Open edX as CATe's Replacement

Pfitzner, Bjarne Cankaya, Cemre Budiono, Daniel
Wu, Yifan Dudzinksi, Maksimilyan

January 9, 2017

Abstract

As we are all already familiar, CATe has served as the Computing Departments Content Management System (CMS) for quite a long time, and is showing its age. It is using outdated technologies which are hard to maintain, is not phone or touchscreen friendly and some of its features are not user friendly.

With our KATe (with a 'K') we intended to give an upgrade to the outdated CATe while still retaining an old feel by keeping its favourite features, such as the personal page, coursework timetable and coursework submission page with cover sheets.

Our project is trying to explore and evaluate the possibility of replacing CATe with Open edX which is being used by other universities such as MIT and Stanford for their online courses. To evaluate the feasibility of replacing CATe with Open edX we are going to set up a new Open edX instance and customise it to the requirements of the Department.

The replacement should be able to deliver what CATe has been delivering to all its users and also include new features and tools that student and staff deem lacking from the current CATe. It is also our aim to make the upgraded CATe become a Learning Management System, which creates a new learning environment for the students.

1 Introduction

1.1 Motivation

Content Management System is a software application which creates and stores content in digital format and allows users to upload, edit and manage this content, electronically [1]. In our case the content that is being managed in the teaching material that is used in the Department of Computing modules. CATe (Continuous Assessment Tracking engine) is implemented to provide an ease of communication between the lecturer or the course owner and the students enrolled in the course. The lecturer provides the students with the lecture notes, exercises and any other material related to their course while the students have the ability to submit any exercises or coursework that is set up by the lecturer.

On the outside this system looks perfect as a way to provide a one way communication from the lecturer to the students or from the students to the lecturer but we think we can make this better by creating a system with a two way communication style. That is why we are trying to create a Learning Management System, an new environment for student learning.

1.2 Aim

While CMS is aimed to manage the content, LMS is trying to manage the learning. This is the core of our project, to make the new system to be more interactive and to form new communication paths between the students and the lecturer. Hence to create a more effective learning process we are creating a suitable environment for both the student and lecturer.

One of the big problems we are trying to solve is that the feedback system currently implemented in CATe is lacking in functionality. Up to now, the feedback that the student obtains from the lecturer is still sent directly to their email address and the lecturer needs to do this manually, not using CATe as the platform. We would like to fix this by adding more feedback functionality to the new system that would not only make this process possible through the platform, but also make it easy and helpful too. Another extension is to make the process of the marking easier for the lecturer or any staff related to the course. This will be particularly a hard challenge, with a lot a feedback needed from people, as we are trying to reach a maximum satisfaction level from all personnel in the Department of Computing.

We were tasked to explore Open edX to be used as the platform to create this replacement. This is suggested as a lot of the online courses from top universities already use edX as their platform. EdX has some of the features that we would like to utilise, hence we look at its open source platform called Open edX to be able to customise it with more flexibility. This is why one of our objectives in this project is to look at the possibility of using Open edX to fully replace CATe, and if it is not possible then we will try to look at a different framework to create the replacement.

1.3 Contributions

To create a new learning environment, we would like to implement a comprehensive feedback system. One that allows students to retrieve their own feedback from the website and for the lecturer to be able to easily mark and publish this feedback back to them.

As to increase the effectiveness, we are letting the user do some basic configuration. We create a customizable deadline notification, timetable that could be filtered and also the flexibility to show or hide courses a student is enrolled to.

We are also able to evaluate the possibility of using Open edX as a replacement platform for CATe. This is achieved by evaluating the process of adding new features to Open edX that CATe currently has. Later on we conclude that Open edX is not a good platform to replace CATe, however that does not mean we stop the project, as our main objective is to create a replacement to CATe and Open edX is only the suggested platform. We continue to build KATe on our own choice of platform that we feel is suitable and will be straightforward enough to be passed on to anyone that will further develop and admin the new LMS.

2 Project Management

2.1 Planning

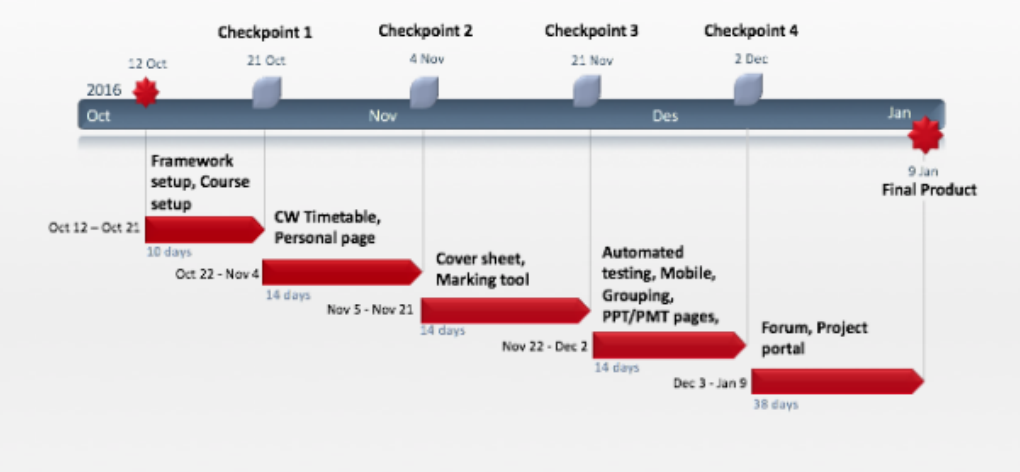

Figure 1: The initial plan created.

Our project plan, above, is based on the checkpoints we will have every two weeks. However our own internal project management will be based on one week iteration cycles. We came up with this plan by first of all discussing this project with our supervisor and from this we've gathered the expected features that our supervisor would like to see in our project. The main thing is that all the features from CATe need to be implemented in our project. These include but are not limited to the cover sheet, calendar and personal page.

We also discussed the features that CATe is lacking, such as the need to hard code the website every time a new course is added, not being mobile friendly, course grouping and the lack of tools for easy marking of student's coursework digitally. The last one becoming apparent as our supervisor received some feedback from one of the lecturers who brought up the issues that giving feedback on student courseworks can be easily done on a tablet but a script is needed to rename the courseworks when downloading them and the feedback can only be given back to students via emails which is not convenient when dealing with a large number of courseworks.

We then estimated the work by making a point based system of how hard each task is and then allocate them by order of priority before making a balance of arrangements on the difficulty level of all the tasks that need to be done.

As we are working in an agile way though, we expect the requirements to change as we get feedback of the product we develop, so we will adjust the plan accordingly.

2.2 Group Organisation

For this group project, we are using Scrum as our agile software development method. We are having a group sprint meeting at least once a week to discuss the work allocation and also the evaluation or some other design changes that we feel the need to be discussed at a certain point. Using a group chat we discuss urgent problems or a section of work that other group members are working on. This replaces the daily meetings of the group as everyone in the group has a different schedule and most of the time it is just not viable to make daily meetings on campus.

The group is self organised without someone acting as a leader, this is how Scrum is intended. So the Scrum master will be in charge of keeping the group in check and guide the people who are less familiar with the Scrum technique.

2.3 Tools

To help with our project management, we used a chat software to communicate with each other and to exchange files. As our source control system we used git on the departmental GitLab server. Our scrum board was on Trello, where we could track which parts we still need to complete and who is currently working on them. We also set up a continuous deployment pipeline in Jenkins, which automatically checks git for changes and deploys these automatically onto our build server.

2.4 Breakdown and Task Allocation

One member mostly focus on the database and create the appropriate changes on the back-end for a new feature to be implemented. One other member focus on the server side in the beginning and then help on the implementation of features later on. Two focus on the implementation of features and the last one focus on the styling and help on the implementation of feature as well.

3 Design

3.1 Open edX

Edx is a provider of Massive Open Online Course (MOOC), similar to other MOOC such as Coursera and Udacity. The main difference between edX and the other two is that edX runs as a non-profit organisation and its technology is open-source. Open edX is the open-source part of edX in the hope that other universities can easily set up their own online courses under the edX platform. Open edX's server-

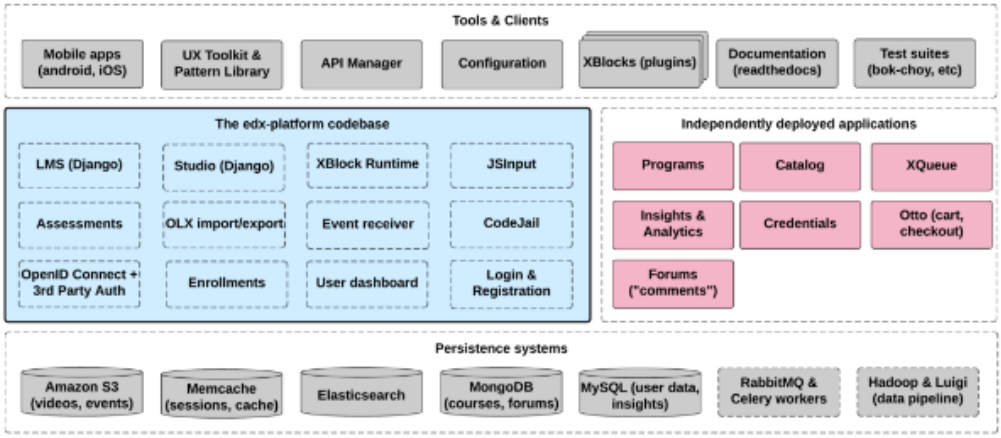

Figure 2: The architecture of Open edX platform.

side is coded mostly using Python and with Django web application framework. For data storage Open edX uses several different databases, MongoDB is used for the courses while the profile data of each learner is stored in MySQL. For the front-end template generation it uses Mako in the LMS and elsewhere, while browser-side code is mainly written with JavaScript with some CoffeeScript. Client-side is coded with Backbone.js framework and for CSS code it uses a combination of Sass and Bourbon framework.

To obtain the same feeling throughout other websites that use Open edX, the replacement for CATe should maintain all the technologies that are currently being used by Open edX. This will create a slow start in the early stages of the development, however the help that can be obtained from either the documentation or forum will be more easily accessible.

3.2 Dynamic Web Pages

Static web pages have been the standard form for website implementation, but recently the amount of users who want a responsive and interactive website is rapidly increasing. Dynamic web pages is the solution to this problem. A dynamic web page

is a web page where the content and appearance can adjust depending on the request or interaction from the user, without needing a direct response from the author of the page.

According to Polzar [2], there are three type of dynamic web pages, server side pages, client side pages and embedded pages. Server side pages handle the adjustment by letting the server make the appropriate decision choice response according to the clients request and other constraints, and sending this back to the browser. Client side pages directly handle the adjustment using the client's computer and usually handle internal actions on a single page, but they are not capable of handling complicated responses. Pages that contain dynamic content such as Java and Flash are classified as embedded pages.

In our LMS we are trying to make our web site more interactive with the aim to be able to increase the effectiveness of the learning process. A dynamic web page is therefore an absolute must for our portal as it is responsive to changes. We need to be able to automatically adjust a page when a lecturer changes the date of a submitted coursework or adjust the timetable when a student changes their module.

3.3 Back-end Development

A Learning Management System will need to handle a lot of data manipulation ranging from a student grade to a teacher teaching schedule. KATe will need to be capable of dealing with the departmental database and synchronizing it with any change necessary created by anyone who might be using KATe in the Department of Computing.

As one of the objective is to evaluate the capability of Open edX to replace CATe it is practical to use its back-end as the framework. Django is the framework that Open edX operates on, hence it becomes the main technology for back-end development. After finding out that Open edX is not capable of replacing the current LMS, a new framework needs to be chosen to continue creating KATe.

The most common back-end technology that is currently used in the industry are Ruby with Rail, Python with Django and PHP. Each one of them comes with their own unique advantages and disadvantages. In the end Django is chosen as the framework for this project seeing that all of the group members are already familiar with Python and no one is confident enough to choose Ruby while PHP does not have a framework that the group found comparable to Django's ease of use and simplicity.

3.4 Front-end Development

As the Django framework is able to quickly and efficiently create front-end part of a quality web application, it is used extensively in KATe. The provided library helps with generation of HTML code, data selection and formatting. Furthermore

it features authentication tools, caching, logging, pagination, sessions, static file management, data validation and many more.

For the styling the Bootstrap framework is chosen to provide a library for the CSS. Bootstrap becomes the main choice because of the group's familiarity with this framework although other option such as Foundation and Semantic UI would work fine for this project.

4 Implementation

4.1 Open edX

4.1.1 XBlock

XBlocks are elements that can be added to courses, which can be something like integrating a YouTube video. A few are available readily written. However, XBlocks can only be inserted inside a course, so there is no way to create an XBlock as a general page within the platform, for example a timetable XBlock. XBlocks are also intended to not support user identification, which means it is not possible to identify the user using the XBlock, which limits the use of them significantly. The documentation is extremely limited, making adjustments to existing XBlocks very hard and not giving a good starting point to create new XBlocks. We were able to reuse some of the code we wrote during our attempts to write our own new XBlocks for our second system.

The first XBlock we tried to make, would include Panopto recordings to a course webpage. Because of the API of Panopto and with some code for the XBlock, we managed to get this running.

Another XBlock we wanted to create from scratch was to dynamically create PDF coversheets for hardcopy coursework submissions, however it was not possible to access all the required data of the student and the coursework from within the XBlock, so we had to make hardcoded changes to Open edX as well, with which we lost one of the main advantages of XBlocks being modular with any Open edX system and potentially introduced security vulnerabilities into the Open edX instance.

The final XBlock would implement the possibility to mark submitted coursework of students. This again was hindered by the complexity of extracting and inputting student data into the system.

There were also a number of already existing XBlocks, that we planned on including. Firstly the "Google Drive XBlock". It would give lecturers the opportunity to share files with the students. Next, we wanted to use the "Group Project XBlock" that would allow us to set up groups of students with peer assessment. However the XBlock was outdated, so it required some patching in order to work with the newest OpenedX platform. Third we planned on including the "PDF XBlock", so that students wouldn't need to download all the pdf files but rather view them online. The "UniPlayer XBlock" was the last one, we were considering to use. It opens up the possibility to include videos via the <embed> tag. There was quite a lot of patching and even more customisation inside the "ready" XBlocks needed to get these running, and there were still some customisations that weren't possible to make.

4.1.2 Learning Tools Interoperability (LTI)

LTI is a standard in integrating rich learning applications with platforms such as LMS, portals, learning object repositories or other academic environments. LTI standard is developed by IMS Global Learning Consortium as a way to create a seamless connection between external applications and the user's platform. With this in mind, we could create anything using LTI standards and that tool could be integrated, by using the Studio functionality, inside the Open edX platform.

The main problem with doing this is that it is quite easy to integrate applications or content inside a course, but there is no feature in Open edX that enables us to create an external tool that can be used as an extra feature addition, such as timetable, coursework submission page or task notification system, that is either available in CATe or we would like to add to CATe. However, adding Piazza or panopto becomes easier as both of those application adapt LTI standards. This creates a more seamless implementation of each module inside a course with the ability to add specific link to a specific video on panopto or discussion on Piazza.

4.1.3 Course Setup

Open edX provides the admin or lecturer with a tool to manage their course which is called Studio. It provides to the lecturer the ability to customise their course however they want by creating external tools using either XBlocks or LTI.

Course Outline: A placeholder for all of the content of a specific course. One course can only have one course outline.

Course Section: Represents time period of a course, course chapter, or any other classification. One outline is required to have at least one section.

Course Subsection: Represents a course topic or other classification. Usually created for each lesson of a course. One Section could have multiple subsections.

Course Units: Is a single page that will contain one or more components that will be seen by the learner. The components will range from videos, discussions, problems that could be created using XBlocks or external components that use the LTI standard.

4.1.4 Database

Open edX uses two databases, MongoDB and MySQL for data storage. MongoDB is used to store the courseware and course content as well as the content of the forums. MySQL is used to store user data and insights. This split of databases made it very hard to extend them in a way that would allow us to store extra information about the already present built in entities such as courses or users. Also adding anything new to them was difficult due to the very intricate internal design of both. This was a big red flag in our evaluation of Open edX and one of the major reasons why we decided Open edX is not right for our goals.

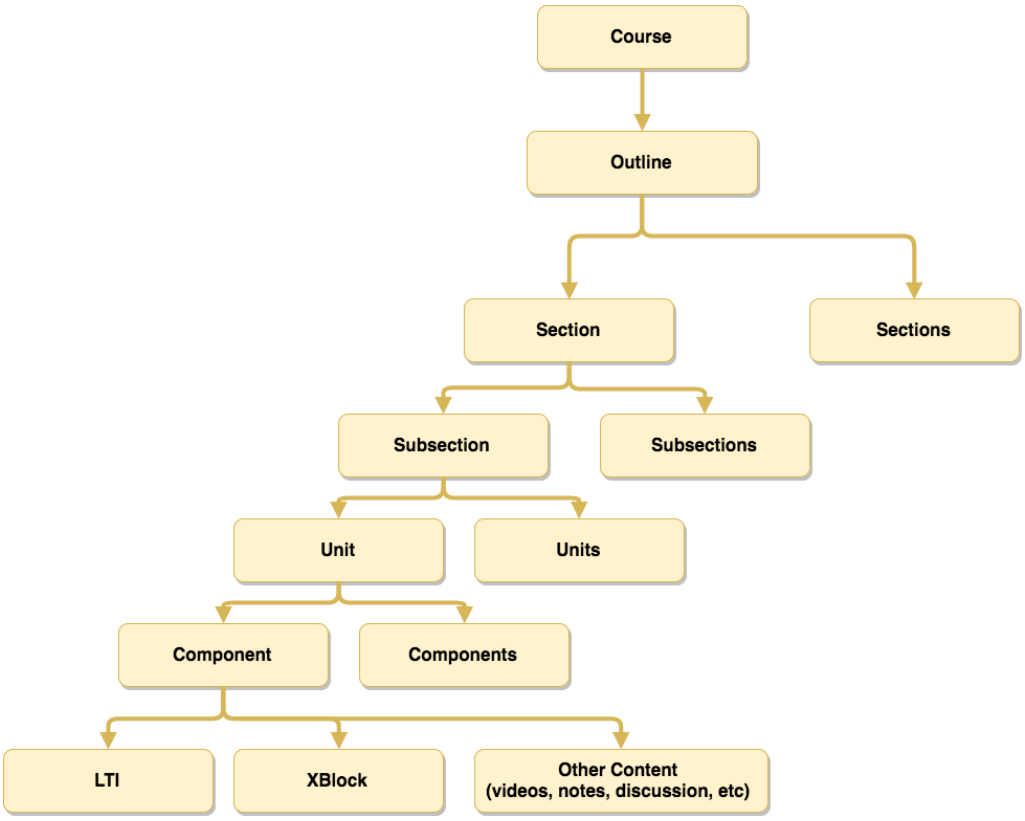

Figure 3: The top down view of course in Open edX.

4.2 KATe

4.2.1 Back-end

For our new learning environment, we decided to use Django as the backend framework, over Ruby or PHP. This is because we all know Python well and now we had some experience with it. Starting a new project in Django was satisfyingly straightforward and fast after the many weeks of thorny work trying to get OpenEDX to play along. We could not get direct access to the teaching database, currently used by the department, as it contains sensitive student information, so after having a meeting with Ms Silvana Zappacosta, we were advised to create our own database and to mirror the teaching database of DoC. We decided to use PostgreSQL to mirror it and to store some of our own data and relations. Again we've used it before and knew Postgres would provide the functionality we needed. Finally, we set up a Jenkins server that automatically pulls the latest commit from Git and builds it on our deployment server, making it available to access from within the college network. All of these are hosted on Imperial's CloudStack service, which we chose over AWS because Cloudstack VMs are located within the college network so can access other college services (like the LDAP server for authentication) easily.

Database

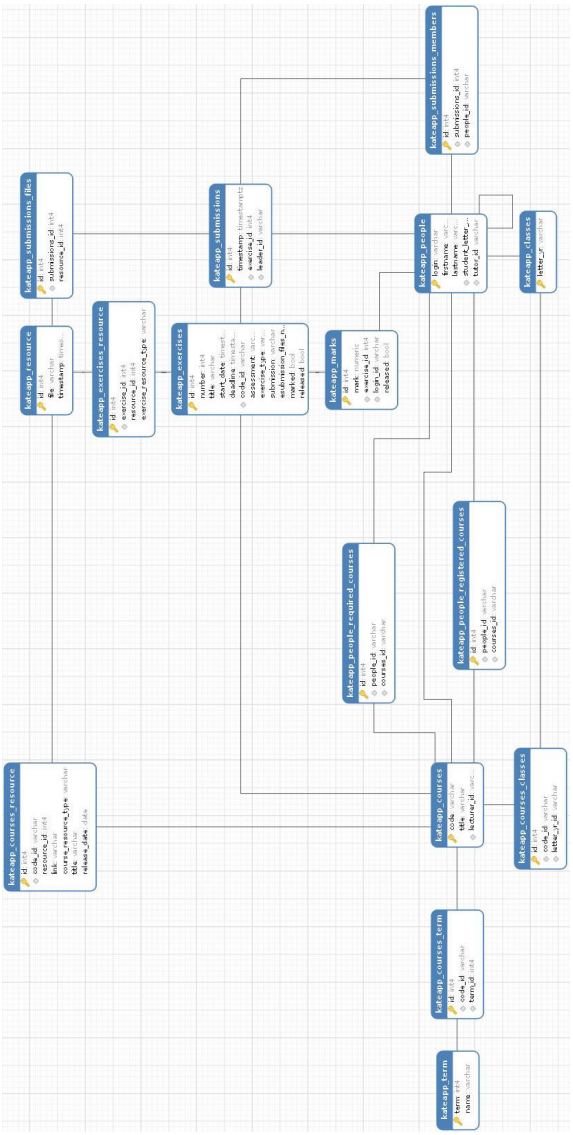

Figure 4: The view of KATe database structure.

When it come to design our database, we first looked into the current departmental teaching database. It is a well structured complex database servers and backups the whole teaching system electronically for DoC. There are parts supporting the CATE and more information for other teaching activities. As a result, we decided to take a subset of the teaching database as our starting point for our new KATe. Since the teaching database contain confidential information, we are not allowed to work straight on it or copy data from it. We set up our own PostgreSQL database on a Cloudstack VM for our app development and connect it to Django environ-

ment. Django framework treats database as Models and manages all tables and relations from defined classes. We try to mirror the teaching database as much as possible in terms of table names and field names in favor of further development and maintenance. For example, the people table is mirrored from the teaching database with unnecessary fields omitted. However, we designed and changed some tables and relations to better suit our new KATe and optimise the underlying operations. The courses_resource table would an example of this case, we expanded the information come with each entry on course detail pages which allow us to make different templates to display various kinds of resources.

```
class People(models.Model):
        login = models.CharField(max_length=200, primary_key=True)
        firstname = models.CharField(max_length=200)
        lastname = models.CharField(max_length=200)
        student_letter_yr = models.ForeignKey(Classes,
    on_delete=models.PROTECT, null=True)
        tutor = models.ForeignKey('self',
    on_delete=models.PROTECT, null=True)
        required_courses = models.ManyToManyField(Courses,
    related_name='required')
        registered_courses = models.ManyToManyField(Courses,
    related_name='registered')

class Courses_Resource(models.Model):
        code = models.ForeignKey(Courses,
    on_delete=models.PROTECT)
        title = models.CharField(max_length=200)
        resource = models.ForeignKey(Resource,
    on_delete=models.CASCADE, null=True)
        link = models.URLField(null=True)
        release_date = models.DateField()

        NOTE = 'NOTE'
        PROBLEM = 'PROBLEM'
        URL = 'URL'
        PANOPTO = 'PANOPTO'
        TYPE_CHOICES = (
        (NOTE, 'Note'),
        (PROBLEM, 'Problem'),
        (URL, 'Url'),
        (PANOPTO, 'Panopto'),
        )
        course_resource_type = models.CharField(
```

```
max_length=15,
choices=TYPE_CHOICES,
default=NOTE,
)
```

4.2.2 Front-end

After setting up everything we started to work on the new portal and implementing the features that CATe currently has but are missing from publicly available learning environments eg. the cover sheets, the timetable of work, the individual student pages. These are the features we were looking to add to OpenEDX in the first place.

Personal Page

Figure 5: The view of Quick Link section on personal page.

Personal page function in KATe is similar to CATe's which is to act as the home page for students and lecturers after they log in to the website. It contains links to the Timetable and Module List for easy page navigation using a navigation bar at the top of the page. Similarly to CATe, it contains the student's details and also quick links to useful external tools such as project portal, module subscription and even the teaching database. The simpler symbols help the user instinctively access the intended more intuitively.

A new feature of KATe's personal page is the task notification. This serves as a quick view for the student to be able to see their deadlines that will come up soon without having to look at the timetable, as well as a reminder to finish and submit them, with the hyperlinking taking you to the exercises submission page.

The twitter feed in CATe are found to be useful which resulted in its implementation on KATe which served as a news feed for anyone who does not have a twitter account.

Task Notification

Exercises Title	Course	Due Date	Time Left
Databases Coursework I	Databases I	Sun, 08 Jan 2017 08:00PM	1 hours
Jan Maths Exam	Mathematical Methods	Wed, 11 Jan 2017 09:00AM	2 days
Presentation Skills PowerPoint	Presentation Skills	Fri, 13 Jan 2017 02:00PM	4 days

Figure 6: The view of Task notification on personal page.

Figure 7: The view of Twitter feed on personal page.

Marking System

When programming the marking system, we wanted to especially improve the lecturer's experience. On the Personal Page of a lecturer, there are all the Courseworks shown which are ready for marking, with the number of days passed since the deadline. This should help the lecturer see which coursework should be marked and also remind them of coursework that has been submitted by the students quite a while ago. The lecturer can then click on "Mark" to view the Marking Page. On there he can download all the student submissions, as well as upload the marked coursework

as a .zip archive, and enter the marks for the students. If a coursework has a group submission, the lecturer has the possibility to enter marks for the whole group or alternatively give different marks for group members for individual groups. Once the marks for every student are entered and saved, the lecturer can press the button "Submit" to submit the marks to the system and either release them directly or at a given time. Once the marks are submitted and released, they are shown to the students in their Individual Record.

Exercise Page and Cover Sheet

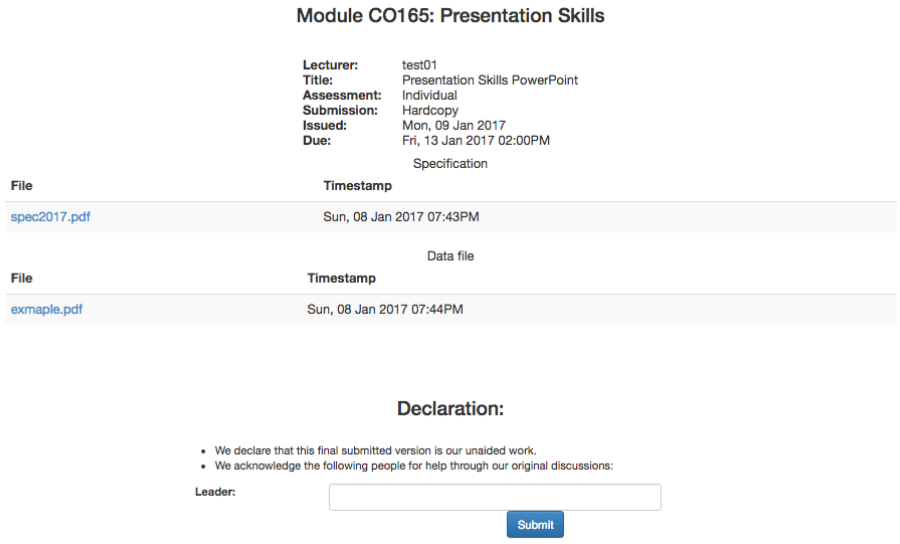

Figure 8: The view of submission page.

Every exercise that the lecturer sets has its own submission page where students can submit their work. This is displayed in one of three ways. If the exercise has no formal submission, such as a Tutorial exercise, the page displayed just shows some info about the exercise, such as the title and the dates when it was set. It will also list any associated files with this exercise, such as the specification, data files, or model answers, once released. The second view that may be displayed is if the exercise has a hard copy submission. This page again shows the info and associated files but also a field to enter the login of the party that is doing the submission. Once submitted a link appears from which the cover sheet can be downloaded. This page will also have extra fields to list the group members, if its a group submission exercise. Lastly if the exercise has electronic submission, a field is added to select the required files so that the files can be uploaded/reuploaded.

The code below shows how the initially called function is structured. We first make sure the exercise exists and then we determine if we'll display the model

answers. We then fetch all the associated files here at the top level and then pass them down into each of the sub functions that display the different views of the submission page as discussed above. These then do all the logic associated with displaying the respectful views, before calling the render function to render the template. Each of them also handles their own POST requests of the form associated.

```
def submission(request, code, number):
        exercise = get_object_or_404(Exercises,
    code=code, number=number)

        user = getUser()
        showAnswers = isTeacher(user) or
    markingReleased(exercise)

        #Uncomment for testing purposes
        #showAnswers = True

        course = get_object_or_404(Courses, pk=code)

        # Computes and puts together the resources
    that the template uses, ie the files associated with this exercise
        specification = list(Resource.objects.filter(...)
        data = list(Resource.objects.filter(...)
        answer = list(Resource.objects.filter(...)
        if showAnswers:
                marking = list(Resource.objects.filter(...)
                resource = (specification, data, answer, marking)
        else:
                resource = (specification, data, answer)

        #We split here depending on what type of submission
    will be available
        if exercise.submission == Exercises.NO:
                return displayPlainSubmissionPage(request,
            course, exercise, resource)

        elif exercise.submission == Exercises.HARDCOPY:
                return displayHardcopySubmissionPage(request,
            course, exercise, resource)

        else:
                return displayElectronicSubmissionPage(request,
            course, exercise, resource)
```

Implementing the cover sheet involved using a third party Python library called reportlab. This contains functions for dynamic generation of pdf files, which is how the cover sheet are output. The cover sheet contain the basic info of the student, course and exercise. The cover sheet also has on it a QR code, that will be scanned to make it easier for the department staff to keep track of what students have submitted work. This QR code is dynamically generated, through a GET request to Google Charts, an API that provides the generation of many graphs and other visual objects.

Student Calendar

The calendar is always one of the very important functionality of CATe and many students go straight to this page after login to CATe for their timetable and exercise deadlines, it acts as a summary page for key information. We decided to adopt the original colour scheme from CATe for easier user transition because students have used to the current exercise colour scheme, sudden changes to it would lead to confusion. All icons are removed from the page leaving only titles for every course and exercise because we have optimise the summary pages of each entry. More details are moved to those pages so the timetable can focus on providing crucial information for students. Making simplification while keeping the informative is the goal of the new timetable design in KATe.

Course Management and Exercise Setup

KATe will provide a very simple and fast Course management page that will allow the lecturer to add course content such as lecture notes, links to panopto recordings of lectures and any other course related files. These will be displayed on the course page above the related course exercises, in chronological order. A lecturer will be also able to send course 'updates' through the course page that will be displayed at the news feed of all students subscribed to that course.

The course page also contains a button, that only the lecturer can see, which allows them to add new exercises, such as Courseworks or Exam entries. This exercise setup page allows them to specify whether there is submission required and of what type this is (individual or group). Theres also a field to upload files for the exercise and classify them, under Specification, Data File, Model Answer or Marking Scheme. New exercises will no longer have to be hardcoded but can be easily created and edited by the lecturer. They then automatically appear in the subscribed students' calendar.

Login

One of the problems we've encountered is trouble with LDAP servers and verifying users when they login in on our portal with the college's authentication system.

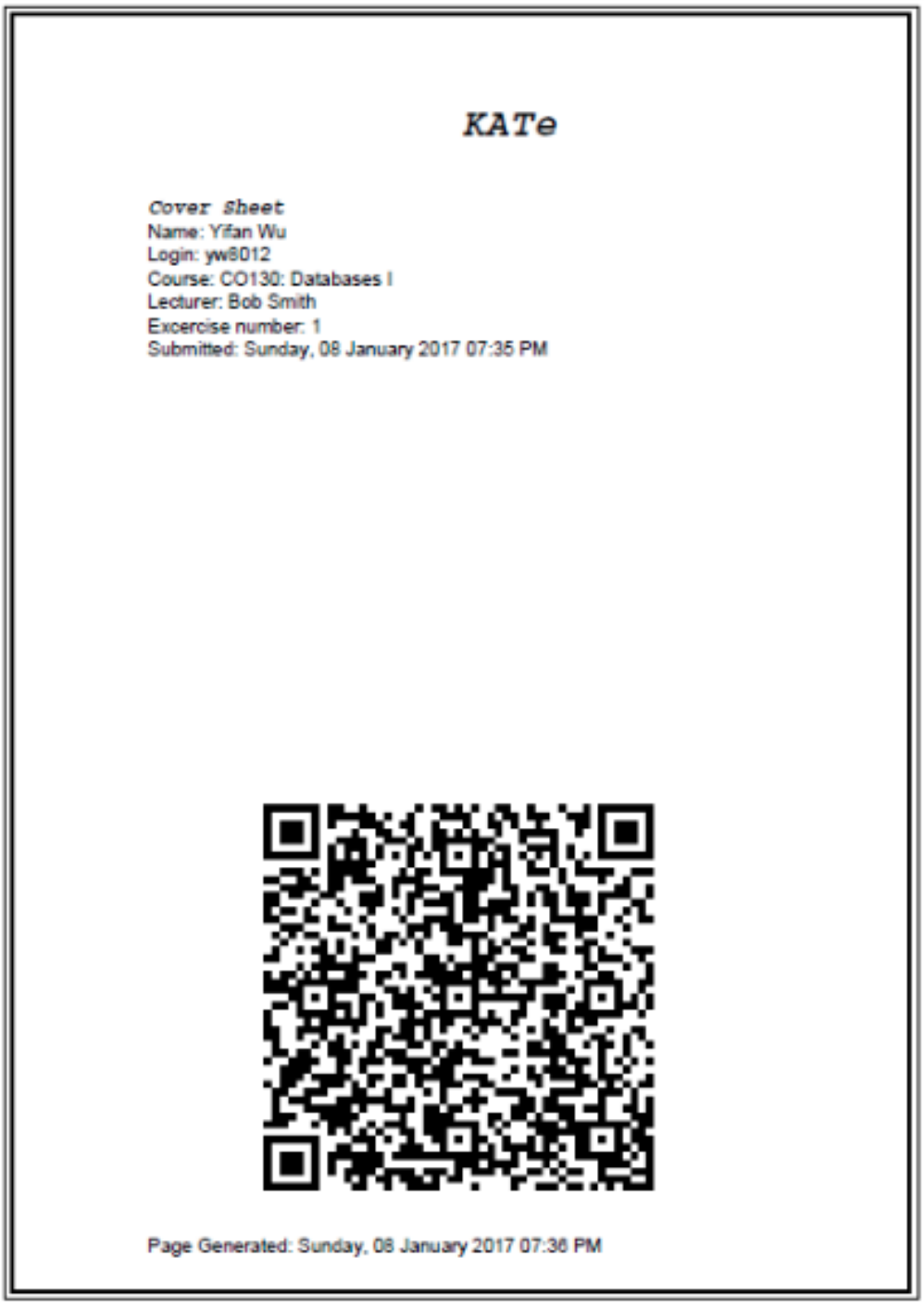

Figure 9: The view of the generated cover sheet.

There were different endpoints we could use with different Django modules which supported these. At first we tried the Shibboleth endpoint with its Django API, but

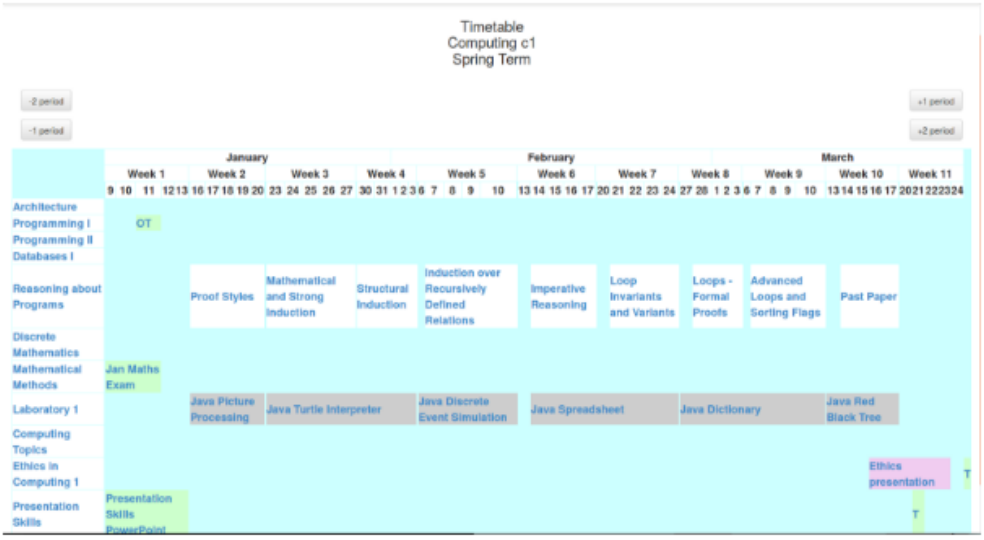

Figure 10: The view of the calendar page.

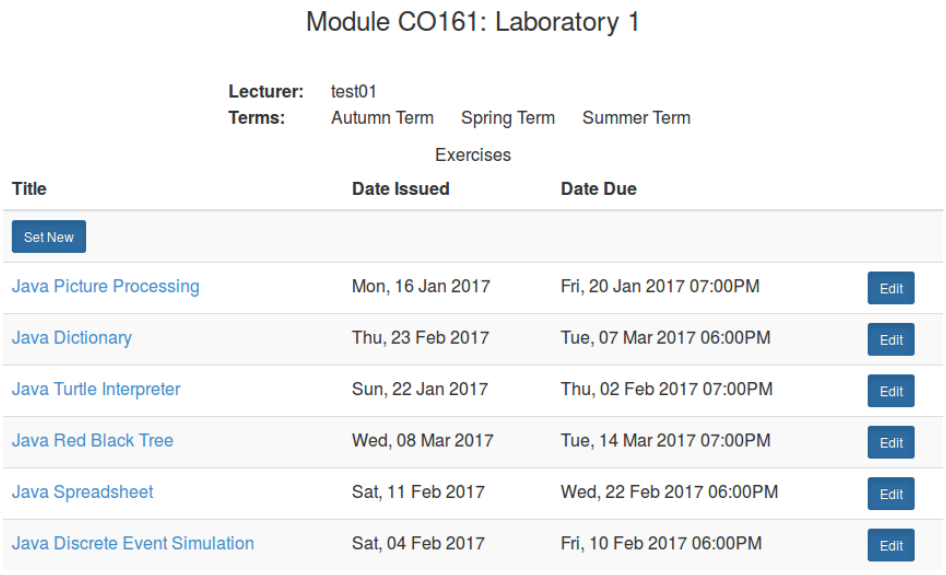

Figure 11: The view of the course page.

it did not give us the functionality to get more details about the user apart from the fact that the user has a valid Imperial login, so for example it was not possible to filter out people from outside DoC. The second system we tried to use was using the Gitlab oAuth login, but again this did not provide enough information about the logged in user. In the end we used the DoC LDAP server with the Django LDAP

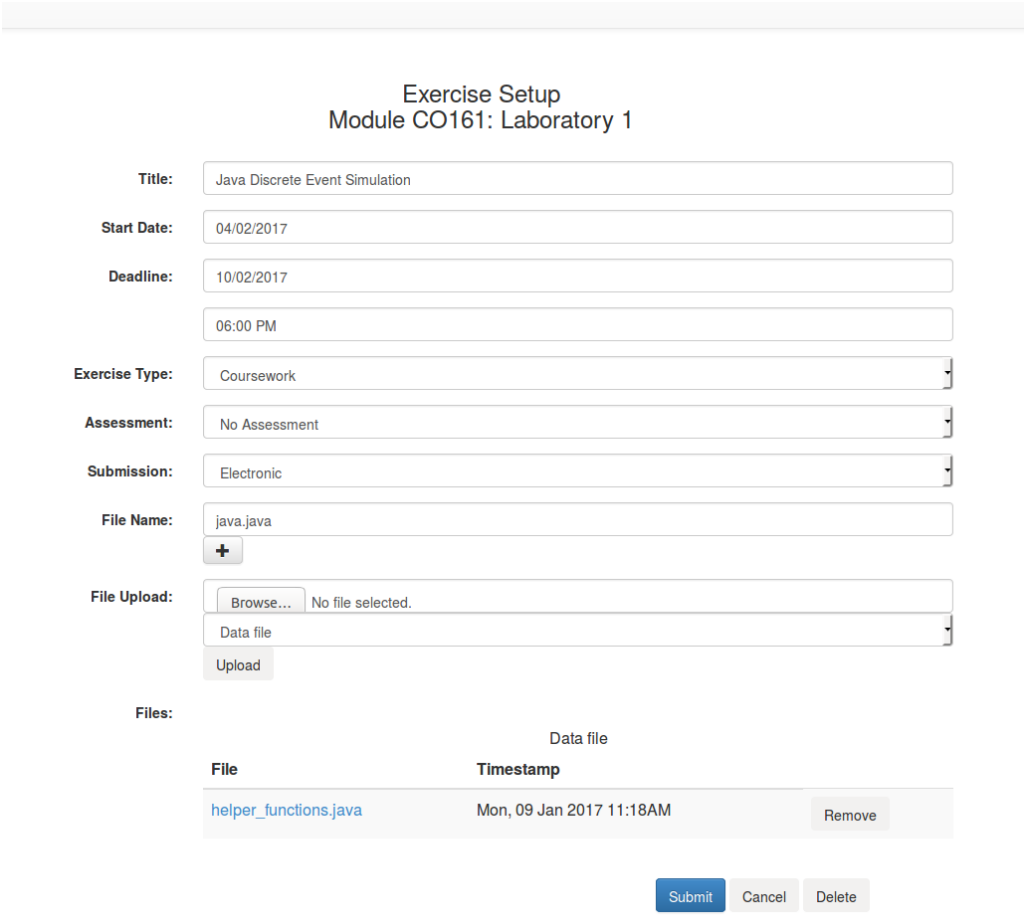

Figure 12: The view of the exercise setup page.

authentication module, which had to be configured in a different way as the LDAP server is using Microsoft Active Directory, and so parameters were different to the usual LDAP server setting.

5 Evaluation

5.1 Open edX as Replacement for CATe

One of the main advantages for using Open edX is that it already comes with a complete package to create a learning environment. Creating courses is relatively straightforward and easily customisable. The tools they provide are really suitable for creating an online course for independent study, although this becomes the reason we decided to conclude halfway through our project that Open edX is not suitable to replace CATe.

Open edX's documentation provides instructions and specification on how to customise the course but massively lacking in specification to modify the web page itself. For example, just trying to add a link to the navigation tab is proven to be a complicated process with most of the time wasted on trying to figure out the code internally, without much help from the documentation. Trying to figure out what a variable used in the HTML template actually is proved really challenging as there is no documentation which gives specifics on the functions of various variables. Also the settings for them are placed in many places without a clear information in the documentation or file on how these variable settings overrule each other.

Furthermore the main reason that Open edX is not a good environment to replace CATe is in its database management. The documentation provides only extremely limited instructions on how to be able to manipulate the databases. And these are several levels down in the documentation making one think they are not wanted to be found, as in fact some of them didn't work at all or where not specified where they are to be entered. A simple query such as trying to change the starting date to a course is very challenging as the platform does not offer any tool to access the database directly.

However the performance of the website is relatively smooth with fast, responsive times and a very user friendly design. Open edX provides both the tools to set up the course using its Studio which is a much more user friendly than the current CATe for course management as the course instructor can freely design the course structure and outline to their teaching style, as well as the LMS which is where the students interact with said course.

5.2 KATe as Replacement for CATe

One method to evaluate KATe is by creating all the courses taught to the first year Computing students exclusively. We do this to simplify the inter-department database relationship, in this case between Computing department and the Mathematics department for JMC students. Hence we create the necessary functions that CATe has as well, such as the timetable and personal page.

The timetable is almost identical to the one of CATe as it is proven to be really effective with a very detail information while still keeping the clean look from. The

timetable performs how it is expected, with a link to each course and submission page of an exercise. There's a bit of tweaking where the main link to the timetable from the homepage will link directly to the default timetable. This is implemented to create a more clean look in the homepage while still keeping the ability to select any other timetable in the timetable page itself.

Comparing KATe to CATe, KATe almost excels CATe in every aspect as it is created to improve the learning experience. Small adjustments are made according to the feedback from our supervisor and students. This resulted exactly on how it is aimed to be, not as something totally new but as a refinement to CATe that will benefit the student and the teaching staff in years to come.

6 Conclusion

Finding a good platform to create a Learning Management System is proven to be challenging as there is a lot of options, each with their own advantages and disadvantages. Open edX is available as one of the choices to create this with open-source technology. It comes with tools for course owners to easily make their courses customisable which enhances the learning experience as they can adjust it accordingly with each to their own unique teaching styles. However, Open edX is not designed to be implemented as LMS for UK university students.

Open edX is designed as a MOOC which means its purpose is to provide an environment for online independent study. It excels at this as it is capable of creating courses unique to each course owner designed with different styles. However with no relation at all between one course to the other course. The system lacks sophistication in the course management part that CATe provides where you can look at all the modules with a very sophisticated timetable feature. The feature where you can monitor deadlines for exercises is also not present in Open edX. CATe is designed optimally to keep track of things for students who take a lot of courses at once while Open edX is designed to encourage people to finish courses one by one with their own pace.

Furthermore, Open edX is not designed to interact with an external database in mind. Inside its documentation there is no part explaining how to use external databases or even on how the internal database could be manipulated. With Department of Computing's student database in mind, this will become a major bottleneck as it is a very difficult task to synchronise two different databases with possibly conflicting fields. Hence, Open edX is not suited to be a replacement for CATe as it is never designed to assist class based teaching the Department of Computing has.

Creating an improvement to CATe is challenging in the design part as most of the students are quite content with all of the features it provides. Most of the problems in CATe are found with the features for the lecturers. The main problem is that the feedback that has been created for an exercise is still sent by email to individual students. Other problems such as the naming of the answers to exercises students have submitted do not give the lecturer enough details about it, making marking harder.

At first these problems seem to not be major, maybe even ones that could be fixed inside the current CATe as none of them require change to the core of CATe. However, a replacement still could be made even with just refinement in small areas as it will create a significant improvement in overall learning experience. This new improved version of CATe is named KATe.

Just as the name suggests KATe is not designed to create big changes but it is designed to provide those small upgrades to each feature it already has. It creates a fresh experience while also maintaining the familiarity that CATe's users are used to. It is aimed to create an intuitive experience to improve the whole learning process

in the Department.

7 Future Extensions

For the personal page, the task notification could become more sophisticated by trying to pull important news or notification such as the maintenance time from an important email update or CSG notification. This resulted in the increase of a wider exposure of crucial information as KATe will be accessed frequently by students. Updates when new exercise is being added or when an exercise's deadline is getting an extension could be added as well. Realising that having a lot of information being shown in the task notification could become too much for some students, then it will supposedly be able to filter the type of information shown on it base on the student's preference.

For the timetable, the design could be changes to create a more sophisticated feel and to increase clarity of information. A customisation could be added to the calendar so that student could add or remove courses that they want to be in the timetable. It could also be a good practice having an option to hide the part of timetable that is already in the past to increase the clarity of the timetable.

Creating an online marking tool so that lecturer could give feedback to student more easily is also a possible extension to consider as this could smooth the feedback process that currently is still manually done. The lecturer should have an option to either use the online marking tool which will save the feedback given by a lecturer until it is published or he could choose the traditional method of downloading all the files needed for feedback and uploading all of them after the all the feedbacks are given.

A short manual for lecturer when creating exercises could also be added to help getting used to the new system. Currently the form used by the lecturer to set up an exercise contain a relatively good amount of selection and could be confusing for some lecturer who is not used to CATe already. A quick description on what each field in the forms are for could be added to ensure the correctness of an exercise.

References

[1] Sadaf Ajmal. Content management system vs learning management system – an overview, January 3, 2017.

[2] Brad Polzar. Dynamic web pages. *UWP Computer Science and Software Engineering Technical Report*, 2007.