Assignment 2

Practical Portfolio

ART PORTFOLIO

SIT120 – Introduction To Responsive Web Apps

Marking Justification:

For this assigned assignment, I aim to obtain a Credit score or higher from this proposed portfolio. (All tasks from week 1-7 have been completed and is ready for assessment)

Week 1 – Introduction to HTML, CSS

Notes

This first week is an introduction to HTML and CSS - I personally, haven't gained much experience from these languages as this is a first learning curve in web development. I am eager to grasp as much knowledge and skills from this unit as I've always peaked interest in learning how to develop a site.

Task 1 – Create an HTML page

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
  </head>
  <body>
    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Create an HTML Page</h2>

    <p>HTML Home Page/Index page</p>
  </body>
</html>
```

Output (index.html):

Task 2 – Add CSS to your HTML page

Input (index.html):

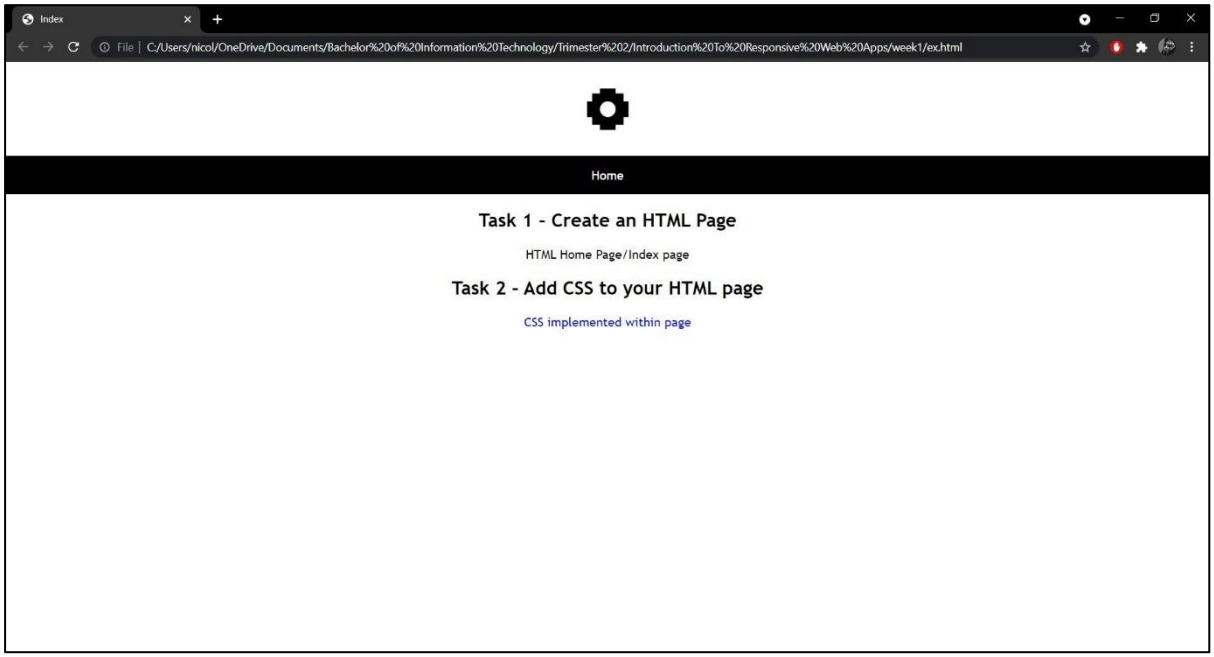
```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul class="nav"> <!-- naviagtion list -->
      <li class="page"><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Create an HTML Page</h2>

    <p>HTML Home Page/Index page</p>

    <h2>Task 2 – Add CSS to your HTML page</h2>

    <p class="text">CSS implemented within page</p>
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

.nav { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

.page { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

p { /* p style */
    text-align: center;
}
```

```
.text{
  text-align: center;
  color: blue;
}
```

Output (index.html):



Task 3 – Adding JavaScript

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul class="nav"> <!-- naviagtion list -->
      <li class="page"><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 - Create an HTML Page</h2>
```

```
    <p>HTML Home Page/Index page</p>

    <h2>Task 2 – Add CSS to your HTML page</h2>

    <p class="text">CSS implemented within page</p>

    <h2>Task 3 - Adding JavaScript</h2>

    <div class="ex">
      <button onclick="myFunction()">Click me</button>
    </div>

    <script>
      function myFunction() {
        alert("Welcome to my HTML page!");
      }
    </script>
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

.nav { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

.page { /* list item style */
    display:inline;
}

a { /* hyperlink style */
```

```
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.text{
  text-align: center;
  color: blue;
}
```

Output (index.html):

Task 4 – Vue.js Framework

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul class="nav"> <!-- naviagtion list -->
      <li class="page"><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Create an HTML Page</h2>

    <p>HTML Home Page/Index page</p>

    <h2>Task 2 – Add CSS to your HTML page</h2>

    <p class="text">CSS implemented within page</p>

    <h2>Task 3 - Adding JavaScript</h2>
```

```
    <div class="ex">
      <button onclick="myFunction()">Click me</button>
    </div>

    <script>
      function myFunction() {
        alert("Welcome to my HTML page!");
      }
    </script>

    <h2>Task 4 – Vue.js Framework</h2>

    <div class="ex" id="example4-1">
      <ul>
        <li v-for="item in items">{{ item }}</li>
      </ul>

    <input id="input" type="text" v-model="newItem">

    <button v-on:click="addItem">Add to do item</button>
    </div>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example4-2">
      {{ message }}
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example4-3">
      <span v-bind:title="message">Date and time based on visit (hover-on)</span>
    </div>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (vue.js):

```
var ex0 = new Vue({

    el:'#example4-1',
    data: {
      items:[]
    },
```

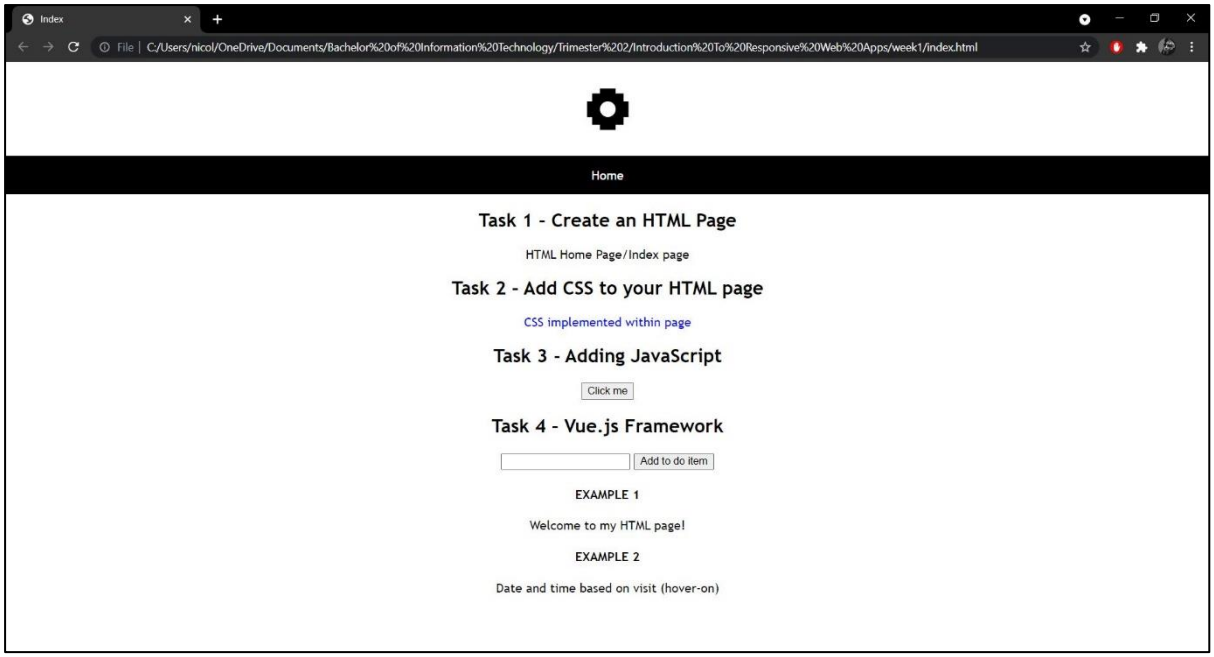
```
  methods:{

    addItem() {
      this.items.push(this.newItem);
      this.newItem='';
    }
  }
  })

  var ex1 = new Vue({
    el: '#example4-2',
    data: {
      message: 'Welcome to my HTML page!'
    }
  })

  var ex2 = new Vue({
    el: '#example4-3',
    data: {
      message: 'You loaded this page on ' + new Date().toLocaleString()
    }
  })
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

.nav { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

.page { /* list item style */
    display:inline;
```

```
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.text{
  text-align: center;
  color: blue;
}

.check-list {
  margin-right: 0.5rem;
}
```

Output (index.html):



Learning Reflection

For this week's tasks we covered to gain experience in learning how to implement HTML, CSS, and JavaScript within our web applications, understanding the knowledge of basic programming methods was able to cast skills for future development tasks.

Week 2 – Responsive web apps design, advanced HTML, CSS, and JavaScript

Notes

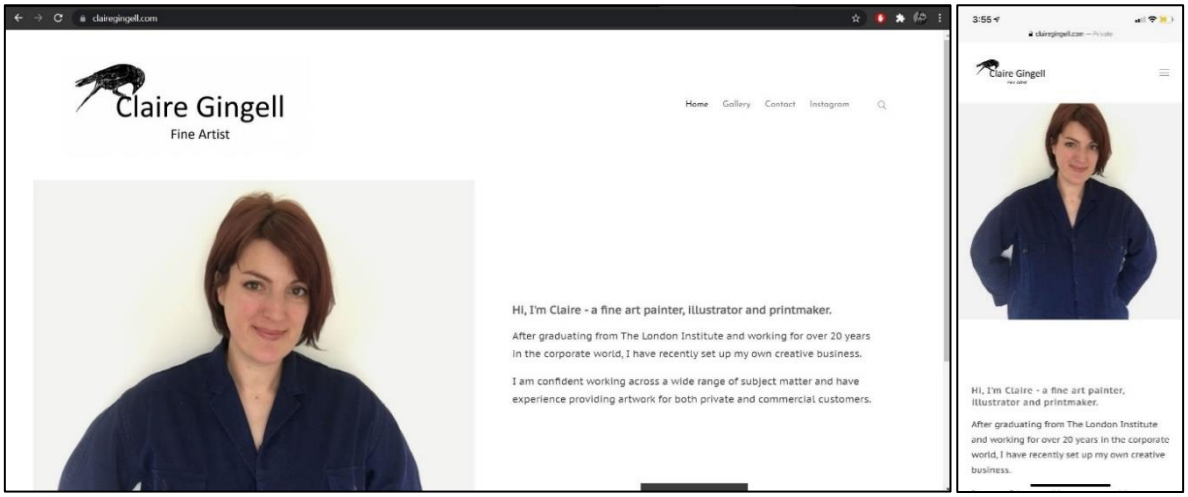
This week's tasks were based upon practicing more on HTML, CSS, and JavaScript. We'd also have to look at UI/UX designing and user stories – as this will be a part of our proposed upcoming project. I'd grasp last week's activities and implement them through here as this week is relatively connected to this weeks tasks.

Task 1 – Understanding about responsive web Pages and apps

SITE 1

The site shown within the link "SITE 1" - shares an identical responsive web design when compared between the mobile and desktop. This site contains a set viewport meta which scales the size of the website through different devices, shown below is an image that is taken on an iPhone XR. As you can see the website is positioned to fit within the screen of the device along with other sections such as the images, paragraphs, and menu.

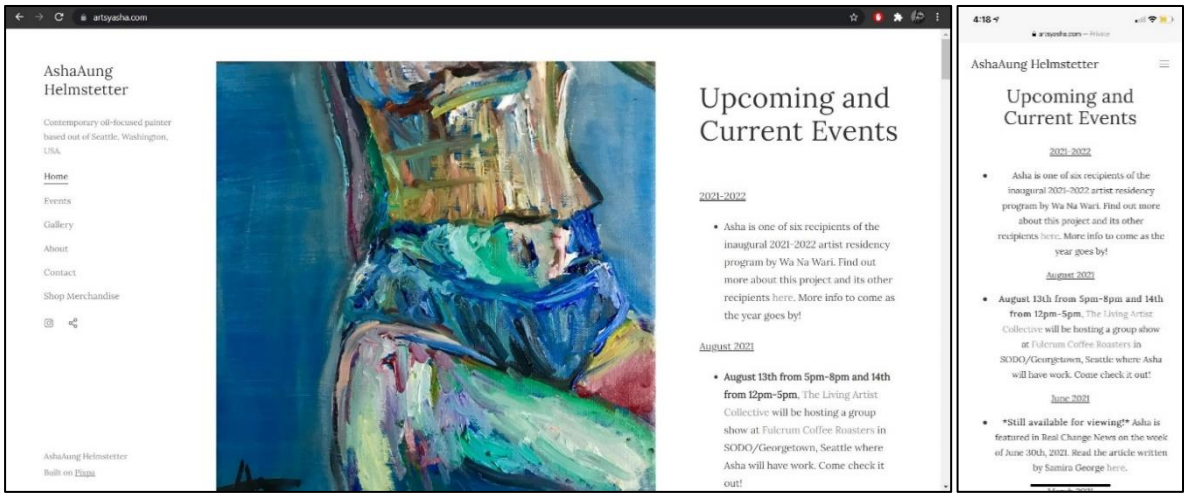
https://www.clairegingell.com/



SITE 2

The site shown within the link "SITE 2" - is somewhat differential when compared to both mobile and desktop devices. As this responsive web design is also set as a viewport meta, sections within the mobile device seem to position in a clustered form due to scaling and sizing although all sections manage to fit within the spaces of both devices.

https://www.artsyasha.com/



Task 2 – CSS and Improve your web page

As shown in this page, I have implemented CSS to improve the content through using styled elements. These include text aligning, padding, margin, colour, and styled fonts. Aspects implemented within my responsive page have been focused to apply structure for incoming content that'll be proposed for this site. Currently, I've chosen to include a title along with a logo with a functional navigation bar in the top section of the page. Within the centre section I have currently included minimal content such as images and text, although down the line this will be frequently updated along the way.

Task 3 – User stories and UI/UX design for your project

USER STORY 1

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user, I want the app to notify me on the latest updates made within the website.	*1. Once user opens app, a prompt will display giving an option whether to turn notifications on or off.* *2. User can manually enable or disable notifications through their Settings on the device.*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 8***	***Priority: 1*** *High Priority*

USER STORY 2

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user, I want to be able to download images within the application.	*1. Once user selects photo.* *2. User can touch down on the screen giving an option to Save Image.* *3. User should download image into their gallery.*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 8***	***Priority: 2*** *Medium Priority*

USER STORY 3

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user, I want the app to be able to have a constant functional look and interaction throughout all platforms.	*1. Application should be constant for all platforms and devices.*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 5***	***Priority: 1*** *High Priority*

USER STORY 4

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user I want there to be a search functionality where I can search for individual paintings from using their title name.	*1. Within each web page there will be a search box.* *2. User can input existing title name of painting they desire to look for.* *3. User will be prompt to a new page to view painting*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 5***	***Priority: 1*** *High Priority*

USER STORY 5

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user I want there to be a function that will allow me to save/add/remove/edit images within the webpage.	*1. Within the home page a user can select an image that they wish to modify* *2. By selecting the options below each image the user may modify from the*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 8***	***Priority: 2*** *Medium Priority*

	provided options given		

USER STORY 6

Statement	***Acceptance Criteria***	***Estimation***	***Priority***
As a user I want to be subscribed to receive a notification to my email on any newsletters related to this site.	*1. Within the contact page there will be a check box* *2. Ticking the check box will enable this function to keep users up to date through sending updates via email*	*According to the planning poker estimate approach this can be considered as Story* ***Points: 5***	***Priority: 1*** *High Priority*

Task 4 – Use graphics, media, and API's

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul class="nav"> <!-- naviagtion list -->
      <li class="page"><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 4 - Use graphics, media and API's </h2>

    <h4>EXAMPLE 1</h4>

    <iframe class="vid" width="420" height="315" src="https://www.youtube.com/embed/4ZDpgaRUedY"></ifr
ame>

    <h4>EXAMPLE 2</h4>
```

```
    <p>Click the button to get geolocation.</p>

    <div class="ex">
        <button onclick="getLocation()">Click me</button>
    </div>

    <p id="example4-2"></p>

    <script>
    var x = document.getElementById("example4-2");

    function getLocation() {
      if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showPosition);
      } else {
        x.innerHTML = "Geolocation is not supported by this browser.";
      }
    }

    function showPosition(position) {
      x.innerHTML = "Latitude: " + position.coords.latitude +
      "<br>Longitude: " + position.coords.longitude;
    }
    </script>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

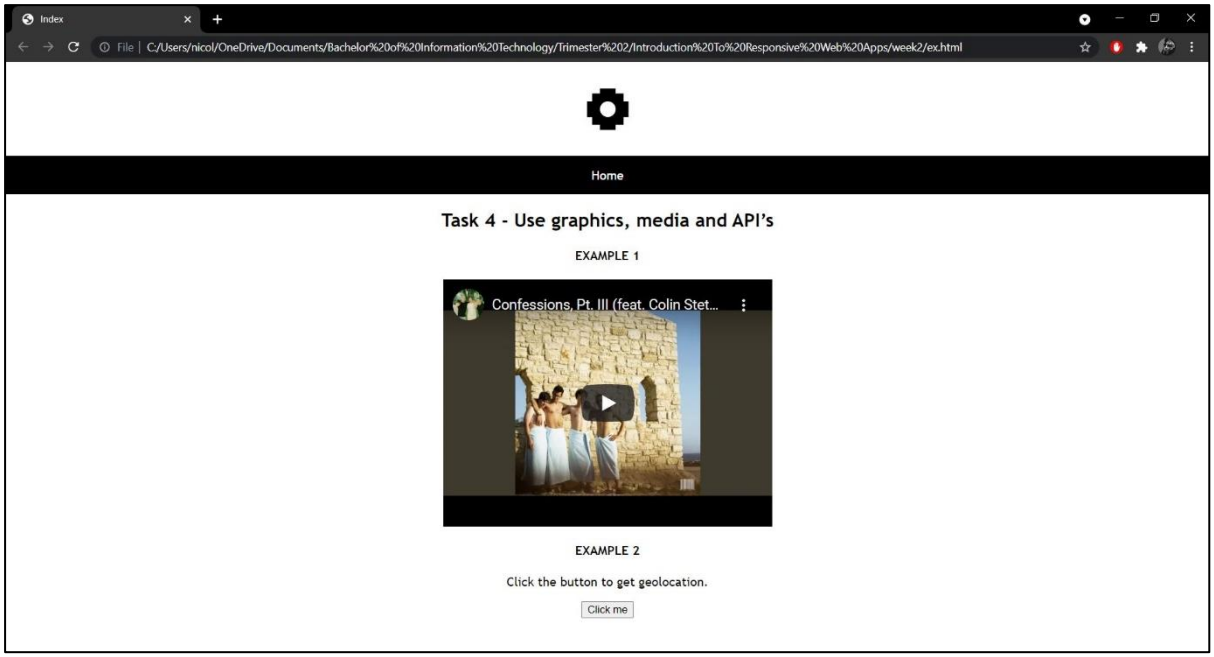
Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

.nav { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
```

```
    margin:0;
    padding:0;
    text-align:center;
}

.page { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.vid {
  display: block;
  border: none;
```

```
  margin-left: auto;
  margin-right: auto;
}
```

Output (index.html):



Learning Reflection

For this week's tasks we've covered in improving our responsive web page as we implemented both CSS and JavaScript. I have contributed to apply minimal changes for my proposed project, these include advanced concepts of HTML, CSS, and JavaScript. To conclude it all, I've had practice with incorporating user stories and UI/UX design, as this will be included to partake the proposed requirements for my upcoming project.

Week 3 – JavaScript Functions

Notes

This week we are learning concepts on JavaScript functions, as this is a high priority for user interactions implemented for our web sites. We can develop all sorts of functionalities that can turn a web application into a responsive site.

Task 1 – JavaScript String Methods

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – JavaScript String Methods</h2>

    <h4>EXAMPLE 1</h4>
    <p>This is an example paragraph that will return the total length of strings written within.</p>

    <p id="example1-1"></p>

    <h4>EXAMPLE 2</h4>

    <div class="ex">
        <button type="button" onclick="document.getElementById('example1-
2').innerHTML = Date()"> Display Date and Time Button</button>
    </div>

    <script src="scripts.js"></script> <!-- linked to scripts.js -->
  </body>
</html>
```

Input (style.css):

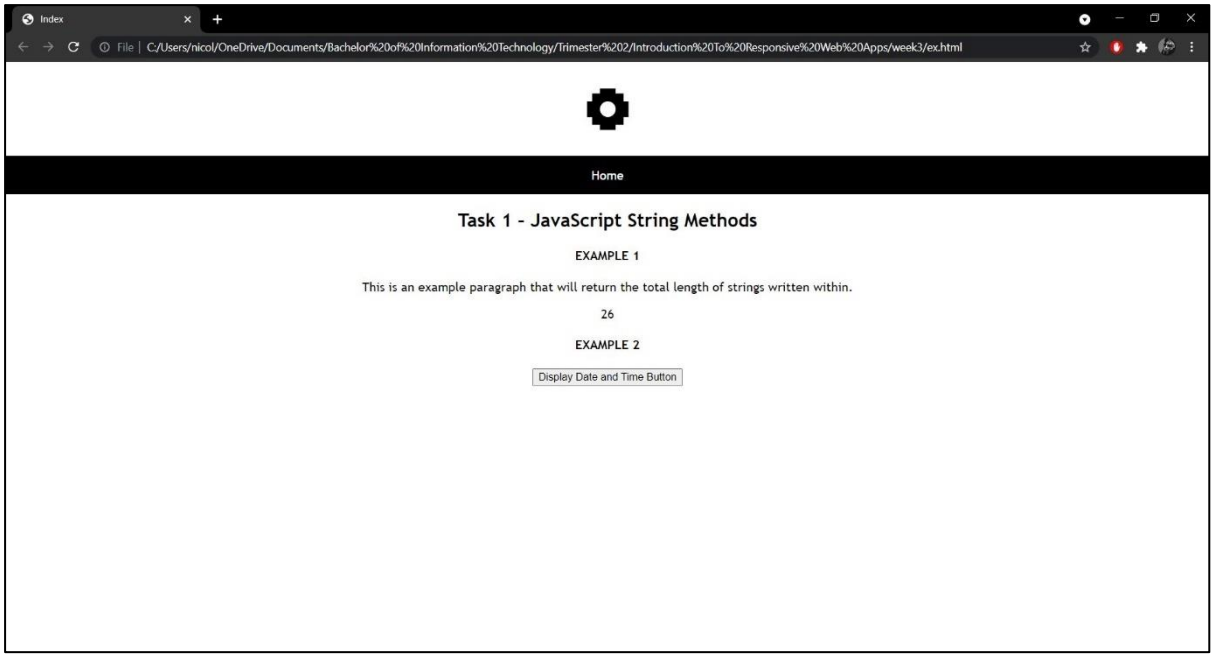
```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}
```

```css
ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (scripts.js):

```
let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
document.getElementById("example1-1").innerHTML = text.length;
```

Output (index.html):



Task 2 – JavaScript Number and Array Methods

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – JavaScript String Methods</h2>

    <h4>EXAMPLE 1</h4>
    <p>This is an example paragraph that will return the total length of strings written within.</p>
```

```
    <p id="example1-1"></p>

    <h4>EXAMPLE 2</h4>

    <div class="ex">
        <button type="button" onclick="document.getElementById('example1-
2').innerHTML = Date()"> Display Date and Time Button</button>
    </div>

    <p id="example1-2"></p>

    <h2>Task 2 – JavaScript Number and Array Methods</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex">
        <button onclick="myFunction()">Button to display the formatted numbers</button>
    </div>

    <p id="example2-1"></p>

    <h4>EXAMPLE 2</h4>
    <div class="ex">
      <script type="text/javascript">
      var num=12;
      document.write("Output : " + num.toString(2));
      </script>
    </div>

    <script src="scripts.js"></script> <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

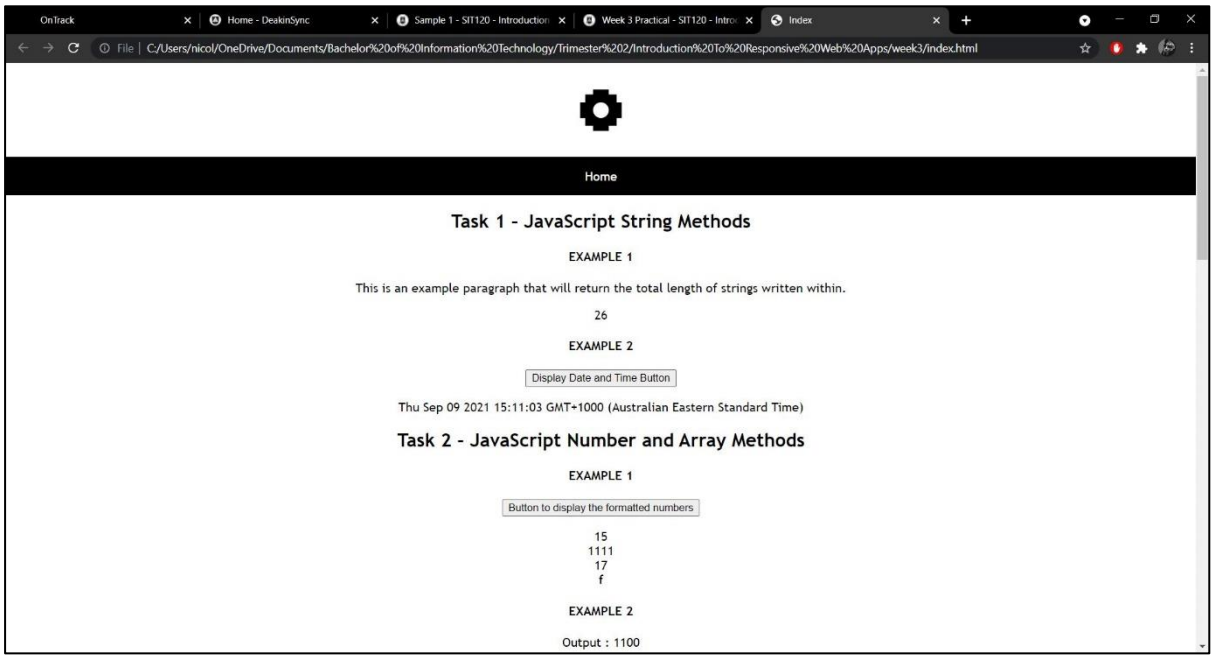
```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
```

```
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (scripts.js):

```
let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
document.getElementById("example1-1").innerHTML = text.length;

function myFunction() {
  var num = 15;
  var a = num.toString();
  var b = num.toString(2);
  var c = num.toString(8);
  var d = num.toString(16);

  var n = a + "<br>" + b + "<br>" + c + "<br>" + d;

document.getElementById("example2-1").innerHTML=n;
}

const element = document.getElementById("intro");
```

Output (index.html):



Task 3 – JavaScript Get/Set Methods

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
```

```
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – JavaScript String Methods</h2>

    <h4>EXAMPLE 1</h4>
    <p>This is an example paragraph that will return the total length of strings written within.</p>

    <p id="example1-1"></p>

    <h4>EXAMPLE 2</h4>

    <div class="ex">
        <button type="button" onclick="document.getElementById('example1-
2').innerHTML = Date()"> Display Date and Time Button</button>
    </div>

    <p id="example1-2"></p>

    <h2>Task 2 – JavaScript Number and Array Methods</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex">
        <button onclick="myFunction()">Button to display the formatted numbers</button>
    </div>

    <p id="example2-1"></p>

    <h4>EXAMPLE 2</h4>
    <div class="ex">
      <script type="text/javascript">
      var num=12;
      document.write("Output : " + num.toString(2));
      </script>
    </div>

    <h2>Task 3 – JavaScript Get/Set Methods</h2>

    <h4>EXAMPLE 1</h4>
```

```
    <p id="intro">"example element"</p>

    <p id="example3-1"></p>

    <h4>EXAMPLE 2</h4>

    <p>Number of letters from this array [a,b,c]:</p>

    <p id="example3-2"></p>

    <script src="scripts.js"></script> <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}
```

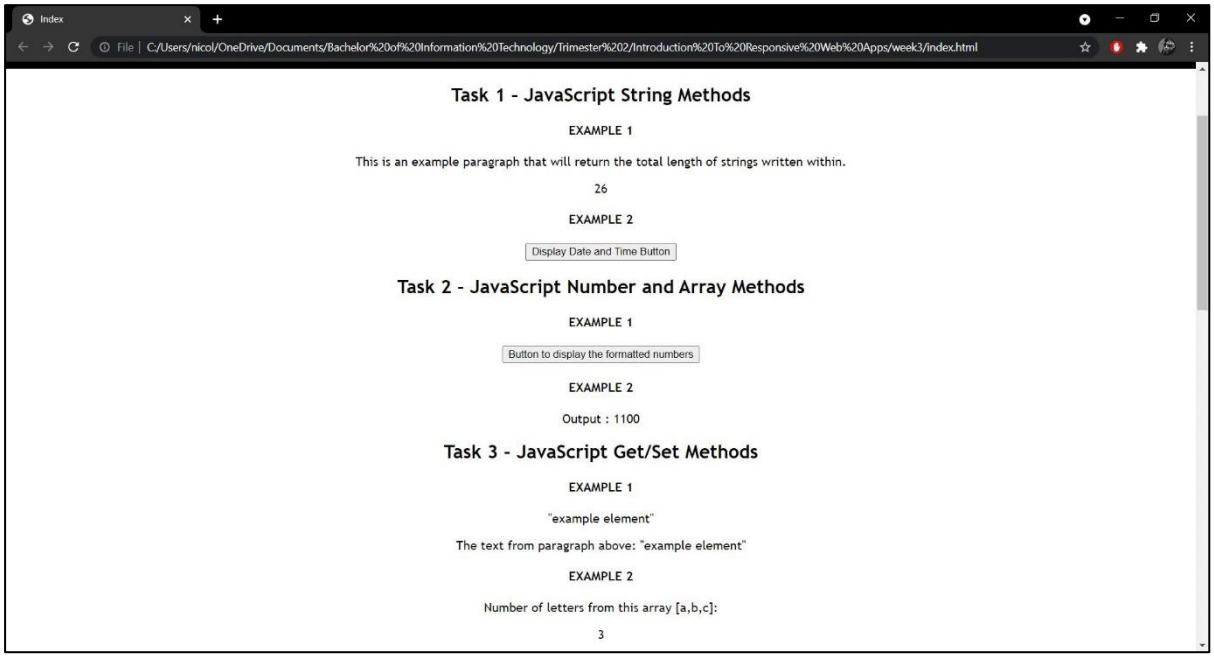
```
img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (scripts.js):

```
let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
document.getElementById("example1-1").innerHTML = text.length;

function myFunction() {
  var num = 15;
  var a = num.toString();
  var b = num.toString(2);
  var c = num.toString(8);
  var d = num.toString(16);

  var n = a + "<br>" + b + "<br>" + c + "<br>" + d;

  document.getElementById("example2-1").innerHTML=n;
}

const element = document.getElementById("intro");
```

```
document.getElementById("example3-1").innerHTML =
"The text from paragraph above: " + element.innerHTML;

const letters = new Set(["a","b","c"]);
document.getElementById("example3-2").innerHTML = letters.size;
```

Output (index.html):



Task 4 – Get started with some new concepts/components

Computed Properties and Watchers:

Computed properties are used to develop reactive props – refers to making changes to properties when reacted, whereas computed watchers can only call these properties/props one at a time.

Class and Style Bindings:

Both class and style bindings are used to control an elements class, this includes an element such as lists along with inline styles. Both are considered as attributes which refers that they can be handles through using a v-bind command.

Conditional Rendering:

Conditional rendering is referred to determine whether a condition is true or false - as a directive v-if command is used to render an expression of a user input. An else block can also be included to return the expression condition.

List Rendering:

List rendering is referred to render a selected list of items that are from an array, a v-for directive command is used to render user inputted items from the array.

Event Handling:

Event handling is used to manipulate and interpret elements that are contained within a DOM through using a v-on command.

Form Input Bindings:

Form input bindings are used to create different bindings on a form that can develop inputted fields for users to update, using a v-model directive can object this.

Components Basics:

Component basics are used to accustom Vue instances along with HTML elements that are implemented within custom variables.

Learning Reflection

For this week's tasks we've covered the standard basics in learning different concepts of JavaScript. I've obtained a grasp on how to implement this programming language within my website, along with understanding the definitions of concepts/components from Vue. This week's activities were mainly focused upon learning java, as I have understood the overview of all 4 tasks.

Week 4 – Vue.js Framework (part 1)

Notes

This week we are grasping to learn further concepts of Vue.js as we are now implementing Vue within our website.

Task 1 – Declaring Rendering

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 - Declarative Rendering</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example1-1">
      {{ message }}
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example1-2">
      <p>{{ message }}</p>
      <input v-model="message">
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}
```

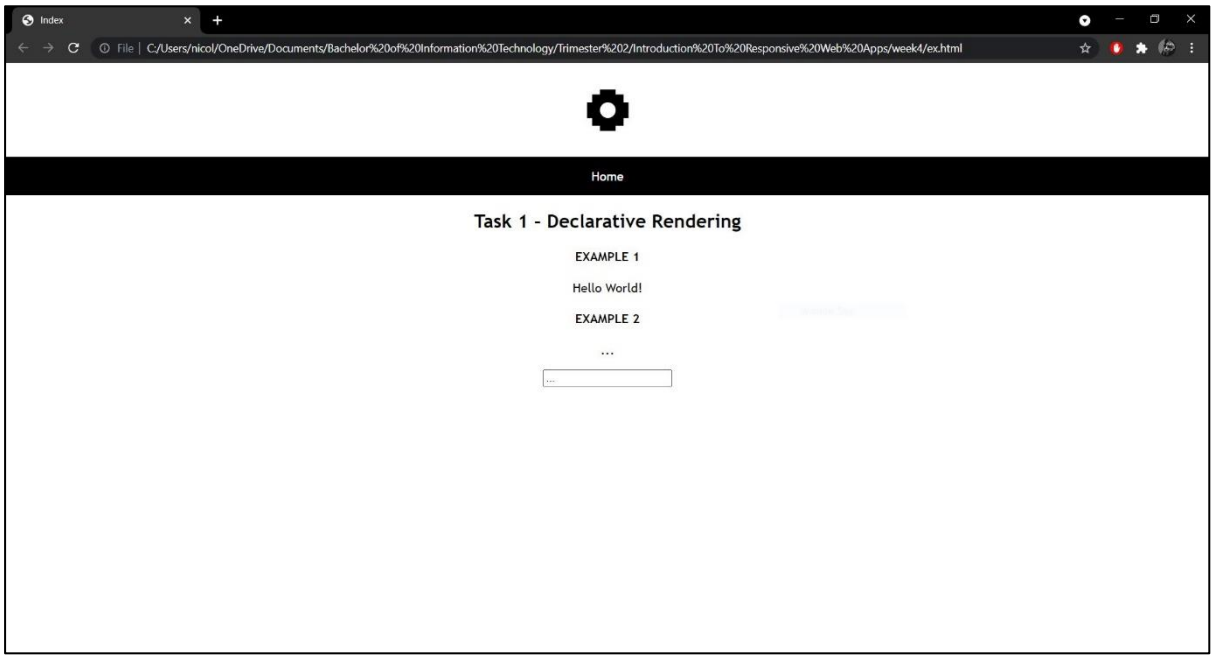
```
ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
```

```
}
```

Input (vue.js):

```
var ex0 = new Vue({
  el: '#example1-1',
  data: {
    message: 'Hello World!'
  }
})

var ex1 = new Vue({
  el: '#example1-2',
  data: {
    message: '...'
  }
})
```

Output (index.html):



Task 2 – Conditional and Loops

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
```

```
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Declarative Rendering</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example1-1">
      {{ message }}
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example1-2">
      <p>{{ message }}</p>
      <input v-model="message">
    </div>

    <h2>Task 2 - Conditionals and Loops</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example2-1">
      <span v-if="seen">The condition is true, which makes this text seen.</span>
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example2-2">
      <ol>
        <li v-for="todo in todos">
          {{ todo.text }}
        </li>
      </ol>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

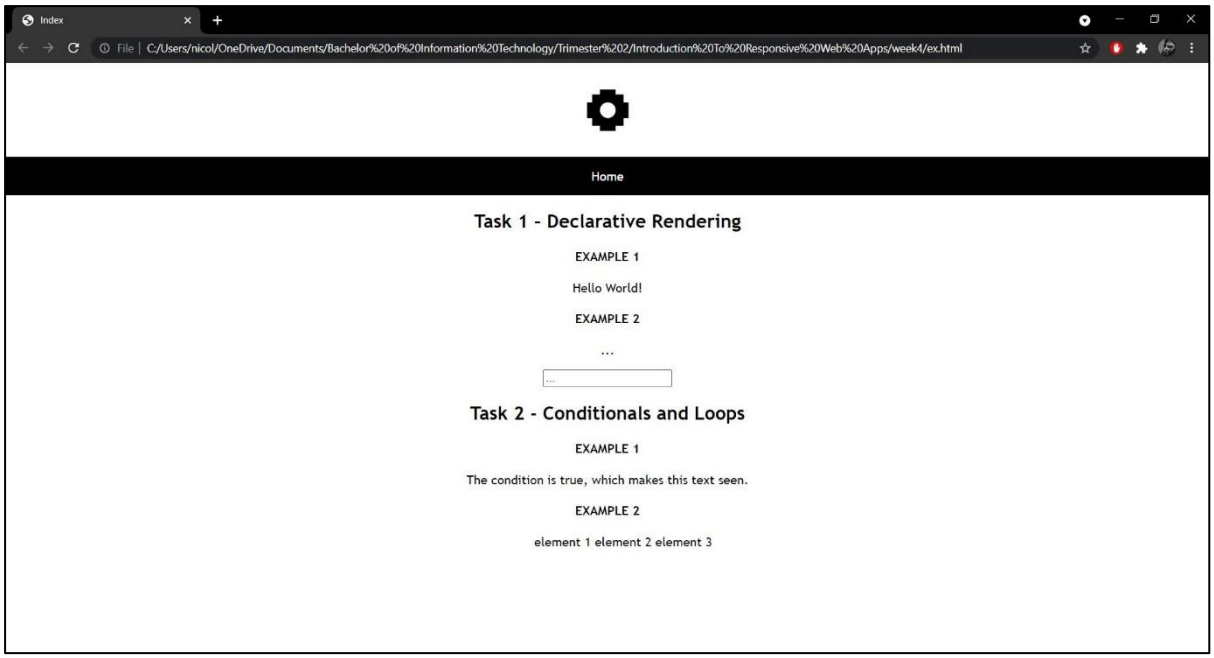
```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}
```

```
button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (vue.js):

```
var ex0 = new Vue({
  el: '#example1-1',
  data: {
    message: 'Hello World!'
  }
})

var ex1 = new Vue({
  el: '#example1-2',
  data: {
    message: '...'
  }
})

var ex2 = new Vue({
  el: '#example2-1',
  data: {
    seen: true
  }
})

var ex3 = new Vue({
  el: '#example2-2',
  data: {
    todos: [
      { text: 'element 1' },
      { text: 'element 2' },
      { text: 'element 3' }
    ]
  }
})
```

Output (index.html):



Task 3 – Discuss your proposal with tutor and peers

Discussion:

Ill must implement Vue components that'll suit my web design, also it is important to take note on user stories as they could help inflict in getting ideas that would improve my website. Its critical to also develop a UX design to interpret the interface on other devices, as this will put together the proposal for the outcome.

Learning Reflection

For this week's tasks we've covered the standard basics in learning different concepts of the Vue framework. We've learnt a brief portion based on the Vue framework, though were now going more in depth about understanding the concepts in implementing Vue – such as different components in-depth.

Week 5 – Vue.js Framework (part 2)

Notes

For this week's task we carried on from last weeks activities as we are now implementing and looking more in-depth into components.

Task 1 – Learn Composing with Components

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Learn Composing with Components</h2>

    <div class="ex" id="example1-1">
      <ol>
        <todo-item v-for="item in groceryList" v-bind:todo="item" v-bind:key="item.id"></todo-item>
      </ol>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
```

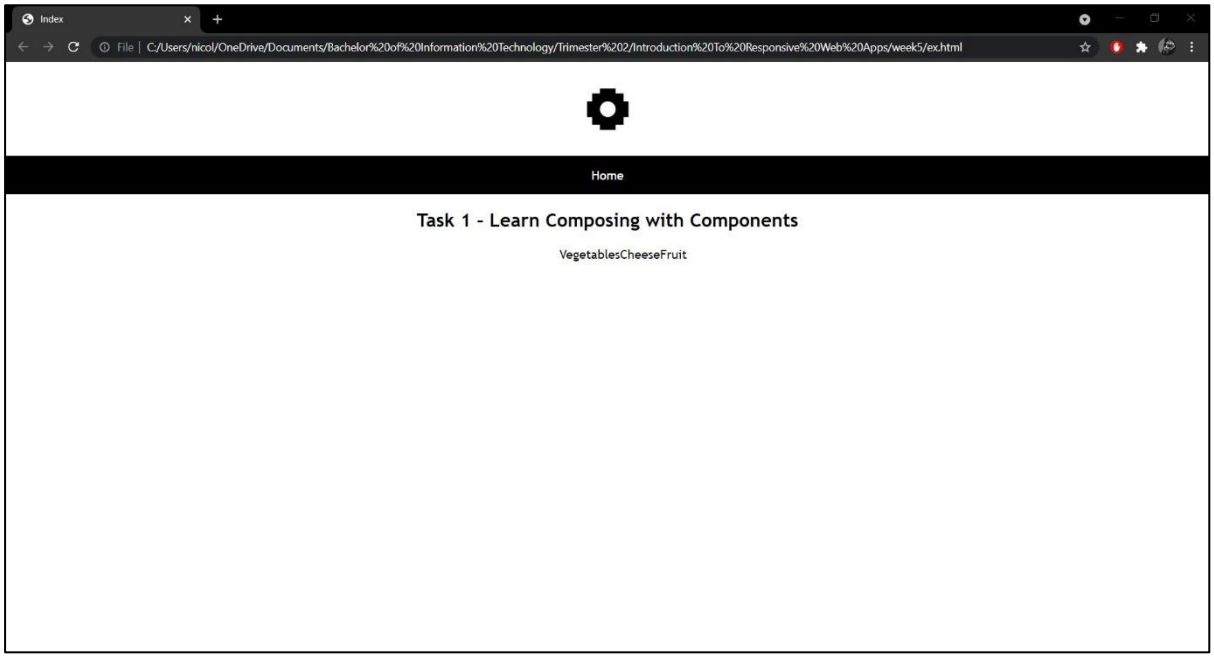
```
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (vue.js):

```
Vue.component('todo-item', {
  props: ['todo'],
```

```
  template: '<li>{{ todo.text }}</li>'
})

var ex0 = new Vue({
  el: '#example1-1',
  data: {
    groceryList: [
      { id: 0, text: 'Vegetables' },
      { id: 1, text: 'Cheese' },
      { id: 2, text: 'Fruit' }
    ]
  }
})
```

Output (index.html):



Task 2 – Exploring Vue Framework

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
```

```
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Learn Composing with Components</h2>

    <div class="ex" id="example1-1">
      <ol>
        <todo-item v-for="item in groceryList" v-bind:todo="item" v-bind:key="item.id"></todo-item>
      </ol>
    </div>

    <h2>Task 2 – Exploring Vue Framework</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example2-1">
       <heading-component></heading-component>
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example2-2">
      <art-name>Title</art-name>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}
```

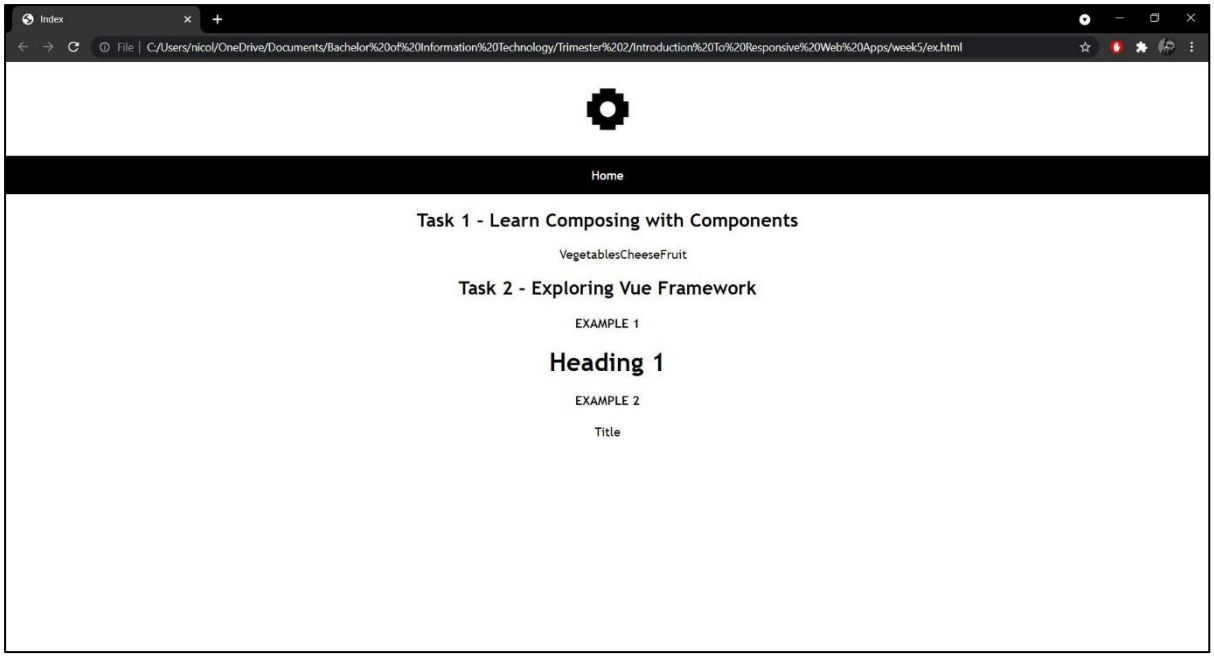
```
ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
```

```
}
```

Input (vue.js):

```
Vue.component('todo-item', {
  props: ['todo'],
  template: '<li>{{ todo.text }}</li>'
})

var ex0 = new Vue({
  el: '#example1-1',
  data: {
    groceryList: [
      { id: 0, text: 'Vegetables' },
      { id: 1, text: 'Cheese' },
      { id: 2, text: 'Fruit' }
    ]
  }
})

Vue.component('heading-component', {
   template : '<div><h1>Heading 1</h1></div>'
})

var ex1 = new Vue({ el: '#example2-1' })

Vue.component('title-component', {
  data: function () {
    return {
    }
  },
  props: ['title'],
  template: '<h4>{{ title }}</h4>'
})

var ex2 = new Vue({ el: '#example2-2' })
```

Output (index.html):



Task 3 – Understanding Handling user input

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Learn Composing with Components</h2>

    <div class="ex" id="example1-1">
      <ol>
        <todo-item v-for="item in groceryList" v-bind:todo="item" v-bind:key="item.id"></todo-item>
      </ol>
    </div>

    <h2>Task 2 – Exploring Vue Framework</h2>
```

```
    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example2-1">
       <heading-component></heading-component>
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example2-2">
      <art-name>Title</art-name>
    </div>

    <h2>Task 3 – Understanding  Handling user input</h2>

    <h4>EXAMPLE 1</h4>

    <div class="ex" id="example3-1">
      <p>{{ message }}</p>
      <button v-on:click="reverseMessage">Reverse Message</button>
    </div>

    <h4>EXAMPLE 2</h4>

    <div class="ex" id="example3-2">
      <p>{{ message }}</p>
      <input v-model="message">
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
```

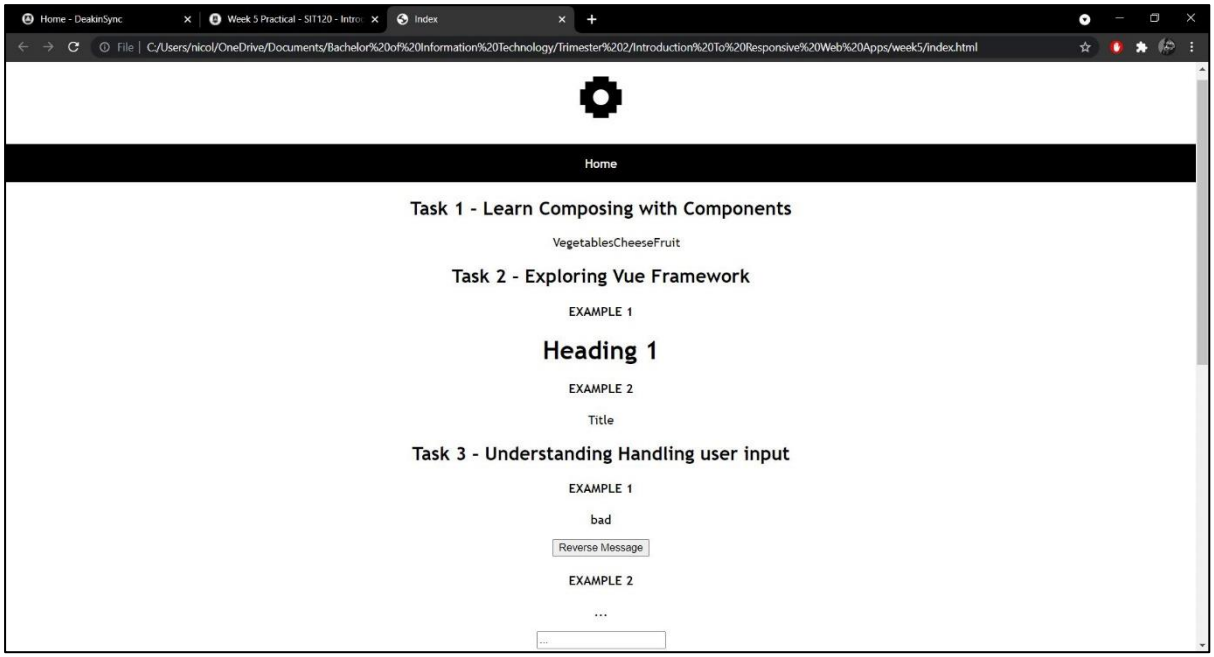
```
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}
```

Input (vue.js):

```
Vue.component('todo-item', {
  props: ['todo'],
  template: '<li>{{ todo.text }}</li>'
})

var ex0 = new Vue({
  el: '#example1-1',
  data: {
    groceryList: [
      { id: 0, text: 'Vegetables' },
      { id: 1, text: 'Cheese' },
      { id: 2, text: 'Fruit' }
    ]
  }
})

Vue.component('heading-component', {
   template : '<div><h1>Heading 1</h1></div>'
})

var ex1 = new Vue({ el: '#example2-1' })

Vue.component('title-component', {
  data: function () {
    return {
    }
  },
  props: ['title'],
  template: '<h4>{{ title }}</h4>'
})

var ex2 = new Vue({ el: '#example2-2' })

var ex3 = new Vue({
  el: '#example3-1',
  data: {
    message: 'bad'
  },
  methods: {
    reverseMessage: function () {
      this.message = this.message.split('').reverse().join('')
    }
  }
})

var ex4 = new Vue({
  el: '#example3-2',
```

```
  data: {
    message: '...'
  }
})
```

Output (index.html):



Task 4 – Learn Components in-depth

Component registration:

Component registration is referred when a name is assigned to a component that can be carried out when called to a specified location within the program.

Props:
Props are prone to be used as attributes that are accustomed for registration to an existing component – for instance, when a prop is set on execution the prop/property is served to that component.

Custom events:

A custom event is a key modifier which manipulates event handlings from the user when applied.

Slots:

Slots is a element that is implemented to allow the user to compose new components from content outlets.

Dynamic and Async components:

Dynamic components are used to switch between various components through using a tab-based interface, whereas Async components divide an app into smaller quantities – meaning that a component will be called by a server individually.

Handling edge cases:
Handling edge cases is implemented when a user wants to receive access to other existing components or take control over manipulating DOM elements, handling edge cases can have consequences to the program though it is efficiently useful if implemented correctly.

Learning Reflection

For this week's tasks we've continued from our previous week, as we have learnt more concepts of the Vue framework. This week we've understood more and have adapted to develop more skills related upon components and explored their functional uses for our responsive pages.

Week 6 – Vue.js Framework (Handling User Input)

Notes

For this week's task we are looking further in Vue's framework as we now are focusing on implementing user inputs by using handler methods.

Task 1 – Using V-model for handling user inputs

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
```

```
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <form class="ex" id="example1-1">
      <h3>Form User</h3>
      <label for="name">Name:</label><br>
      <input id="name" type="text" placeholder="Enter name"><br><br>

      <label for="password">Password:</label><br>
      <input id="password" type="password" placeholder="Enter password"><br><br>

      <label for="email">Email:</label><br>
      <input id="email" type="text" placeholder="Enter email"><br><br>

      <button v-on:click="checkInput">Submit</button>
    </form>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
```

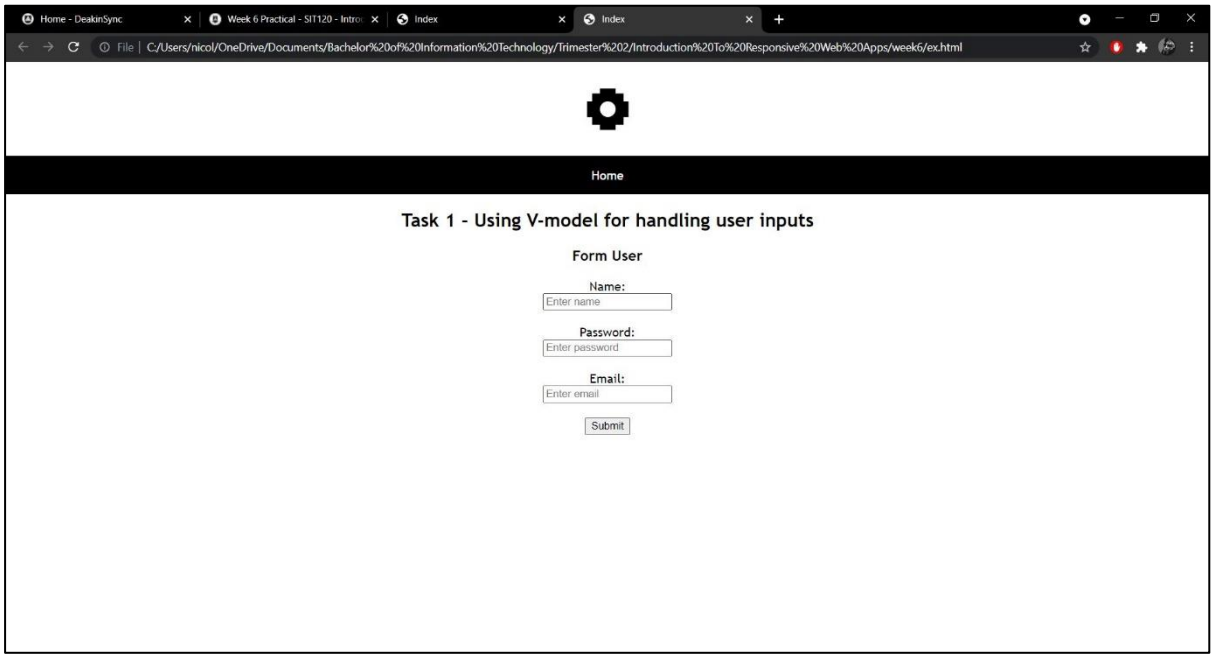
```
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  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;
}
```

Input (vue.js):

```
var ex0 = new Vue({
  el:"#example1-1",
  data:{
    username:'',
    password:'',
    email:'',
  },
  methods:{
    checkInput: function() {
      var str='';
      if (this.username) {
        str = str +"username: " + this.username;
      }
      if (this.password) {
        str = str +"password: " + this.password;
      }
      if (this.email) {
        str = str +"email: " + this.email;
      }
      if (str) {
        alert(str)
      } else {
        alert("Please input required info")
      }
    }
  }
})
```

Output (index.html):



Task 2 – Checkbox in your project

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <form class="ex" id="example1-1">
      <h3>Form User</h3>
      <label for="name">Name:</label><br>
      <input id="name" type="text" placeholder="Enter name"><br><br>

      <label for="password">Password:</label><br>
      <input id="password" type="password" placeholder="Enter password"><br><br>
```

```
      <label for="email">Email:</label><br>
      <input id="email" type="text" placeholder="Enter email"><br><br>

      <button v-on:click="checkInput">Submit</button>
    </form>

    <h2>Task 2 - Checkbox  in your project</h2>

    <div class="ex" id="example2-1">
      <input type="checkbox" id="artsize1" value="1920x1080 px" v-model="checkedDimensions" />
      <label for="artsize1">1920x1080 px</label>
      <input type="checkbox" id="artsize2" value="1280x720 px" v-model="checkedDimensions" />
      <label for="artsize2">1280x720 px</label>
      <input type="checkbox" id="artsize3" value="1080x1080 px" v-model="checkedDimensions" />
      <label for="artsize3">1080x1080 px</label>
      <br />
      <span>Checked dimensions: {{ checkedDimensions }}</span>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}
```

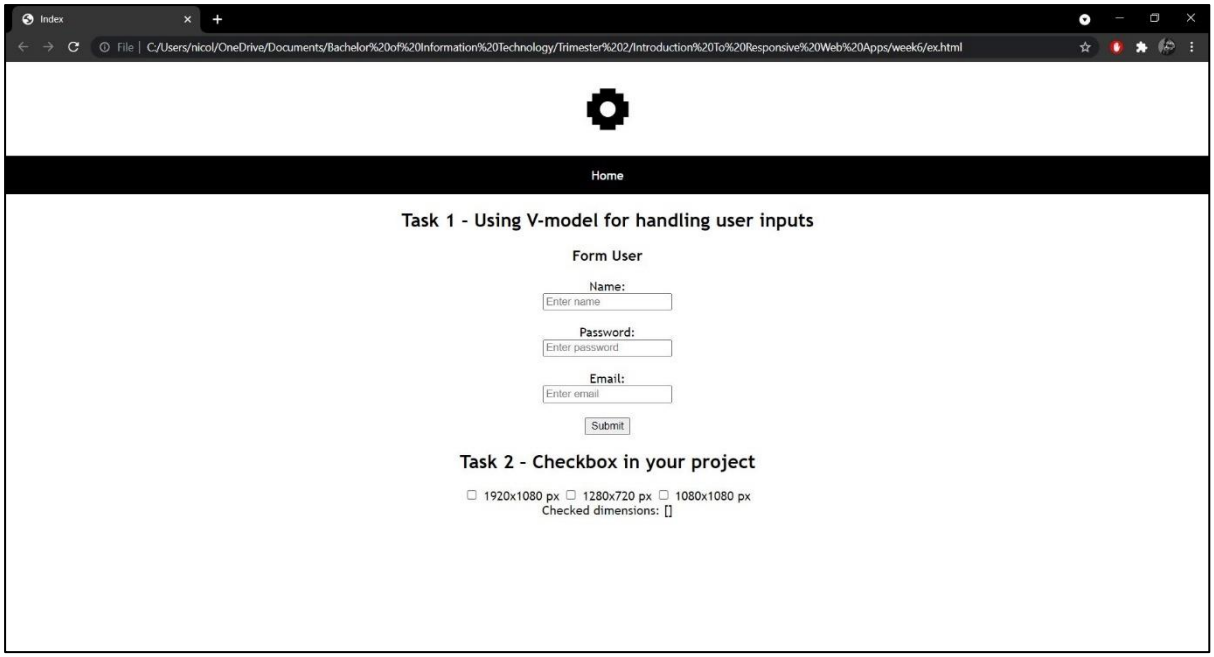
```
li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  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;
}
```

Input (vue.js):

```
var ex0 = new Vue({
  el:"#example1-1",
  data:{
    username:'',
    password:'',
    email:'',
  },
  methods:{
    checkInput: function() {
      var str='';
      if (this.username) {
        str = str +"username: " + this.username;
      }
      if (this.password) {
        str = str +"password: " + this.password;
      }
      if (this.email) {
        str = str +"email: " + this.email;
      }
      if (str) {
        alert(str)
      } else {
        alert("Please input required info")
      }
    }
  }
})

var ex1 = new Vue({
  el: "#example2-1",
  data: {
    checkedDimensions:[],
  },
})
```

Output (index.html):



Task 3 – Checkbox in your project

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <form class="ex" id="example1-1">
      <h3>Form User</h3>
      <label for="name">Name:</label><br>
      <input id="name" type="text" placeholder="Enter name"><br><br>

      <label for="password">Password:</label><br>
      <input id="password" type="password" placeholder="Enter password"><br><br>
```

```
      <label for="email">Email:</label><br>
      <input id="email" type="text" placeholder="Enter email"><br><br>

      <button v-on:click="checkInput">Submit</button>
    </form>

    <h2>Task 2 – Checkbox  in your project</h2>

    <div class="ex" id="example2-1">
      <input type="checkbox" id="artsize1" value="1920x1080 px" v-model="checkedDimensions" />
      <label for="artsize1">1920x1080 px</label>
      <input type="checkbox" id="artsize2" value="1280x720 px" v-model="checkedDimensions" />
      <label for="artsize2">1280x720 px</label>
      <input type="checkbox" id="artsize3" value="1080x1080 px" v-model="checkedDimensions" />
      <label for="artsize3">1080x1080 px</label>
      <br />
      <span>Checked dimensions: {{ checkedDimensions }}</span>
    </div>

    <h2>Task 3 – Dynamic options rendering v-for</h2>

    <div class="ex" id="example3-1">
      <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://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
```

```
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
```

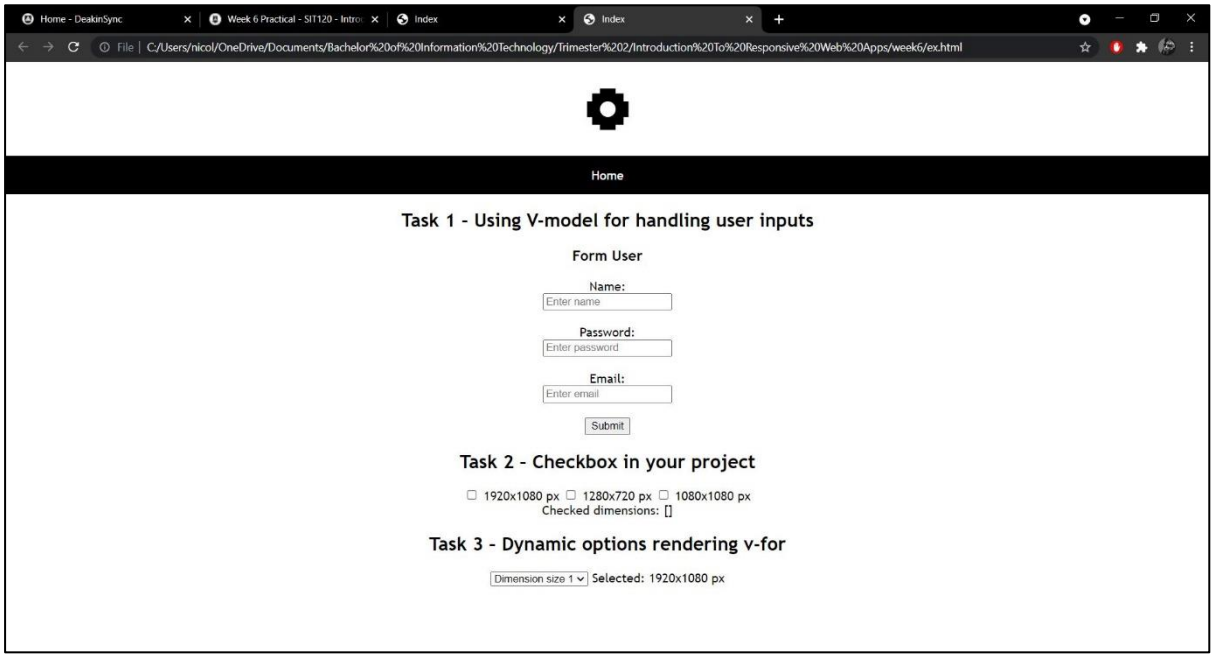
```
  text-align: center;
}

.demo {
  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;
}
```

Input (vue.js):

```
var ex0 = new Vue({
  el:"#example1-1",
  data:{
    username:'',
    password:'',
    email:'',
  },
  methods:{
    checkInput: function() {
      var str='';
      if (this.username) {
        str = str +"username: " + this.username;
      }
      if (this.password) {
        str = str +"password: " + this.password;
      }
      if (this.email) {
        str = str +"email: " + this.email;
      }
      if (str) {
        alert(str)
      } else {
        alert("Please input required info")
      }
    }
  }
})

var ex1 = new Vue({
  el: "#example2-1",
  data: {
    checkedDimensions:[],
```

```
  },
})

var ex2 = new Vue({
  el: "#example3-1",
  data: {
    selected: '1920x1080 px',
    options: [
      { text: 'Dimension size 1', value: '1920x1080 px' },
      { text: 'Dimension size 2', value: '1280x720 px' },
      { text: 'Dimension size 3', value: '1080x1080 px' }
    ]
  },
})
```

Output (index.html):



Task 4 – Using Modifiers in your projects

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
```

```
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <form class="ex" id="example1-1">
      <h3>Form User</h3>
      <label for="name">Name:</label><br>
      <input id="name" type="text" placeholder="Enter name"><br><br>

      <label for="password">Password:</label><br>
      <input id="password" type="password" placeholder="Enter password"><br><br>

      <label for="email">Email:</label><br>
      <input id="email" type="text" placeholder="Enter email"><br><br>

      <button v-on:click="checkInput">Submit</button>
    </form>

    <h2>Task 2 – Checkbox  in your project</h2>

    <div class="ex" id="example2-1">
      <input type="checkbox" id="artsize1" value="1920x1080 px" v-model="checkedDimensions" />
      <label for="artsize1">1920x1080 px</label>
      <input type="checkbox" id="artsize2" value="1280x720 px" v-model="checkedDimensions" />
      <label for="artsize2">1280x720 px</label>
      <input type="checkbox" id="artsize3" value="1080x1080 px" v-model="checkedDimensions" />
      <label for="artsize3">1080x1080 px</label>
      <br />
      <span>Checked dimensions: {{ checkedDimensions }}</span>
    </div>

    <h2>Task 3 – Dynamic options rendering v-for</h2>

    <div class="ex" id="example3-1">
      <select v-model="selected">
        <option v-for="option in options" v-bind:value="option.value">
          {{ option.text }}
        </option>
      </select>
      <span>Selected: {{ selected }}</span>
  </div>
```

```
  <h2>Task 4 - Using Modifiers in your project</h2>

  <form class="ex" id="example4-1">
    <h3>Form User</h3>
    <label for="name">Name:</label><br>
    <input id="name" type="text" v-model.lazy="username" placeholder="Enter name"><br><br>
    <p>Username: {{ username }}</p><br><br>

    <label for="password">Password:</label><br>
    <input id="password" type="password" v.model="pasword" placeholder="Enter password"><br><br>

    <label for="email">Email:</label><br>
    <input id="email" type="text" v.model="email" placeholder="Enter email"><br><br>

    <button v-on:click="checkInput">Submit</button>
  </form>

    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
```

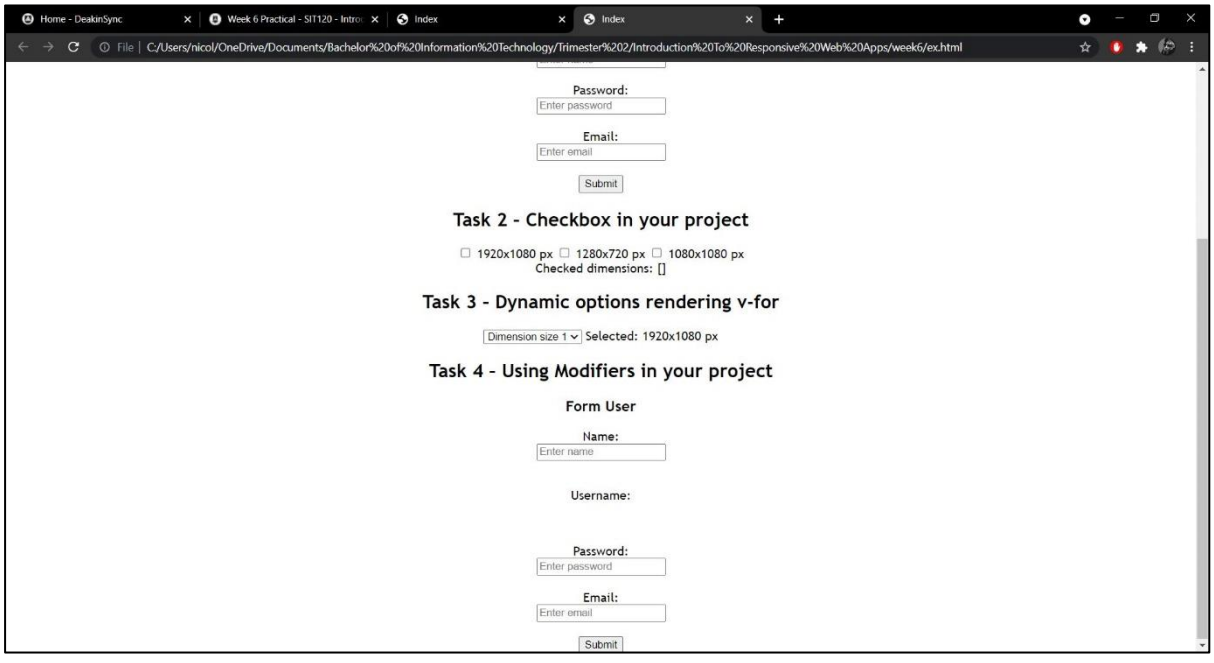
```
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  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;
}
```

Input (scripts.js):

```
var ex0 = new Vue({
  el:"#example1-1",
  data:{
    username:'',
    password:'',
    email:'',
  },
  methods:{
    checkInput: function() {
      var str='';
      if (this.username) {
        str = str +"username: " + this.username;
      }
      if (this.password) {
        str = str +"password: " + this.password;
      }
      if (this.email) {
        str = str +"email: " + this.email;
      }
      if (str) {
        alert(str)
      } else {
        alert("Please input required info")
      }
    }
  }
})

var ex1 = new Vue({
  el: "#example2-1",
  data: {
    checkedDimensions:[],
  },
})

var ex2 = new Vue({
  el: "#example3-1",
  data: {
    selected: '1920x1080 px',
    options: [
      { text: 'Dimension size 1', value: '1920x1080 px' },
      { text: 'Dimension size 2', value: '1280x720 px' },
      { text: 'Dimension size 3', value: '1080x1080 px' }
    ]
  },
})
```

```
var ex3 = new Vue({
  el:"#example4-1",
  data:{
    username:'',
    password:'',
    email:'',
  },
  methods:{
    checkInput: function() {
      var str='';
      if (this.username) {
        str = str +"username: " + this.username;
      }
      if (this.password) {
        str = str +"password: " + this.password;
      }
      if (this.email) {
        str = str +"email: " + this.email;
      }
      if (str) {
        alert(str)
      } else {
        alert("Please input required info")
      }
    }
  }
})
```

Output (index.html):



1. Do you want to use lazy modifiers to sync after change events? Think and discuss with peers.

For the most case a v-model will sync the input during an input event, although adding in a lazy modifier is an alternative than syncing after change events.

2. Do you need to trim whitespace from user inputs automatically?

It is an optional choice for a user input to be trimmed automatically, though it benefits the v-model to manage inputs.

3. What about automatic typecasting? Will that be useful for your project?

It will be useful to an extent, depending whether you want the user input to be an automatic typecast as a number

Learning Reflection

For this week's tasks we've continued from our previous weeks 4 & 5, as we now developed in-depth skills on Vue's framework. In this week's task we've experimented with handling user inputs, such as creating a form input binding. We've also had a look at more components and the concepts on how bindings and modifiers work. Understanding this weeks concepts has made me adapt deeper knowledge on Vue's functions and components.

Week 7 – Vue.js Framework (Registration, Props, Custom Events, Dynamic/Async)

Notes

For this week's tasks we look further in-depth on Vue's framework which include registration, props, custom events, and dynamic/async.

Task 1 – Using V-model for handling user inputs

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <script src="https://unpkg.com/vue@next"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
```

```
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <div class="ex" id="example1-1">

      <h4>LOCAL REGISTRATION</h4>

      <demo-component
      v-bind:title="title"
      v-bind:likes="40"
      v-bind:commentids="commentids"
      v-bind:author="author"
      ></demo-component>

      <h4>GLOBAL REGISTRATION</h4>

      <demo-component
      v-bind:title="post.title"
      v-bind:likes="post.likes"
      v-bind:commentids="post.commentids"
      v-bind:author="post.author"
      v-on:say-hi="sayHi"
      ></demo-component>

      <component-a></component-a>
    </div>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}
```

```css
.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}
```

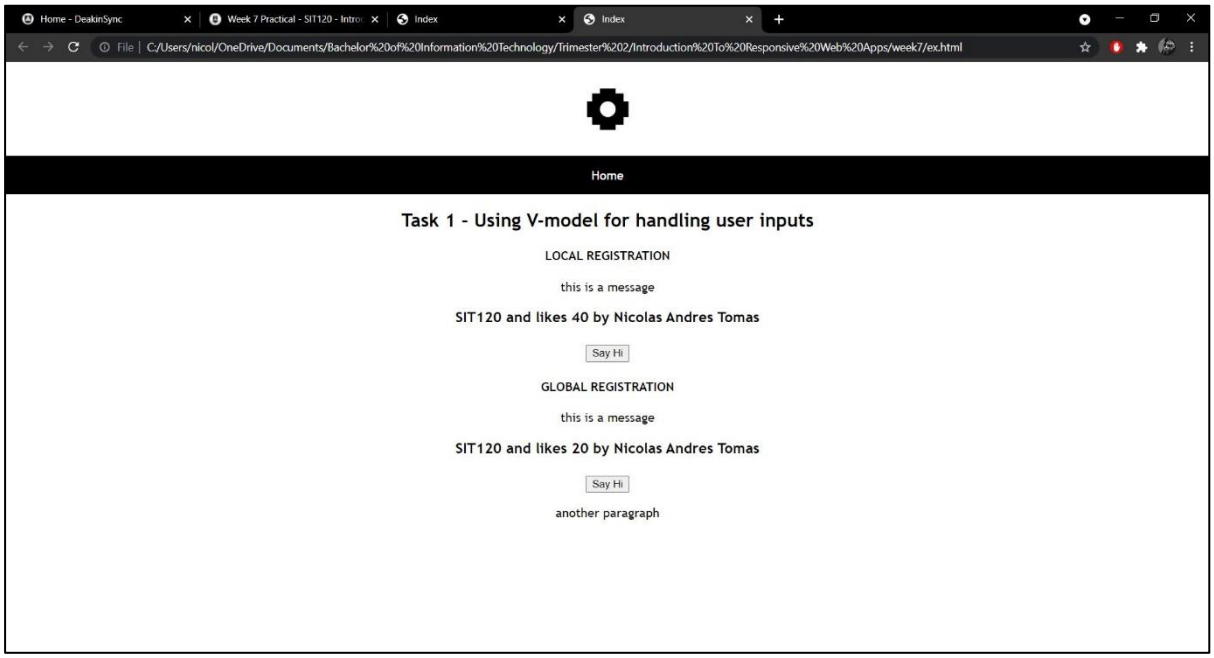
```
p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  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;
}
```

Input (vue.js):

```
Vue.component("demo-component", {
  props:{
    title:String,
    likes:Number,
    commentids:Array,
    author:Object,
  },
  template:'<div><p><demoComponent></demoComponent></p> \
  <h3>{{title}} and likes {{likes}} by {{author.name}}</h3> \
  <h6></h6> <button v:on:click="$emit(\'say-hi\')">Say Hi</button>\
  </div>',
})

Vue.component("demoComponent", {
  template:"<p>this is a message</p>",
})

var componentA = {
  template:"<p>another paragraph</p>",
}

var ex1 = new Vue({
  el: "#example1-1",
  data: {
    title:'SIT120',
    likes: 20,
    commentids:[10,11,12],
```

```
    author: {
      name: 'Nicolas Andres Tomas',
    },
    post: {
      title:'SIT120',
      likes: 20,
      commentids:[10,11,12],
      author: {
        name: 'Nicolas Andres Tomas',
      },
    },
  },
  components: {
    "component-a": componentA,
  },
  methods: {
    sayHi: function() {
      alert("hi");
    },
  },
})

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

Output (index.html):



Task 2 – Using V-model for handling user inputs

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <script src="https://unpkg.com/vue@next"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>

    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <div class="ex" id="example1-1">

      <h4>LOCAL REGISTRATION</h4>
```

```
      <demo-component
      v-bind:title="title"
      v-bind:likes="40"
      v-bind:commentids="commentids"
      v-bind:author="author"
      ></demo-component>

      <h4>GLOBAL REGISTRATION</h4>

      <demo-component
      v-bind:title="post.title"
      v-bind:likes="post.likes"
      v-bind:commentids="post.commentids"
      v-bind:author="post.author"
      v-on:say-hi="sayHi"
      ></demo-component>

      <component-a></component-a>
    </div>

    <h2>Task 2 – Coding Props</h2>

    <div class="ex" id="example2-1">
      <child :text="message"></child>
    </div>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
```

```
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  font-family: sans-serif;
  border: 1px solid #eee;
```

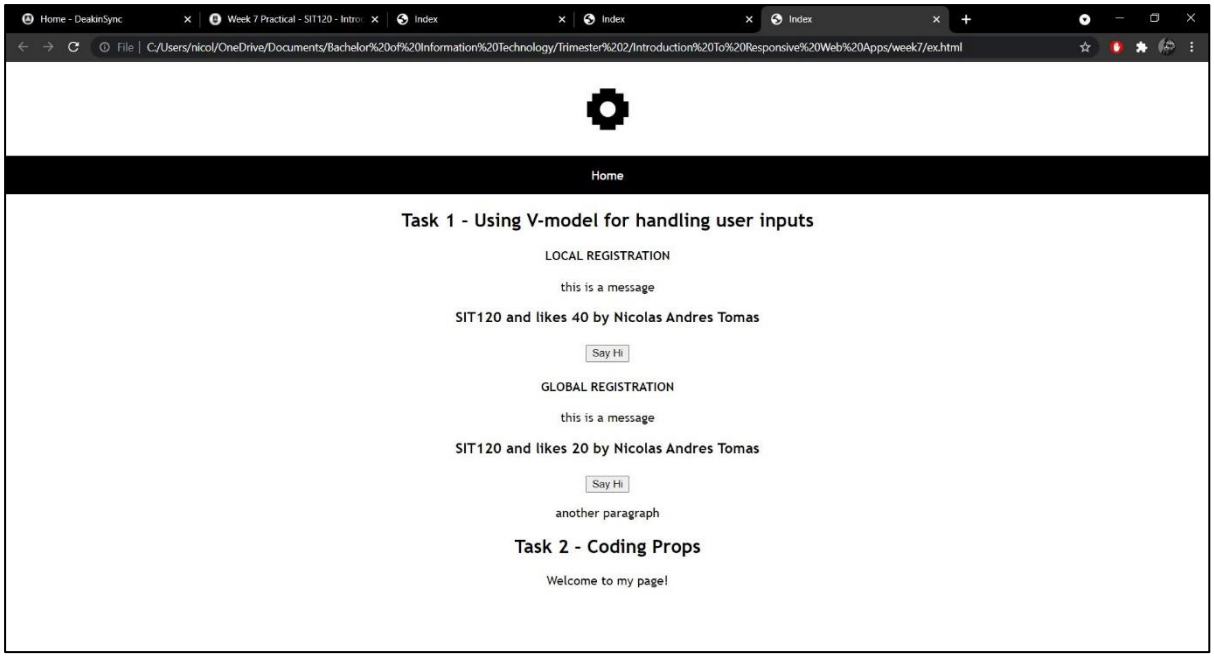
```
  border-radius: 2px;
  padding: 20px 30px;
  margin-top: 1em;
  margin-bottom: 40px;
  user-select: none;
  overflow-x: auto;
}
```

Input (vue.js):

```
Vue.component("demo-component", {
  props:{
    title:String,
    likes:Number,
    commentids:Array,
    author:Object,
  },
  template:'<div><p><demoComponent></demoComponent></p> \
  <h3>{{title}} and likes {{likes}} by {{author.name}}</h3> \
  <h6></h6> <button v:on:click="$emit(\'say-hi\')">Say Hi</button>\
  </div>',
})

Vue.component("demoComponent", {
  template:"<p>this is a message</p>",
})

var componentA = {
  template:"<p>another paragraph</p>",
}

var ex1 = new Vue({
  el: "#example1-1",
  data: {
    title:'SIT120',
    likes: 20,
    commentids:[10,11,12],
    author: {
      name: 'Nicolas Andres Tomas',
    },
    post: {
      title:'SIT120',
      likes: 20,
      commentids:[10,11,12],
      author: {
        name: 'Nicolas Andres Tomas',
      },
    },
```

```
  },
  components: {
    "component-a": componentA,
  },
  methods: {
    sayHi: function() {
      alert("hi");
    },
  },
})

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

new Vue({
  el: '#example2-1',
  data() {
    return {
      message: 'Welcome to my page!'
    }
  }
})

Vue.component('child-component', {
  template: '#child-component',
  data() {
      return {
        childTitle: 'title from child component',
        childSubtitle: 'subtitle from child component'
      }
  },
  props:{
    subtitle: {
      type: String,
      required: true
    }
  }
})
```

Output (index.html):



Task 3 – Custom Events

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <script src="https://unpkg.com/vue@next"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>

    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <div class="ex" id="example1-1">

      <h4>LOCAL REGISTRATION</h4>

      <demo-component
```

```
    v-bind:title="title"
    v-bind:likes="40"
    v-bind:commentids="commentids"
    v-bind:author="author"
    ></demo-component>

    <h4>GLOBAL REGISTRATION</h4>

    <demo-component
    v-bind:title="post.title"
    v-bind:likes="post.likes"
    v-bind:commentids="post.commentids"
    v-bind:author="post.author"
    v-on:say-hi="sayHi"
    ></demo-component>

    <component-a></component-a>
  </div>

  <h2>Task 2 – Coding Props</h2>

  <div class="ex" id="example2-1">
    <child :text="message"></child>
  </div>

  <h2>Task 3 - Custom Events</h2>

  <div class="demo" id="v-model-example">
     <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>
    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')
    </script>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}
```

```
li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

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

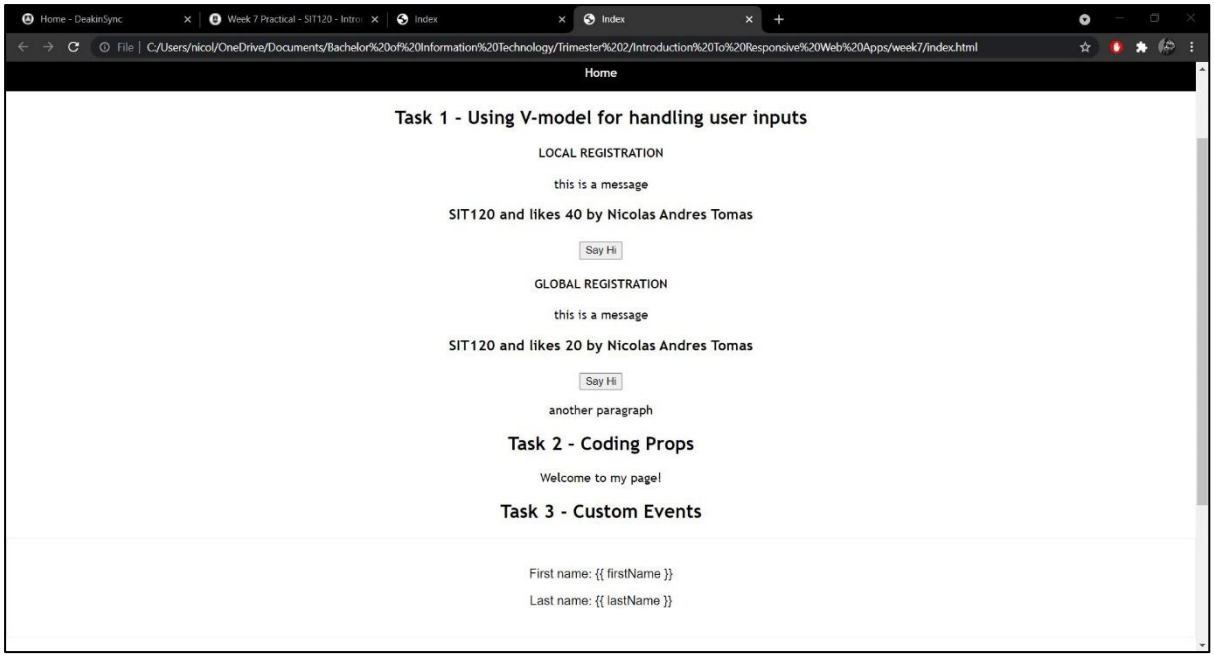
```
  overflow-x: auto;
}
```

Input (vue.js):

```
Vue.component("demo-component", {
  props:{
    title:String,
    likes:Number,
    commentids:Array,
    author:Object,
  },
  template:'<div><p><demoComponent></demoComponent></p> \
  <h3>{{title}} and likes {{likes}} by {{author.name}}</h3> \
  <h6></h6> <button v:on:click="$emit(\'say-hi\')">Say Hi</button>\
  </div>',
})

Vue.component("demoComponent", {
  template:"<p>this is a message</p>",
})

var componentA = {
  template:"<p>another paragraph</p>",
}

var ex1 = new Vue({
  el: "#example1-1",
  data: {
    title:'SIT120',
    likes: 20,
    commentids:[10,11,12],
    author: {
      name: 'Nicolas Andres Tomas',
    },
    post: {
      title:'SIT120',
      likes: 20,
      commentids:[10,11,12],
      author: {
        name: 'Nicolas Andres Tomas',
      },
    },
  },
  components: {
    "component-a": componentA,
  },
  methods: {
```

```
    sayHi: function() {
      alert("hi");
    },
  },
})

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

new Vue({
  el: '#example2-1',
  data() {
    return {
      message: 'Welcome to my page!'
    }
  }
})

Vue.component('child-component', {
  template: '#child-component',
  data() {
      return {
        childTitle: 'title from child component',
        childSubtitle: 'subtitle from child component'
      }
  },
  props:{
    subtitle: {
      type: String,
      required: true
    }
  }
})
```

Output (index.html):



Task 4 – Vue Slots – coding Named Slots

Input (index.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Index</title> <!-- page title -->
    <script src="https://unpkg.com/vue@next"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>

    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
  </head>
  <body>
    <img class="logo" src="images/logo.png" alt="logo"> <!-
- page logo "logo.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="index.html">Home</a></li> <!-- index.html link -->
    </ul>

    <h2>Task 1 – Using V-model for handling user inputs</h2>

    <div class="ex" id="example1-1">

      <h4>LOCAL REGISTRATION</h4>

      <demo-component
```

```
    v-bind:title="title"
    v-bind:likes="40"
    v-bind:commentids="commentids"
    v-bind:author="author"
    ></demo-component>

    <h4>GLOBAL REGISTRATION</h4>

    <demo-component
    v-bind:title="post.title"
    v-bind:likes="post.likes"
    v-bind:commentids="post.commentids"
    v-bind:author="post.author"
    v-on:say-hi="sayHi"
    ></demo-component>

    <component-a></component-a>
  </div>

  <h2>Task 2 – Coding Props</h2>

  <div class="ex" id="example2-1">
    <child :text="message"></child>
  </div>

  <h2>Task 3 - Custom Events</h2>

  <div class="demo" id="v-model-example">
     <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>
    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')
  </script>

  <h2>Task 4 - Vue Slots - coding Named Slots</h2>

  <div id="example4-1">
     <section class="hero is-fullheight is-warning is-bold">
       <div class="hero-body">
         <div class="container">
           <p class="title is-1">{{ title }}</p>
           <p class="subtitle is-3">{{ subtitle }}</p>
          <child-component>
            <p slot="bottom">DECRIPTION</p>
            <p>title goes here</p>
            <p slot="top">TITLE</p>
          </child-component>
          <child-component></child-component>
         </div>
       </div>
    </section>
  </div>

  <script type="text/x-template" id="child-component">
    <div>
      <slot name="top">
      </slot>
      <slot>
        <p>decription goes here</p>
      </slot>
      <slot name="bottom">
```

```
        </slot>
      </div>
    </script>

    <script src="vue.js"></script>  <!-- linked to scripts.js -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 330px;
```

```
}

h2 { /* h2 style */
    text-align: center;
}

h4 { /* h4 style */
    text-align: center;
}

button { /* h4 style */
  margin-left: auto;
  margin-right: auto;
}

p { /* p style */
    text-align: center;
}

.ex {
  text-align: center;
}

.demo {
  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;
}
```

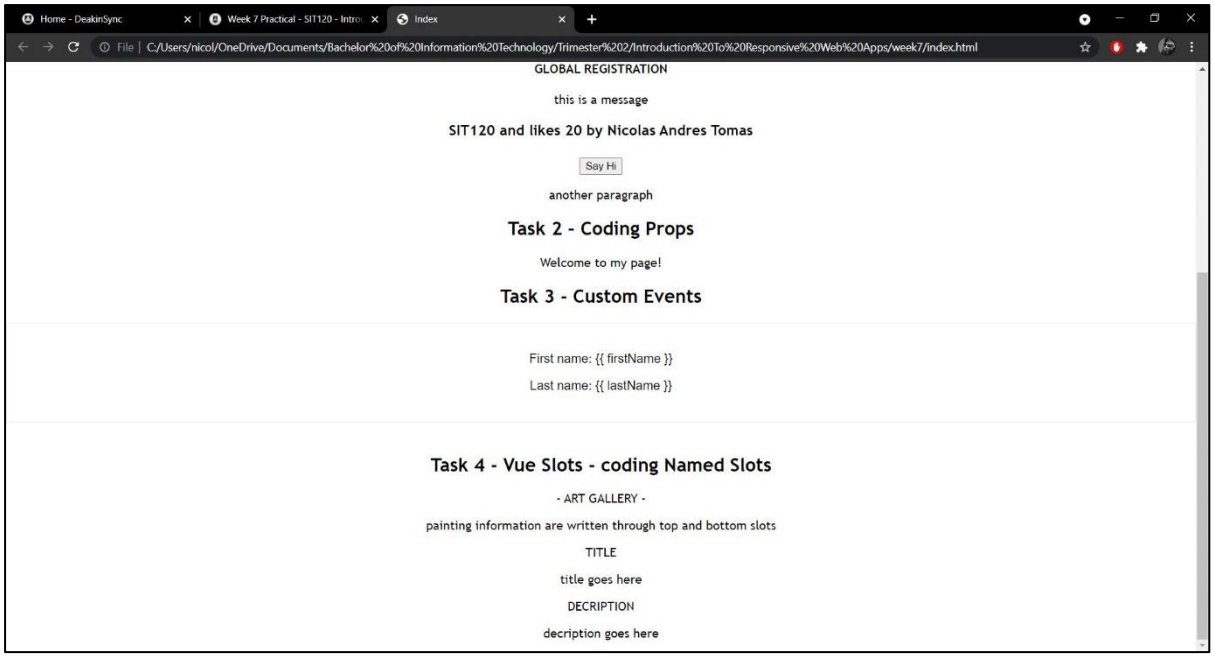
Input (vue.js):

```
Vue.component("demo-component", {
  props:{
    title:String,
    likes:Number,
    commentids:Array,
    author:Object,
  },
  template:'<div><p><demoComponent></demoComponent></p> \
  <h3>{{title}} and likes {{likes}} by {{author.name}}</h3> \
  <h6></h6> <button v:on:click="$emit(\'say-hi\')">Say Hi</button>\
  </div>',
})
```

```
Vue.component("demoComponent", {
  template:"<p>this is a message</p>",
})

var componentA = {
  template:"<p>another paragraph</p>",
}

var ex1 = new Vue({
  el: "#example1-1",
  data: {
    title:'SIT120',
    likes: 20,
    commentids:[10,11,12],
    author: {
      name: 'Nicolas Andres Tomas',
    },
    post: {
      title:'SIT120',
      likes: 20,
      commentids:[10,11,12],
      author: {
        name: 'Nicolas Andres Tomas',
      },
    },
  },
  components: {
    "component-a": componentA,
  },
  methods: {
    sayHi: function() {
      alert("hi");
    },
  },
})

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

new Vue({
```

```
  el: '#example2-1',
  data() {
    return {
      message: 'Welcome to my page!'
    }
  }
})

Vue.component('child-component', {
  template: '#child-component',
  data() {
      return {
        childTitle: 'title from child component',
        childSubtitle: 'subtitle from child component'
      }
  },
  props:{
    subtitle: {
      type: String,
      required: true
    }
  }
})

new Vue ({
  el: '#example4-1',
  data() {
      return {
        title: '- ART GALLERY -',
        subtitle: 'painting information are written through top and bottom slots'
    }
  }
})
```

Output (index.html):



Learning Reflection

For this week's tasks we've continued from our previous weeks 4, 5, and 6 – we've now obtained enough knowledge of the Vue framework, as now we can implement all of our abilities to our proposed project. This week we gained an insight on more components with such as registration, props, custom events, and dynamic/async.

MY WEBSITE

All weekly tasks and practical's from 1-7 have been applied to my website.
GitHub: https://github.com/NicolasAndresTomas/Introduction-To-Responsive-Web-Apps

Input (home.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Home</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14"></script>
    <script src="https://unpkg.com/vue"></script>
  </head>
  <body>
    <img class="logo" src="images/chankana.png" alt="logo"> <!-
- page logo "chankana.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="home.html">Home</a></li> <!-- home.html link -->
```

```
      <li><a href="contact.html">Contact</a></li> <!-- contact.html link -->
      <li><a href="about.html">About</a></li> <!-- about.html link -->
    </ul>

  <div class="row">
    <div class="column">
      <img src="images/image01.jpg" alt="art"> <!-- image "painting.jpeg" -->
    </div>

    <div class="column">
      <img src="images/image02.jpg" alt="art"> <!-- image "painting.jpeg" -->
    </div>

    <div class="column">
      <img src="images/image03.jpg" alt="art"> <!-- image "painting.jpeg" -->
    </div>
  </div>

  <div id="title-component">
    <art-name class="demo1" title="El esqueleto">El esqueleto</art-name>
    <art-name class="demo2" title="Polynartha">Polynartha</art-name>
    <art-name class="demo3" title="Pet">Pet</art-name>
  </div>

  <p class="footer">© 2021 - Nicolas Andres Tomas, 221351413</p> <!-
- footer classed as "footer" linked to styles.css -->

  <script src="vue.js"></script> <!-- linked to javascript -->
  </body>
</html>
```

Input (contact.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Contact</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14"></script>
    <script src="https://unpkg.com/vue"></script>
  </head>
  <body>
    <img class="logo" src="images/chankana.png" alt="logo"> <!-
- page logo "chankana.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
```

```
      <li><a href="home.html">Home</a></li> <!-- home.html link -->
      <li><a href="contact.html">Contact</a></li> <!-- contact.html link -->
      <li><a href="about.html">About</a></li> <!-- about.html link -->
    </ul>

  <div class="contact-form"> <!-- div tag, classed as "contact-form" linked to styles.css -->
    <form method="post" name="form" id="form"> <!-- form -->
      <div class="contact"> <!-- div tag classed as "contact" linked to styles.css -->
        <label for="name">Name: </label> <!-- label "Name: " for form -->
        <input type="text" id="name" name="name" aria-describedby="name-
format" placeholder="Enter your Name"> <span class="error"><p id="name_error"></p></span> <!-
- input "name" for form -->
      </div>

      <div class="contact"> <!-- div tag "contact" linked to styles.css -->
        <label for="name">Email: </label> <!-- label "Email: " for form -->
        <input type="text" id="email" name="email" placeholder="Enter your Email"> <span class="error"
><p id="email_error"></p></span> <!-- input "email" for form -->
      </div>

      <div class="contact"> <!-- div tag "contact" linked to styles.css -->
        <label for="name">Subject: </label> <!-- label "Subject: " for form -->
        <input type="text" id="subject" name="subject" aria-describedby="subject-
format" placeholder="Enter your Subject"> <span class="error"><p id="subject_error"></p></span> <!-
- input "subject" for form -->
      </div>

      <div class="contact"> <!-- div tag "contact" linked to styles.css -->
        <label for="name">Message: </label> <!-- label "Message: " for form -->
        <input type="text" id="message" name="message" aria-describedby="message-
format" placeholder="Enter your Message"> <span class="error"><p id="message_error"></p></span> <!-
- input "message" for form -->
      </div>

      <div id="prompt">  <!-- submit-
1 is the event handler from vue.js - containing an inline handler within -->
        <button v-
on:click="say('You have clicked on the submit button')" class="btn" type="submit" value="submit">Submi
t</button> <!-- button for form classed as "btn" linked to styles.css -->
      </div>
      </form>
    </div>

    <p class="footer">© 2021 - Nicolas Andres Tomas, 221351413</p> <!-
- footer classed as "footer" linked to styles.css -->

    <script src="form.js"></script> <!-- linked to javascript -->
    <script src="vue.js"></script> <!-- linked to javascript -->
```

```
  </body>
</html>
```

Input (about.html):

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>About</title> <!-- page title -->
    <link rel="stylesheet" href="styles.css"> <!-- linked to styles.css -->
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14"></script>
    <script src="https://unpkg.com/vue"></script>
  </head>
  <body>
    <img class="logo" src="images/chankana.png" alt="logo"> <!-
- page logo "chankana.png" classed as "logo" linked to styles.css-->

    <ul> <!-- naviagtion list -->
      <li><a href="home.html">Home</a></li> <!-- home.html link -->
      <li><a href="contact.html">Contact</a></li> <!-- contact.html link -->
      <li><a href="about.html">About</a></li> <!-- about.html link -->
    </ul>

    <div class="title" id="artist-name">  <!-- title aritst name -->
       <heading-component></heading-component>
    </div>

    <anchored-heading-
component class="sentence" :level="1">- Ernesto Vaquero is an abstract artist - originated from the so
uth-eastern suburbs of Melbourne, Australia -</anchored-heading-componenet> <!-
- paragraphs for biography -->

    <img src="images/image04.jpg" alt="artist"> <!-- image "artist.jpeg" -->

    <p>Art that i would like to showcase will present you the artists prolonged process of painting wi
th ADHD – it depicts a strong message to the viewer explaining A journey that Art
      takes, Art is never complete, there is never a point where the artist is satisfied with what’s o
n the canvas, the dull motive gives the viewer an opputunity to capture there own
      interpretation of what is seen on canvas, each direction the Art is being taken changes througho
ut the days the artist works on the piece - changing and forming something beyond
      interpretation - subliminal key references are implemented for the artists own pleasure. the res
t are puzzled images that are only recognised when the viewer dissects the canvas rather than
      viewing it as one whole.</p> <!-- paragraph for biography -->

    <p>The Art that is viewed gives one interpretation but in a split moment can give you something el
se - before you realise, you’ve been staring at it for at least an hour only to be
```

```
      questioning your own interpretation.</p> <!-- paragraph for biography -->



    <p class="footer">© 2021 - Nicolas Andres Tomas, 221351413</p> <!-
- footer classed as "footer" linked to styles.css -->

  <script src="vue.js"></script> <!-- linked to javascript -->
  </body>
</html>
```

Input (styles.css):

```
body {  /* body style */
    margin: 0;
    font-family: 'Trebuchet MS', sans-serif;
}

.logo { /* class "logo" style */
    margin-left: auto;
    margin-right: auto;
    width: 60px;
    padding: 30px;
}

ul { /* unordered list style */
    background-color: black;
    border-width:1px 0;
    list-style:none;
    margin:0;
    padding:0;
    text-align:center;
}

li { /* list item style */
    display:inline;
}

a { /* hyperlink style */
    display:inline-block;
    padding: 15px;
    color: white;
    text-decoration: none;
}

img { /* image style */
    display: block;
    padding: 65px;
```

```
    margin-left: auto;
    margin-right: auto;
    width: 330px;
}

p {  /* paragraph */
  width: 50%;
  color: black;
  font-size: 13px;
  text-align:center;
  padding-top: 30px;
  padding-bottom: 10px;
  border: 3px;
  margin: auto;
}

.contact-form { /* class "contact-form" style */
    display: block;
    padding: 65px;
    margin-left: auto;
    margin-right: auto;
    width: 40%;
    box-sizing: border-box;
    text-align: center;
}

.contact-form h1 { /* class "contact-form" heading 1 style */
    margin-top: 0;
    font-weight: 200;
}

.contact { /* class "contact" style */
    border:1px solid gray;
    margin: 8px 0;
    padding: 12px 18px;
}

.contact label { /* class "contact" label style */
    display: block;
    text-align: left;
    color: black;
    text-transform: uppercase;
    font-size: 14px;
}

.contact input, .contact textarea { /* class "contact" input and text area style */
    width: 100%;
    border: none;
```

```
    background: none;
    outline: none;
    font-size: 18px;
    margin-top: 6px;
}

.btn { /* class "btn" style */
    width: 100%;
    border: none;
    background: black;
    font-size: 15px;
    margin-top: 8px;
    display: inline-block;
    padding: 14px;
    color: white;
    text-transform: uppercase;
    cursor: pointer;
}

h2 { /* h2 style */
    text-align: center;
}

.footer { /* class "footer" style */
    width: 100%;
    color: black;
    font-size: 10px;
    text-align:center;
    padding-top: 70px;
    padding-bottom: 10px;
}

.column {
  float: left;
  width: 32.6%;
  padding: 5px;
}

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

.demo1 {
  position: absolute;
  margin: 0 auto;
  left: 67%;
```

```
  right: 0;
  top: 95%;
  text-align: center;
  width: 10%;
}

.demo2 {
  position: absolute;
  margin: 0 auto;
  left: 0;
  right: 0;
  top: 95%;
  text-align: center;
  width: 10%;
}

.demo3 {
  position: absolute;
  margin: 0 auto;
  left: 0;
  right: 67%;
  top: 95%;
  text-align: center;
  width: 10%;
}

.title {
  width: 100%;
  color: black;
  font-size: 20px;
  text-align:center;
  padding-top: 30px;
  padding-bottom: 10px;
}

.sentence {
  width: 50%;
  padding-top: 30px;
  padding-bottom: 10px;
  box-sizing: border-box;
  margin: 25%  ;
}
```

Input (form.js):

```
document.getElementById("form").onsubmit = function () { // form function

    var name = document.forms["form"]["name"].value; // variable for name
```

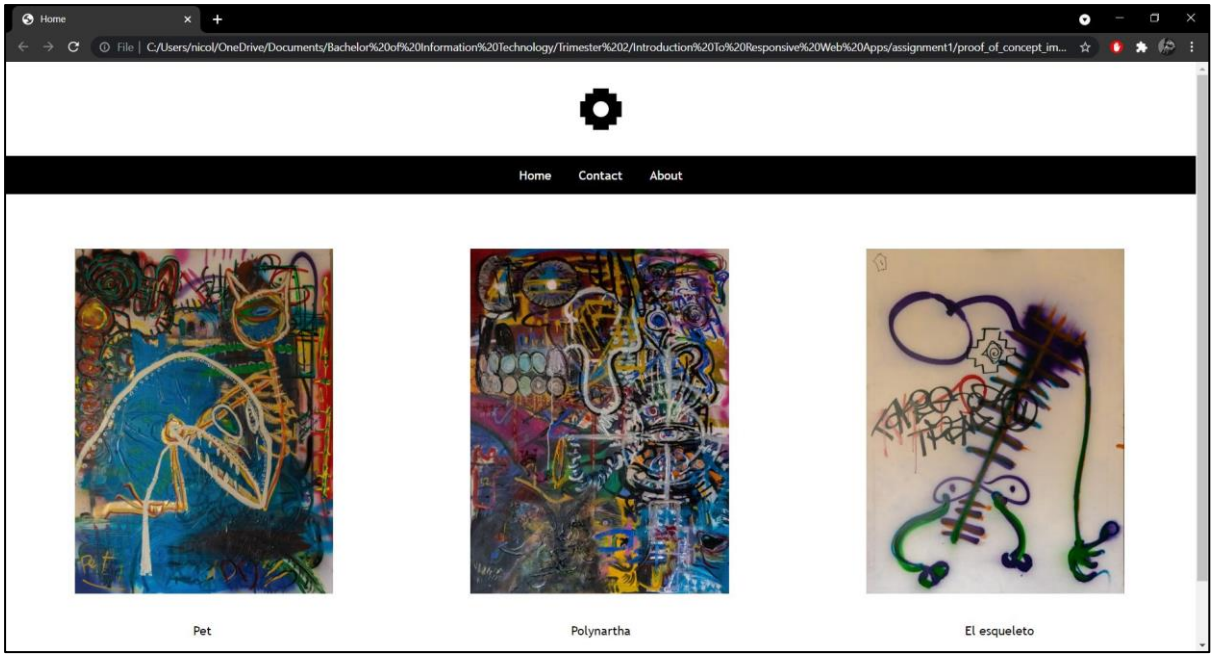
```
    var email = document.forms["form"]["email"].value; // variable for email
    var subject = document.forms["form"]["subject"].value; // variable for subject
    var message = document.forms["form"]["message"].value; // variable for message
    var pattern = /\S+@\S+\.\S+/; // variable for pattern
    var submit = true; // variable for submit


    if (name == null || name == "") { // if name is null or not null
        nameError = "Please enter your name"; // display error message for name
        document.getElementById("name_error").innerHTML = nameError;
        submit = false; // if submit is null do not submit
    }

    if (!pattern.test(email)){ // email pattern
        emailError = "Please enter your email"; // error message for email
        document.getElementById("email_error").innerHTML = emailError;
        submit = false; // if email is null do not submit
    }

    if (subject == null || subject == "") { // if subject is null or not null
        subjectError = "Please enter your subject"; // display error message for subject
        document.getElementById("subject_error").innerHTML = subjectError;
        submit = false; // if subject is null do not submit
    }

    if (message == null || message == "") { // if message is null or not null
        messageError = "Please enter your message"; // display error message for message
        document.getElementById("message_error").innerHTML = messageError;
        submit = false; // if message is null do not submit
    }
    return submit; // if all valid than returns submit
}

function removeWarning() { // remove warning messages once variables arent null
    document.getElementById(this.id + "_error").innerHTML = "";
}

document.getElementById("name").onkeyup = removeWarning; // remove warning message for name
document.getElementById("email").onkeyup = removeWarning; // remove warning message for email
document.getElementById("subject").onkeyup = removeWarning; // remove warning message for subject
document.getElementById("message").onkeyup = removeWarning; // remove warning messages for message
```

Input (vue.js):

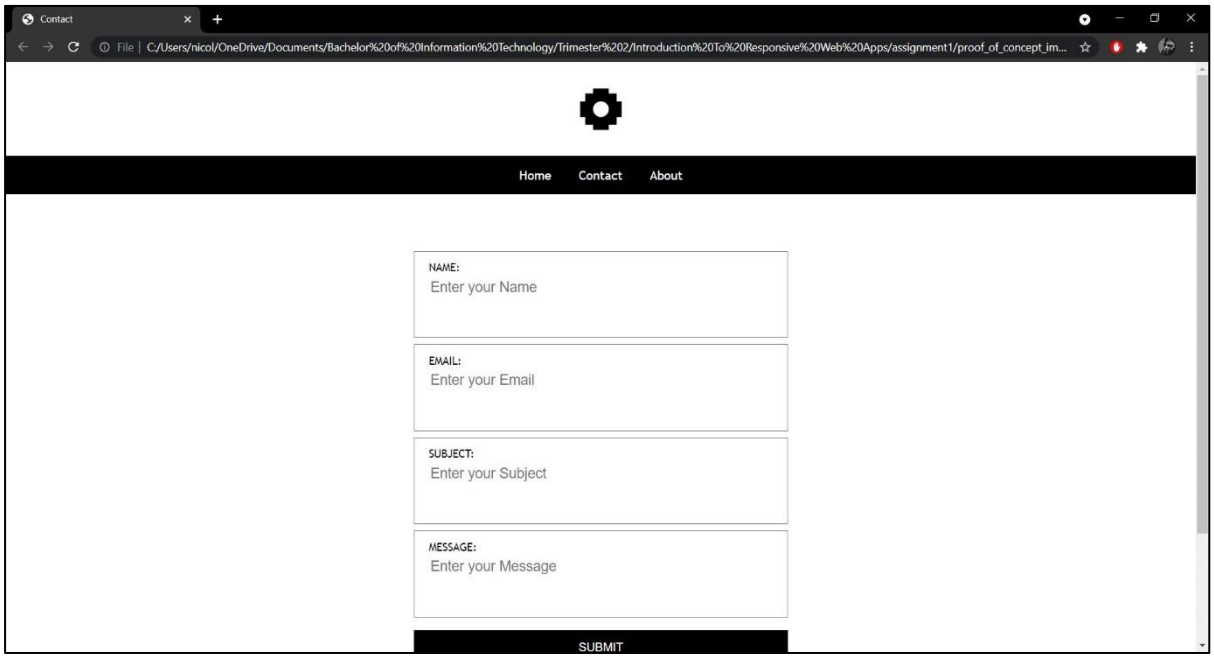
```
// variable called popup-component - handler #1
new Vue({
  el: '#prompt',
  methods: {
```

```
    say: function (message) {
      alert(message)
    }
  }
})

// Define a new component called title-component - prop #2
Vue.component('title-component', {
  data: function () {
    return {
    }
  },
  props: ['title'],
  template: '<h4>{{ title }}</h4>'
})

new Vue({ el: '#art-name' })

// Define a new component called heading-component #3
Vue.component('heading-component', {
   template : '<div><h1>Ernesto Vaquero</h1></div>'
})

new Vue({ el: '#artist-name' })

// Define a new component called anchored-heading-component - rendering #4
Vue.component('anchored-heading-component', {
  render: function (createElement) {
    return createElement(
      'h' + this.level,
      this.$slots.default
    )
  },
  props: {
    level: {
      type: Number,
      required: true
    }
  }
})
```

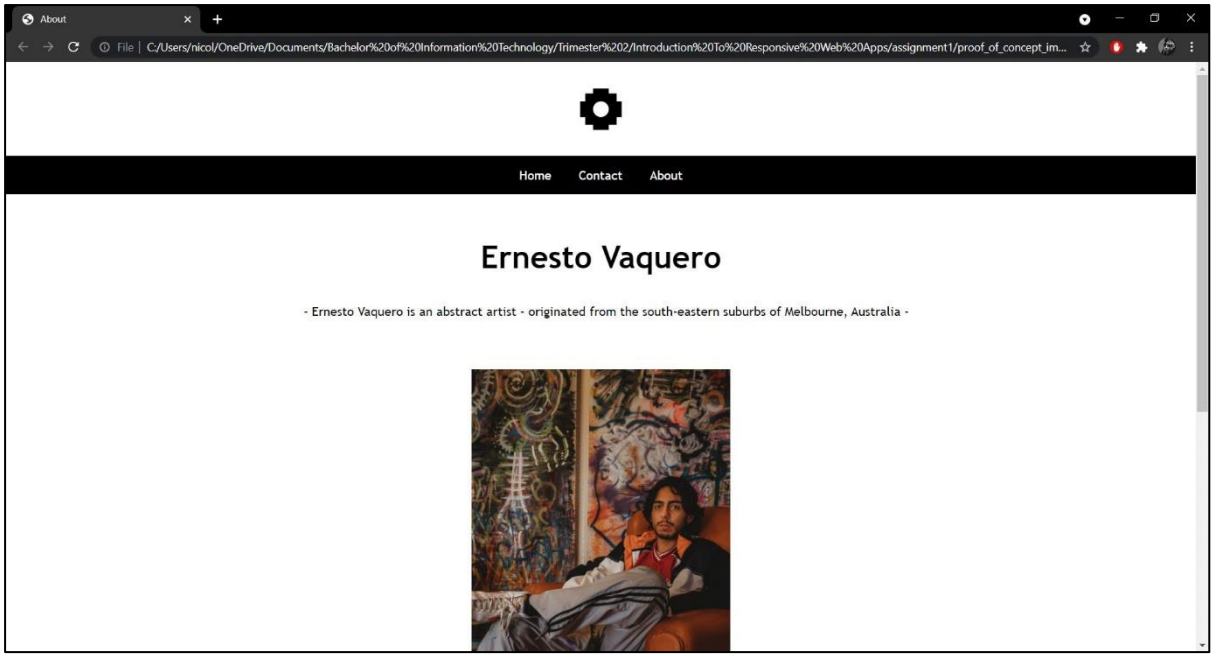
Output (home.html):



Output (contact.html):



Output (about.html):


About
File | C:/Users/nicol/OneDrive/Documents/Bachelor%20of%20Information%20Technology/Trimester%202/Introduction%20To%20Responsive%20Web%20Apps/assignment1/proof_of_concept_im...
Home
Contact
About
Ernesto Vaquero
- Ernesto Vaquero is an abstract artist - originated from the south-eastern suburbs of Melbourne, Australia -