PRACTICAL PORTFOLIO

SIT 120 Assignment 2

Practical Portfolio

Details

Student ID: *221287236*

Student Name: *Nandinee Pardasani*

Name Of The Project: *Practical Portfolio*

WEEK 1 | HTML, CSS & JavaScript

Reflection

I found week 1 content quite easy to understand as I have had previous knowledge of HTML, CSS & JAVASCRIPT.

What is "Responsive Web Application Design"?

Responsive web design (RWD) is a web development approach that creates dynamic changes to the appearance of a website, depending on the screen size and orientation of the device being used to view it. Responsive design relies on proportion-based grids to rearrange content and design elements.

Task 1 | Create an HTML Page

Reflection

I found this task quite enjoyable and exciting as it allowed me to explore my creativity as well as challenge my skills.

Notes

What is HTML

HTML is short for Hypertext Markup Language. It's a file format used for containing HTML language which constructs a web page. This file format is based on markup codes intended to be used in Web Browsers. HTML can contain formatted text, images, and other elements in a way that can be translated with web browsers.

More Facts About HTML

- *HTML tells browsers which part of a webpage is a header, which is a footer, where paragraphs belong, where images, graphics, and videos are placed, etc.*
- *Browsers take that HTML content and translate it into what you see on your device's screen.*
- *HTML is an industry standard language that's guaranteed to be understood by all web browsers (applications like Safari, Firefox, and Google Chrome),*
- *HTML is also the universally accepted standard for making websites "findable" by search engineske Google, Yahoo, and Bing based on relevant search terms (i.e.: the search words you type into the search bar).*

HTML Page Structure

These are some very basic HTML tags that are included in pretty every HTML page

- *The <!DOCTYPE html> declaration defines that this document is an HTML5 document*
- *The <html> element is the root element of an HTML page*
- *The <head> element contains meta information about the HTML page*
- *The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)*
- *The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.*
- *The <h1> element defines a large heading*
- *The <p> element defines a paragraph*

This is a visualization of an HTML Page Structure

```
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Task 2 | Add CSS to your HTML page

Reflection

Adding CSS to my page just gave it an extra element which made it more appealing to my audience.

Notes

What is CSS?

Like HTML, CSS is not a programming language. It's not a markup language either. CSS is a style sheet language. CSS is what you use

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to selectively style HTML elements

HTML Elements

The HTML element is everything from the start tag to the end tag:

<tagname>Content goes here...</tagname>

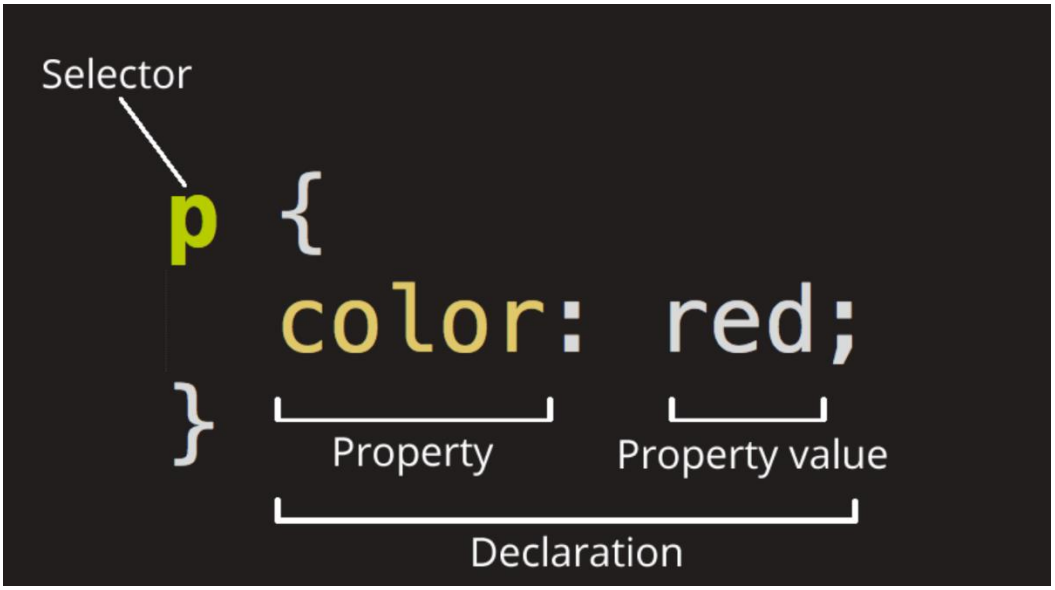
Examples of some HTML elements:

<h1>My First Heading</h1>

* My first paragraph. *

Anatomy of CSS

Example Red Paragraph



Selector

This is the HTML element name at the start of the ruleset. It defines the element(s) to be styled (in this example, `<p>` elements). To style a different element, change the selector.

Declaration

*This is a single rule like `color: red;`. It specifies which of the element's **properties** you want to style.*

Properties

These are ways in which you can style an HTML element. (In this example, `color` is a property of the `<p>` elements.) In CSS, you choose which properties you want to affect in the rule.

Property value

*To the right of the property—after the colon—there is the **property value**. This chooses one out of many possible appearances for a given property. (For example, there are many `color` values in addition to `red`.)*

Note the other important parts of the syntax:

- *Apart from the selector, each ruleset must be wrapped in curly braces. ({})*
- *Within each declaration, you must use a colon (:) to separate the property from its value or values.*
- *Within each ruleset, you must use a semicolon (;) to separate each declaration from the next one.*

Types of CSS

Inline CSS

Inline CSS contains the CSS property in the body section attached with element is known as inline CSS. This kind of style is specified within an HTML tag using the style attribute.

Internal CSS

This can be used when a single HTML document must be styled uniquely. The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file.

External CSS

External CSS contains separate CSS file which contains only style property with the help of tag attributes (For example class, id, heading, … etc). CSS property written in a separate file with .css extension and should be linked to the HTML document using **link** *tag. This means that for each element, style can be set only once and that will be applied across web pages.*

Properties of CSS

Inline CSS has the highest priority, then comes Internal/Embedded followed by External CSS which has the least priority. Multiple style sheets can be defined on one page. If for an HTML tag, styles are defined in multiple style sheets then the below order will be followed

- *As Inline has the highest priority, any styles that are defined in the internal and external style sheets are overridden by Inline styles.*
- *Internal or Embedded stands second in the priority list and overrides the styles in the external style sheet.*
- *External style sheets have the least priority. If there are no styles defined either in inline or internal style sheet then external style sheet rules are applied for the HTML tags.*

Task 3 | Adding JavaScript

Reflection

Adding JavaScript is what makes my page more interactive and appealing to my audience.

Notes

What is JavaScript?

JavaScript ("JS" for short) is a full-fledged dynamic programming language that can add interactivity to a website. It was invented by Brendan Eich (co-founder of the Mozilla project, the Mozilla Foundation, and the Mozilla Corporation).

JavaScript itself is relatively compact, yet very flexible. Developers have written a variety of tools on top of the core JavaScript language, unlocking a vast amount of functionality with minimum effort.

These Include

- *Browser Application Programming Interfaces (APIs) built into web browsers, providing functionality such as dynamically creating HTML and setting CSS styles; collecting and manipulating a video stream from a user's webcam, or generating 3D graphics and audio samples.*
- *Third-party APIs that allow developers to incorporate functionality in sites from other content providers, such as Twitter or Facebook.*
- *Third-party frameworks and libraries that you can apply to HTML to accelerate the work of building sites and applications.*

What is the main use of JavaScript?

JavaScript is commonly used for creating web pages. It allows us to add dynamic behavior to the webpage and add special effects to the webpage. On websites, it is mainly used for validation purposes. JavaScript helps us to execute complex actions and also enables the interaction of websites with visitors.

Task 4 | Vue.js Framework

Reflection

This task was slightly challenging for me as it was totally new content and something I haven't used before. However, since the implementation was mainly based upon JavaScript it made it a slightly bit easier to understand.

Notes

What is Vue.js?

Created by Evan You, Vue.js is an open-source progressive JavaScript framework for building user interfaces (UIs) and single-page applications; it is commonly referred to as Vue. This framework uses "high decoupling", allowing developers to progressively create user interfaces (UIs).

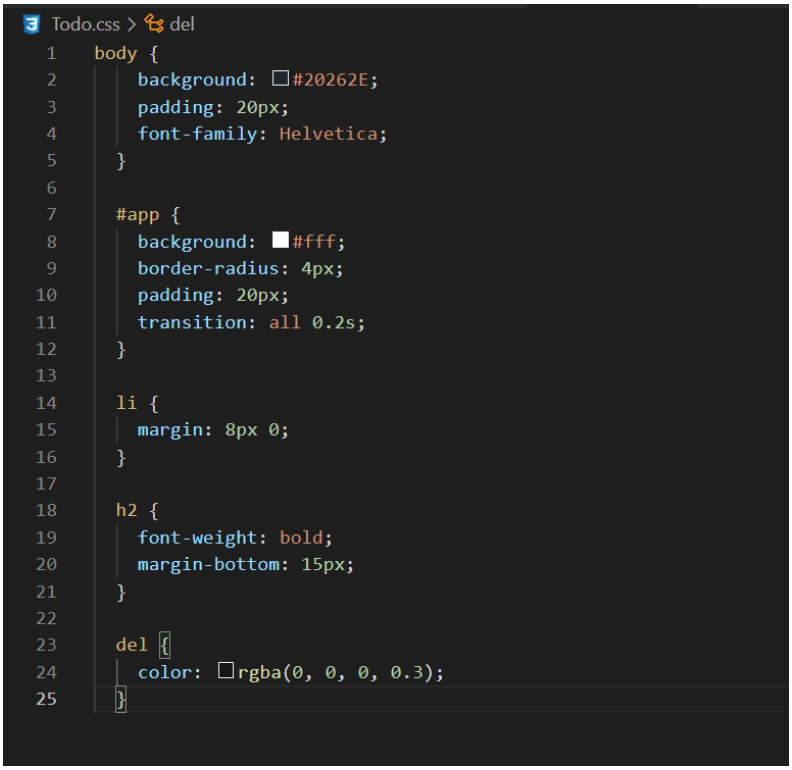
Screenshot Task 4

Input HTML

Todo.html > html > body > div#app > ol > li > label > del

```
<!DOCTYPE html>
<html>

<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="Todo.css">

</head>

<body>

    <div id="app">
        <h2>Todo's For Today :</h2>
        <ol>
          <li v-for="todo in todos">
            <label>
              <input type="checkbox"
                v-on:change="toggle(todo)"
                v-bind:checked="todo.done">

              <del v-if="todo.done">
                {{ todo.text }}
              </del>
              <span v-else>
                {{ todo.text }}
              </span>
            </label>
          </li>
        </ol>
      </div>

      <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js%22%3E"></script>
      <script src="Todo.js"></script>
</body>
</html>
```

Input CSS



Input JS

Output

Todo's For Today :

- [ ] Watch SIT120 Lecture 3
- [ ] Attend SIT120 Practical Class
- [x] ~~Finish Reflection For Task 2~~
- [x] ~~Buy Groceries~~

Todo's For Today :

- [ ] Watch SIT120 Lecture 3
- [x] ~~Attend SIT120 Practical Class~~
- [ ] Finish Reflection For Task 2
- [x] ~~Buy Groceries~~

Screenshot Task 1-2-3

Input HTML

```html
<!DOCTYPE html>
<html>

<head>
  <link rel="stylesheet" href="Week_1.css">
</head>

<body>

  <h1 style="text-align:center;">Welcome To This Amazing Website</h1>

  <h3><i>Why Is This Website Amazing</i>
  </h3>

  <h4><i>Here Are 4 Reasons Why </i>
  </h4>

  <ol>
    <li> It's Authentic </li>
    <li>It's Colourful</li>
    <li>It Contains An Image </li>
  </ol>

  <h4><b> Finally The Last Reason Why This Websiste is amazing is because </b>
  </h4>
  <h6><i> ...Read The Image Below & While You At It Click On The Word Amazing </i>
  </h6>


  <img src="https://t3.ftcdn.net/jpg/02/44/00/76/360_F_244007637_3vUG5MWUNLKiDfbxxmG7pYbpcu5BIuNC.jpg" alt="Workplace"
    usemap="#workmap" width="350" height="350">

  <map name="workmap">
    <area shape="rect" coords="34,44,270,350" alt="Amazing"
      href="https://tinybuddha.com/blog/40-reasons-youre-amazing-and-worth-appreciating/">
  </map>



  <h3><i>Do You Want To Know Why Else This Website Amazing Click The Button Below To Find Out</i>
  </h3>

  <button onclick="myFunction()">Click Me</button>

  <p id="demo"></p>



    <script src=" Week_1.js"></script>

  <button type="button" onclick="document.getElementById('demo').innerHTML = Date()">
    Click me to display Date and Time.</button>

  <p id="demo"></p>

  <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js%22%3E"></script>
  <script src="Week_1.js"></script>

</body>

</html>
```

Input CSS

```css
body {
    background: url(https://media.istockphoto.com/photos/abstract-soft-red-background-picture-id939609996?b=1&k=20&m=939609996&s=170667a&w=0&h=jZ7MHq0XyH9_jbvmnbWYYsskkicOpTX3f84CAHDODm0=);
    background-repeat: no-repeat;
    background-size: cover;
  }

  h1 {
    color: white;
  }
```

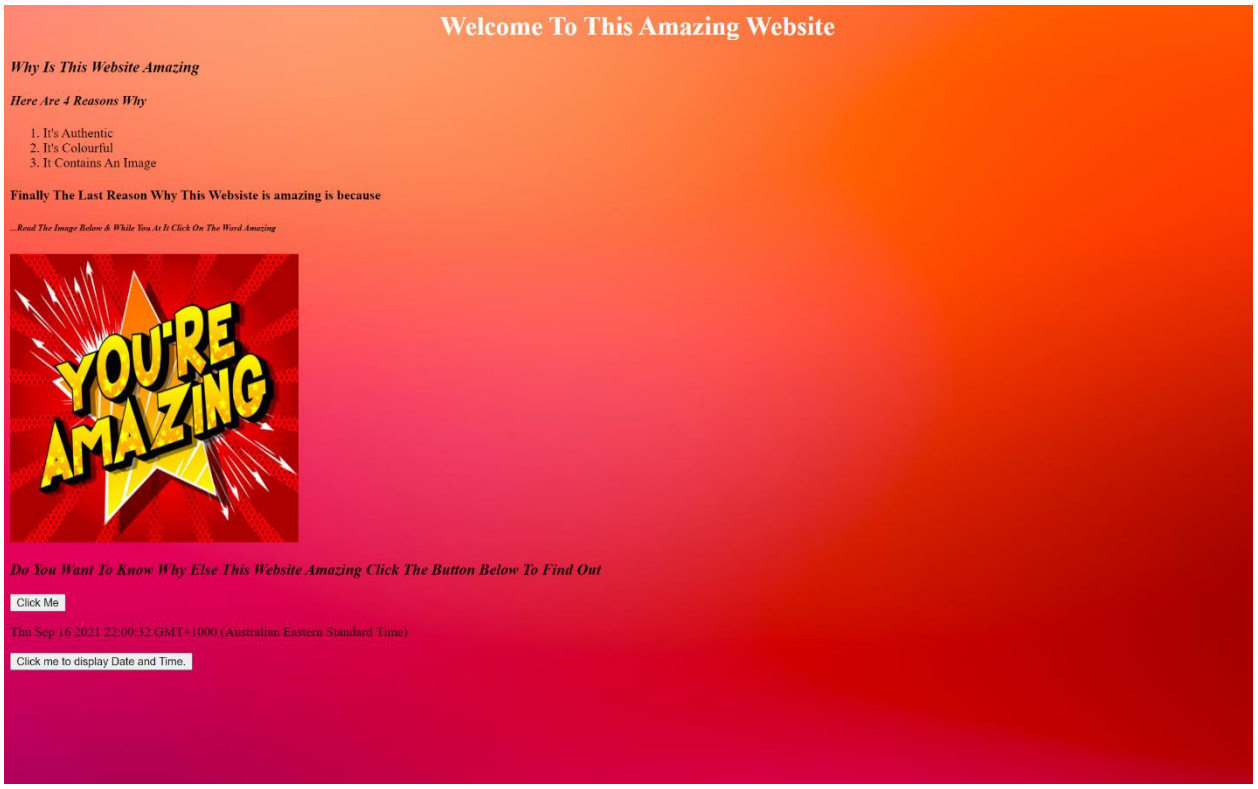
Input JS

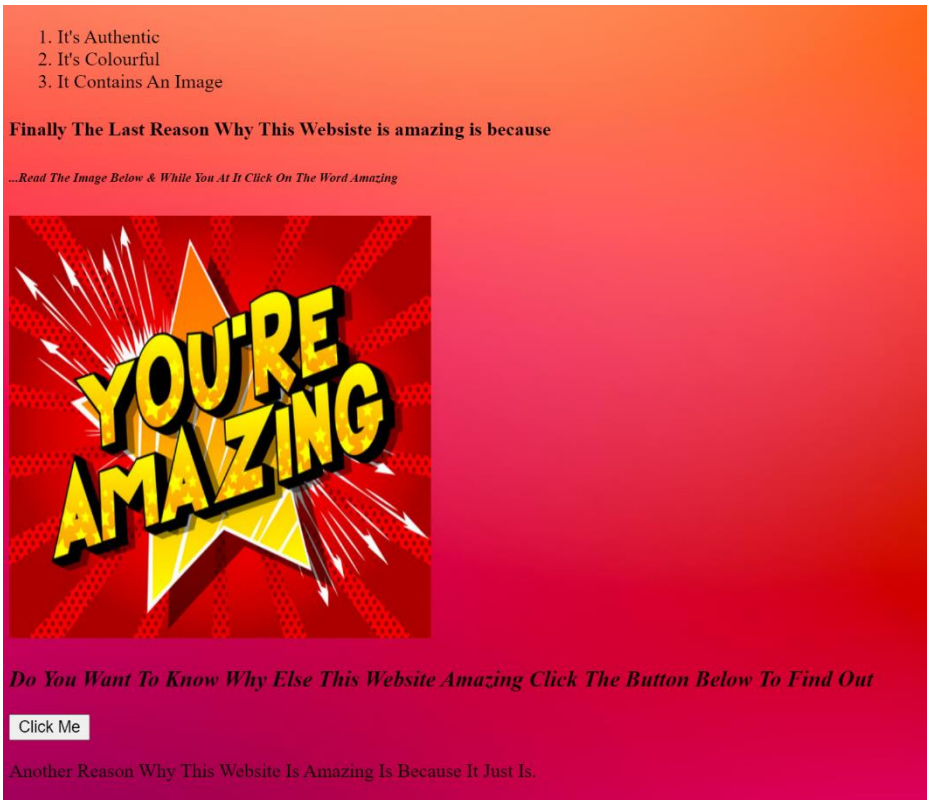
```javascript
function myFunction() {
    document.getElementById("demo").innerHTML = "Another Reason Why This Website Is Amazing Is Because It Just Is. ";
  }

```

GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

JSFiddle link: https://jsfiddle.net/Nandinee/pebdL3xs/2/show

Output


Welcome To This Amazing Website
Why Is This Website Amazing
Here Are 4 Reasons Why
1. It's Authentic
2. It's Colourful
3. It Contains An Image
Finally The Last Reason Why This Websiste is amazing is because
...Read The Image Below & While You At It Click On The Word Amazing
YOU'RE AMAZING
Do You Want To Know Why Else This Website Amazing Click The Button Below To Find Out
Click Me
Thu Sep 16 2021 22:00:32 GMT+1000 (Australian Eastern Standard Time)
Click me to display Date and Time.


1. It's Authentic
2. It's Colourful
3. It Contains An Image
Finally The Last Reason Why This Websiste is amazing is because
...Read The Image Below & While You At It Click On The Word Amazing
YOU'RE AMAZING
Do You Want To Know Why Else This Website Amazing Click The Button Below To Find Out
Click Me
Another Reason Why This Website Is Amazing Is Because It Just Is.

WEEK 2 | Responsive Web Apps Design | Advanced – HTML CCS & JavaScript

Reflection:

This week we learned about responsive web app designs. This concept was very interesting to learn about because before this I hadn't paid attention to how each components location change depending on the device, you're on.

Task 1 | Understand About Responsive Web Pages & Apps Responsiveness Analysis

Reflection:

This task really allowed me to explore and understand the responsive components within a website. It gave me an insight on how responsive web pages function on each device. Which then helped me to better implement it on my website.

Notes

What is Responsive web design (RWD)

*Responsive web design (RWD) is a web development approach that creates **dynamic changes** to the appearance of a website, **depending on the screen size and orientation** of the device being used to view it. RWD is one approach to the problem of designing for the multitude of devices available to customers, ranging from tiny phones to huge desktop monitors.*

Why Do We Need A Responsive Web Design

- *With responsive web design, your website becomes a fluid display of information that reorganizes and rebuilds itself according to the needs of the individual.*
- *This solution allows your message to be portrayed in the most effective and efficient way possible which creates a customized and intuitive user experience and improving performance across all user-types*

How do we make a responsive web design

We can use HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices (desktops, tablets, and phones)

1. Set The Viewport

Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50 cm wide, 1vw is 0.5 cm

Pages optimized for a variety of devices must include a meta viewport tag in the head of the document. A meta viewport tag gives the browser instructions on how to control the page's dimensions and scaling.

Using the meta viewport value `width=device-width` *instructs the page to match the screen's width in device-independent pixels. A device (or density) independent pixel being a representation of a single pixel, which may on a high density screen consist of many physical pixels. This allows the page to reflow content to match different screen sizes, whether rendered on a small mobile phone or a large desktop monitor.*

2. Graphics

We need to make sure that all the graphics are also responsive within the web page because ***An image has fixed dimensions and if it is larger than the viewport will cause a scrollbar. A common way to deal with this problem is to give all images a*** `max-width` ***of*** `100%`.

Responsive Images

This will cause the image to shrink to fit the space it has, should the viewport size be smaller than the image.

Example

Using the max-width Property

If the max-width property is set to 100%, the image will scale down if it has to, but never scale up to be larger than its original size:

```
<img src="img_girl.jpg" style="max-width:100%;height:auto;">
```

Responsive Text Size

The text size can be set with a "vw" unit, which means the "viewport width".

That way the text size will follow the size of the browser window:

```
<h1 style="font-size:10vw">Hello World</h1>
```

3. Media Queries

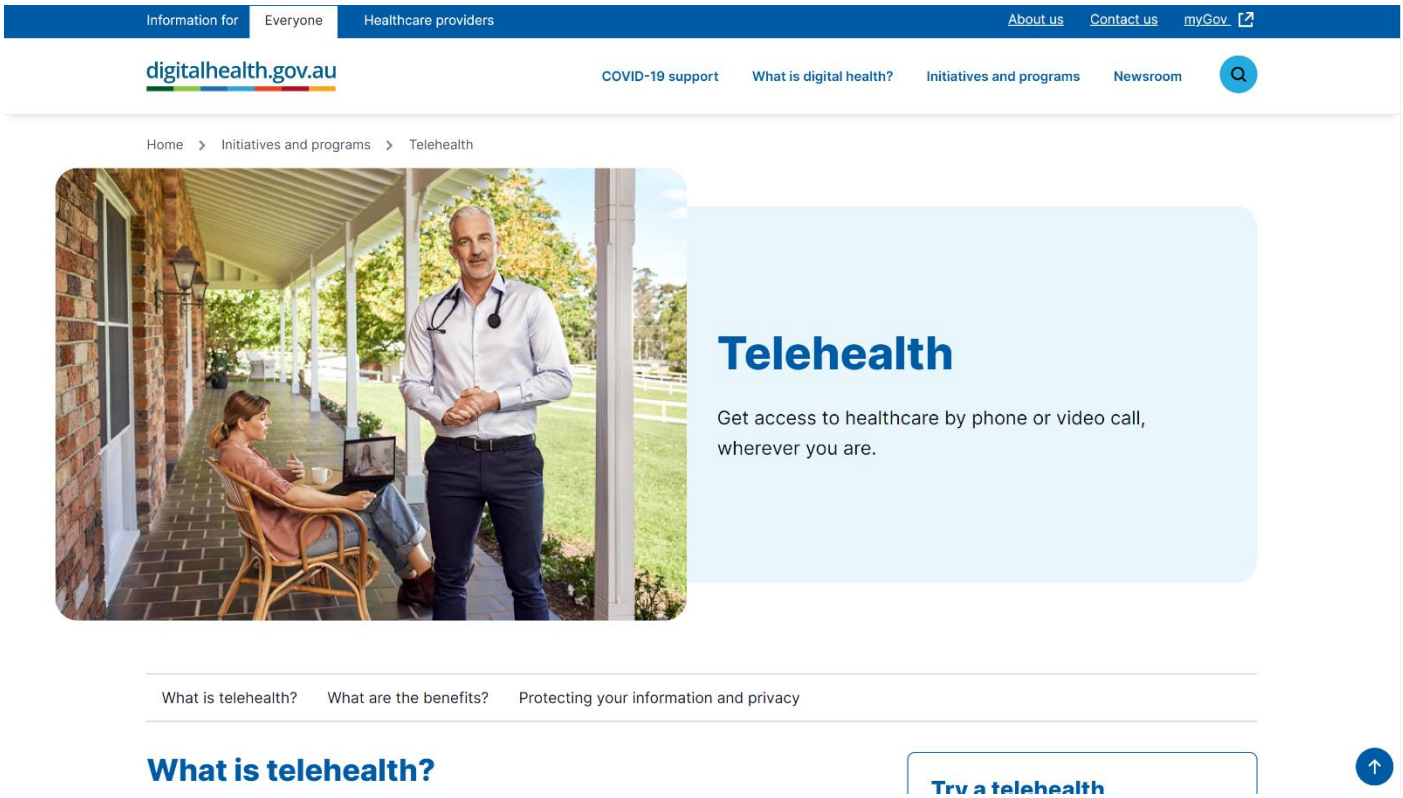
Media queries consists of a media type and zero or more expressions that check for the conditions of particular media features.

By using media queries, presentations can be tailored to a specific range of output devices without changing the content itself.

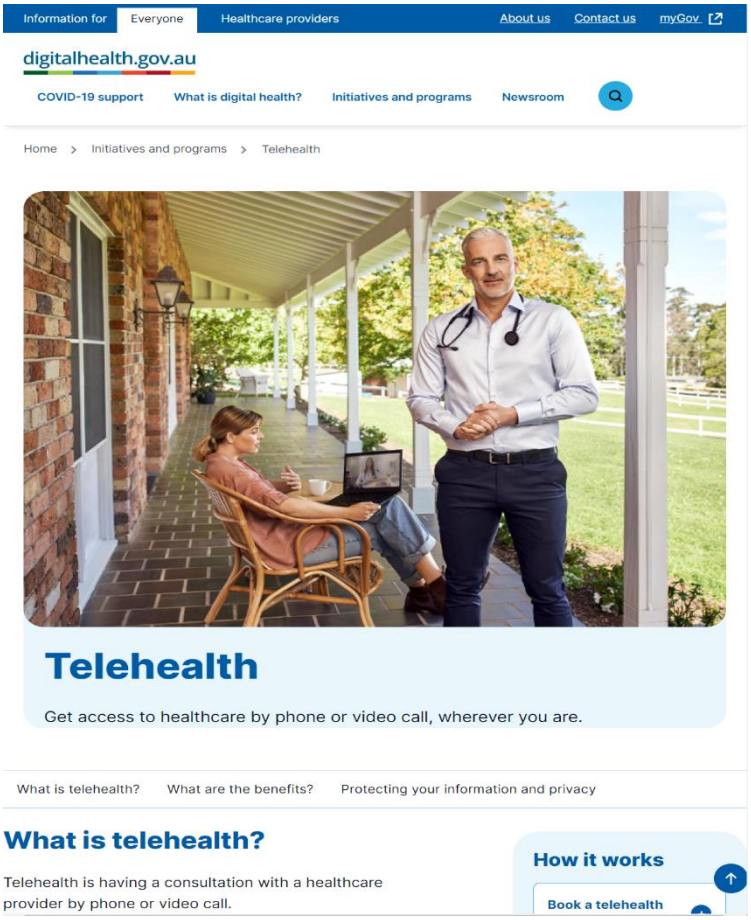
In other words, media queries allow your website to look good on all kinds of displays, from smartphones to big screens.

Screenshots

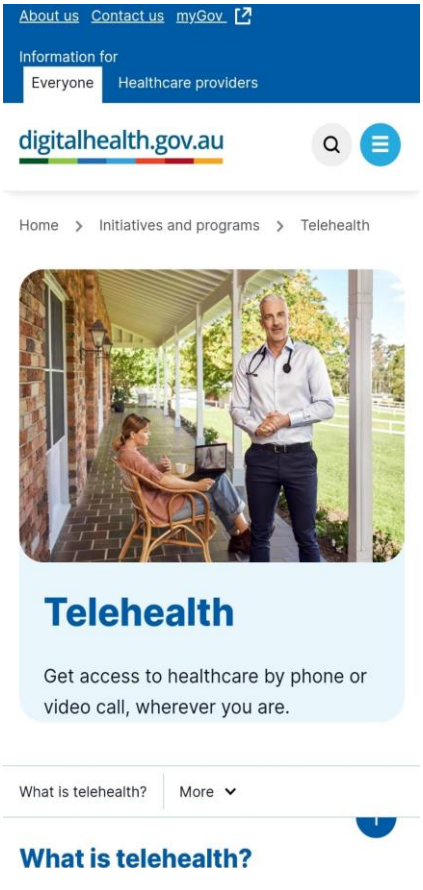
Desktop View



Tablet View



Phone View



Explanation

As we can see that the desktop version has a better layout then the tablet and phone version. For example, the navigation bar is more spread out on the desktop compared to the tablet and phone. The image is shown clearly in all versions however the position of the word "Telehealth" is different in all versions, larger and positioned on the side (Desktop View) smaller and positioned on the bottom (Tablet & Phone view).

Finally, we see that in all cases the navigation bar hasn't been compromised as all the links are present in some way or the other.

Task 2 | Add CSS to your HTML page

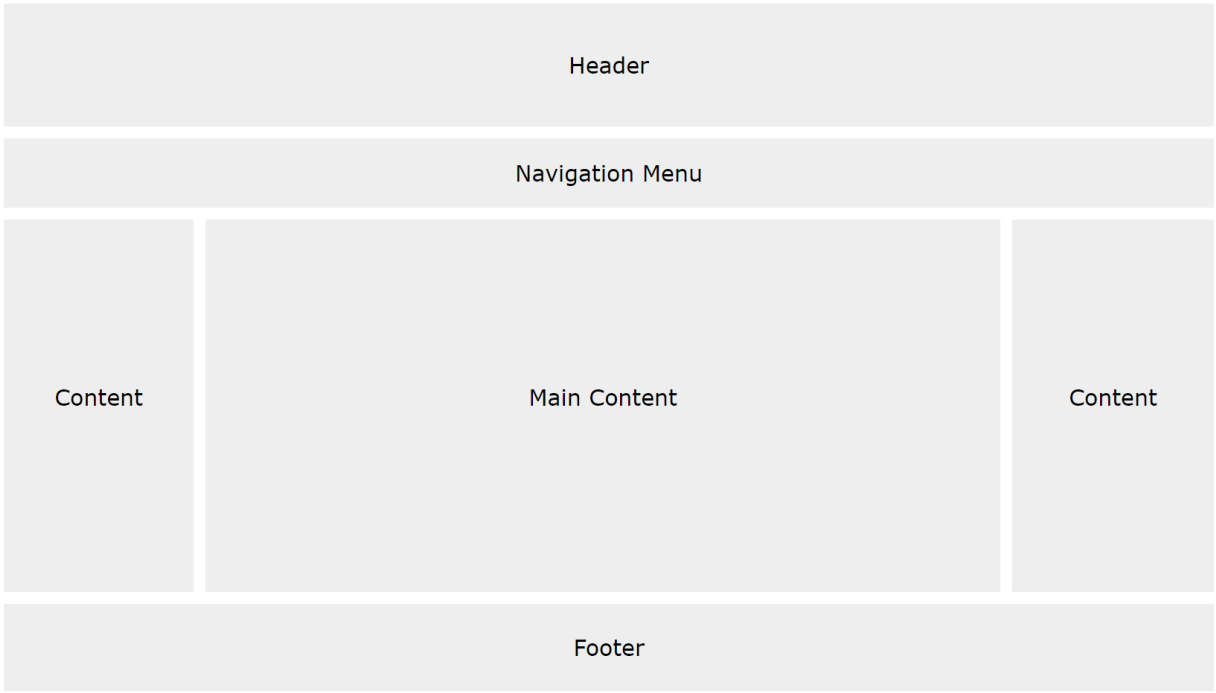
Reflection

This task has deepened my understanding of the following elements which has further helped in the development of my website. I have never considered these elements as important until I studied them in more detail.

Notes

- *Page layout*

Very generic web page layout but after viewing all the other options I have decided to go with this one because it's simple and not complicated. Another advantage of this layout is that it is very familiar to everyone which means better user experience as they will be quite comfortable with using the website and the ease of usability almost always attracts the user back to the website.



- *Navigation*

The navigation bar is an important element of a website's design since it allows users to quickly visit any section within the site. This also Increases the functionality of the website.

- *Colour and Graphics*

Colour and graphics is another way to attract attention of the users but it can also help to organize content within your website.

❖ *Content*

The content needs to be meaningful and concise. Too much content could overwhelm the user creating a bad impression which could possibly lead the user to click off the website.

❖ *Functionality*

Website functionality is defined by the ease of how a user can navigate through the website, get the information they are seeking efficiently, and/or purchase the product they want.

Screenshot Task 2

Input CSS Navigation Bar

```css
/* Add a black background color to the top navigation bar */
.topnav {
  overflow: hidden;
  background-color: #6B705C;
}

/* Style the links inside the navigation bar */
.topnav a {
  float: left;
  display: block;
  color: black;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}

/* Change the color of links on hover */
.topnav a:hover {
  background-color: rgb(165, 165, 141);
  color: black;
}

/* Style the "active" element to highlight the current page */
.topnav a.active {
  background-color: #50503F;
  color: white;
}

/* Style the search box inside the navigation bar */
.topnav input[type=text] {
  float: right;
  padding: 6px;
  border: none;
  margin-top: 8px;
  margin-right: 16px;
  font-size: 17px;
}
```

Output CSS Navigation Bar

Task 3 | A User Stories & UI/UX Design For Your Project

Reflection

This task taught me the importance of developing user stories before actually building the website first as it allows you to highlight key user requirements. Through research it is known that poor use of user stories could lead to misunderstanding and loss but when developed correctly it increases efficiency and potentially saves money.

Notes

Template

As a [type of user], I want to [action] so that [purpose].

What are User Stories

A user story is an informal, natural language description of one or more features of a website. A user story is a tool used in many fields to capture a description of a website from an end-user perspective. A user story describes the type of user, what they want and why. A user story helps to create a simplified description of a requirement.

User stories are often recorded on index cards, on Post-it notes, or in project management software. Depending on the project, user stories may be written by various stakeholders such as clients, users, managers or development team members.

Why Are User Stories Important

The process of writing User Stories is crucial. User stories help to reveal what to code and why which saves a lot of time on development.

My User Stories / Project Progress

Statement	Acceptance Criteria	Priority
As a user I want to able to quickly book online appointment's so that I don't have to waste time by calling them up.	1. *The user can simply book an appointment by checking for availabilities for their preferred service.* 2. *The user can filter their search by date time in order to make the process quicker.* 3. *The user can also filter their search by location and price range.*	*High Priority*
As a user I want to be able to read more about the service provider than listed on the website so that I can have a better sense of the service provider.	1. *When the user selects their service provider, they're able to click on their name which will hyperlink them to the service providers website.* 2. *The user is also provided with an extensive introduction of the service provider on our website*	*High Priority*
As a user I want to able to save my details so that I don't have to fill them in every time.	1. *The user can create an account which will automatically save all their details.*	*High Priority*
As a user I want to be able to get extra discount's for booking through this website	1. *When a user signs up with us they get exclusive deals from the service provided through us* 2. *The user also gets a complimentary discount when they book through us.*	*Medium Priority*
As a user I want to be able to keep track of my appointments so that I don't forget the bookings I have made	1. *The user can create an account which will save all their detail ana also keep track of their booking* 2. *The user can also link there booking to their calendar*	*High Priority*

As a user I want to able to be guaranteed that this website isn't a scam, so I don't lose all my money.	1. *The user will get money back if they get scammed in any way while using our website.* 2. *The user can always send us an email or contact us if they have any concerns.*	*Medium Priority*
As a user I want to be able to cancel my booking if I can't make It to it so that I don't lose my money and I don't receive the service	1. *The user must inform the service provider 24 hours before the appointment to get a refund.* 2. *The user can even reschedule if they want to*	*Medium Priority*
As a user I want to be able to access this website on any device so even when I am on the go, I can access it.	1. *The user can always access the website on their phone.* 2. *This website is a responsive website hence it works on every device.*	*High Priority*
As a new service provider, I would want to be able to contact you directly so I can add my business	1. *The service provider can fill the contact form available on our website.* 2. *The service provider can also call us directly.*	*Medium Priority*
As a user I want to be able to search for my desired service so I can be time efficient	1. *The user can quicky search their desired service through the search bar.*	*High Priority*

Acceptance Criteria

Acceptance criteria is basically a list of clear parameters that are testable and must be met before considering the user story is complete.

Priority

Helps to organize each user story by importance

Task 4 | Vu Use Graphics – Media & API'S

Reflection

This task is what encouraged me to add more graphics into my project. However, What I also learnt is that Graphic and media can be added to a webpage through linking and embedding images and video. Additionally In my project I am not really required to add embedded videos, but I have started to add images and I have also started to create a contact form which will later have a Vue component so that it can collect and record users' details. (API)

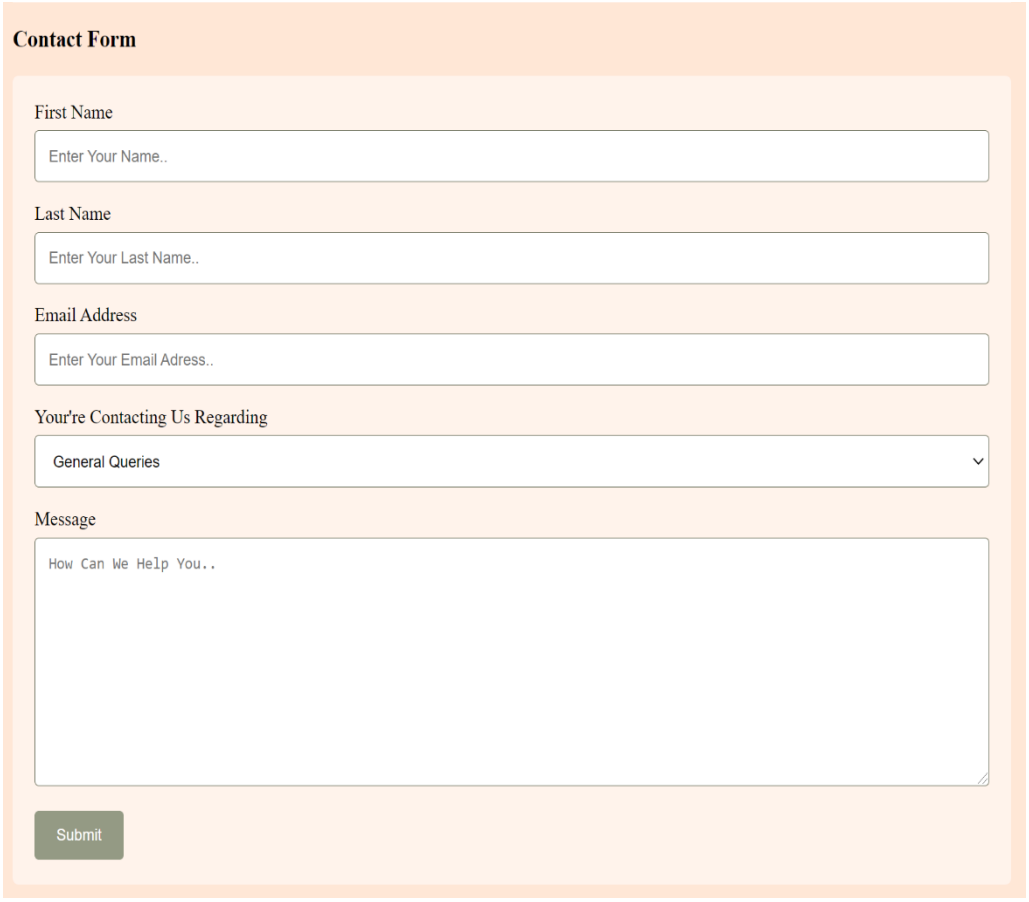
Screenshots

Input Contact Us Form

```
<!-- CONTACT FORM  -->

<h3>Contact Form</h3>

<div class="container">
    <form action="/action_page.php">
        <label for="fname">First Name</label> <!-- Title of the box  -->
        <input type="text" id="fname" name="firstname" placeholder="Enter Your Name..">

        <label for="lname">Last Name</label> <!-- Title of the box  -->
        <input type="text" id="lname" name="lastname" placeholder="Enter Your Last Name..">

        <label for="lname"> Email Address</label> <!-- Title of the box  -->
        <input type="text" id="lname" name="Email Address" placeholder="Enter Your Email Adress..">

        <!-- Drop Down Menu  -->
        <label for="country">Your're Contacting Us Regarding </label> <!-- Title of the box  -->
        <select id="country" name="country">
            <option value="australia">General Queries</option>
            <option value="canada">Experience Issue</option>
            <option value="usa">Membership</option>
            <option value="usa">Login Issues</option>
            <option value="usa">Joining Our Team</option>
            <option value="usa">Other</option>

        </select>

        <label for="subject">Message</label> <!-- Title of the box  -->
        <textarea id="subject" name="subject" placeholder="How Can We Help You.." style="height:200px"></textarea>

        <input type="submit" value="Submit"> <!-- Submit Button  -->
    </form>

</div>
```

```
<style>   /* Style Sheet */
    body {  /* Font Family  */
        font-family: Arial, Helvetica, sans-serif, ;
        background-color: #FFE7D6;
        margin: 0;
        padding-top: 50px;
        padding-right: 30px;
        padding-bottom: 50px;
        padding-left: 80px;
    }

        /* Border */
    * {
        box-sizing: border-box;
    }


    /* Border Of the contact form boxes */
    input[type=text],
    select,
    textarea {
        width: 100%;
        padding: 12px;
        border: 1px solid #6B705C;
        border-radius: 4px;
        box-sizing: border-box;
        margin-top: 6px;
        margin-bottom: 16px;
        resize: vertical;
    }
    /* Submit Button */
    input[type=submit] {
        background-color: #949A84;
        color: white;
        padding: 12px 20px;
        border: none;
        border-radius: 4px;
        cursor: pointer;
    }
    /* Submit Button when you hover */
    input[type=submit]:hover {
        background-color: #B7B7A4;
    }
    /* Background of the contact form */
    .container {
        border-radius: 5px;
        background-color:#FFF3EB;
        padding: 20px;
    }
```

Output Contact Us Form



GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

WEEK 3 | JavaScript Functions

Reflection

I learnt quite a bit from this week's tasks and lectures. It taught me why JavaScript is important and how is it different to the other languages. What stood out the most for me is that all JavaScript functions convey one thing, which is to try and make your website more interactive and dynamic for the users. Overall, these tasks gave me a better insight upon how I should make my website more interactive and dynamic.

Researching components and concepts allowed me to gain a deeper understanding of how Vue works and what type of functions it contains. It really allowed me to better understand the Vue components/concepts Additionally it made clearer to me on what components I should implement in my project.

Task 1 | JavaScript String Methods

Notes

The JavaScript string is an object that represents a sequence of characters.

String methods help you to work with strings.

What are the different string methods in JavaScript?

There are 3 methods for extracting a part of a string: slice (start, end) substring(start, end) substr(start, length)

Task 2 | JavaScript Number &Array Methods

Notes

JavaScript Array Methods

What is an Array in JavaScript

A pair of square brackets [] represents an array in JavaScript. All the elements in the array are comma(,) separated.

In JavaScript, arrays can be a collection of elements of any type. This means that you can create an array with elements of type String, Boolean, Number, Objects, and even other Arrays.

Converting Arrays to Strings

The JavaScript method toString() converts an array to a string of (comma separated) array values.

JavaScript Number Methods

The Number() function converts the object argument to a number that represents the object's value. Converts different object values to their numbers

Task 3 | JavaScript Get/Set Methods

Notes

In JavaScript, accessor properties are methods that get or set the value of an object

For that, we use these two keywords:

***get** - to define a getter method to get the property value. It's a function without arguments, that works when a property is read.*

***set** - to define a setter method to set the property value. It's a function with one argument, that is called when the property is set*

Screenshots

JavaScript String Methods Input

HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Week_3.css">
</head>

<body>

<h1><i><b>JavaScript Functions</i></b></h1>

<h1><i><b>Task 1</i></b></h1>

<h2><i>JavaScript String Methods</i></h2>

<h3>The Total Lenght of "Hi My Name is Nandinee" Is:</h3>
<p id="Trial"></p>

<h2><i>JavaScript String Methods</i></h2>

<h3>The Slice() Method Extract A Part Of A String & Returns The Extracted Parts In A New String:</h3>
<p id="Trial_2"></p>

```

JavaScript String Methods Output

JavaScript Functions

Task 1

JavaScript String Methods

The Total Lenght of "Hi My Name is Nandinee" Is:

22

JavaScript String Methods

The Slice() Method Extract A Part Of A String & Returns The Extracted Parts In A New String:

Shiven

JavaScript Number & Array Methods Input

HTML

```
<h1><i><b>Task 2</i></b></h1>


<h2><i>JavaScript Array Methods</i></h2>

<h3>The toString() Method Returns An Array</h3>
<h3>This Array Contains The Following Brand Names: </h3>

<p id="Trial_3"></p>

<h2><i>JavaScript Number Methods</i></h2>
<h3>The toString() Method Converts A Number To A String.</h3>

<p id="Trial_4"></p>
```

JavaScript Number & Array Methods Output

JavaScript Get/Set Methods Input

HTML

```
<h1><i><b>Task 3</i></b></h1>

<h2><i>JavaScript getTime() Method</i></h2>
<h3>The internal clock in JavaScript starts at midnight January 1, 1970.</h3>
<h3><h3>Click the button below to display the number of milliseconds since midnight, January 1, 1970.</h3></h3>
<button onclick="newFunction()">Try it</button>

<p id="Trial_5"></p>

<br>

<h2><i>JavaScript getFullYear() Method</i></h2>
<h3>Click The Button Below To Reveal Our Current Year</h3>
<button onclick="myFunction()">Try it</button>

<p id="Trial_6"></p>
```

JavaScript Get/Set Methods Output

Task 3

JavaScript getTime() Method

The internal clock in JavaScript starts at midnight January 1, 1970.

Click the button below to display the number of milliseconds since midnight, January 1, 1970.

Try it

1631852241400

JavaScript getFullYear() Method

Click The Button Below To Reveal Our Current Year

Try it

2021

JS | CSS WEEK 3

```
// Task 1
let text = "Hi My Name is Nandinee";
document.getElementById("Trial").innerHTML = text.length;

// Task 1
let str = "Megan, Shiven, Lucy";
document.getElementById("Trial_2").innerHTML = str.slice(7,13);

// Task 2
var brands = [" Dior", " Chanel", " Louis Vuitton", " Prada"];
document.getElementById("Trial_3").innerHTML = brands.toString();

let x = 1234;
document.getElementById("Trial_4").innerHTML =
  x.toString() + "<br>" +
   (1234).toString() + "<br>" +
   (100 + 1134).toString();



   function newFunction() {
    var d = new Date();
    var n = d.getTime();
    document.getElementById("Trial_5").innerHTML = n;

}

// Task 3
function myFunction() {
    var d = new Date();
    var n = d.getFullYear();
    document.getElementById("Trial_6").innerHTML = n;
  }
```

```
body {
    background-color: #F3E9E2;
  }

  h1 {
    color: #81876E;
    margin-left: 50x;
  }

  h2 {
    color: #CB997E;
    margin-left: 50x;
  }

```

Task 4 | Concepts / Components

Notes

Computed Properties and Watchers

Computed Properties

computed properties are cached based on their reactive dependencies. A computed property will only re-evaluate when some of its reactive dependencies have changed. This means as long as `message` has not changed, multiple access to the `reversedMessage` computed property will immediately return the previously computed result without having to run the function again.

Computed properties are by default getter-only, but you can also provide a setter when you need it

Watchers

While computed properties are more appropriate in most cases, there are times when a custom watcher is necessary.

That's why Vue provides a more generic way to react to data changes through the `watch` option. This is most useful when you want to perform asynchronous or expensive operations in response to changing data.

Computed Properties VS Watchers

In this case, using the `watch` option allows us to perform an asynchronous operation (accessing an API), limit how often we perform that operation, and set intermediary states until we get a final answer. None of that would be possible with a computed property.

In addition to the `watch` option, you can also use the imperative.

Class and Style Bindings

Bindings

v-bind lets you bind an HTML attribute to a JavaScript expression. There are two broad use cases for this one-way data binding: Binding to built-in attributes, like href or class. Passing props to a child component.

Class Bindings

Class binding is used to set a class property of a view element. We can add and remove the CSS class names from an element's class attribute with class binding.

Style Binding

Style binding is used to set a style of a view element. We can set the inline styles of an HTML element using the style binding in angular. You can also add styles conditionally to an element, hence creating a dynamically styled element.

Conditional Rendering

Conditional Rendering in Vue makes it easy to toggle the presence of any element in the DOM based on a certain condition. The directives v-if and v-else are used for this purpose. The v-if directive can be used to conditionally render a block. The v-else-if directive can also be used to chain multiple conditionals.

List Rendering

List Rendering is a component that allows me to store data about the services we offer to our users such as their location

Form Input Bindings

The v-model directive to create two-way data bindings on form input, text area, and select elements. It automatically picks the correct way to update the element based on the input type.

The v-bind is called one-way binding which means it binds our data one way. It can also be used to bind HTML attributes. This example shows a one-way databinding using our style element using v-bind.

The v-model is essentially syntax sugar for updating data on user input events, plus special care for some edge cases.

Basic Usage of the v-model

1. *text and text area elements use value property and input event;*
2. *checkboxes and radio buttons use checked property and change event;*
3. *select fields use value as a prop and change as an event.*

Difference Between v-model and v-bind

V-MODEL	*V-BIND*
v-model can be changed or assigned.	*v-bind can only be assigned.*
v-model is a two-way binding.	*v-bind is a one-way binding.*
v-model is used for binding form elements like inputs, radio buttons, textarea, checkboxes.	*It is used for binding data, attributes, expressions, class, styles.*

Components Basics

Components are one of the most powerful features of Vue. js. They help you extend basic HTML elements to encapsulate reusable code.

Components can be reused as many times as you want or used in another component, making it a child component.

Component Registration

When we define components, we normally use camel case. Usually, a component's scope is global, but we can register it locally as well. When the scope is global it can be used in the template of any root Vue instance. If we make the component registration private, then it can be left out of our final build meaning less code for the client to download which therefore improves overall performance.

Props

- *Props allows us to pass variables and other information around & between different components.*
- *A Components can have as many props as desired and by default, any value can be passed to any props.*

3 key things that need to be kept in mind when using props

1. *All props form a one-way-down binding between the child property and the parent one so the way that we pass data from a parent component to its child components is by passing it down from the parent to its child components*
2. *You cannot pass data from a child to a parent.*
3. *Props are read-only and cannot be modified. As the parent component "owns" that value it passed down, the child can't modify it*

Therefore, every time the parent component is updated, all props in the child component will be refreshed with the latest value.

I will use props when the user wants to check their schedule, the data will be passed into the schedule using props

Custom Events

In Vue, we can listen for the HTML DOM events using the v-on directive. This adds an event listener to listen for the built-in DOM events

We can create a custom event from our Vue components by using the $emit method.

we can pass in an object as the second parameter. This is useful if we need to provide our parent components with more information like a user object

Like the HTML DOM events. An element will "emit" an event and the data from that event. Just like we can "listen" for when an element emits an event, we can create custom events that come from our components. For instance, we can have an HTML button that emits a click event. We can also have a custom component called <ColorPicker/> that emits a color- chosen event

```
<!-- listening on a button for a click event -->
<button @click="doSomething">Click Me</button>
<!-- listening on a color-picker for a custom event called color-chosen -->
<ColorPicker @color-chosen="doSomething" />
```

This is a robust way to handle custom events because we know that a child component has inputs (props) and outputs (custom events). We can reuse this same component in many places and always know that it has the same API.

An element will "emit" an event and the data from that event. This is a robust way to handle custom events because we know that a child component has inputs (props) and outputs (custom events).

Slots

Slots are a mechanism for Vue components that allows you to compose your components in a way other than the strict parent-child relationship. Slots give you an outlet to place content in new places or make components more generic.

***Slot props** allow us to turn slots into reusable templates that can render different content based on input props.*

This is most useful when you are designing a reusable component that encapsulates data logic while allowing the consuming parent component to customize part of its layout.

Slot Content

Vue implements a content distribution API inspired by the Web Components spec draft, using the <slot> element to serve as distribution outlets for content.

Dynamic & Async Components

Dynamic Component

Vue dynamic components enable users to switch between two or more components without routing, and even retain the state of data when switching back to the initial component.

***The central idea** is to let users dynamically mount and unmount components in the user interface without using routers.*

What Dynamic Components Are

Dynamic means, that the components' location in the application is not defined at buildtime. That means, that it is not used in any angular template. Instead, the component is instantiated and placed in the application at runtime.

Async Component

An async component can render its state in a meaningful way like any other component or be logic-only. In that case it doesn't render any UI but instead passes its state down to its children.

These components can be tooltips, popovers, modals, etc, and can be used as async components

We can define an async component by creating a function that returns an object that has the promise that resolves to the component as the component that's rendered.

Project Progress

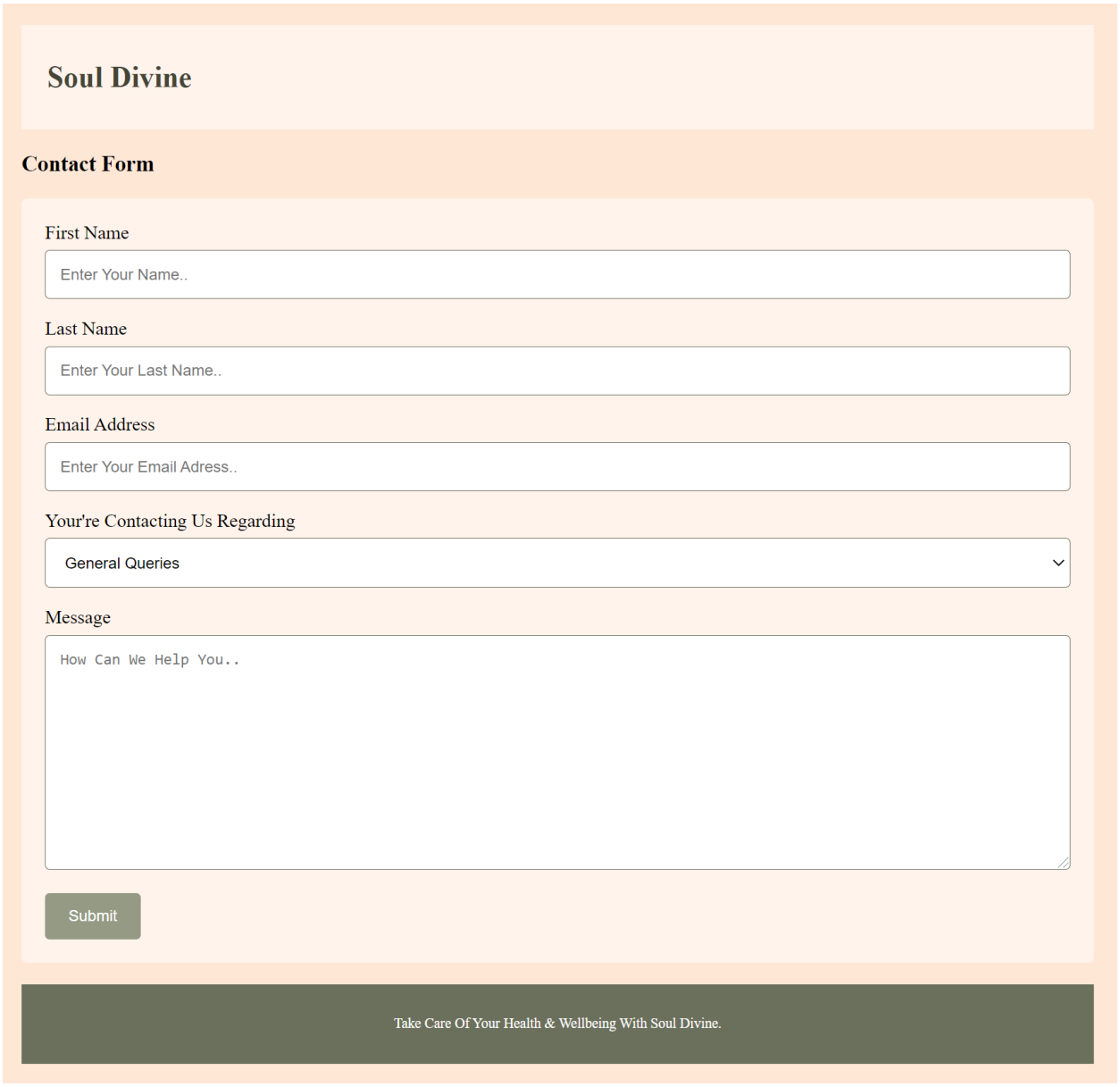
Fully Finished My Contact Us Form Page & About Us Page

Contact Us Input

```
        /* Background of the header */
        .header {
            overflow: hidden;
            background-color: #FFF3EB;
            padding: 20px 10px;
        }

        /* Style the header links */
        .header a {
            float: left;
            color: #454536;
            text-align: center;
            padding: 12px;
            text-decoration: none;
            font-size: 18px;
            line-height: 25px;
            border-radius: 4px;
        }

        /* Style the logo link (notice that we set the same value of line-height and font-size to prevent the header to increase when the font gets bigger */
        .header a.logo {
            font-size: 25px;
            font-weight: bold;
        }

        /* Change the background color on mouse-over */
        .header a:hover {
            background-color: #6B705C;
            color: black;
        }

        /* Style the active/current link*/
        .header a.active {
            background-color: #6B705C;
            color: #6B705C;
        }

        /* Float the link section to the right */
        .header-right {
            float: right;
        }

        .footer {
            background-color: #6B705C;
            color: #ffffff;
            text-align: center;
            font-size: 12px;
            padding: 15px;
        }

        /* Add media queries for responsiveness - when the screen is 500px wide or less, stack the links on top of each other */
        @media screen and (max-width: 500px) {
            .header a {
                float: none;
                display: block;
                text-align: left;
            }

            .header-right {
                float: none;
            }
        }
    </style>
</head>

<body>

    <!-- The interactive Header that takes you back to the home page  -->
    <div class="header">
        <a href="Code.html" class="logo">Soul Divine</a>
        <div class="header-right">
        </div>
    </div>
```

```
    <br>
        <div class="footer">  <!-- Footer -->
            <p>Take Care Of Your Health & Wellbeing With Soul Divine.</p>
        </div>

</body>

</html>
```

Contact Us Output



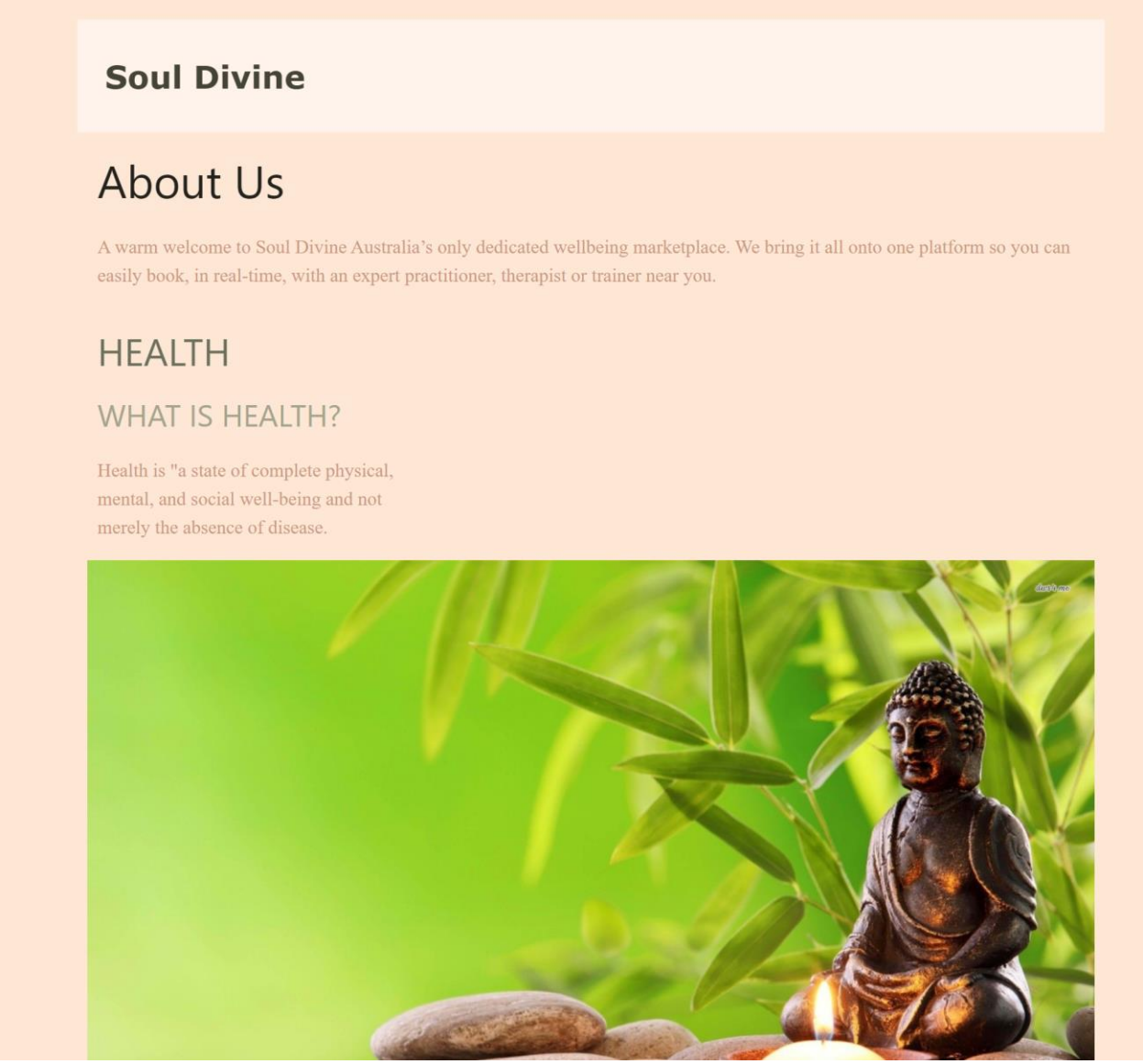
About Us Input

```
<!DOCTYPE html>
<html>

<head>
    <!-- Responsive Design -->
    <title>W3.CSS</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <style>
        body {
            /* Font Family  */
            font-family: Arial, Helvetica, sans-serif, ;
            background-color: #FFE7D6;
            margin: 0;
            padding-top: 20px;
            padding-right: 40px;
            padding-bottom: 20px;
            padding-left: 70px;
        }

        /* Style Code for Heading 1 */
        h1 {
        font-family: "Times New Roman", Times, serif;
        color: #22221B;
        }

        /* Style Code for Heading 2  */
        h2 {
        text-transform: uppercase;
        color: #6B705C;
        }

        /* Style Code for Heading 3  */
        h3 {
        text-transform: uppercase;
        color: #A5A58D;
        }

        /* Style Code For paragraph 1  */
        .p1 {
        font-family: "Times New Roman", monospace;
         color: #CB997E;
}
        /* Background of the header */
        .header {
            overflow: hidden;
            background-color:#FFF3EB;
            padding: 20px 10px;
        }

        /* Style the header links */
        .header a {
            float: left;
            color: #454536;
            text-align: center;
            padding: 12px;
            text-decoration: none;
            font-size: 18px;
            line-height: 25px;
            border-radius: 4px;
        }

        /* Style the logo link (notice that we set the same value of line-height and font-size to prevent the header to increase when the font gets bigger */
        .header a.logo {
            font-size: 25px;
            font-weight: bold;
        }

        /* Change the background color on mouse-over */
        .header a:hover {
            background-color: #6B705C;
            color: black;
        }
```

```
        /* Style the active/current link*/
        .header a.active {
            background-color: #6B705C;
            color: #6B705C;
        }

        /* Float the link section to the right */
        .header-right {
            float: right;
        }

        /* Footer */
        .footer {
            background-color: #6B705C;
            color: #ffffff;
            text-align: center;
            font-size: 12px;
            padding: 15px;
        }

        /* Add media queries for responsiveness - when the screen is 500px wide or less, stack the links on top of each other */
        @media screen and (max-width: 500px) {
            .header a {
                float: none;
                display: block;
                text-align: left;
            }

            .header-right {
                float: none;
            }
        }

    </style>
```

```
<body>

    <!-- The interactive Header that takes you back to the home page  -->
    <div class="header">
        <a href="Code.html" class="logo">Soul Divine</a>
        <div class="header-right">
        </div>
    </div>

    <!-- Header  -->
    <div class="w3-container ">
        <h1>About Us</h1>
        <p class="p1">A warm welcome to Soul Divine Australia’s only dedicated wellbeing marketplace. We bring it all onto one platform so you can easily book, in real-time, with an expert practitioner, therapist
    </div>

 <!-- Health Section   -->
    <div class="w3-row-padding">
        <div class="w3-third">
            <h2>Health</h2>
            <h3>What Is Health?</h3>
            <p class="p1">Health is "a state of complete physical, mental, and social well-being and not merely the absence of disease.</p>
        </div>

        <img src=" https://i.pinimg.com/originals/0a/37/a0/0a37a07f55454ee25aae8391f4c7a8e0.jpg " alt="Nature"
            class="responsive" style="width:100%;">
    <!-- Wellbeing Section  -->
    <div class="w3-row-padding">
        <div class="w3-third">
            <h2>Wellbeing</h2>
            <h3>What Is Wellbeing?</h3>
            <p class="p1"> Wellbeing, also known as wellness, prudential value or quality of life, refers to what is intrinsically valuable relative to someone. So the well-being of a person is what is ultimately

        </div>

        <img src="https://img.thedailybeast.com/image/upload/v1542230904/181114-deopoukos-reiki-tease_cexhiu.jpg" alt="Nature"
            class="responsive" style="width:100%;">

    <!-- Fitness Section  -->

        <div class="w3-third">
            <h2>Fitness</h2>
            <h3>What Is Fitness?</h3>
            <p class="p1"> Fitness is the condition of being physically fit and healthy and involves attributes that include, but are not limited to mental acuity, cardiorespiratory endurance, muscular strength, m
            </p>

        </div>

        <img src="https://www.fitnessfirst.com.au/static/1702f7a4cb68d57bb7854b794bddb188/39f27/THIS_INTENSE_CORE_WORKOUT_WILL_FIRE_UP_YOUR_MIDDLE_-_thumb.jpg" alt="Nature"
            class="responsive" style="width:100%;">

        <!-- Organic Treatment Section  -->
        <div class="w3-third">
            <h2>Organic Treatment</h2>
            <h3>What Is Organic Treatment ?</h3>
            <p class="p1">By using a combination of healthy diet, simple self-help techniques, for example, breathing and relaxation exercises, beneficial herbs and general exercise, naturopathy seeks to promote t

        </div>

        <img src=" https://www.israel21c.org/wp-content/uploads/2019/02/main-pic-8-1168x657.jpg" alt="Nature"
            class="responsive" style="width:100%;">

            <br>


    </div>
        <br>
        <div class="footer">  <!-- Footer -->
            <p>Take Care Of Your Health & Wellbeing With Soul Divine.</p>
        </div>

</body>

</html>
```

About Us Output



GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

WEEK 4 | Vue.js Framework

Reflection

I learnt quite a bit from this week's tasks and lectures. It taught me why JavaScript is important and how is it different

The main focus of this week was learning about Vue in more depth. This week's tasks challenged me a little but as my Vue wouldn't connect with my visual studio code and it was such a hassle to make it work. However, after re-installing and re-starting Vue it finally worked. Learning about Vue in more depth gave me a clearer idea on how I can implement these concepts within my websites.

What I also leant and found interesting is how the internet is connected to everything from our phones to tv.

We also learnt about the difference between dynamic and static web page I applied this knowledge to my user stories to add more meaning and depth. This helped to me vision my website even better.

Task 1 | Declarative Rendering

Notes

Declarative rendering in Vue enables us to render data to the DOM using straightforward template syntax. Double curly braces are used as placeholders to interpolate the required data in the DOM.

Screenshots

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Week_4.css">
    <title>Vue js CDN</title>
</head>


<body>

    <h1><i><b>Week 4 Task 1</i></b></h1>

    <h2><i>Declarative Rendering</i></h2>


    <div id="app">
        <p>
            {{ name }} {{ address }}
        </p>

        <p v-html="htmlrendering">
            {{ htmlrendering }}
        </p>

        <p v-html="htmlparagraph">
            {{ htmlparagraph }}
        </p>

        <img v-img="htmlimage"
            src="

    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>

    <script src="Week_4.js"></script>

</body>

</html>
```

Input JS

```
var app = new Vue({
    el: "#app",
data:{
    name:"Hi I'm Nandinee Pardasani",
    address:"Who Is Currently Studying SIT120 In Year 2021  ",
    htmlrendering:"<h3> For This task I Will Rendar HTML Content Throught Vue </h3>",
    htmlparagraph: "<p> This Is A HTML Paragraph Rendered Through Vue. </p>",
    htmlimage:"",
}
});
```

Output

Week 4 Task 1

Declarative Rendering

Hi I'm Nandinee Pardasani Who Is Currently Studying SIT120 In Year 2021

For This task I Will Rendar HTML Content Throught Vue

This Is A HTML Paragraph Rendered Through Vue.

Task 2 | Conditional & Loops

Notes

You can define sections of code that either repeat in a loop or conditionally execute.

Conditional and loops in Vue are used to render the element based on the condition

v-if

Conditionally renders the element or View.

v- else

Conditionally renders the else block of a `v-if` directive.

v-show

Conditionally display an element or View.

v-for

Renders a list of items using the data from an Array.

It also Demonstrates that we can bind data, and dynamically render the UI elements to the View based on the values inside the Array we are looping through. V-for is basically a For Each loop

Screenshots

Input HTML

```html
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Week_4_T2.css">
    <title>Vue js CDN</title>
</head>
<body>

<h1><i><b>Week 4 Task 2</i></b></h1>

<h2><i> Conditionals & Loops</i></h2>

<h3>Testing If True...</h3>

        <div id="new">
            <span v-if="seen">This Message Will Only Be Displayed When Set To True</span>
            <span v-else> You Can't See What Is Written Above This Only Happens When Set To False</span>
        </div>

<script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>

<script src="Week_4_T2.js"></script>

</body>

</html>
```

Input JS When True

```js
var newapp = new Vue({
    el: "#new",
    data: {
    seen: true,
    },
});
```

Input JS When False

```js
var newapp = new Vue({
    el: "#new",
    data: {
    seen: false,
    },
});
```

Week 4 Task 2

Conditionals & Loops

Testing If True...

This Message Will Only Be Displayed When Set To True

Output When False

Week 4 Task 2

Conditionals & Loops

Testing If True...

You Can't See What Is Written Above This Only Happens When Set To False

Project Progress

Fully Finished My Home Page & Rate Us Page & Prototypes

Home Page Input CSS

```css
/* background colour */
body {
  background-color: #FFF3EB;

  padding-top: 20px;
  padding-right: 10px;
  padding-bottom: 20px;
  padding-left: 40px;
}

 /* SIDE Panals Responsive Design  */


.row::after {
  content: "";
  clear: both;
  display: table;
}

[class*="col-"] {
  float: left;
  padding: 15px;
}

html {
  font-family: "Lucida Sans", sans-serif;
}

.header {
  background-color: #9933cc;
  color: #ffffff;
  padding: 15px;
}

.menu ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}

.menu li {
  padding: 8px;
  margin-bottom: 7px;
  background-color: #B7B7A4;
  color: #ffffff;
  box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);
}
 /* background colour when you hover the side panals  */
.menu li:hover {
  background-color: #D0A98A;
}
  /* Side bar */
.aside {
  background-color: #D0A98A;
  padding: 18px;
  color: #ffffff;
  text-align: center;
  font-size: 14px;
  box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);
}

 /* Footer */
.footer {
  background-color: #6B705C;
  color: #ffffff;
  text-align: center;
  font-size: 12px;
  padding: 15px;

}

/* For mobile phones: */
[class*="col-"] {
  width: 100%;
}
```

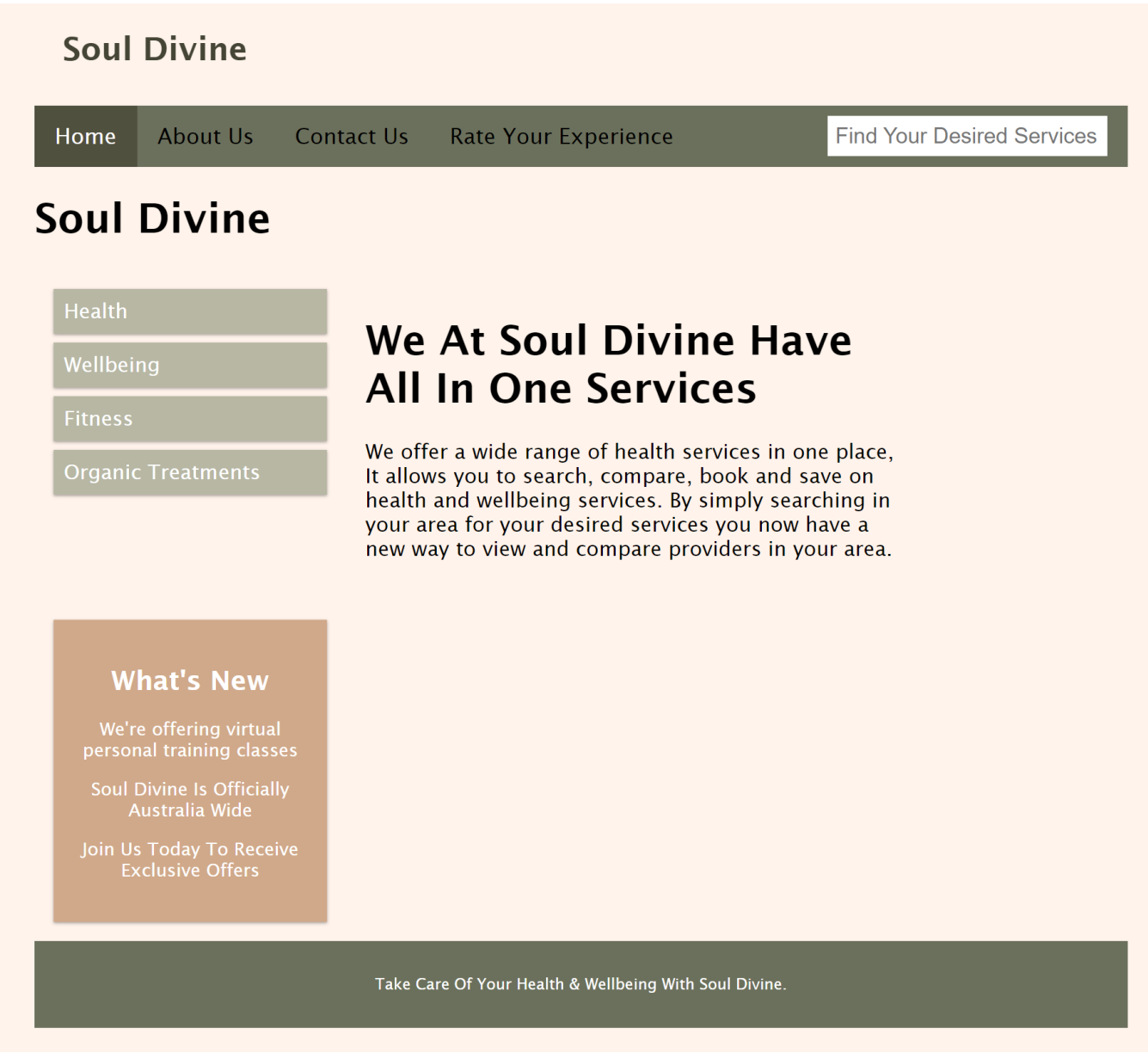
```css
@media only screen and (min-width: 600px) {
  /* For tablets: */
  .col-s-1 {width: 8.33%;}
  .col-s-2 {width: 16.66%;}
  .col-s-3 {width: 25%;}
  .col-s-4 {width: 33.33%;}
  .col-s-5 {width: 41.66%;}
  .col-s-6 {width: 50%;}
  .col-s-7 {width: 58.33%;}
  .col-s-8 {width: 66.66%;}
  .col-s-9 {width: 75%;}
  .col-s-10 {width: 83.33%;}
  .col-s-11 {width: 91.66%;}
  .col-s-12 {width: 100%;}
}
@media only screen and (min-width: 768px) {
  /* For desktop: */
  .col-1 {width: 8.33%;}
  .col-2 {width: 16.66%;}
  .col-3 {width: 25%;}
  .col-4 {width: 33.33%;}
  .col-5 {width: 41.66%;}
  .col-6 {width: 50%;}
  .col-7 {width: 58.33%;}
  .col-8 {width: 66.66%;}
  .col-9 {width: 75%;}
  .col-10 {width: 83.33%;}
  .col-11 {width: 91.66%;}
  .col-12 {width: 100%;}
}



.header {
  overflow: hidden;
  background-color: #FFF3EB;
  padding: 20px 10px;
}

/* Style the header links */
.header a {
  float: left;
  color:#454536;
  text-align: center;
  padding: 12px;
  text-decoration: none;
  font-size: 18px;
  line-height: 25px;
  border-radius: 4px;
}

/* Style the logo link (notice that we set the same value of line-height and font-size to prevent the header to increase when the font gets bigger */
.header a.logo {
  font-size: 25px;
  font-weight: bold;
}

/* Change the background color on mouse-over */
.header a:hover {
  background-color: #6B705C;
  color: black;
}

/* Style the active/current link*/
.header a.active {
  background-color: #6B705C;
  color: #6B705C;
}

/* Float the link section to the right */
.header-right {
  float: right;
}
```

```css
/* Add media queries for responsiveness - when the screen is 500px wide or less, stack the links on top of each other */
@media screen and (max-width: 500px) {
  .header a {
    float: none;
    display: block;
    text-align: left;
  }
  .header-right {
    float: none;
  }
}
```

Home Page Input HTML

```
<!DOCTYPE html>
<html>

<head> <!-- Responsive Design -->
    <title>Page Title</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" href="code.css">

</head>


<body>

    <div class="header">  <!-- Header/ Title Of the page -->
        <a href="#code.html" class="logo">Soul Divine</a>

        <div class="header-right">
        </div>
      </div>


    <div class="topnav">  <!-- Navigation Bar -->
        <a class="active" href="Code.html">Home</a>
        <a href="aboutus.html">About Us</a>
        <a href="contact.html">Contact Us</a>
        <a href="Rate.html">Rate Your Experience</a>
        <input type="text" placeholder="Find Your Desired Services">
    </div>







     <!-- The title  -->
    <h1> Soul Divine </h1>
    </div>

    <div class="row">
        <div class="col-3 col-s-3 menu">
            <ul>
                <!-- 4 Side Bars -->
                <li>Health</li>
                <li>Wellbeing</li>
                <li>Fitness</li>
                <li>Organic Treatments</li>
            </ul>
        </div>
        <!-- Welcome Message -->
        <div class="col-6 col-s-9">
            <h1>We At Soul Divine Have All In One Services </h1>
            <p> We offer a wide range of health services in one place, It allows you to search, compare, book and save on health and wellbeing services. By simply s
        </div>

        <div class="col-3 col-s-12">
            <div class="aside">
                <h2>What's New </h2>  <!-- Side Bar -->
                <p>We're offering virtual personal training classes</p>
                    <p> Soul Divine Is Officially Australia Wide </p>
                    <p>Join Us Today To Receive Exclusive Offers </p>
            </div>
        </div>
    </div>




    <div class="footer">  <!-- Footer -->
        <p>Take Care Of Your Health & Wellbeing With Soul Divine.</p>
    </div>

</body>
```

Home Page Output



Rate Us Input CSS

```
<!DOCTYPE html>
<html>


<head>
    <!-- Responsive Design  -->
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style>
        body {  /* Font Fmaily */
            font-family: Arial, Helvetica, sans-serif, ;
            background-color: #FFE7D6;
            margin: 0;
            padding-top: 50px;
            padding-right: 30px;
            padding-bottom: 50px;
            padding-left: 80px;
        }


        /* Background Of the header   */
        .header {
            overflow: hidden;
            background-color: #FFF3EB;
            padding: 20px 10px;
        }

        /* Style the header links */
        .header a {
            float: left;
            color: #454536;
            text-align: center;
            padding: 12px;
            text-decoration: none;
            font-size: 18px;
            line-height: 25px;
            border-radius: 4px;
        }

        /* Style the logo link (notice that we set the same value of line-height and font-size to prevent the header to increase when the font gets bigger */
        .header a.logo {
            font-size: 25px;
            font-weight: bold;
        }

        /* Change the background color on mouse-over */
        .header a:hover {
            background-color: #6B705C;
            color: black;
        }

        /* Style the active/current link*/
        .header a.active {
            background-color: #6B705C;
            color: #6B705C;
        }

        /* Float the link section to the right */
        .header-right {
            float: right;
        }


        /* Add media queries for responsiveness - when the screen is 500px wide or less, stack the links on top of each other */
        @media screen and (max-width: 500px) {
            .header a {
                float: none;
                display: block;
                text-align: left;
            }

            .header-right {
                float: none;
            }
        }
    </style>
```

Rate Us Input HTML / JS

```
<body>
    <!-- The interactive Header that takes you back to the home page  -->
    <div class="header">
        <a href="Code.html" class="logo">Soul Divine</a>
        <div class="header-right">
        </div>



    </div>

    <h2>Rate Your Experince With Us </h2>  <!-- Title  -->

    <p>Please Rate Us Out Of 5 </p>   <!-- Title  -->

     <!-- Submit Button  -->

    <input id="numb">

    <button type="button" onclick="myFunction()">Submit</button>

    <p id="demo"></p>

     <!-- Java Script  -->


    <script>
        function myFunction() {
            // Get the value of the input field with id="numb"
            let x = document.getElementById("numb").value;
            // If x is Not a Number or less than one or greater than 10
            let text;
            if (isNaN(x) || x < 1 || x > 5) {
                text = "Please Rate Us Out Of 5 ";
            } else {
                text = "Thank You ";
            }
            document.getElementById("demo").innerHTML = text;
        }
    </script>

<br>
<br>

</body>

</html>
```

Rate Us Output

Soul Divine

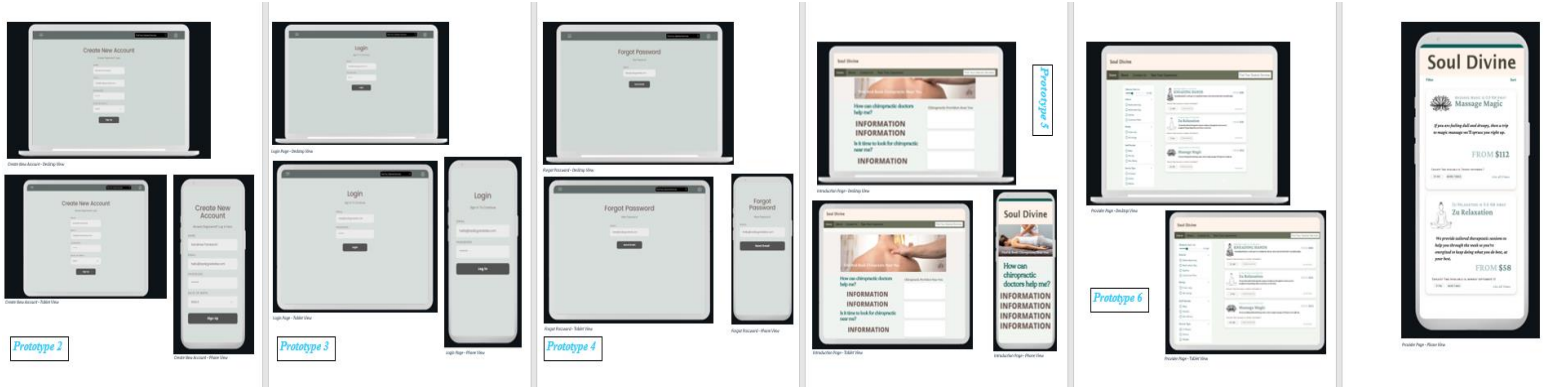
Rate Your Experince With Us

Please Rate Us Out Of 5

5 Submit

Thank You

Prototypes



GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

WEEK 5 | Vue.js Framework

Reflection

The main focus of this week was learning more about Vue. This week we took a deeper look at Vue components. We learnt about the hierarchy in which everything is positioned in Vue. This task gave me a better understand on how to implement Vue components. We also learnt the basic concept of handling user input which was actually quite interesting to see. This week for me was all about how to implement Vue in my future projects but it was also about learning how each component worked together

Task 1 | Learning Composing With Components

Composing With Components allows you to extend and inherit one or more components with minimal effort in order to provide reusable functionality across your app. Extension works by merging another component's options with your component.

Task 2 | Exploring Vue Framework

Conditional Rendering

Conditional Rendering in Vue makes it easy to toggle the presence of any element in the DOM based on a certain condition. The directives v-if and v-else are used for this purpose. The v-if directive can be used to conditionally render a block. The v-else-if directive can also be used to chain multiple conditionals.

Conditional rendering will be used multiple times in my website

1. *I will use conditional rendering as a way of verifying that the users have entered the correct email and password. Upon successful login the user will automatically be taken to the members home page.*
2. *I will use conditional rendering as a way of checking and verifying that the user has entered correct and appropriate details in the signing up process. When that's successful the user will automatically be taken to the members home page.*

List Rendering

List Rendering is a component that allows me to store data about the services we offer to our users such as their location

Task 3 | Understanding Handling User Input

Screenshots

Input JS

```
var app5 = new Vue({
    el: '#app-5',
    data: {
      message: 'Nandinee'
    },
    methods: {
      reverseMessage: function () {
        this.message = this.message.split('').reverse().join('')
      }
    }
  })



  new Vue({
    el: ".vue",
    data() {
      return {
        statementIsTrue: true
      };
    }
  });



  new Vue({
    el: ".vue_2",
    data() {
      return {
        namesThatRhyme: []
      };
    }
  });


  new Vue({
    el: ".vue_3",
    data() {
      return {
        howAreYouFeeling: "great"
      };
    }
  });

  new Vue({
    el: '.vue_4',
    data: {
      selected: ''
    }
  })
```

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Week_5.css">
</head>

<body>


<h1><i><b>Week 5 Task 3</i></b></h1>
<h2><i>Understanding Handling User Input</i></h2>

<h3> Check How Your Name Looks Like In Reverse: </h3>

    <div id="app-5">
      <p>{{ message }}</p>
      <button v-on:click="reverseMessage">Reverse Name </button>
    </div>

    <br>
    <h3> Are You Looking For Acupuncture  Practitioner  </h3>

    <div class="vue">
      <p>You have Decided This Statement Is <i><b> {{statementIsTrue}}</i></b></p>
      <label>
        <input type="checkbox" v-model="statementIsTrue" />

      </label>
    </div>

    <br>


    <div class="vue_2">

      <h3> Choose Your Prefered Service Provider  </h3>

      <p> {{namesThatRhyme.join(', ')}}</p>
      <label>
        <input type="checkbox" value="Luxe Therapy" v-model="namesThatRhyme" />
        Luxe Therapy
      </label>
      <label>
        <input type="checkbox" value="Merge Health" v-model="namesThatRhyme" />
        Merge Health
      </label>
      <label>
        <input type="checkbox" value="Max Therapy" v-model="namesThatRhyme" />
        Max Therapy
      </label>
    </div>

    <br>

    <div class="vue_3">
      <h3> Select To Your Comfort  </h3>
      <label>
        <input type="radio" value="A Female" v-model="howAreYouFeeling" />
        Female
      </label>
      <label>
        <input type="radio" value="A Male" v-model="howAreYouFeeling" />
        Male
      </label>
      <label>
        <input type="radio" value="Any" v-model="howAreYouFeeling" />
        Either
      </label>
      <p>I Would Be Most Comfortable With  <em>{{howAreYouFeeling}}</em> Doctor.</p>
    </div>

    <br>

    <h3> Select A Suitable Day  </h3>

    <div class="vue_4">
    <select v-model="selected">
      <option disabled value="">Please Select A Day </option>
      <option>Monday</option>
      <option>Tuesday</option>
      <option>Wednesday</option>
      <option>Thursday</option>
      <option>Friday</option>
    </select>
    <span>Selected Day: {{ selected }}</span>
  </div>
```

Output Without User Input

Week 5 Task 3

Understanding Handling User Input

Check How Your Name Looks Like In Reverse:

Nandinee

Reverse Name

Are You Looking For Acupuncture Practitioner

You have Decided This Statement Is ***true***

☑

Choose Your Prefered Service Provider

☐ Luxe Therapy ☐ Merge Health ☐ Max Therapy

Select To Your Comfort

○ Female ○ Male ○ Either

I Would Be Most Comfortable With *great* Doctor.

Select A Suitable Day

Please Select A Day Selected Day:

Output With User Input

Week 5 Task 3

Understanding Handling User Input

Check How Your Name Looks Like In Reverse:

eenidnaN

Reverse Name

Are You Looking For Acupuncture Practitioner

You have Decided This Statement Is ***true***

☑

Choose Your Prefered Service Provider

Luxe Therapy, Merge Health

☑ Luxe Therapy ☑ Merge Health ☐ Max Therapy

Select To Your Comfort

○ Female ◉ Male ○ Either

I Would Be Most Comfortable With *A Male* Doctor.

Select A Suitable Day

Wednesday Selected Day: Wednesday

Input Vue Form

Form.vue

```vue
<template>
  <form @submit.prevent="submitForm">
    <div class="form-container">
      <div class="form-control">
        <label for="user-name">Your Name:</label>
        <input id="user-name" name="user-name" type="text" v-model="userName" />
      </div>
      <div class="form-control">
        <label for="referrer">What do you use for your social links?</label>
        <select id="referrer" name="referrer" v-model="referrer">
          <option value="google">Google</option>
          <option value="Linkedin">Linkedin</option>
          <option value="Discord">Discord</option>
        </select>
      </div>
      <div class="form-control">
        <div>
          <input
            id="interest-news"
            name="interest"
            type="checkbox"
            value="communication"
          />
          <label for="interest-news">communication</label>
        </div>
        <div>
          <input
            id="interest-tutorials"
            name="interest"
            type="checkbox"
            value="tutorials"
            v-model="interest"
          />
          <label for="interest-tutorials">Tutorials</label>
        </div>
        <div>
          <input
            id="interest-nothing"
            name="interest"
            type="checkbox"
            value="nothing"
            v-model="interest"
          />
          <label for="interest-nothing">Nothing</label>
        </div>
      </div>

      <button>Submit</button>
    </div>
  </form>
</template>

<script>
export default {
  data() {
    return {
      userName: "",
      referrer: "",
      interest: [],
    };
  },
  methods: {
    submitForm() {
      console.log("=========UserName=========");
      console.log("UserName:", this.userName);
      console.log("=========Referrer=========");
      console.log("Referrer:" + this.referrer);
      this.referrer = "wom";
      console.log("=========Checkboxes=========");
      console.log("Checkbox:", this.interest);
    },
  },
};
</script>
olll
```

GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

Task 4 | Advanced Components

Notes

Transitions and Animations

Vue attaches event listeners to know when a transition has ended. It can either be transitioning or animating , depending on the type of CSS rules applied.

VueJS offers a couple of ways to implement transitions:

1. *CSS transition/animation styling*
2. *Javascript hooks to make edits to the DOM*
3. *Integrating 3rd party CSS/JS libraries*

What's the <transition> element
The transition element is a wrapper that helps you add transition functionality to your elements. Essentially, it sets up different hooks and adds classes to your changing elements so we can style them throughout different stages of the transition

Reusability & Composition

Composition
This API allows features to be grouped together logically, rather than having to organize your single-file components by function.

Reusability

Component Inheritance
When you inherit a Component, all logic inside is equally inherited.

Mixins
Like component inheritance, we can use mixins in order to share component logic.

Composing Reusable Components
Another way to reuse functionality is by creating reusable components that just receive props and emit events, which makes it a more explicit solution. Additionally, this solution is more "universal" since the same approach can be applied in any other component-based technology.

Summary
Component inheritance and mixins give us a way to magically separate part of the logic of a component and merge it together, which can be suitable for some cases as long as it doesn't become too confusing. Mixins, in particular, are more powerful since they allow to combine multiple of them.Using component composition is a cleaner and more explicit solution.

WEEK 6 | Vue.js Framework

Reflection

The main focus of this week was learning how to handle user input. Handling user input is a very interesting concept it was something that I was curious about. These tasks showed me how we handle user inputs which is an important skill that will be useful when implementing it in on my own website.

These tasks also allowed me to explore and analyze how the v-model, v- for works which deepened my understanding. Overall, I learnt how important it is to know how to handle user input.

This was very useful, as it allowed me to apply my theory into my practical. By applying my theory skills into this practical, made it easier to implement these concepts in my future projects.

Task 1 | Using V-model for handling user inputs

v-model is a common directive used in almost every Vue application You can use the v-model directive to create two-way data bindings on form elements and works perfectly with input, checkbox, select, textarea and radio.

v-model will ignore the initial value, checked, or selected attributes found on any form elements. It will always treat the Vue instance data as the source of truth.

Overall, the v-model in Vue is used to store data from an element and then can output data into another component.

Screenshot

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Week_6.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.4.4/css/bulma.min.css">
</head>

<body>

    <div class="columns" id="app">
        <div class="column is-two-thirds">
          <section class="section">
            <h1 class="title">Contact Us </h1>
            <p class="subtitle">
              Soul Divine Always There When You Need Us  <strong>Always!</strong>!
            </p>
            <hr>

            <!-- form starts here -->
            <section class="form">
              <form v-on:submit.prevent="$validator.validateAll(); console.log(form);">
                <div class="field">
                  <label class="label">Name</label>
                  <div class="control">
                    <input name="name" v-model="form.name" v-validate="'required|min:3'" v-bind:class="{'is-danger': errors.has('name')}" class="input" type="text" placeholder="
                  </div>
                  <p class="help is-danger" v-show="errors.has('name')">
                    {{ errors.first('name') }}
                  </p>
                </div>

                <div class="field">
                  <label class="label">Message</label>
                  <div class="control">
                    <textarea name="message" class="textarea" v-validate="'required|polite'" placeholder="How Can We Help You..." v-bind:class="{'is-danger': errors.has('message
                  </div>
                  <p class="help is-danger" v-show="errors.has('message')">
                    {{ errors.first('message') }}
                  </p>
                </div>

                <div class="field">
                  <label class="label">Inquiry Type</label>
                  <div class="control">
                    <div class="select">
                      <select v-model="form.inquiry_type">
                                                <option disabled value="">Nothing Selected</option>
                                                <option v-for="option in options.inquiry" v-bind:value="option.value">
                                                    {{ option.text }}
                                                </option>
                                            </select>
                    </div>
                  </div>
                </div>

                <div class="field">
                  <label class="label">LogRocket Usecases</label>
                  <div class="control">
                    <div class="select is-multiple">
```

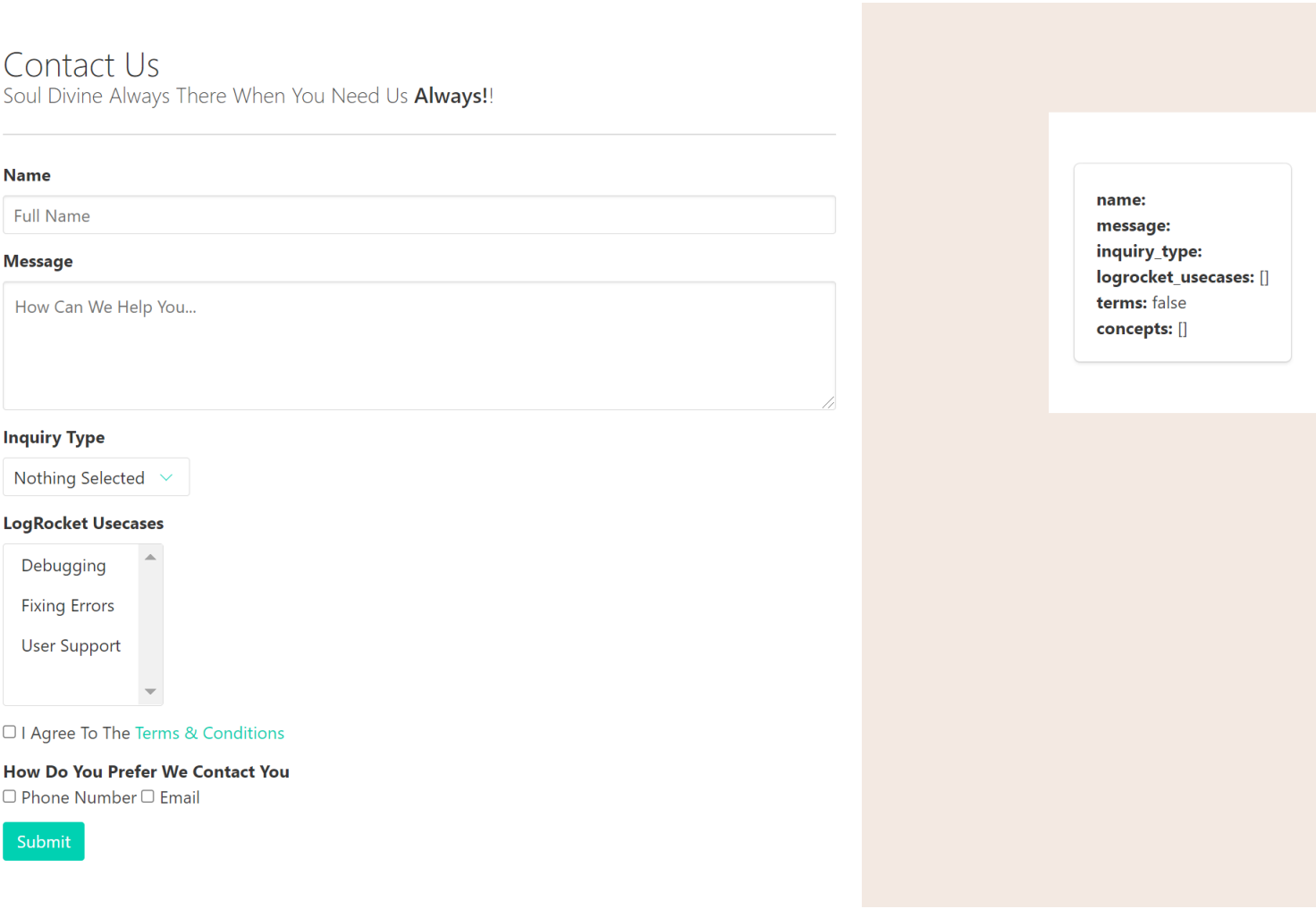
Input HTML

```
                                    <option>Debugging</option>
                                    <option>Fixing Errors</option>
                                    <option>User Support</option>
                                  </select>
                    </div>
                  </div>
                </div>

                <div class="field">
                  <div class="control">
                    <label class="checkbox">
                                    <input type="checkbox" v-model="form.terms">
                                    I Agree To The <a href="#">Terms & Conditions</a>
                                  </label>
                  </div>
                </div>

                <div class="field">
                  <label>
                                  <strong>How Do You Prefer We Contact You  </strong>
                                </label>
                  <div class="control">
                    <label class="checkbox">
                                    <input type="checkbox" v-model="form.concepts"
                                    value="Phone Number">
                                    Phone Number
                                  </label>
                    <label class="checkbox">
                                    <input type="checkbox" v-model="form.concepts"
                                    value="Email">
                                    Email
                                  </label>
                  </div>
                </div>



                <div class="field is-grouped">
                  <div class="control">
                    <button v-bind:disabled="errors.any()" class="button is-primary">
                                  Submit
                                </button>
                  </div>
                </div>

              </form>
            </section>
          </section>
        </div>

        <div class="column">
          <section class="section" id="results">
            <div class="box">
              <ul>
                <li v-for="(item, k) in form">
                  <strong>{{ k }}:</strong> {{ item }}</li>
              </ul>

            </div>
          </section>
        </div>
      </div>
```

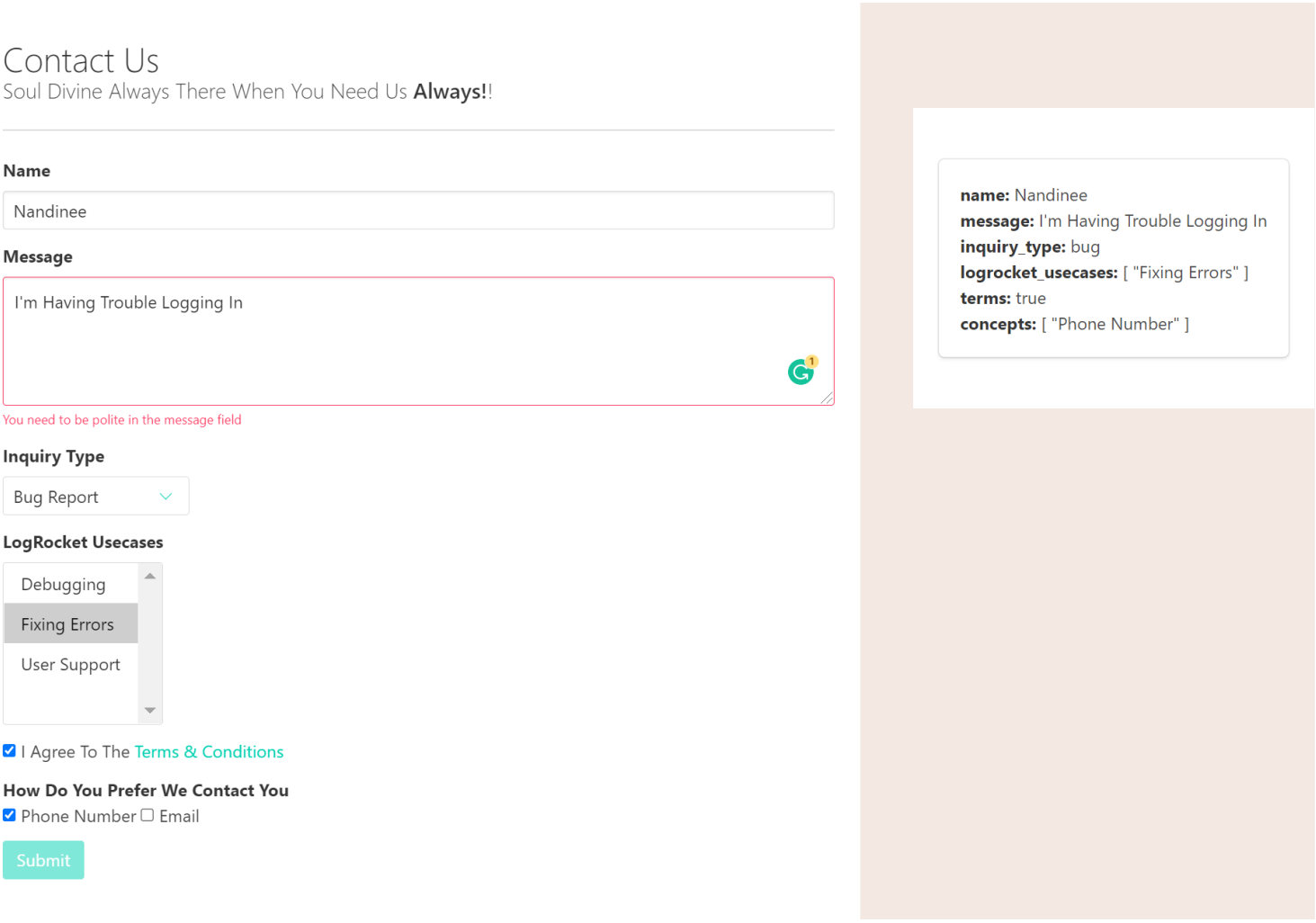
Input JS

```
Vue.use(VeeValidate);
VeeValidate.Validator.extend("polite", {
  getMessage: field => `You need to be polite in the ${field} field`,
  validate: value => value.toLowerCase().indexOf("please") !== -1
});
new Vue({
  el: "#app",
  data: {
    form: {
      name: "",
      message: "",
      inquiry_type: "",
      logrocket_usecases: [],
      terms: false,
      concepts: [],
    },
    options: {
      inquiry: [
        { value: "feature", text: "Feature Request" },
        { value: "bug", text: "Bug Report" },
        { value: "support", text: "Support" }
      ]
    }
  }
});
```

Output With No User Input



Output With User Input



Task 2 | Checkbox in your project

Notes

Checkboxes are very useful in surveys and contact forms it allows us to gain user input on a certain topic or issue. Checkboxes are also used to present the user with a range of options, from which the user may select any number of options.

Task 3 | Dynamic options rendering v-for

Notes

Dynamic options rendering in Vue can display options in a menu and when the user selects an option a verification of the option selected is produced

Screenshots Task 2-3

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Tasks.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.4.4/css/bulma.min.css">
</head>

<body>

    <h1><i><b>Week 6 Task 2</i></b></h1>

    <h2><i>Checkbox</i></h2>

    <h3> Select Your Prefered Doctor </h3>

    <div id="v-model-multiple-checkboxes" class="demo">
        <input type="checkbox" id="jack" value="Dr. Mark Hyman" v-model="checkedNames" />
        <label for="jack"> Dr. Mark Hyman </label>
        <input type="checkbox" id="john" value="Dr. Andrew Weil" v-model="checkedNames" />
        <label for="john"> Dr. Andrew Weil </label>
        <input type="checkbox" id="mike" value="Dr. Robert Grant" v-model="checkedNames" />
        <label for="mike"> Dr. Robert Grant </label>
        <br />
        <span>Checked names: {{ checkedNames }}</span>
      </div>


      <br>

    <h1><i><b>Week 6 Task 3</i></b></h1>

    <h2><i>Dynamic Options Rendering</i></h2>

    <h3> Select Your Prefered Time </h3>

    <div id="v-model-select-dynamic" class="Trial_1">
        <select v-model="selected">
          <option v-for="option in options" v-bind:value="option.value">
            {{ option.text }}
          </option>
        </select>
        <span>Selected: {{ selected }}</span>
      </div>









    <script src=" https://unpkg.com/vue@next"></script>

    <script src="Tasks.js"></script>

</body>

</html>
```

Input JS

```
Vue.createApp({
    data() {
      return {
        checkedNames: []
      }
    }
  }).mount('#v-model-multiple-checkboxes')


  Vue.createApp({
    data() {
      return {
        selected: 'Nothing',
        options: [
          { text: '11', value: '11 AM' },
          { text: '12', value: '12 PM' },
          { text: '02', value: '2 PM' },
          { text: '03', value: '3 PM' },
          { text: '05', value: '5 PM' }
        ]
      }
    }
  }).mount('#v-model-select-dynamic')
```

Output With No User Input

Week 6 Task 2

Checkbox

Select Your Prefered Doctor

☐ Dr. Mark Hyman ☐ Dr. Andrew Weil ☐ Dr. Robert Grant

Checked names: []

Week 6 Task 3

Dynamic Options Rendering

Select Your Prefered Time

Selected: Nothing

Output With User Input

Week 6 Task 2

Checkbox

Select Your Prefered Doctor

☑ Dr. Mark Hyman ☑ Dr. Andrew Weil ☐ Dr. Robert Grant

Checked names: ["Dr. Mark Hyman", "Dr. Andrew Weil"]

Week 6 Task 3

Dynamic Options Rendering

Select Your Prefered Time

12 Selected: 12 PM

Task 4 | Using Modifiers in your project

Notes

Modifiers

Modifiers are used for binding a directive in a special way.

.stop runs event.stopPropagation() before the rest of the event handler code is run.

.capture lets us capture the event. That is, when we run the event handler in an inner element, then the same event handler will also run in the outside elements.

.self will only trigger the event handler if the event.target is the element itself.

.once will trigger an event handler at most once.

Model Modifiers

v-*model Has Modifiers*

- . lazy
- . number
- . trim

The .lazy modifier will make the model sync after the change event instead of the input event.

The .number modifier will automatically cast whatever is inputted in an input to a number.

The .trim modifier will trim whitespace from whatever is inputted.

GitHub Link: https://github.com/Nandoss07/SIT120-Portfolio-

WEEK 7 | Registration, Props, Custom Events, Dynamic/Async

Reflection

This week we learned about local and global registration, custom event and dynamic/async. this week was basically an in depth and hands on version of task 4 in week 3. this helped me to gain a better perspective of these concepts and making it easier to implement these concepts in my future projects I enjoy these hands-on activities because this way I'm able to apply more learning and also check my understanding.

Task 1 | Local and Global Registration

Notes

Global registration in Vue is used for components that are used multiple times once the components are registered, they then can be used anywhere in the web page

Screenshot

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Tasks_7.css">
</head>


<body>


    <h1><i><b>Week 7 Task 1</i></b></h1>

    <div id="app"></div>



    <script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.10/vue.min.js"></script>

    <script src="Tasks_7.js"></script>

    </body>

    </html>
```

Input JS

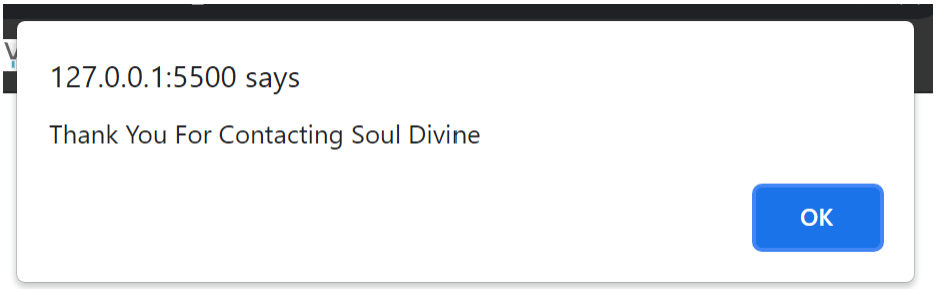
```
const PromptComponent = {
    template:'<div><p>${message}</p><button @click="sayHi">Submit Form </button></div>',
    delimiters: ['${','}'],
    data: function() {
      return {
        message: 'Contact Form '
      }
    },
    methods: {
      sayHi: function(){
        alert('Thank You For Contacting Soul Divine ');
      }
    }
  };
  let vm = new Vue({
    el:'#app',
    components: {
      'prompt-component':PromptComponent
    },
    data: {
      num: 1
    },
    template: `<prompt-component/>`
  });
```

Output

Week 7 Task 1

Contact Form

Submit Form



Task 2 | Coding Props

Screenshot

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Task_7_2.css">
</head>


<body>


    <h1><i><b>Week 7 Task 2</i></b></h1>

    <div id="app">
        <child :text="message"></child>
    </div>



    <script src="https://cdnjs.cloudflare.com/ajax/libs/vue/2.1.10/vue.min.js"></script>

    <script src="Task_7_2.js"></script>

    </body>

    </html>
```

Input JS

```
Vue.component('child',{
    props: {
      text: {
        type: String,
        required: true
      }
    },
    template: `<div>{{ text }}</div>`
  });

  new Vue({
    el: '#app',
    data() {
      return {
        message: 'Welcome To Soul Divine Mr David Lin'
      }
    }
  })
```

Output

Week 7 Task 2

Welcome To Soul Divine Mr David Lin

Task 3 | Custom Events

Screenshot

Input HTML

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Task_7_3.css">
</head>


<body>


    <h1><i><b>Week 7 Task 3</i></b></h1>

    <h2><i> Enter Your Full Name  </i></h2>

    <div id="v-model-example" class="Trial_8">
        <p>First Name: {{ firstName }}</p>
        <p>Last Name: {{ lastName }}</p>
        <user-name
         v-model:first-name="firstName"
         v-model:last-name="lastName"
        ></user-name>
     </div>



    <script src="https://unpkg.com/vue@next"></script>


    <script src="Task_7_3.js"></script>

    </body>

    </html>
```

Input JS

```
const UserName = {
    props: {
      firstName: String,
      lastName: String
    },
    template: `
      <input
        type="text"
        :value="firstName"
        @input="$emit('update:firstName', $event.target.value)">

      <input
        type="text"
        :value="lastName"
        @input="$emit('update:lastName', $event.target.value)">
    `
  };

  const HelloVueApp = {
    components: {
      UserName,
    },
    data() {
      return {
        firstName: 'John',
        lastName: 'Doe',
      };
    },
  };

  Vue.createApp(HelloVueApp).mount('#v-model-example')
```

Input CSS

```
.Trial_8 {
    font-family: sans-serif;
    border: 1px solid #eee;
    border-radius: 2px;
    padding: 20px 30px;
    margin-top: 1em;
    margin-bottom: 40px;
    user-select: none;
    overflow-x: auto;
  }
```

Output

Week 7 Task 3

Enter Your Full Name

First Name: Nandinee

Last Name: Pardasani

Nandinee	Pardasani

Task 4 | Vue Slots

Notes

Slots are reserved space offered by Vuejs to display content passed down from one component to another.

Screenshot

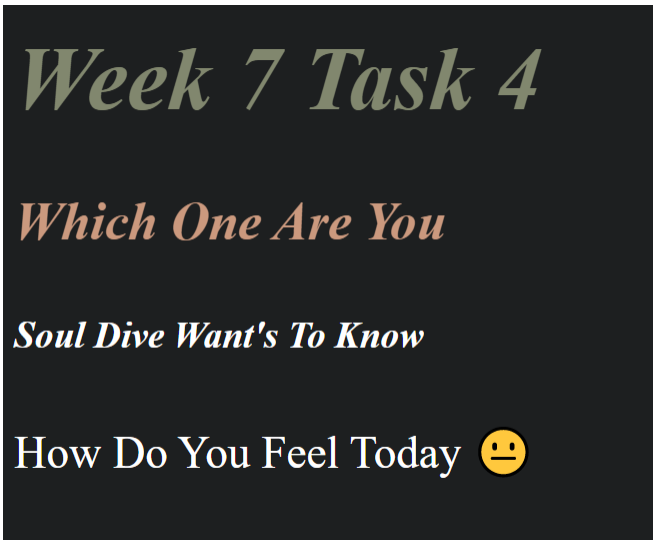
Input HTML

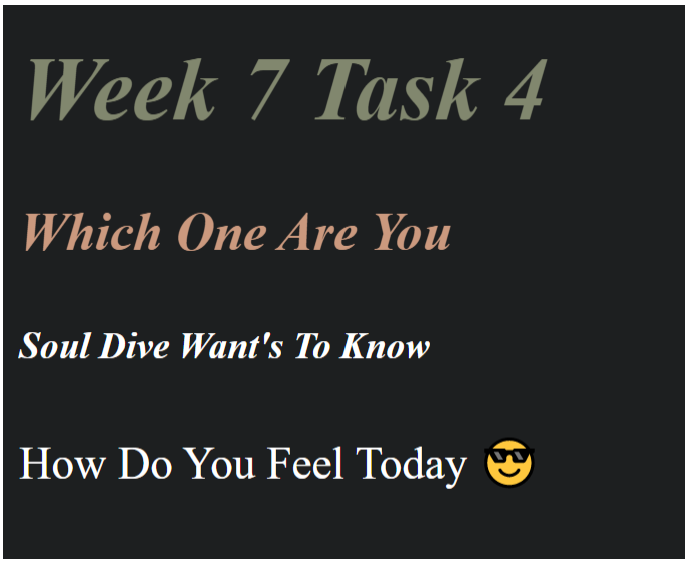
```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="Task_4.css">
</head>


<body>


    <h1><i><b>Week 7 Task 4</i></b></h1>


    <h3><i> Which One Are You</i></h3>

    <h5><i> Soul Dive Want's To Know </i></h5>

    How Do You Feel Today <my-element cool></my-element>


    <script src="https://unpkg.com/vue@next"></script>


    <script src="Task_4.js"></script>

    </body>

    </html>
```

Input JS

```
customElements.define(
  "my-element",
  class extends HTMLElement {
    constructor() {
      super();
    }

    connectedCallback() {
      this.style.cursor = "pointer";
      this.style.userSelect = "none";
      this.style.webkitUserSelect = "none";
      this.style.MozUserSelect = "none";
      this.render();

      this.addEventListener("click", this.onClick);
    }

    disconnectedCallback() {
      this.removeEventListener("click", this.onClick);
    }

    /**
     * Render the content. Will render a
     * cool face if the `cool` attribute
     * is set and a neutral face otherwise
     */
    render() {
      this.innerHTML = this.cool ? "&#x1f60e" : "&#128528;";
    }

    /**
     * Click handler. Toggles the `cool`
     * property.
     */
    onClick() {
      this.cool = !this.cool;
    }

    static get observedAttributes() {
      return ["cool"];
    }

    attributeChangedCallback() {
      this.render();
    }

    get cool() {
      return this.hasAttribute("cool");
    }

    set cool(value) {
      if (!!value) {
        this.setAttribute("cool", "");
      } else {
        this.removeAttribute("cool");
      }
    }
  }
);
```

Output





GitHub Link: ***https://github.com/Nandoss07/SIT120-Portfolio-***