Tutorial/Laboratory 02

Styling and Responsive Design using CSS & Bootstrap

1. LEARNING OUTCOMES

Upon completion of these laboratory exercises, you should be able to:

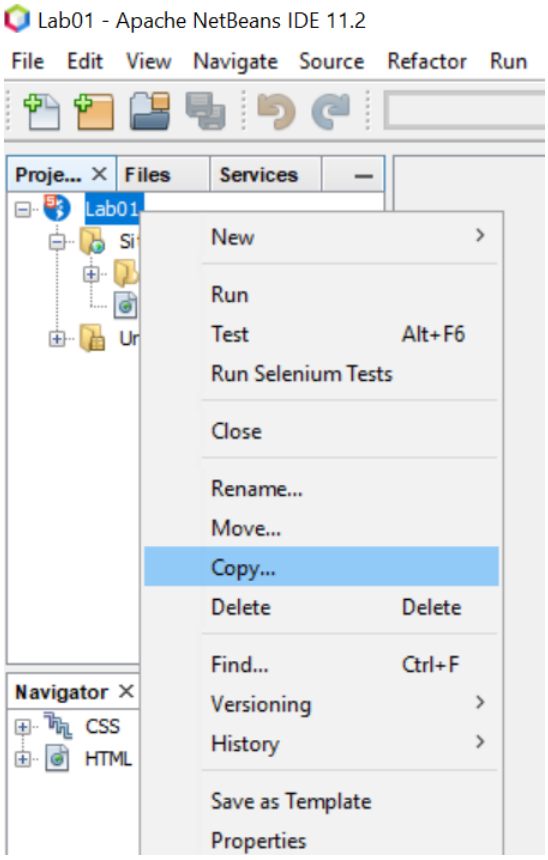
- Style web pages using CSS and understand the advantages of external style sheets.
- Implement the Bootstrap framework and build responsive, attractive websites.
- Understand positioning and grid layout concepts.

2. REQUIRED SOFTWARE

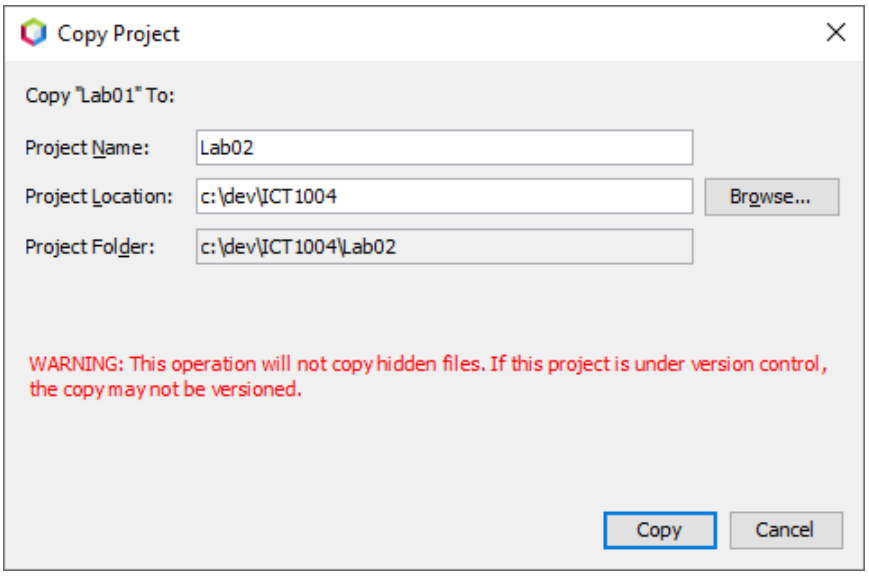
- Apache NetBeans 11.2 (or later): https://netbeans.apache.org/download/index.html
- FireFox (https://www.mozilla.org/en-US/firefox/new/) or Chrome (https://www.google.com/chrome/) web browser.
- Bootstrap 4: https://getbootstrap.com/

3. INITIAL SETUP

3.1 In this practical assignment, we'll continue with the *World of Pets* website that we created in Lab01. Therefore, you should first make a copy of the Lab01 project in NetBeans and rename it to Lab02 (or similar). You can either copy from the file system, or you can do it easily from within the NetBeans IDE by right-clicking on the project name and selecting **Copy...** from the menu:



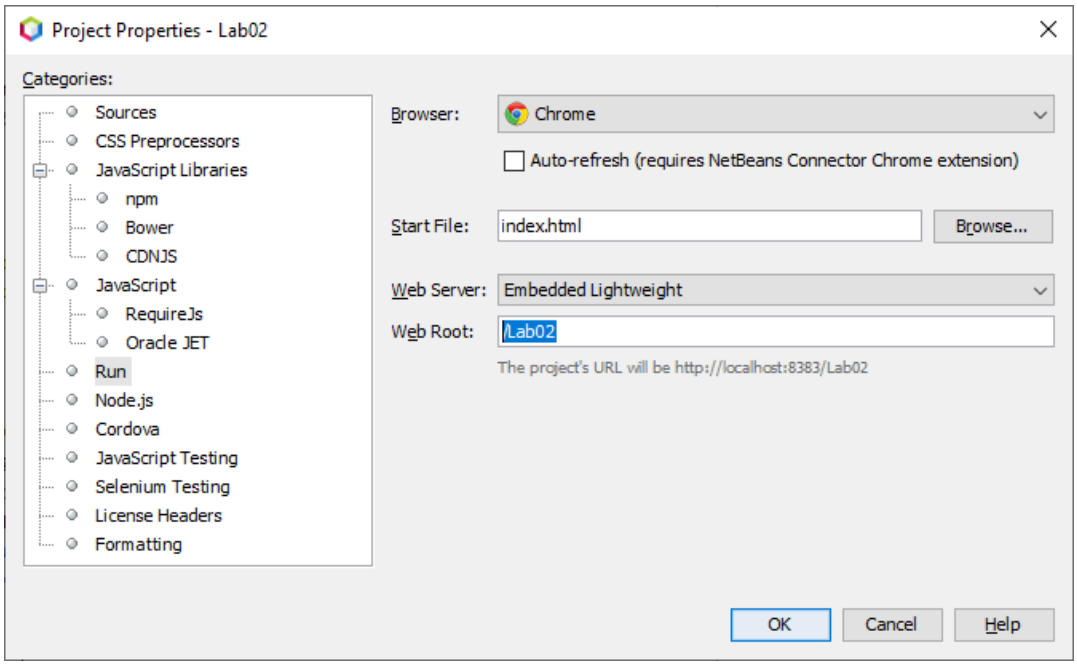
Then in the Copy Project dialog, enter the new name and location. Remember, do NOT use spaces or special characters when naming your projects, as this becomes part of the URL when you run or publish to the web server.



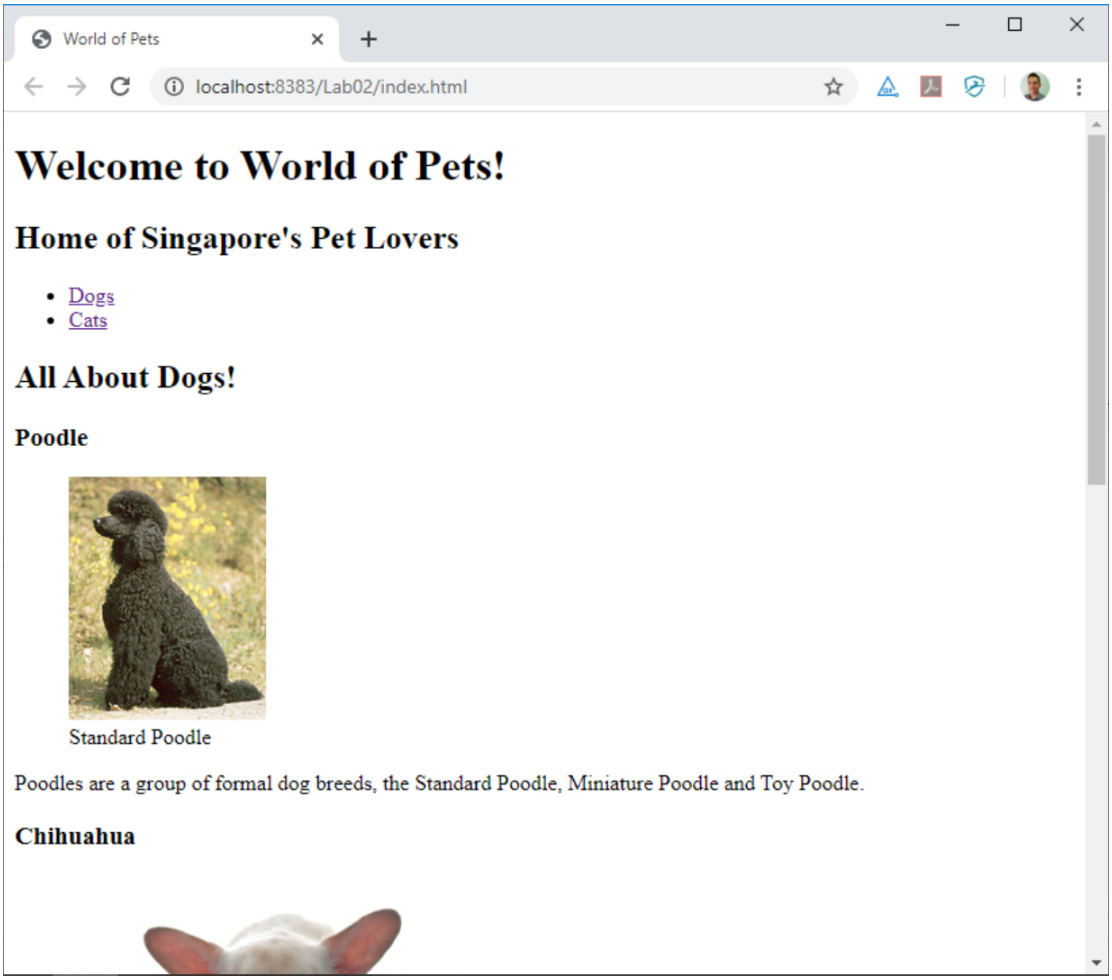
Click **Copy** to complete the operation.

3.2 **IMPORTANT**: when you copy a project, the project settings are not carried over properly to the new project, so we need to edit the **Run** configuration for the newly

copied project, just like we did in Lab01. Choose all the same options as before, and be sure to change **Web Root** setting to **/Lab02** instead of **/Lab01**:



3.3 Test the new project by clicking the green **Run** button, and let's look at the website as it is now:



If we were living in the year 2000, this website wouldn't be all that bad, but since this is 2020, we need to do much better. Namely:

1. Make the site more aesthetically pleasing. This is exactly what CSS is for.
2. Make it responsive and viewable on any size device. This means automatically repositioning and/or resizing elements intelligently depending on the screen size.
3. Implement a proper navigation menu.

In the following exercises we'll utilize CSS and the Bootstrap framework to achieve all of the above.

4. EXERCISE 1: INSTALLING BOOTSTRAP

4.1 Our first step is to get the Bootstrap CSS code into our website. There are two ways to do this: 1) download a copy of the Bootstrap files and link to them locally, or 2) link to the Bootstrap files online using a content delivery network (CDN). Here are some pros & cons of each method for you to consider:

Method	Pros	Cons
CDN	• Faster loading due to caching. • Easy to select desired versions. • Up to date with latest bug fixes. • Saves local space and bandwidth.	• Requires more vigilance due to security risks, e.g. hackers taking over the CDN and injecting malicious script. • Must always be connected to the internet (no off-line development). • IDEs such as NetBeans may not be able to provide syntax checking on non-local files.
Local	• Can be read and parsed at build time by IDEs, allowing syntax highlighting and hints. • Can be used offline. • Easily viewed by developers for educational or debugging purposes. • Eliminates external dependencies, i.e. website will still work even if connection to CDN goes down for some reason.	• Additional overhead to downloaded and maintain files locally. • Takes up disk space on your server. • Load times can be slower, especially as web traffic scales up.

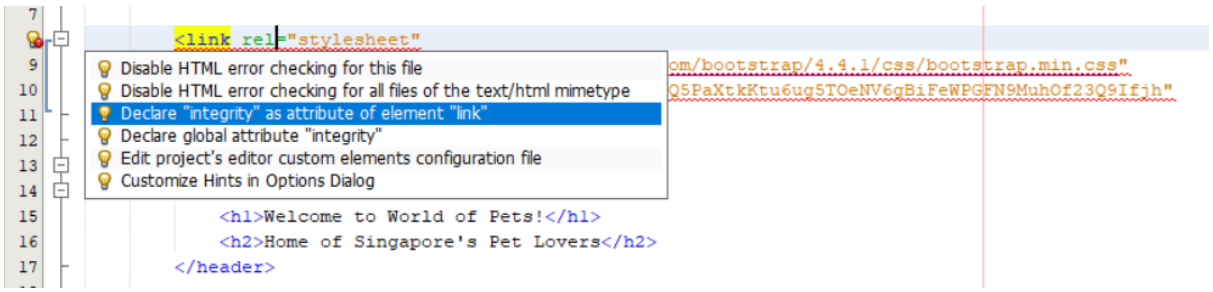
For the exercises in this Lab, we'll use the CDN method to keep things simple. However, for your projects, you are free to choose whichever method you feel is best.

Note: there is also a downloadable Site Template in NetBeans that lets you add the Bootstrap files during project creation, however this contains an outdated version of Bootstrap and is not recommended.

4.2 Copy the following code into the <head> element in index.html:

```
<link rel="stylesheet"
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
    integrity=
        "sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
    crossorigin="anonymous">
```

You'll notice that NetBeans 11.2 does not recognize the relatively new 'integrity' attribute and will mark this as a syntax error. You can fix this by clicking on the element and hitting ALT-ENTER, then selecting **Declare "integrity" as attribute of element "link"**:



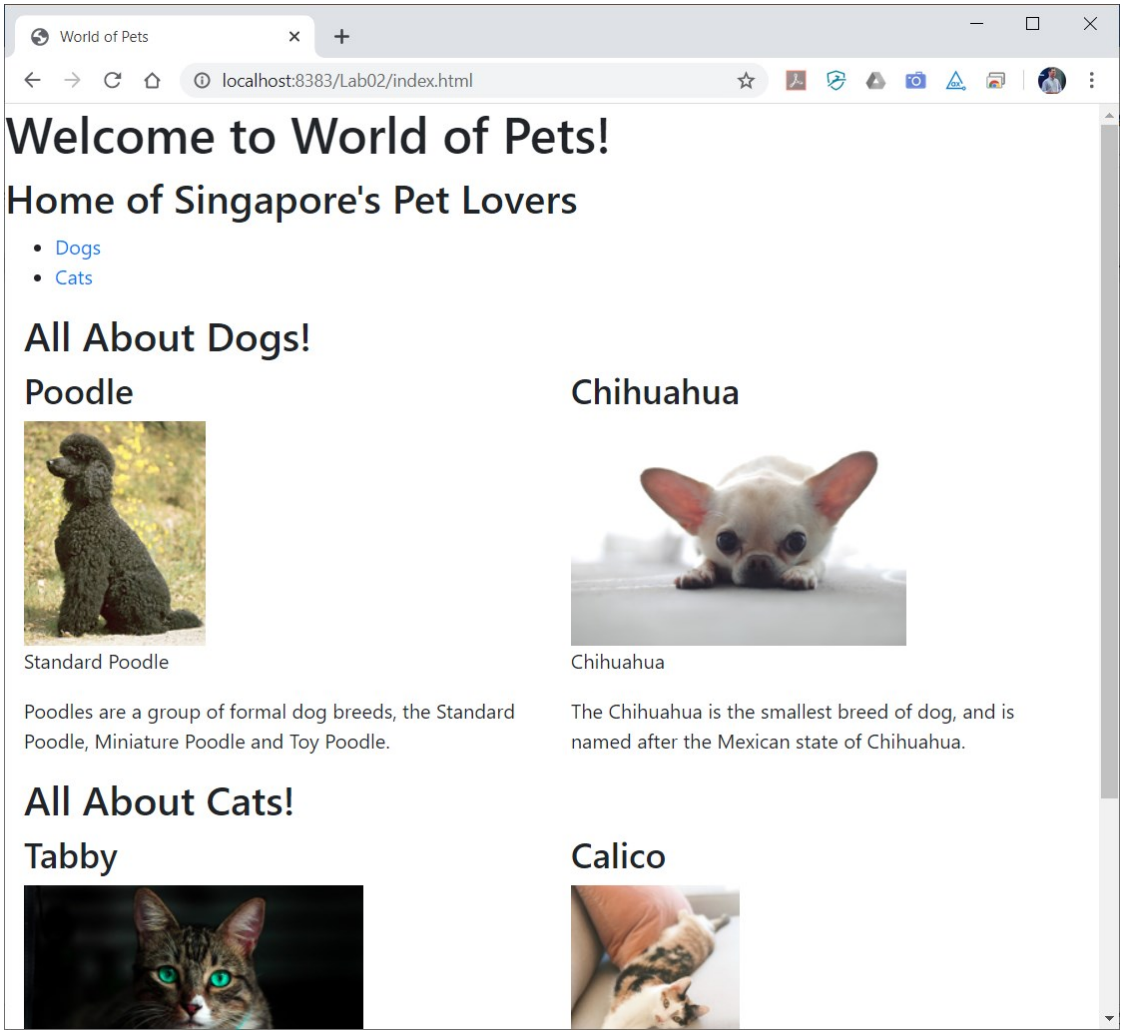
The purpose of the 'integrity' attribute is to tell the browser to check the source file (bootstrap.min.css in this case) against the given hash value to be sure the file has not been altered. As mentioned, the use of CDNs requires extra effort to protect against malicious attacks.

4.3 After linking to Bootstrap CSS, you should notice that the website already looks different, as the default browser fonts and formatting have been replaced by Bootstrap defaults.

5. EXERCISE 2: RESPONSIVE DESIGN

5.1 Now that we have Bootstrap in our project, we can easily implement a responsive layout simply by adding Bootstrap class attributes to the elements. We'll be using Bootstrap's Grid system, which you can learn more about here: https://getbootstrap.com/docs/4.4/layout/grid/.

5.2 For our design, we'll divide the sections into rows, then within each row, the articles are laid out in auto-sizing columns, like this:



5.3 To accomplish this, start by adding the **container** Bootstrap class to our <main> element:

```
<main class="container">
```

This makes the entire <main> element a Bootstrap container, which is required for the grid layout.

5.4 Next, we add the rows for each section. For this we can use elements (this is an example of when using instead of a semantic element is perfectly OK, since it's for styling/design only and has no semantic meaning). Add the following element after the <h2> element and before the first <article> element:

```
<div class="row">
```

Then be sure to add the closing tag just before the closing </section> tag.

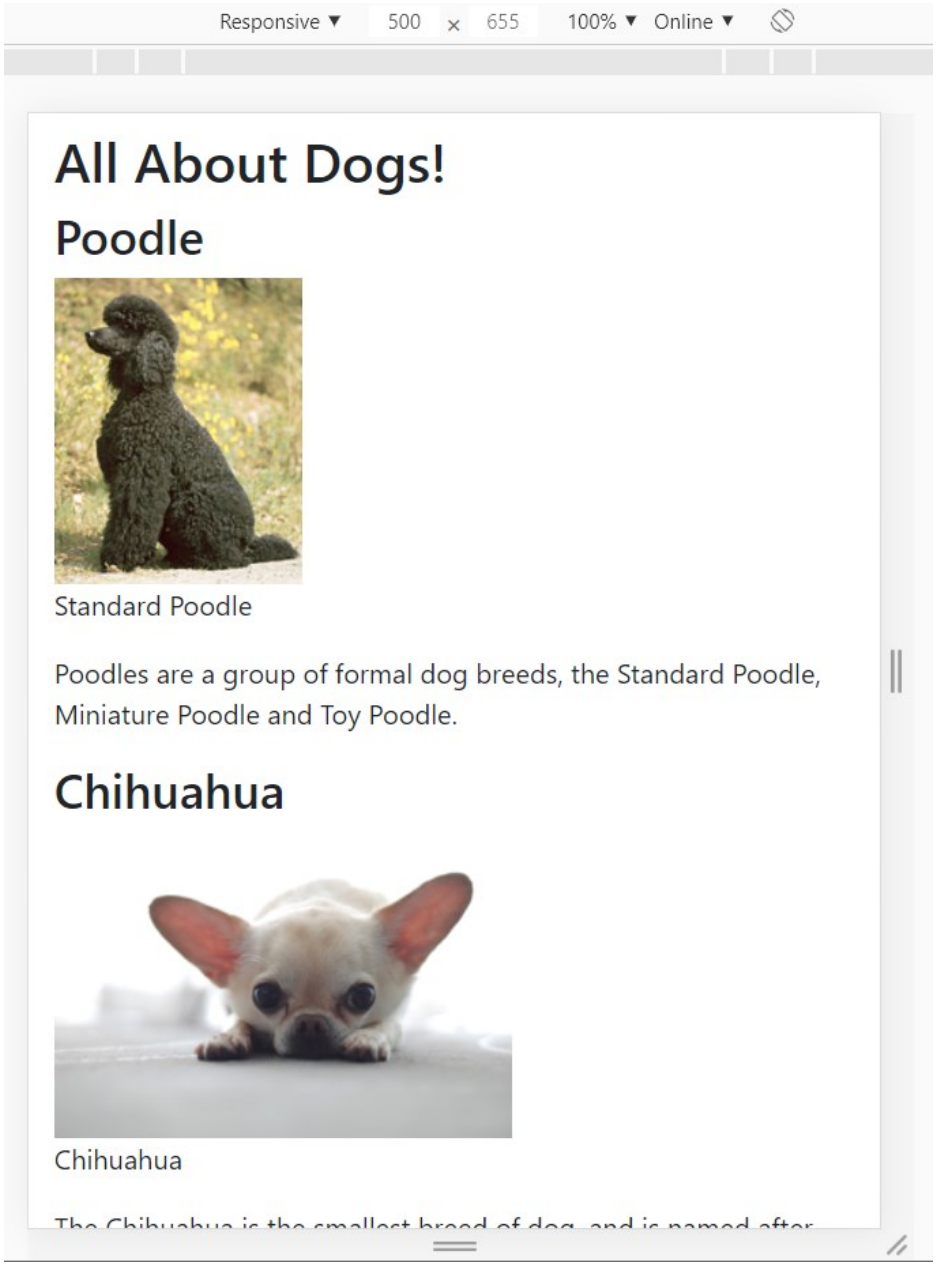
5.5 Now to add the columns, we simply add the **col-sm** class to each of the <article> elements:

```
<article class="col-sm">
```

Here's how your entire first <section> element (dogs) should look after the above changes:

```
<section id="dogs">
    <h2>All About Dogs!</h2>
    <div class="row">
        <article class="col-sm">
            <h3>Poodle</h3>
            <figure>
                <a href="images/poodle_large.jpg">
                    <img src="images/poodle_small.jpg" alt="Poodle"
                         title="View larger image..."/>
                </a>
                <figcaption>Standard Poodle</figcaption>
            </figure>
            <p>
                Poodles are a group of formal dog breeds, the Standard
                Poodle, Miniature Poodle and Toy Poodle.
            </p>
        </article>
        <article class="col-sm">
            <h3>Chihuahua</h3>
            <figure>
                <a href="images/chihuahua_large.jpg">
                    <img src="images/chihuahua_small.jpg" alt="Chihuahua"
                         title="View larger image..."/>
                </a>
                <figcaption>Chihuahua</figcaption>
            </figure>
            <p>
                The Chihuahua is the smallest breed of dog, and is named
                after the Mexican state of Chihuahua.
            </p>
        </article>
    </div>
</section>
```

The **col-sm** class tells Bootstrap that we wish to distribute the columns horizontally for large and medium screens, then when the small breakpoint is hit, the articles will flow vertically instead, so that the webpage still looks good on small devices such as mobile phones. You can test this using Chrome's device toolbar (Developer Mode):



5.6 Now implement the same modifications to the second <section> element (cats). Our main content is now fully responsive. We'll work on the header and footer elements in the next exercise.

6. EXERCISE 3: HEADER, MENU AND FOOTER

6.1 To make the web page stand out, we'll use Bootstrap's **jumbotron** class. We'll also use the **display** class to make our <h1> heading more attractive. Modify the <header> element as follows:

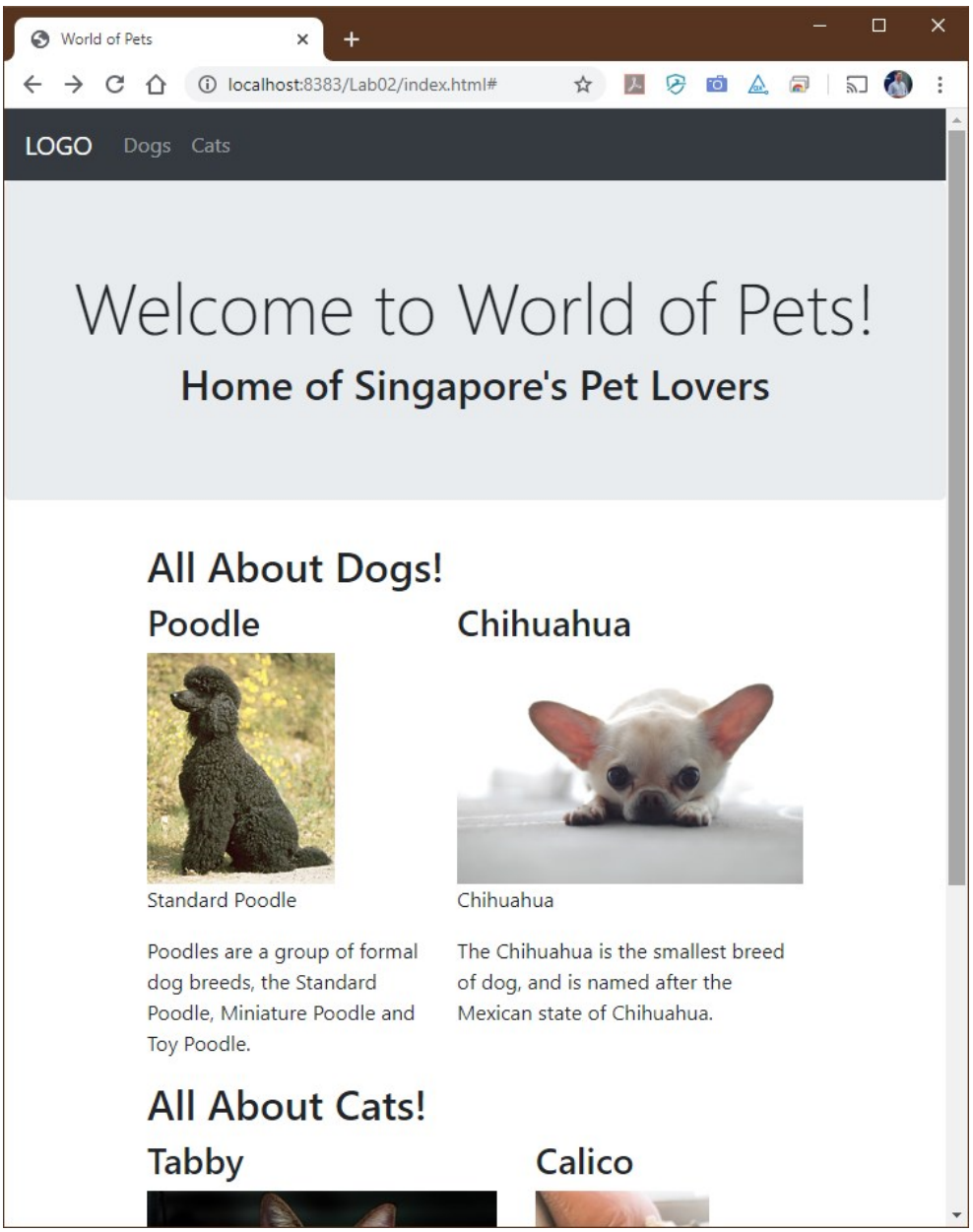
```
<header class="jumbotron text-center">
    <h1 class="display-4">Welcome to World of Pets!</h1>
    <h2>Home of Singapore's Pet Lovers</h2>
</header>
```

6.2 Next, we'll add a Bootstrap style menu. For now, we'll just use a simple responsive menu that flows vertically at the small break point, but after we introduce JavaScript in a future Lab, we'll replace this with a collapsible mobile menu.

Move the existing <nav> element above the <header> element (just after the opening <body> tag), then add Bootstrap classes as follows:

```
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
    <a class="navbar-brand" href="#">LOGO</a>
    <ul class="navbar-nav">
        <li class="nav-item">
            <a class="nav-link" href="#dogs">Dogs</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="#cats">Cats</a>
        </li>
    </ul>
</nav>
```

Your web page should now look something like this:



Feel free to experiment with different Bootstrap classes.

6.3 **Challenge:** try replacing the text LOGO with an image of your choice.

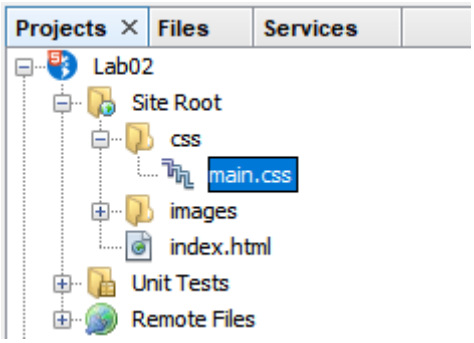
6.4 Lastly, add the **container** class to the <footer> element so that it aligns with our main content. You can also remove the tags if any - we'll restyle the footer in the next exercise.

```
<footer class="container">
    <p>Copyright &copy; 2020 World of Pets Pte. Ltd.</p>
</footer>
```

7. EXERCISE 4: CUSTOM STYLING WITH CSS

7.1 We've greatly improved our website by making it responsive and leveraging Bootstrap's built-in styles. In this exercise, we'll add some of our own custom styles to make the web page look even better.

7.2 First create a sub-folder called **css** under Site Root. Then create a new .css file (New->Cascading Style Sheet…) called main.css:



7.3 Add the appropriate <link> element to link main.css in index.html:

```
<link rel="stylesheet" href="css/main.css">
```

7.4 Open main.css in the editor and add the necessary CSS rules to achieve all of the following requirements:

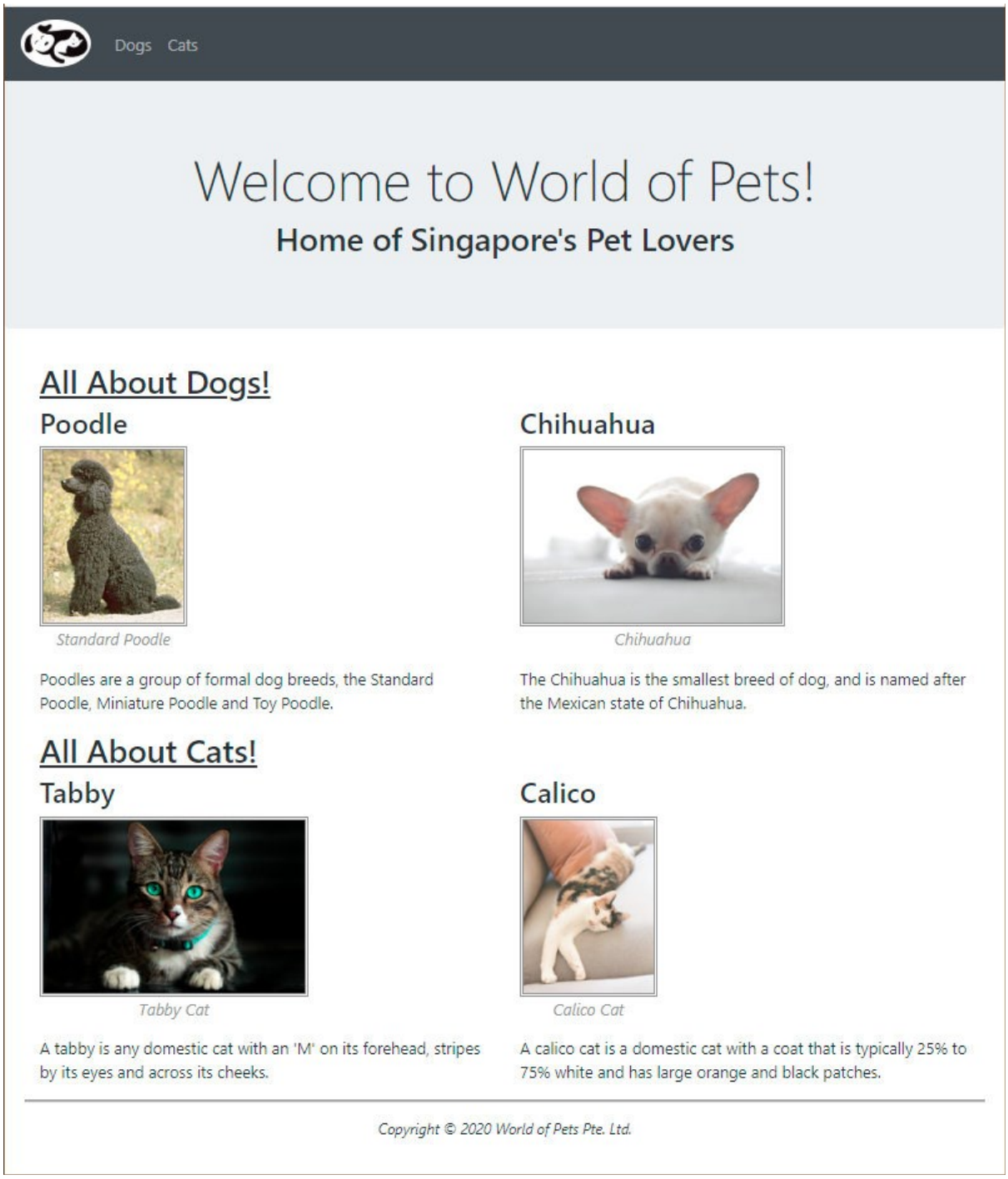
a. Section headings (<h2>) should be underlined.

b. Article images should have a double "thumbnail" style border.

c. Image captions should be gray and italicized, and centered relative to the image.

d. The footer should have a top border with content centered and italicized. The font should also be slightly smaller than the page's default font.

You may refer to the W3Schools CSS tutorial to help you in this exercise. The final web page should look something like this:



8. SUBMISSION OF LAB ASSIGNMENT

8.1 In order to receive credit for this Lab assignment, you must submit your completed work to xSiTe LMS before the end of the Lab session. To submit your work:

a. Save all files and close NetBeans.

b. In File Explorer, navigate to the location where you saved your project and right-click on the folder name, then select 'Send to -> Compressed (zipped) folder' and

ZIP up your entire NetBeans project. **Note: only .zip format is acceptable, do not use .rar, .7z, or any other format.**

c. In the ICT1004 module on xSiTe, go to **Assessments->DropBox** and locate the Dropbox folder corresponding to this Lab. Click the link to open the Dropbox then hit the **Add a File** button to submit your .zip file. You may also add comments if desired. Be sure to hit **Submit** to complete your submission.

d. Remember to save a copy of your work as we will be building upon this website in subsequent Lab assignments.

9. ADDITIONAL PRACTICE

9.1 Once you've completed this Lab assignment, you are encouraged to try out the following online tutorials:

a. CSS (https://www.w3schools.com/css/css_exercises.asp)

b. Bootstrap 4 (https://www.w3schools.com/bootstrap4/default.asp)