**TABLE OF CONTENTS**

1. **INTRODUCTION**
   1. Concept
   2. Project Scope
2. **PLANNING**
   1. Approach
   2. Agile Development Techniques
   3. Group Management
   4. Technologies Used
3. **RESEARCH & ANALYSIS**
   1. Early Assumption Testing
   2. Market Analysis
   3. Stakeholder Analysis
   4. Design Heuristics
4. **APPLICATION REQUIREMENTS**
   1. Functional Requirements
      1. System Requirements
      2. User Requirements
   2. Non-functional Requirements
5. **PROTOTYPING & ITERATION**
   1. Concept
   2. User Interface
   3. Name
   4. Homepage Page
   5. Reviews
   6. Profiles
6. **SYSTEM DEVELOPMENT**

**(Detail focused)**

* 1. Agile Project Techniques
  2. Development of Source Code
     1. Front-end
     2. Back-end
     3. APIs
     4. Data Structures
  3. Test-driven Development
  4. Unit Tests
  5. TDD Practice

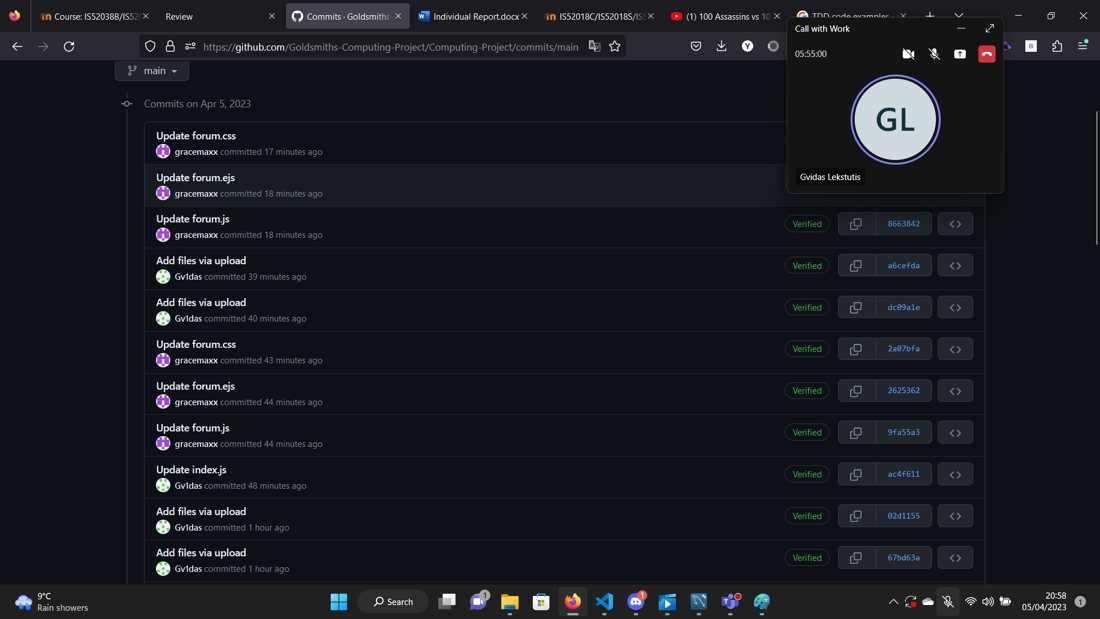
1. **DISCUSSION**
   1. Thematic based analysis
   2. Concept based analysis
2. **EVALUATION**
   1. Technical Difficulties
   2. Teamwork and Management
   3. Aims Reached
   4. Going Forward
3. **CONCLUSION**
4. **USER GUIDE**
5. **REFERENCES**

**GitLab Link**

<https://github.com/Goldsmiths-Computing-Project/Computing-Project/tree/main/Html>

**Commit Logs:**

[**https://github.com/Goldsmiths-Computing-Project/Computing-Project/commits/main**](https://github.com/Goldsmiths-Computing-Project/Computing-Project/commits/main)



1. **INTRODUCTION**

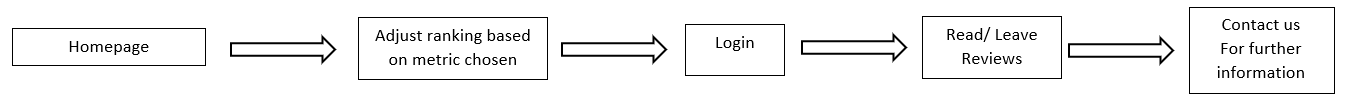
**1.1 Concept**

As per our proposal, our aim is to develop a web application that ranks the universities in London. Unlike other university ranking sites, we will use social aspects such as crime, transportation, and societies in addition to courses and modules, to form of how we rank the universities. The users of this application can use this site to help them decide where they want to study in London.

We will try to use APIs to gather data from external sites to help us build a ranking system as we do not currently have any data of our own to build an accurate list. We will also use a star grading system based off reviews left by users to rank the courses and modules. Users will be required to log in via their university credentials to leave a review.

**1.2 Project Scope**

Before we could decide on a minimum viable product (MVP), we developed a backlog to help us prioritise what were the most key features needed for this web application to function as we intended. Core functionality would look as follows:



We then broke down what is done at each step:

Diagram

Description automatically generated

From this we prioritised the features that were most important, these are highlighted in grey.

Diagram

Description automatically generated

As a minimum viable product (MVP), we aim to produce a web application that allows the user to view the current rankings of universities in the UK. As we do not currently have any data of our own to base the rankings from, we will be using data from an external site (The Complete University Guide, <https://www.thecompleteuniversityguide.co.uk/league-tables/rankings>) which already has this information. They use the following metrics for their rankings; overall ranking, entry standards, student satisfaction, research quality and graduate prospects, they also allow the user to choose a category is more important to them and see the rankings based on their selection. The user should also be able to read reviews left by other users and be able to submit a review if they wish, we will also include a contact page. Below is a detailed breakdown of these features and addition features we would like to include should the initial MVPs be met.

**MAIN MVPS**

**Homepage**

This will be our landing page where the user will see an up-to-date list ranking the universities in the UK. This will be achieved using APIs. Here, they can sort via the 4 subheadings: Overall score, Student Satisfaction, Social and Graduate prospects. By default, the rankings will be sorted by Overall score.

**Reviews Page – Read reviews**

This page will allow the user to read reviews left by other users and see how they have ranked subjects or social aspects.

**Reviews Page – Submit reviews**

Within the read reviews page, there will be an optional text box for users to submit their own rankings, to do this the user will need to create a profile.

**Profile Page**

Users who wish to leave a review can create their profile here, providing a name and surname.

**Contact Page**

Users can contact any of the team members via the details given on the page. As a minimum we will include email, telephone, and an address.

**ADDITIONAL FEATURES**

**Profile Page**

If the initial MVP for this page is met, we will aim to develop the profile page, and include a login system that requires the user to enter their university credentials so that we can verify them. This will help avoid discriminatory reviews being left by the user. Having verified users also protects the integrity of the site and information submitted.

**Contact Page:**

The user will be able to see our contact information on this page.

1. **PLANNING**

**2.1 Approach**

The intention was to meet on a weekly basis, this would give us time to work on our individual tasks and the meetings would be an opportunity for troubleshooting and to review if our current plan were working or if any changes needed to be made.

* 1. **Agile Development Techniques**

We looked at agile development techniques to help us plan how we would work on this project. We decided that Kanban’s and Test-driven development (TDD) were important for us to achieve our end goal. Using a kanban, would allow us to visualise how much progress we are making and keep track of work that needs to be done.

A screenshot of a video game

Description automatically generatedWe created our first Kanban using a platform called Asana. Here, we can see who is responsible for what task, set deadlines for individual task, set, follow-ups task and much more.

Test driven development will be an important part of our development, used regularly to measure our progress and functionality of the web application. For each stage of the TDD we would need to write some tests that we expect to fail, run the tests against our software and refactor the code accordingly.

Diagram

Description automatically generated

We will also use User-feedback as part of our testing as we want to ensure the user requirements are being met and that we are working in line with the proposal.

**2.3 Group Management**

Our first group meeting was to discuss each other’s strengths and weakness, and how we can best utilise our skills to build our web application. As there were 7 members of the group, we decided it best to have 2 groups of 2, and one group of 3 working on separate tasks.

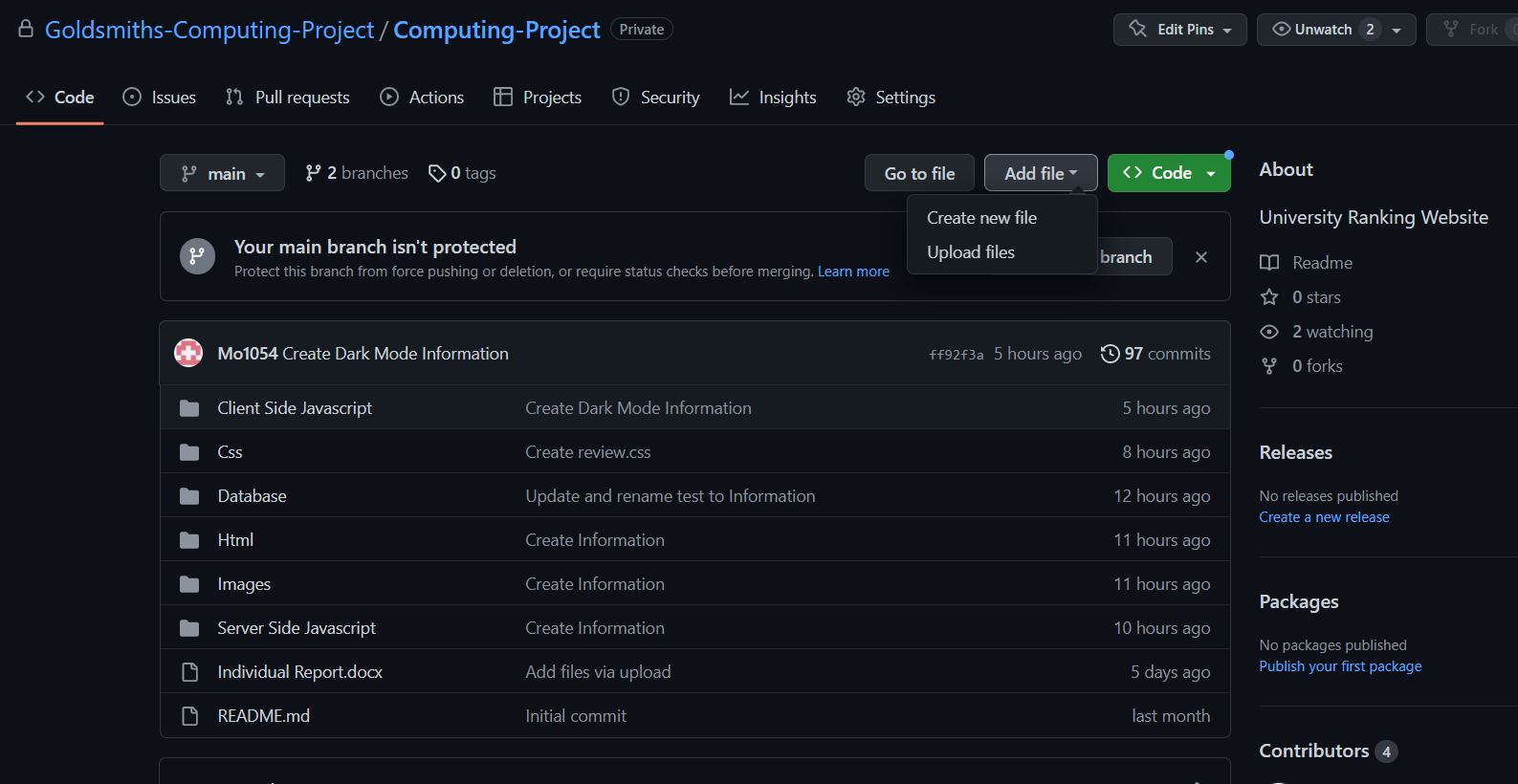
The first group would be working on the rankings page; this would be the landing page for the website and the first page users will see. We considered that the rankings could be linked to student reviews but as there will not be enough reviews initially to generate accurate rankings, we would need to do some research on APIs and try to work out how to use it within our software so that we can provide accurate rankings.

The second group will be working on the reviews page, this could be split into two separate pages; one for the reviews and another for the user to leave a review. We agreed that we would for now, try and create it as one page to meet our initial MVP.

We agreed that the forum page required more people working on it as it would be a more complex task than the others; we would need to build a database to hold the information in addition to creating the actual page in html.

**2.4 Technologies Used**

We are using GitHub for version control, with all group members added as collaborators. We created a repository and uploaded our iterations of our MVPS here.



1. **RESEARCH & ANALYSIS PLANNING**

**3.1 Early Assumption Testing**

We achieved our early assumption testing in our proposal, in the form of evaluating our stakeholder and user needs, prior knowledge and market survey. Our market and stakeholder analysis explains our findings and how we built on them as part of our development process.

* 1. **Market Analysis**

To build our market analysis, we looked at other university ranking sites to give us an idea of what features are used that we could apply to our own application. We then created a comparison table to help us understand what features would be important for the user’s experience.

Table

Description automatically generated

We referred to this table to decide what features to include and how it would look. We agreed that keeping the design of the site simple was important for ease of navigation for the user. If the user can navigate the site easily, it may make their experience more positive and encourage them to use the site again or recommend it to others. This decision was supported by the results from our questionnaire regarding the use of ranking sites.

Graphical user interface, application, Teams

Description automatically generatedIn our questionnaire we asked our subjects 6 questions relating to using university ranking sites.

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application, Teams

Description automatically generatedThe results relating to the design and navigation were as follows, based on 20 subjects.

* 1. **Stakeholders Analysis**

We initially identified our main stakeholders to be college leavers, or those returning to education, we achieved this through creating user profiles.

Diagram

Description automatically generatedA picture containing diagram

Description automatically generated

The user profiles will help us to keep in mind who we are building this web application for. We do not want to get distracted with building features that are not necessary to the user’s needs. Now that we are in the process of building the application, we have identified further stakeholders.

Developers

The developers will be responsible for building and maintain the application. Where we are the developers who will be working on this project, we want to keep our deliverables realistic and achievable based on our current skill levels.

Testers

The testers will be used at various stages of the development process, for example the design stage and for user testing. They will mostly be made up of our friends and family, and their feedback will help us identify any barriers, how we can resolve them and if we are on track to meeting our MVPS.

**3.4 Design Heuristics**

When thinking about design heuristics we wanted to make sure that our web application met the following criteria by achieving the following targets below. By using design heuristics, we can approach problems in a systematic way and generate solutions that are more likely to meet the needs of users and stakeholders. This should result in a better design outcome.

Simplicity

We want to keep the design of our web application simple, so that it is more accessible and easier to understand and navigate for the user.

Feedback

The design should provide clear feedback to the users, to help them understand the impact of their actions whilst navigating the site and how the system works.

Affordances

We want to ensure that the interface is clear and intuitive to the user.

Consistency

This is an important aspect of our design, it needs to be consistent in its colours, typography, and layout, to create a unified and cohesive visual expression.

Error prevention

We want our design to guide users towards correct actions and prevent mistakes, this ties in with the simplicity; having an easy to navigate system will help prevent errors by the user.

1. **APPLICATION REQUIREMENTS**

**4.1 Functional Requirements**

Functional requirements are a set of specifications that define what a software system should so. These requirements will describe the functions, capabilities and features our system must have to meet the needs of its users. As we move through iterations of the development, the requirements may change. Sections marked \* will only be implemented should the initial MVPs be met.

**4.1.1 System Requirements**

We are creating a responsive web application that will be accessible to users via a browser on PC or smart phones, there are also no hardware requirements. The site will be accessible to users with disabilities, adhering to standards such as Web Content Accessibility Guidelines (WCAG). Further information on this can be found at <https://www.w3.org/WAI/standards-guidelines/wcag/>

**4.1.2 User Requirements**

Here we outline how users will interact with the web application, and the requirements needed to achieve a fluid user experience.

Navigation

* The site will be easy to navigate, there will be a clear, intuitive menu and a search bar to allow the user to quickly find the information they need.

Homepage

* The user can see the rankings list in full, this will be sorted by overall score by default.
* The user can sort the ranking according to either, Overall, Student Satisfaction, Degree Completion, Academic Services Spend, Facilities Spend.

Forum Page

* The user can view and engage in discussions with other users, about topics posted to the forum.

Review Page

* The user will be able to read, and post reviews and ratings of the universities and can include comments about social aspects, the course and other relevant topics.

Sign up Page\*

* The user will be able to register a profile to the site, using their university credentials, they will then be able to log in via a username and password. This will allow them to submit reviews and ratings.

Contact Page

* Users will be able to see the contact details for the ‘owners’ of the web application, this will include emails and names.

**4.2 Non-functional Requirements**

These are the characteristics that are not related to specific functions or features, but rather quality, performance and user experience.

Usability

* The site will be user friendly and easy to navigate, with clear and intuitive interfaces.
* The main language will be English, with easy-to-understand words and sentences.
* Visual aspects of the site will be appealing and consistent throughout the site.

Reliability & Performance

* The site will be reliable and available to the user for access, with minimal downtime or outages.
* The site should perform quickly and efficiently, with minimal load and response times.

Security

* The site will be secure, with measures in place to protect user data and prevent unauthorised access.
* User will be required to enter their university credentials to set up their profile.
* All users will have a unique ID
* Passwords are created upon setting up their profile, this is never displayed in the application.

Accessibility

* As stated previously, the site should be accessible to users with disabilities, adhering to WCAG.

Maintainability

* The site will be designed to be easy to maintain and update, with clear and well documented code.

Compliance

* The site should comply with the relevant laws and regulations relating to privacy, data protection and security.
* Inappropriate and offensive content will be filtered.

1. **PROTOTYPING & ITERATION**

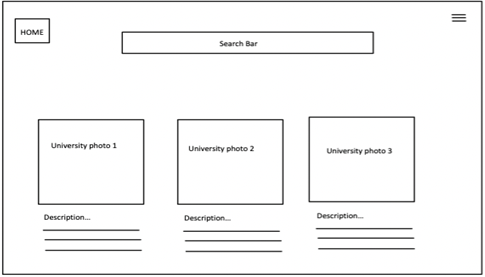
**5.1 Concept**

As outlined in the proposal, the objective of this project is to develop a web application that ranks universities in London; to help users choose a place of study that suits them best. The website will focus on ranking by social aspects such as crime, transportation, and societies. In addition, there will be rankings by academic specifications such as courses, modules, and lecturers.

We have kept in line with our initial concept but made a few changes to how many categories we will aim to develop; these are outlined in our main MVPs. Some of the iteration processes for each MVP are detailed below.

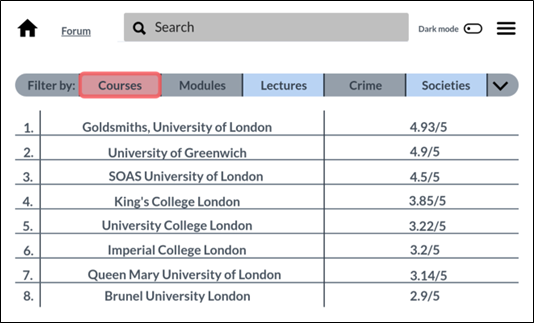
**5.2 Homepage**

Our first idea of how the homepage would look was as follows with a low-fidelity wireframe. We would have each university listed with a short description of it.

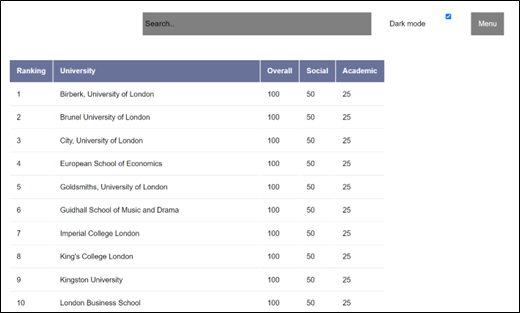


We then looked at the feedback from our questionnaire, in particular the comments regarding navigation and information, we decided that we needed to make the homepage more visually engaging as the first point of contact, and that the necessary information was being projected to the user.

Our high-fidelity wireframe (below) shows the ranking list as the main point of information on the homepage. Here the user can straight away see the rankings before having to do anything else. That way, if all they wanted was to see the rankings, their requirements would be met without any further navigation. If they wanted to sort by the different categories, they could do this within the same page also. We decided to add a dark mode option as this has become an increasingly popular feature with users.



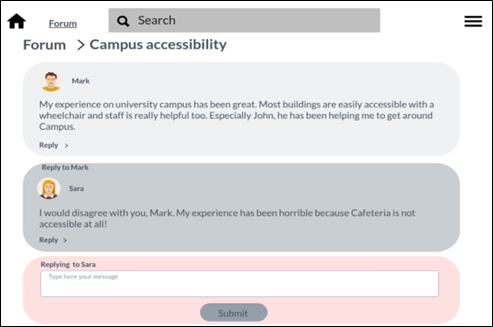
When it came to converting the wireframe into a built html, this was our first technical prototype (below). The layout is the same as what was outlined in the wireframe; the ranking table is visible with the categories and scores listed, we also kept the search bar, dark mode, and menu. The data presented was used as a placeholder as the APIs had not yet been integrated at this point.



The intention was to use APIs from an already built university ranking site as we did not have our own data yet to have a functional ranking list. However, research into APIs and how they work proved this to be a difficult task. The site we had intended to use for the data, <https://www.thecompleteuniversityguide.co.uk/> did not have the APIs available meaning we were unable to pull the data needed.

**5.3 Forum page**

The high-fidelity wireframe for the forum page shows one of the topics selected and how it would look when users post to the page. Users are also able to view and reply to posts made by other users.



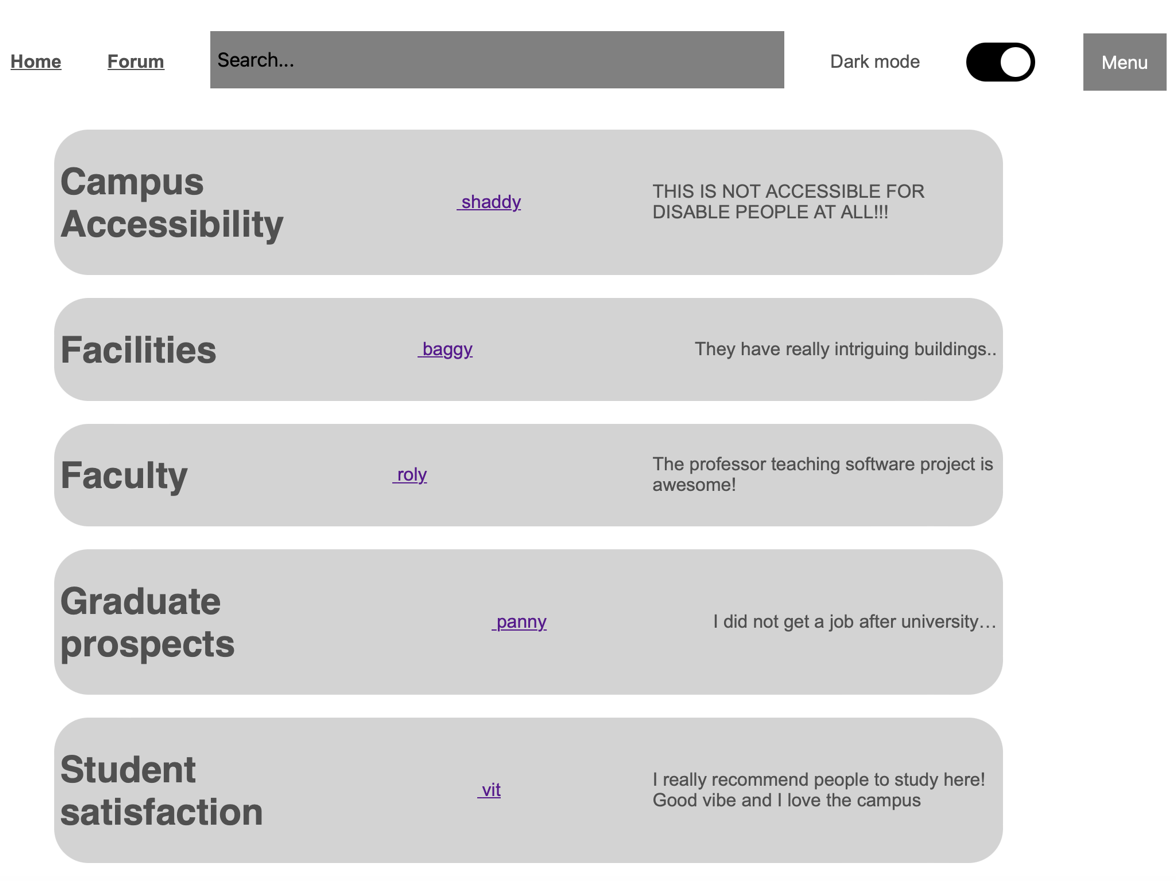
This was translated into the first technical prototype (see below).

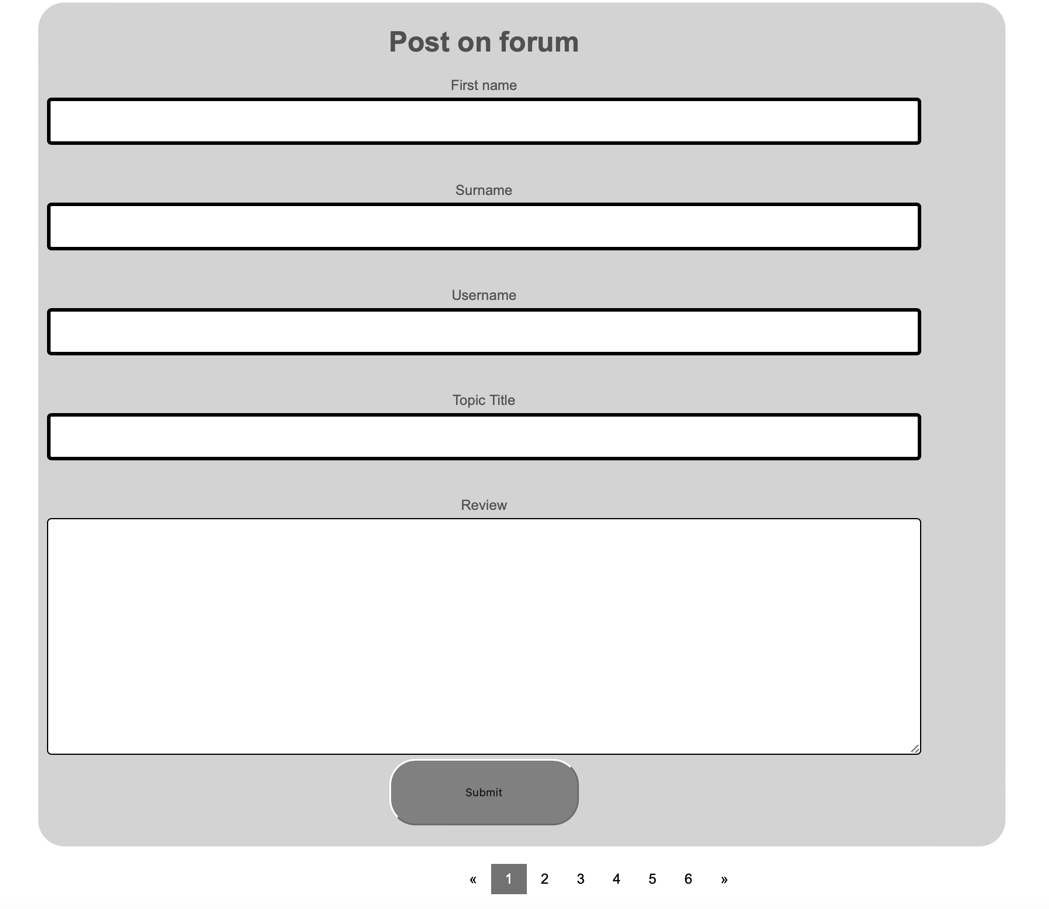
Graphical user interface, text, application, chat or text message

Description automatically generated

The forum website can connect with the database and present it on the website.

Also, users can post a new thread on the forum after entering their first name, surname, username, topic title and the content.





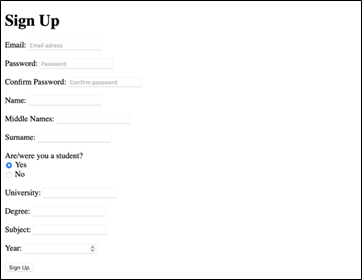
**5.4 Login and User Page**

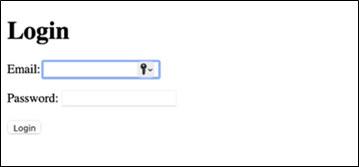
In the development of our web application, we focused on creating an engaging and user-friendly experience. This was achieved through the implementation of critical features such as the sign-in, sign-up, and profile pages.

Initially, we started by building these pages using simple HTML code. This provided a basic structure for each page, allowing us to conceptualize the layout and overall design.

Although we had no wireframes for the Login and User Page, we created a technical prototype, keeping in mind the design features and layouts used in the other pages produced. We kept the pages simple and easy to read for the user. The first page created is for new users who are not previous students. The next page is for users who select yes to the questions ‘Are/were you a student?’. The last page is the login page for users.





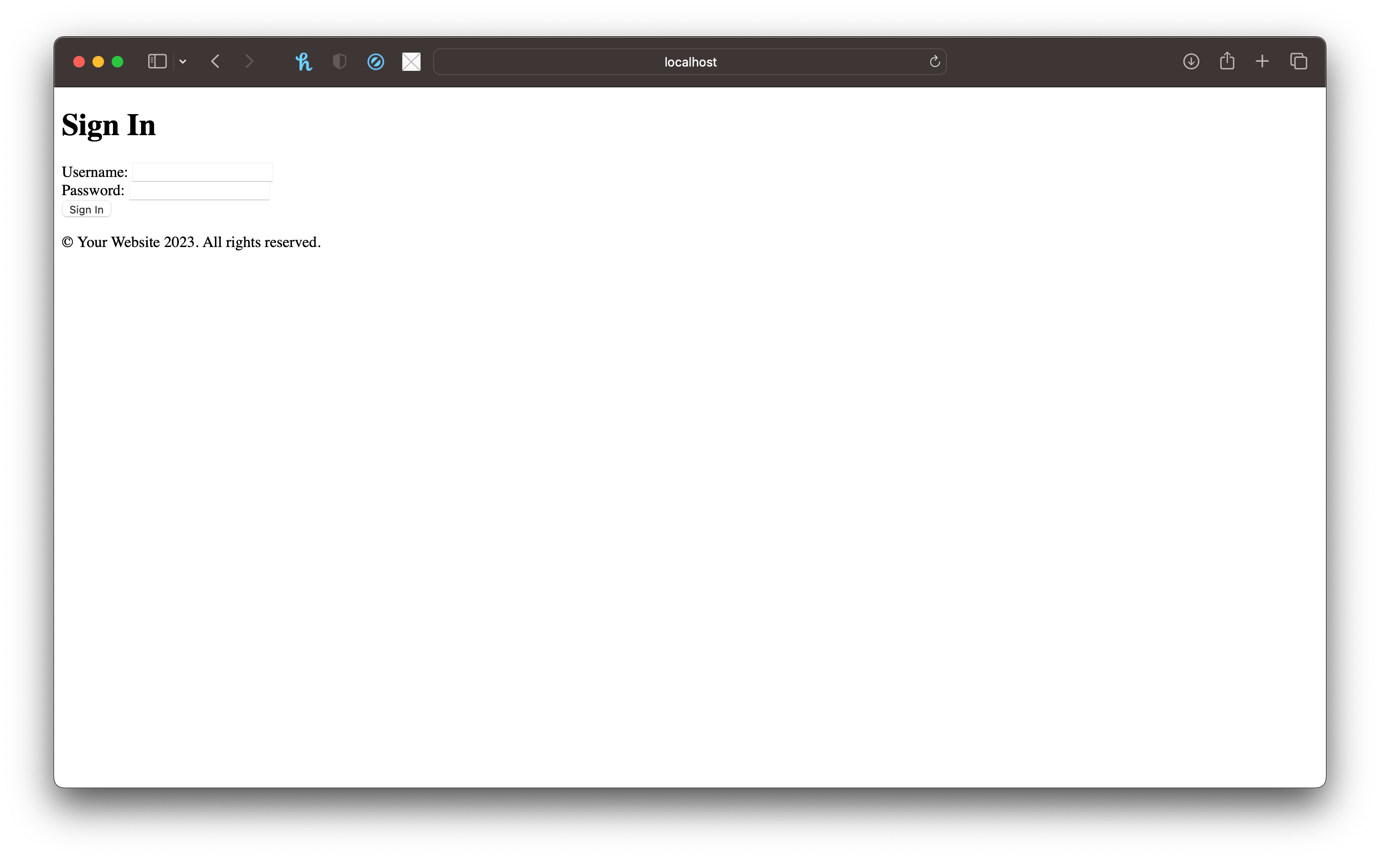


As the project evolved, the developer responsible for the sign-in, sign-up, and profile pages incorporated JavaScript to enhance the user experience and add interactivity to these specific pages. JavaScript enabled the creation of dynamic and responsive elements that facilitated seamless user interaction.

Furthermore, this developer connected the sign-in, sign-up, and profile pages to a MySQL database. This allowed for the secure and efficient storage and management of user data. The integration of the database with these pages ensured that the application could authenticate user credentials and display personalized information on the profile page.

As the design of these pages became more complex, the developer decided to implement CSS to refine the appearance and enhance the visual appeal. This allowed for a consistent look and feel across these pages, improving the overall user experience.

To promote modularity and maintainability in the code, the developer chose to split these pages into separate layout, footer, navbar, and content files. This decision allowed for the reuse of common elements across different pages, significantly reducing code duplication and making the codebase more manageable. Moreover, this separation of concerns facilitated easier updates and modifications to the design and content of these pages.

Later, the team had to merge the data from the two databases, which proved to be a lot of work. By prioritizing both frontend and backend designs for these specific pages, the developer was able to create a cohesive and user-friendly web application that successfully catered to the target audience's needs.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Updated pages with CSS:

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

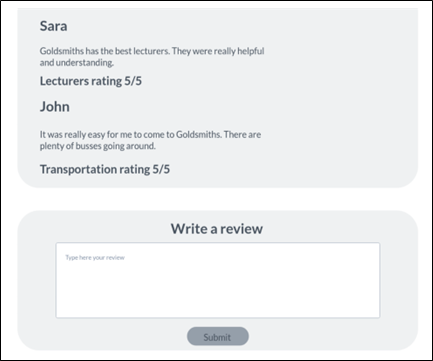
Description automatically generated

Graphical user interface, text, application

Description automatically generated

**5.5 Reviews**

The idea for the review was to keep it simple as shown in the high-fidelity wireframe below. Users will at first see any posts left by other users, they will also have the option to write their own review. Users will need to log in to their profile, to be able to submit their review.

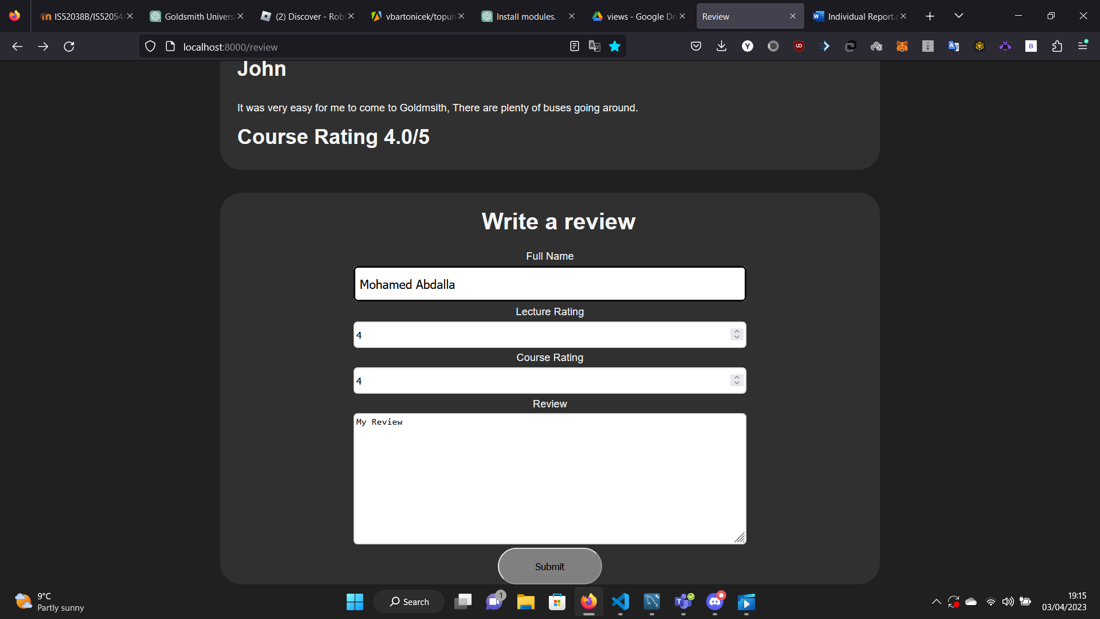


First technical prototype looks like what we had outlined in the prototype. We have kept the layout the same and tried to keep it consistent with the layout for the homepage and forum page. We do not yet have the option to submit a review added.

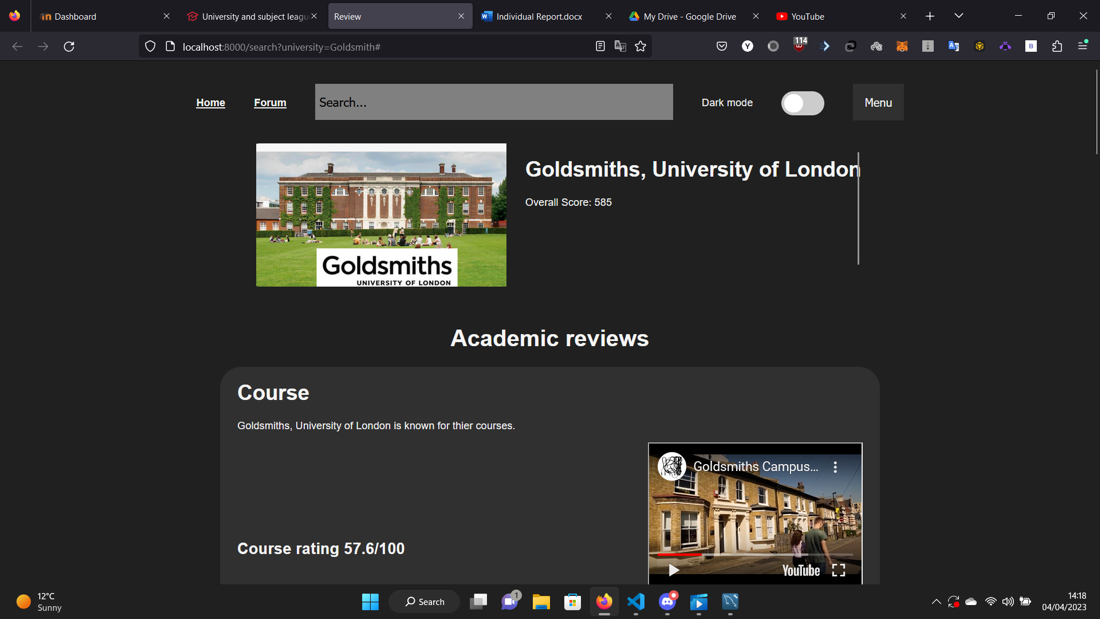
Graphical user interface, application

Description automatically generated

In the second iteration of the reviews page, the submit a review section was updated to make the submit button functional, CSS was also added to improve the look of the section.



We also dynamically changed the information on the page based on what was searched:



1. **SYSTEM DEVELOPMENT**

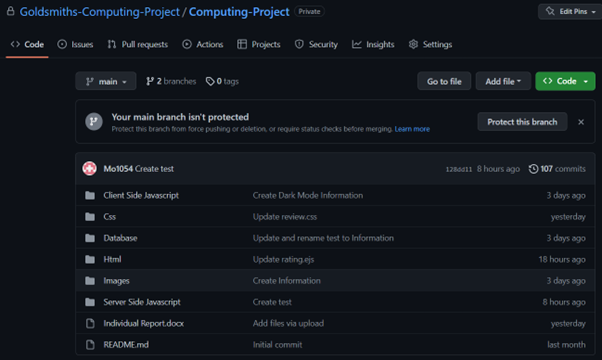
**6.1 Agile Project Techniques**

The goal of agile development techniques is to help teams work more collaboratively, respond quickly to change and deliver high-quality software that meets the needs of the user. It can be used in several ways depending on the needs of the project. This chapter will outline how we used the agile principles and applied them to our software development process.

* 1. **Development of Source Code**

We created a GitHub repository to maintain our source code and allow version control. All members had access to the repository and were able to commit any changes as and when needed. Our repository contains the following folders and files:

* Client-side JavaScript
  + Dark Mode information
  + Information
  + Review.js
* CSS
  + Grid Layout Information
  + Information
  + forum.css
  + home.css
  + review.css
  + styles.css
* Database
  + Information
  + create\_uni\_table.sql
  + create\_user.sql
  + review\_table.sql
* HTML
  + Information
  + \_footer.ejs
  + \_layout.ejs
  + \_navbar.ejs
  + \_profileContent.ejs
  + \_signinContent.ejs
  + \_signupContent.ejs
  + contact.ejs
  + forum.ejs
  + homepage.ejs
  + profile.ejs
  + review.ejs
  + signin.ejs
  + signup.ejs
  + sort-Institution-asc.ejs
  + sort-Institution-dsc.ejs
  + sort-academic\_services\_spend-asc.ejs
  + sort-academic\_services\_spend-dsc.ejs
  + sort-degree\_completion-asc.ejs
  + sort-degree\_completion-dsc.ejs
  + sort-facilities\_spend-asc.ejs
  + sort-facilities\_spend-dsc.ejs
  + sort-overall\_score-asc.ejs
  + sort-overall\_score-dsc.ejs
  + sort-ranks-asc.ejs
  + sort-ranks-dsc.ejs
  + sort-student\_satisfaction-asc.ejs
  + sort-student\_satisfaction-dsc.ejs
* Images
  + Information
  + university.png
* Server-side JavaScript
  + Node modules
  + Information
  + contact.js
  + forum.js
  + homepage.js
  + index.js
  + profile.js
  + review.js
  + signin.js
  + signup.js
  + validation.js



**6.2.1 Front-End**

For our web application, we have designed the HTML and CSS code for the front-end layout of our website. we have used a combination of HTML5 and CSS3 to create a responsive and interactive web layout. In this section, we will discuss the advantages and disadvantages of the HTML and CSS code that we have created.

**Advantages of Grid Layout**

One of the main advantages of using Grid Layout in CSS is the ability to create complex layouts with ease. Grid layout enables the developer to create multi-dimensional layouts, with both rows and columns, which can be easily customized for different devices. Additionally, Grid Layout allows for easy placement of elements within the layout, as elements can be placed in any cell or group of cells within the grid. This means that the developer can create complex layouts without relying on fixed position elements.

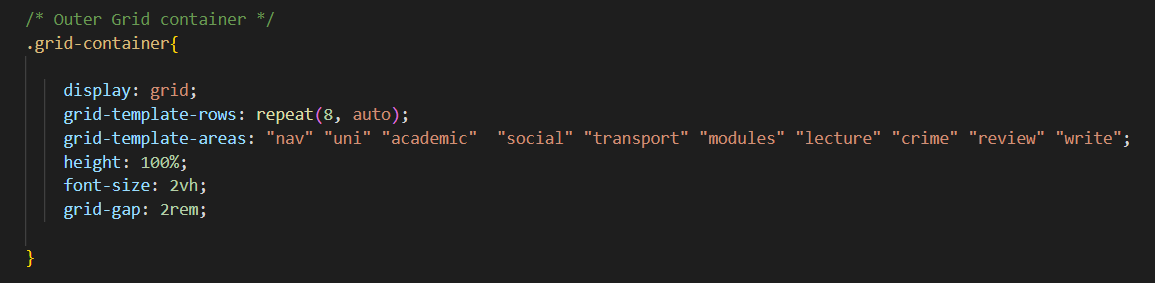
**Disadvantages of Grid Layout**

One of the main disadvantages of Grid Layout is its lack of browser support, especially in older browsers. This means that some users may not be able to view the website correctly if they are using an older browser. Additionally, Grid Layout can be complex and difficult to understand, especially for beginners. Developers must have a good understanding of CSS to create and manage Grid Layouts effectively.

In the CSS code, we used Grid Layout extensively for creating the front-end layout of our website. We used the "grid-container" class to define the outer container and set its display property to "grid". We then used the "grid-template-rows" property to set the size of each row and the "grid-template-areas" property to define the layout of each section of the website.

Additionally, we used the "justify-self" and "align-self" properties to position elements within the grid. For example, we used "justify-self: centre" to centre the text in the "university" section and "align-self: flex-end" to align the "titles" element to the bottom of the section.

We also used Grid Layout for the navbar, setting its display property to "grid" and using the "grid-template-columns" property to define the number of columns and their size. We have then used the "grid-gap" property to set the gap between the columns.



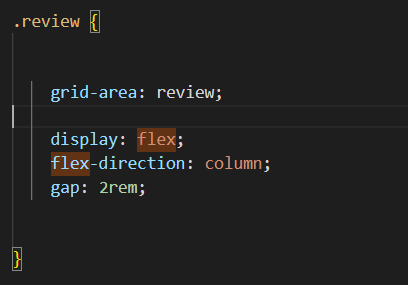
**Advantages of Flex Layout:**

Flex Layout is a simpler and more flexible layout option than Grid Layout. One of the main advantages of Flex Layout is that it is supported by almost all modern browsers. Additionally, Flex Layout is easy to learn and understand, making it a popular choice for beginners. Flex Layout also allows for responsive design, which means that elements can easily be rearranged for different screen sizes and devices.

**Disadvantages of Flex Layout:**

One of the main disadvantages of Flex Layout is that it is limited to one-dimensional layouts, which means that it is not as powerful as Grid Layout. Additionally, Flex Layout can be difficult to use for complex layouts, especially if the developer wants to create a multi-dimensional layout. Flex Layout also requires careful management of flex properties, which can be time-consuming.

We also used Flex Layout in certain areas where a simpler layout was required, such as in the review section. By combining both layouts, we were able to create a web layout that is both powerful and flexible, while still being easy to manage.



**Conclusion:**

Overall, both Grid Layout and Flex Layout have their advantages and disadvantages. Grid Layout is more powerful and can create complex layouts, but it is less supported and can be difficult to use. Flex Layout, on the other hand, is simpler and more flexible, but it is limited to one-dimensional layouts. When deciding which layout to use, it is important to consider the needs of the website and the skills of the developer. In our project, we have used Grid Layout to create a complex and responsive web layout, which is supported by modern browsers.

**6.2.2 Back-End**

We created the EJS and JavaScript code for the back end of our web application. This code is responsible for handling the database connection and rendering the dynamic content on the web pages. In this section, we will discuss the advantages and disadvantages of the EJS and MySQL code that we have created.

Advantages of EJS:

EJS is a popular template engine for Node.js that allows developers to generate HTML markup with plain JavaScript. One of the main advantages of using EJS is its simplicity and ease of use. EJS templates can be easily integrated into Node.js projects and provide a powerful way to generate dynamic content. Additionally, EJS provides a flexible syntax that allows developers to create complex templates with ease. The syntax is like traditional HTML, making it easy for developers to learn and use.

Disadvantages of EJS:

One of the main disadvantages of EJS is that it is not as performant as some other template engines. This is because EJS relies heavily on JavaScript to generate the HTML markup, which can slow down the rendering process. Additionally, EJS does not provide as many features as some other template engines, which can limit its usefulness in certain situations.

Advantages of MySQL:

MySQL is one of the most popular open-source relational database management systems (RDBMS) in the world. One of the main advantages of MySQL is its scalability and flexibility. MySQL can handle copious amounts of data and can be easily scaled to meet the needs of any project. Additionally, MySQL is widely supported and has a large community of developers, which means that there are plenty of resources available for developers who need help.

Disadvantages of MySQL:

One of the main disadvantages of MySQL is its complexity. MySQL can be difficult to set up and configure, especially for beginners. Additionally, MySQL requires a good understanding of SQL, which can be a steep learning curve for some developers. Finally, MySQL can be less performant than some other databases, especially when handling complex queries or large datasets.

Modules used with EJS:

In our web application, we have used several modules with EJS to handle different aspects of the application. One of the main modules that we have used is Express.js. Express.js is a popular web framework for Node.js that provides a powerful way to handle HTTP requests and responses. Additionally, we have used the Body-Parser module to parse incoming request bodies, and the Session module to manage user sessions.

Conclusion:

Overall, EJS and MySQL provide a powerful and flexible way to create dynamic web applications. EJS allows developers to create complex templates with ease, while MySQL provides a scalable and flexible way to manage data. When using EJS and MySQL, it is important to consider the advantages and disadvantages of each technology, and to choose the technology that best suits the needs of the project. In our web application, we have used EJS with Express.js, Body-Parser, and Session modules to create a powerful and dynamic back end that can handle a wide range of user requests and data.

**6.2.3 APIs**

We have used an external CSV file instead of an API. This decision was made due to the latency issues that can arise when using an API.

APIs are powerful tools that allow developers to retrieve data from various sources in real-time. However, when an API is called repeatedly or by many users at once, it can cause a delay in response time. This delay is known as latency and can be a major issue in a web application, especially one that requires real-time data.

To avoid latency issues, I decided to use an external CSV file. CSV files are lightweight and can be easily stored on a server or on the client side. By using a CSV file, we were able to avoid the delay caused by API calls and provide users with immediate access to the data they need. Additionally, we were able to update the CSV file easily and quickly, ensuring that the data displayed on the website is always up to date.

One of the main advantages of using a CSV file instead of an API is the flexibility it provides. With an API, developers are often limited to the data provided by the API provider. However, by using a CSV file, we were able to tailor the data to meet the specific needs of the web application. This allowed us to provide a more customized and personalized experience for users.

Another advantage of using a CSV file is that it is easily accessible and readable by humans. CSV files can be opened in a plain text editor or a spreadsheet program, making it easy for developers to understand and manipulate the data. This ease of use also makes it easier to debug any issues that may arise with the data.

Despite its advantages, there are some disadvantages to using a CSV file. One of the main disadvantages is that it is not as secure as using an API. With an API, data can be encrypted and protected, making it more difficult for unauthorized users to access it. Additionally, CSV files can become unwieldy and difficult to manage as the amount of data grows. This can lead to issues with performance and data integrity.

In conclusion, the decision to use an external CSV file instead of an API was made to avoid latency issues and provide a more customized experience for users. While there are advantages and disadvantages to using a CSV file, it was the best option for the specific needs of my web application. By carefully considering the needs of the application and the available options, we were able to create a responsive and efficient web application that meets the needs of its users.

**6.2.4 Data Structures**

In this section, we will outline the data structures used in the web application and how they impact the way our site operates.

Arrays:

Arrays are a fundamental data structure in JavaScript and are used to store a collection of elements. In the web application, arrays are used to store reviews and search results. The advantage of using arrays is their flexibility and ease of use. Elements can be easily added or removed from an array, and they can be easily looped through to perform operations on each element. However, the disadvantage of using arrays is that they are not ideal for storing copious amounts of data, as they can become slow and inefficient when the size of the array grows.

Objects:

Objects are used in JavaScript to store data in a key-value format. In the web application, objects are used to store search results and review information. The advantage of using objects is their flexibility and ability to store complex data structures. Additionally, objects can be easily modified and accessed using their keys. However, the disadvantage of using objects is that they can be difficult to manipulate when the data structure becomes more complex, and it can be challenging to access and manipulate nested data.

CSV files:

In the web application, an external CSV file is used to store university information. The advantage of using a CSV file is its ability to store copious amounts of data in a structured format. Additionally, CSV files are easy to generate, edit and parse with a wide range of tools. The disadvantage of using CSV files is the need for the application to read from and write to the file each time data is needed. This can lead to slower performance compared to using a database.

MySQL:

MySQL is a popular relational database management system used in the web application to store review information. The advantage of using MySQL is its ability to handle copious amounts of data efficiently and store it in a structured format. Additionally, MySQL provides powerful tools for querying and manipulating data, making it an excellent choice for complex applications. The disadvantage of using MySQL is its need for configuration and management, which can be challenging for beginners.

In summary, each data structure used in our web application has its advantages and disadvantages. Arrays and objects are flexible and easy to use but can become slow and inefficient when handling copious amounts of data. CSV files are efficient for storing copious amounts of data, but slower compared to using a database. MySQL provides powerful querying and manipulation tools but requires configuration and management. By understanding the strengths and weaknesses of each data structure, developers can make informed decisions about which data structure to use for a specific application.

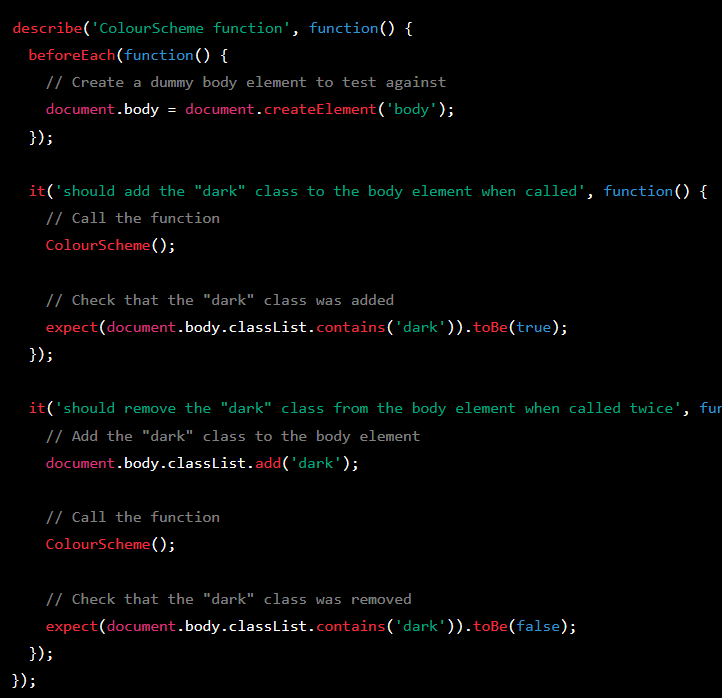
**6.3 Test-Driven Development**

This is a practice in which we, the developers write tests for our code before writing the code itself, with the aim of catching errors early in the development process. How we achieved this is outlined in section 6.5 below.

**6.4 Unit Tests**

We applied unit testing to check that each isolated part of our code behaves as we expect it to and meets their functional requirements.

**Unit test on Dark mode:**



The unit test that was performed on the code was designed to test the ColourScheme function. The test was created using the Jasmine testing framework, which is a popular choice for testing JavaScript applications.

The test was designed to ensure that the ColourScheme function added the dark class to the body element when it was called. It did this by first creating a dummy body element, and then calling the ColourScheme function on it. The test then checked whether the dark class had been added to the body element using the toHaveClass matcher provided by Jasmine.

The purpose of this unit test is to ensure that the ColourScheme function worked correctly, and that it added the dark class to the body element when it was called. By performing this test, we can be confident that the function is working as intended and that it will correctly change the layout of the website to dark mode when called. Additionally, this test can be used to ensure that any future changes to the function do not break its existing functionality.

Unit test on posting the review data into the table



The unit test created here tests the behaviour of the code in the app.post('/post') route handler function. The test checks whether the function correctly inserts a new review into the database, retrieves all reviews from the database, and stores them in the req.session.reviews object before redirecting to the /review page.

To do this, we have created a mock request object and a mock response object using the Jest testing framework. We then simulate a post request to the /post endpoint, passing in the required parameters (name, lecture, course, and review).

The test function first checks whether the response has been redirected to the /review page, which indicates that the review has been successfully inserted and the user has been redirected. It then checks whether the req.session.reviews object has been populated with the newly inserted review and all other reviews from the database.

To ensure that the test is consistent, we have created a test database and used it to create a test table with the same structure as the reviews table. Before running the test, the test database is connected to and initialized with test data using the beforeAll() method. After the test is complete, the test database is cleaned up and disconnected using the afterAll() method.

This unit test ensures that the app.post('/post') route handler function is working as intended and that new reviews are successfully inserted into the database and retrieved when the /review page is loaded.

**6.5 TDD Practice**

In the unit test for the app.post('/post', ...) function, the TDD methodology was followed by creating test cases for each step of the function's implementation.

Firstly, the test case checks whether the function receives the expected request and response objects as arguments. This ensures that the function can receive, and process incoming data as expected.

Next, the test case checks that the SQL query generated by the function is correct and that the values are properly inserted into the database. This helps to ensure that the data is being stored correctly and that the data can be easily retrieved later.

After that, the test case verifies that the SELECT query used to retrieve the review data from the database is working correctly. This helps to ensure that the data can be retrieved and displayed on the website as expected.

Finally, the test case checks that the function redirects the user to the correct page after the review has been submitted. This helps to ensure that the user experience is smooth and that they are not left confused or lost after submitting their review.

By following the TDD methodology, each step of the function's implementation is carefully tested to ensure that it works as expected. This helps to catch any bugs or errors early in the development process, which can save time and resources in the long run.

1. **DISCUSSION**

**7.1 Thematic based analysis**

|  |  |
| --- | --- |
| **Theme** | **Description** |
| 1. University Rankings | Homepage displays up-to-date UK university rankings; users can sort by Overall score, Student Satisfaction, Social, and Graduate prospects; default sorting method is based on the Overall score. |
| 2. Reviews and User-Generated Content | Reviews Page for reading reviews on subjects and social aspects; users can submit their own reviews by creating a profile and using the text box on the Reviews Page. |
| 3. User Profiles | Profile Page for creating a user account by providing name and surname; user profile is necessary for submitting reviews. |
| 4. Contact and Support | Contact Page with team members' contact information: email, telephone, and address; users can reach out to the team for assistance or inquiries. |
| 5. Student Forum | Dedicated forum for students to engage in discussions, ask questions, and share experiences; accessible to registered users. |

* 1. **Concept based analysis**

|  |  |
| --- | --- |
| **Concept** | **Related Features** |
| 1. University Information | Up-to-date UK university rankings, sorting by Overall score, Student Satisfaction, Social, and Graduate prospects |
| 2. User-Generated Content | Reading reviews on subjects and social aspects, submitting personal reviews and rankings |
| 3. User Interaction | Creating user profiles, engaging in student forum discussions, asking questions, and sharing experiences |
| 4. Communication & Support | Contact Page with team members' contact information, reaching out for assistance or inquiries |
| 5. Personalization | Customizable user profiles, personalized recommendations based on user preferences |
| 6. Search and Navigation | Search options, filters for browsing university rankings and reviews, easy-to-use navigation menu |
| 7. Accessibility & Usability | Responsive design for various devices, adherence to accessibility guidelines, intuitive user interface |

1. **EVALUATION**

**8.1 Technical Difficulties**

During the development of our website, we initially chosen PHP and MySQL as the main technologies for our back end and database management, respectively. However, we encountered several challenges that led us to switch from PHP to JavaScript (Node.js) for the back end while maintaining MySQL for database management. This section will outline the problems we faced with PHP, the API integration issues, and the rationale behind choosing JavaScript.

Challenges with PHP and MySQL:

1. Dependency on additional modules: PHP relies on various modules to extend its functionalities, such as MySQL or PDO for MySQL integration. We encountered difficulties installing and configuring these modules, which hindered our progress in setting up the back-end infrastructure for our website.
2. Incompatibility issues: While installing the required PHP modules, we faced compatibility issues with the server and the PHP version we were using. Troubleshooting these issues demanded significant time and effort, which could have been invested in the actual development of the website.
3. Multiple databases: During the development process, different team members created separate databases, resulting in two distinct databases that needed to be integrated with the website. PHP's approach to handling multiple databases complicated the process of consolidating the data and implementing a single, unified database for the website. Although PHP proved to be better suited as a resolution to this issue, it was difficult for those working on this task to learn the language, and so it was agreed that we would continue to use MySQL and CSV files to build the databases.
4. API integration challenges: We encountered issues with integrating the API for our ranking page. We discovered that not all websites freely provide their API to external users, as some block them to protect their data. Unfortunately, this was the case with <https://www.thecompleteuniversityguide.co.uk/>. One potential solution to this issue could have been to reach out to the website administrators and request access to their API or look for alternative sources of data that provide open APIs. However, upon integrating an alternative API, we found that the load time for the website was affected by up to 3 minutes. We agreed that it would not be worth it to keep the API if it were going to affect the user experience by having too long of a wait time. Switching to JavaScript (Node.js) and MySQL

After facing the challenges with PHP, we decided to switch to JavaScript (Node.js) as the back-end technology, while continuing to use MySQL for database management. The reasons for choosing JavaScript were as follows:

1. Easier setup and fewer dependencies: Node.js allows for a streamlined installation process with minimal dependencies, making it easier to set up the back-end infrastructure compared to PHP. Additionally, JavaScript has a vast ecosystem of libraries and frameworks, such as Express.js, which simplifies web application development.
2. Full-stack development: By using JavaScript for both front-end and back-end development, we could maintain a consistent programming language across the entire project, making it easier for the team members to collaborate and share code.
3. Improved handling of multiple databases: JavaScript and its libraries, such as Sequelize, provide a more flexible approach to handling multiple databases. This made it easier for us to consolidate the data from the two databases created by different team members and implement a single, unified database for the website.
4. Better performance: Node.js is known for its non-blocking, event-driven architecture, which enables it to handle multiple simultaneous connections efficiently. This results in better performance and scalability compared to PHP, which is important for our website's growing user base.

In conclusion, although we initially chose PHP and MySQL for our website's back end and database management, we faced numerous challenges during the development process, including difficulties with API integration. The decision to switch to JavaScript (Node.js) for the back end allowed us to overcome these obstacles and achieve a more efficient, consistent, and scalable solution for our website.

Challenges with Review page:

Due to time constraints, I was unable to implement the functionality to change videos for each university and change images based on what was searched in the review page using APIs. While using APIs would have provided more flexibility and real-time data, I had to work with the available resources to deliver a functional web application within the project timeline. However, this limitation did not affect the core functionality of the web application, and users can still access and post reviews for different courses and lecturers. In future iterations, I plan to explore and incorporate APIs to improve the user experience and make the web application more dynamic.

* 1. **Teamwork and Management**

At the beginning of the project, we had 7 group members. However, two of the members failed to contribute to the work required to achieve our objectives and we had to make the difficult decision to separate from them. Unfortunately, this meant that we were not able to reach the higher expectations that we had set for ourselves due to having less people working on the project.

We had realised early on that this may become an issue as they were absent from the initial team meetings, where we discussed roles and responsibilities and had assigned at least two people to a task. To resolve this, we assigned them jobs that would not require too much technical work, for example, providing early user feedback on the website, so that the developers in the group could make the necessary improvements where needed to make the website functional.

Despite our efforts to communicate regarding the tasks set for them, we had very little response. This put a mental strain on the remaining members as we had to pick up the work not completed by them and was not something we had properly prepared for. This resulted in a few weeks of no work being completed, while we tried to process the impact this would have on everyone. We eventually arranged another group meeting to discuss how we would move forward and agreed that it would be best for those who were actively contributing to continue as a separate group of five.

As there were only a few weeks remaining on the project, we reassigned tasks based on what each person felt they would be able to achieve comfortably and tried to evenly distribute the workload so that no member was too overwhelmed.

* 1. **Aims Reached**

As outlined in the project scope, we were able to meet our MVPs. The homepage has been developed, allowing users to view the ranking of UK universities based on several subheadings. The reviews page has been implemented, allowing users to read reviews left by other users and submit their own reviews after creating a profile.

While we have a functioning website that meets its current user requirements, it is important to note that ongoing testing and optimisation will be necessary to ensure that the site continues to meet the needs of the user and remains relevant in the market.

* 1. **Going Forward**

There are several steps that we can take to further improve the website. One key area to focus on would be increasing user engagement and interactivity. This could be achieved through features such as live chat, user polls and interactive maps. By providing users with more opportunities to engage with the site, they can build a sense of community and increase the value of the platform.

Another area to focus on is expanding the content available on the site. While the homepage, review page, forum page and profile page have been completed, there is always room for additional content to help make the site more informative and useful to users. This could include adding more detailed information on individual universities, such as their academic programs, and research initiatives. Additionally, we could collaborate with universities to provide exclusive content, such as virtual tours, interviews with faculty members, and behind the scenes look at campus life.

Finally, it will be important for the group to continue to optimize the site for user experience and search engine visibility. This would involve further unit testing to identify areas where the site could be improved. By taking a data-driven approach to site optimization, we can ensure that the site is both user-friendly and highly visible to search engines, helping to attract users.

1. **CONCLUSION**

In conclusion, the project has been completed with the minimum viable products (MVP’s) being met, although not without encountering some technical issues and difficulties with group dynamics. Despite the technical issues with the CSS and APIs, we have managed to overcome them and deliver a functional website with the MVPs in place. The site includes a homepage displaying up-to-date rankings of UK universities based on various subheadings, a reviews page for users to read and submit reviews, and a profile page for users to create their profiles. These technical issues may have affected the user experience and could have been avoided with more careful planning, time, and testing.

By providing a platform for users to compare and rank universities, the site has the potential to influence how universities compete and market themselves. Policymakers may also eventually be interested in the sites data as it can provide an insight into the higher education landscape in the UK.

In a dynamic workflow, the completed project represents an important milestone, but it is important to note that the website will require ongoing maintenance to ensure it remains relevant and user-friendly.

We also faced challenges with group members not contributing, which impacted the project timeline and quality of work. It is important for group members to be accountable and reliable to ensure that the project is completed to the best of its potential. Nevertheless, the group should be commended for their hard work and implementation of the project. With more time and work on the project, it has the potential to be a valuable resource for current and potential students.

**9.1 Individual reflection**

*\*Write your personal reflection here\*0.*

As a group, we have created a web application that aims to provide users with a comprehensive and interactive platform to access university rankings, read and submit reviews, and get in touch with our team.

Our landing page displays an up-to-date list of university rankings in the UK, which we have achieved through the use of APIs. Users can sort the rankings via four subheadings: Overall Score, Student Satisfaction, Social, and Graduate Prospects. By default, the rankings are sorted by Overall Score, providing users with a quick and easy way to access the information they need.

The Reviews Page is split into two parts: Read Reviews and Submit Reviews. The Read Reviews section allows users to read reviews left by other users and see how they have ranked subjects or social aspects. The Submit Reviews section provides users with the option to submit their own rankings, which requires the creation of a profile.

The Profile Page is where users can create their profile, providing a name and surname, which is necessary to leave a review. This feature helps us ensure that reviews are left by real users, thus maintaining the integrity of the platform.

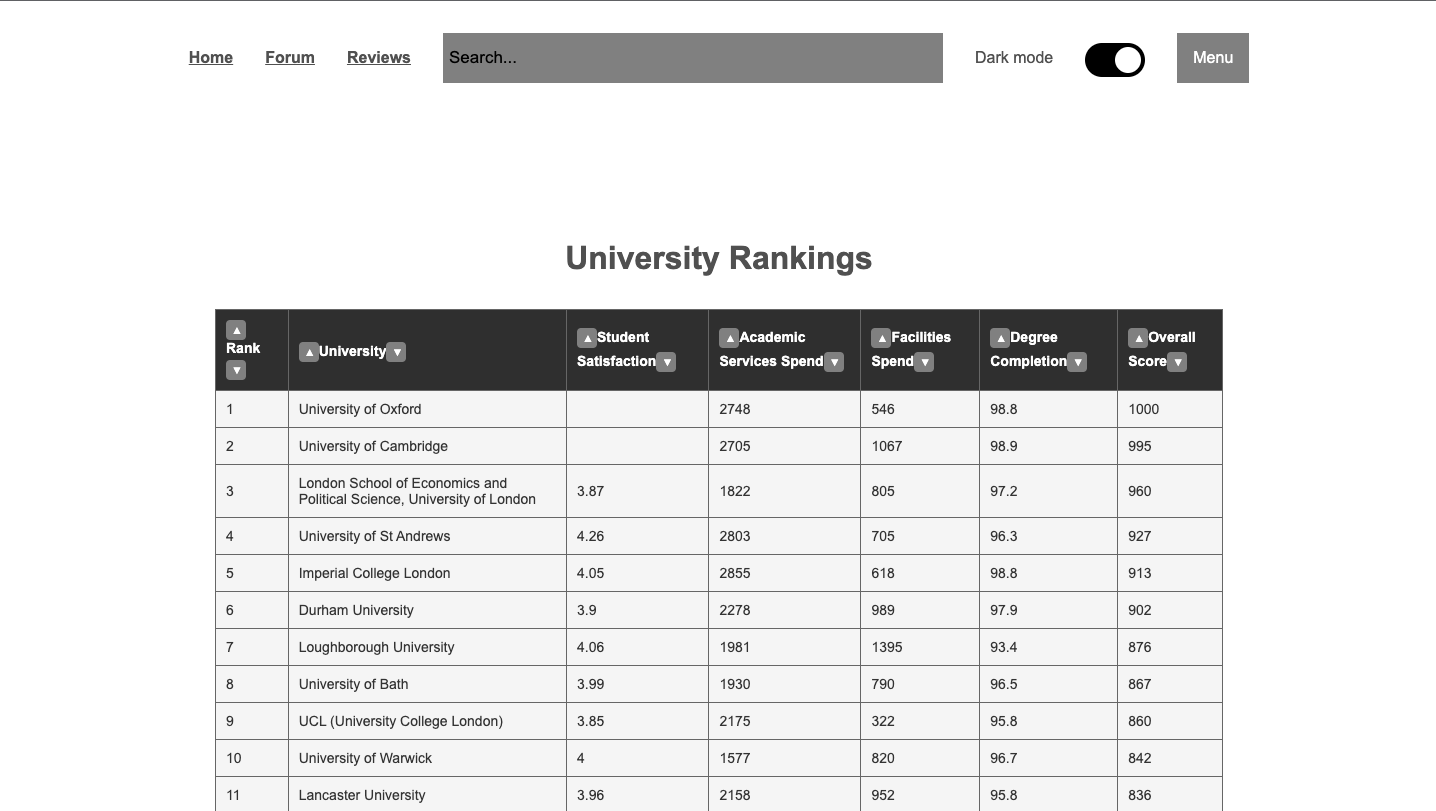
Lastly, we have a Contact Page, where users can get in touch with any of our team members via the details given on the page. We have provided email, telephone, and an address as a minimum, so users can reach us through their preferred method of communication.

In future, we aim to enhance the homepage's functionality by adding more sorting options and filters, enabling users to customize their search further. We also plan to improve the review submission process by adding more fields to give users the option to rate different aspects of their university experience. Additionally, we will work on enhancing the user interface to make the platform more user-friendly and accessible to a broader audience.

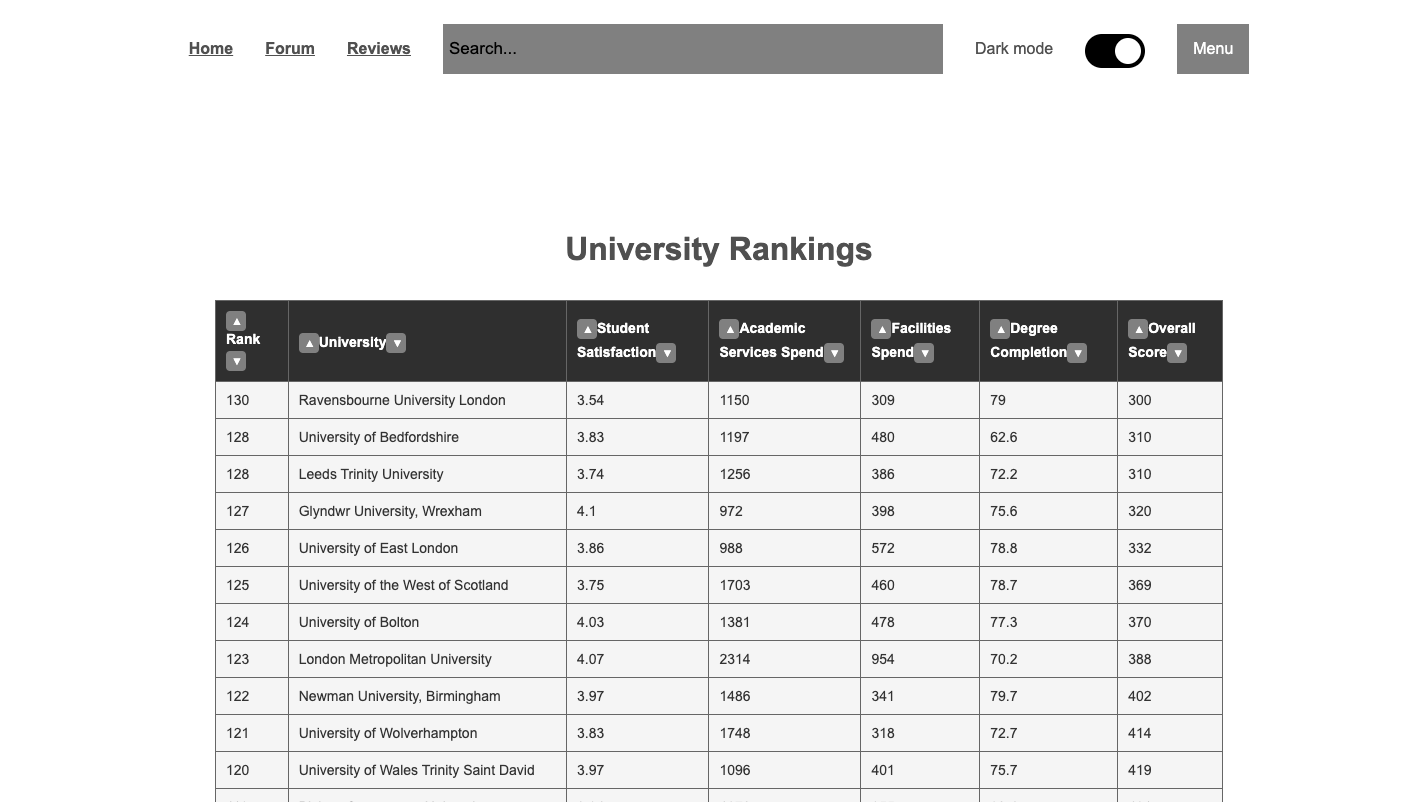
1. **USER GUIDE**

**- Homepage**

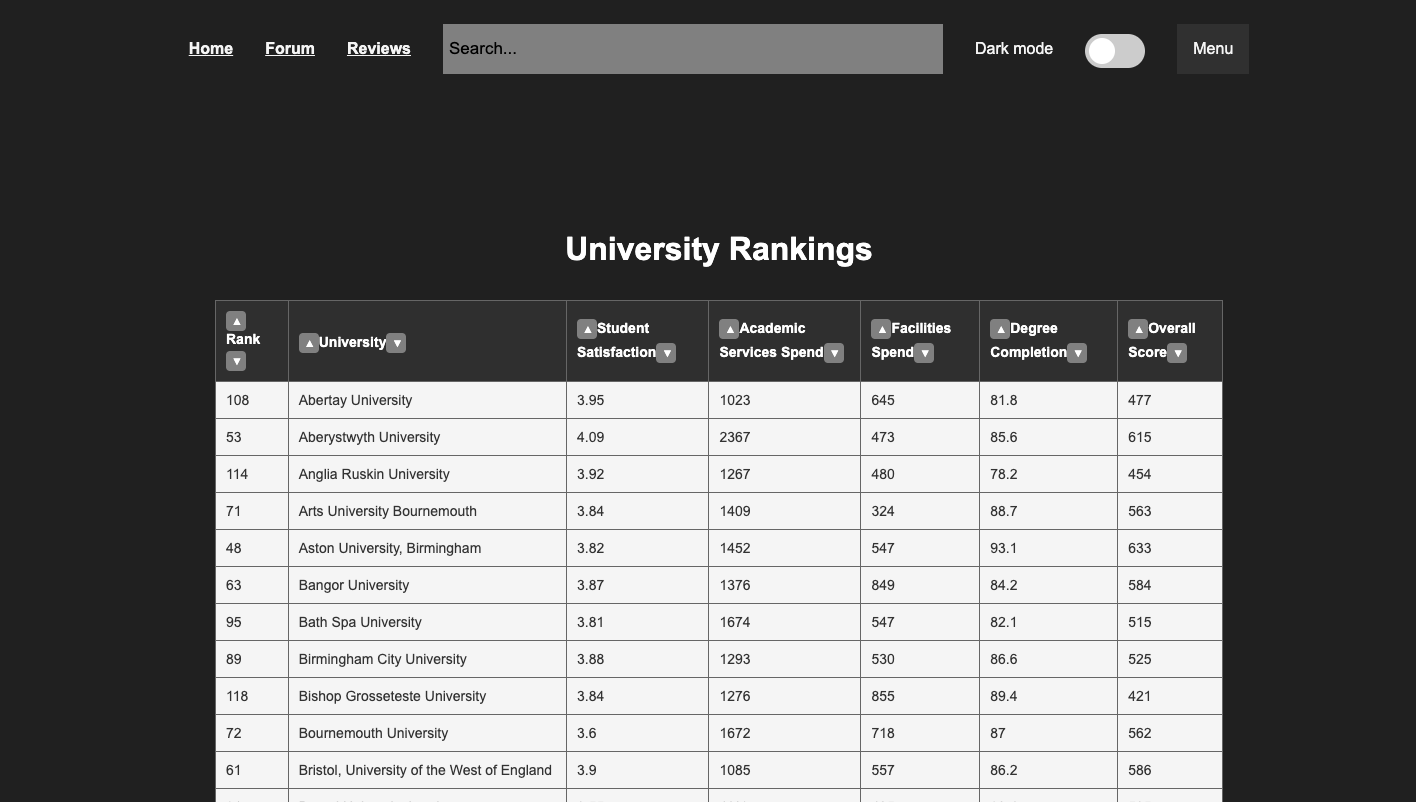
**This is what the homepage looks like, you can click on ‘Home’, ‘Forum’ or ‘Reviews’. It will take you to each of those pages respectively.**



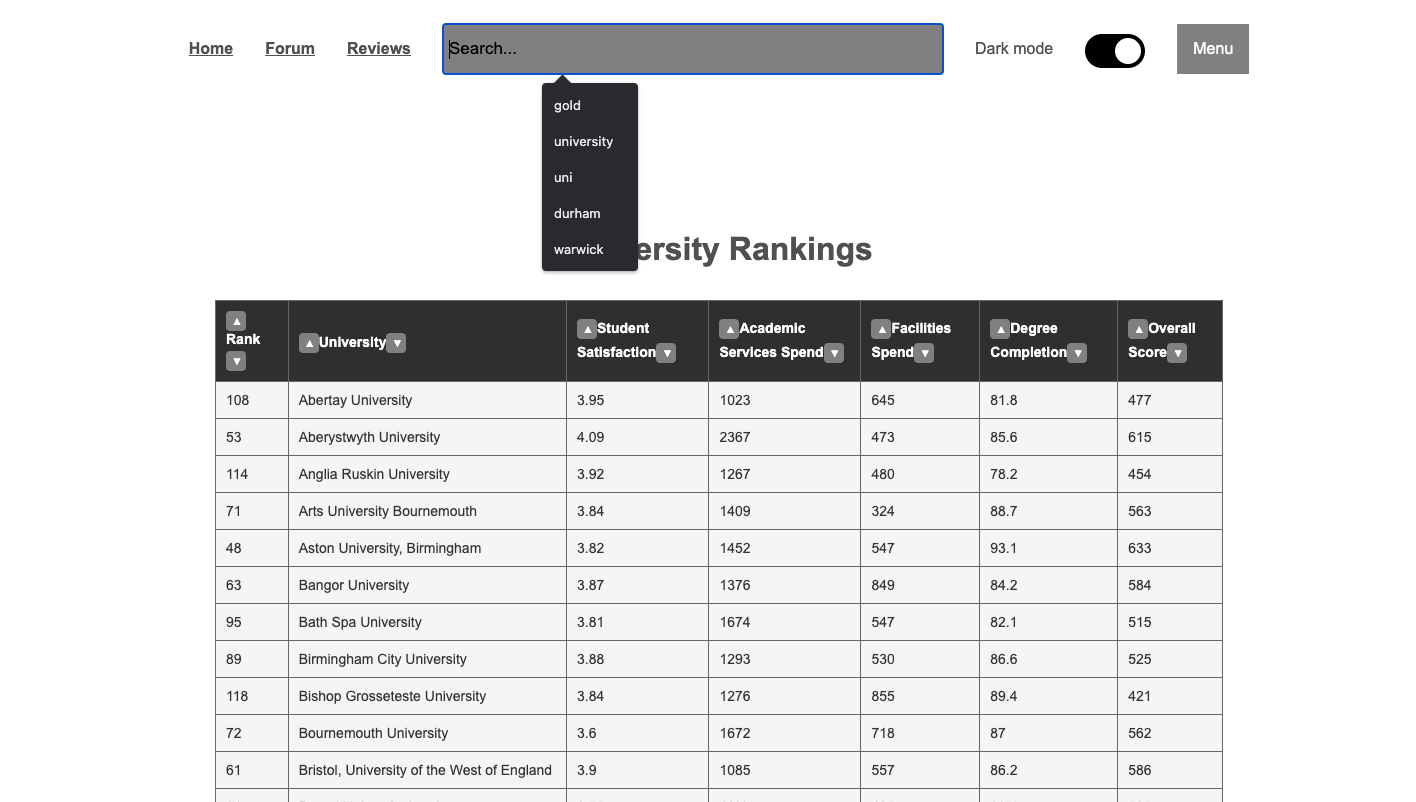
**Click on the up or down arrow to sort each column in the table into ascending order or descending order.**



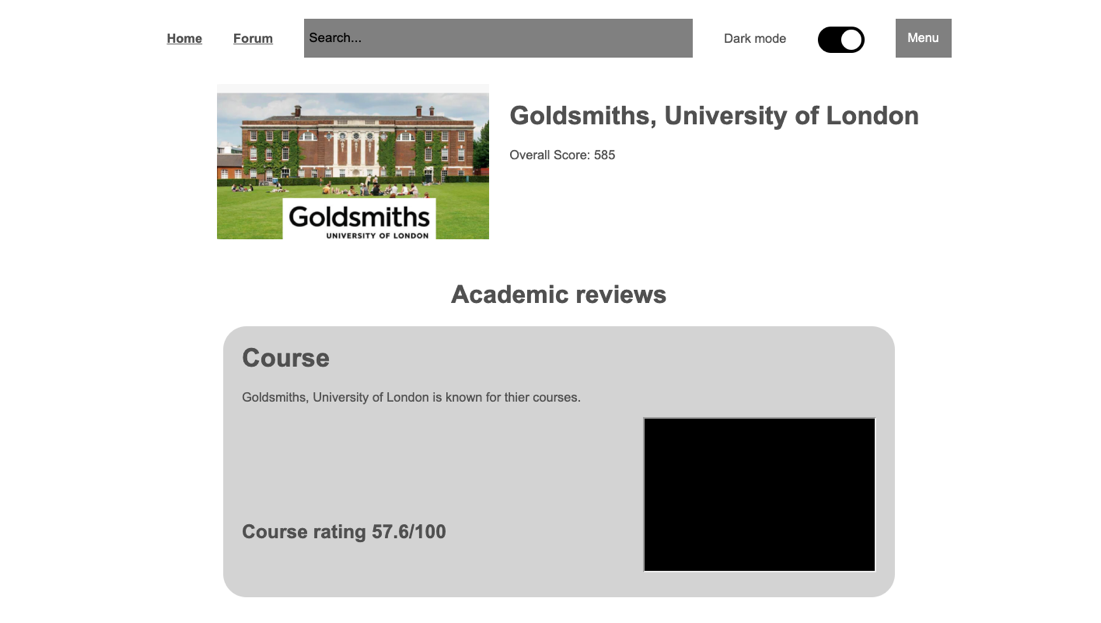
Click on the dark mode button to toggle dark mode on and off. This is what the homepage looks like with dark mode on



**Click on the search bar and type in a name of a university**

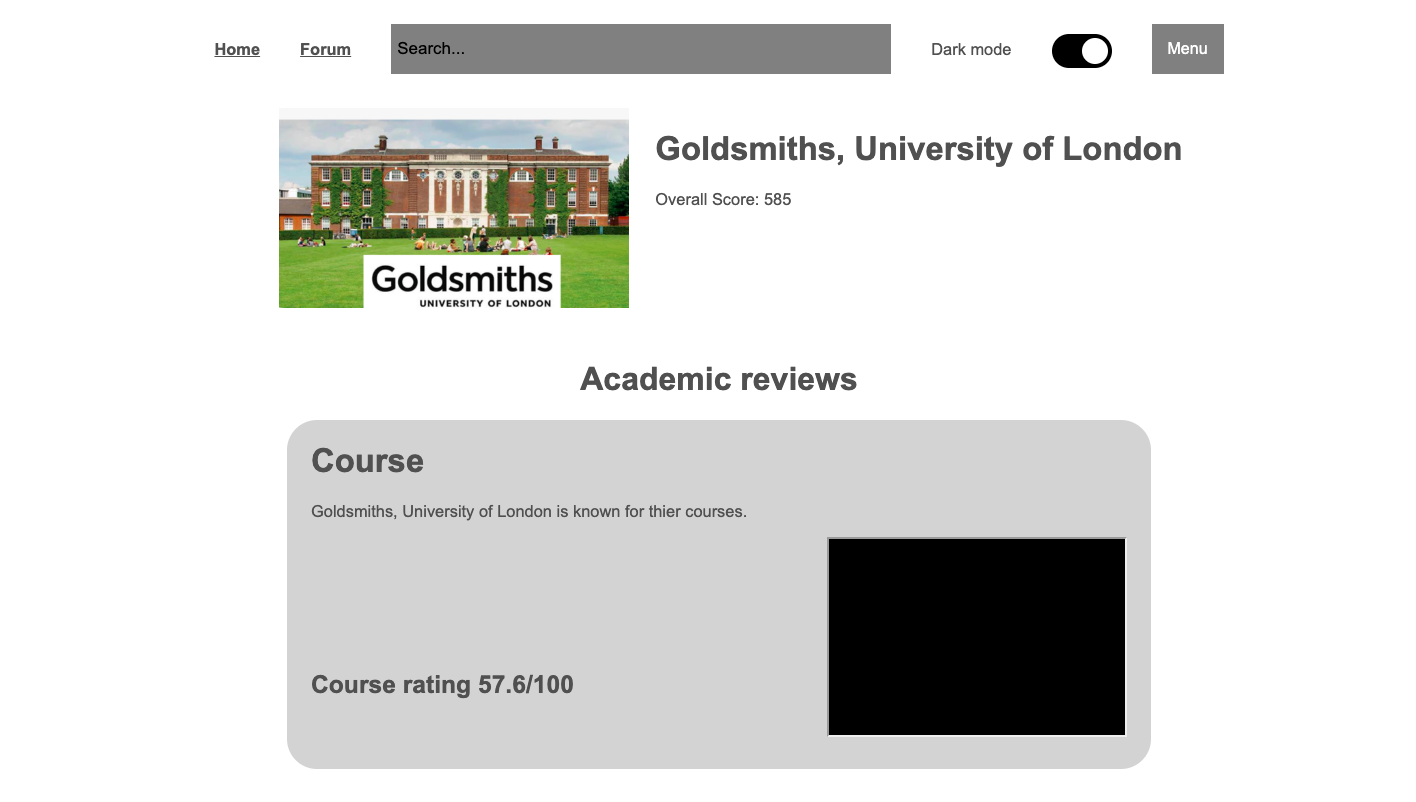


**Here it now shows the reviews for that university, in this example goldsmiths**

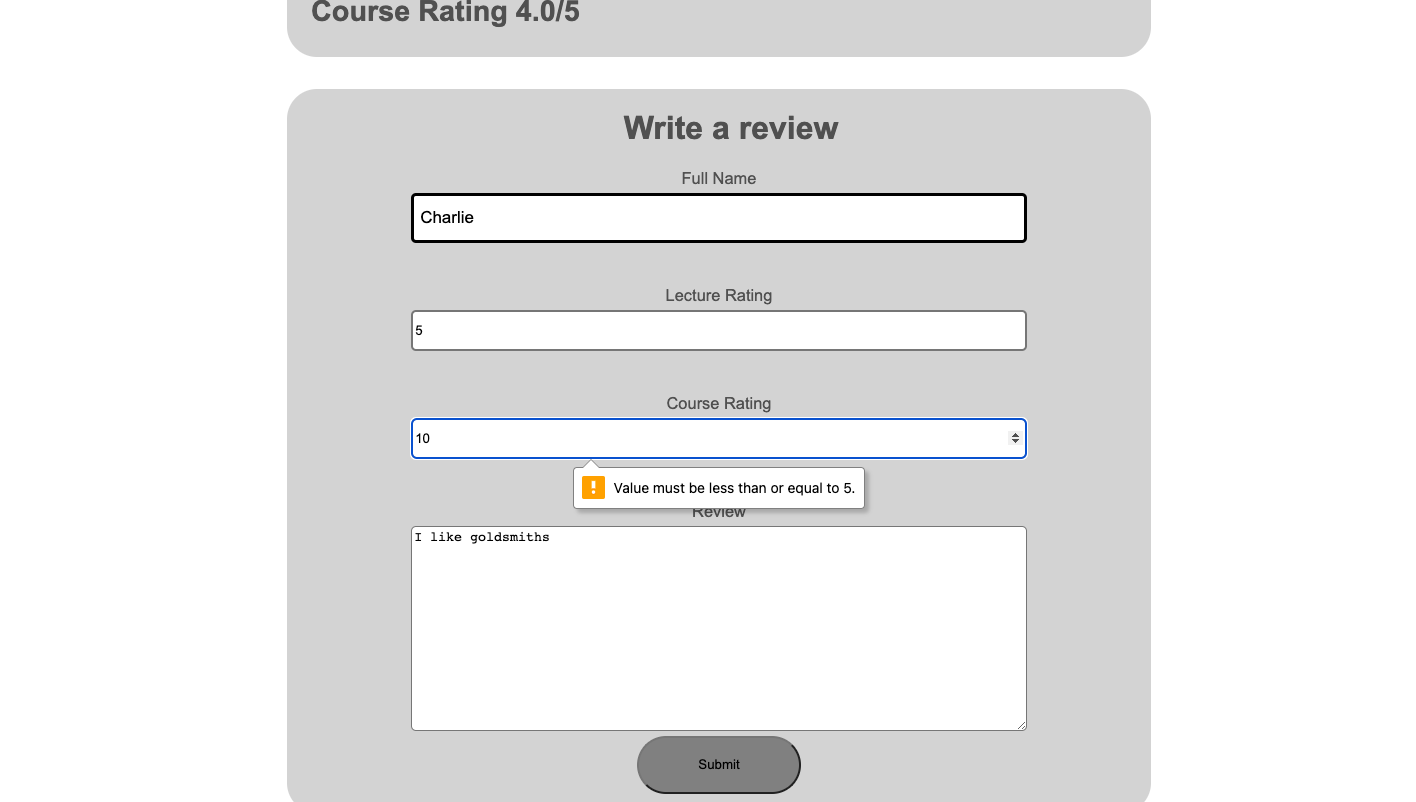


**Reviews Page**

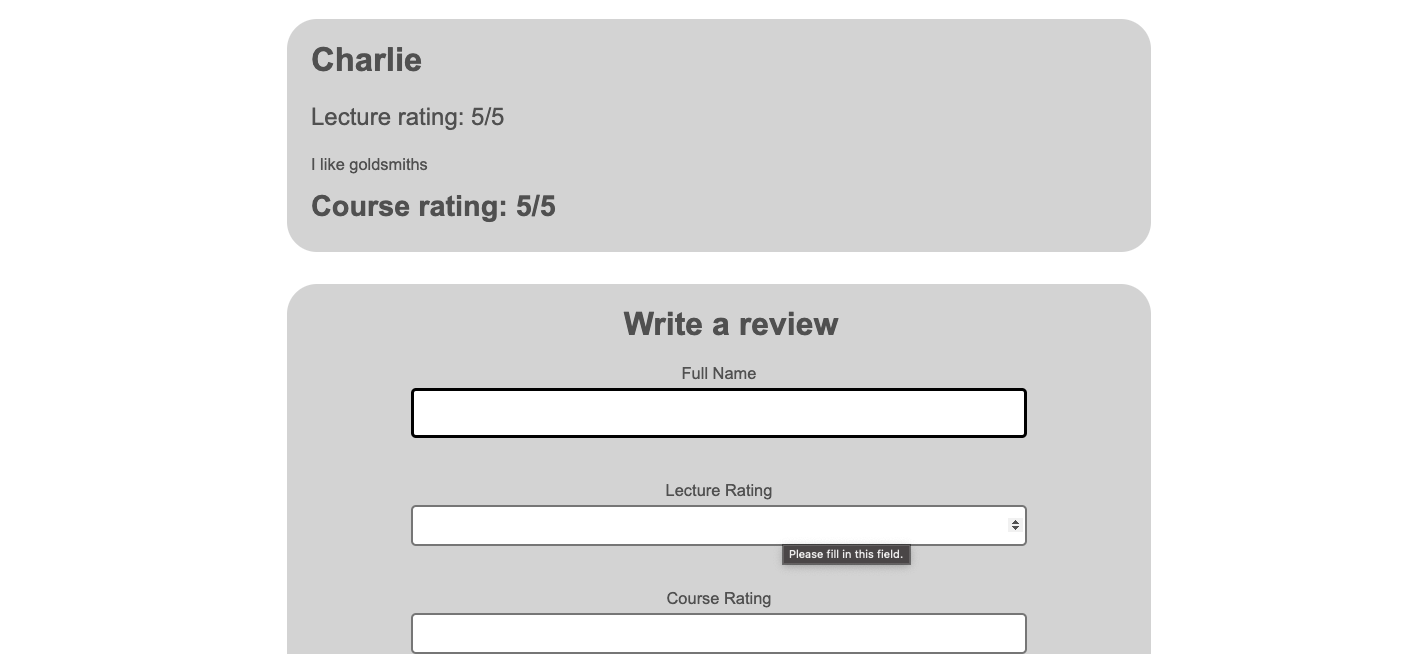
**This is the reviews page, you can scroll through reviews of a university you have searched for**



**Here you can enter you name, type in the lecture rating and course rating and type some comments in the review box. The course rating and lecture rating must be 5 or lower. Once you have finished click submit**

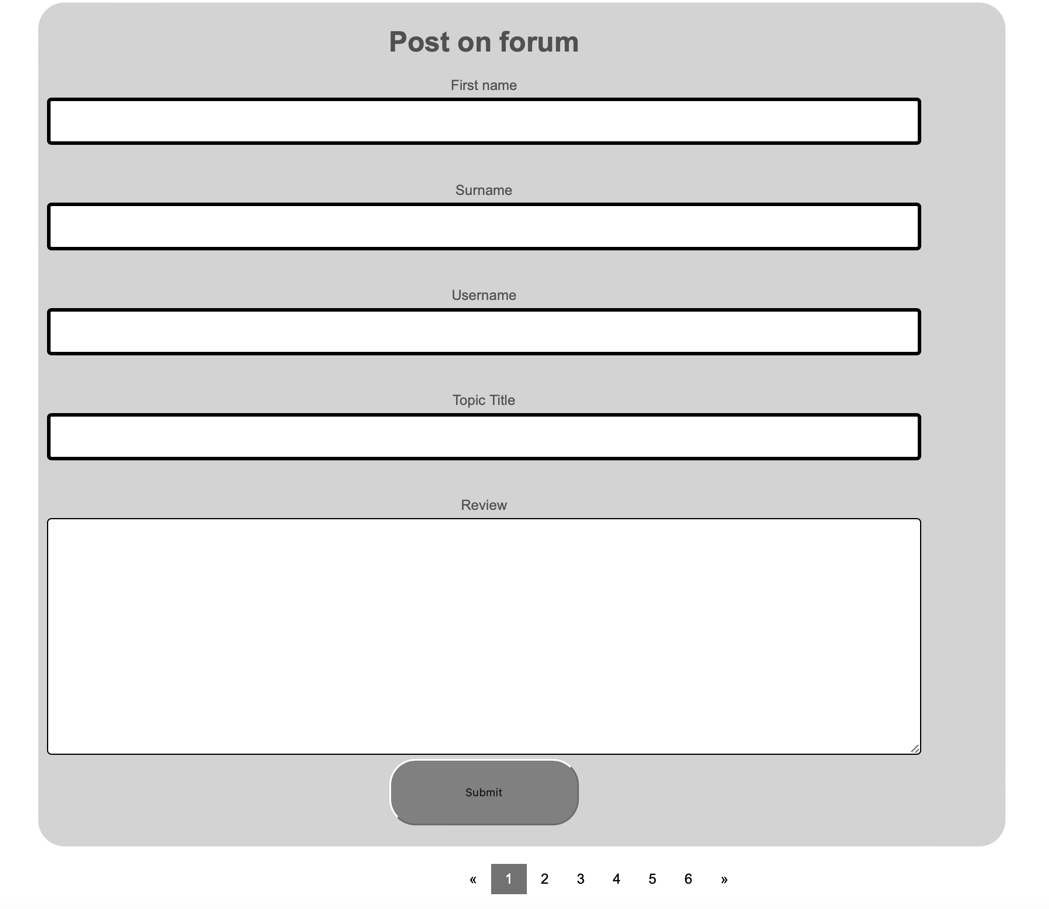
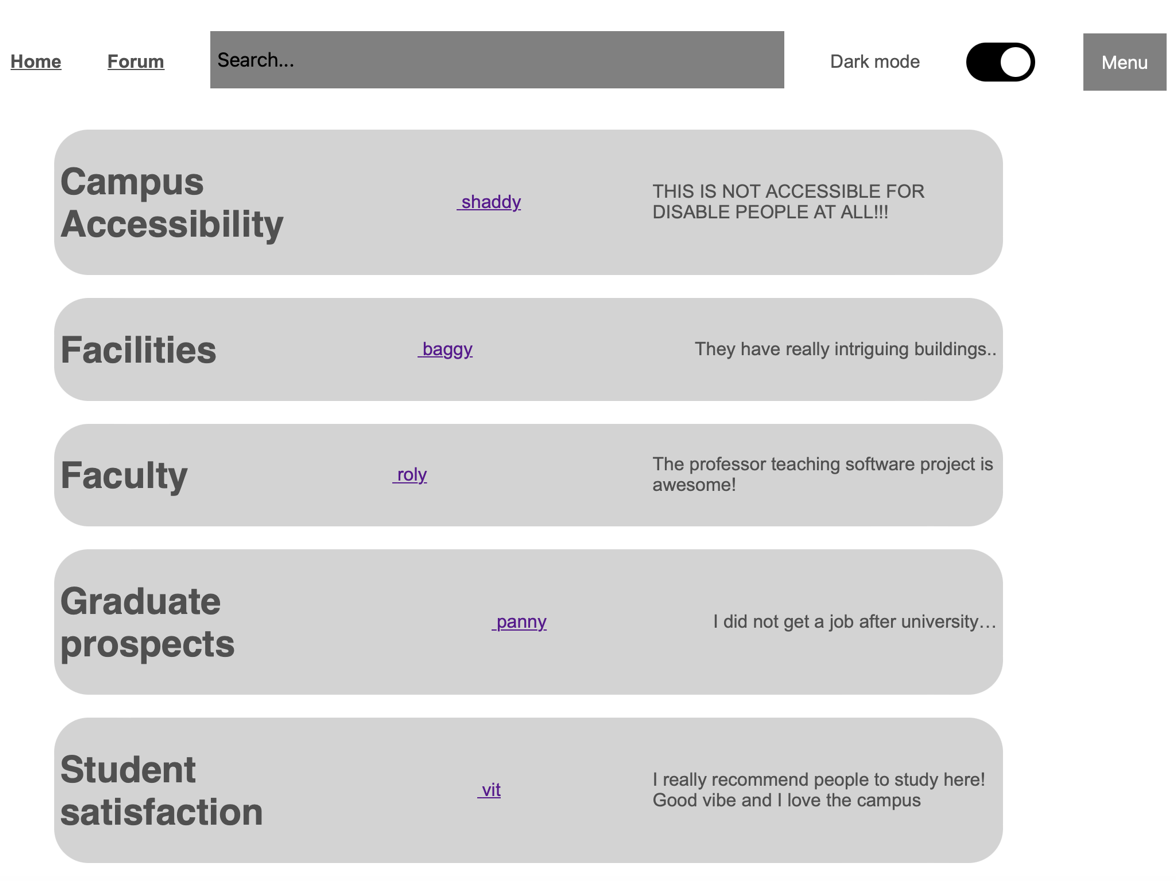


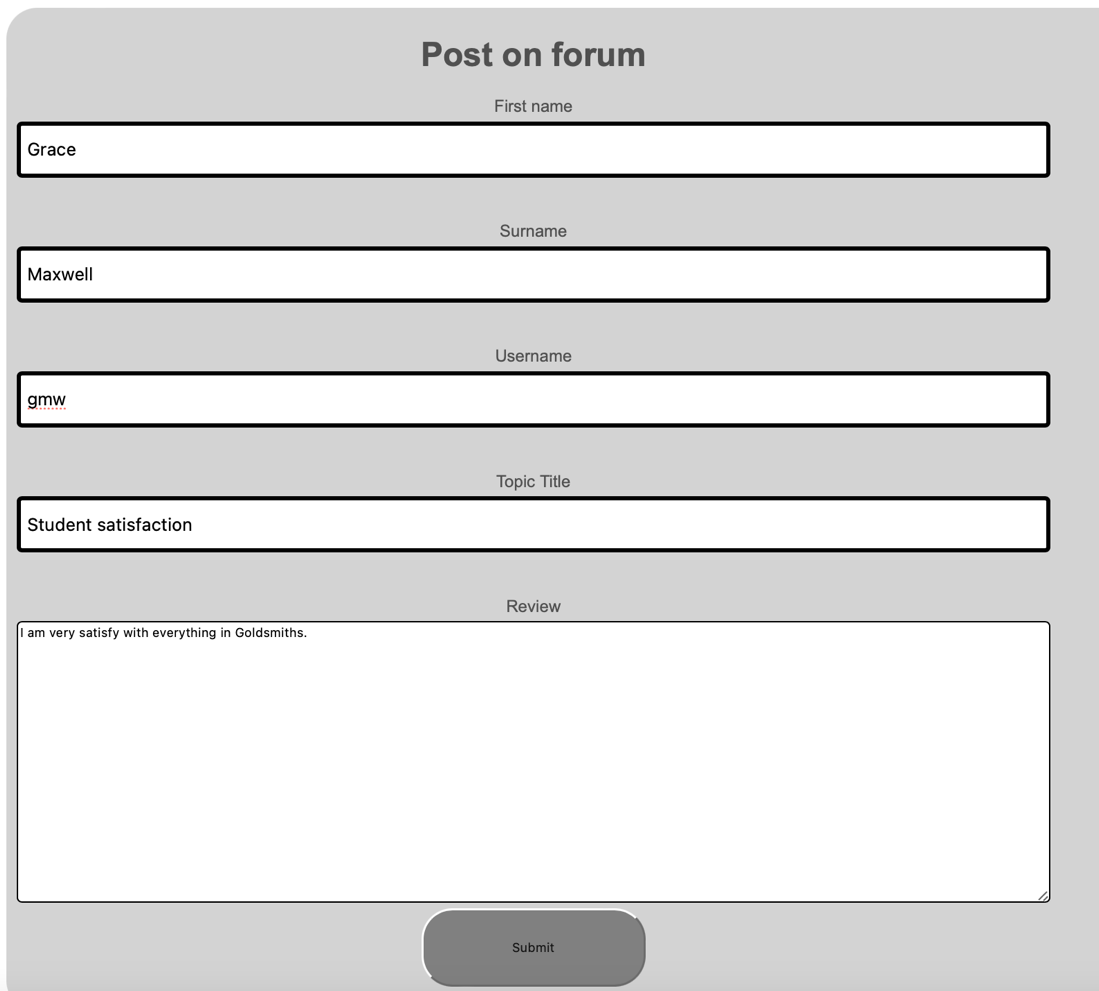
**Here we can see the review has been posted**



**Forum Page**

**This is the forum page; you can scroll through thread in the forum about different aspect of a university.**



**Users can post on the forum by entering their first name, surname, username, topic title and review like following image:**

**By clicking the submit button, the thread would be posted on the forum**



**Sign in/Sign Up**

**Click on Menu then Sign in to load the sign in page**

**Enter your username and password and click sign in**

Graphical user interface, application

Description automatically generated

**Click on sign up and enter your information below.**

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**After signing up this is what will be displayed on screen**

Graphical user interface, text, application

Description automatically generated

1. **References**

Hazelkorn, Ellen (2019) University Rankings: there is room for error and "malpractice". [Online]

Vu, K.P.L., Proctor, R.W. and Hung, Y.H., 2021. Website design and evaluation. Handbook of human factors and ergonomics, pp.1016-1036.

Guide.freecodecamp.org. (2019). Advantages and Disadvantages of JavaScript. [online] Available at: <https://guide.freecodecamp.org/javascript/advantages-and-disadvantages-of-javascript/> [Accessed 3 Apr. 2023].

Redbytes: Custom Mobile Application Development Company [iOS, Android, Windows]. (2019). 15 Best Programming Languages For Mobile Apps 2019 | Redbytes. [online] Available at: <https://www.redbytes.in/best-programming-language-for-mobile-apps/> [Accessed 28 Mar. 2023].

Spencer, J. (2019). Top 9 Advantages of JavaScript. [online] MarkupBox. Available at: <https://www.markupbox.com/blog/advantages-of-javascript/> [Accessed 22 Mar. 2023]

Kathy lund Dean (2014), It’s not just Rate My Professor anymore! Ethical issues with student evaluations of teaching [online], Available at <https://connect.aom.org/blogs/kathy-lund-dean/2014/01/21/its-not-just-rate-my-professor-anymore-ethical-issues-with-student-evaluations-of-teaching-set> [Accessed 20 Mar. 2023]

Webaccess.berkeley.edu. (2019). Top 10 Tips for Making Your Website Accessible | Web Access. [online] Available at: <https://webaccess.berkeley.edu/resources/tips/web-accessibility> [Accessed 31 Mar. 2023].

Bcs.org. (2019). [online] Available at: <https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/> [Accessed 26 Mar. 2023]

Lee, J. and Shin, D. (2018). Understanding and preventing academic plagiarism: A critical review. Journal of Hospitality, Leisure, Sport & Tourism Education, 23, pp.47-54.

Bickford, D. J., & Wright, S. J. (2019). Examining the Impact of Instructional Design and Course Content Delivery on Student Success in Online and Blended Environments. Journal of Educational Technology Development and Exchange, 12(1), 1-18.

Schneier, B. (2019). The Internet of Things Will Turn Large-Scale Hacks into Real World Disasters. [online] Available at: <https://www.theatlantic.com/technology/archive/2014/01/the-internet-of-things-will-turn-large-scale-hacks-into-real-world-disasters/282860/> [Accessed 29 Mar. 2023]

Michalopoulos, M., & Triantafillidou, A. (2019). A survey of cyber security risks in cloud computing. Journal of Network and Computer Applications, 127, 1-25. [Accessed 27 Mar. 2023]

Tinto, V. (2017). From Theory to Action: Exploring the Institutional Conditions for Improving Student Retention. In Student Success in College (pp. 63-84). Routledge. [Accessed 25 Mar. 2023

Van Dusen, D. S. (2022). Measuring Student Engagement and Retention in Online Learning. Online Learning Journal, 26(1), 106-127. [Accessed 1 Apr. 2023]

Hass, R. W., & Daugherty, M. K. (2022). Cybersecurity in higher education: A multi-method examination of students' security behaviors. Computers & Security, 119, 102277. [Accessed 30 Mar. 2023]

O'Neill, D. K., & Saipe, D. (2022). A framework for student-centered online learning: A case study in graduate educational leadership. Journal of Education and Learning, 11(1), 1-16. [Accessed 29 Mar. 2023]