

University of Wales

Trinity Saint David

MODULE CODE/ COD Y MODIWL: ACCC5004 WEB
DEVELOPMENT

MODULE TUTOR: Suneth Fernando

Antoci Bianca

1817784

Table of contents

Research	2
Introduction	4
Testing	6
Evaluation	9
References	13
Appendix	14

Research

Web development is the construction process of a website using HTML, CSS, and JavaScript. There are many frameworks to help the construction of the Website.

Web frameworks provide structure to the applications, and they give additional features. Frameworks give a starting place, so the developers can focus on specific features rather than configuration details. Web frameworks can be split into two parts: backend and frontend [1].

The following frontend frameworks were evaluated:

Angular

Angular is a front-end framework created by Google that uses Typescript as a primary programming language. This framework has grown a lot and comes with a variety of features and capabilities.

It includes Bootstrap and the Material Design components for an easy approach to UX development. There are a number of ways to style an Angular Web App including pure CSS, SCSS, or SASS [2].

React

React is the most famous library for creating user interfaces. React is a front-end framework created by Facebook and is currently used by the Facebook team to build its web platform. It is a very flexible framework allowing complex UI to be loaded in runtime as React is a client-side front-end framework. React breaks down the UI into smaller pieces of code that can be reused; these are called 'components' [3].

JQuery + Bootstrap

JQuery and Bootstrap are very common choices when it comes to building a responsive web application. JQuery was created in 2006 and since then it has become one of the most used frameworks (currently used by 76% of the most visited websites) [4].

Bootstrap is a 'UI' framework that allows you to manipulate the HTML in a very flexible way. Many concepts like "flex-box" and columns allow for a quick and responsive web application on all 3 device types: tablet, desktop, and mobile [4],[5].

Firebase

Firebase is another Google product that allows developers to scale their products using Google infrastructure. Firebase offers various products and solutions to host, manage, and deploy web and server applications. It also offers very advanced Analytics and AI solutions on the shelf services like image, video, voice recognition [6].

Introduction

The web application was designed using a design tool called Invision. The UX was the first step of the creation process [7].

After the design was finalized, Angular was chosen as a frontend framework [8] and Firebase Firestore Database was used to host the application data, [6].

The main idea of the project is to allow the university and students to connect on a web platform. The university “admin” users have the option to create courses that will be visible in real-time to students and make it easier to pick them in the registration process.

The registration process allows students to register and, if successful, use the platform to take exams for different courses.

After the registration process is completed the university “admin” user has the ability to approve the student application from a dedicated screen in the web application [9].

The application is made of the following screens:

1. **Courses screen**

This screen allows students to see the available courses created by the university admin users without logging into the platform. Each course can be selected to register a new account.

2. **Registration screen**

It allows students to register with name, email, password, and one course. After registering a student can use the various student-related features.

3. Student home page

This page displays the list of modules that a student can take and subsequently re-take. Each item on the list has a call to action that will take the student to a different page.

4. Student profile page

The student profile page allows a student to change their personal details and see the course they've selected in the registration process.

5. University admin courses page

On this page, the admin users can see the list of courses they've created with the ability to tap a button and create a new course or modify old ones.

6. University admin create a course page

Admin users can create new courses by inserting a course name and course details. Once completed the data is stored in the Firestore Database in real-time. Admins can then see the course they've just created.

7. University admin students acceptance screen

Every time a student joins the platform their application has to be approved by an administrator. In this screen, the admins can see the list of student's pending applications with the ability to accept or decline.

The UX of the platform was developed using a product called Invision (appendix figure 27). Each screen was designed with mobile screen size in mind and no colors or UI design was used on purpose to accentuate the user experience flow.

Each screen contains all the fields, buttons, and navigation elements required for the user to perform the intended action on the screen.

Testing

To demonstrate the platform's capabilities and functionality the following tests were conducted:

Test Cases	Expected result	Actual result	Discrepancies	Test result
As a student I should see the list of courses	When the student navigates to the /courses page they should see the list of course with name, description and a CTA button called "Select course"	As expected	No	Appendix figure 01
As a student I should be able to register using name, email, password and course	When the student fills the form all the entries should be stored on the Firestore Database and the student should be redirected to their home page(fig 02 and fig 03). The database should reflect the new student record with a 'pending' state(fig 04). After registration the user name and credentials should be stored on the browser.	As expected	No	Appendix figure 02 figure 03 figure 04
As a student I should be able to login if I already created an account	If the login fields are invalid the "Continue" button should be disabled(fig 05). If the credentials are valid the student should see the "home page" which is the module list(fig 06). If the email or password are invalid an error message should be displayed(fig 07).	As expected	No	Appendix figure 05 figure 06 figure 07
As an admin I should be able to login	The university admin users can login with email and password(fig 08) If the credentials are valid the admin user should see the "home page" which is the courses details page(fig 09).	As expected	No	Appendix figure 08 figure 09 figure 10

	The admin user in the database should be flagged as expected as admin(fig 10).			
As an admin I should see the list of course which I can delete and edit	The admin user should see the list of courses. For each course an edit and delete button should be displayed(fig 11). The list of courses should be stored and retrieved from the cloud database(fig 12).	As expected	No	Appendix figure 11 figure 12
As admin I should be able to create a new course	The create new course page should display a course name and details field(fig 13). When 'Create course' is pressed the data should be sent to the server side and stored in a table(fig 14).	As expected	No	Appendix figure 13 figure 14
As admin I should approve a student that is pending approval	Every time a new student registers on the platform the admins have the ability to approve their application(fig 15). The flag that indicates if a student is approved or not should be changed on the database(fig 16). Once a student is approved the UI should indicate that with a "tick" icon(fig 17).	As expected	No	Appendix figure 15 figure 16 figure 17
As a student I should see the list of modules	The student home page is composed of the list of modules the student has to undertake and each module should have a button to take or re-take that specific module(fig 18).	As expected	No	Appendix figure 18
As a student I should see the profile page	As a logged in student clickin on the top right corner should display a dropdown with 'Profile' and 'Logout'(fig 19). Clicking on logout should redirect the student to the login page. Clicking on the profile page should display the user profile with the ability to edit all the details(fig 20).	As expected	No	Appendix Figure 19 Figure 20 Figure 21 Figure 22

	Clicking save on the profile page should update the user details on the database.(fig 21 and fig 22)			
As a student or admin I should be able to see and navigate using the sidebar	As a student or admin I should see the side nav on desktop(fig 23) and on mobile only the icons should be displayed(fig 24). Tapping on the side menu button should open and close the menu(fig 25)	As expected	No	Appendix Figure 23 Figure 24 Figure 25
As a student or admin I should be able to click on the help button and the email client should show	In order for a student or admin to ask for help on using the platform they can tap on the help button and the OS email client should open.	As expected	No	Appendix Figure 26

Evaluation

Challenges

UX is the hardest part of designing a platform for the end-users. It requires an understanding of customer requirements. In our case, we had to interview both students and teachers to understand their requirements. Invision helps a lot to design a UX flow but there are other tools like Figma or Zeplin.

The typescript was a challenge to learn as a new programming language. It was also interesting because it is a strongly typing based programming language compared to JavaScript, which allows for cleaner code. Typescript is native to Angular, which was used with the frontend development for the project [10].

Angular has a few challenging concepts like page routing, modules vs components, CSS vs SCC. To be able to code an Angular application, it requires a basic understanding of JavaScript as Typescript is built on top of JavaScript [2].

Firestore Database is a NoSQL database which means no JOIN queries are allowed. This was a challenge as in a few places the platform required to query multiple tables, like for example listing the courses and students on one page [6].

Lessons

Angular as a framework and its project structure can be difficult to understand initially. It provides two ways to create a routing implementation:

- **Basic routing:** dictates that all components should be downloaded at the same time
- **Lazy routing:** dictates that some components should be downloaded as “chunks” as the user navigates within the web application.

Each strategy has its advantages but in summary, the basic routing is good for smaller applications while lazy routing is more fit for bigger applications.

In our case, we choose the basic routing as the application contains around seven different screens and side navigation [2].

Typescript as a programming language is very flexible and it works very well with the Angular framework. Typescript is updated independently from Angular which means that we get frequent updates of potential bugs.

The transition between JS and TS is seamless and somewhat easy to understand [10].

Regarding the layout, Angular uses two ways of writing style:

- **CSS:** the classical way of writing style for HTML which Angular provides by default.
- **SCS:** the more advanced way of composing our style classes and is a “preprocessor” scripting language, meaning that it needs to be translated into CSS by a compiler(in our case Angular).

Responsive design is one of the most important things to consider when building a modern website. There are three different types of screen sizes that we need to consider when designing the UX of our platform:

- mobile(with breakpoint between 480px - 768px),
- tablet(with breakpoint between 768px and 1024px),
- Desktop(with breakpoint between 1024px and above).

Bootstrap is included in the Angular framework and allows easy use of breakpoints, flex-containers and tables in a very responsive way [7].

Sending data to a remote Database is a mandatory integration in any dynamic web platform. In our case, we choose Firestore Database which is a NoSQL fully managed database by Google. It requires minimum integration and is completely separated from the frontend code. The database is contained in a different environment from the frontend, which is hosted on a separate server that allows scalability [12].

Limitations.

While researching and later developing the platform a few limitations were encountered:

- NoSQL DB doesn't allow JOIN queries, which is a huge limitation for a complex platform. It can be resolved by duplicating data which is not always ideal [11].

Improvements

The UX and UI had to be implemented for 3 different screen sizes: desktop, tablet, and mobile. Angular and Bootstrap together take care of adjusting some of the layouts automatically but extra care has to be taken to make sure that all screen sizes work as expected.

More functionality should be added before releasing the app in the real world:

- advanced student profile, with the ability to request the student data hosted on the servers as per GDPR rules,
- social sign in with Google, Facebook and, Apple, which most of the modern platforms have,
- a messaging feature to allow students and university admins to communicate in real-time.

The platform should allow in the future to gather feedback for continuous improvement. To get user feedback various tools like Zendesk or JIRA can be used.

Deployment in a real environment

Cloud providers can be used to deploy the platform into a real-life environment. For example, the website can be hosted in AWS S3 buckets and the MongoDB atlas in Google cloud can be used to scale the backend.

Hosting costs should be estimated before a real-life deployment is made. Different Cloud providers offer different pricing and scalability should be considered when choosing a hosting provider [13].

Continuous integration must be implemented before an actual production release happens. It allows developers to continuously develop, test, and release the product in a very agile way.

References

- [1] Fernando Monteiro (2014). *Learning Single-page Web Application Development*. Birmingham,UK: Packt Publishing Ltd.. 214.
- [2] Shyam Seshadri (2018). *Angular: Up & Running*. United States: O'Reilly Media, Inc... 311.
- [3] O'Reilly Media, Inc. (2017). *Learning React* . United States: O'Reilly Media, Inc.. 153.
- [4] W3Schools. (2020). *Bootstrap Get Started*. Available:
https://www.w3schools.com/bootstrap/bootstrap_get_started.asp.
- [5] W3Schools. (2020). *jQuery Introduction*. Available:
https://www.w3schools.com/jquery/jquery_intro.asp.
- [6] Laurence Moroney (2017). *The Definitive Guide to Firebase*. United States: Apress. 281.
- [7] InVision. (2020). *Design Better. Faster. Together..*. Available:
<https://www.invisionapp.com/studio/learn>.
- [8] Angular. (2020). *One framework. Mobile & desktop..*. Available: <https://angular.io/>.
- [9] University of Wales, Trinity Saint David, "Web Development(ACCC5004_LON)"[Online]. Available:
https://moodle.uwtsd.ac.uk/pluginfile.php/245756/mod_resource/content/5/ACCC5004-Web-Development-Oct-2020%20coursework%20v2.pdf
- [10] Boris Cherny (2019). *Programming TypeScript*. United States: O'Reilly Media, Inc.. 322.
- [11] mongoDb. (2020). *What is NoSQL?*. Available:
<https://www.mongodb.com/nosql-explained>.
- [12] Firebase. (2020). *Firebase Database* . Available:
<https://console.firebase.google.com/u/0/>.
- [13] Google Cloud. (2020). *Solve more with Google Cloud*. Available:
<https://cloud.google.com/>.

Appendix

Useful links:

Deployed website in the cloud:

<https://student-management-5c7a0.web.app/pages/courses>

Github project source code:

<https://github.com/bianca-antoci/student-management-platform>

Link to Firebase Database:

<https://console.firebase.google.com/project/student-management-5c7a0/firestore/data/~2Fusers~2F1604783837054>

Project instructions:

1. Pull the project from GitHub
2. Run npm install
3. Run ng serve

To login like an Admin (fig. 10):

- email "john.smith78@gmail.com"
- password "1234567"

Implementation screenshots:

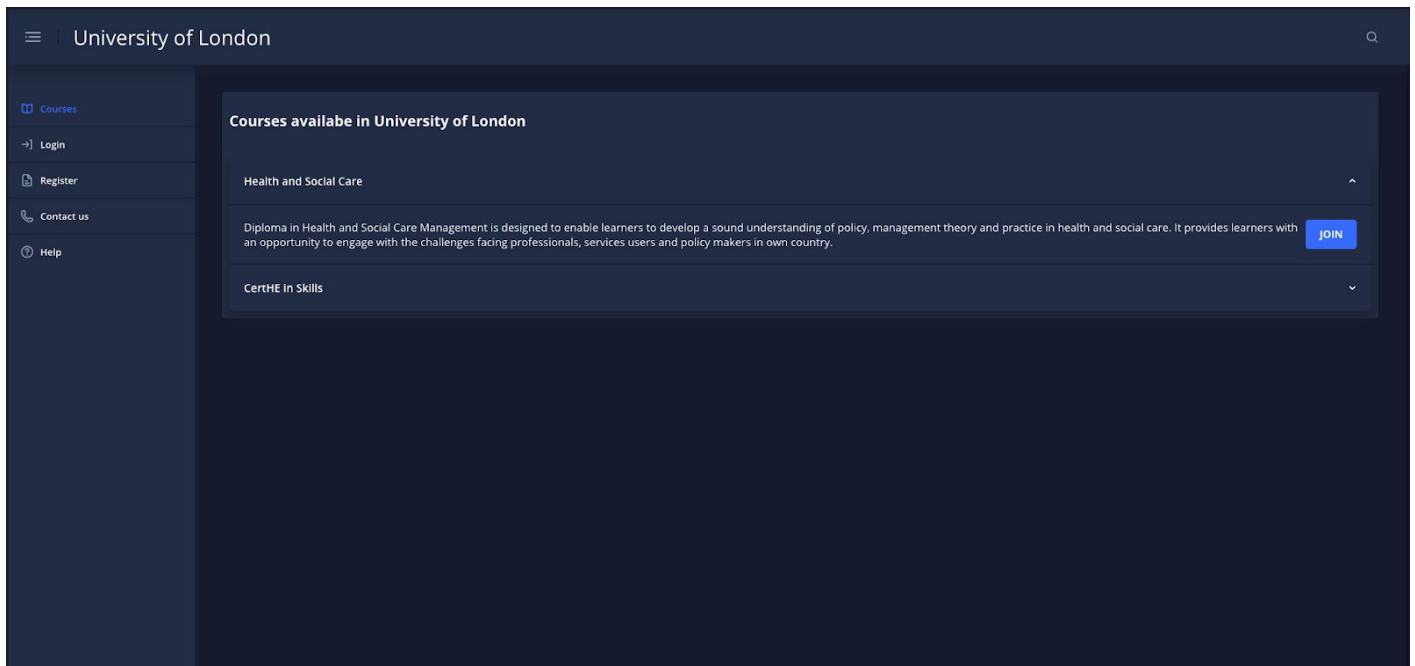


Figure 01

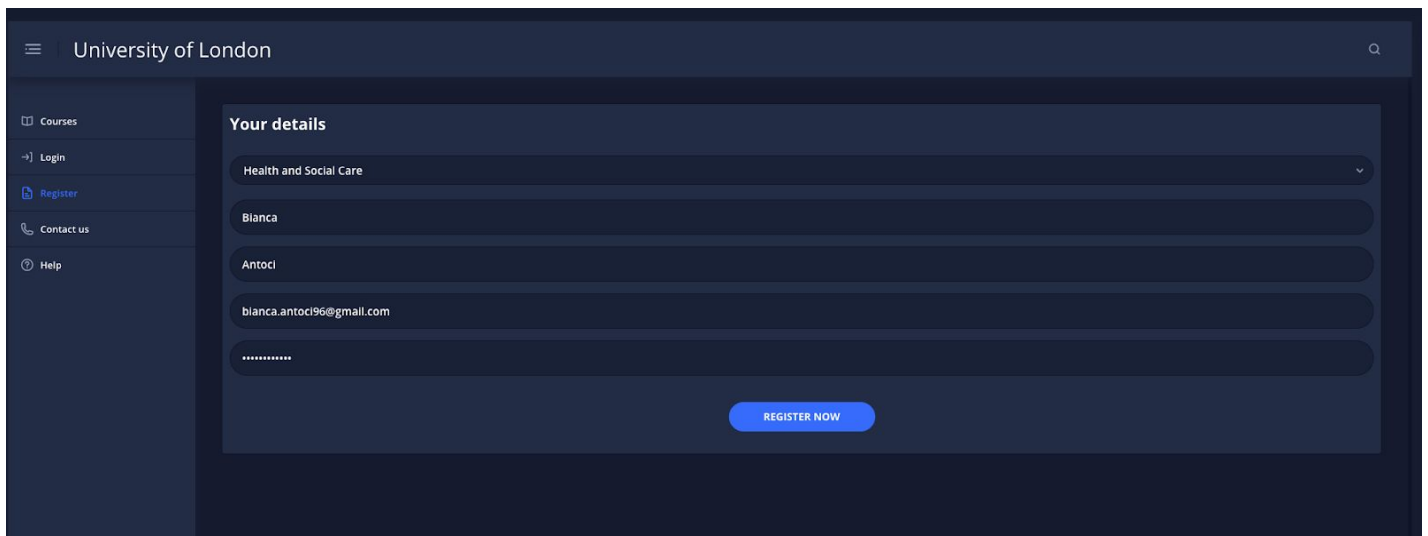


Figure 02

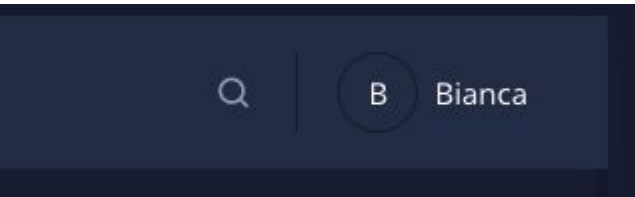


Figure 03

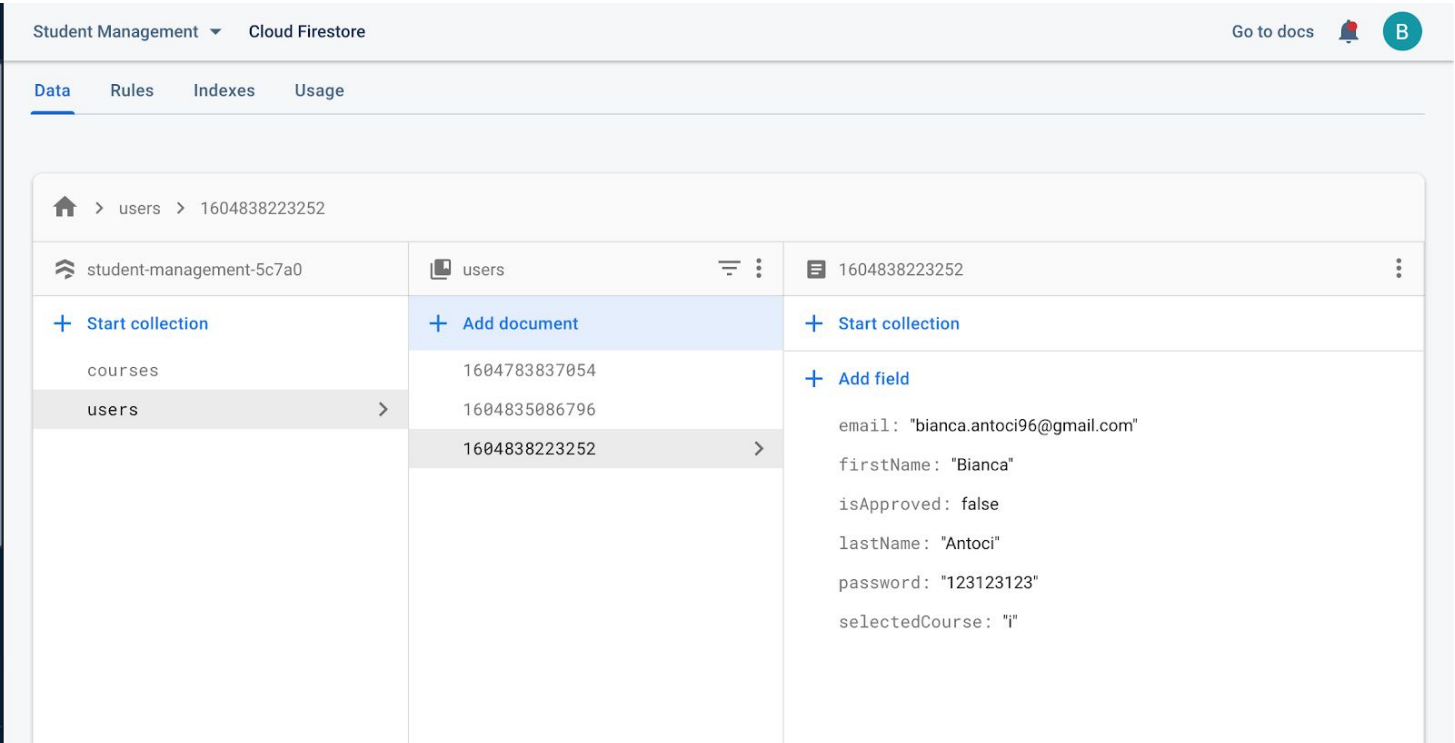


Figure 04

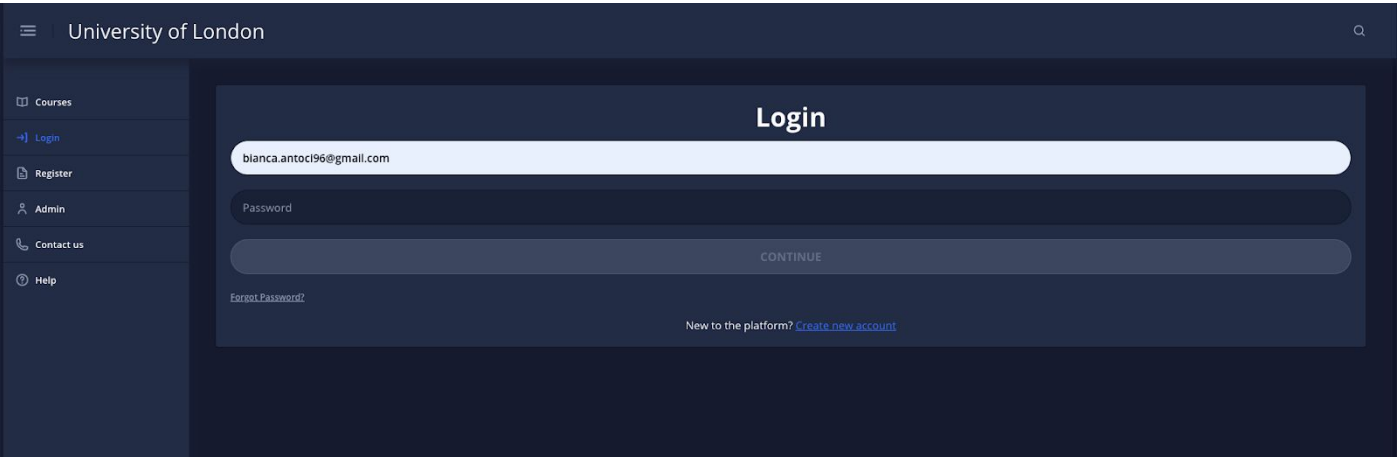


Figure 05

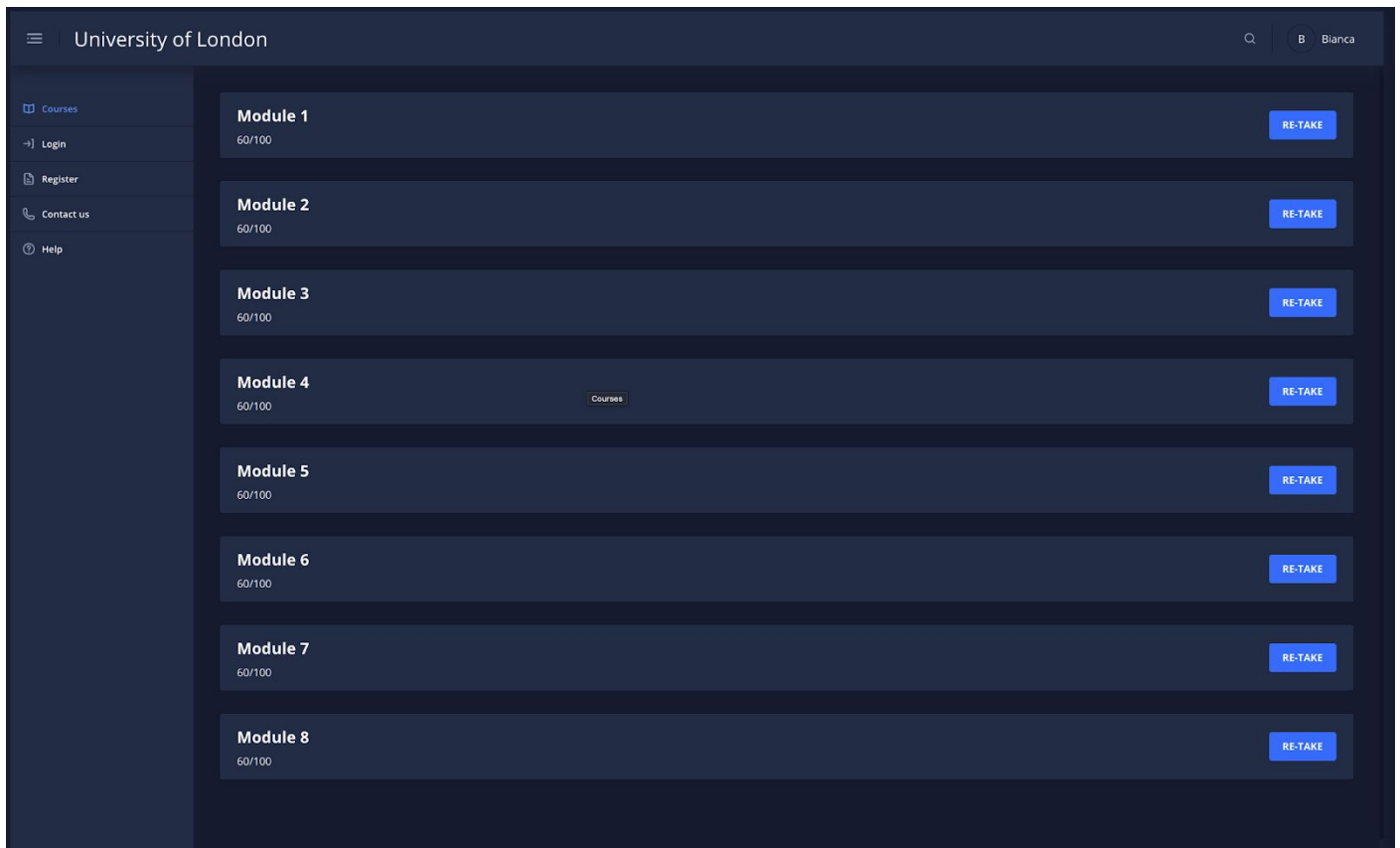


Figure 06

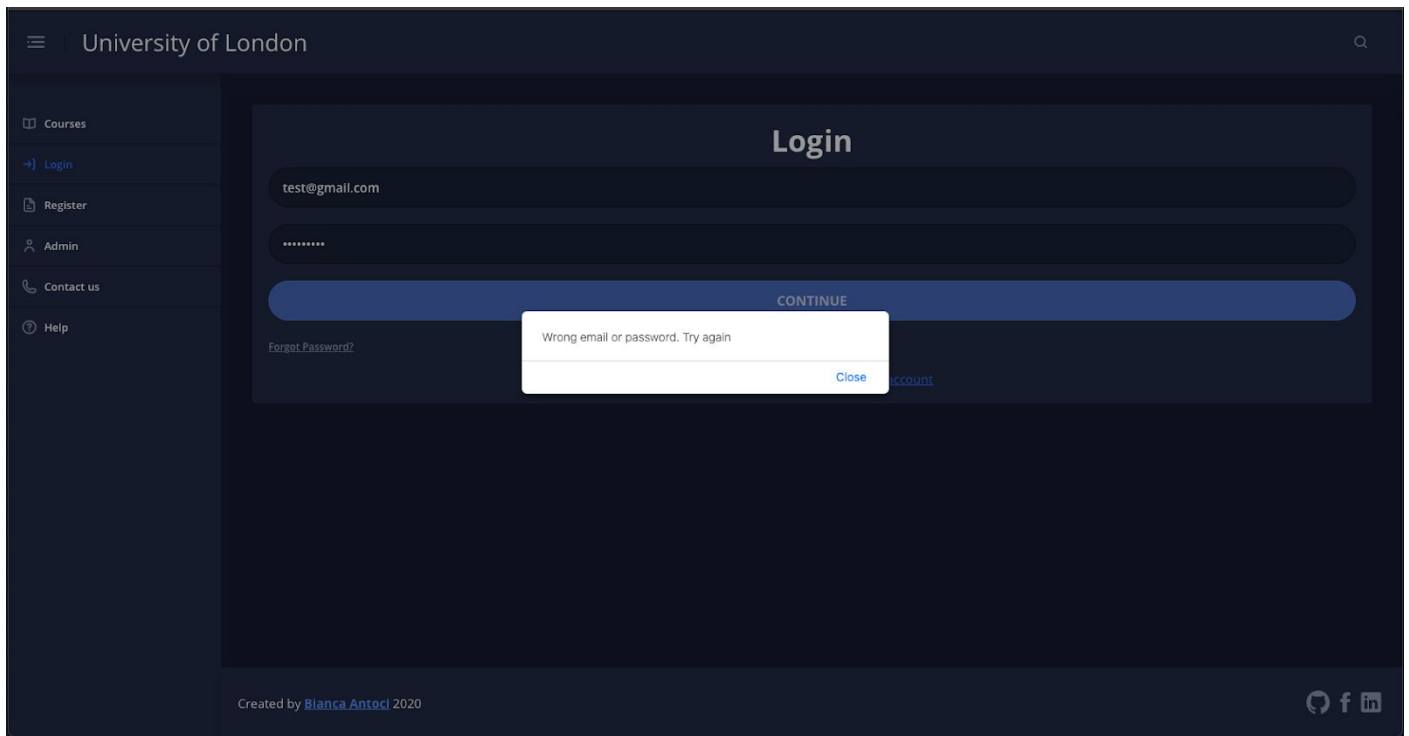


Figure 07

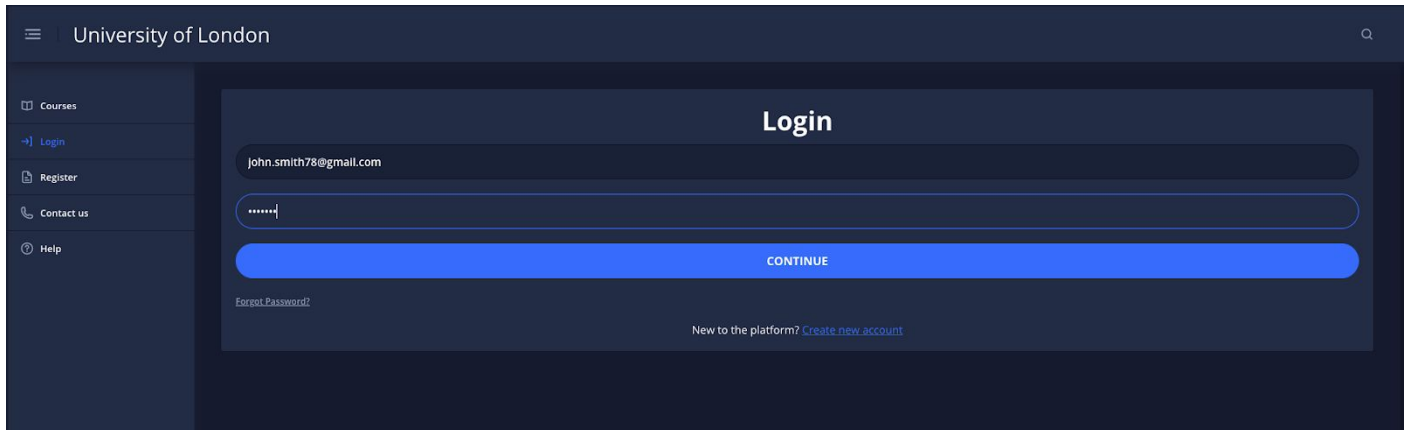


Figure 08

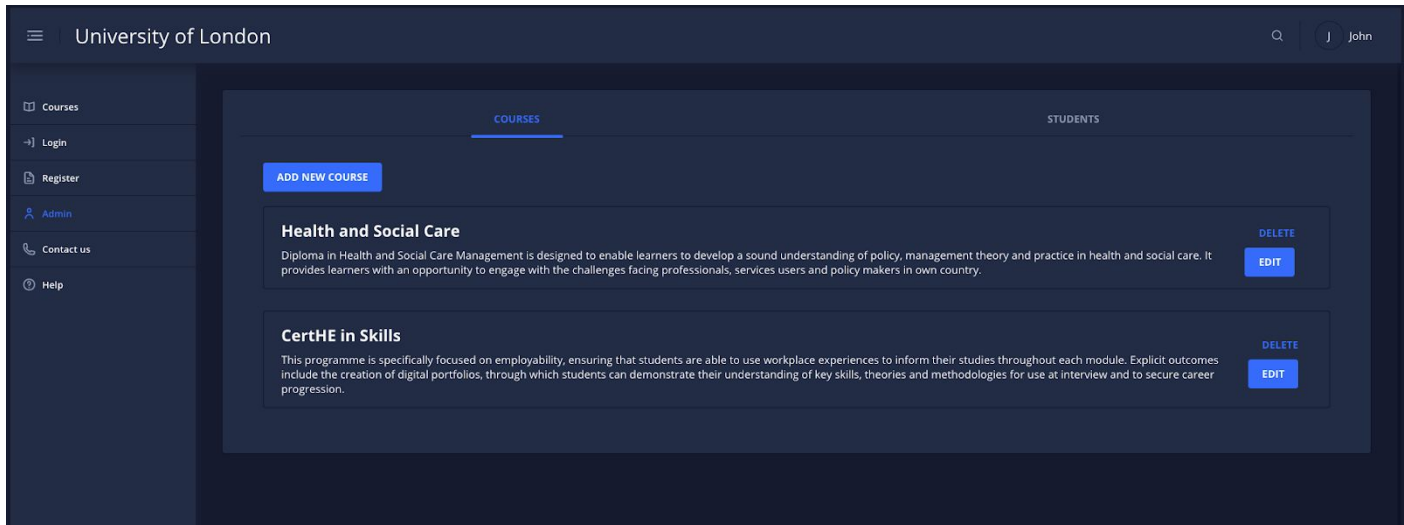


Figure 09

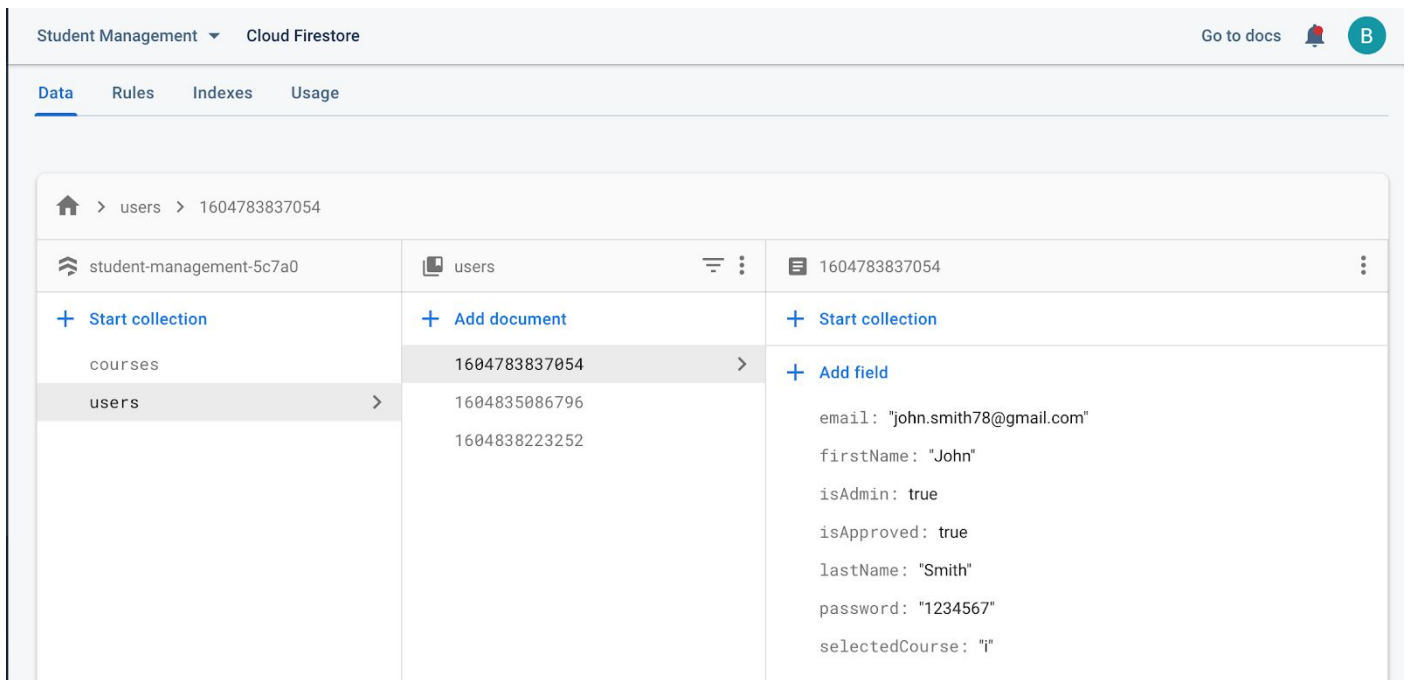


Figure 10

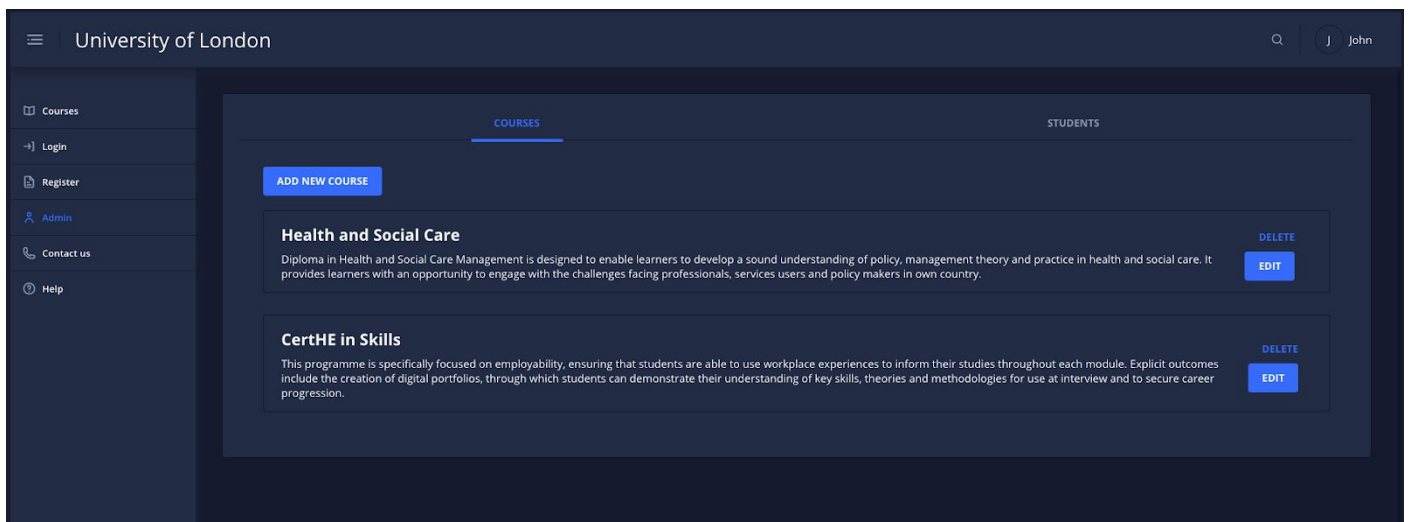


Figure 11

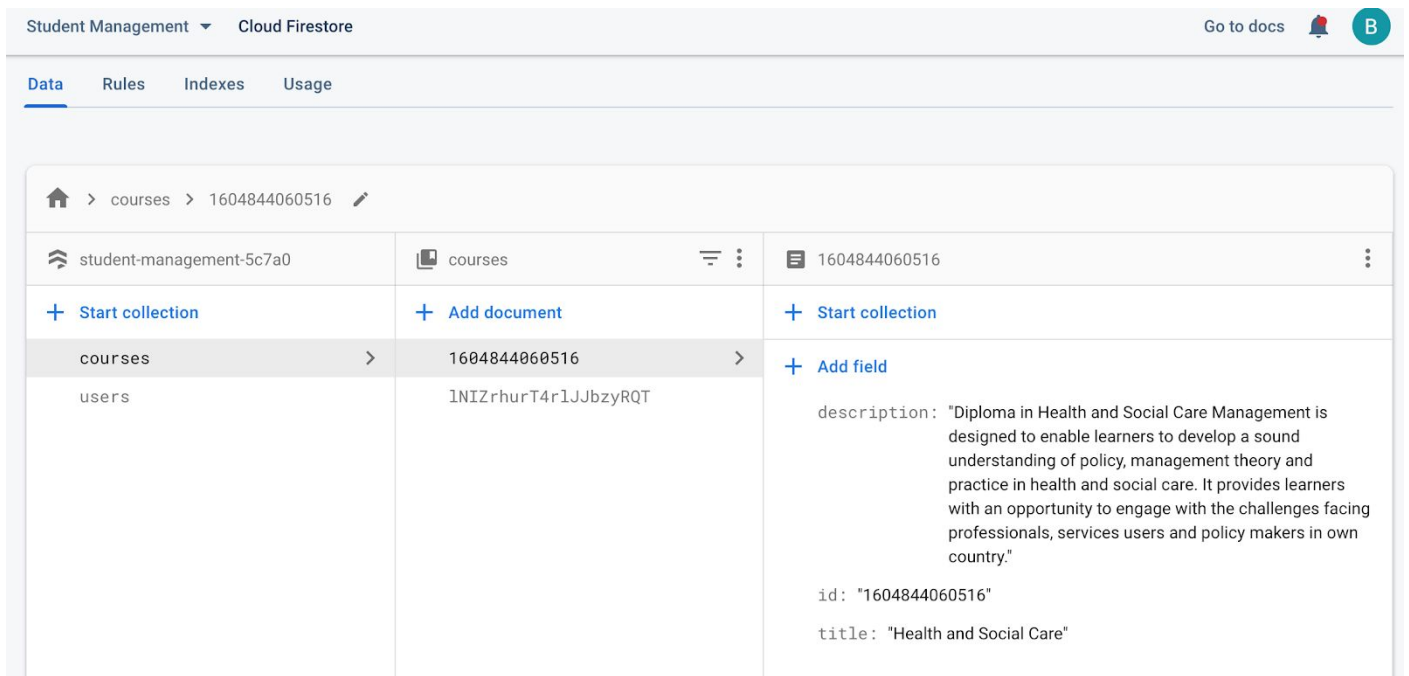


Figure 12

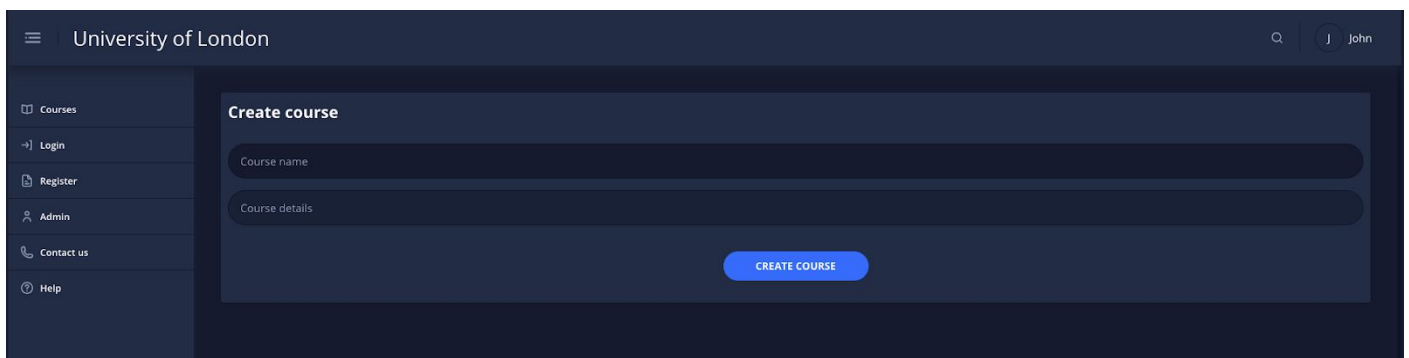


Figure 13

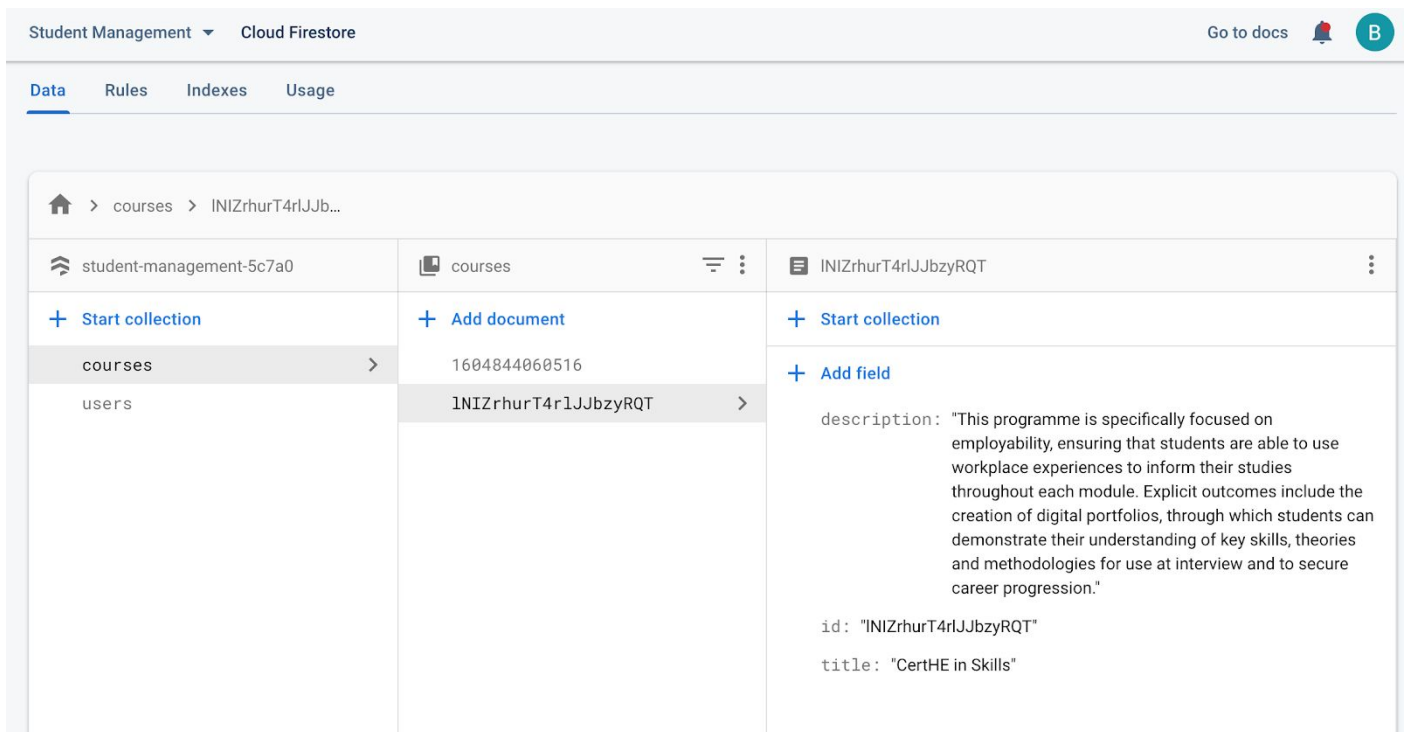


Figure 14

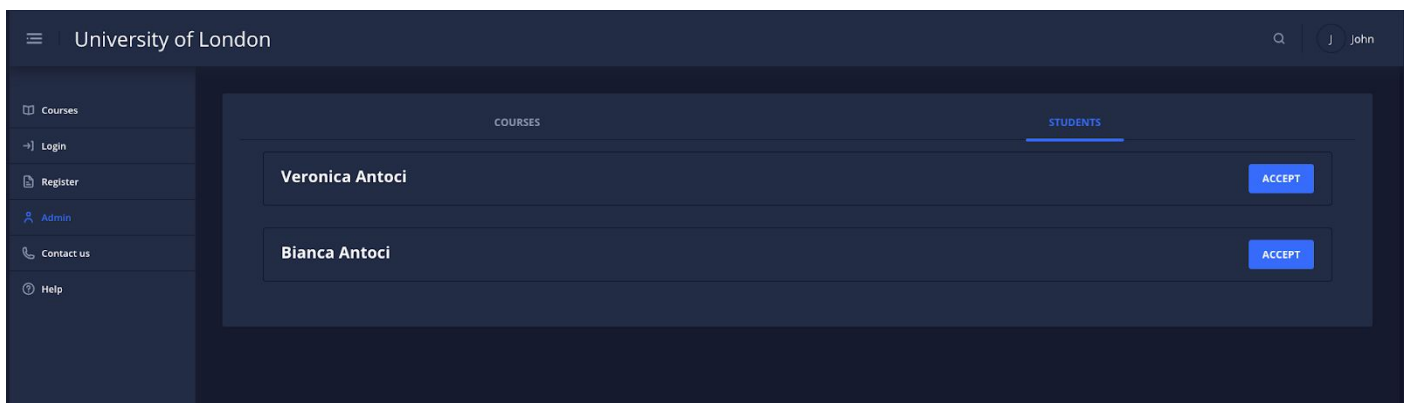


Figure 15

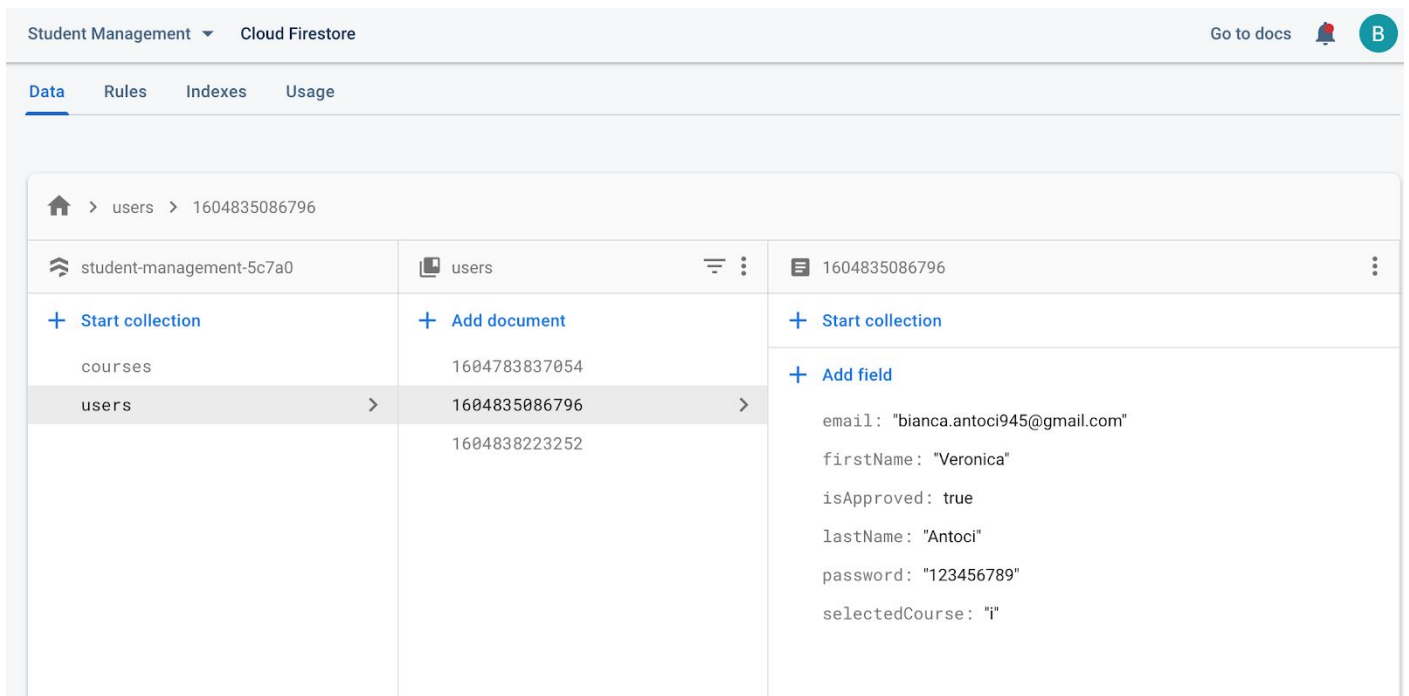


Figure 16

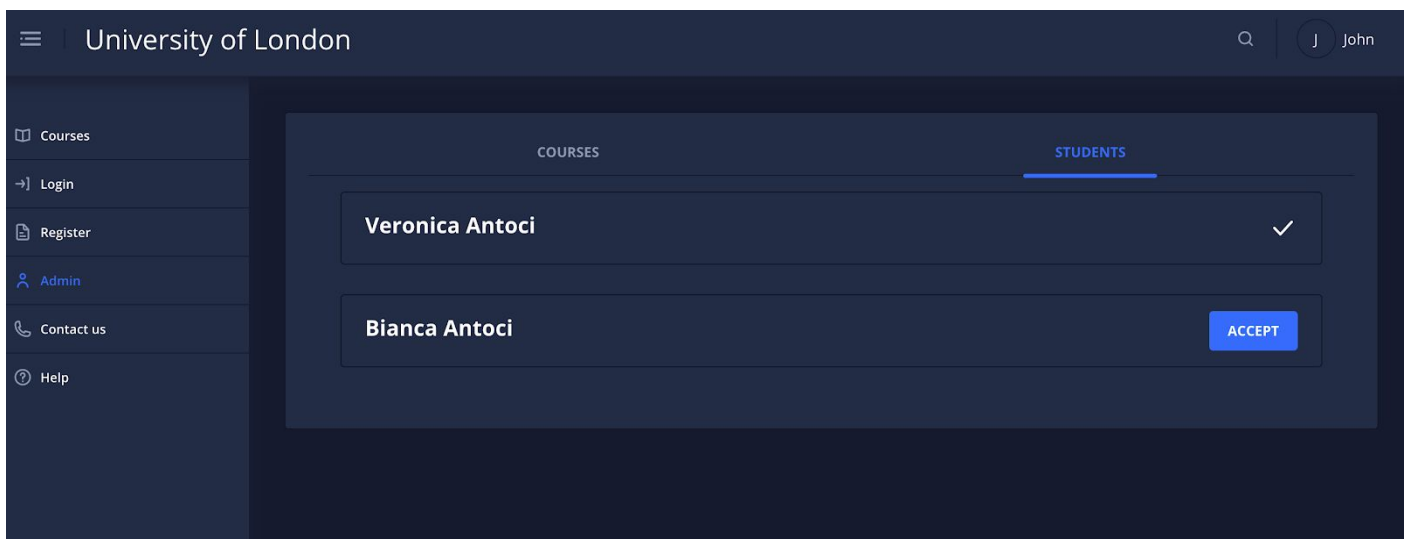


Figure 17

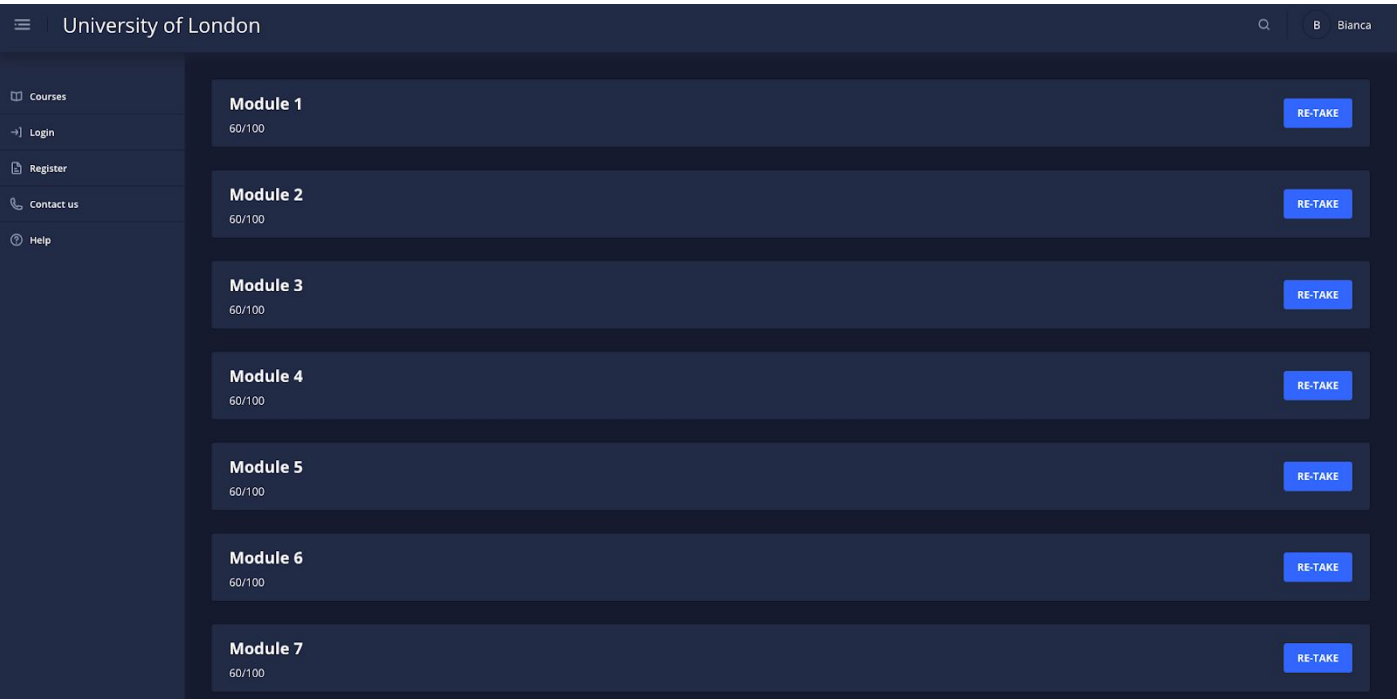


Figure 18

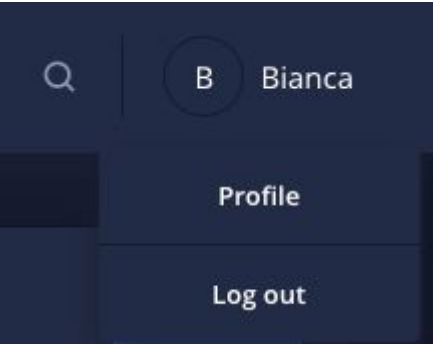


Figure 19

University of London

Q

B Bianca

Courses

Login

Register

Contact us

Help

Bianca

Antoci

bianca.antoci96@gmail.com

Course

Health and Social Care

SAVE

Figure 20

University of London

Q

B Bianca

Courses

Login

Register

Contact us

Help

Bianca

Antoci

bianca.antoci97@gmail.com

Course

Health and Social Care

SAVE

Figure 21

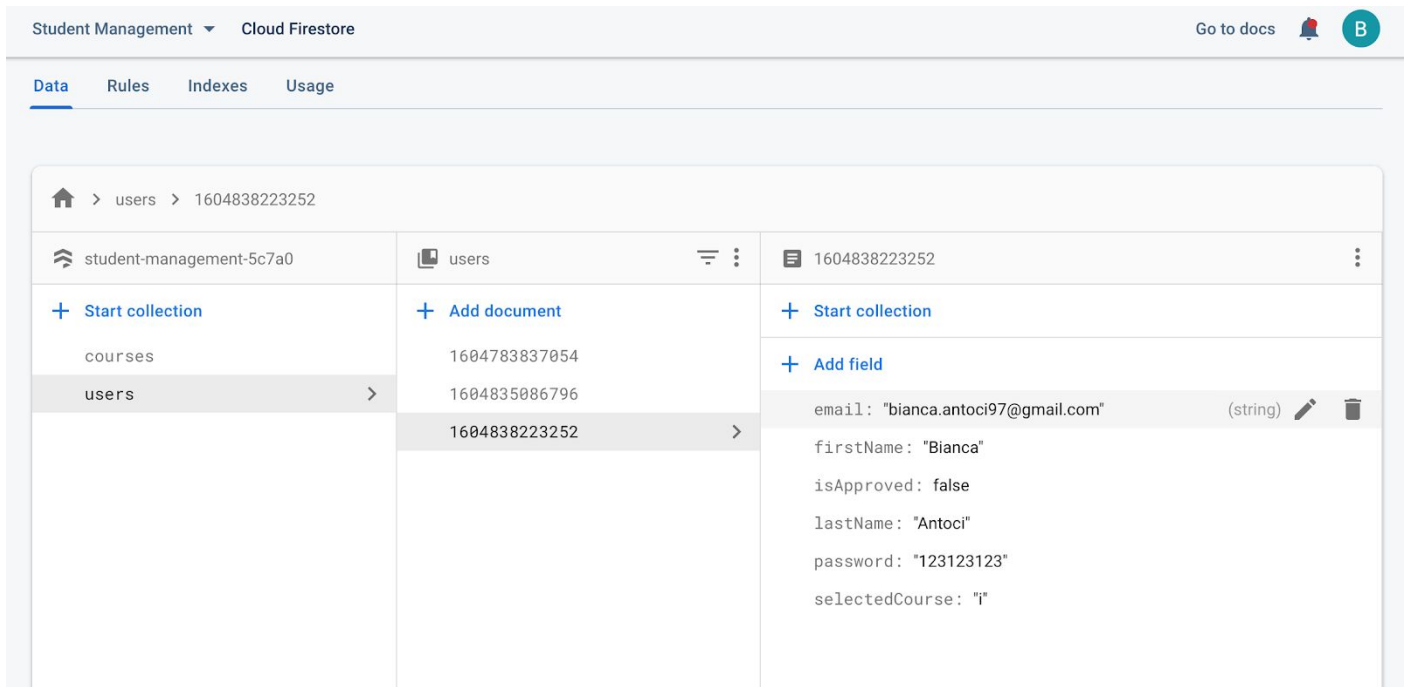


Figure 22

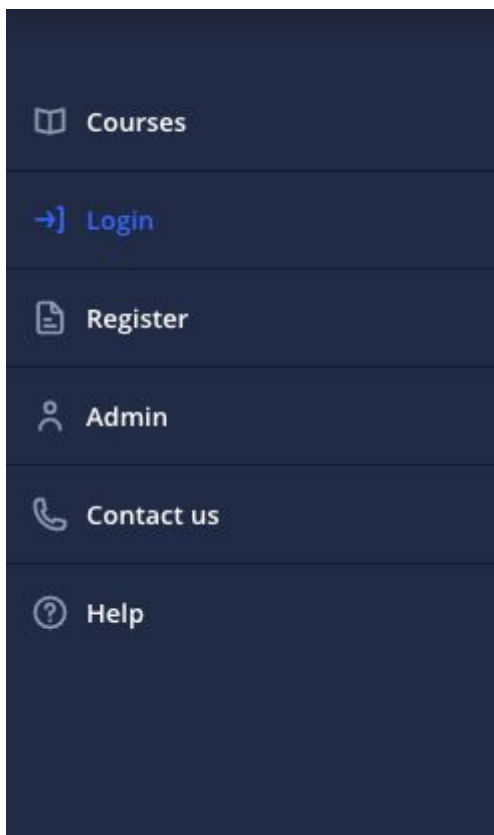


Figure 23

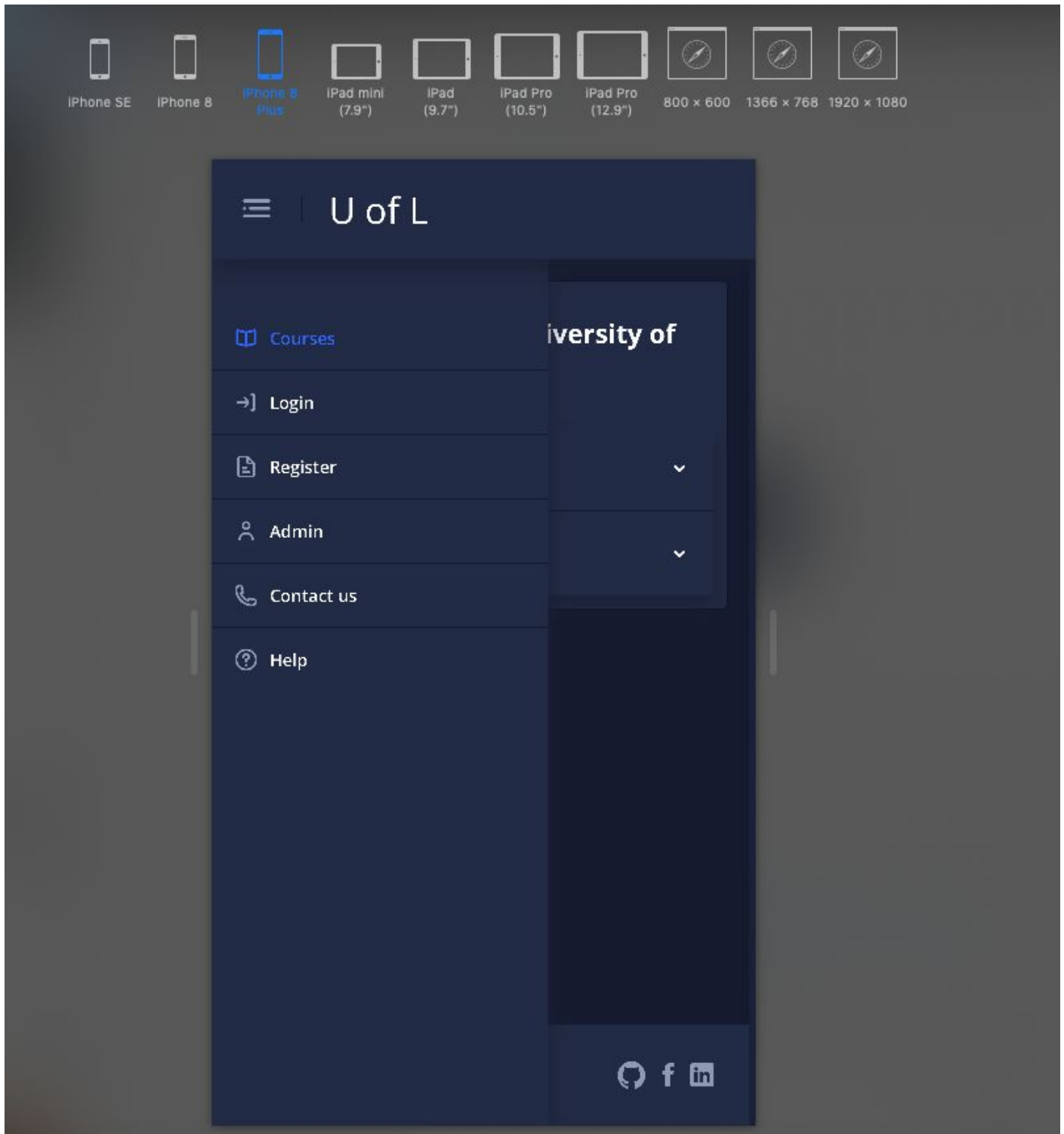


Figure 24

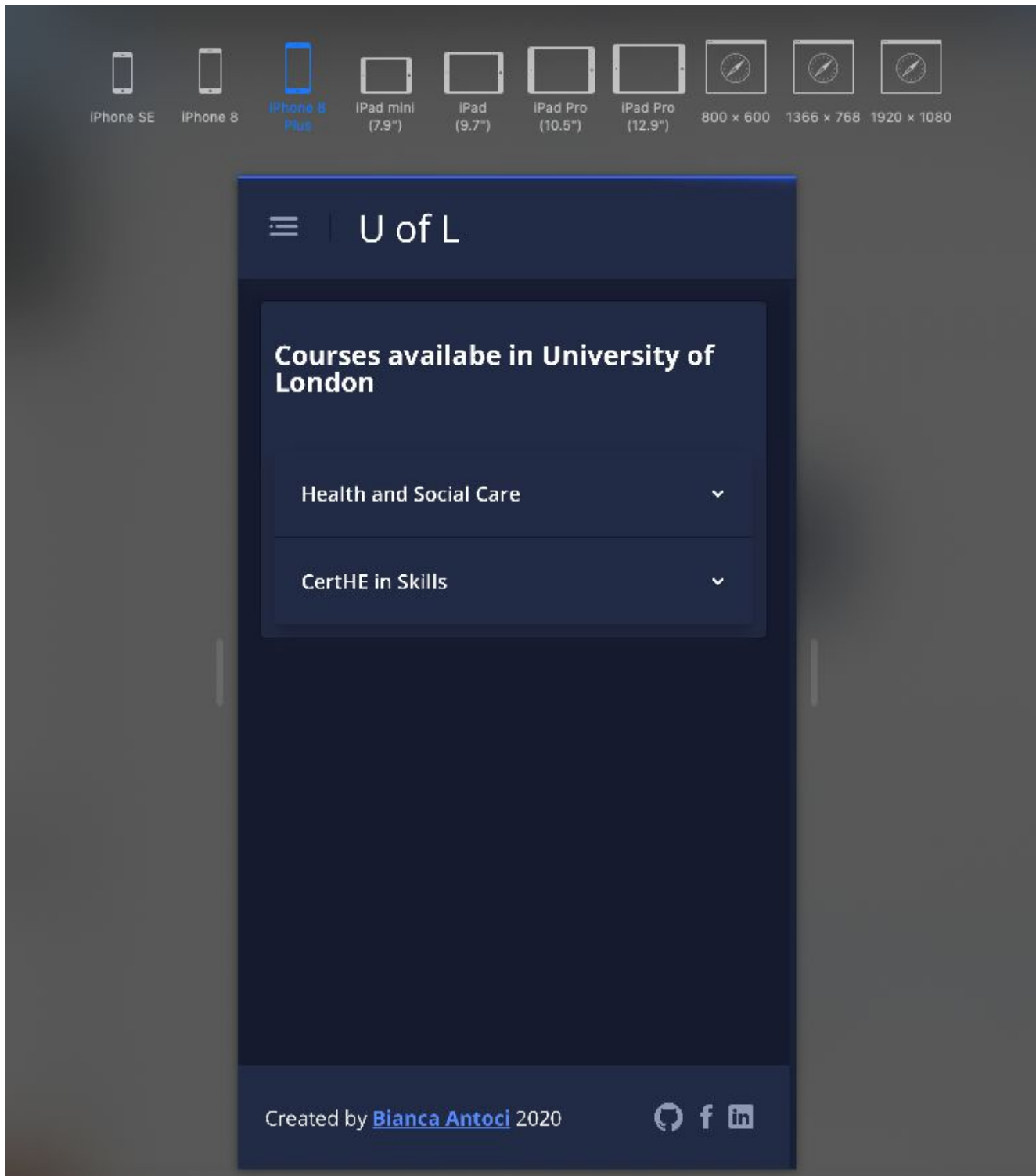


Figure 25

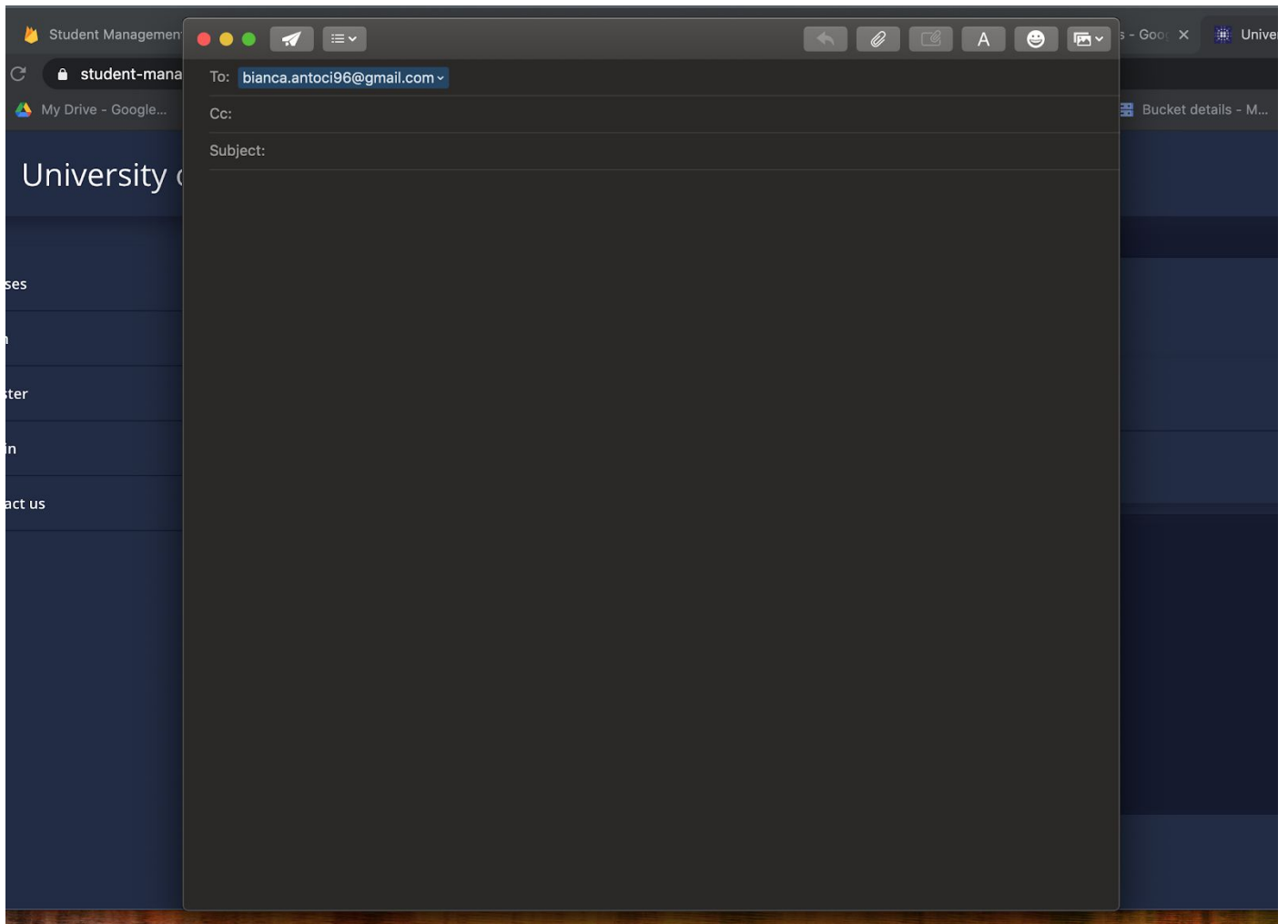


Figure 26

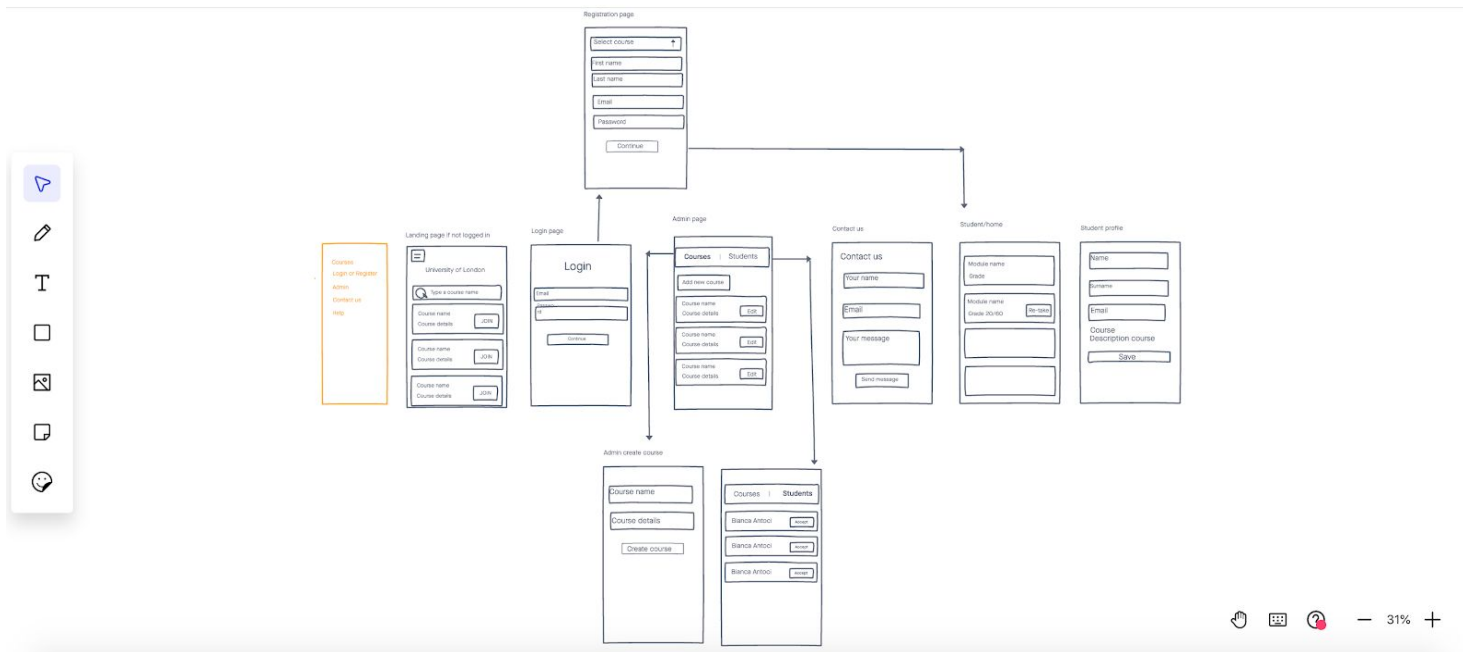


Figure 27