract

The purpose of my project is to create an application for students of the university that they can use to provide anonymous feedback throughout the course of a lecture or workshop. The staff presenting these lectures can then use the system to respond to feedback either during or after a lecture.

This report details my process of the design, building and testing of this system along with an evaluation of that process and of the solution I have produced. I attempt to evaluate the requirements I have outlined and how well those requirements have been met.

Contents

1. Background, Analysis & Process 7

1.1. Background 7

1.1.1. Motivation 7

1.1.2. Research 7

1.2. Analysis 8

1.2.1. Changes from Project Description 9

1.2.2. Requirements (Feature List) 9

1.3. Process 10

2. Design 11

2.1. Overall Architecture 11

2.1.1. Behavioural Diagrams 11

2.1.2. Structural Diagrams 13

2.2. Detailed Design 14

2.3. User Interface Design 15

3. Implementation 17

3.1. Implementation tools 17

3.2. Feature Set 1 – Auth & i18n 17

3.3. Feature Set 2 – Lecture Management 18

3.4. Feature Set 3 – Session Management 19

3.5. Feature Set 4 - Providing Feedback 20

4. Testing 22

4.1. Overall Approach to Testing 22

4.2. Unit Testing 22

4.3. System Testing 22

4.4. User Testing 22

5. Critical Evaluation 23

5.1. Technical Evaluation 23

5.1.1. Requirements & Design Evaluation 23

5.1.2. Process & Implementation Evaluation 24

5.1.3. Testing & Usability Evaluation 24

5.2. Work to Extend the Project 25

5.3. Alternate Approach If Redoing the Project 25

5.4. Overall Evaluation 25

6. Annotated Bibliography 27

7. Appendices 28

A. Third-Party Code and Libraries 28

B. Ethics Submission 29

C. Feature List 32

D. Test Cases 33

E. User Testing 39

Figures

[Figure 1 Original project suggestion description 9](#_Toc7707099)

[Figure 2 Use-Case Diagram 12](#_Toc7707100)

[Figure 3 Staff User Activity Diagram 13](#_Toc7707101)

[Figure 4 Student User Activity Diagram - Providing Feedback 13](#_Toc7707102)

[Figure 5 Entity Relationship diagram describing the structure of my applications database 14](#_Toc7707103)

[Figure 6 Original Component Diagram 15](#_Toc7707104)

[Figure 7 Final Component Diagram 15](#_Toc7707105)

[Figure 8 Class Diagram describing structure of ORM classes in models.py 15](#_Toc7707106)

[Figure 9 Login Page Original 16](#_Toc7707107)

[Figure 10 Login Page Final 16](#_Toc7707108)

[Figure 11 Lecture List Page Original 16](#_Toc7707109)

[Figure 12 Lecture List Page Final 16](#_Toc7707110)

[Figure 13 New/Edit Lecture Page Original 16](#_Toc7707111)

[Figure 14 New/Edit Lecture Page Final 16](#_Toc7707112)

[Figure 15 Lecture Detail Page Original 17](#_Toc7707113)

[Figure 16 Lecture Detail Page Final 17](#_Toc7707114)

[Figure 17 Lecture Detail Page (Canvas Graphs at bottom) Original 17](#_Toc7707115)

[Figure 18 Feedback Detail Page Final 17](#_Toc7707116)

[Figure 19 Connect Page Original 17](#_Toc7707117)

[Figure 20 Connect Page Final 17](#_Toc7707118)

[Figure 21 Feedback Page Original 17](#_Toc7707119)

[Figure 22 Feedback Page Final 17](#_Toc7707120)

[Figure 23 Lecture Detail Page Layout Example 20](#_Toc7707121)

[Figure 24 Example of unit testing project through command line while developing inside atom text editor 23](#_Toc7707122)

# Background, Analysis & Process

## Background

The aim of this project is to build a responsive web application that will allow students in the university to give feedback and ask questions anonymously throughout a lecture or workshop.

The idea being that the students would connect to the system by entering a 6-digit code that would be read out at the start of the lecture, the lecturer taking the session could then respond to the feedback and questions as they are provided by perhaps explaining things further or re-visiting misunderstood material. Any data given would be recorded by the system for review by a lecturer at some point in the future enabling the lecturer to easily see a summary of the feedback and adjust future content delivery accordingly.

### Motivation

This project was of interest because it presented the opportunity to learn and gain experience with web development, in particular the use of a web framework technology, something I had struggled with in the past despite having an interest in and knowing that the ability to develop using a full-stack web framework is a valuable, if not essential, skill in the world of web development.

I was also inclined to take the project because of the accessibility of the problem domain. I myself have been a student for years therefore the transition of starting to think like my end users (students) was an easy one.

### Research

During research all activity, links and ideas were recorded in a project diary that was heavily used during the first few weeks of the project and continued to be used for some weeks into the project.

I knew I wanted to use a web framework approach because my product was to be a web app. The main three I considered were Ruby on Rails [1], Laravel and Django [2] because all three are popular and well established/supported tools. I chose Django in the end despite having previous experience with Ruby on Rails because I am most competent with the python programming language [3] (Django being the web framework for python). This did mean I had to learn Django from scratch, but that learning experience was one of the reasons I chose the project in the first place.

There were many options for my choice of IDE because the standard approach was simply to use a text editor alongside folder/file navigation. The most common editors I saw used in the Django tutorials I watched were either “Sublime Text 3” [3] or “atom” [4] text editors. I chose “atom” because it has all the functionality of sublime text but with the added benefit of being open source and therefore having many plugins available that I could make use of during the project; I installed a command line console plugin [5] that allowed me to run the server and access the database directly without having to switch desktop windows and a beautify plugin [6] that auto-formatted my HTML and python code.

I knew the system would have to be responsive because most student users would access it through their mobile phones during a lecture. My CSS skills are limited and therefore chose to use a CSS library to assist in this aspect of the project. Bootstrap was my choice of technology for assisting with the responsive design because the purpose of the technology is to “Build responsive, mobile-first projects on the web” [1]. I have also had some experience with older versions of it so had a feel for how it was used already.

The application needs to store data in a database. I chose to use a Relational Database Management System (RDBMS) over a NoSQL Database because it is common of this type of web project. I had previous experience with MySQL [9], SQLite3 [10] and PostgreSQL. I chose PostgreSQL [8] because it has the most advanced features and is therefore the most flexible, there is a lot supporting tutorials for using it with Django and it is the database technology I am most competent with having used it all throughout my year in industry.

I wanted to have the feedback display in a visual way and had previous experience using JavaScript along side the HTML canvas to produce client-side graphics; it was my aim at the beginning of the project to use these skills to produce some form of visualisation of feedback in graphs or charts.

As inspiration for my system I looked at a quiz system (quiz.dcs.aber.ac.uk) produced by a student (Adam Lancaster) as a major project in a previous year. It was a good place to start as it showed me the type of style and quality I should be aiming for. The simplistic style of the site and the way he presented data visually stood out to me as something I would want in my app. I also discussed with my supervisor the means of authenticating staff using a login and connection over LDAP to a university server, this was how the student achieved the staff login functionality on his project; a functionality I too would need to implement.

As result of this I tried reading the LDAP RFC specifications and investigated the use of a python library (ldap3) [9]that would function as an LDAP API for my Django application. My supervisor also forwarded me an email that was previously provided to Adam with instructions on how to connect and process the data returned by the LDAP application level protocol.

I then read the tutorials on writing a custom authentication backend to accommodate for my authentication via LDAP, this would still allow me to use Django’s built-in authentication system.

The system was to be used by the university. Every other web service the university provides is available in both Welsh and English because the university is bilingual. It seemed appropriate that the application should also be available in Welsh and English.

## Analysis

After the above-mentioned background preparation, the project direction was decided. The system was to be built in python following the Django web framework using a customised version of the atom text editor as an IDE. JavaScript was to be used to add some form of data visualisation and Bootstrap was to be used to ensure responsive design.

Sessions for lectures would only be managed by members of university staff this would be made possible by a user login functionality that utilised authentication over LDAP to a university server. Feedback would only be provided by those with an active 6-digit session code given out by a member of staff at the start of a lecture.

### Changes of Project Title



Figure 1 Original project suggestion description

The original project title was “Lecture traffic lights and feedback system” because it was thought the feedback would be in the form of Green, Red and Amber responses. In discussion with my supervisor it was mentioned that those responses may not be that useful because they tell little about what was positive or negative. The feedback was therefore expanded to include a rating on multiple lecture aspects. Below are the options chosen to present to student users:

|  |  |
| --- | --- |
| * Delivery Speed   + Very Slow   + Little Slow   + Just Right   + Little Fast   + Very Fast * Content Complexity   + Very Easy   + Slightly Easy   + Normal   + Slightly Difficult   + Very Difficult | * Content Presentation   + Very Well Presented   + Well Presented   + Not Well Presented * Level of Engagement   + Very Engaging/Interesting   + Engaging/Interesting   + Not Engaging/Interesting * Overall Feedback   + Good   + Bad   + So-So |

### Requirements (Feature List)

Shortly after initial research and learning a feature list was produced; as is process convention in an FDD project. The problem was broken down into functionally valuable features each separated into four feature sets with estimated weeks for each feature set.

The motivation for having sperate Lecture Management and Session Management came from the idea that a single lecture may be given many times to different classes.

|  |  |
| --- | --- |
| Authentication | The authentication feature set started with only one feature which was LDAP authentication, this was because at the time I was unsure on the difficulty of the feature I later included internationalisation in this feature set due to feature being easier to implement than first expected. |
| Lecture Management | Lecture Management is more basic CRUD functionality which expanded throughout the project to include search query functionality and functionality to work with PDF uploads. |
| Session Management | Session Management is more to do with managing live sessions through a control panel style page this made use of JavaScript, jQuery and AJAX. This feature set also included the feature for visually displaying feedback data. |
| Providing Feedback | Providing Feedback was meant to be the simpler UI as it was intended for use by students on mobiles. It required connecting to an active session and maintaining user data through use of a session on the server. |

Figure 2 Description of the four feature sets

The content of these feature sets evolved throughout the project and were not all ordered by dependency. Work was switching back and forth between the Session Management and Feedback feature sets frequently as updates to features in one feature set prompted changes in another. The feature list itself is included in the appendix A.C.

## Process

I originally intended to follow a personalised version of feature-driven development (FDD) as my engineering process [13]. I chose FDD because it is recommended when building a project which is well defined and would allow me to get a lot of design out of the way during the first three FDD process steps (AKA iteration zero) [13]. This approach worked well at the start of my project as it resulted in me producing a range of high-level design diagrams to model the system which served as a useful starting point to which only incremental updates were necessary throughout the project. I also produced an ordered feature list that functioned as a requirements specification and reference point for noting my own progress.

As the project progressed, I found my progress tracking to be unaligned with FDD because my iterations did not follow the standard six FDD iterative milestones. I was working through the features on my feature list but was writing code, updating design and tests at the same time. This hinted at my adoption of XP-style iterations and caused me to abandon my attempts at maintaining an FDD progress tracking report.

Into the second half of the project I found that I was almost entirely writing code and unit tests with very little updates to the formal test or design. This apparent decrease in discipline was due I think to my attempt to make unplanned changes that I was often unsure of and resulted in failures/rollbacks or minor advancements. It is for this reason I have kept this report in the structure of a plan-based project because from an honest perspective the project structure now reflects that with investment in design, implementation and testing in that order.

I have kept this report in the structure of a plan-based project because from honest perspective the project structure somewhat reflects that with most design at the start and most testing at the end.

# Design

## Overall Architecture

Towards the start of the project I produced several high-level Unified Modelling Language (UML) designs to model the system; these diagrams were updated as the project progressed. I used the UML modelling software Visual Paradigm (community edition) to produce the diagrams [13].

### Behavioural Diagrams

Behavioural diagrams are used to display the system functionality and detail the intended user workflow. I used a use-case diagram to brainstorm application functions that both staff and student users would want to access.

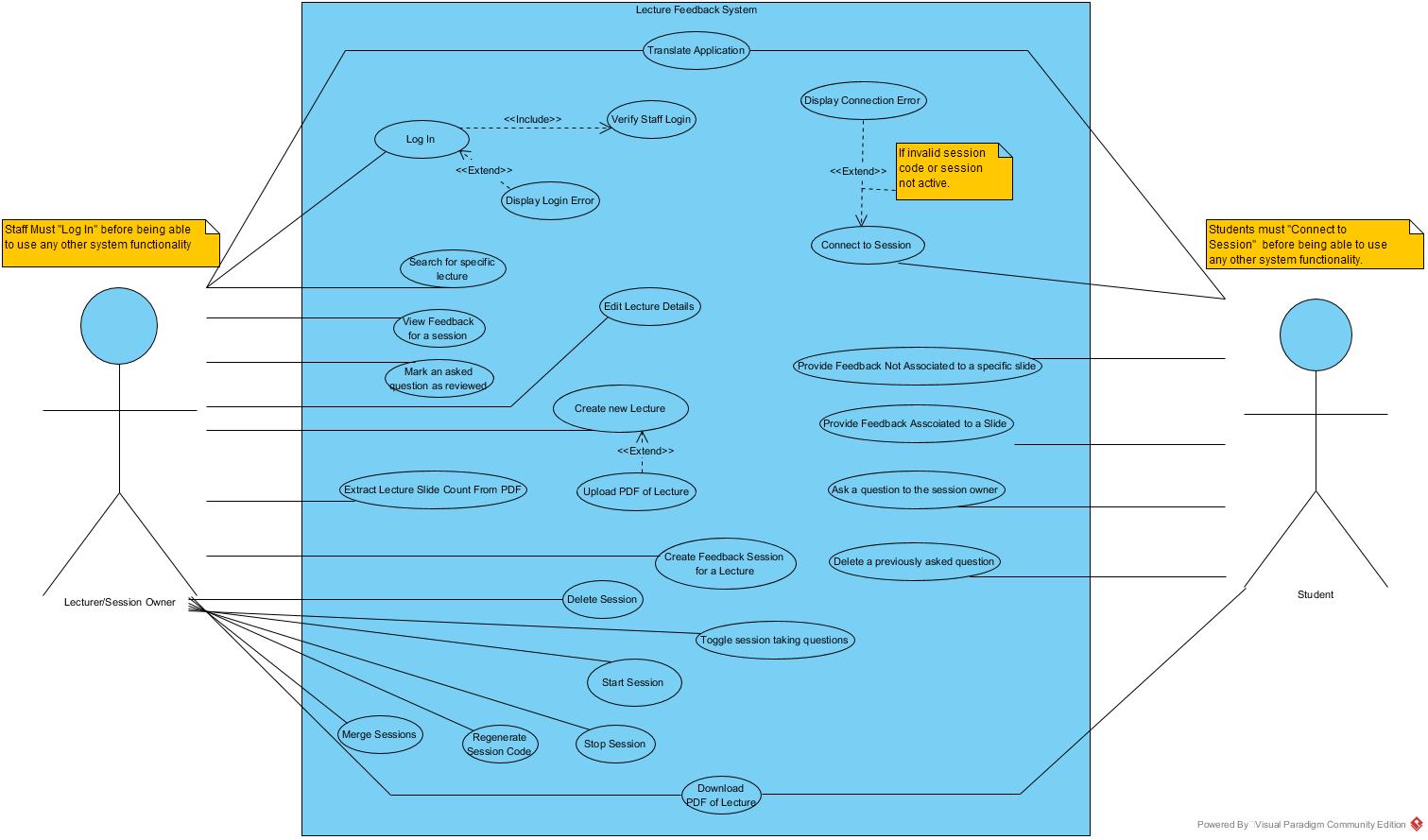


Figure 3 Use-Case Diagram

Activity diagrams to describe in detail the set of actions and decisions users would take when using the application. There is one activity diagram describing typical student workflow and another describing typical staff workflow.

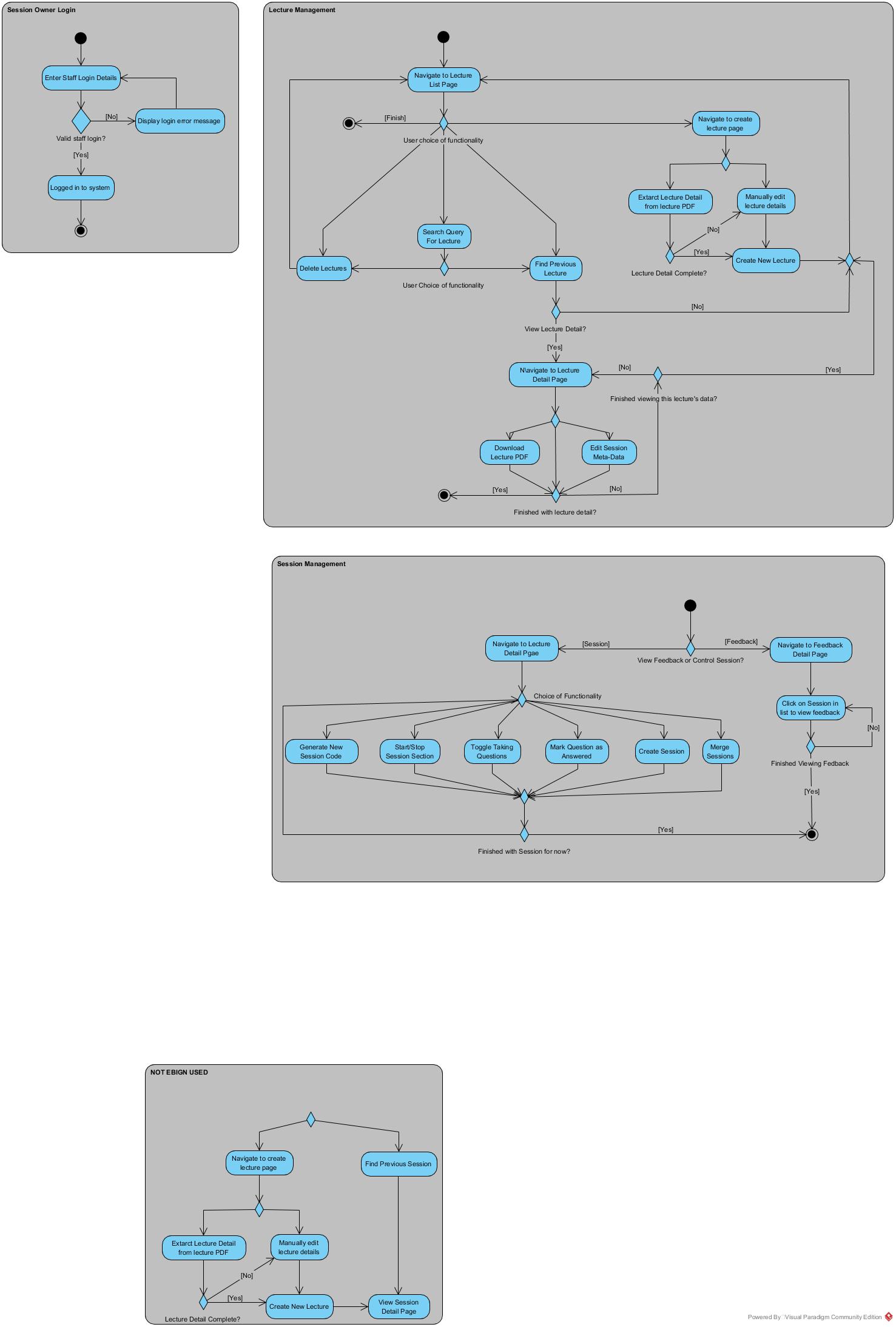


Figure 4 Staff User Activity Diagram

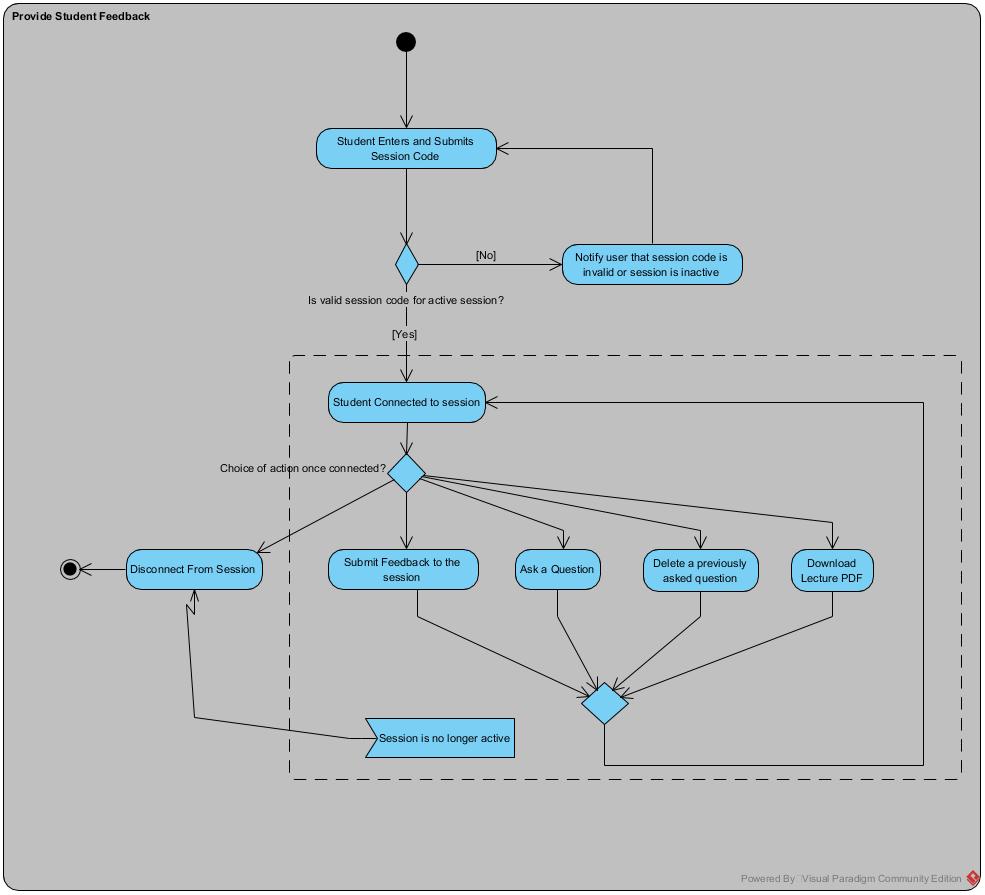


Figure 5 Student User Activity Diagram - Providing Feedback

### Structural Diagrams

An entity relationship diagram was produced describing the underlying data needing to be stored by the application. Focusing on data first was something recommended by my tutor during my year in industry.

The database was redesigned in the middle of the project to reduce redundant data, make the addition of extra functionality easier and allow the use of more Django conventions.

* Relations produced by Django’s authentication contribution were utilised therefore enabling the removal of some attributes from the lecture relation.
* A ‘Time’ relation was added allowing feedback sessions to be re-started by introducing multiple start and end times.
* Shifted attributes from the Lecture relation to the Session relation resulting in more flexibility and direct control over the sessions.
* Removed ‘enum’ relations for the type of feedback options available by instead using a simple varchar type attribute on the feedback table which was restricted only by the hard-coded values supplied to a ChoiceField on a Django model class. This is a more conventional approach for fixed drop-down options in Django.

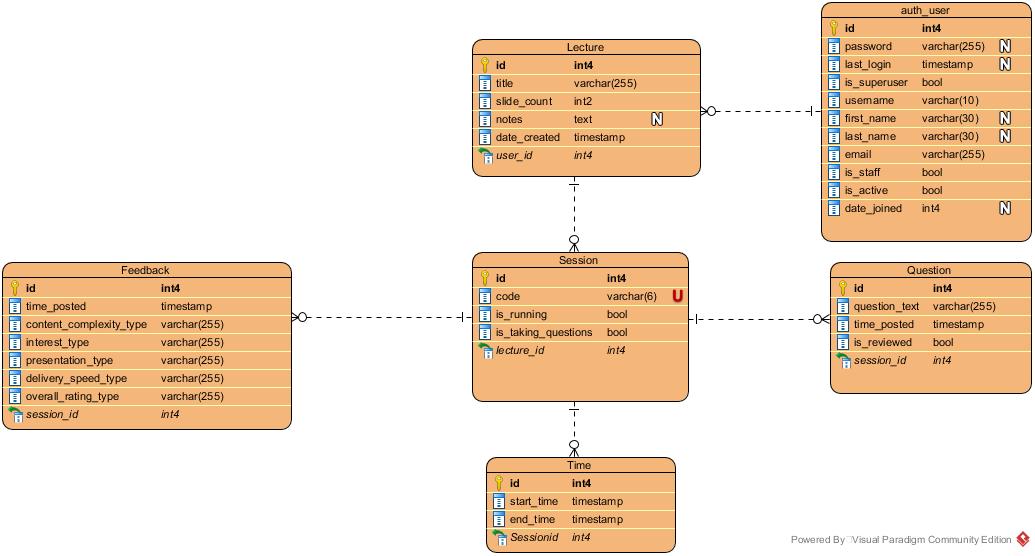


Figure 6 Entity Relationship diagram describing the structure of my applications database

The component architecture of the system is quite basic, it was originally intended to write a desktop program to go along with the system but that feature never got implemented. The exclusion of this feature plus the removal of unnecessary articles on the diagram is the reason for the final diagram’s simplicity.

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated  Figure 7 Original Component Diagram | Figure 8 Final Component Diagram |

## Detailed Design

The code was refactored frequently throughout the project which led to an uncertainty with the low-level design. The original goal was to auto-generate it using some tool, it was attempted to use the pygraphviz python package [16] to do this however there was an issue to do with missing C libraries that could not be resolved.

There was a general lack of documentation for the standard of how to model Django applications perhaps due to that fact that the code you write into a Django application is mostly not class structured albeit the MVC framework is.

A common recommendation for low-level design was to describe the Object-Relational mapping classes with a class diagram. This is only obvious aspect of the code requiring a class diagram of its own since all ORM in Django is always defined in classes.

This might not add that much value as it is close to structure described in the database, which it would be considering these classes are used to build the migrations that in turn build the database.

Below is the class diagram describing the ORM classes from the applications models.py file. These are the model part of the MVC design pattern on which the Django framework is built.

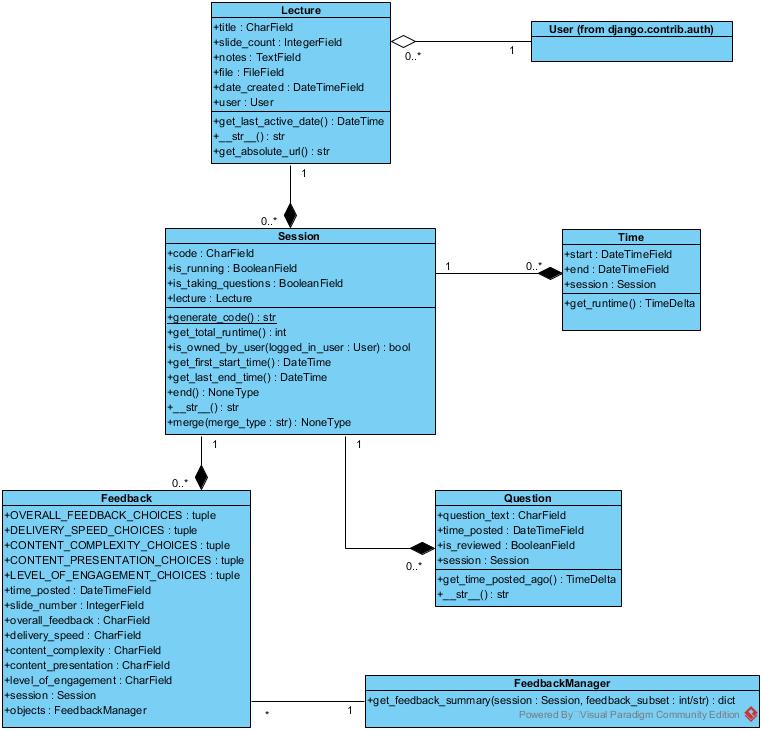


Figure 9 Class Diagram describing structure of ORM classes in models.py

## User Interface Design

While learning Bootstrap during research mock UI designs were created of how I imagined different pages of the site would look. This provided a talking point with my supervisor and others about the look of the site; it also served as a detailed starting point for my HTML templates.

The design of the UI evolved over time as the exact functionality of the pages became clearer. Below is a comparison between the original UI design and the final result.

|  |  |
| --- | --- |
| Figure 10 Login Page Original | Figure 11 Login Page Final |
|  | |
| Figure 12 Lecture List Page Original | Figure 13 Lecture List Page Final |
|  | |
| Figure 14 New/Edit Lecture Page Original | Figure 15 New/Edit Lecture Page Final |
|  | |
| Figure 16 Lecture Detail Page Original | Figure 17 Lecture Detail Page Final |
| Figure 18 Lecture Detail Page (Canvas Graphs at bottom) Original | Figure 19 Feedback Detail Page Final |
|  | |
| Figure 20 Connect Page Original | Figure 21 Connect Page Final |
|  | |
| Figure 22 Feedback Page Original | Figure 23 Feedback Page Final |

One of the main changes between the original design and the final design is the simplification of the interface, which is mainly achieved through removing unneeded text and separating sections more clearly.

I read Steve Krug’s book “DON’T MAKE ME THINK” to revise what to focus on when thinking about a web user interface. Making things concise, visually hierarchically structured and as clear/explicit as possible were all points mentioned in the book and influenced the design of the layout.

# Implementation

## Implementation tools

A python virtual environment [11] was created for the project so that the project’s environment would only contain the dependencies required for it to run. The Python Package Installer (pip) was used to install and manage the project dependencies [12]. There is a requirements.txt file in the project directory that can be used by pip to install all python packages the project depends upon at once through use of the ‘pip install -r requirements.txt’ command.

During implementation python code was debugged through the command line using the Python Debugger (pdb) an interactive source code debugger [10]; the pdb module is part of the python standard library.

The application was developed by running a web server locally on a desktop machine over port 8000. The machine was either within the university network or connected to it via a Virtual Private Network (VPN) during development because it depends on access to ldap.dcs.aber.ac.uk for staff user authentication. The system was used in the google chrome web browser on a desktop and mobile device. Chromes developer tools were used to debug JavaScript and view the page style and HTML DOM.

Microsoft paint was used to create the favicon.ico image for display next to the page title in the browser address bar. Example of image produced below.

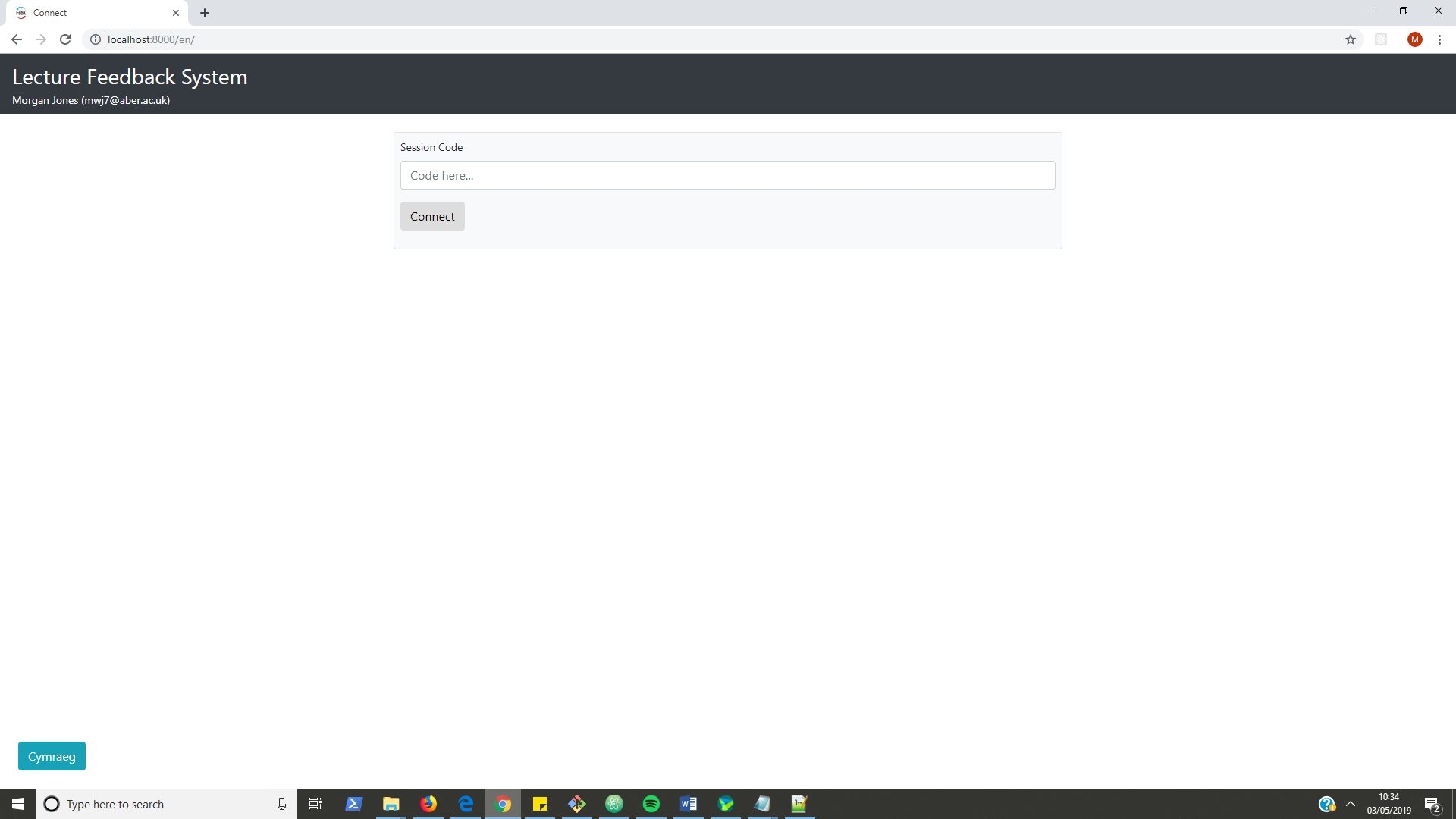


Figure 24 App logo in address bar

Git [20] and GitHub [21] were used for version control and backup during the project. The project repository has a master branch for major working checkpoints, a development branch for regular commits and a learning branch for trying out new ideas and spike testing.

## Feature Set 1 – Auth & i18n

I used the ldap3 python package to communicate over the Lightweight Directory Access Protocol (LDAP) to the university’s server (ldap.dcs.aber.ac.uk). I connected to the server with credentials entered into a login form and then ran a search query for the entry of the logged in user. I parsed the resulting entry for the ‘gecos’ field which the university uses to represent user type.

|  |
| --- |
| Example gecos field - [ABSM]:   * AB – Aberystwyth Campus * SM – Staff User |

I used Django’s user authentication system with a custom authentication backend that used LDAP for authenticating staff as described above. The database relations for the user were therefore created by the django authentication contribution code and not by me. Some of the attributes in these relations are used by my application but not all therefore the database does store some data that is not needed for the application to run.

During development I allowed undergraduate users to log in because I didn’t have a staff account, I only know it works as intended for staff users because my supervisor has tried it out. It would have been better if I could have got a temporary staff account from IS to use while doing the project that way I wouldn’t have to change the code at the last minute to prevent undergraduates logging in also I wouldn’t have to have my password littered throughout the unit tests.

For making the application available in Welsh and English I used Django’s LocaleMiddleware. Use of the “djano-admin makemessages” command would create a .po file listing all string in the code that are marked for translation. I would then go through this file and translate the strings to welsh manually. The “django-admin compilemessages” command was then used to compile messages into a .mo file for use by the middleware. I translated the strings using google translate manually, to save time I could have automated this process by making a small program to use google cloud’s translation API [13].

## Feature Set 2 – Lecture Management

Getting to grips with CRUD, pagination and search query on Django

I created an Object-Relational-Mapped class inside my models.py file for lecture objects this was to be mapped to the ‘Lecture’ relation in my PostgreSQL database. Attributes and attribute restrictions are defined in this class. I used SQL migrations generated as result of the code in my django models to create my database. I found using migrations useful because I didn’t have to switch to write SQL often and all database migrations were saved in order so I could easily switch between different versions or undo incorrect changes.

I started by writing all view code in my app’s view.py file as simple function-based views. The views relating to lecture management mostly only covered the Creation, Retrieval, Updating and Deleting (CRUD) of lecture resources so I later changed them to use slightly customised generic class-based views this is highly recommended as it reduces the amount of code and improves the quality of code. The generic views are inherited from with customisations made in the form of a select few overwritten methods.

The class-based views use a multiple inheritance approach for combing view functionality. These classes are called ‘mixins’ and can be brought into a view through inheritance to provide functionality to multiple views without having to repeat code; this keeps the code DRY which is a principle of Django and other web development frameworks. In my application users have to be logged in to access the views so my class-based views make use of the ‘LoginRequiredMixin’ to ensure lecture resources are only accessed by logged in users. For other views that are still function-based I have used the @login\_required decorator to prevent access if not logged in.

If an unauthorized user attempts to access a view that requires the user to be logged in the user is redirected to the login page with a parameter ‘next’ holding the original URL attempt. Upon successful login the user will be redirected to the page they previously tried to access. Both the login\_required decorator and the LoginRequiredMixin class are provided by Django’s authentication app (django.contrib.auth).

Later in the project I updated the code so each lecture can have an associated PDF file that can be downloaded by students as a convenience functionality. The PDF can be uploaded when creating or editing a lecture resource and is stored locally on the server in the media/documents/ directory. I used a custom validator to only allow PDF files to be uploaded. My approach was to associate feedback to specific slides so there is a field for entering the number of slides a lecture has. It occurred to me that users may not know the slide count of their lecture so I added a functionality where users can have that information extracted from the PDF file they have uploaded. I use the python package pyPDF2 to extract this detail. At the start I extracted the title, author and page number but in the end only made use of the page number since that was the only thing users may be unsure of.

Lectures are listed after login on the ‘Lecture List’ page. On this page I have implemented pagination and a search functionality for ease of use. The lectures can be searched by either name or date. Search queries and current page are added to the URL as query strings for easy bookmarking and navigation.

## Feature Set 3 – Session Management

More focus on client-side heavy features requiring jQuery and AJAX.

I wanted to allow for each lecture to have multiple feedback sessions so created a Session model and linked it to the Lecture model. My idea was that feedback would not be provided to the lecture but to a feedback session run for that lecture, this way a lecture could be ran on multiple different occasions each having its own session.

I planned the ‘Lecture Detail’ page to be a form of control page where the staff users could view the lecture data and control sessions for that lecture. My layout for this page is shown below, left side has location of the session management buttons and the lecture details and right side is split with session list at top and question list for the selected session at bottom.

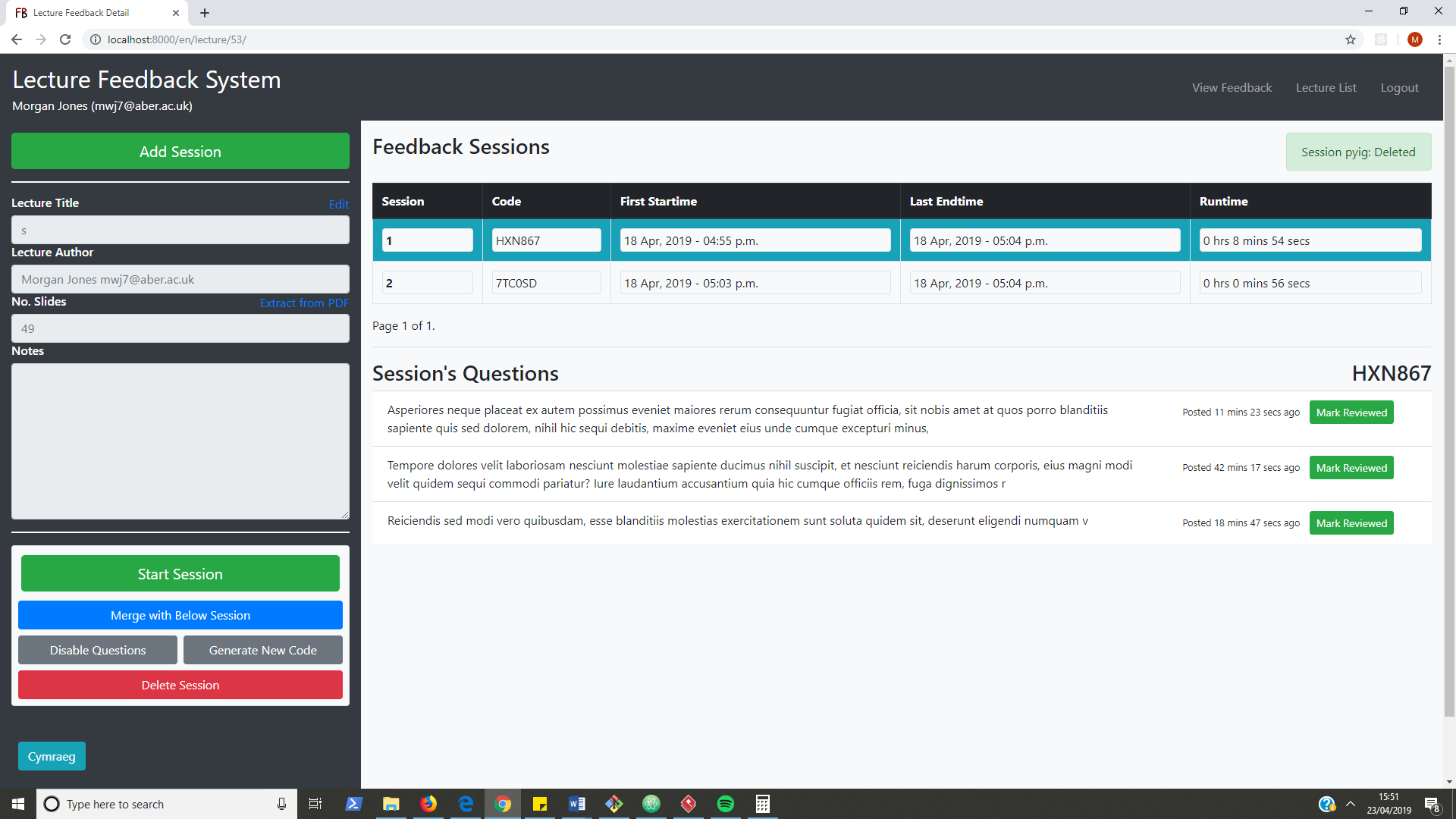


Figure 25 Lecture Detail Page Layout Example

At first I only allowed sessions to be created if there was not a session running and every session was automatically started upon creation. This approach meant only having one session running for one lecture at a time. I felt this was restrictive and my supervisor recommended providing flexibility in use when unsure of exactly what should be the user workflow. I changed the design so any number of sessions could be created and started or restarted at any time. The implementation of this required the addition of another relation to keep track of a session start times and end times which could then be used to sum the total runtime for a session.

The sessions are displayed in a sessions table which is paginated to show only five sessions at a time. I previous had questions paginated as well but as more functionality was added this became an issue so question pagination was removed. This will mean the page will get longer and although not too professional it isn’t really an issue because it is assumed a lecturer would answer the questions in the order they are asked (from the top down) anyway.

There are buttons to control sessions that are displayed on the right, when a session row is clicked on it is highlighted in blue and the options for that sessions are displayed. This functionality was implemented using the jQuery library to manipulate the page HTML by copying across hidden buttons in the table rows to a sperate <div> for display. I use JavaScript local storage to save which session is currently highlighted between page reloads.

I used jQuery to make AJAX calls to refresh session and question data every second without the user having to reload the page. I made sub templates in my Django app for the session list and the question list and included them in the lecture\_detail.html template. The AJAX calls would then query the HTML for these sub templates every second and replace the HTML content in the DOM. My AJAX calls return HTML where I would have ideally wanted to return JSON because this less data to transfer from the server. I did not use JSON because that would have required more code in the JavaScript, and I was struggling with maintaining the client-side code already.

Sessions display runtime and questions display time ago posted both of these are timedelta objects calculated in the model code. Django templates do not have python code embedded into the HTML because the template code should be about display not logic therefore to better represent the timedelta object I created custom template filters that I then registered for use in my templates. I also created a custom template tag that allowed continuous numbering of sessions (1st-nth) across my session list pagination.

I had an issue with users being able to access resources that do not belong to them. I solved this by writing custom decorators that I could attach to view functions to ensure that users could only access the lectures, sessions or questions that they owned. The decorators act as function wrappers that take in the function they are attached to as a parameter and only allow the function to be called if the target resource is owned by the logged in user. If not owned the decorators raise a PermissionDenied exception and a 403 error is returned to the client. I then went on to overwrite the templates used for 404 (page not found) and 403 (forbidden/permission denied) errors so users could easily navigate away from an error page on my application. These templates are the 404.html and 403.html files respectively.

To display feedback data on the feedback detail page I planned to use vanilla JavaScript to draw on the HTML canvas. When I came to implement this feature, I found an easy to use JavaScript library called chart.js that can build good quality responsive charts. I used this library to create the bar and pie charts in my application. Use of this library saved me having to write and test a lot of code.

Feedback data is provided to the feedback\_detail.html page through a jQuery AJAX call that interacts with an API on the server that returns subsets of the feedback data associated with the selected session as JSON. The JavaScript then destroys and recreates the charts using the new data. The collation of this feedback data on the server requires access across multiple feedback instances therefore I wrote a FeedbackManager class to extend the functionality of the default manager used by my Feedback model class. This is the preferred way to add table-level functionality to a Djano model; it also tidied up my model code by placing a relatively large method in its own class.

## Feature Set 4 - Providing Feedback

Focus on being simple, accessible by mobile and made use of Django session middleware.

Student users are not logged in so to implement state I made use of Django’s session middleware. There is a relation in the database holding each user’s session key, encrypted session data and the expiry date of the session. The three pieces of session data I stored for connected student users are described below:

* **connected\_session\_id**
  + Stores the id of the active session the student is connected to
  + Used to associate feedback with correct session
* **questions\_asked**
  + Stores a list of IDs of all the questions the student has asked
  + Used so students can view their asked questions and delete them
* **slides\_with\_feedback**
  + Stores a list of slide numbers for which the user has provided feedback
  + Used so that providing feedback to a slide more than once will overwrite the old feedback not keep adding new feedback (makes accidentally double clicking the submit button an idempotent operation)

Questions/Comments that are asked by students are listed so the student can delete them if they wish. This question list content reloads every second using AJAX this works the same way as implemented in the lecture\_detail.html page (described in the previous section).

When testing this feature set it was found that, although functioning well, the application was not notifying users very well. This was true for most of my application. To fix this Django’s messaging framework was used to easily provide application alerts between views. I replaced all the different template context variables that stored string alert messages with either success, warning or error messages provided to Django’s messaging class. The messages could then easily be displayed with the same section of reusable template code.

# Testing

## Overall Approach to Testing

A set of unit tests and test cases were built up throughout the project for testing the application. Individual user usability tests were run to gather feedback on the application. The plan was to do a full documented run through of all the test cases before usability tests and then a full run through of test cases after implementing the changes inspired by the usability test results.

## Unit Testing

Python code was unit tested according to Django’s built in testing framework. As is convention in Django unit tests are structured inside a /tests folder with each file of implementation code having a corresponding file of test code. The test file matches the name of the code file with ‘test\_’ prepended to the file name. Django’s unit tests use a Python standard library module: unittest. This module defines tests using a class-based approach therefore every class or function of code in my project has an associated unit test class to test it.

I ran the tests through the command line using the ‘python manage.py test’ command.

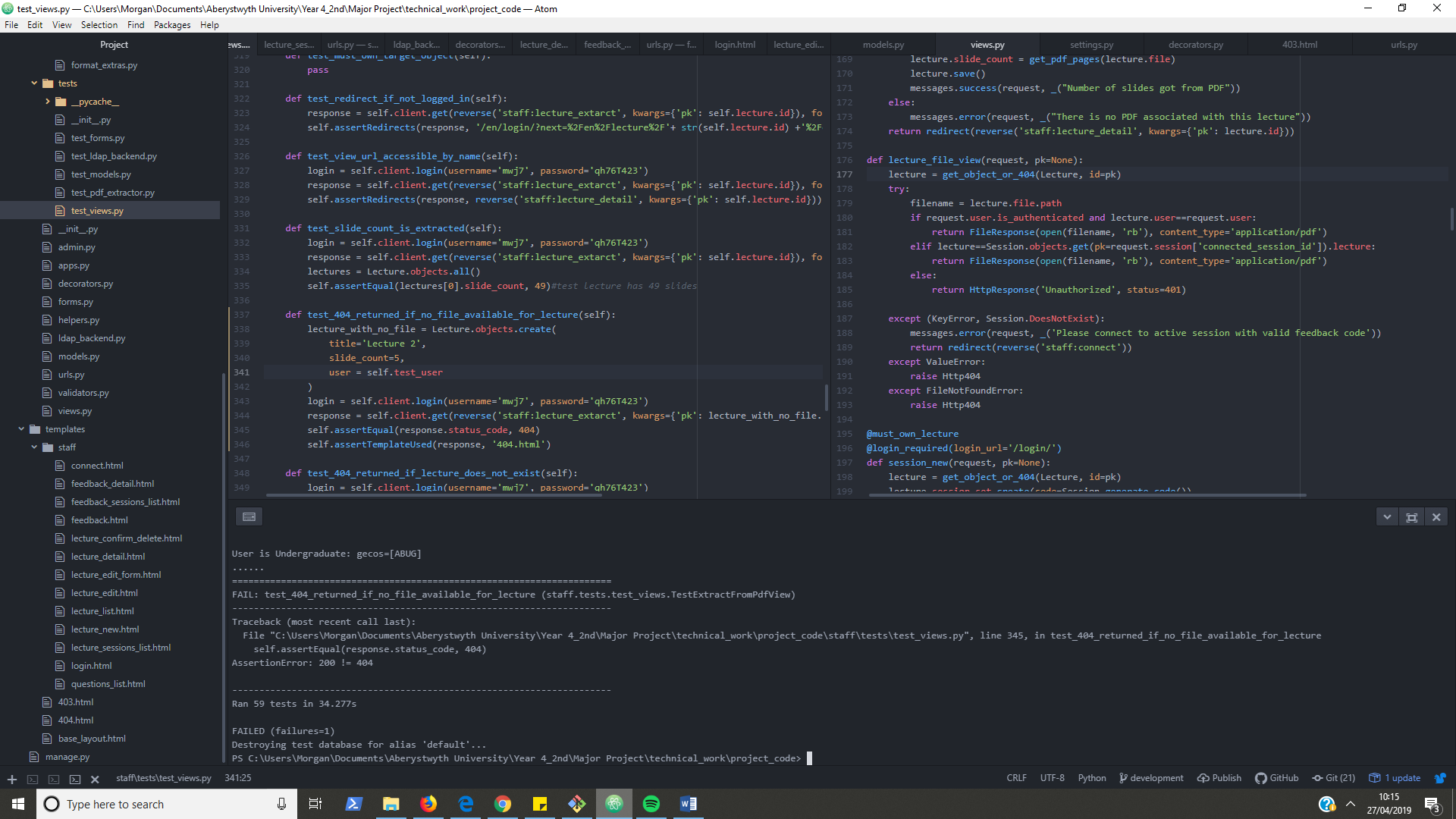


Figure 26 Example of unit testing project through command line while developing inside atom text editor

## System Testing

The test cases for this system are included in appendix A.D.

Every feature on the feature list has a corresponding test scenario; all test cases for a given scenario collectively test a feature has been implemented correctly. Testing is mapped to requirements by each feature in the feature list having a test scenario ID from the test case table.

## User Testing

I performed supervised user testing of the system with six individuals. The document for this testing is in the appendix A.E.

A list of tasks was performed by my volunteers some tasks as staff users and some tasks as student users. While the volunteers were performing user tasks I was acting as a staff member to control sessions. While the volunteers were performing staff tasks I was acting as a student user to provide feedback.

The tasks chosen were a selection of standard system use-cases. Every volunteer got the same tasks in the same order I did not vary the tasks between volunteers because I wanted to be able to easily compare and collate their feedback.

I had half my volunteers run through tasks using a desktop and half run through the tasks using a mobile. I didn’t have all volunteers use both mobile and desktop because once they had used the system once they would already know its layout and that felt like cheating.

The feedback for usability of the application was fairly consistent across all participants. Aside from the general need for more notification and clarity of the various system functions the least usable part of the system was the session management, when accessing through a mobile device workflow was not obvious and parts of the visual ques were hidden from view.

The easiest to use part of the application was the student feedback on both desktop and mobile with little to no issues being raised during those tasks. The changing of locale between Welsh and English was also found very easy by the users however none of my users were Welsh speakers so flaws may have been not pointed out if any were there.

There was a notable comment made by one of my volunteers during testing, he said that it would be better to move all controls to a top section that was consistent across the pages and the content below would change based on what you are working on. I would imagine this would be more like a single page application; something I was considering myself half way through the project.

# Critical Evaluation

## Technical Evaluation

### Requirements & Design Evaluation

Session and lecture being separate (should just be session) confuses users. this was a bad design decision.

More time should have been spent figuring the project requirements and what users would expect from such a system more research of similar style systems would have helped with this.

The high-level design for the system was, in my opinion, adequate. I do think that the low-level design was slightly lacking but that was due to my uncertainty with the convention when modelling a django project.

The layout of the UI was good for the simpler pages but the layout of the lecture\_detail.html page and the feedback\_detail.html page could have been much better I think more time should have been spent redesigning the UI for these pages because they have the most features in. The approach for building these pages was sort of ad-hock and the application could benefit from a more uniform layout of controls between both these pages; this was something suggested during user testing.

The layout for the page to provide feedback is simple and clear. I think it also works well mobile and the feedback I got from users was pretty good all round.

I wrote a function to merge two sessions allowing all questions and feedback for the sessions to be combined. In the current version of my system a session can only be merged with the immediately previous or next session this was by design originally but now sessions can be restarted so it makes more sense to allow merging of any arbitrary pair of sessions. I did not have enough time to make this update.

Improvement to feedback would be allow multiple session feedback to be viewed as one.

### Implementation Evaluation

On reflection I may have benefited from making use of a build tool during implementation to automate the process of readying the system for review. The below tasks are things I had to ensure were done before trying out the system:

* Make migrations
* Migrate database
* Make messages
* Translate messages (could be automated by using google translate API)
* Compile messages
* Run tests
* Run server

Automating some of the above would have added to the professionalism of the project and perhaps saved me some time.

### Testing Evaluation

Testing of the system was not extensive although a lot of the system is covered. I think if testing was made more a priority throughout perhaps through the adoption of a TDD strategy maybe this would not be the case.

User testing was very useful clearly highlighting the good and bad points of the UI layout. I did not have enough time to implement many changes as a result of these tests but if I did the system would probably be improved considerably. Next time I would ensure I allocated more time to make changes as result of this testing, I would also change the testing itself to use a larger set of tasks and included a few welsh speaking users.

## Work to Extend the Project

The addition of a desktop python program that could display feedback data to a lecturer without the browser having to be open may add value to the system. This program would make use of a web API of my application which could be implemented by use of the Django REST framework.

My program has feedback options that I have defined myself. This is limiting as the feedback that can be given is pre-restricted. It would be an improvement if users running feedback sessions could define their own feedback forms that the application would use allowing each member of staff to tailor feedback options to their individual needs.

## Alternate Approach If Redoing the Project

The most difficult aspect of my project was getting the functionality I wanted on the client side. I ran into multiple issues due to an increase in the amount of JavaScript and jQuery I had to add to the project. In the end I stripped away functionality to get it to a good enough working state.

These problems would not have been an issue if I had taken a different approach with the technologies I had used. I found myself trying to develop two pages that were themselves needing to be more like control panel style single page apps with heavy use of jQuery code and AJAX calls.

The functionality I wanted could be developed much easier if using a client-side view framework instead of fixed templates build on the server-side. I invested time during the project attempting to fix this by trying to learn and integrate react.js into my project. I found it difficult to get working and due to already falling behind my self-set targets decided to discard my changes and continue with my original plan of using Django templates.

If doing the project again I would almost certainly start using react.js integrated with Django from the get-go. I would convert the Django app into a pure web API using the Django REST framework to serialise all data into JSON; then use react.js as an API client to dynamically build and reload components of the UI.

In the final weeks of the project I have been reading about vue.js a client-side framework similar to react but far easier to get started with for beginners due to less initial configuration being required. If I had discovered it earlier, I could have added vue.js into my project for use on the two pages that are client-code heavy; which could have improved the quality of my project considerably.

## Overall Evaluation

I think the project went well overall. I had to refactor continuously throughout as I learnt better ways to implement things using more advanced topics of the Django framework. I learnt about problems I was not aware of at the start and ways to solve them using new techniques and technologies.

The project gave me more exposure to web development and in that respect was a success. In my opinion the application is functional as a web application and meets the requirements of a basic feedback system. I think the project in its current state would serve as a good starting point for developing a more robust and advanced system if I could start again taking what I have learned and my reflections into consideration.

As the project went on my focus on updating design, tests and balancing other aspects of the project decreased. This lack of discipline in my engineering process is something that makes the project weak when compared to the standard of a professional software project. I believe the decrease in discipline stems from an uncertainty with the requirements, technologies used to implement them and general lack of experience working on large projects. This is something that comes in time and this project has done nothing but help by providing me with a valuable learning experience.

# Annotated Bibliography

|  |  |
| --- | --- |
| [1] | “Ruby On rails Guides,” [Online]. Available: https://guides.rubyonrails.org/. [Accessed 26 January 2019]. |
| [2] | “Django documentation,” Django Software Foundation, [Online]. Available: https://docs.djangoproject.com/en/2.1/. [Accessed 26 January 2019]. |
| [3] | “Python Webstite,” The Python Software Foundation, [Online]. Available: https://www.python.org/. [Accessed 02 May 2019]. |
| [4] | “Sublime Text,” [Online]. Available: https://www.sublimetext.com/. [Accessed 22 April 2019]. |
| [5] | “Atom Homepage,” [Online]. Available: https://atom.io/. [Accessed 22 April 2019]. |
| [6] | “Atom Terminal-Plus,” [Online]. Available: https://atom.io/packages/terminal-plus. [Accessed 2019 January 30]. |
| [7] | “Atom Beautify,” [Online]. Available: https://atom.io/packages/atom-beautify. [Accessed 2019 April 22]. |
| [8] | “Bootstrap Documentation Introduction,” Bootstrap, [Online]. Available: https://getbootstrap.com/docs/4.0/getting-started/introduction/. [Accessed 31 January 2019]. |
| [9] | “MySQL Homepage,” Oracle Corporation, [Online]. Available: https://www.mysql.com/. [Accessed 02 May 2019]. |
| [10] | “SQLite Homepage,” [Online]. Available: https://www.sqlite.org/index.html. [Accessed 2 May 2019]. |
| [11] | “PostgreSQL 11.1 Documentation,” The PostgreSQL Global Development Group, [Online]. Available: https://www.postgresql.org/docs/11/index.html. [Accessed 26 January 2019]. |
| [12] | “The ldap3 Project,” [Online]. Available: https://ldap3.readthedocs.io. [Accessed 28 January 2019]. |
| [13] | “Documentation of pdb module,” [Online]. Available: https://docs.python.org/3/library/pdb.html. [Accessed 28 04 2019]. |
| [14] | “The Hitchhiker's Guide to Python,” [Online]. Available: https://docs.python-guide.org/dev/virtualenvs/. [Accessed 29 January 2019]. |
| [15] | “pip - The Python Package Installer,” The Python Software Foundation, [Online]. Available: https://pip.pypa.io/en/stable/. [Accessed 28 April 2019]. |
| [16] | “Cloud Translation API documentation,” Google, [Online]. Available: https://cloud.google.com/translate/docs/. [Accessed 01 05 2019]. |
| [17] | “The Django Project,” [Online]. Available: https://www.djangoproject.com/. [Accessed 2019 04 27]. |
| [18] | “Get Bootstrap,” [Online]. Available: https://getbootstrap.com/. [Accessed 27 April 2019]. |
| [19] | “jQuery,” The jQuery Foundation, [Online]. Available: https://jquery.com/. [Accessed 28 April 2019]. |
| [20] | “django-autofixture 0.12.1,” The Python Software Foundation, [Online]. Available: https://pypi.org/project/django-autofixture/. [Accessed 28 April 2019]. |
| [21] | “django-bootstrap3,” The Python Software Foundation, [Online]. Available: https://pypi.org/project/django-bootstrap3/. [Accessed 28 April 2019]. |
| [22] | “ldap3 2.6,” The Python Software Foundation, [Online]. Available: https://pypi.org/project/ldap3/. [Accessed 28 April 2019]. |
| [23] | “psycopg2 2.8.2,” The Python Software Foundation, [Online]. Available: https://pypi.org/project/psycopg2/. [Accessed 28 April 2019]. |
| [24] | “PyPDF2 1.26.0,” The Python Software Foundation, [Online]. Available: https://pypi.org/project/PyPDF2/. [Accessed 28 April 2019]. |
| [25] | “Chart.js on GitHub,” [Online]. Available: https://github.com/chartjs/Chart.js. [Accessed 28 April 2019]. |

# Appendices

* 1. Third-Party Code and Libraries

**Django** – This is the framework I have used to build my application. It is open source and available from the Django Software Foundation [10]. The framework is released using a BSD license. The framework was used without modification.

**Bootstrap** – This is a CSS framework used to add style and responsive design to my application. It is open source and available the Bootstrap Core Team [12]. The framework is released under the MIT License. The framework was used without modification.

**jQuery** – This is a JavaScript Library used for DOM manipulation and event handling on the client-side. It is open source and available from the jQuery Foundation [15]. The library is released using the MIT License. The library was used without modification.

**Django-autofixture** – This is a python package written by Gregor Müllegger for use with the Django framework that allows me to generate random data which I used while testing the system. It is open source and available from the Python Software Foundation [16]. The library is released using the BSD License. The library was used without modification.

**Django-bootstrap3**– This is a python package written by Dylan Verheul for use with the Django framework that allows bootstrap to work better with Django. It is open source and available from the Python Software Foundation [16]. The library is released using the BSD License. The library was used without modification.

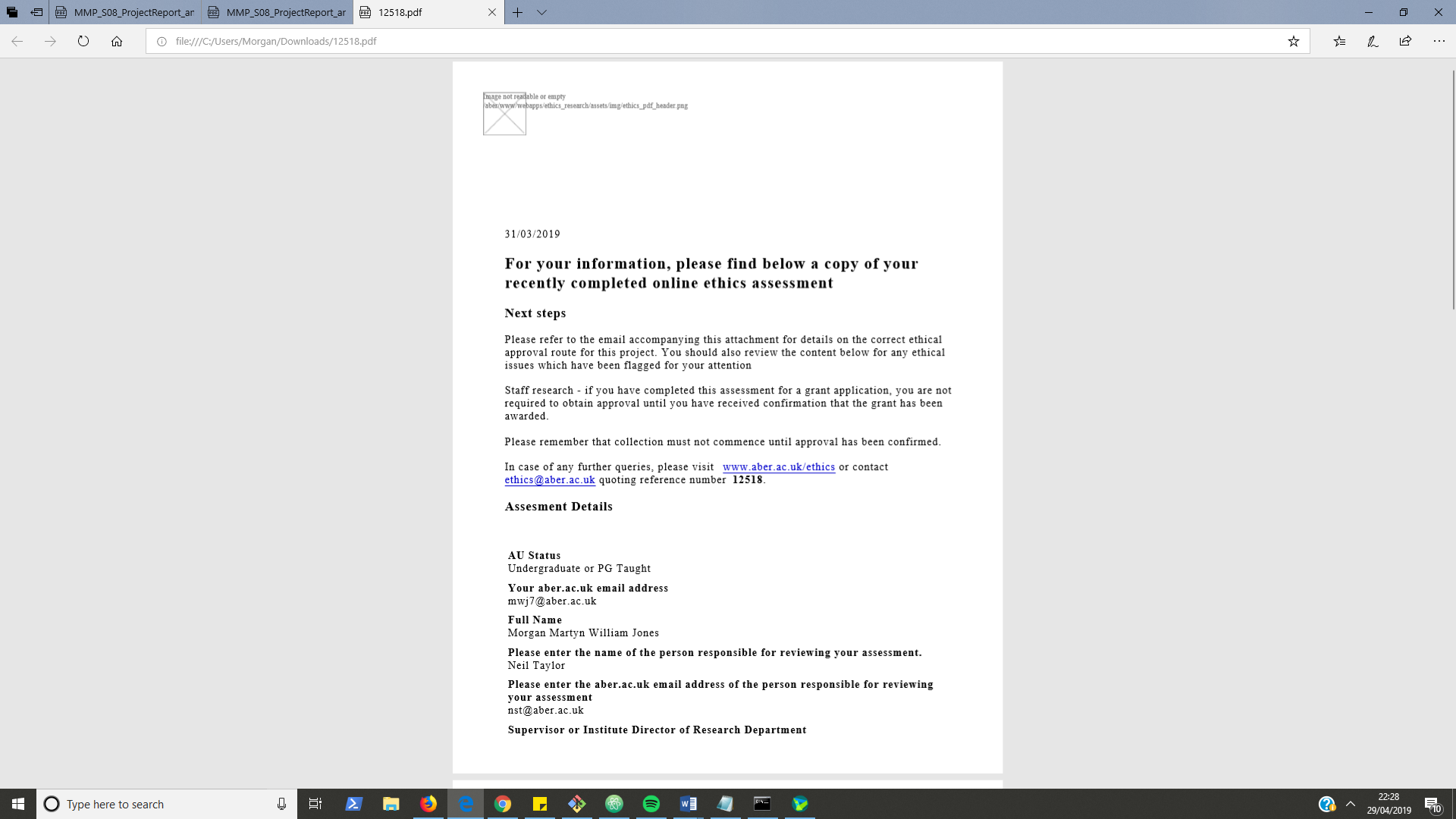
**Ldap3**– This is a python package written by Giovanni Cannata it provides a ldap API to my system that is needed for authentication. It is open source and available from the Python Software Foundation [16]. The library is released using the GNU Lesser General Public License. The library was used without modification.

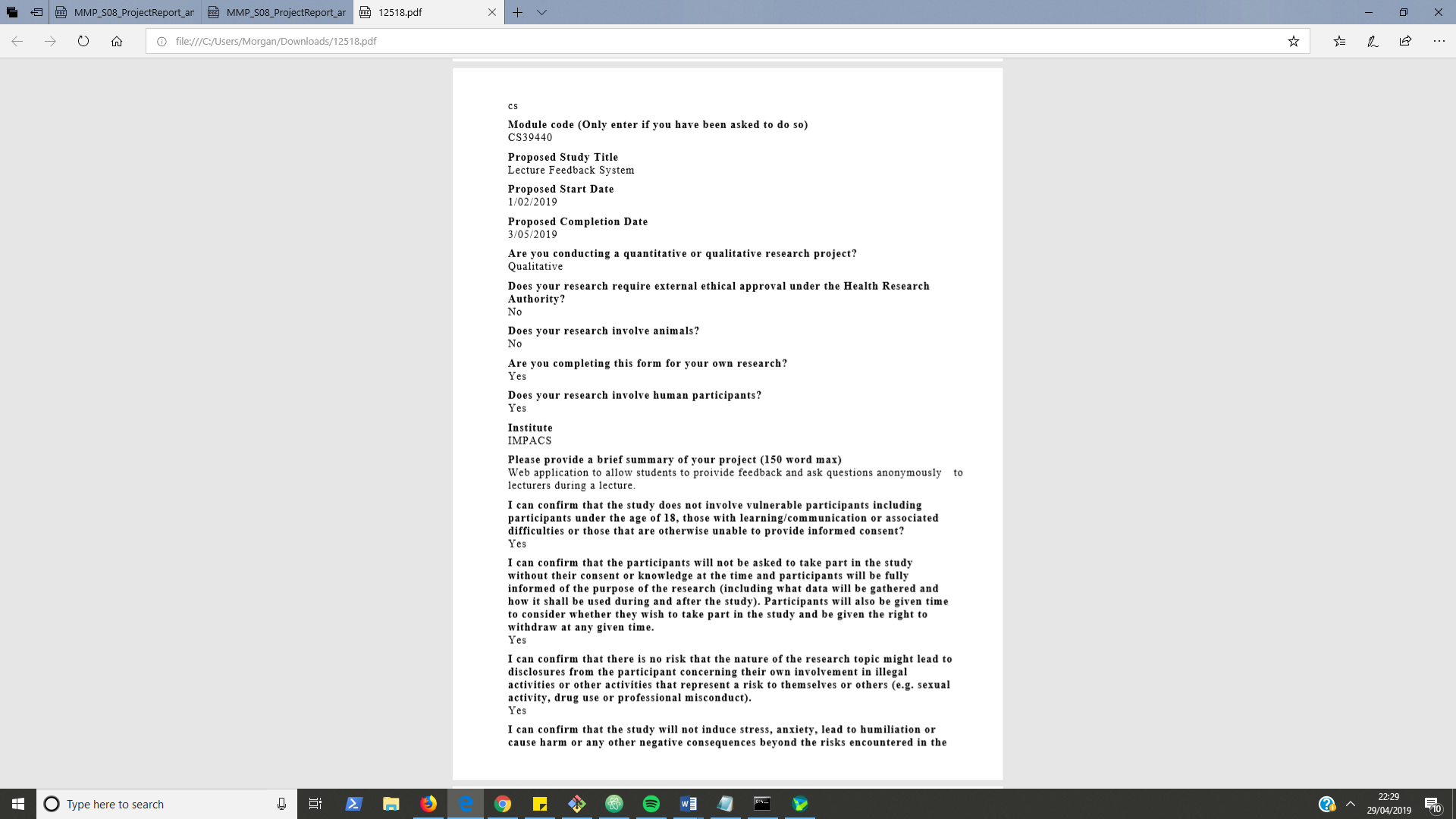
**Psycopg2** – This is a python package written by Federico Di Gregorio it is PostgreSQL database adapter that allows my application to use a PostgreSQL database. It is open source and available from the Python Software Foundation [16]. The library is released using the GNU Library or Lesser General Public License. The library was used without modification.

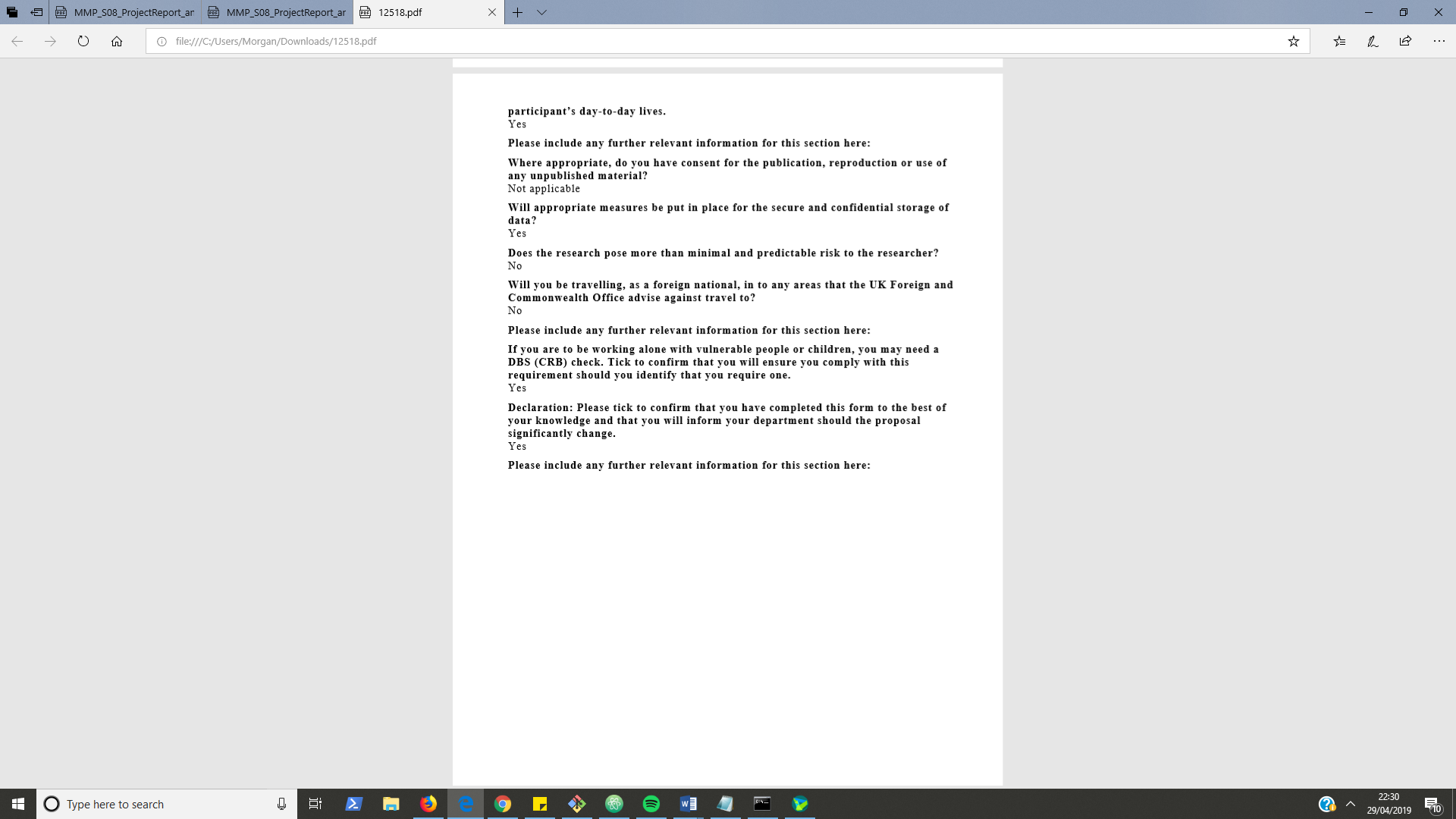
**PyPDF2** – This is a python package used in my application to extract meta data from uploaded pdf files. It is open source and available from the Python Software Foundation [16]. The library is released using the BSD License. The library was used without modification.

**Chart.js** – This is a JavaScript library used to create responsive pie and bar charts in my application. It is open source and available from GitHub [15]. The library is available under the MIT License. The library was used without modification.

* 1. Ethics Submission







* 1. Feature List

|  |  |  |
| --- | --- | --- |
| **Feature** | **Done?** | **Test Scenario** |

**Authenticating User – Feature Set – 1 Week – (20th-27th February)**

|  |  |  |
| --- | --- | --- |
| Login form for staff with authentication through use of LDAP server via python LDAP API | Done | TS01 |
| Add I18n with Welsh localisation | Done | TS02 |

**Lecture Management – FS– 2 Weeks – (27th-13th)**

|  |  |  |
| --- | --- | --- |
| Create new lecture | Done | TS03 |
| List & delete any/all lectures   * Only those created by the currently logged in user | Done | TS04 |
| View each lecture data | Done | TS05 |
| Edit each lecture data | Done | TS06 |
| Search the list of lectures for a lecture with a specific title or date | Done | TS07 |
| Have download PDF functionality for students and staff on the app | Done | TS08 |
| Extract the meta-data of a lecture/workshop   * Number of Slides | Done | TS09 |

**Session Management – FS – 1 week – (13th – 22nd March)**

|  |  |  |
| --- | --- | --- |
| Start a feedback session for a lecture | Done | TS10 |
| Generate the 6-digit code for a session | Done | TS11 |
| View the questions of a session as they are asked | Done | TS12 |
| Stop a feedback session for a lecture | Done | TS13 |
| Delete a feedback session | Done | TS14 |
| Mark an asked Question as Answered | Done | TS15 |
| Toggle Questions for a session | Done | TS16 |
| Merge a session with the immediately previous session | Done | TS17 |
| Merge a session with the immediately next session | Done | TS18 |
| Use AJAX to update page data with having to reload page | Done | ------ |
| Graphically display the feedback of a session using JavaScript and canvas on separate page | Done | TS19 |

**Providing Feedback - FS – 2 weeks– user Testing Upon Completion**

|  |  |  |
| --- | --- | --- |
| Connect the users to a session   * Only if they have the session code and the session is active | Done | TS20 |
| Associate Feedback to a specific lecture slide or provide overall feedback | Done | TS21 |
| Change feedback for a specific lecture slide | Done | TS22 |
| Ask a question to the person running the session | Done | TS23 |
| Delete a previously asked question | Done | TS24 |
|  |  |  |

* 1. Test Cases

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | **Test Scenario** | **Test Case ID** | **Test Case Name** | **Test Steps** | **Test Data** | **Expected Outcome** | **Actual Outcome** | **Result** |
| TS01 | Check Staff Login | TC01 | Check staff login with correct data | 1. Go to <http://localhost:8000/login/> 2. Enter Username 3. Enter Password 4. Click Login | username = mwj7  password = <my\_password>  *(Currently allowing me to login as staff)* | User should be redirected to Lecture List page (<http://localhost:8000/en/lectures/>) User name and email should be displayed in the left of the navigation bar | As Expected | Pass |
|  |  | TC02 | Check staff login with invalid data | 1. Go to <http://localhost:8000/login/> 2. Enter Username 3. Enter Password 4. Click Login | username = mwj7  password = wrong123 | User should be redirected <http://localhost:8000/en/login/> and there should be a red message box appear below the login form with the text “Invalid Staff Login Details” | As Expected | Pass |
|  |  | TC03 | Check undergraduate valid data cannot login | 1. Go to <http://localhost:8000/login/> 2. Enter Username 3. Enter Password 4. Click Login | username = mwj7  password = <my\_password> | User should be redirected <http://localhost:8000/en/login/> and there should be a red message box appear below the login form with the text “Invalid Staff Login Details” | As Expected | Pass |
|  |  | TC04 | Check login with required data missing | 1. Go to <http://localhost:8000/login/> 2. Miss out either or both the username and passwords fields 3. Click Login | N/A | Should display prompt next to first empty field with text “Please fill in this field” | As Expected | Pass |
|  |  | TC05 | Check Redirect to login page when attempting URL navigation to another part of staff site without being logged in | 1. Go to any or all of the possible site URLs | See Possible Site URLs above | Should redirect to <http://localhost:8000/en/login/> with a GET parameter next in the URL set to the URL you attempted to navigate to. | As Expected | Pass |
|  |  |  | Check alerts user when trying to login if already logged in | 1. Login as described in TC01 2. Navigate to URL /login/ | N/A | * Should display blue alert with text “You are already logged in” * Should also display “Lecture List” and “Logout” links in the right side of the navigation bar. * Should display name and email at the left side of the navigation bar | As Expected | Pass |
|  |  | TC06 | Check logout | 1. Login as described in TC01 2. Click ‘Logout’ in the navigation bar | N/A | Should Redirect to /en/login/ page and no longer display name and email in the navigation bar. | As Expected | Pass |
|  |  | TC07 | Check the if being redirected to the login page that the ‘next’ GET parameter works in redirecting the user to the page they first tried to access | 1. Enter /en/lectures/ in the URL bar 2. Enter Username 3. Enter Password 4. Click Login | Username = mwj7  Password = <my\_password> | Should be redirected to Lecture List page (/en/lectures/) and logged in with name and email displayed in the left of the navigation bar. | As Expected | Pass |
| TS02 | Check welsh language localisation | TC08 | Check site displays in Welsh | 1. Go to any or all of the possible site URLs 2. Click on Cymraeg in the bottom left corner of the screen | See Possible Site URLs above | Redirects to the same URL with /cy/ prepended on the URL  All static display text should be in Welsh | As Expected | Pass |
|  |  | TC09 | Check I18n URLs work | 1. Go to any or all of the possible site URLs with /en/ appended to URL 2. Go to any or all of the possible site URLs appended with /cy/ | See possible Site URLs above | * All static text is displayed in English for /en/ URLs * All static text is displayed in Welsh for /cy/ URLs | As Expected | Pass |
| TS03 | Check create lecture | TC10 | Check create new lecture with valid data | 1. Login as described in TC01 2. Click on ‘New Lecture’ in the navigation bar 3. Enter title 4. Enter Number of slides 5. Enter Notes 6. Click Choose and select the test lecture 7. Click Create | title = My Lecture  number of slides = 15  notes = <text>  See test lecture location above. | * Redirect to Lecture List Page (/en/lectures/). * New lecture ‘My Lecture’ will be added to list. With Creation Date set to today’s date. * There will be a green success message in the top right corner with the text “Lecture Created: My Lecture” * When Navigating clicking on lecture title to navigate to ‘Lecture Detail’ page the info in the fields in the left panel will be the same as that entered in the create form. * When then clicking on ‘View Lecture PDF’ in the navigation bar the PDF will be displayed in the browser. | As Expected | Pass |
|  |  | TC11 | Check create new lecture with valid data and optional data missed out | 1. Login as described in TC01 2. Click on ‘New Lecture’ in the navigation bar 3. Enter title 4. Enter Number of slides 5. Click Create | title = My Lecture  number of slides = 15 | * Redirect to Lecture List Page (/en/lectures/). * New lecture ‘My Lecture’ will be added to list. With Creation Date set to today’s date. * There will be a green success message in the top right corner with the text “Lecture Created: My Lecture” * When Navigating clicking on lecture title to navigate to ‘Lecture Detail’ page the info in the fields in the left panel will be the same as that entered in the create form. * The ‘View Lecture PDF’ navigation link will NOT be there because there is no uploaded file to link to | As Expected | Pass |
|  |  | TC12 | Check create lecture with invalid data | 1. Login (described in TC01) 2. Click on ‘New Lecture’ in the navigation bar 3. Enter title 4. Enter Number of slides 5. Click Create | title = My Lecture  number of slides = eeee | Displays a prompt with text “Please enter a number” on the number of slides field | As Expected | Pass |
|  |  | TC13 | Check create lecture with missing required data | 1. Login (described in TC01) 2. Click on ‘New Lecture’ in the navigation bar 3. Miss out either or both of the title and number of slides fields 4. Click Create | N/A | Displays a prompt next to the first missing field with text “Please fill in this field” | As Expected | Pass |
|  |  | TC14 | Check only PDF file uploads are accepted. | 1. Login (described in TC01) 2. Click on ‘New Lecture’ in the navigation bar 3. Enter title 4. Enter Number of slides 5. Enter Notes 6. Click Choose and select any file that is not a PDF   Click Create | title = My Lecture  number of slides = 15  notes = <text> | Should redisplay the create form with the added text ‘Unsupported file extension: PDF Only.’.  Should also display a red alert below the form saying the form data is invalid | As Expected | Pass |
| TS04 | Check lecture listing & deletion | TC15 | Check Lecture delete | 1. Create Lecture (Described in TC10) 2. Click on Delete for the lecture just created 3. Click on Confirm | N/A | Should Add a lecture to the list. When clicking on Delete should navigate to ‘Confirm Delete’ page.  Clicking on Confirm should navigate back to lecture list with the lecture gone from the list. There will be a green success message in the top right corner with the text “Lecture Deleted: My Lecture” | As Expected | Pass |
|  |  | TC16 | Check lecture confirm delete | 1. Create Lecture (Described in TC10) 2. Click on Delete for the created lecture 3. Navigate to the previous page with the back button | N/A | Should add a lecture to the list. When clicking on Delete should navigate to ‘Confirm Delete’ page. Clicking on back button will navigate back to lecture list where the lecture should still be in the list. | As Expected | Pass |
|  |  | TC17 | Check message display if no lectures present | 1. Login (described in TC01) 2. If any lectures are in list delete them following the process described in TC13 | N/A | There should be a grey alert box displayed with the text “No Lectures”. | As Expected | Pass |
|  |  | TC18 | Check lecture list pagination | 1. Ensure there is enough lectures to require pagination by creating a minimum of 9 new lectures following the process described in TC10 2. Click on any or all of the pagination links below the lecture list | N/A | No more than 8 lectures should be on any pagination page. | As Expected | Pass |
|  |  | TC19 | Check lecture ordering | 1. Create multiple lectures (Described in TC10) 2. Delete some of them (Described in TC13) | N/A | Lectures should be ordered with the most recently created being at the top | As Expected | Pass |
| TS05 | Check view lecture data | TC20 | Check able to view lecture data | 1. Create a lecture (Described in TC10) 2. Click on title of the lecture you just created in the lecture list | N/A | Should be on ‘Lecture Detail’ page  On the left side of the screen there should be a section with a dark background that contains the values for the lecture’s title, slide count and notes; these values should match the ones you entered upon lecture creation. | As Expected | Pass |
| TS06 | Check edit lecture data | TC21 | Check displays correct data in Edit Lecture page | 1. Login, create and view a lectures data as described in TC18 2. Click on the ‘Edit’ link above the lecture title field | N/A | Should be on ‘Edit Lecture’ page with a form similar to the one displayed on the ‘Create Lecture’ page however all three fields should be filled in with the values previously viewed on the ‘Lecture Detail’ page. | As Expected | Pass |
|  |  | TC22 | Check lecture details are updated correctly when providing correct data | 1. Go to the ‘Edit Lecture’ page (Described in TC19) 2. Enter new\_title into title field 3. Enter new\_notes into notes field 4. Click Update | new\_title = Awesome New Title  new\_notes = Awesome New Notes | Should be on ‘Lecture Detail’ page with new title and notes displayed in the fields in the panel on the left side of page. | As Expected | Pass |
|  |  | TC23 | Check lecture details cannot be submitted if providing invalid data | 1. Go to the ‘Edit Lecture’ page (Described in TC19) 2. Provide incorrect data or invalid file to the lecture details form (like TC12 & TC14) | N/A | Should see same result for missing data, invalid data or wrong file type as described above in TC?? & TC??  (Missing data prompt user to fill in field, Invalid data reloads form with red alert and wrong file type reloads form and display unsupported file type notification*.*) | As Expected | Pass |
| TS07 | Check Lecture Search | TC24 | Check Search Lecture by full title | 1. Login & Create a lecture (Described in TC10) 2. Create two more lectures with titles provided 3. Type ‘Second Lecture’ into the search field 4. Click ‘Search’ | title(2) = ‘Second Lecture’  title(3) = ‘Third Lecture’ | Only the lecture titled ‘Second Lecture’ should be displayed in the list. | As Expected | Pass |
|  |  | TC25 | Check Search Lecture by partial title | 1. Login & Create a lecture (Described in TC10) 2. Create two more lectures with titles provided 3. Type ‘Lecture’ into the search field 4. Click ‘Search’ | title(2) = ‘Second Lecture’  title(3) = ‘Third Lecture’ | All three lectures should be displayed in the list (My Lecture, Second Lecture, Third Lecture) | As Expected | Pass |
|  |  | TC26 | Check Lecture Search by full Date | 1. Login & Create a lecture (Described in TC10) 2. Type the date of creation into the search field 3. Click “Search” | N/A | Only the lectures created on the entered date should be shown | As Expected | Pass |
|  |  | TC27 | Check Empty Search displays all lectures | 1. Ensure there are some lectures in the lecture list 2. Ensure the search field is empty 3. Click ‘Search’ | N/A | All lectures that have been created by the user should be displayed. | As Expected | Pass |
| TS08 | Check PDF download functionality | TC28 | Check staff user can view and download the uploaded PDF file | 1. Create a lecture with a PDF file (Described in TC14) 2. Click on the lecture’s title in the lecture list to go to the ‘Lecture Detail’ page 3. Click on ‘View Lecture PDF’ in the navigation bar on ‘Lecture Detail’ page 4. Click on the Download button on this page | N/A | * After clicking on ‘View Lecture PDF’ the PDF should be displayed in the browser. * Clicking Download will download the pdf into the downloads folder on computer | As Expected | Pass |
| TS09 | Check extract from PDF functionality | TC29 | Check slide count can be extracted from uploaded PDF file | 1. Create a lecture with a PDF file (Described in TC14) 2. Click on the lecture’s title on the ‘Lecture List’ page to go to the ‘Lecture Detail’ page 3. Click on ‘Extract from PDF’ | N/A | Slide count displayed for the lecture should now be the same as the number of PDF pages/slides. | As Expected | Pass |
| TS10 | Start a feedback session | TC30 | Check session can be created | 1. Login, create lecture and navigate to ‘Lecture Detail’ page (Described in TC20) 2. Click ‘Create Feedback Session’ | N/A | Session should be displayed as table row under feedback session heading in ‘Lecture Detail’ page.  The first start time of the session should be ‘N/A’  The last end time of the session should be ‘N/A’ | As Expected | Pass |
|  |  | TC31 | Check session can be started | 1. Create session (Described in TC30) 2. Click on session row in table 3. Click on ‘Start Session’ | N/A | First start time of session should change to when ‘Start Session’ was clicked  Last end time of session should change to ‘Session Running’ | As Expected | Pass |
|  |  | TC32 | Check Session can be stopped | 1. Create Session & Start Session (Described in TC30 & TC 31) 2. Click on session row in table 3. Click on ‘End Session’ | N/A | Last end time of session should change to when ‘End Session’ was clicked. | As Expected | Pass |
|  |  | TC33 | Check session can be restarted | 1. Create, start and stop session (TC30, TC31, TC32) 2. Ensure session row is highlighted by clicking on session row 3. Click ‘Start Session’ | N/A | Last end time of session should change to ‘Session Running’  When hovering over the first start time value of the session there should be popup displaying both times the session was started one will be the time when ‘Start Session’ was just clicked. | As Expected | Pass |
| TS11 | Generate Session Code | TC34 | Check session is created with valid code | 1. Create session (Described in TC30) | N/A | Session should be displayed as table row and entry for ‘Code’ column should be a 6-character uppercase Alphanumeric value. | As Expected | Pass |
|  |  | TC35 | Check code can be changed | 1. Create session (Described in TC30) 2. Click on the row containing that session in the table 3. Click Generate New Code | N/A | Session should display a different code than before. | As Expected | Pass |
| TS12 | View questions asked to current session | TC36 | Check un reviewed questions can be viewed on the lecture detail page | 1. Start a Feedback Session for the lecture (Described in TC31) 2. Connect to the session just started (Described in TC??) 3. Post a question to the session (Described in TC??) 4. Go to the ‘Lecture Detail’ page for the lecture the session was created for. 5. Click on the row for the session | N/A | The session’s question list on the lecture detail page should update to display the new question, with the time it was posted. | As Expected | Pass |
| TS14 | Stop Session Running | TC37 | Check running session can be stopped | 1. Navigate to a lecture with a running session or start a new session (Described in TC??) 2. Click on the row for that session in the Feedback Sessions table 3. Click on ‘Stop Session’ | N/A | Session should be displayed as table row under feedback session heading in ‘Lecture Detail’ page.  The start time of the session should be the time of session start and the end time should be the time that ‘Stop Session’ was clicked. | As Expected | Pass |
| TS15 | Mark a question as reviewed | TC38 | Mark an asked question as reviewed | 1. Create a session and start it (Described in TC31) 2. On another tab, connect to the session and ask a question to the session (Described in TC56) 3. Go back to original tab to view the session in the ‘Lecture Detail’ page 4. Ensure the session row is highlighted blue by clicking on it 5. Click ‘Mark Reviewed’ next to the asked question in the Session’s Questions list | N/A | Question should be removed from list under Session’s Questions heading on the ‘Lecture Detail’ page. | As Expected | Pass |
| TS16 | Toggle question for a session | TC39 | Disable questions for a session | 1. Create a session and start it (Described in TC31) 2. Click on disable questions on the session action button panel 3. On another tab, connect to the session | N/A | When you click disable questions you should see the grey text “[Disabled]” next to the Session Question’s heading  When connected to the session you should see an alert telling you questions are disabled | As Expected | Pass |
|  |  | TC40 | Re-Enable questions for a session | 1. Create a session and start it (Described in TC31) 2. Click on disable questions on the session action button panel 3. On another tab, connect to the session 4. Go back to previous tab on ‘Lecture Detail’ page and click enable questions | N/A | When you click enable questions you should see the grey text “[Disabled]” next to the Session disapear | As Expected | Pass |
| TC17 | Merge session with previous session | TC41 | Check can merge when there is a previous session | 1. Create two sessions 2. Click on one of them | N/A | There will be a blue button to merge the first one with the one below or a blue button to merge the last one with the one above. | As Expected | Pass |
|  |  | TC42 | Check cannot merge when there is not a previous session | 1. Ensure you only have one session created for a lecture 2. Click on the session row for that session | N/A | There should be no option to merge it in the action panel once it is clicked on because there is only one session there | As Expected | Pass |
| TS18 | Merge session with next session | TC43 | Check can merge when there is a next session to merge with | 1. Create two sessions 2. Click on one of them | N/A | There will be a blue button to merge the first one with the one below or a blue button to merge the last one with the one above. | As Expected | Pass |
|  |  | TC44 | Check cannot merge when there is not a next session to merge with | 1. Ensure you only have one session created for a lecture 2. Click on the session row for that session | N/A | There should be no option to merge it in the action panel once it is clicked on because there is only one session there | As Expected | Pass |
| TS19 | View session feedback in graphs | TC45 | Check can view all feedback for a session | 1. Create session and start it (Described in TC31) 2. Connect to the running session (Described in TC50) 3. Submit feedback to the session. 4. Go to the ‘Feedback Detail’ page for the active session 5. Click on the session in the list | N/A | The feedback options entered should be displayed in pie charts on the ‘Feedback Detail’ page  The change in options and amount of feedback should be the same as was entered. | As Expected | Pass |
|  |  | TC46 | Check changes in feedback automatically updates on the ‘Feedback Detail’ page | 1. Ensure a session is running and has feedback already (Described in TC54) 2. Connect to the session and submit some more feedback to the session on another tab in the browser 3. Go back to the original browser tab to view the ‘Feedback Detail’ page again (without reloading the page) | N/A | The added feedback should be noticed as a change in numbers of the pie charts or bar graphs being displayed on the ‘Feedback Detail’ page | As Expected | Pass |
|  |  | TC47 | Check can switch between viewing data as pie or bar charts | 1. Submit feedback for a session and view all feedback for a session (Described in TC45) 2. Change the value of the drop down select from ‘Pie Charts’ -> ‘Bar Charts’ (or Visa-Versa) | N/A | Upon switching the drop-down the displayed data should change to be displayed as the other type of chart; the amount and type of data itself should not change.  The new charts will replace the positions of the old charts. | As Expected | Pass |
|  |  | TC48 | Check can view session feedback for a specific lecture slide | 1. Ensure there is a session running (Described in TC31) 2. Ensure the lecture that the session is for has a slide count more than 1 3. Connect to the session on a different browser tab (/connect) by entering the session code (Described in TC50) 4. Select arbitrary feedback options and choose the ‘Slide 1’ option for the ‘Feedback for which slide’ field. 5. Click ‘Submit’ to submit the feedback 6. Select any feedback options but choose the ‘Slide 2’ option for the ‘Feedback for which slide’ field. 7. Click ‘Submit’ to submit the feedback 8. On previous browser tab go to ‘Feedback Detail’ page for the associated lecture 9. Change the drop-down options from ‘All Feedback’ to ‘Slide 1 Feedback Only’ 10. Change the drop-down options from ‘Slide 1 Feedback Only’ to ‘Slide 2 Feedback Only’ | N/A | * The submitted feedback data will show collectively in pie charts on the ‘Feedback Detail’ page when ‘All Feedback’ is the value of the drop-down select. * When changing the drop-down to ‘Slide 1 Feedback Only’ only the submitted feedback entry that was associated with slide 1 should be displayed. * When changing the drop-down to ‘Slide 2 Feedback Only’ only the submitted feedback entry that was associated with slide 2 should be displayed in the pie chart data. | As Expected | Pass |
|  |  | TC49 | Check can switch between viewing feedback data for different sessions | 1. Create a session and set it running (Described in TC31) 2. Create another session and set it running 3. Submit feedback to that session (Described in TC54) 4. Submit feedback to the second session 5. Go to the ‘Feedback Detail’ page for the lecture the sessions belong to 6. Click on the first session to view its feedback 7. Click on the second session to change to view its feedback | N/A | When clicking on first session its feedback data should be displayed in the charts  When clicking on the second the session its feedback data should be displayed in the charts | As Expected | Pass |
| TS20 | Check connect to session | TC50 | Check users can connect to active session | 1. Start a session (Described in TC31) 2. Go to /connect 3. Enter the sessions code in the Session Code field 4. Click ‘Connect’ | N/A | Should be redirected to ‘Lecture Feedback’ page. With the session code displayed in the top left of the page. | As Expected | Pass |
|  |  | TC51 | Check users cannot connect to inactive session | 1. Start a session (Described in TC31) 2. Stop the session by clicking the ‘End Running Session’ button 3. Go to /connect 4. Enter the sessions code in the Session Code field 5. Click ‘Connect’ | N/A | Page should reload with red alert under form with the text ‘Lecture is not active’ | As Expected | Pass |
|  |  | TC52 | Check users are disconnected when session is ended | 1. Start a session (Described in TC31) 2. Go to /connect 3. Enter the sessions code in the Session Code field 4. Click ‘Connect’ 5. Go back on different tab to ‘Lecture Detail’ page and end the session by clicking ‘End Running Session’ 6. Go back to other tab and reload the ‘Feedback’ page | N/A | Should be redirected to the ‘Connect’ page with a red alert with the text ‘Lecture is not active’ | As Expected | Pass |
|  |  | TC53 | Check session disconnect | 1. Create a session and connect to session (Described in T50?) 2. Click ‘Disconnect’ in the navigation bar | N/A | Should be back on connect page (/connect)  Attempted navigation to /feedback should result in ‘Connect’ page redirect with red alert with text ‘Please connect to active session with valid feedback code’ | As Expected | Pass |
| TS21 | Provide feedback | TC54 | Check feedback can be submitted to a session | 1. Create a session and start it (Described in TC31) 2. Connect to the session on a different browser tab (Described in TC50) 3. Choose options for each of the drop-downs under the ‘Give Feedback’ heading. 4. Click ‘Submit’ 5. Navigate to the ‘Feedback Detail’ page on the previous tab | N/A | When submitting feedback a green notification should be displayed underneath he Submit button saying the feedback has been submitted.  Feedback entered should match that displayed in the graphs on the ‘Feedback Detail’ page | As Expected | Pass |
| TS22 | Change provided feedback | TC55 | Check previously submitted feedback can be overwritten by user who submitted it | 1. Submit feedback to a session (Described in TC54) but associate the feedback with Slide 1 by selecting ‘Slide 1’ in the last drop-down. 2. Then change some of the options but still associate the feedback with ‘Slide 1’ | Feedback for which slide option = Slide 1 | When submitting the feedback for slide 1 the first time a green alert with the text ‘Slide 1 Feedback Submitted’ should be displayed. When submitting different feedback for slide 1 a green alert with the text ‘Slide 1 Feedback Re-submitted’ should be displayed.  Only the values of the last feedback submission for slide 1 should be displayed in the ‘Feedback Detail’ charts (Assuming no other feedback was previously provided for the session). | As Expected | Pass |
| TS23 | Ask a question to a session | TC56 | Check can post a session | 1. Start a session (Described in TC31) 2. Go to /connect 3. Enter the sessions code in the Session Code field 4. Click ‘Connect’ 5. Enter a question in the ‘Ask Question? ‘text area 6. Click ‘Post’ | question = Why is the sky blue? | Question should be displayed in a list group below the input text area with the time ago posted displayed.  If navigating back to the ‘Lecture Detail’ page that contains the session the question should also appear there under ‘Session’s Questions’ area of the page (assuming the corresponding session row is highlighted) with the same time posted ago. | As Expected | Pass |
| TS24 | Delete a previously asked question | TC57 | Check can delete a posted session | 1. Ask a question to a session (Described in TC56) 2. Ensure you are on the ‘Feedback’ page where you can view your asked question in the list under the Ask Question? text area 3. Click ‘Delete’ button next to the asked question | N/A | Question should be removed from list. Question should also be removed from listing in the corresponding ‘Lecture Detail’ page. | As Expected | Pass |

* 1. User Testing

**Participants**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Location | Background | Date |
| Alicja Kutek (alk36) | Hugh Owen Library | English Speaker  Computer User Creative Arts Student | 29/04/2019 |
| Bram Weston (brw17) | Hugh Owen Library | English Speaker  Computer User  Computer Science Student | 29/04/2019 |
| Ben Rowlands (ber17) | Hugh Owen Library | English  English Speaker Computer User  Business Management Student | 29/04/2019 |
| Joasia Syposz (Jos70) | Hugh Owen Library | English  Speaker English  Computer User  Languages student | 29/04/2019 |
| Gwilym Griffith (gsg3) | Hugh Owen Library | English  English Speaker  Computer User | 29/04/2019 |
| Patrick Grey (pag15) | Rosser Lounge | English  English Speaker  Computer User  Agriculture student | 30/04/2019 |

**Prompts for Notes and Issues**

* Where would you start to look for this information?
* What keyword or title are you specifically looking for?
* Upon finding the page, is this what you expected?
* Is this where you expected this information to be?
* What do you think about the page layout? Why do you say that?
* How did you find this task?
* Was this an easy/difficult task? Why do you say this?
* Can you suggest any improvements?

**Task List**

Run through the task list once on a desktop and once on a mobile browser.

* As Staff (Starting point: /login/)
  1. Login as a member of staff and create a lecture on the system.
  2. Create a feedback session for a lecture and set it running.
  3. Change the website from English to Welsh or Visa-Versa
  4. Merge two feedback sessions, after creating another
  5. View the feedback for the first slide of a lecture as bar graphs
  6. Download a PDF of the lecture slides for a lecture
  7. Search for a specific lecture using the search box
* As a student (Starting point / OR /connect/)

1. Provide feedback to a running lecture feedback session.
2. Ask a question to a running lecture
3. Download a PDF of the lecture slides for a lecture
4. Disconnect from a session

**Test Feedback**

**Example Person – Example Date**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |

**Alicja Kutek – 29/04/2019 - Desktop**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | Optional extra explanation needed  Unaware if students can see notes so not sure what the content should be  Unsure the purpose of the extra notes field | No | No | * Thinks new lecture should be better seen button not in navigation bar * Confused by how many slides I should put I’m not sure in the very beginning |
| 2 | Completed | Didn’t know I had to click row to highlight session for use; maybe have a prompt | No | No | * Title should be made bigger in lecture detail page * If only one session is there should highlight it automatically? |
| 3 | Completed | No | No | No | None |
| 4 | Completed? | Not understanding what merge does  Need a prompt of its success and a description of its use | Didn’t know merge had finished unsure when I had completed the task. | No | None |
| 5 | Completed | Options to change to bar charts and feedback subset was not noticeable enough took a while to find/figure out. | No | No | Weird colour and text No. responses what is there for?? |
| 6 | Completed | No | Couldn’t see where the link to the pdf file might be. Assumed it might be near the rest of the lecture details but what was there was ‘Extract from PDF’ which looked like the most appropriate although still confusing, so I clicked it and the slide count changed for a reason I did not understand.  Only after long period of confusion did I see View PDF in the navigation bar.  *Need to move PDF and feedback link from nav bar to lecture detail control panel section* | No | If uploading slides why have extract from pdf manually why not just do it automatically upon slide upload.?? |
| 7 | Completed | No | No | No | Didn’t’ need to use search box because only a few lectures (perhaps try next test with more lectures?) |
| 8 | Completed | No | No | No | Unsure if questions and feedback forms are separate thing or if submitted as one? |
| 9 | Completed | No | No | No | Might be nice to associate question to specific slides |
| 10 | Completed | No | No | No | Think for student the link to download PDF should stay as it is in Navigation bar |
| 11 | Completed | No | No | No | More notification when disconnected |

**Bram Weston – 29/04/2019 - Mobile**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | No | Cymraeg button covering create button on create new lecture | No | None |
| 2 | Failed | No | Completely unsure of how to create the session in the first place – work flow not obvious | No | None |
| 3 | Completed | No | No | No | Position of button sometimes blocking content |
| 4 | Completed | No | No | No | None |
| 5 | Completed | Looking towards session action panel for link to feedback.  Not obvious link to feedback | No | No | Possibly automatically load the first charts without the need to click on a session (or remove highlighting and add a prompt) |
| 6 | Completed | Move PDF link down near lecture detail it took a bit longer to find it in the navigation bar. | No | No | None |
| 7 | Completed | No | No | No | None |
| 8 | Completed | No | No | No | None |
| 9 | Completed | No | No | No | None |
| 10 | Completed | No | No | No | None |
| 11 | Completed | No | No | No | None |

**Ben Rowlands – 29/04/2019 – Desktop**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | Maybe Make it explicit that only accepts pdfs I was thinking I might want to upload a file of a different type | No | No | None |
| 2 | Completed | Took a moment to realise that had to click on session row/tab to bring up options. But once first clicked on it was obvious after that. | Someone else’s questions appeared when creating my new feedback session | No | None |
| 3 | Completed | No | No | No | None |
| 4 | Completed | No | No | No | None |
| 5 | Completed | No | No | No | Expecting some feedback or visual to display immediately upon entering feedback page not an issue but split-second delay |
| 6 | Completed | No | No | No | None |
| 7 | Completed | No | No | No | Couldn’t search for a while because did not alert me to extra spaces I couldn’t see. Either strip spaces off the end or alert me to the issue |
| 8 | Completed | No | No | No | None |
| 9 | Completed | No | No | No | None |
| 10 | Completed | No | No | No | None |
| 11 | Completed | No | No | No | None |

**Joasia Syposz – 29/04/2019 - Mobile**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | Not clear file upload is option add prompt  Cymreag button has blocked the create lecture button on mobile | No | No | I recommend using asterisk for easily displaying mandatory fields |
| 2 | Failed | Use of search bar for finding help or links – the search is only for searching through lectures therefore make it clear that is the case | No obvious change clicking create feedback session because changes to page were not visible upon click action | No | Idea some overlay icon to prompt user to scroll down would have made it easier |
| 3 | Completed | No | No | No | None |
| 4 | Completed | Easier if check boxes to select and merge sessions this is what I would I would be more used to using.  It wasn’t obvious with the text of the merge button perhaps a clearer button text sating exactly what two sessions would merge | No | No | None |
| 5 | Completed | Not clear where link is not obvious it’s in navigation bar  Session Table row keeps shifting back after horizontal scroll  Instructions on feedback page not noticeable in current form perhaps more colour or better position | No | No | Remove bar and pie charts until session highlighted  Maybe automatically highlight session on page load |
| 6 | Completed | No | No | No | Change text to “lecture PDF” because it both covers download and view |
| 7 | Completed | No | No | No | None |
| 8 | Completed | No | No | No | Deciding which slide at the end after entering all the feedback may make me have to rethink my feedback choices I recommend putting this first. |
| 9 | Completed | Delete button slightly off page in mobile browser | No | No | None |
| 10 | Completed | No | No | No | None |
| 11 | Completed | No | No | No | None |

**Gwilym Griffith – 29/04/2019 - Desktop**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | Unclear on what notes field is; what are the notes to be used for students staff??ect  Choose lecture field is not obvious in its function perhaps the word upload would make it more clear | No | No | None |
| 2 |  | Session row looks like field to enter data into perhaps change appearance (colour) |  |  | Going from a list of lectures to the lecture detail page with multiple sessions is initially confusing, unsure of why there are multiple sessions able to be created – some explanation might help but after first success it was clear |
| 3 | Completed | Cymraeg button not obvious place usually right top side | No | No | None |
| 4 | Completed | No | No | No | None |
| 5 | Completed | No | No | No | None |
| 6 | Failed | No | The lecture I created did not have a pdf so I could not find any link to download its pdf. If a lecture does not have a pdf make it clear that it does not have one and give me an option to upload one. | No | None |
| 7 | Completed | No | No | No | Maybe show can search date also in search box placeholder  *Note for whole site: idea to move all control to a uniform top section control panel that is constant throughout the site and would make it easier to use instead of different button locations on each page* |
| 8 | Completed | No | No | No | None |
| 9 | Completed | No | No | No | None |
| 10 | Completed | No | No | No | None |
| 11 | Completed | No | No | No | None |

**Patrick Grey – 30/04/2019 – Mobile**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Success Rate | Minor Usability Issues | Serious Usability Issues | Critical Usability Issues | Notes and Comments |
| 1 | Completed | Purpose and if optional or not was not clear for upload file field | On mobile Cymraeg button blocks create button | No | Tried scrolling to bottom first – new lecture quite faint on mobile |
| 2 | Failed | No | No clear change when hitting create session not sure where the session was.  How to set running? | When on one section couldn’t see the other section changes so there was no clear prompt of what to do | Be clearer with more prompting maybe? |
| 3 | Completed | No | No | No | None |
| 4 | Completed | No | No | No | None |
| 5 | Completed | No | No | No | None |
| 6 | Completed | No | No | No | None |
| 7 | Completed | No | No | No | None |
| 8 | Completed | Took a second to realise submitted because had to scroll down for success alert  Could not realise had option to ask question because end of feedback feels like it could be end of the page | No | No |  |
| 9 | Completed | No | No | No | None |
| 10 | Completed | No | No | No | None |
| 11 | Completed | No | No | No | None |

**User Testing Summary Notes**

1. Summary – login & create lecture
   * Not clear if fields are optional or not
   * Not clear of notes and file upload purpose – who are notes for what file is to be uploaded?
   * Cymraeg button covering create button on mobile
   * New Lecture faint button on mobile device
2. Summary – create and start session
   * Lecture title should be part of a main heading for lecture detail page
   * Unsure of how to control session not obvious the row should be clicked
   * On mobile cannot see change when hitting create feedback session
   * Unclear on why lectures have sessions (maybe more help or description)
   * Tried to use search bar for help during this task (indicates purpose of search bar is not clear)
3. Summary – Change between English & Welsh
   * Location of button sometimes hidden on mobile
4. Summary – Merge two sessions
   * Not clear on what merge does
   * Not clear when merge had finished
   * Not obvious text on merge button describing which two session were to be merged
   * Check boxes to select multiple sessions to merge would be more user friendly
5. Summary – View feedback
   * Drop down options not noticeable enough
   * Wrong colour next to NO. Responses that doesn’t make sense and should be removed
   * Looking towards session action panel for feedback link feels unnatural to look for it in navigation bar because it should be associated to a session
   * Expecting some visual display upon entering feedback page, grey background on mobile does not hint well to what should be done
6. Summary – Download PDF Lecture file
   * Move PDF download near the rest of the lecture details
   * Make it clear the difference between extract PDF and download PDF
   * Change text from ‘View Lecture PDF’ to ‘Lecture PDF’ because it covers both downloading and viewing
   * If the lecture does not have a PDF make it clear that it does not
7. Summary – Search for lecture
   * Make explicit search box functionality – “Search for lecture by name or date…”
   * Strip of extra spaces at end search didn’t work because had extra whitespace but user did not know
8. Summary – Provide Feedback to a running lecture feedback session
   * Success alert at bottom is not immediate to see on mobile because you must scroll down
   * Selecting slide to associate with should be first field
9. Summary – Ask Question
   * Question delete button slightly of page in mobile browser
10. Summary – Download PDF
    * No Issues
11. Summary – Disconnect from Session
    * Add notification of successful disconnect from session