SIX WEEKS SUMMER TRAINING REPORT

ON

Frontend Web Development using ReactJS

Submitted by, Ankur Paul (11901599)

Registration Number: 11901599

Program Name: P132::B.Tech. (Computer Science & Engineering)

Under the guidance of

1. Aniket Methi
2. Surya Saxena
3. Kanishka
4. Mukesh Dubey
5. Rest of the upGrad team

School of Computer Science and Engineering,
Lovely Professional University, Phagwara
(June-July, 2021)

Declaration

I, Ankur Paul hereby declare that I have completed my six weeks summer training at upGrad Ltd. from 31st May 2021 to 12th July 2021 under the guidance of Aniket Methi. I have declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfil the requirements of training for the award of degree of Bachelor of Technology, Computer Science and Engineering, Lovely Professional University, Phagwara.

Ankur Paul

(Place left intentionally for signature)

Name of Student: Ankur Paul

Registration Number: 11901599

Date: 16th July, 2021.

Introduction

My name is Ankur Paul, in the due time of the summer training as instructed from my college, I have completed a Frontend Web Development using ReactJS by upGrad. The listing below shows the modules that were completed during the entirety of the course.

I have practiced industry specific tech that is used daily by web developers to push their code to the customers daily. They include using a Version Control System (Git), and PaaSs (Platform as services, like Firebase for backend hosting and Netlify for hosting the site).

Course Index

1. Pre-Launch Preparatory Content
2. Introduction of Web Development
3. Advanced JavaScript & ReactJS Framework.

Course 1 of 3

In this session

I learnt —

That version control is a very powerful tool. Developers all across the world are using version control and Git specifically for all the good reasons.

Version control has the power to solve almost all problems that I might face while working on a project.

I will learn about the issues that I would face if I were not using version control for software development.

One of the major scenarios where version control is used is

Suppose I are working on a code, and after making many changes, I realize that I have really messed it up and now I want to revert to the last good version of my project. How would I do that without version control?

I will also dive deep into learning what version control is and how it comes to our rescue.

I will learn about two types of version control systems (VCS)

- Centralized
- Distributed

I will be able to differentiate between centralized and distributed version control systems and conclude how and why using the latter is beneficial to us.

I will learn about Git, which is a distributed version control system and why we prefer using it over all the other distributed version control systems

Next, I will learn about GitHub and get to know that, Git != GitHub.

In this session, I will do everything practically using the command line and see how files move.

Course 2 of 3

HTML and CSS

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications. Learn more below about:

What is HTML?

HTML is the language for describing the structure of Web pages. HTML gives authors the means to:

Publish online documents with headings, text, tables, lists, photos, etc.

Retrieve online information via hypertext links, at the click of a button.

Design forms for conducting transactions with remote services, for use in searching for information, making reservations, ordering products, etc.

Include spread-sheets, video clips, sound clips, and other applications directly in their documents.

With HTML, authors describe the structure of pages using markup. The elements of the language label pieces of content such as "paragraph," "list," "table," and so on.

What is XHTML?

XHTML is a variant of HTML that uses the syntax of XML, the Extensible Markup Language. XHTML has all the same elements (for paragraphs, etc.) as the HTML variant, but the syntax is slightly different. Because XHTML is an XML application, you can use other XML tools with it (such as XSLT, a language for transforming XML content).

What is CSS?

CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation.

What is WebFonts?

WebFonts is a technology that enables people to use fonts on demand over the Web without requiring installation in the operating system. W3C has experience in downloadable fonts through HTML, CSS2, and SVG. Until recently, downloadable fonts have not been common on the Web due to the lack of an interoperable font format.

The WebFonts effort plans to address that through the creation of an industry-supported, open font format for the Web (called "WOFF").

Assignment 1

This module contains a project Submission. Please refer the below link to see the snapshot of the project as per the requirements.

https://github.com/nooobcoder/upGradAssignment/tree/scribbler-project

Rubrics and Instructions for the Project

Hosted Site URL

The site is now accessible at the below URL: https://nooobcoder.github.io/upGradAssignment/

Goals (Milestone)

- Design Part A ✓
- Design Part B ✓
- Design Part C ✓
- Test HTML Validity ✓
- Test JS reporting errors ✓
- Test for performance ✓
- Release for production ✓

Assignment - Course 2

Project Brief

Till now, you have learnt about the basics of HTML, CSS and JavaScript and fiddled around with some of the notable features of each of these languages. In the upcoming project, you would be creating the front-end of a blogging website using HTML, CSS and JavaScript. This blogging website enables to register a new user, create his/her own posts, edit an already existing post, add comments on a post, like a post, etc.

Goal

With the Blogging website, you can put into practice the skills and knowledge that you learned while understanding HTML, CSS and JavaScript. You will be given certain guidelines that will help you create this application. These guidelines will provide the proper instructions as to how you should create the front-end of this webiste. You will be able to put into practice all that you've learned about HTML, CSS and JavaScript in this course.

Overall, developing this website will help you understand how the front-end of real-world websites is created while following the best software development practices.

Towards the end of this project, you would have independently created the front-end of a blogging website! Also, after developing this application, you will feel confident in building the user interface of any website.

Delivery

Note that the project is divided into three parts where each part represents a page in the website. All the pages have their independent project statements and evaluation rubrics. Your website will be evaluated for all the three parts.

Important Points

- You will be given a zipped folder (look for it at the end of this segment) containing the basic folder structure of your blogging project.

Please note that you must not alter the overall folder structure of the blogging project, otherwise your project might be wrongly evaluated. The root directory will consist of an "index.html" file which will be the homepage of your blogging website.

- The root folder must consist of the folders named html, styles, and scripts. All your CSS files must be referenced from and kept inside the styles folder. Similarly, all the HTML files (except "index.html" file) must be placed inside the html folder. All the JavaScript files must be referenced from the scripts folder. You would need to create these folders and segregate your files as per the given guidelines.
- The root folder consists of a folder named images. All the images used in the website should reside in this folder. Note that you are already provided with all the images inside this folder which will be used in the website.
- You can use HTML5, CSS3, Bootstrap, JavaScript, FontAwesome icons, and Google Fonts.

Version Control Best Practices

You must develop this wesbite 'individually'.

You should create a master repository for the application, and push the initial code stub (starter code provided to you at the end of this segment) to this master repository.

Please follow the following best practices when using Git and GitHub to strive towards becoming a better software engineer:

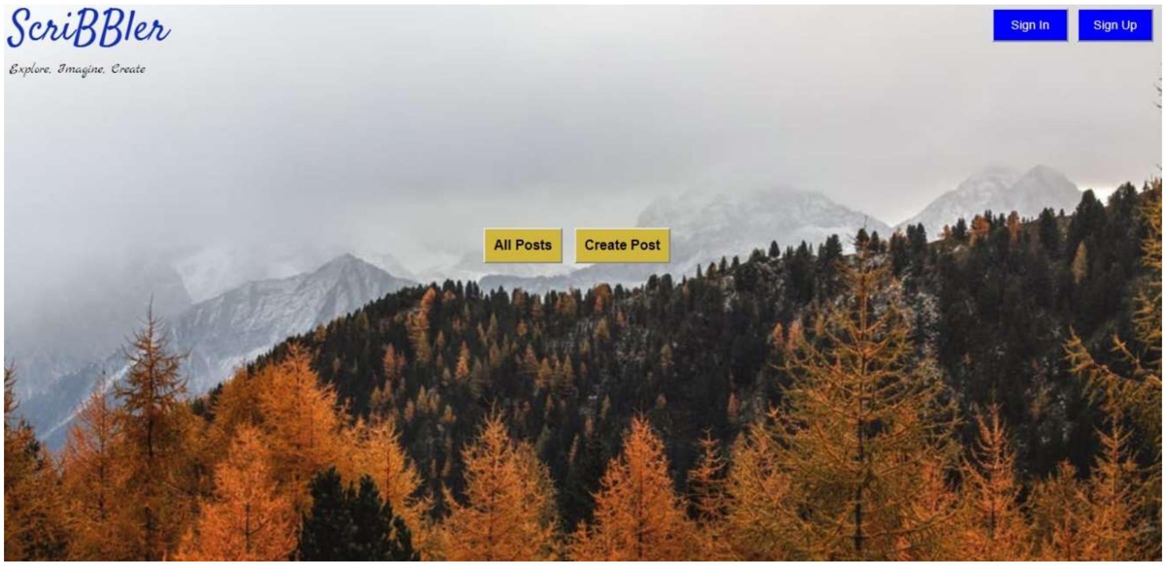
- Commit your code often and frequently.
- Make small and incremental commits.
- Make sure your code works before committing it.
- Write brief and relevant commit messages.

Part A: Creating the Homepage

Relevant files to work on: index.html, index.js, header.js, index.css, header.css

Once you download the zipped folder, you will see a file called "index.html" at the root level of that folder. You must open that file and add your code in that file.

This "index.html" file will be the file of your homepage. The final outcome of creating your homepage should look like this:-



Some very important things to keep in mind:

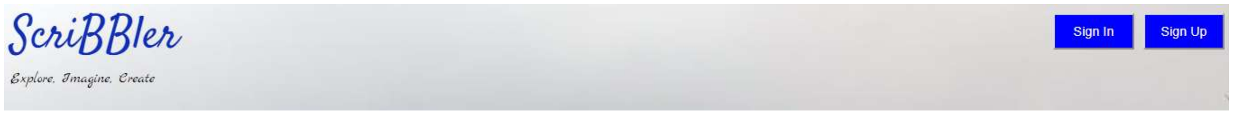
- You must strictly keep HTML, CSS and Javascript portion of the code separate from each other (in separate files)
- You must refer to the "index.css" file inside the "css" folder and define all your css inside that file (for the "index.html" file)
- You must refer to the "index.js" file inside the "js" folder and define all your Javascript inside that file (for the "index.html" file)
- Unless explicitly mentioned otherwise (for example, at some places, it might be mentioned that a certain button MUST be red in colour), you can customize your own CSS for any HTML element you want. However, please keep in mind that the overall appearance of any HTML page must be very similar (if not identical) to the outcome of the screen mentioned prior of each part of the assignment. This means that if certain elements are placed towards the top-right of the screen in the final outcome figure, they MUST be placed at the top-right corner ONLY, failing to do so will lead to deduction of marks. Note that there might be multiple ways in which a certain CSS effect can be achieved. So the exact syntax of the CSS code does not matter, but the overall appearance of any given page must be very similar to the expected outcome.

Let us now begin our step by step process of creating our homepage.

Our homepage will consist of 2 main segments:-

a) Header = You will be re-using it at many other places in the assignment. Hence, we will keep it as a separate class altogether

This will be the header portion which should look like this:



You are already provided with the Scribbler Logo and The tagline. You've also been provided with the buttons. You need to add functionality to them.

You will need to create a total of 2 modals for the header. You must find on the Internet ways to create a simple basic modal. You can use the below link to find a simple modal code:

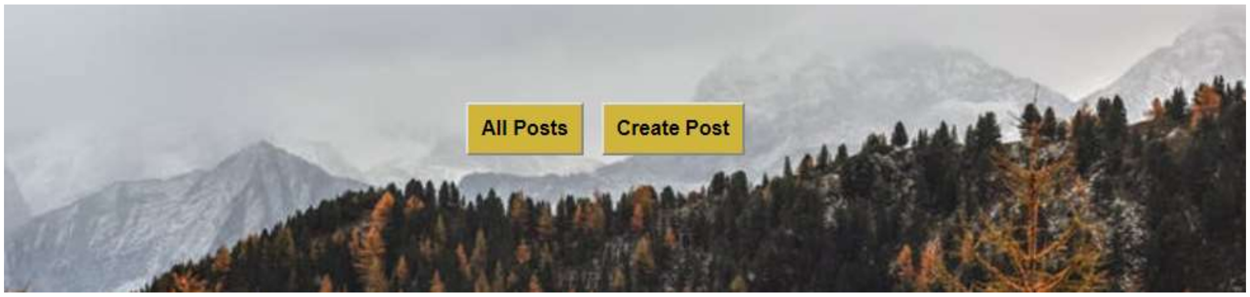
You have been provided with the Create Post Modal in the Stub provided to you. You can reference that for further information.

https://www.w3schools.com/howto/howto_css_modals.asp

Dont forget to keep different div id's for the different modals, so that they are distinct from one another!

b) The segment called "postButtons" consist of 2 buttons -

"All Posts" and **"Create Post"**



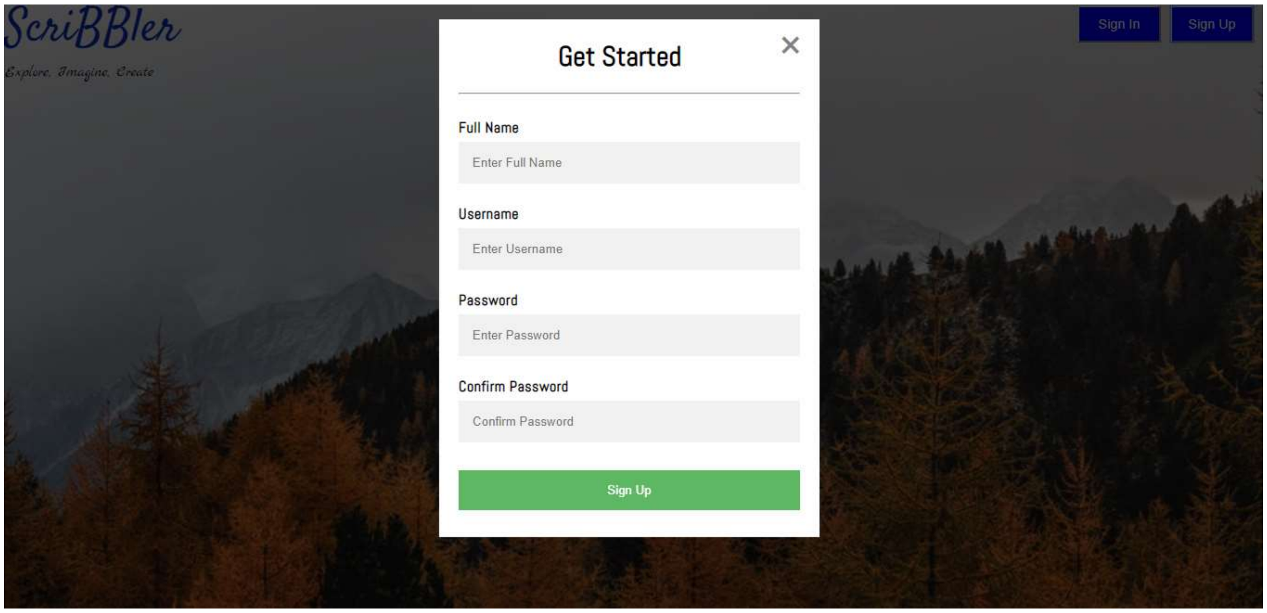
Let us follow some steps to modify our header inside the <div class="header"> :-

Inside the <div class="header">, you need to create:-

On clicking the "Sign Up" button, a modal should appear on screen. (You must search on the Internet yourself how you can create a simple modal or you can take a reference from the 'Create Post' modal already created). You can use the below link to find a simple modal code:

- https://www.w3schools.com/howto/howto_css_modals.asp

The modal corresponding to the "Sign Up" button must look like this:

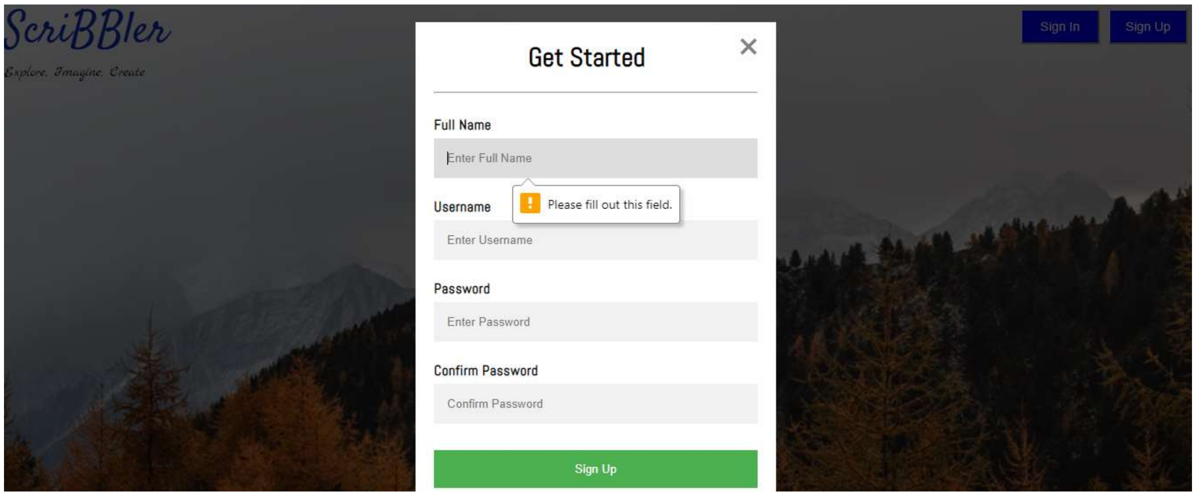


- The title of the modal should be "Get Started". It should be displayed in the center of the modal and you can use any font you wish. You should also display a tiny cross button towards the top-right, which when clicked, would close this modal.
- This modal would consist of 4 input boxes - for "Full Name", "Username", "Password" and "Confirm Password". Note that the input type of each of these should be - text, text, password, password - respectively.

Also note that when nothing is typed into the input boxes yet, they should display a placeholder with the text "Enter Full Name", "Enter Username", "Enter Password" & "Confirm Password" respectively.

You can use any font you want for this.

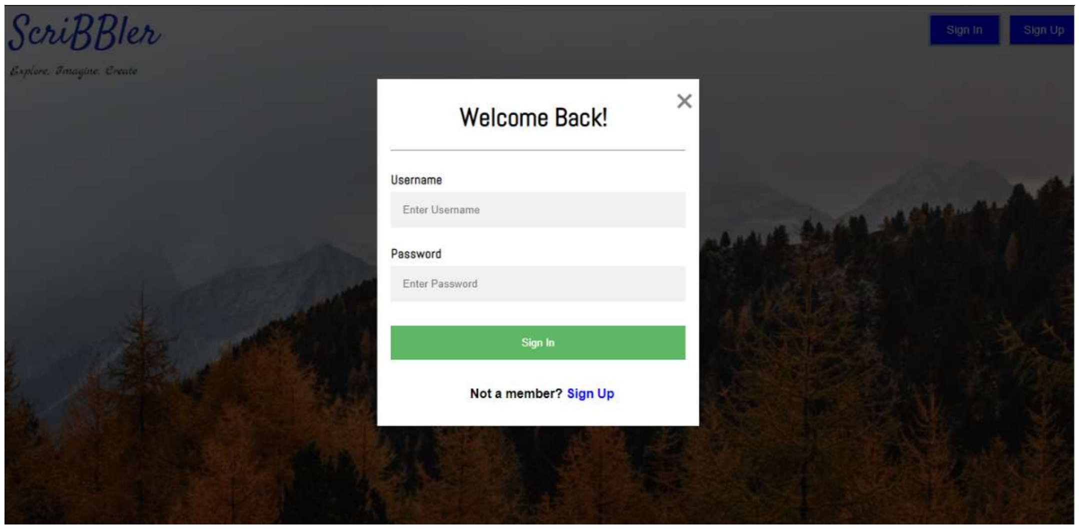
- Below these placeholders, you must create a green-coloured "Sign Up" button. Note that you must make sure that the above 4 input fields are mandatorily filled in by the user (the input fields must be made "required"), without which he/she cannot proceed with clicking the "Sign Up" button. For example, when a user tries to proceed without filling all the fields, such an indication would be displayed:



Note that you can do this simply by making all the input fields as “required”

You do not need to define any specific action on clicking the Sign Up button.

- On clicking the "Sign In" button, a modal should appear on screen. The modal corresponding to the "Sign In" button must look like this:



- The title of the modal should be "Welcome Back!". You should also display a tiny cross button towards the top-right, which when clicked, would close this modal.

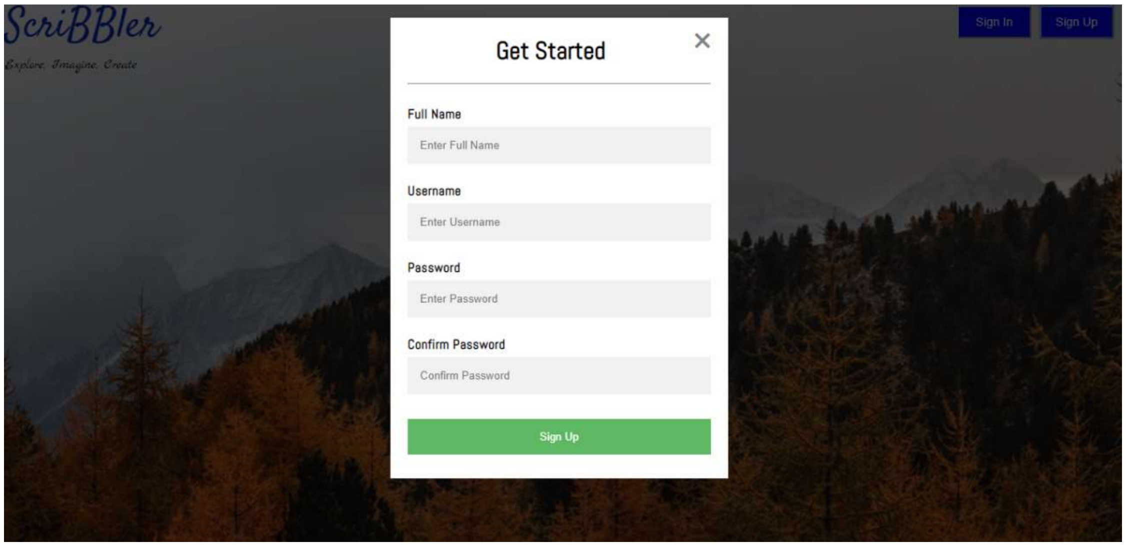
- This modal would consist of 2 input boxes - for "Username" and "Password". Note that the input type of each of these should be - text and password - respectively.

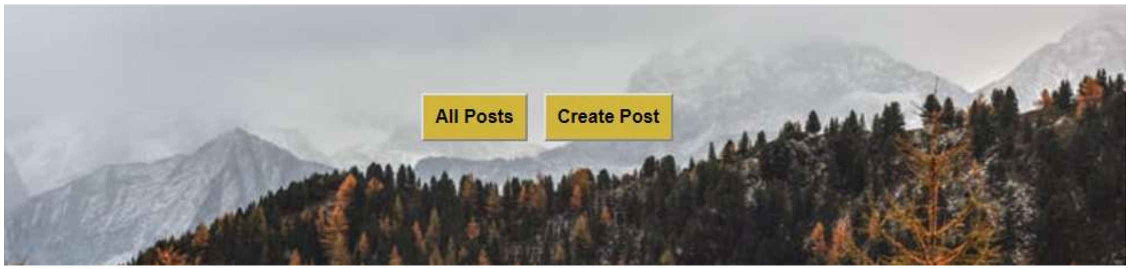
- Note that when nothing is typed into the input boxes yet, they should display a placeholder with the text "Enter Username" and "Enter Password" respectively.

- Below these placeholders, you must create a green-coloured "Sign In" button. Note that you must make sure that the above 2 input fields are mandatorily filled in by the user (the input fields must be made "required"), without which he/she cannot proceed with clicking the "Sign In" button.

- You need not define any specific action on clicking the Sign In button.

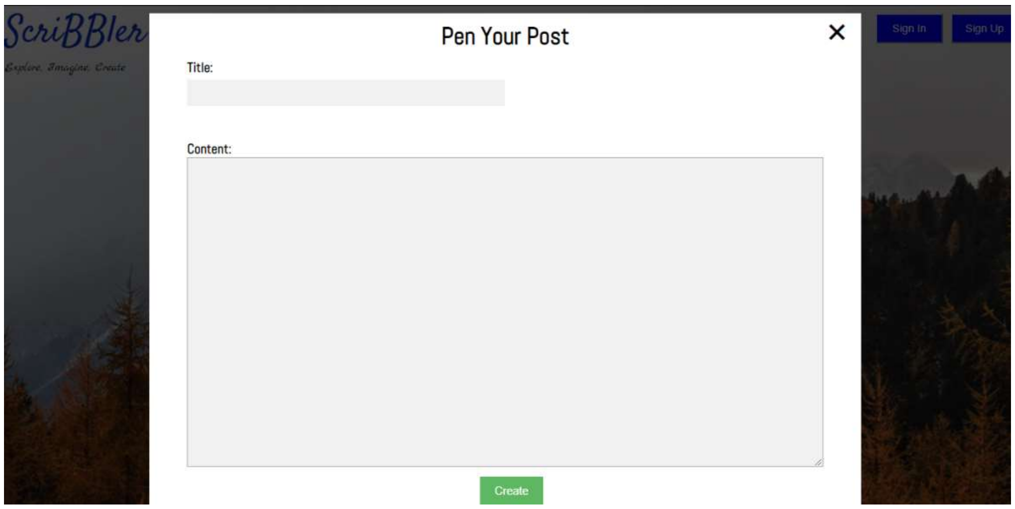
- Below the "Sign In" button, you must display a text called "Not a member? Sign Up". The "Not a member?" portion of the text should be in black, whereas the "Sign Up" portion of the text should be a hyperlink, which when clicked, would hide the sign in modal and display the sign-up modal which you created in the previous segment like this:





- On clicking the "All Posts" button, it should redirect to a file called "bloglist.html" which will be placed inside our html folder. (For now, this will be an empty page, which we would create fully in the next part of our assignment)
- On clicking the "Create Post" button, it should display a modal like this:

a) A narrow input box for "Title:"
b) The title of the modal should be "Pen Your Post" as shown in the image below
c) A text area for "Content:"
d) Below them, create a green-coloured "Create" button at the centre which at the present moment, would not do anything when you click on it (later on, we will write an API request for posting a new blog article on the backend, but for now, you don't need to worry about it). The modal should look like this:

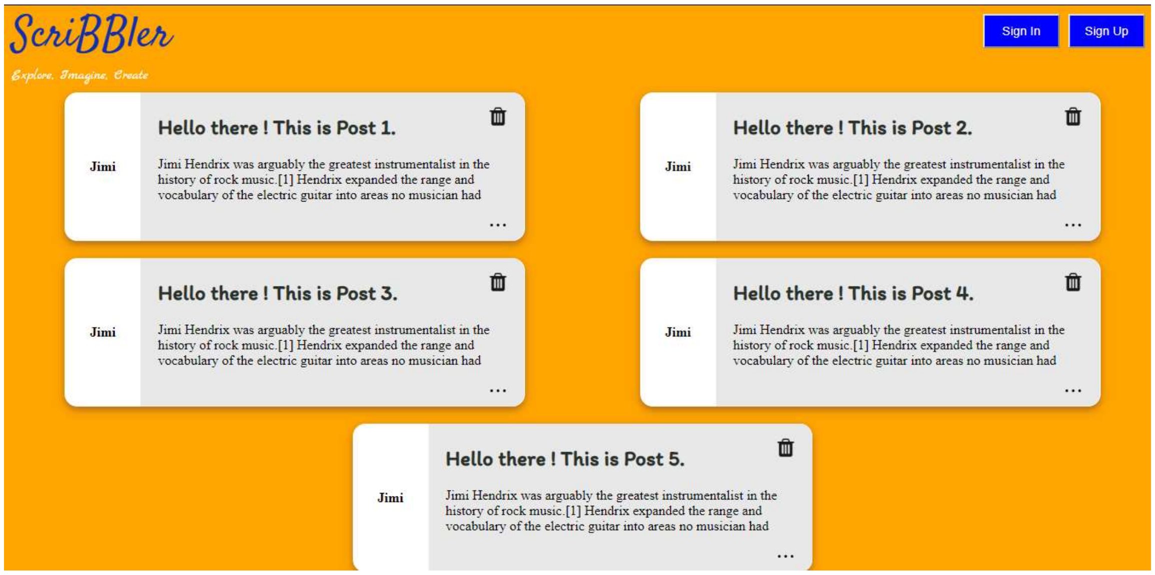


Part B: Creating list of blogs

In this section, we would be creating a page called "bloglist.html" which would display the list of all blog posts.

Relevant files to work on: bloglist.html, bloglist.css, bloglist.js

Recall that in the previous section, you created an "All Posts" button on the centre of the screen which when clicked, would redirect to a "bloglist.html" page. We would be designing this page ("bloglist.html") in this section. The final outcome of creating this page would be like this:-



Jimi
Hello there ! This is Post 1.
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ...

Jimi
Hello there ! This is Post 2.
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ...

Jimi
Hello there ! This is Post 3.
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ...

Jimi
Hello there ! This is Post 4.
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ...

Jimi
Hello there ! This is Post 5.
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ...

Some very important things to keep in mind:

- You must strictly keep HTML, CSS and Javascript portion of the code separate from each other (in separate files)
- You must refer to the "bloglist.css" file inside the "css" folder and define all your css inside that file (for the "bloglist.html" file)
- You must refer to the "bloglist.js" file inside the "js" folder and define all your Javascript inside that file (for the "bloglist.html" file)
- Unless explicitly mentioned (for example, at some places, it might be mentioned that a certain button MUST be red in colour), you can customize your own CSS for any HTML element you want. However, please keep in mind that the overall appearance of any HTML page must be very similar (if not identical) to the outcome of the screen mentioned prior of each part of the assignment. This means that if certain elements are placed towards the top-right of the screen in the final outcome figure, they MUST be placed at the top-right corner ONLY, failing to do so will lead to deduction of marks. Note that there might be multiple ways in which a certain CSS effect can be achieved. So the exact syntax of the CSS code does not matter, but the overall appearance of any given page must be very similar to the expected outcome.

Let us move step-by-step to create this page:-

- Write the header class into our page.

Carefully place the code for the header class which you created in the last section of the assignment. Basically, we want the header part of the page to be exactly identical to the header portion which we created for the last section.

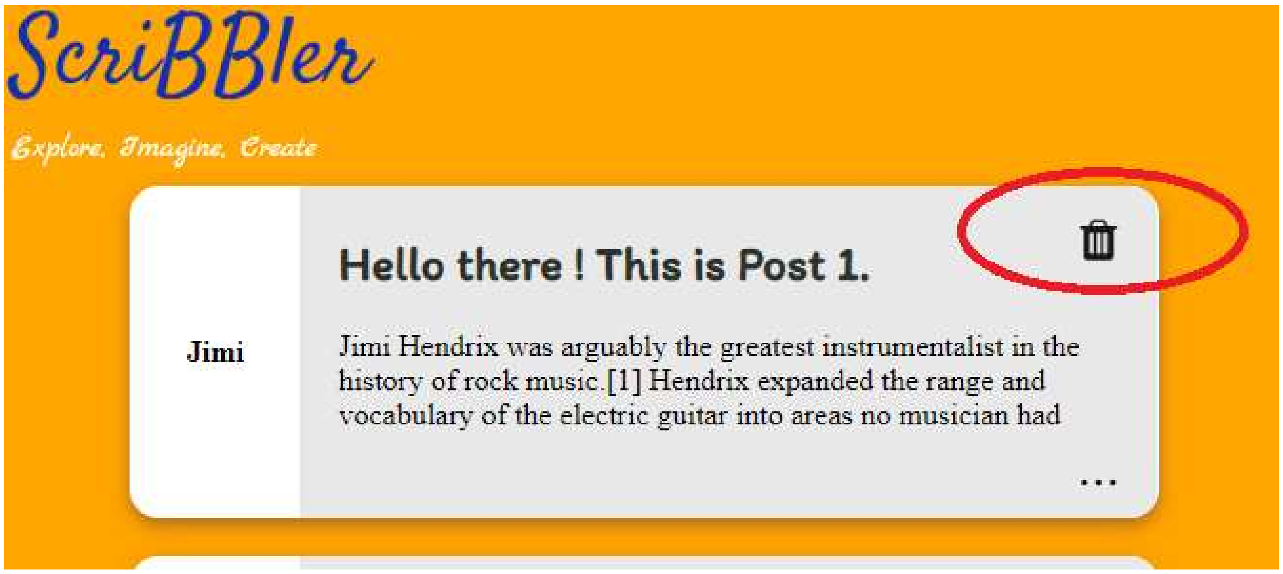
- Below the header, you will create a segment which will consist of 5 identical blog posts and place them next to each other in pairs. Each blog post will display the username of the user towards the left (which will be already provided). Keep the background color of the page as orange. One such blog post without any CSS has been provided to you. You need to make the new blog post as the copy of this blog post. We would suggest that you modify this blog post first and then copy paste this blog post to the new one.

a) Towards the right of the username, you need to partially display the text of the blog. Note that you must not display the entire text of the blog on this page itself. (You must search the Internet to find out a way such that for each blog, the text is only partially displayed on this page.

A hint: there is a CSS display property which enables you to partially hide content. (You can search on the Internet about it)

You need not alter the sample text already specified in the blog author/title/body. You only need to change its appearance.

b) You need to add a "trash" icon towards the top-right of each blog post (inside each element of the "post-content" class) like this:



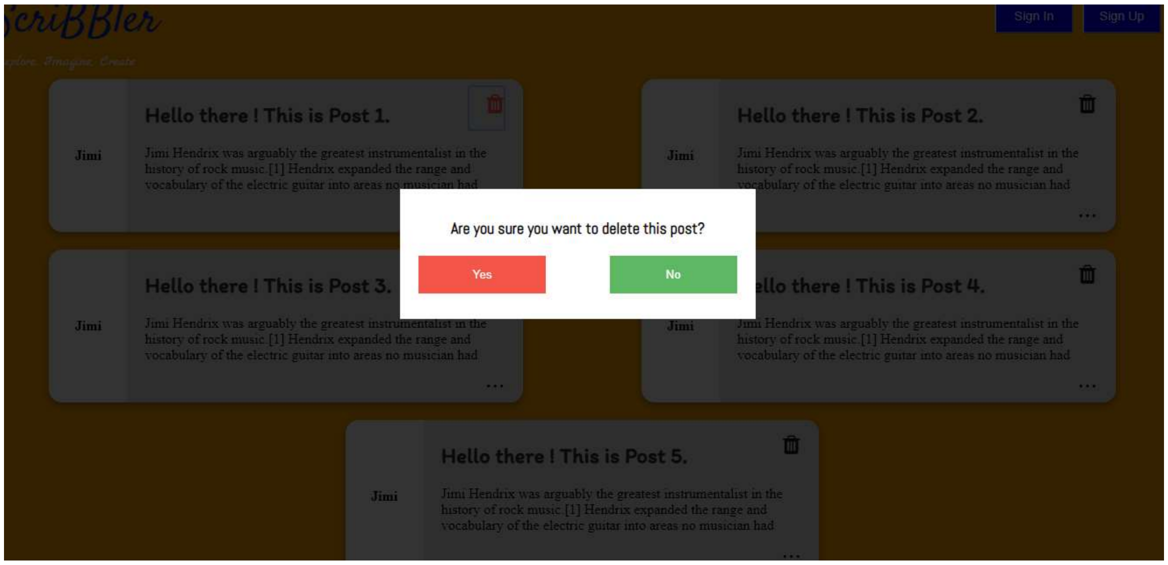
Jimi

Hello there ! This is Post 1.

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

...

On clicking the trash icon, it should display the following modal like this:-



Hello there ! This is Post 1.

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

Hello there ! This is Post 2.

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

Are you sure you want to delete this post?

Hello there ! This is Post 3.

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

llo there ! This is Post 4.

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

Hello there ! This is Post 5.

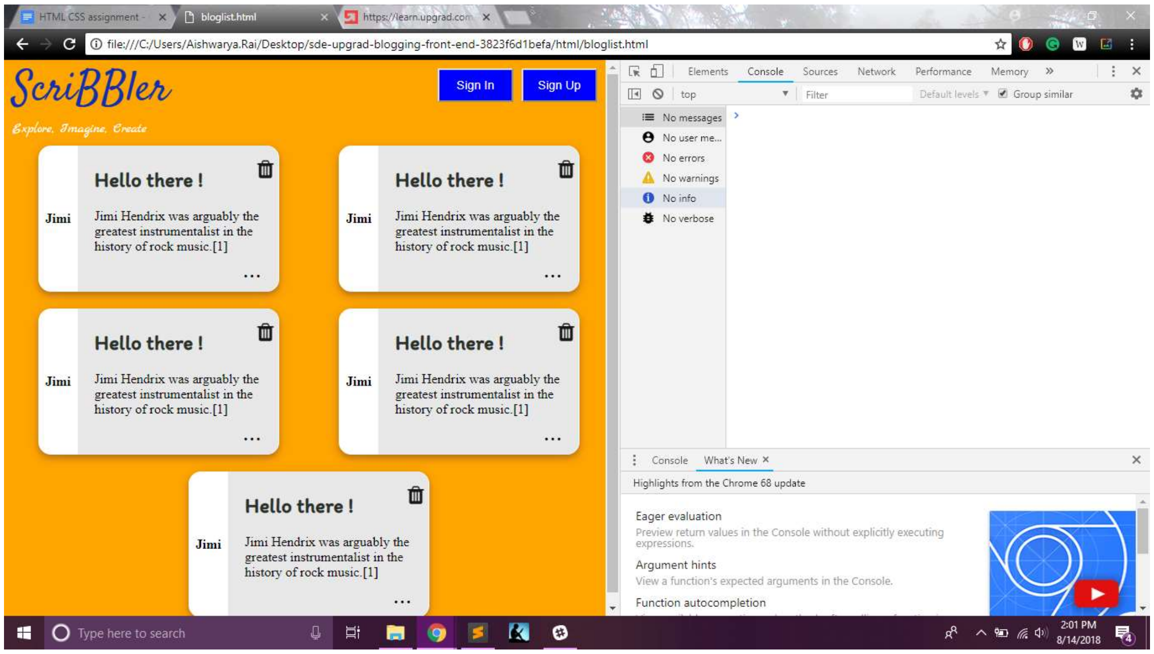
Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

This modal would consist of a red-coloured "Yes" button and a green-coloured "No" button. No action would take place on clicking the "Yes" button. On clicking the "No" button, the modal should disappear and the bloglist page should appear again as it is.

- You need to adjust the width/height/flex/margin/other properties of each blog post carefully such that only two of such blog posts appear in one line.

Additionally, you need to make the display of these boxes flexible so that when you change the display size of the screen, the blog posts adjust their placement automatically. This means that whatever screen size we reduce our screen to, there should be exactly two blog posts in each row. They should adjust their height/width automatically when the screen is resized. For example, if I resize my window, the blog posts should get adjusted on their own like this:



Hello there !

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1]

Hello there !

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1]

Hello there !

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1]

Hello there !

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1]

Hello there !

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1]

Highlights from the Chrome 68 update

Eager evaluation

Preview return values in the Console without explicitly executing expressions.

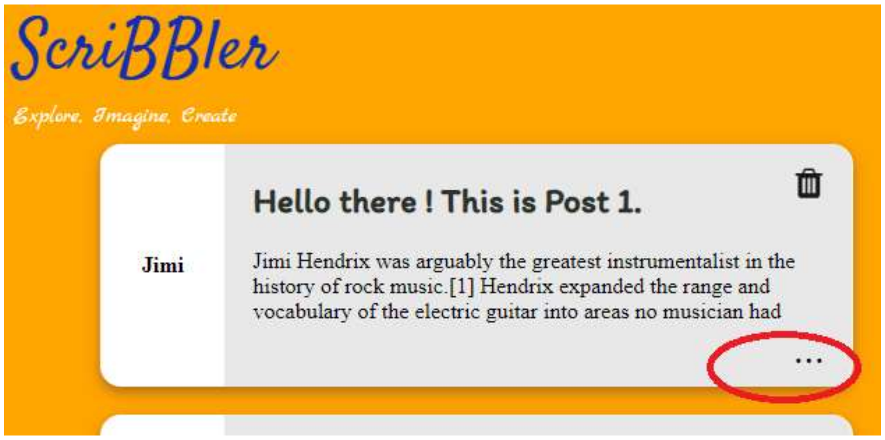
Argument hints

View a function's expected arguments in the Console.

Function autocompletion

You must search online to find out a way to display only two blog posts at a time such that every new post either gets added to the right or below the current post.

d) There is a small icon/text "..." next to each post like this:



Hello there ! This is Post 1.

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had

On clicking this icon/text, it should open the individual blog post. As of now, what you can do, is redirect it to a new page called as "post.html" whenever the "..." button is clicked.

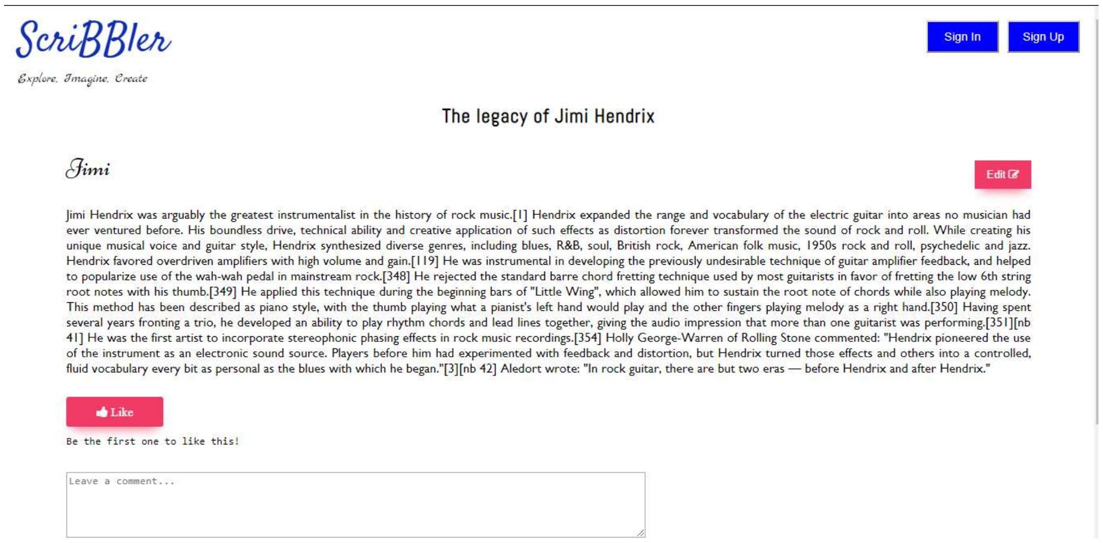
Part C: Creating Blog Post

In this section, you would be creating the functionality of the "post.html" page.

Relevant files to work on: **post.html, post.css, post.js**

This page will consist of the individual blog post of our blogging website. The final outcome of this page would look like this:

Image 1:

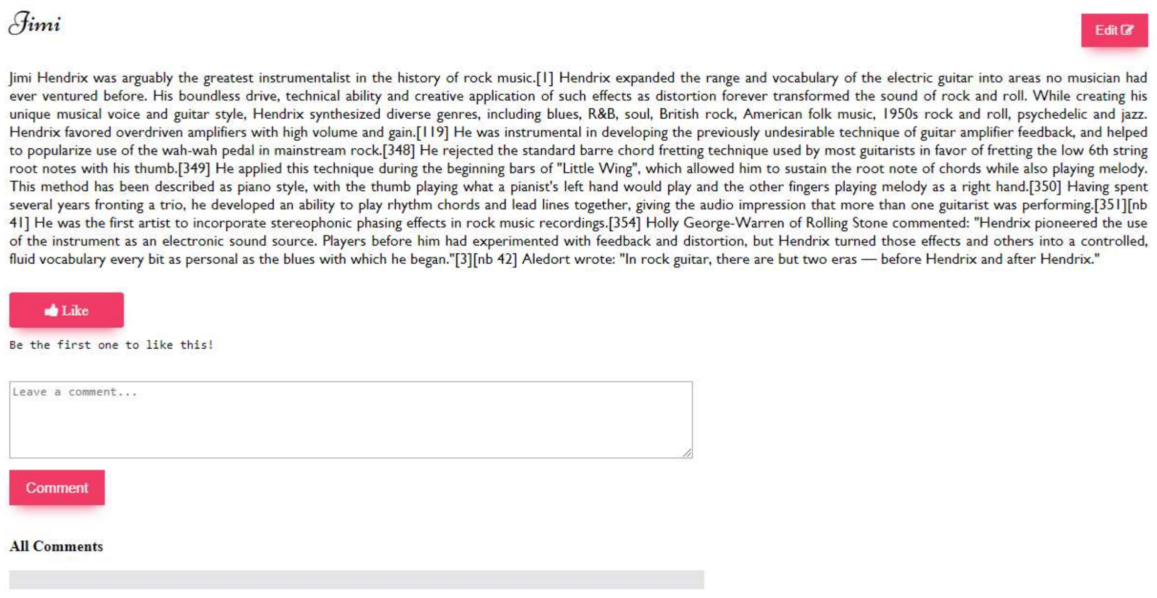


The legacy of Jimi Hendrix

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Be the first one to like this!

Image 2:


Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Be the first one to like this!

All Comments

Some very important things to keep in mind:

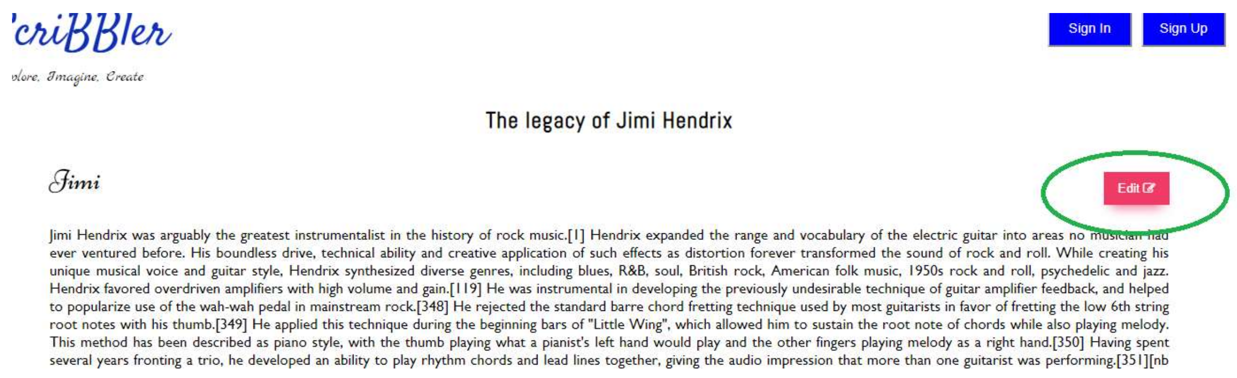
- You must strictly keep HTML, CSS and Javascript portion of the code separate from each other.
- You must refer to the "post.css" file inside the "css" folder and define all your css inside that file for the "post.html" file
- You must refer to the "post.js" file inside the "js" folder and define all your Javascript inside that file for the "post.html" file

Unless explicitly mentioned otherwise (for example, at some places, it might be mentioned that a certain button MUST be red in colour), you can customize your own CSS for any HTML element you want. However, please keep in mind that the overall appearance of any HTML page must be very similar (if not identical) to the outcome of the screen mentioned prior of each part of the assignment. This means that if certain elements are placed towards the top-right of the screen in the final outcome figure, they MUST be placed at the top-right corner ONLY, failing to do so will lead to deduction of marks. Note that there might be multiple ways in which a certain CSS effect can be achieved. So the exact syntax of the CSS code does not matter, but the overall appearance of any given page must be very similar to the expected outcome.

Write the header class into our page.

Carefully place the code for the header class which you created in the last section of the assignment. Basically, we want the header part of the page to be exactly identical to the header portion which we created for the last section.

a) You need to add functionality to the "Edit" button that is positioned towards the top-right corner of the blog body, which would look like this:

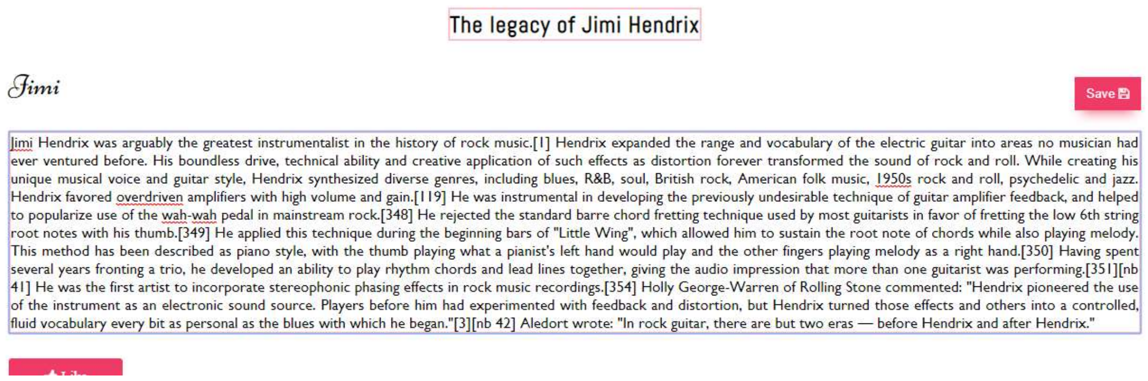


The legacy of Jimi Hendrix

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb

When clicked, this edit button would make the blog body editable like this:



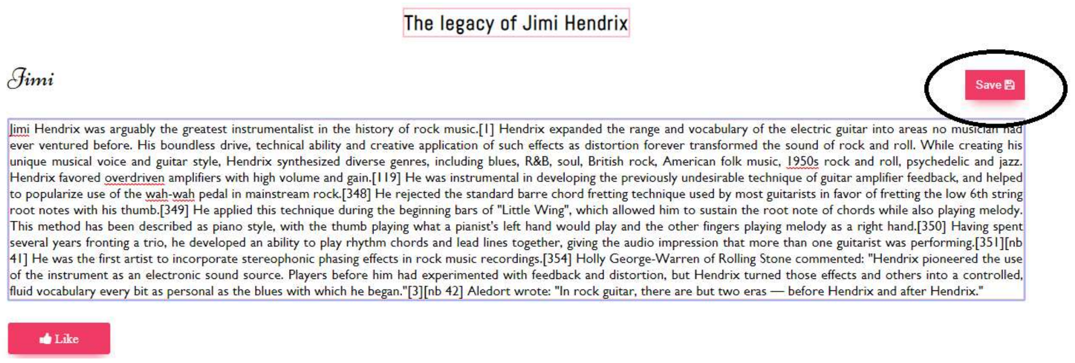
The legacy of Jimi Hendrix

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Note that when the blog body and title are in the "editable" mode, there should be a cursor which would enable the user to edit/erase/add any text he/she wants in it.

While the blog body is in the editable mode, a button called "Save" should be displayed in place of the original "Edit" button like this:



The legacy of Jimi Hendrix

Jimi

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

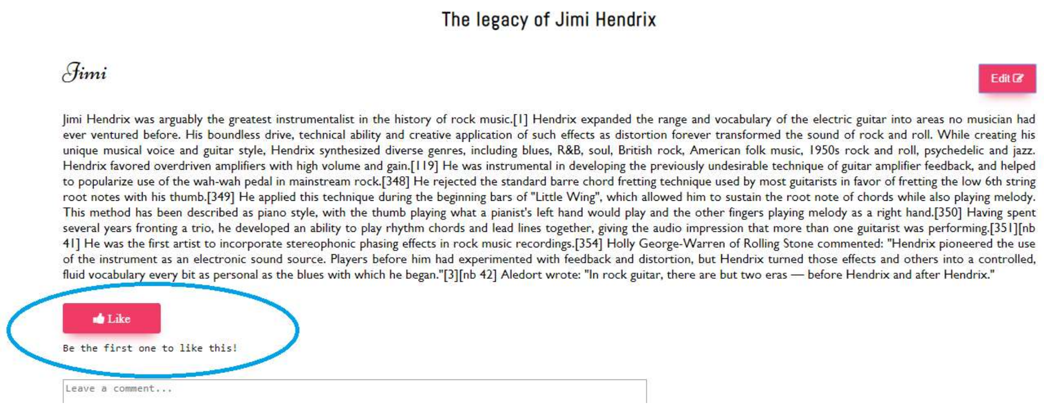
The blog body/title should be placed inside a textarea when it is in the Edit mode.

You need not add any icon/picture in the Save button and just a written text "Save" would be enough.

This Edit button need not necessarily be on the same horizontal line as the blog author name.

While the editable mode is on, the user should be able to edit the text of the blog. When the user is done editing the blog body, he should click on the "Save" button. On clicking the "Save" button, the blog should be saved as it is (new changes which the user has done must be saved). The blog body should reflect the changes that we made in edit mode. After the edit mode is over, the blog body/title should no longer be editable and the "Edit" button should reappear (in place of the "Save" button).

b) Below the body of the blog, there is a "Like" button. In the default state, a statement called "Be the first one to like this!" is displayed just below the "Like" button.


The legacy of Jimi Hendrix

Jimi

Edit

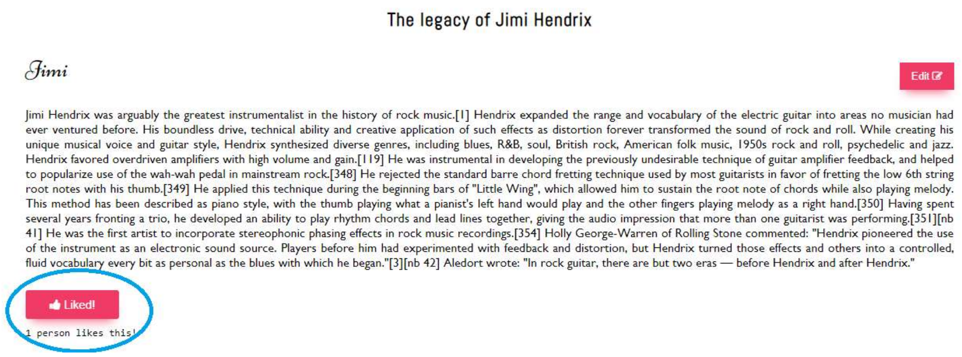
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Like

Be the first one to like this!

Leave a comment...

On liking the blog for the first time, this statement must be updated to "1 person likes this!" and the "Like" button must morphe into a button "Liked!" as shown in the figure.


The legacy of Jimi Hendrix

Jimi

Edit

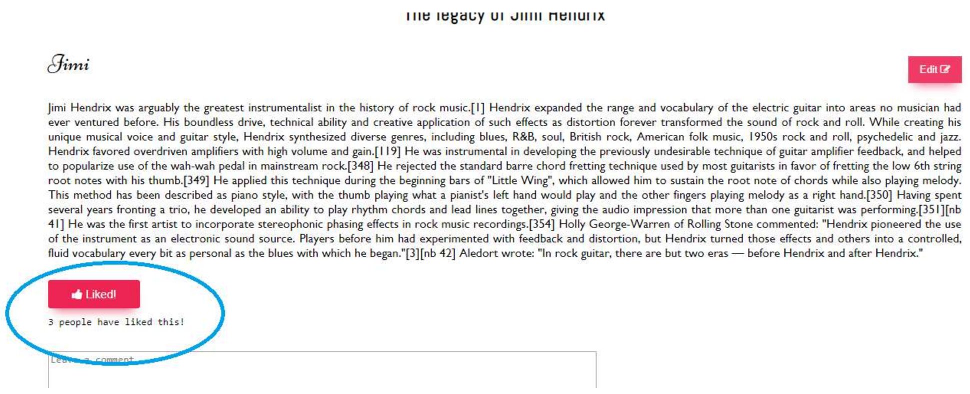
Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Liked!

1 person likes this!

You need not add any icon/picture in the Like button and just a written text "Liked" would be enough.

On liking the blog for the first time, this statement must be updated to "1 person likes this!" and the "Like" button must morphe into a button "Liked!" as shown in the figure.


The legacy of Jimi Hendrix

Jimi

Edit

Jimi Hendrix was arguably the greatest instrumentalist in the history of rock music.[1] Hendrix expanded the range and vocabulary of the electric guitar into areas no musician had ever ventured before. His boundless drive, technical ability and creative application of such effects as distortion forever transformed the sound of rock and roll. While creating his unique musical voice and guitar style, Hendrix synthesized diverse genres, including blues, R&B, soul, British rock, American folk music, 1950s rock and roll, psychedelic and jazz. Hendrix favored overdriven amplifiers with high volume and gain.[119] He was instrumental in developing the previously undesirable technique of guitar amplifier feedback, and helped to popularize use of the wah-wah pedal in mainstream rock.[348] He rejected the standard barre chord fretting technique used by most guitarists in favor of fretting the low 6th string root notes with his thumb.[349] He applied this technique during the beginning bars of "Little Wing", which allowed him to sustain the root note of chords while also playing melody. This method has been described as piano style, with the thumb playing what a pianist's left hand would play and the other fingers playing melody as a right hand.[350] Having spent several years fronting a trio, he developed an ability to play rhythm chords and lead lines together, giving the audio impression that more than one guitarist was performing.[351][nb 41] He was the first artist to incorporate stereophonic phasing effects in rock music recordings.[354] Holly George-Warren of Rolling Stone commented: "Hendrix pioneered the use of the instrument as an electronic sound source. Players before him had experimented with feedback and distortion, but Hendrix turned those effects and others into a controlled, fluid vocabulary every bit as personal as the blues with which he began."[3][nb 42] Aledort wrote: "In rock guitar, there are but two eras — before Hendrix and after Hendrix."

Liked!

3 people have liked this!

Leave a comment...

c) Below the "Like" button, there is an input box with a placeholder text "Leave a comment...". The user can type his comments in this box.

d) Below this box, there is a red coloured "Comment" button, which when clicked must display the typed comment inside the "All Comments" section.

Note that every new comment should be added at the TOP as explained in the figure. You can style the comment section in the way you want, just keeping in mind that individual comments have white as a background and are separated from each other. The background of the entire comment box should be grey as shown in the figure.

Course 3 of 3

Advanced JavaScript and ReactJS

About JavaScript:

JavaScript (often shortened to **JS**) is a lightweight, interpreted, object-oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.

JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behavior.

Contrary to popular misconception, **JavaScript is** *not* **"Interpreted Java"**. In a nutshell, JavaScript is a dynamic scripting language supporting prototype based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language. Language constructs, such as if statements, for and while loops, and switch and try ... catch blocks function the same as in these languages (or nearly so).

JavaScript can function as both a procedural and an object oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects **at run time**, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.

JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation (via eval), object introspection (via for ... in), and source code recovery (JavaScript programs can decompile function bodies back into their source text).

For a more in depth discussion of JavaScript programming follow the JavaScript resources links below.

What JavaScript implementations are available?

The Mozilla project provides two JavaScript implementations. The first **ever** JavaScript was created by Brendan Eich at Netscape, and has since been updated to conform to ECMA-262 Edition 5 and later versions. This engine, code named SpiderMonkey, is implemented in C/C++. The Rhino engine, created primarily by Norris Boyd (also at Netscape) is a JavaScript implementation written in Java. Like SpiderMonkey, Rhino is ECMA-262 Edition 5 compliant.

Several major runtime optimizations such as TraceMonkey (Firefox 3.5), JägerMonkey (Firefox 4) and IonMonkey were added to the SpiderMonkey JavaScript engine over time. Work is always ongoing to improve JavaScript execution performance.

Besides the above implementations, there are other popular JavaScript engines such as:-

Google's V8, which is used in the Google Chrome browser and recent versions of Opera browser. This is also the engine used by Node.js.

The JavaScriptCore (SquirrelFish/Nitro) used in some WebKit browsers such as Apple Safari.

The Chakra engine used in Internet Explorer (although the language it implements is formally called "JScript" in order to avoid trademark issues).

Each of Mozilla's JavaScript engines expose a public API which application developers can use to integrate JavaScript into their software. By far, the most common host environment for JavaScript is web browsers. Web browsers typically use the public API to create **host objects** responsible for reflecting the DOM into JavaScript.

Another common application for JavaScript is as a (Web) server side scripting language. A JavaScript web server would expose host objects representing a HTTP request and response objects, which could then be manipulated by a JavaScript program to dynamically generate web pages. Node.js is a popular example of this.

About ReactJS:

React is a JavaScript library created for building fast and interactive user interfaces for web and mobile applications. It is an open-source, component-based, front-end library responsible only for the application's view layer. In Model View Controller (MVC) architecture, the view layer is responsible for how the app looks and feels. React was created by Jordan Walke, a software engineer at Facebook.

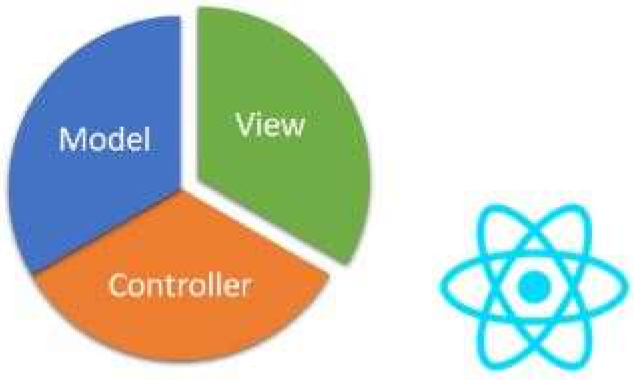


Fig: MVC architecture

Let's take a look at an Instagram webpage example, entirely built using React, to get a better understanding of how React works. As the illustration shows, React divides the UI into multiple components, which makes the code easier to debug. This way, each component has its property and function.

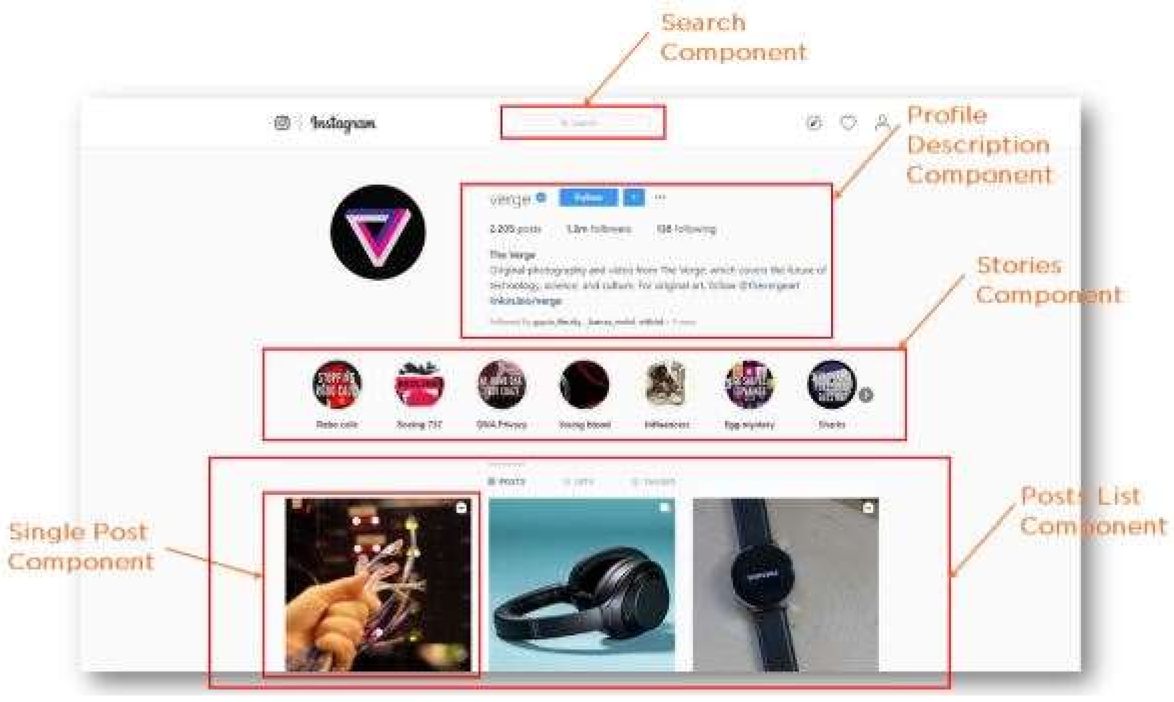


Fig: Instagram Components

Now that we know what React is let’s move on and see why React is the most popular front-end library for web application development.

Why React?

React’s popularity today has eclipsed that of all other front-end development frameworks. Here is why:

Easy creation of dynamic applications: React makes it easier to create dynamic web applications because it requires less coding and offers more functionality, as opposed to JavaScript, where coding often gets complex very quickly.

Improved performance: React uses Virtual DOM, thereby creating web applications faster. Virtual DOM compares the components’ previous states and updates only the items in the Real DOM that were changed, instead of updating all of the components again, as conventional web applications do.

Reusable components: Components are the building blocks of any React application, and a single app usually consists of multiple components. These components have their logic and controls, and they can be reused throughout the application, which in turn dramatically reduces the application’s development time.

Unidirectional data flow: React follows a unidirectional data flow. This means that when designing a React app, developers often nest child components within parent components. Since the data flows in a single direction, it becomes easier to debug errors and know where a problem occurs in an application at the moment in question.

Small learning curve: React is easy to learn, as it mostly combines basic HTML and JavaScript concepts with some beneficial additions. Still, as is the case with other tools and frameworks, you have to spend some time to get a proper understanding of React’s library.

It can be used for the development of both web and mobile apps: We already know that React is used for the development of web applications, but that’s not all it can do. There is a framework called React Native, derived from React itself, that is hugely popular and is used for creating beautiful mobile applications. So, in reality, React can be used for making both web and mobile applications.

Dedicated tools for easy debugging: Facebook has released a Chrome extension that can be used to debug React applications. This makes the process of debugging React web applications faster and easier.

The above reasons more than justify the popularity of the React library and why it is being adopted by a large number of organizations and businesses. Now let’s familiarize ourselves with React’s features.

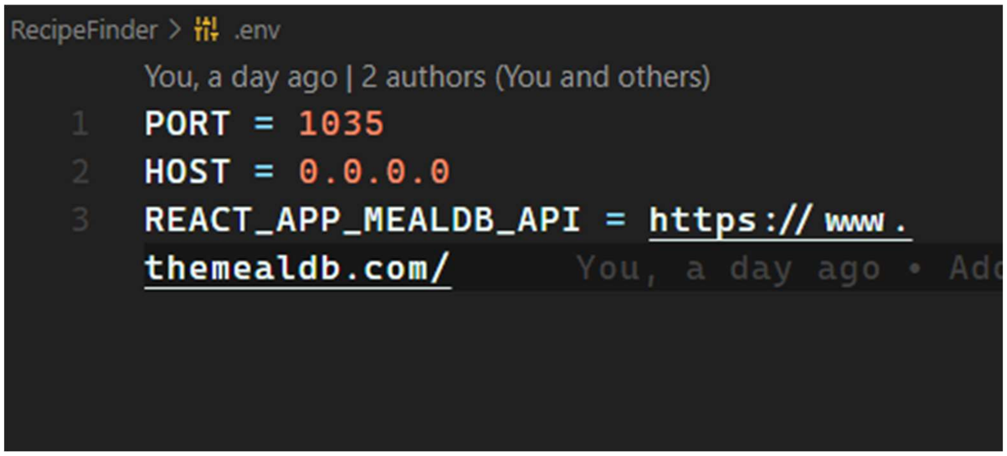
Assignment 2
GitHub URL: https://github.com/nooobcoder/upGradAssignment/tree/recipe-finder/RecipeFinder

Site Deployed on: https://sharp-tesla-44f31a.netlify.app/

TODOS

☑ The app is highly dependent on the Meal DB API Create an .env file in the root of the folder (if does not exists) and specify the following key

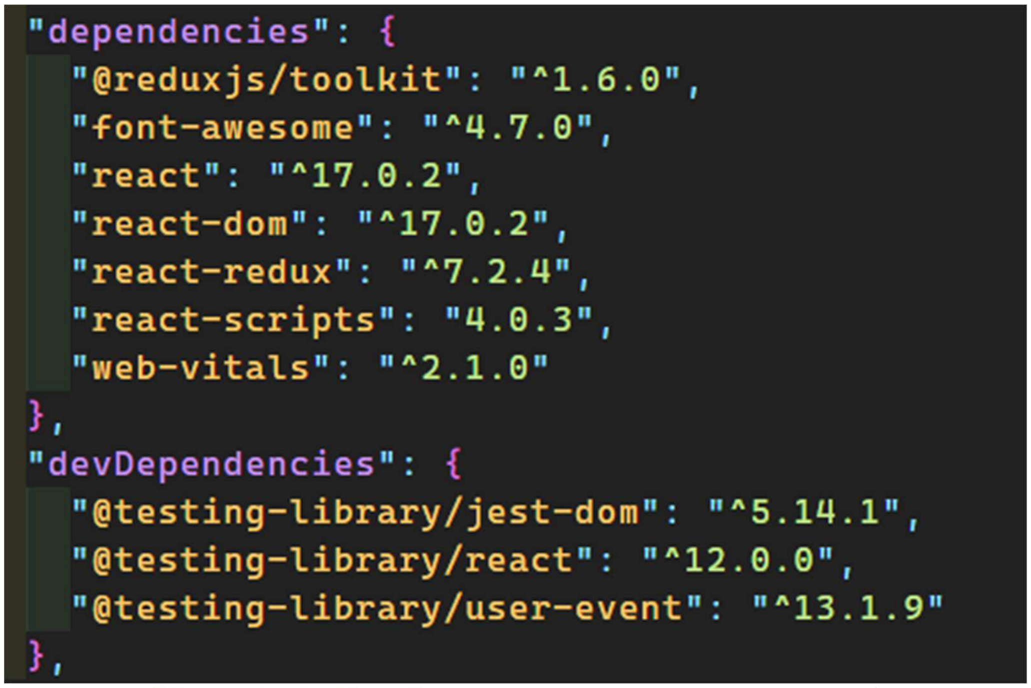
KEY	VALUE
REACT_APP_MEALDB_API	https://www.themealdb.com/



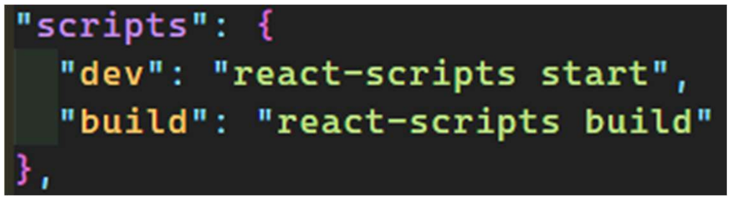
```
PORT = 1035
HOST = 0.0.0.0
REACT_APP_MEALDB_API = https:// www.
themealdb.com/      You, a day ago • Add
```

Build Steps

After you have cloned the repository, head out to the **RecipeFinder** folder and have a look at the package.json file in the root of that folder.



```
"dependencies": {
  "@reduxjs/toolkit": "^1.6.0",
  "font-awesome": "^4.7.0",
  "react": "^17.0.2",
  "react-dom": "^17.0.2",
  "react-redux": "^7.2.4",
  "react-scripts": "4.0.3",
  "web-vitals": "^2.1.0"
},
"devDependencies": {
  "@testing-library/jest-dom": "^5.14.1",
  "@testing-library/react": "^12.0.0",
  "@testing-library/user-event": "^13.1.9"
},
```

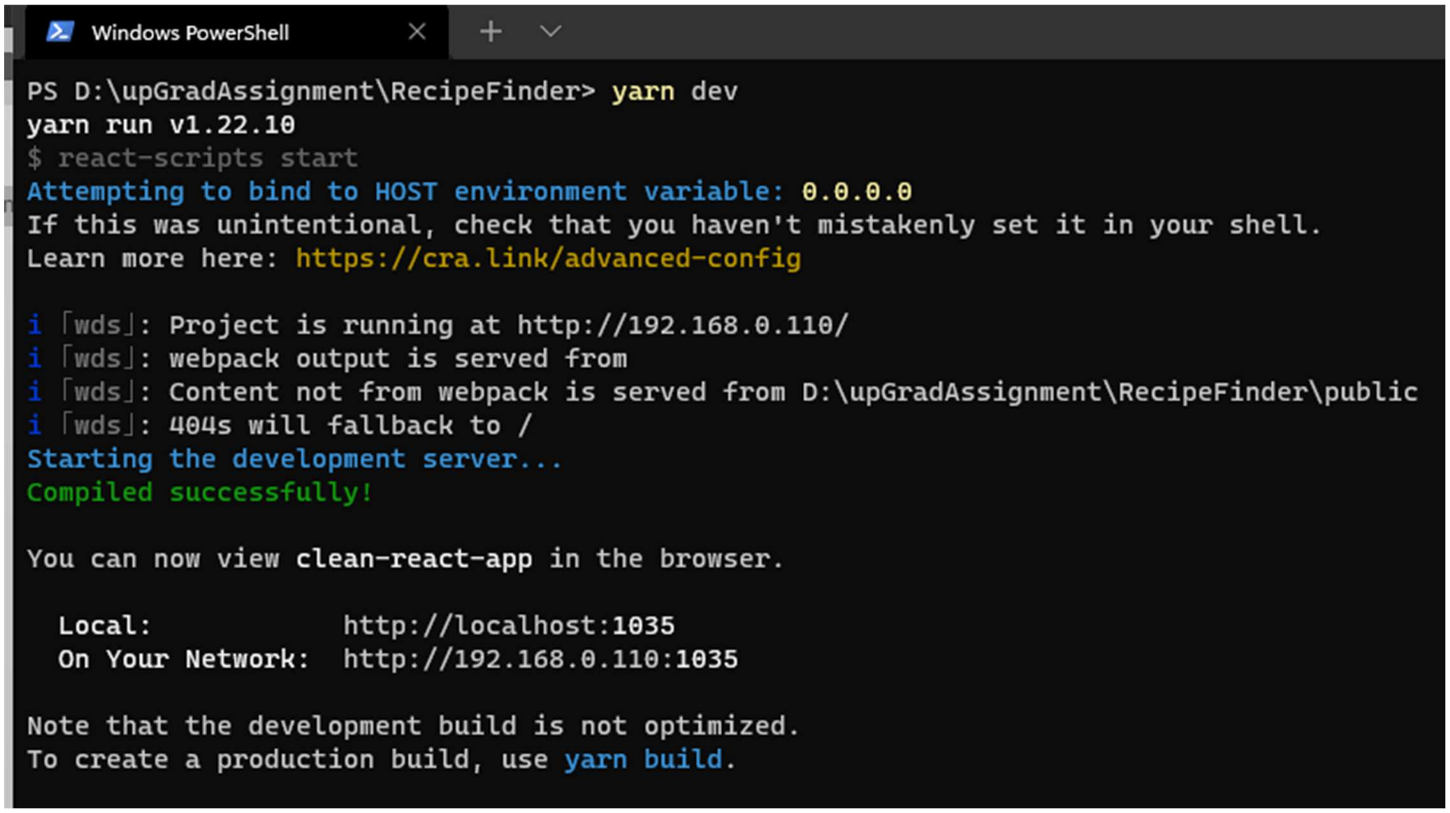


```
"scripts": {
  "dev": "react-scripts start",
  "build": "react-scripts build"
},
```

Have a look at the dependencies, and the build scripts

After you have looked into the supported scripts, you should have decided to serve a development version or a production version.

yarn run dev to create a development server. NOTE: The port of the dev server is reflected in the **dev** server as in
the .env file.

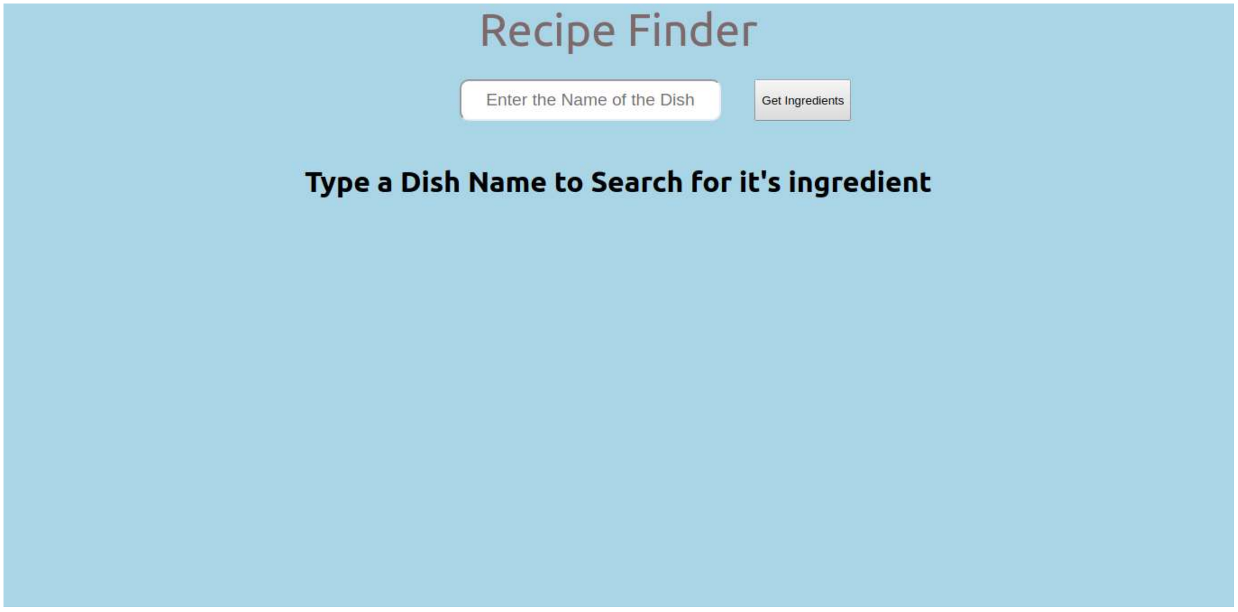


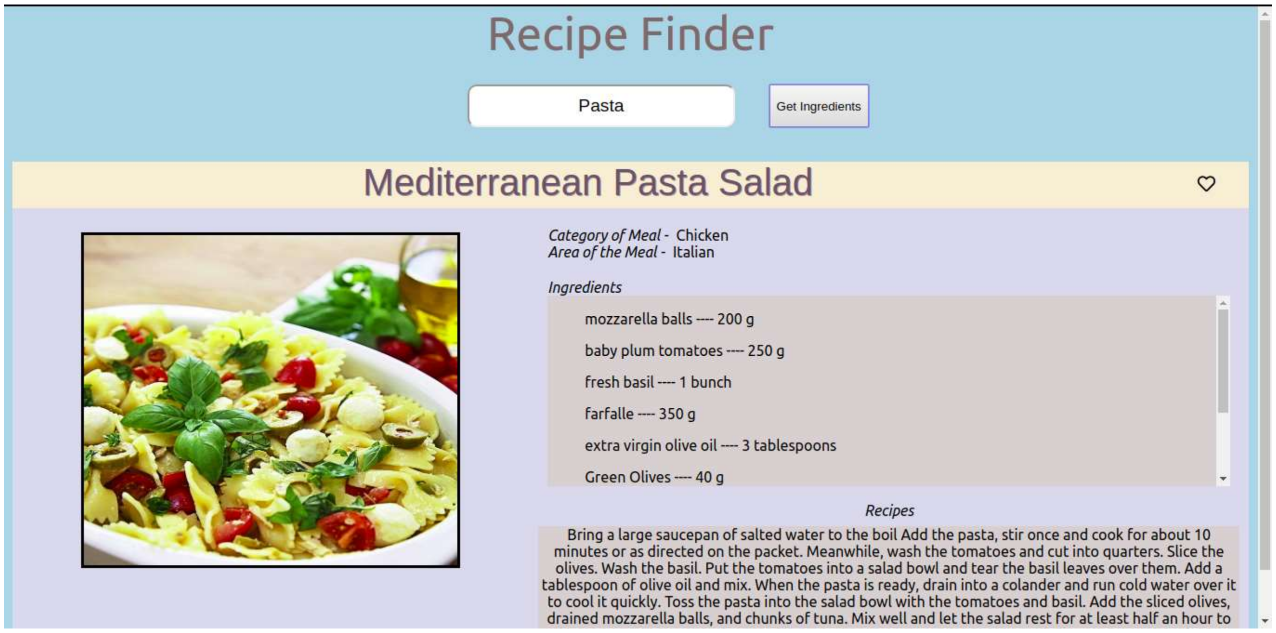
```
PS D:\upGradAssignment\RecipeFinder> yarn dev
yarn run v1.22.10
$ react-scripts start
Attempting to bind to HOST environment variable: 0.0.0.0
If this was unintentional, check that you haven't mistakenly set it in your shell.
Learn more here: https://cra.link/advanced-config

i ｢wds｣: Project is running at http://192.168.0.110/
i ｢wds｣: webpack output is served from
i ｢wds｣: Content not from webpack is served from D:\upGradAssignment\RecipeFinder\public
i ｢wds｣: 404s will fallback to /
Starting the development server...
Compiled successfully!

You can now view clean-react-app in the browser.

  Local:            http://localhost:1035
  On Your Network:  http://192.168.0.110:1035

Note that the development build is not optimized.
To create a production build, use yarn build.
```

yarn build to create a production version of the app. After this command is executed, find the static assets generated into the **build/** folder from which you can serve the static site.

Problem Statement

Goal: You have to create a web page using React, which looks like the image below:


Recipe Finder
Enter the Name of the Dish
Get Ingredients
Type a Dish Name to Search for it's ingredient



Category of Meal - Chicken
Area of the Meal - Italian

Ingredients

mozzarella balls ---- 200 g
baby plum tomatoes ---- 250 g
fresh basil ---- 1 bunch
farfalle ---- 350 g
extra virgin olive oil ---- 3 tablespoons
Green Olives ---- 40 g

Recipes

Bring a large saucepan of salted water to the boil Add the pasta, stir once and cook for about 10 minutes or as directed on the packet. Meanwhile, wash the tomatoes and cut into quarters. Slice the olives. Wash the basil. Put the tomatoes into a salad bowl and tear the basil leaves over them. Add a tablespoon of olive oil and mix. When the pasta is ready, drain into a colander and run cold water over it to cool it quickly. Toss the pasta into the salad bowl with the tomatoes and basil. Add the sliced olives, drained mozzarella balls, and chunks of tuna. Mix well and let the salad rest for at least half an hour to

On this web page, you can enter the name of a dish that you want information on into a search input field. Once you find the required information, you can display it on the web page.

You'll be using an API provided by TheMealDB. You can read more about the API at this link - https://www.themealdb.com/

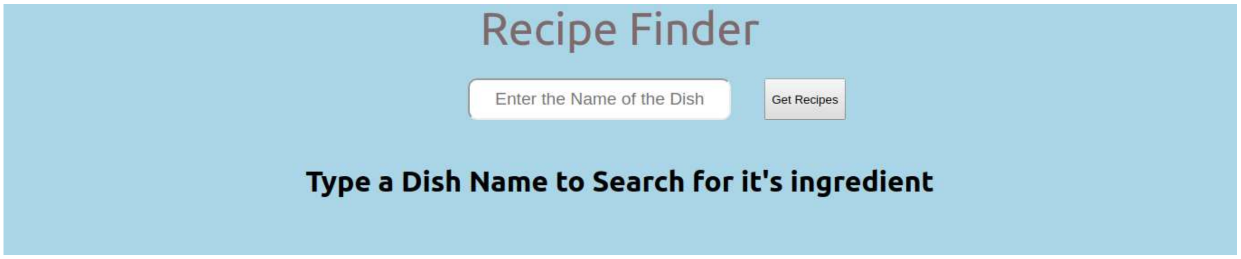
Guidelines

Heading

- The page should have a prominent heading on the lines of "Recipe Finder".
- When the page loads for the first time, there should be a sub-header on the main page saying 'Type a Dish Name to Search for its Ingredients', as shown in the screenshot below.
- This Sub-heading should be below both the heading of the page and the search bar

Search Bar

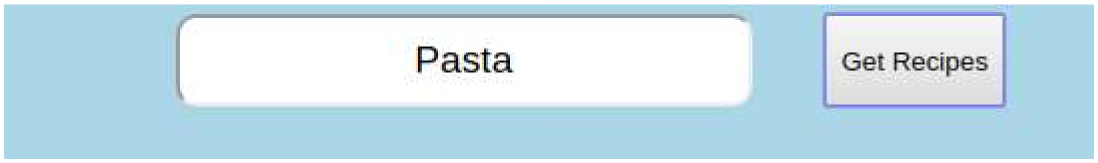
- There should be a search bar at the top of the web page, where the user can enter the name of a dish that he/she wants to search.
- The search bar should contain a placeholder telling the user to type the search query in the input box



Type a Dish Name to Search for it's ingredient

There should be a button to the right of the search bar, by clicking which you can make an API call to TheMealDB.

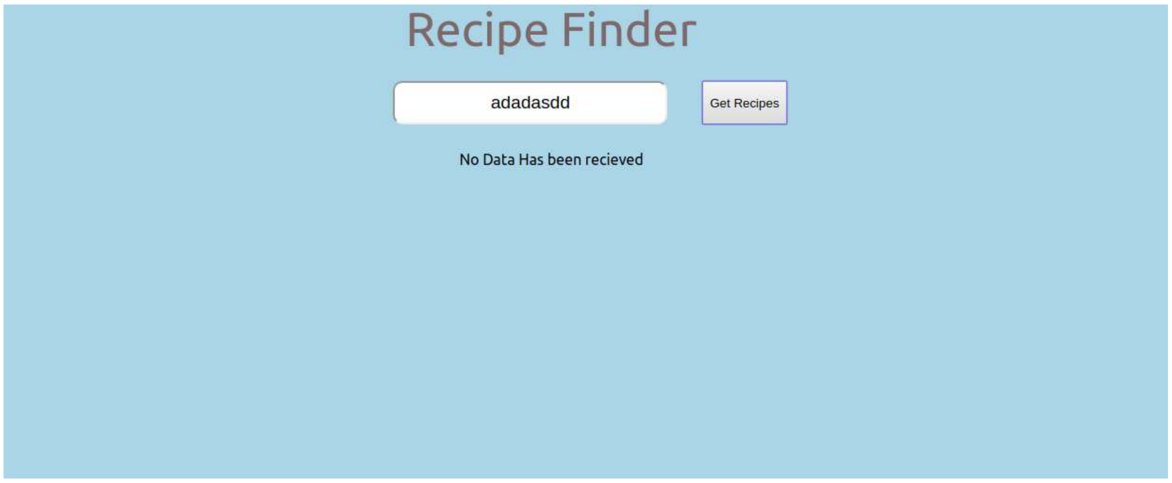
Here's what the search bar and the button should look like:



Error Page

In case the API call fails, you need to show the following error message on the screen: “No Data has been received”.

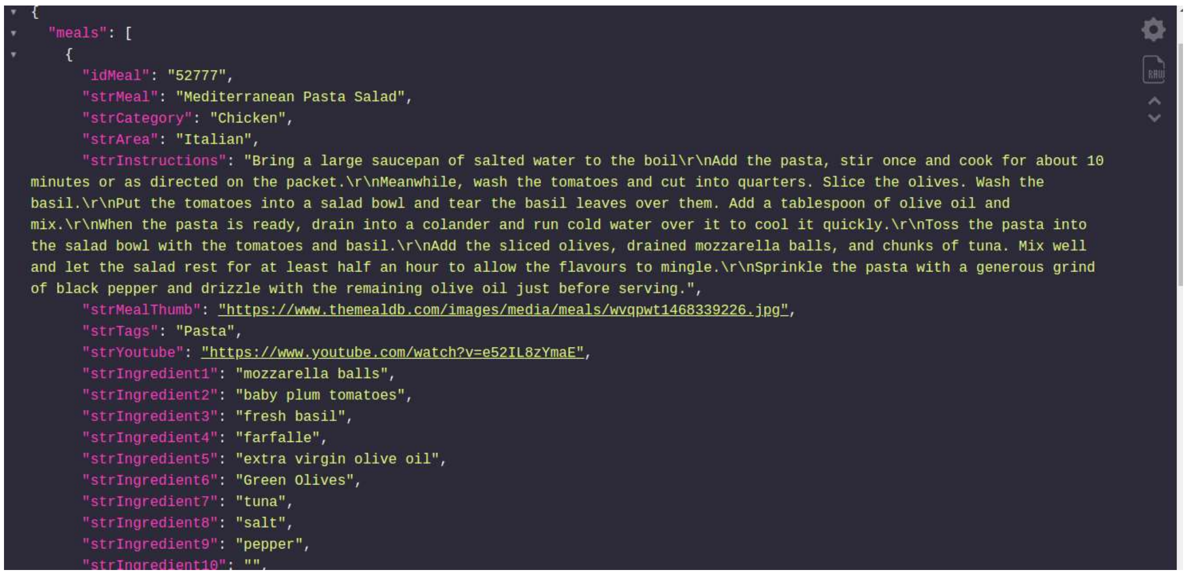
This is what the home screen is supposed to look like if there is an error:



No Data Has been recieved

Search Results

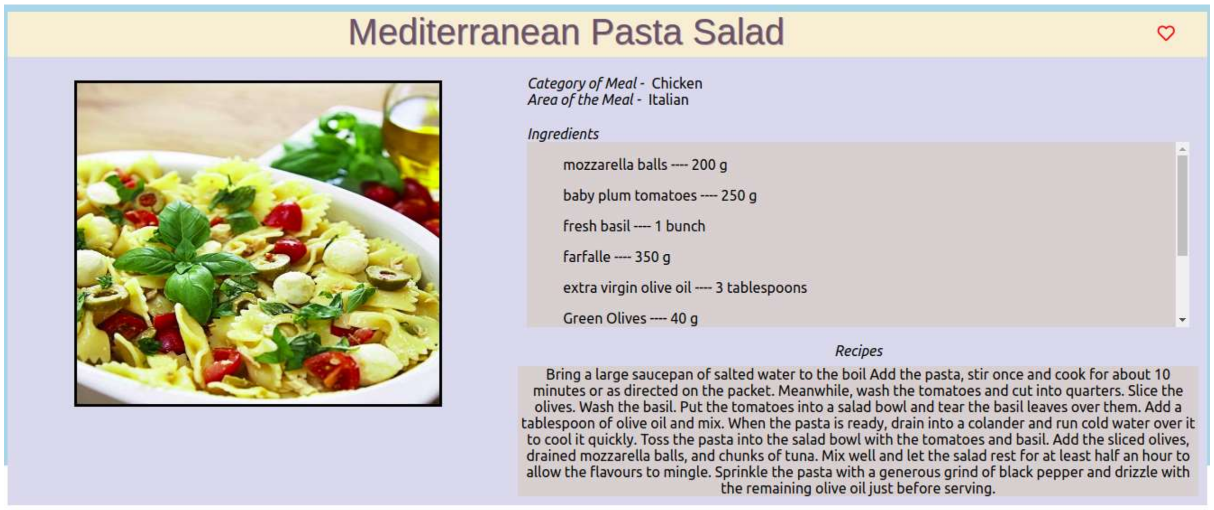
TheMealDB API will return the search results in the form of a JSON response. A sample JSON response would be in this format:



```
{
  "meals": [
    {
      "idMeal": "52777",
      "strMeal": "Mediterranean Pasta Salad",
      "strCategory": "Chicken",
      "strArea": "Italian",
      "strInstructions": "Bring a large saucepan of salted water to the boil\r\nAdd the pasta, stir once and cook for about 10
minutes or as directed on the packet.\r\nMeanwhile, wash the tomatoes and cut into quarters. Slice the olives. Wash the
basil.\r\nPut the tomatoes into a salad bowl and tear the basil leaves over them. Add a tablespoon of olive oil and
mix.\r\nWhen the pasta is ready, drain into a colander and run cold water over it to cool it quickly.\r\nToss the pasta into
the salad bowl with the tomatoes and basil.\r\nAdd the sliced olives, drained mozzarella balls, and chunks of tuna. Mix well
and let the salad rest for at least half an hour to allow the flavours to mingle.\r\nSprinkle the pasta with a generous grind
of black pepper and drizzle with the remaining olive oil just before serving.",
      "strMealThumb": "https://www.themealdb.com/images/media/meals/wvqpwt1468339226.jpg",
      "strTags": "Pasta",
      "strYoutube": "https://www.youtube.com/watch?v=e52IL8zYmaE",
      "strIngredient1": "mozzarella balls",
      "strIngredient2": "baby plum tomatoes",
      "strIngredient3": "fresh basil",
      "strIngredient4": "farfalle",
      "strIngredient5": "extra virgin olive oil",
      "strIngredient6": "Green Olives",
      "strIngredient7": "tuna",
      "strIngredient8": "salt",
      "strIngredient9": "pepper",
      "strIngredient10": "",
```

You will receive an array of meals in a successful response, and you have to show the different meals that you’ve received as a response in panels. Each meal should be given a panel of its own.

The panels should look like this:



Mediterranean Pasta Salad

Category of Meal - Chicken
Area of the Meal - Italian

Ingredients

mozzarella balls ---- 200 g

baby plum tomatoes ---- 250 g

fresh basil ---- 1 bunch

farfalle ---- 350 g

extra virgin olive oil ---- 3 tablespoons

Green Olives ---- 40 g

Recipes

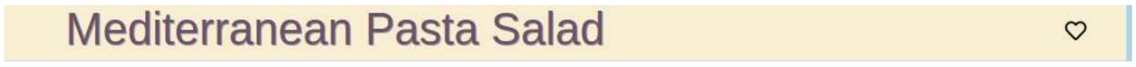
Bring a large saucepan of salted water to the boil Add the pasta, stir once and cook for about 10 minutes or as directed on the packet. Meanwhile, wash the tomatoes and cut into quarters. Slice the olives. Wash the basil. Put the tomatoes into a salad bowl and tear the basil leaves over them. Add a tablespoon of olive oil and mix. When the pasta is ready, drain into a colander and run cold water over it to cool it quickly. Toss the pasta into the salad bowl with the tomatoes and basil. Add the sliced olives, drained mozzarella balls, and chunks of tuna. Mix well and let the salad rest for at least half an hour to allow the flavours to mingle. Sprinkle the pasta with a generous grind of black pepper and drizzle with the remaining olive oil just before serving.

The information contained in the panel includes the following:

Title: The title of the meal should be mentioned in bold, similar to a panel header. There should be a heart emoji beside the title that acts as a 'like' button for a particular recipe/meal. If you like that recipe/meal, you can click on the heart button, which will then turn red. Clicking on the heart button again should 'unlike' the recipe/meal. This should happen for all the meals independently.

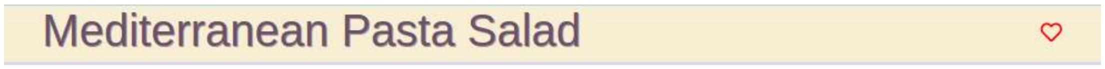
Upon clicking on the title of the meal, the page should be redirected to the source URL of the meal that is provided in the JSON response.

This is what the header would look like when this functionality is implemented:



Mediterranean Pasta Salad

When a particular meal is liked, it should look like this:



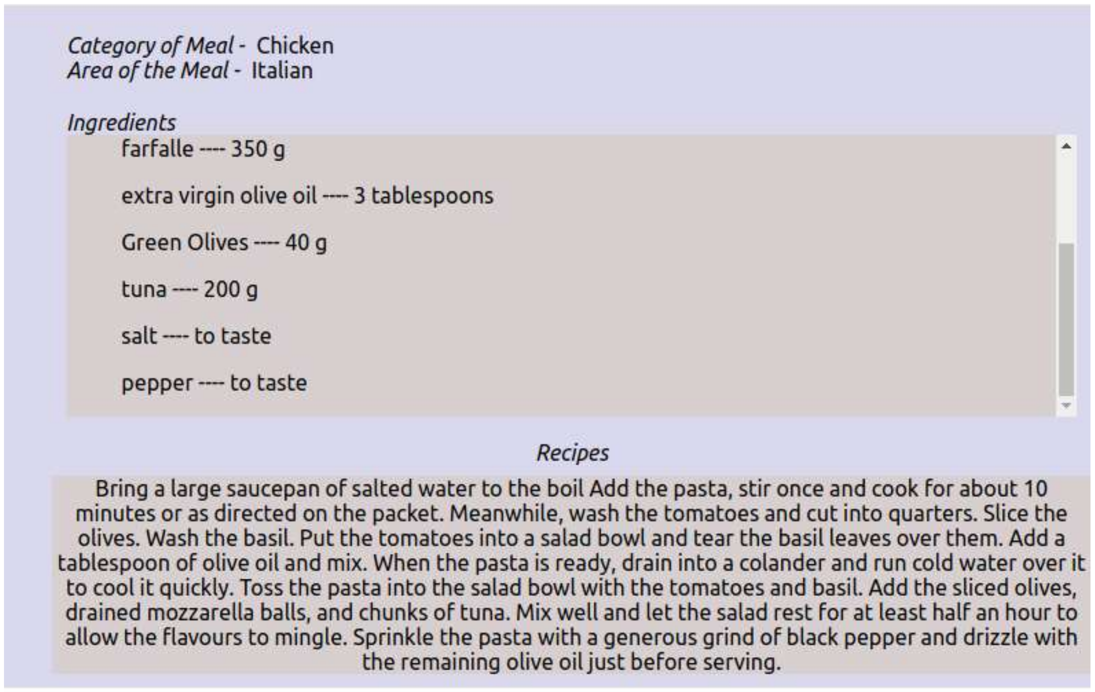
Mediterranean Pasta Salad

The body panel should be comprised of the following information:

- The picture of the meal
- The category of the meal
- The area of the meal
- The ingredients
- The recipe
- This information should be arranged in the following manner:
- The picture of the meal should be displayed prominently to the left.
- On the right-hand side, the category of the meal and the area of the meal should be mentioned.
- The ingredients should be mentioned, with their quantities against them.
- The ingredient panel should have a fixed height (250px). Any extra text should overflow.

Below the list of ingredients, show the recipe. Similar to the ingredient panel, the recipe div should have a fixed height, and the text should overflow.

This is what the right-side panel is supposed to look like:



Category of Meal - Chicken
Area of the Meal - Italian

Ingredients

farfalle ---- 350 g

extra virgin olive oil ---- 3 tablespoons

Green Olives ---- 40 g

tuna ---- 200 g

salt ---- to taste

pepper ---- to taste

Recipes

Bring a large saucepan of salted water to the boil Add the pasta, stir once and cook for about 10 minutes or as directed on the packet. Meanwhile, wash the tomatoes and cut into quarters. Slice the olives. Wash the basil. Put the tomatoes into a salad bowl and tear the basil leaves over them. Add a tablespoon of olive oil and mix. When the pasta is ready, drain into a colander and run cold water over it to cool it quickly. Toss the pasta into the salad bowl with the tomatoes and basil. Add the sliced olives, drained mozzarella balls, and chunks of tuna. Mix well and let the salad rest for at least half an hour to allow the flavours to mingle. Sprinkle the pasta with a generous grind of black pepper and drizzle with the remaining olive oil just before serving.

Evaluation Rubrics

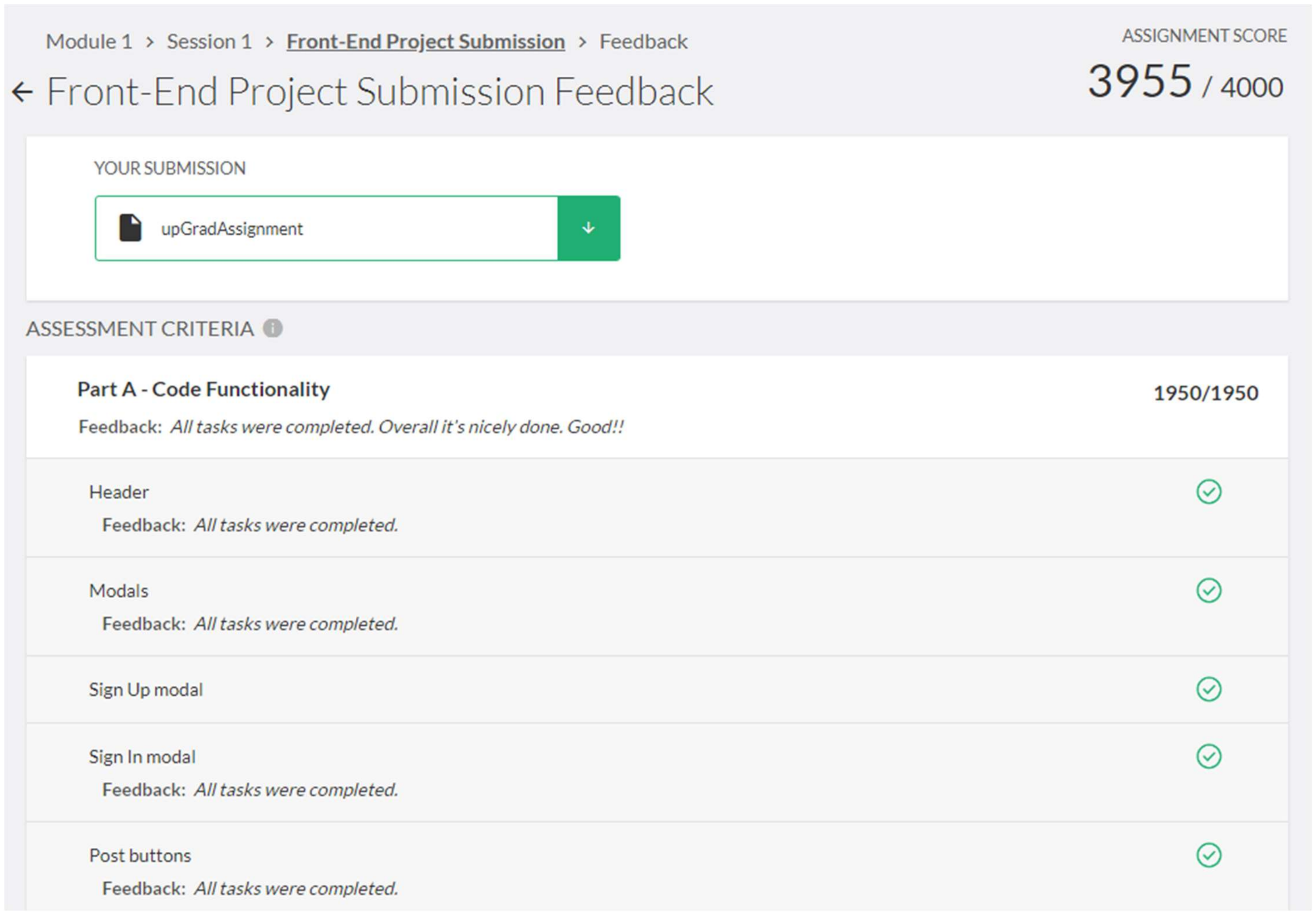
Criteria	**Meets Specifications**	**Does Not Meet Specifications**
Code Functionality (80%)		
Does the code work?	The JavaScript code produces no error.	The code produces compilation errors or run-time errors when executed.
Is the web page CSS styled properly?	The styling of the page should match the styling of the screenshots given in the problem statement. It does not need to be exactly the same, but it should resemble the given structure	The styling of the page does not match the styling of the screenshots given in the problem statement.
Is the project made in React?	React should be used to create the web page. The React project should be hosted on the localhost URL and start upon running 'npm start'.	React has not been used to create the web page. The React project is not hosted on the localhost URL and does not start upon running 'npm start'.
Meal information is displayed correctly on page load	There should be a heading when the page loads, with the message that you should search a particular dish name to know what its recipe is. This message should only show on the page load.	There is no heading when the page loads. The heading mentioning that you have to search a particular dish name to know its recipe is displayed at other times too apart from the page load.

Criteria	Meets Specifications	Does Not Meet Specifications
Does the web page call the correct API?	The web page should call the correct API upon the click of the button “Get Recipes”.	The web page does not call the correct API upon the click of the button “Get Recipes”.
Error handling	If the API call returns an error, then the error message stating “No Data has been received” should be shown.	If the API call returns an error, then no error message should be shown.
API success	If the API call is successful, the data returned by the API is displayed in the web page panels.	If the API call is a success, there is no data in panels. These panels don’t look similar to the panels as in the problem statement.
Meals/recipe panels header	In the panel header, the title should be linked to the URL of the recipe. There should be a heart-shaped button present in the header, which can be clicked to indicate that you like the recipe. Clicking this button again indicates that you have unliked the recipe. This ‘like’ action should happen for each recipe individually. Liking a particular recipe should not indicate that you like some other recipe(s).	In the panel header, the title is not linked to the URL of the recipe. There is no heart-shaped button present in the header to help indicate that you like a recipe. This ‘like’ action is not happening per recipe level. A like on a particular recipe should trigger the like in that recipe only.
Meal/recipe panel body	In the panel body, the image should be shown on the left-hand side. To the right, the information that has been asked in the problem statement should be mentioned. The ingredients should be listed along with their quantities, The height of their div should not expand beyond 250px; if the content is more than that, the div should overflow vertically. The recipe should be present in the panel, and the height of its div should not expand beyond 250px; if the content is more than that, the div should overflow in a vertical scroll.	The panel is not styled in a manner similar to the guidelines given. The ingredients are either not present or are present without their quantities. The height of the div exceeds 250 px, and no vertical scroll appears if there is an overflow. The recipe is not present in the panel The height of the div exceeds 250 px, and no vertical scroll appears if there is an overflow.
Is the code formatted correctly and easy to read?	The code is formatted correctly; it uses the right spacing and indentation and follows the formatting guidelines laid out in the Google HTML/CSS Style	The code is not formatted correctly. Extra spaces, line breaks, incorrect indentations, and bad formatting are used throughout the code. The code

Criteria	Meets Specifications	Does Not Meet Specifications
	Guide. The code contains useful comments that explain how the complicated portions of the code work. HTML/CSS/JS objects, classes, or variables have proper and logical names.	does not contain any comment, or it contains poor comments that do not explain properly how the complicated portions of the code work. HTML/CSS/JS objects, classes, or variables do not have proper or logical names.
Does the student use Git and GitHub to conduct version control on his/her code?	The student uses Git and GitHub to conduct version control on the assignment code. The student makes small, incremental commits. The student writes clear and concise commit messages.	The student does not use Git or GitHub to conduct version control on the assignment code. The student makes big commits that contain multiple features or bug fixes. The student writes short or unclear commit messages.

Grades

Assignment 1



Module 1 > Session 1 > Front-End Project Submission > Feedback

ASSIGNMENT SCORE

3955 / 4000

Front-End Project Submission Feedback

YOUR SUBMISSION

upGradAssignment

ASSESSMENT CRITERIA

Part A - Code Functionality **1950/1950**

Feedback: *All tasks were completed. Overall it's nicely done. Good!!*

Header ✓

Feedback: *All tasks were completed.*

Modals ✓

Feedback: *All tasks were completed.*

Sign Up modal ✓

Sign In modal ✓

Feedback: *All tasks were completed.*

Post buttons ✓

Feedback: *All tasks were completed.*

All Posts button ✓

Feedback: *All tasks were completed.*

Create Post button ✓

Feedback: *All tasks were completed.*

Part A - Code Readability & Guidelines — 25/50

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

Was this helpful to you?

Code Segregation ◐

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

Was this helpful to you?

Part B - Code Functionality — 980/980

Header ✓

Feedback: *All tasks were completed.*

Blog list ✓

Feedback: *All tasks were completed.*

Part B - Code Readability & Guidelines

10/20

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

Code Segregation ◐

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

Was this helpful to you?

Part C - Code Functionality

980/980

Header ✓

Feedback: *All tasks were completed.*

Positioning of blog elements ✓

Feedback: *All tasks were completed.*

Creating Edit button and adding edit functionality ✓

Feedback: *All tasks were completed.*

Creating the "Like" button ✓

Feedback: *All tasks were completed.*

Comment button and comment box ✓

Feedback: *All tasks were completed.*

Part C - Code Functionality **980/980**

Header ✓

Feedback: *All tasks were completed.*

Positioning of blog elements ✓

Feedback: *All tasks were completed.*

Creating Edit button and adding edit functionality ✓

Feedback: *All tasks were completed.*

Creating the "Like" button ✓

Feedback: *All tasks were completed.*

Comment button and comment box ✓

Feedback: *All tasks were completed.*

Part C - Code Readability & Guidelines **10/20**

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

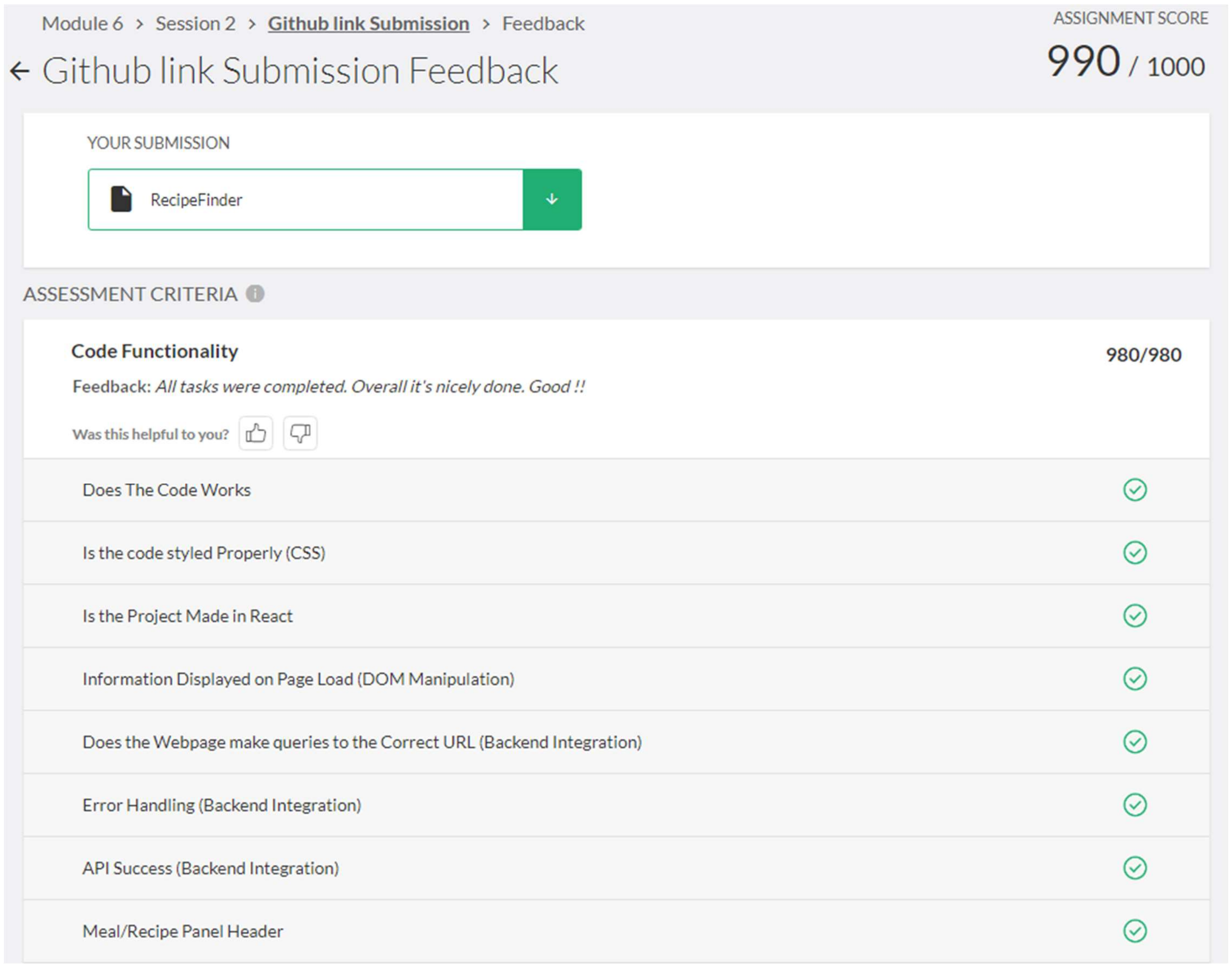
Was this helpful to you?

Code Segregation ◐

Feedback: *The HTML, CSS and Javascript portion of the code must be kept strictly in separate files, and under their respective html, css and js folders.*

Was this helpful to you?

Assignment 2



Module 6 > Session 2 > Github link Submission > Feedback

ASSIGNMENT SCORE

990 / 1000

Github link Submission Feedback

YOUR SUBMISSION

RecipeFinder

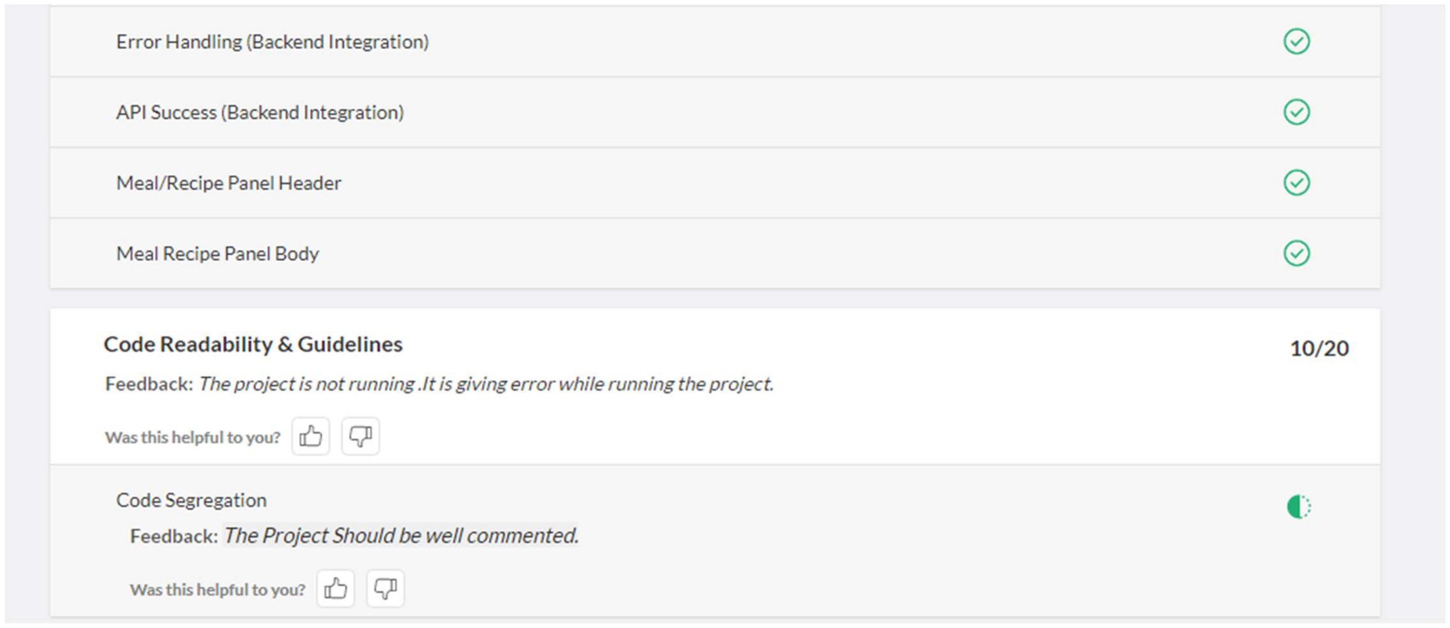
ASSESSMENT CRITERIA

Code Functionality 980/980

Feedback: *All tasks were completed. Overall it's nicely done. Good !!*

Was this helpful to you?

Does The Code Works	✓
Is the code styled Properly (CSS)	✓
Is the Project Made in React	✓
Information Displayed on Page Load (DOM Manipulation)	✓
Does the Webpage make queries to the Correct URL (Backend Integration)	✓
Error Handling (Backend Integration)	✓
API Success (Backend Integration)	✓
Meal/Recipe Panel Header	✓



Error Handling (Backend Integration)	✓
API Success (Backend Integration)	✓
Meal/Recipe Panel Header	✓
Meal Recipe Panel Body	✓

Code Readability & Guidelines 10/20

Feedback: *The project is not running .It is giving error while running the project.*

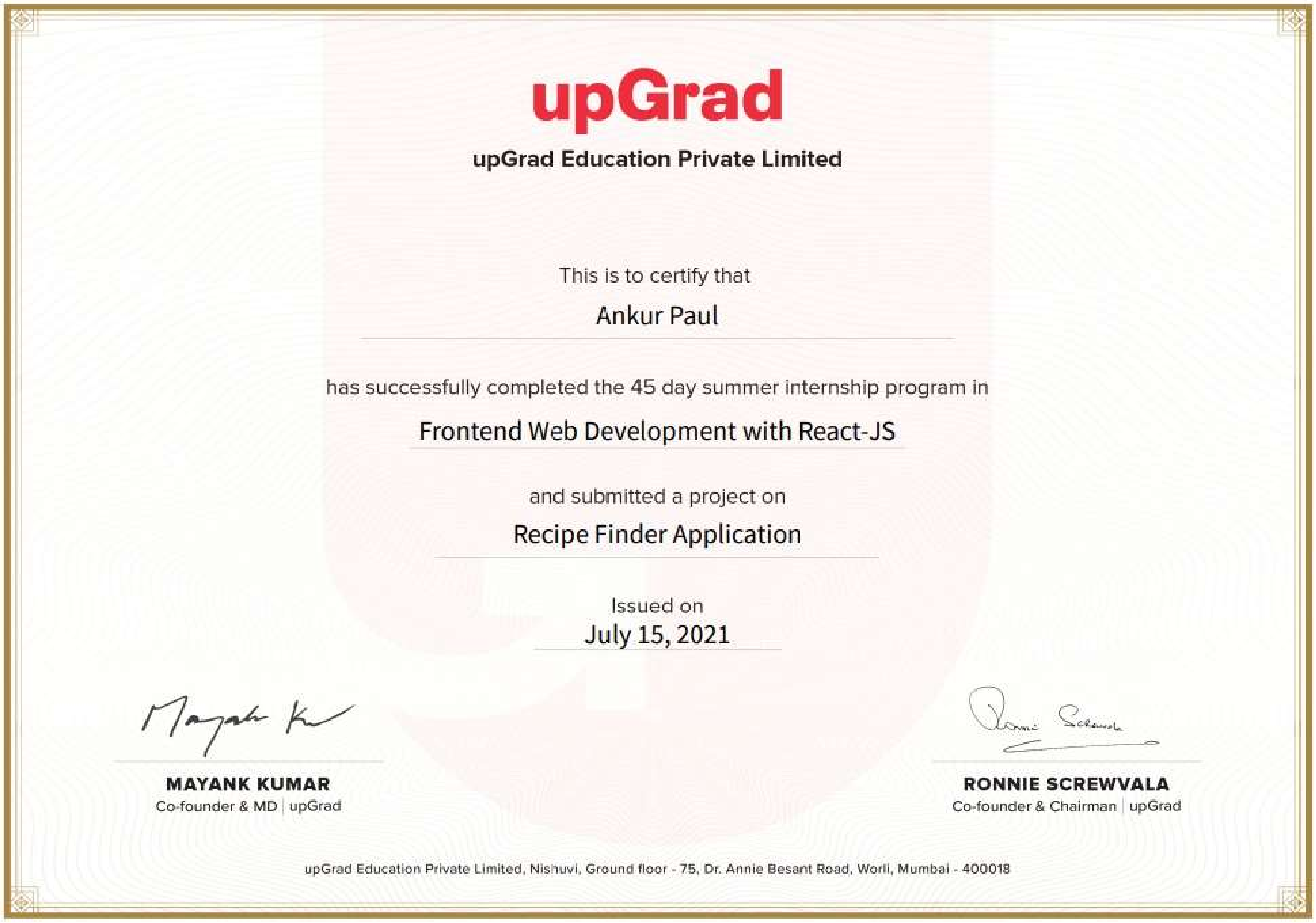
Was this helpful to you?

Code Segregation ◐

Feedback: *The Project Should be well commented.*

Was this helpful to you?

Certification



upGrad

upGrad Education Private Limited

This is to certify that

Ankur Paul

has successfully completed the 45 day summer internship program in

Frontend Web Development with React-JS

and submitted a project on

Recipe Finder Application

Issued on

July 15, 2021

[illegible]

MAYANK KUMAR

Co-founder & MD | upGrad

[illegible]

RONNIE SCREWVALA

Co-founder & Chairman | upGrad

upGrad Education Private Limited, Nishuvi, Ground floor - 75, Dr. Annie Besant Road, Worli, Mumbai - 400018