**Reflective summary of the notes**

The week one of responsive web application unit explains some basic and important theories of HTML,CSS,GUI and UX, and also completed weekly practicals will be describes with those important basic theories.

HTML ( hypertext markup language )

Web pages are written in HTML, also referred to as HyperText Markup Language. A Web page with text, pictures, sound, and video can be made using HTML.

I was able to have a better understanding of the front end, back end, and full-stack developer web design sectors after finishing this weekly project. The presenter clearly explained the topic using examples. In the coming weeks, I want to deepen my understanding of responsive web design.

**Tags**

Tags are the foundation of HTML programming.

A tag is a keyword that is encased in angle brackets (for instance, "I").

Many tags, but not all, have opening and closing tags; Between the two tags is the text that is impacted.

**Narrow Tags**

When using HTML tags inside of other HTML tags, the closest tag must always be closed first.

For instance:

<H1> <I> The Nation (I) and (H1)

**Formatting Text**

• HTML text manipulation can be challenging; frequently, what you see is NOT what you get

what you receive

• To make paragraphs, go to the next line, and create headings, for instance, certain HTML tags are required.

**Tags for Text Formatting**

Strong Face

Italics

Underline

A fresh paragraph

After Line

**Headings**

• Sections with headings are frequently used to divide up web pages;

To design a heading use the phrase...

where n is a number in the range of 1 to 6

• In this instance, the 1 represents the heading with the largest size, while the 6

is equivalent to the smallest size

**Statements of Comment**

• HTML comment statements are notes that explain significant terms in the code.

components of the code

**Formatting Text**

• HTML text manipulation can be challenging; frequently, what you see is NOT what you get

what you receive

• To make paragraphs and move between them, for instance, certain HTML tags are required.

the next line, and add headers

**Tags for Text Formatting**

Strong Face

Italics

Underline

A fresh paragraph

After Line

**Headings**

• Sections with headings are frequently used to divide up web pages;

To design a heading use the phrase...

where n is a number in the range of 1 to 6

• In this instance, the 1 represents the heading with the largest size, while the 6

is equivalent to the smallest size

After this body comes which contains the main aspects of the code and finally have to close the html ( </html>) and end the code.

**CSS ( Cascading Style Sheets )**

Cascading Style Sheet, or CSS. A text file with an extension is a typical CSS file. css and consists of a number of instructions or rules. These guidelines specify how to show in HTML.

It is a language for style sheets that specifies how a webpage should appear. For web designers, CSS was developed to separate presentations from content, such as layouts, colors, and fonts. Your stuff may be more approachable thanks to this separation. There will be more freedom and control over how presentation properties are specified. Separately defining the necessary CSS A CSS file allows multiple web pages to share the same format, which lessens the redundancy and complexity of structural information. To speed up page loading between pages that share the file and its format, allow.css files to be cached.

**CSS Benefits**

* Separate the presentation's structure from its content.
* There is access to advanced presentation control.
* It's easy to handle several pages.
* It takes less time to load a page.
* improved accessibility for those with disabilities.
* By the third week, I was able to understand how to add a CSS file to an HTML page. I
* was able to comprehend external, internal, and inline styling better.
* External style sheet: This is a style sheet that controls how different pages are styled.
* A style sheet that controls the styling of a single page is an internal style sheet.
* Inline style guide: Instead of not using CSS, HTML tags directly use it.
* the heading; attached.
* It is easy to understand.

**HTML and CSS work together to produce beautiful and functional Web sites**

● HTML = structure

● CSS = style

**There are 3 ways to attach CSS to a page:**

1. External Style Sheet: Best used to control styling on multiple pages.

<link rel="stylesheet" type="text/css" media="all" href="css/styles.css" />

2. Internal Style Sheet: Best used to control styling on one page. <style type=“text/css”> h1 {color: red} </style> Attaching a Style Sheet

3. Inline Style Sheet\*: CSS is not attached in the <header> but is used directly

within HTML tags.

<p style=“color:red”>

Some Text</p>

**GUI**

The actual buttons, text, and pixels that show up on the screen are referred to as GUI.

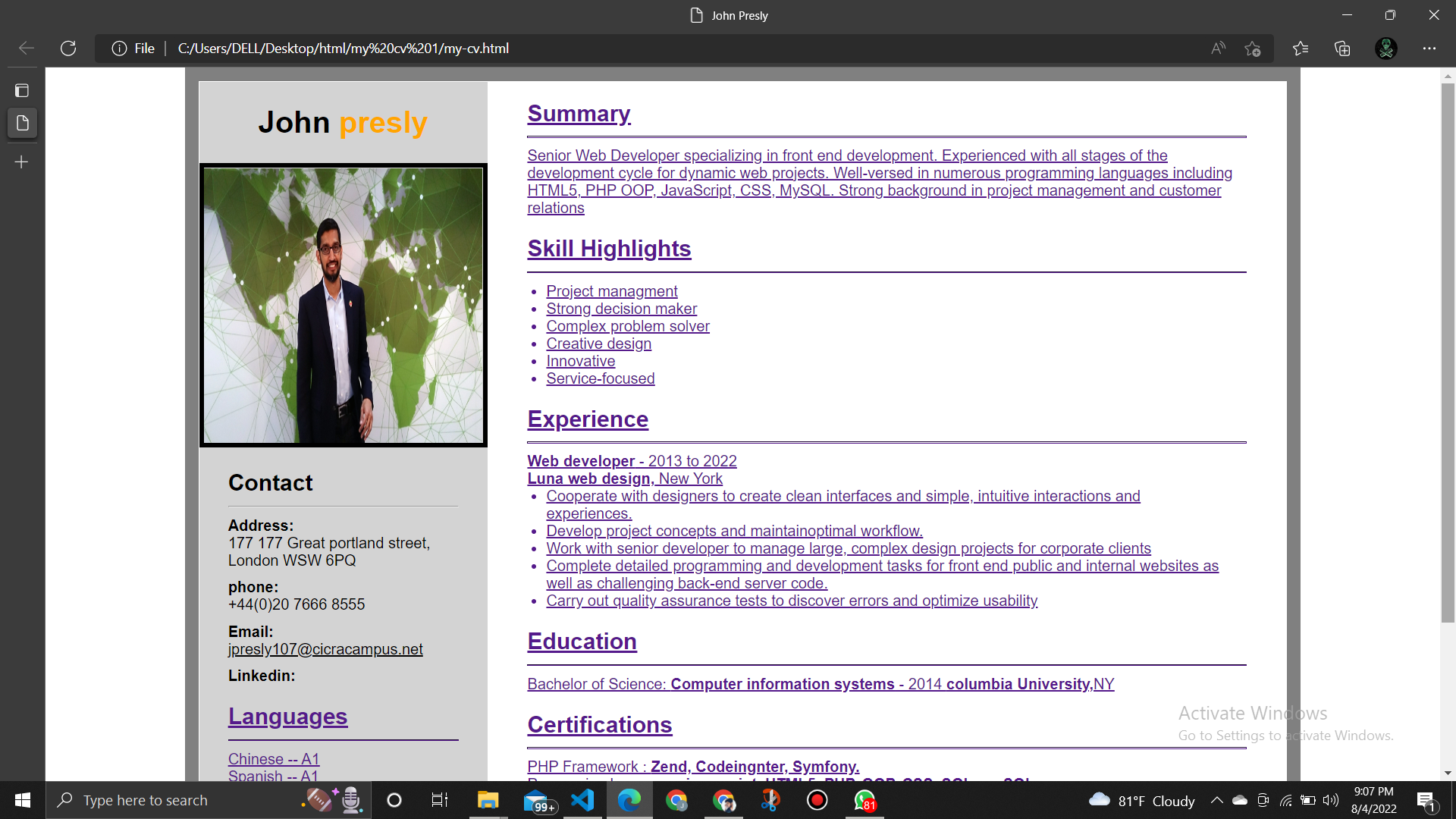
Everything that pertains to how a website or application appears in the interface would be the responsibility of the GUI designer. This refers to both the page's graphics and general design. For instance, where should the "Sign Up" button be placed?

**UX**

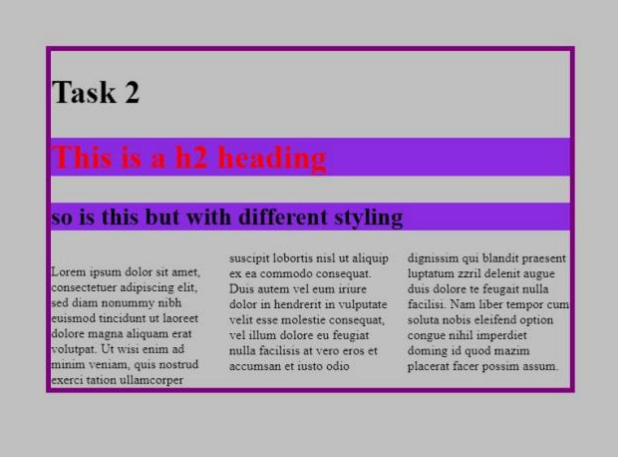
Because it covers all the systems and interactions that support the interface in addition to the interface itself, this discipline is more larger and more advanced than GUI. As an illustration, the UX designer should be concerned with both the appearance of pixels on the screen and what occurs when a frustrated user calls the help desk.

**Practicals**

My-cv

  
  
this indicates a well user friendly and responsive web page.

**Excercis 2**



this indicates a well user friendly, responsive web page with heading 2 type.

GITHUB LINK :- <https://github.com/jpcicra/SIT120-RWA-Assignments.git>

**Document Object Model (DOM)**

The programming environment for online content is called Document Object Model (DOM). Programs can alter the document's content, appearance, and structure by using it as a page representation. The document is represented by nodes and objects in the Document Object Model (DOM), which can be interacted with through computer languages.

A web page is a document with HTML source code that may be downloaded or read in a browser window. The content in all instances is the same, but thanks to the Content Object Reference Model (DOM) representation, customisation is possible. It is an object-oriented representation of a website page, so it can be altered with the aid of a programming language like Java.

**java script (.js)**

**Scripting or programming language used today on the Web.**

It is used to enhance the functionality and appearance of web pages an also JavaScript is a client-side scripting language - Because JavaScript is interpreted by a browser, it is considered to be a client-side scripting language.

JavaScript is an object-based programming language and has pre-defined a lot of objects that can be directly used to keep/obtain information (via object properties) and perform operations (via object methods), Furthermore, JavaScript also considers the elements of a web page as objects which can be identified and manipulated by JavaScript.

Case sensitive.

**Implement client-side scripts**

JavaScript was initially intended to be a client-side scripting language. JavaScript is a client-side programming language used by many websites. JavaScript is a scripting language used by web designers to build unique client-side scripts that work seamlessly with HTML. JavaScript can also be used to increase user interaction, speed up user response times, and create richer user interfaces on websites while reducing server activity and load.

**Three ways of using JavaScript in a web page**

**External JavaScript** – External JavaScript are scripts that are written and saved in a separate document

**Internal JavaScript** - Internal JavaScript are scripts enclosed between the tag <script> and <script> and embedded within the head or body section of the web documents.

**Inline JavaScript** – Included within the values of attributes (Event Handler)

**There are three kinds of declarations in JavaScript.**

**Var**

Declares a local or global variable (depends on where it is defined). Optionally initialize it to a value – DO NOT USE IT.

**Let**

Declares local variable, optionally initialize it to a value

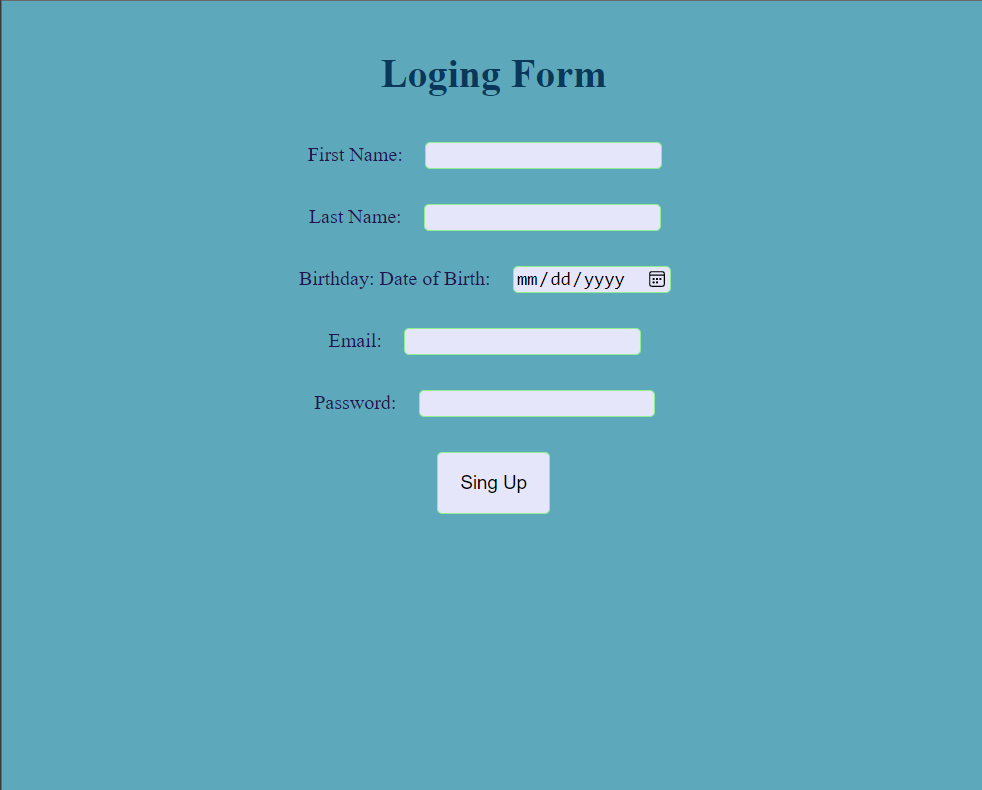
**Const**

Declares a real-only named constant.

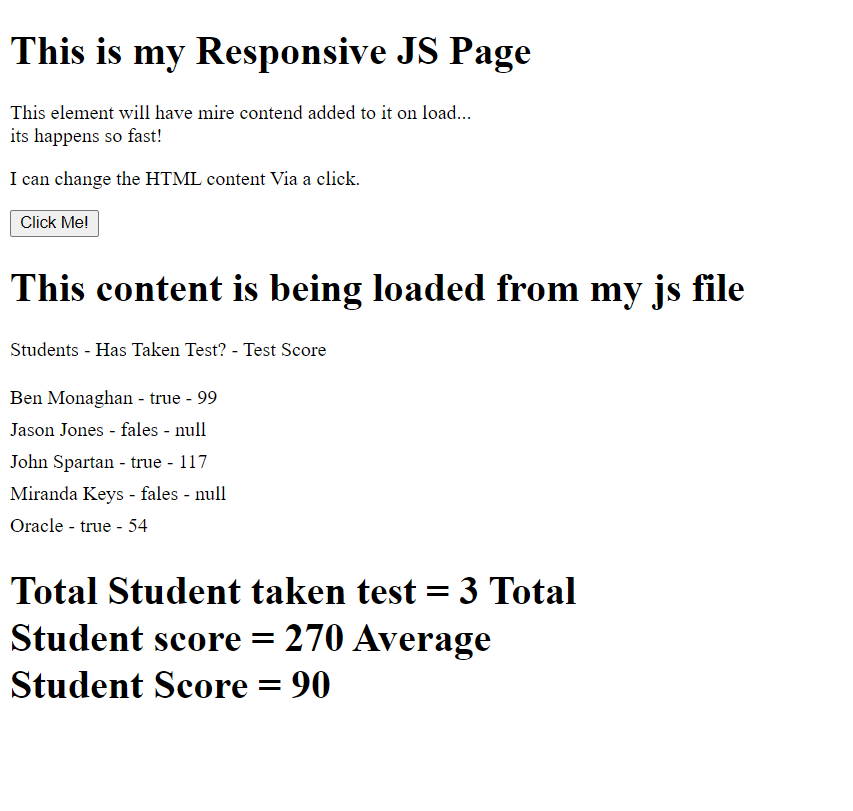
**Java script has similar aspect of which some other object oriented programing languages has such as comments, data types, logical operators, control structures, if for while and do wile statements, arrays and etc**

**Practicals**

**Login form .**

****

**Exercise 2**

****

**Vue Framework**

• JavaScript framework for building user interfaces

• provides a declarative and component-based programming model that aids in the speedy development of user interfaces, whether they are basic or complicated. It builds on top of standard HTML, CSS, and JavaScript.

**Core features of Vue :**

• Declarative Rendering: With the help of a template syntax that Vue adds to normal HTML, we can declaratively describe HTML output based on JavaScript state.

• Reactivity: When JavaScript state changes, Vue automatically tracks them and quickly updates the DOM.

**Vue is designed to be flexible and incrementally adoptable**

• Depending on your use case, Vue can be used in different ways:

• Enhancing static HTML without a build step

• Embedding as Web Components on any page

• Single-Page Application (SPA)

• Fullstack / Server-Side Rendering (SSR)

• Jamstack / Static Site Generation (SSG)

• Targeting desktop, mobile, WebGL, and even the terminal

**Vue Built-in Directives**

• V-show: On the basis of the expression value's truthiness, change the element's visibility.

• similar to that of v-if almost exactly, however there is no "alternative" condition.

**Computed Properties**

• You can use computed

properties to calculate

and display values based

on a value or set of values

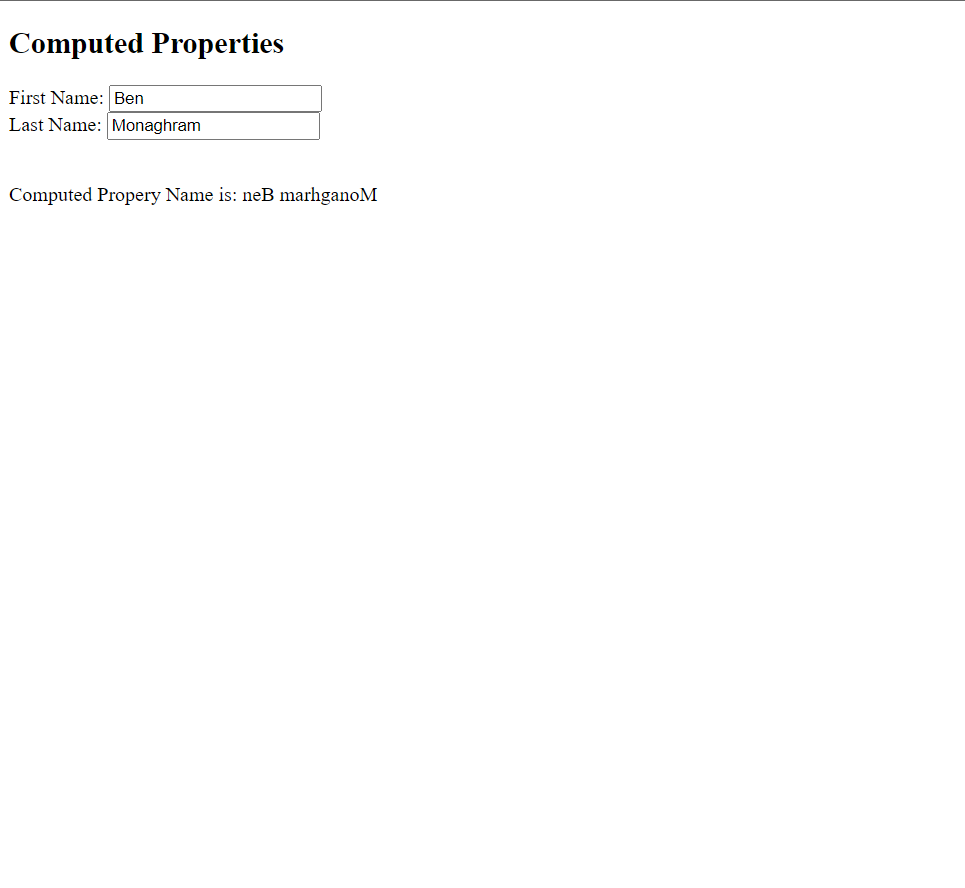
in the data model.

**Practicals**

**Tip calculator**

****

**Computed properties**

****

**Github link :-** [**https://github.com/jpcicra/SIT120-RWA-Assignments.git**](https://github.com/jpcicra/SIT120-RWA-Assignments.git)

**User Input Handling with Vue**

V- model directive is used to create two-way data bindings on form

input, textarea, and select elements

• It automatically picks the correct way to update the element based

on the input type.

• V-model will ignore the initial value, checked, or

selected attributes found on any form elements. It will always treat

the Vue instance data as the source of truth. You should declare the

initial value on the JavaScript side, inside the data option of your

component

**V-model**

V-model internally uses different properties and emits different

events for different input elements:-

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

**Practicals**



**Github link :-** [**https://github.com/jpcicra/SIT120-RWA-Assignments.git**](https://github.com/jpcicra/SIT120-RWA-Assignments.git)

**Transition Effects**

• Vue transition apply automatic transition effects when elements are inserted into or removed from the DOM

**There are three options to implement transitions with Vue.js:**

• CSS transitions

• CSS animations

• JavaScript functions

**Benefits of transitions and animations**

• It helps the user navigate the app by understanding connections

between elements.

• It let’s the user know what’s happening by giving feedback in

response to an action.

• It can draw attention to new or essential features and elements.

• It can encourage user engagement.

• It creates a unique and memorable user experience.

**The <transition> component**

• Transition element is a wrapper that helps you add transition

functionality to your elements

• Used to apply enter and leave animations on elements or

components passed to it via its default slot.

**• The enter or leave can be triggered by one of the following:-**

* Conditional rendering via v-if
* Conditional display via v-show
* Dynamic components toggling via the <componenr> special element

**Transition classes**

* v-enter-from / v-leave-form start state of the transition; removed once transition starts (in vue2 thi is just v-enter and v-leave )
* v-enter-active / v-leave-active: active state of the transition
* v-enter-to / v-leave-to end state of the transition

**Vue Components**

• Components allow to split the UI into independent and reusable

pieces, and think about each piece in isolation.

• It's common for an app to be organized into a tree of nested

components:

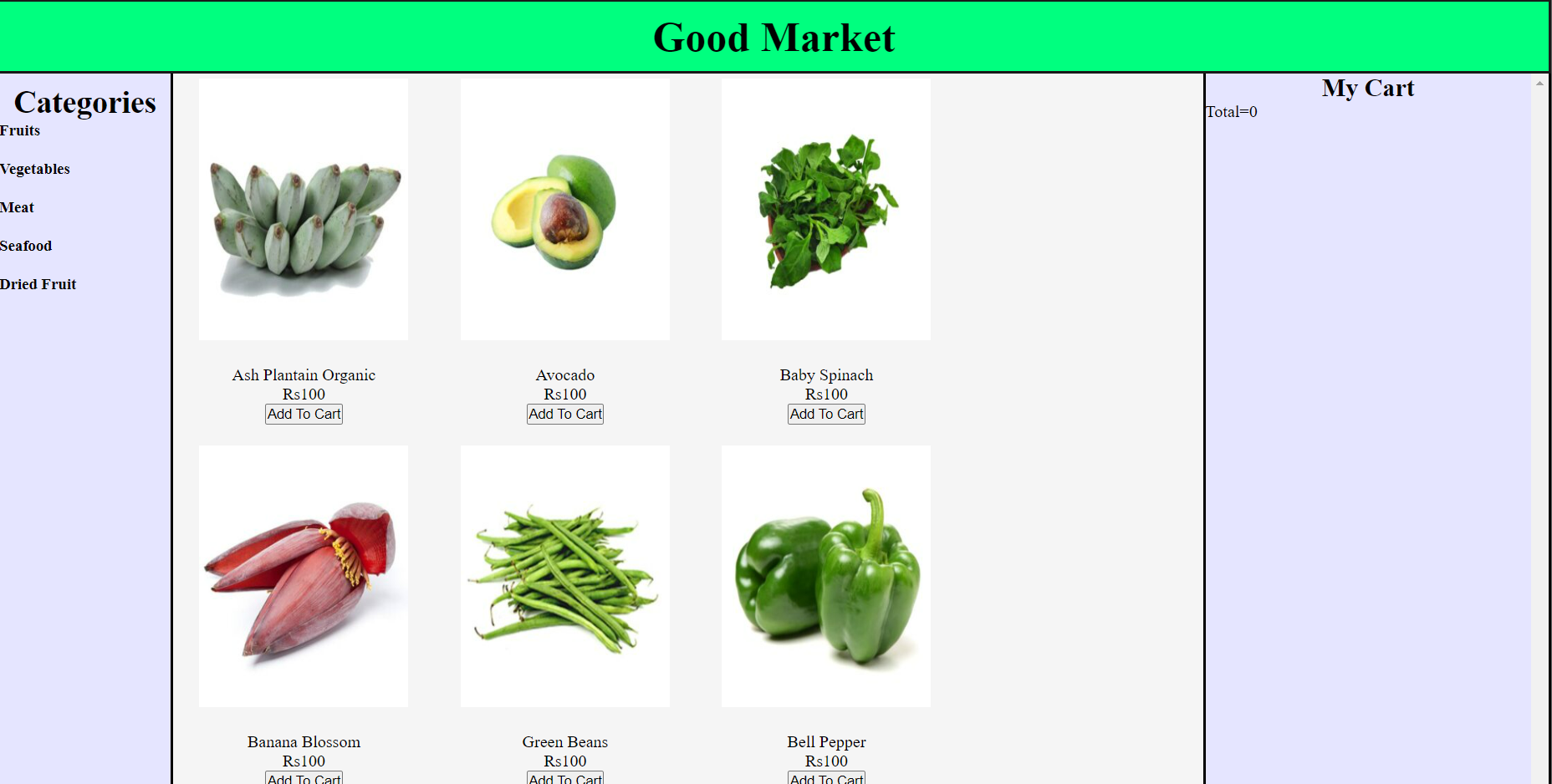
**Dynamic Components**

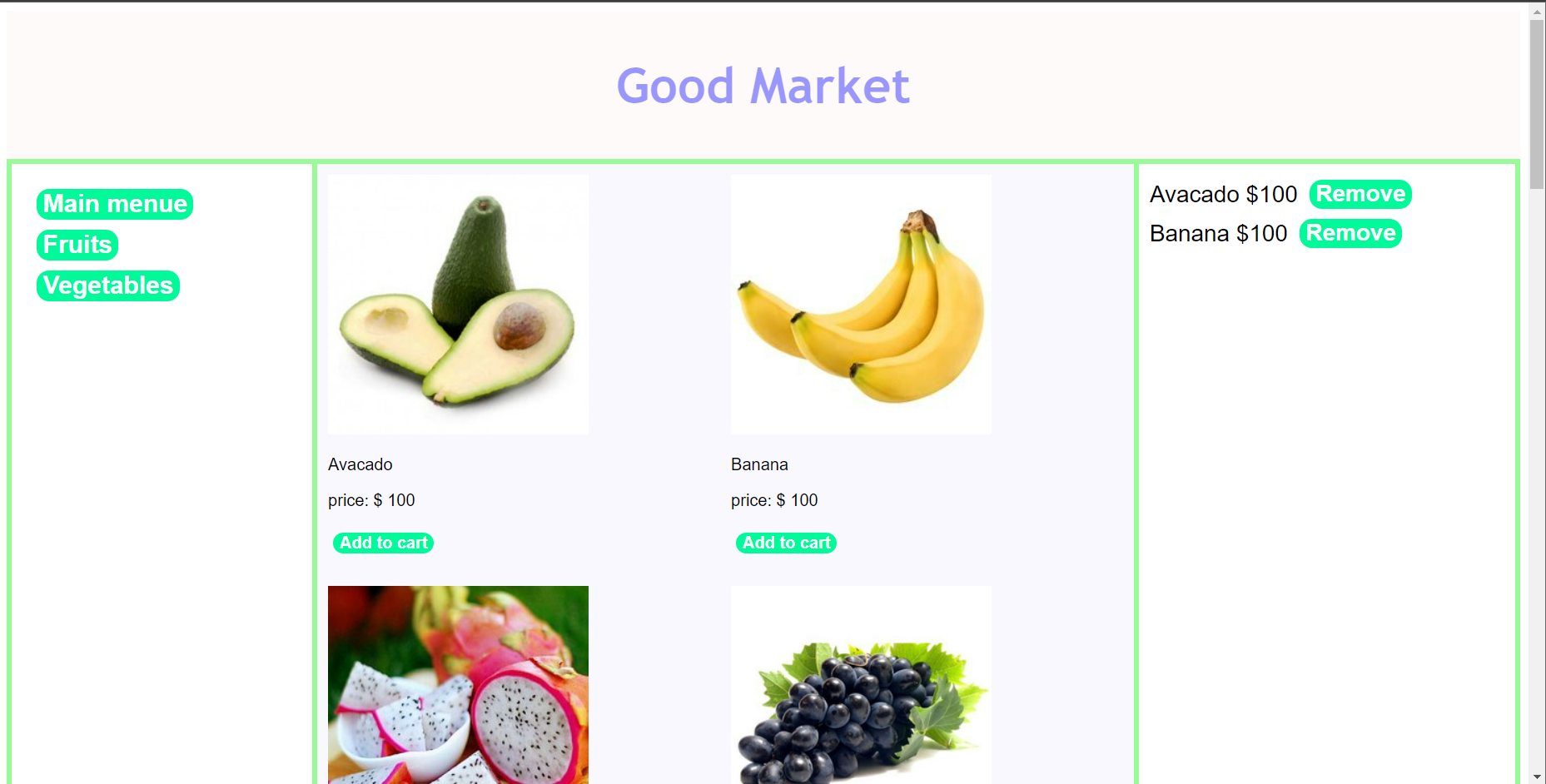
• Dynamic components are created using the

keyword <component></component> and it is bound using a

property

**practicals**

****



**Github link :-** [**https://github.com/jpcicra/SIT120-RWA-Assignments.git**](https://github.com/jpcicra/SIT120-RWA-Assignments.git)