Project 1

In this project, you'll begin your learning journey starting with basic HTML and CSS.

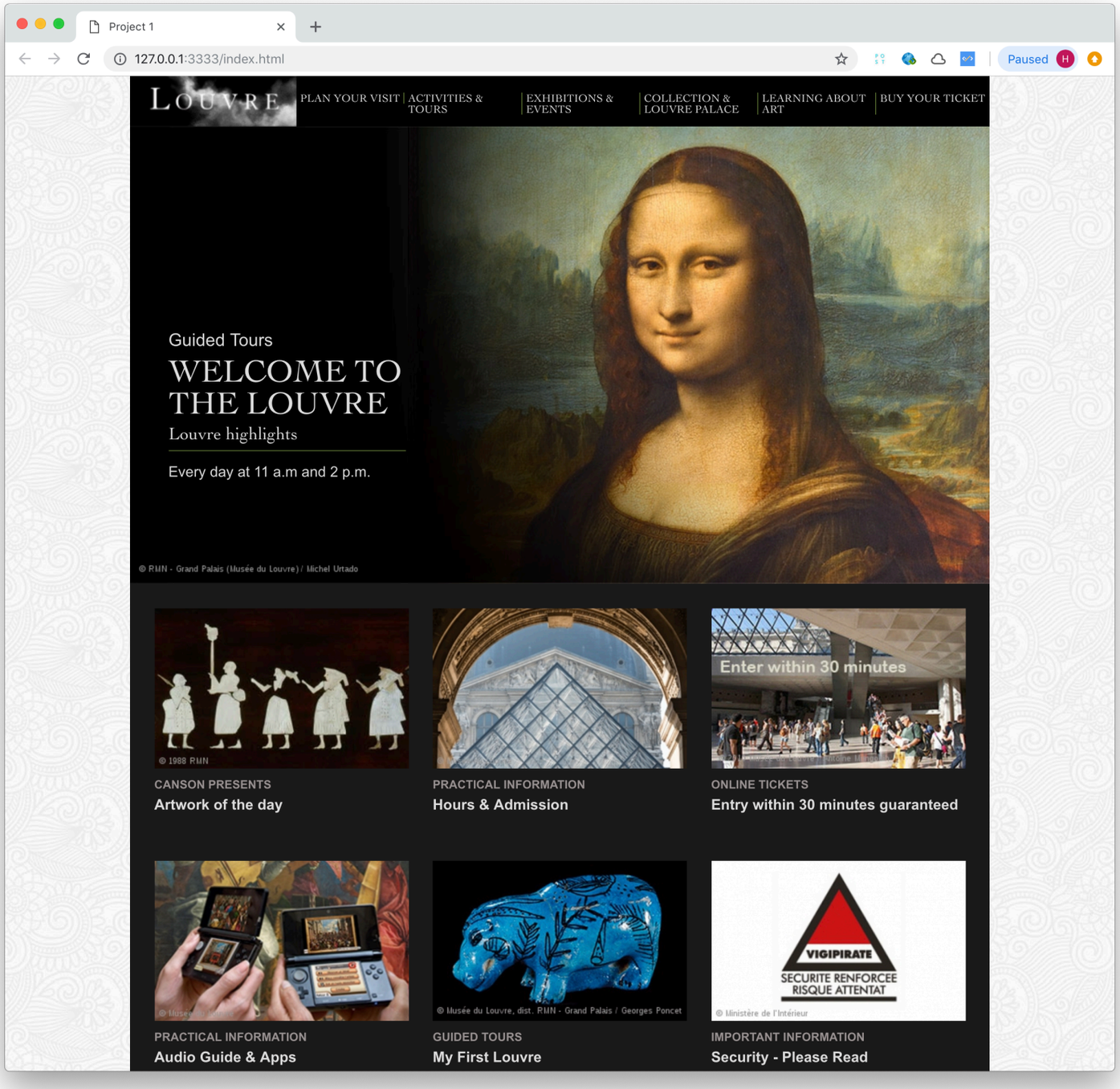
YOUR FIRST PROJECT

Professional front-end programming largely centers around manipulating web pages using JavaScript.

So, before we can get to JavaScript, you first need to get comfortable with the two technologies that work together to produce web pages: HTML and CSS.

You'll get the most out of this project if you read the Guides, *Intro to HTML*, and *Intro to CSS* first.

The first thing we'll do is identify the broad sections for our HTML, creating names for each of the sections.

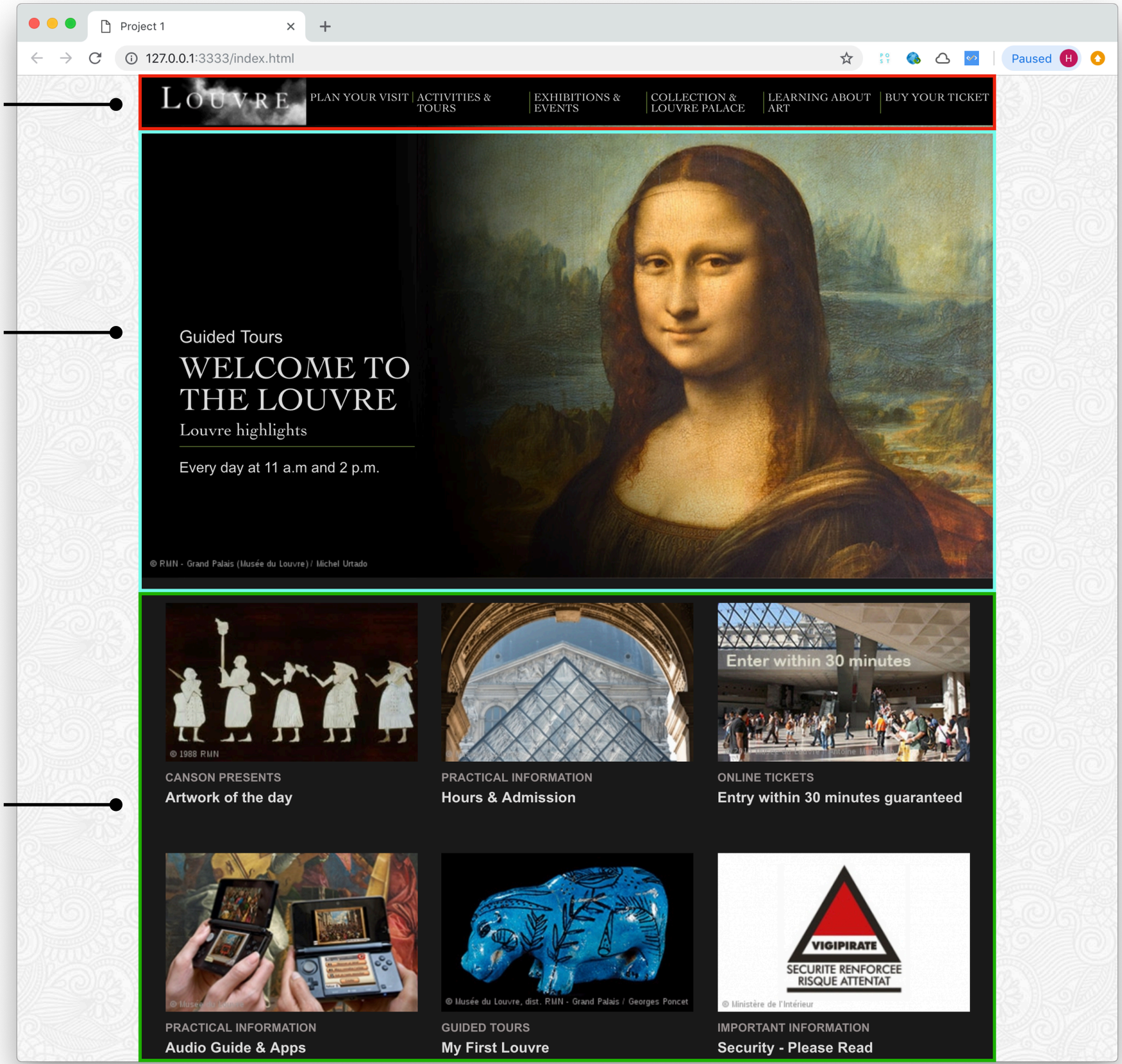


The finished project

We'll create a with an id of "header"

We'll create a with an id of "featured"

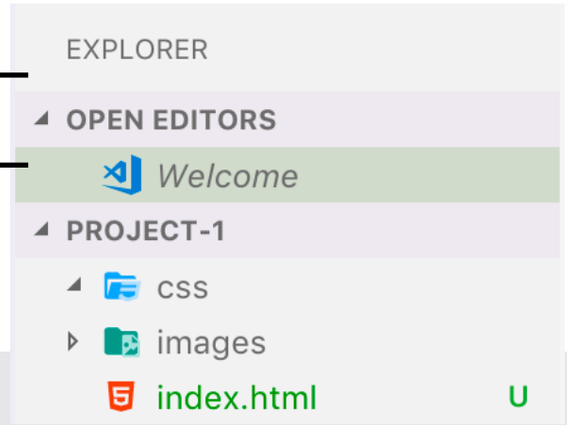
We'll create a with an id of "departments"



SETTING UP THE PROJECT IN VISUAL STUDIO CODE

Read the Guide, "Intro to git & GitHub", to learn how to clone a GitHub project

- From your `projects` directory, clone the `wise-guides/project-1` project from GitHub.
- Open VS Code and navigate to the newly-cloned project. From there create 2 folders: `css` and `images`.
- Create an `index.html` file within your project's root directory.
- Place the following code in `index.html`:



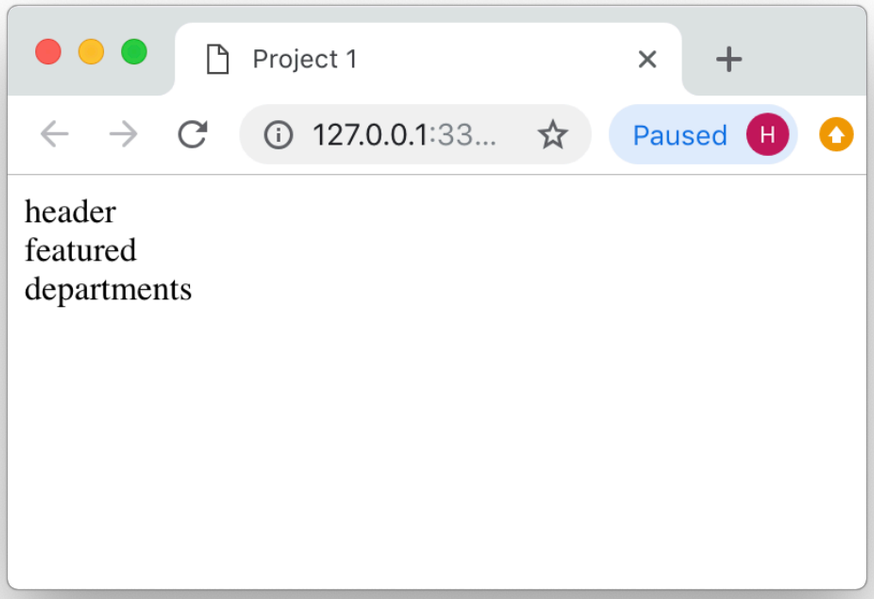
Read the Guide, "Intro to VS Code", to learn how to use shortcuts.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Project 1</title>
</head>
<body>
  <div id="header">header</div>
  <div id="featured">featured</div>
  <div id="departments">departments</div>
</body>
</html>
```

These are the divs we identified on the previous page.

Read the Guide, "Intro to VS Code", to learn how install the Live Server module.

- [ ] Open `index.html` in a browser using Live Server.



Once you have Live Server installed in VS Code, right-click within the file and select "Open with Live Server" to display the page in a browser window

You should definitely read the Guide, "Intro to CSS", before adding your CSS.

ADDING SOME CSS

If, someday, you're fortunate enough to have someone else writing your CSS, you are lucky indeed. In large development organizations, individual developers often end up specializing, but starting out, it's important that you get your hands dirty and write your own CSS. That said, it must be acknowledged that, for many front-end programmers, CSS is tough to really *master*. Our approach throughout the course will be to encourage you to learn as you go. And for this, Google is very much your friend.

We're going to create *two* CSS files for use in this project. The first one, `reset.css`, will remove much of the styling that browsers apply by default. Default styling affects things like *margin*, *padding*, *border*, *line-height*, and *list-styling*.

Why the need for a `reset.css` file? Because *we* want to be in control of styling. Without a good reset file, those browser defaults can introduce unwanted behavior that's hard to track down.

The code for `reset.css` comes from an expert CSS practitioner, Eric Meyer. You can download this file at `https://meyerweb.com/eric/tools/css/reset/`.

- [ ] In the css folder, create a `reset.css` file.

- [ ] Add the following CSS code:

Be sure to read the Guide, "Intro to CSS Selectors". It will help you make sense of this CSS.

These are CSS properties to be used for all the above HTML elements

```
/* http://meyerweb.com/eric/tools/css/reset/
   v2.0 | 20110126
   License: none (public domain) */
html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote,
pre, a, abbr, acronym, address, big, cite, code, del, dfn, em, img, ins, kbd, q, s,
samp, small, strike, strong, sub, sup, tt, var, b, u, i, center dl, dt, dd, ol, ul,
li, fieldset, form, label, legend, table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed, figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary, time, mark, audio, video {
 margin: 0;
 padding: 0;
 border: 0;
 font-size: 100%;
 font: inherit;
 vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav,
section { display: block; }
body { line-height: 1; }
ol, ul { list-style: none; }
blockquote, q { quotes: none;}
blockquote:before, blockquote:after,
q:before, q:after {
 content: '';
 content: none;
}
table {
 border-collapse: collapse;
 border-spacing: 0;
}
```

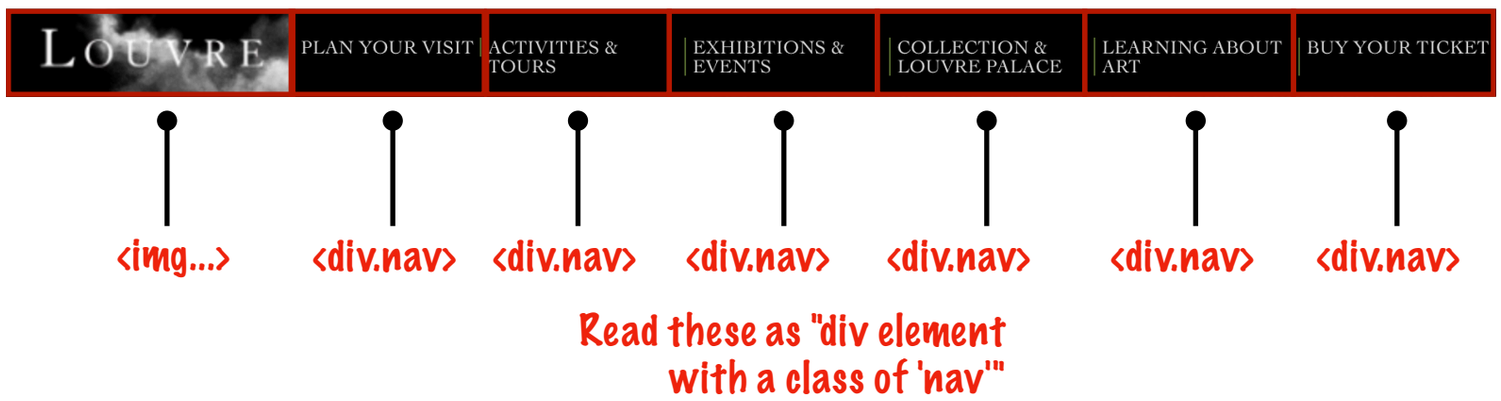
- Back in your `index.html` file, link `reset.css` into the HTML file:

Insert this line

```
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<link rel="stylesheet" href="./css/reset.css">
<title>Project 1</title>
```

Notice that the blank spaces to the top and left have been removed. That's the result of the application of `reset.css`.

We have our three main sections: `header`, `featured`, and `departments`. In the *Intro to CSS* Guide, we said that it's helpful to think of HTML as a group of boxes. Let's examine `header` more closely to see which inner boxes we can identify.



Why do we give the six `<div>` elements a `class` of `"nav"`? Classes are used when we want to *style* elements similarly. The *contents* of these elements may vary (as the text in ours do), but, as we'll see, the style for each one is the same.

- Back to `index.html`, let's add in the HTML code for these newly-identified elements:

Insert these lines

```
<div id="header">
  <img src="./images/logo-louvre.jpg" alt="" />
  <div class="nav">plan your visit</div>
  <div class="nav">activities & tours</div>
  <div class="nav">exhibitions & events</div>
  <div class="nav">collection & louvre palace</div>
  <div class="nav">learning about art</div>
  <div class="nav">buy your ticket</div>
</div>
```

BEGINNING STYLING

- It's time to create our second CSS file. It should also be created in the `css` directory. We'll call this one `index.css`. Add the following code snippets to the file.

Selecting the body element will cause all child elements to inherit the properties (unless children override the properties)

Provides the background image

Sizes the background image to fit window

Sets the font color to "gainsboro" — an off-white

```
body {
    background-image: url("../images/backgrounds/symphony.png");
    background-size: cover;
    font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
    color: gainsboro;
}
```

Sets the font to the first matching font found on the user's computer

Selects the three main elements

```
#header,
#featured,
#departments {
    width: 940px;
    margin: auto;
}
```

Creates a set width to these elements of 940 pixels

Centers the selected elements on the page

Sets the height of the header div

```
#header {
    height: 55px;
    background-color: black;
}
```

Sets the background color of the header div to black

Selects all <img> elements found inside the element with an id of "header"

```
#header img {
    float: left;
}
```

 elements, like all block-level elements, "stack up" atop each other by default. Floating them left causes them to push to the left where possible, ignoring the default stacking behavior.

Create the separation bars between nav elements

Turn into uppercase letters

Maximum width set to 120px. This will cause text to wrap.

```
.nav {
  border-right: 1px solid darkolivegreen;
  padding: 0 4px 4px 4px;
  margin-top: 18px;
  font-size: 12px;
  text-transform: uppercase;
  float: left;
  max-width: 120px;
}
```

Top: 0, Right: 4, Bottom: 4, Left: 4

Select the last .nav element on the page

```
.nav:last-child {
  border-right: 0;
}
```

Turn off the right border. For the last child, there's no other nav element to separate it from.

ON TO THE FEATURED SECTION

We've finished with the header and its nav elements. Now, let's break down the `featured` section.

Let's add in the newly-identified elements to our `index.html`:

Insert these lines of code

```
<div id="featured">
    <img src="./images/mona-lisa.jpg" alt="Mona Lisa by Da Vinci" />
    <div class="featured-group">
      <p class="intro">Guided Tours</p>
      <h1 class="main">welcome to the louvre</h1>
      <p class="summary">Louvre highlights</p>
      <hr />
      <p class="details">Every day at 11 a.m and 2 p.m.</p>
    </div>
 </div>
```

And add in the styling in `index.css`:

```
#featured {
  height: 500px;
}
```

Same height as the image of Mona Lisa

The purpose of <div.featured-group> is to wrap all of the text elements in a single element to make positioning easier

```
.featured-group {
  position: absolute;
  top: 25%;
  margin-left: 42px;
  max-width: 260px;
}
```

Check out the Guide, "Intro to CSS", for information about positioning

With positioning set to "absolute", we can "push" the element down from the top

Select everything inside featured-group

```
.featured-group * {
  padding-top: 7px;
}
```

Select elements with a class of "intro" that are inside element/s of class "featured-group"

```
.featured-group .intro {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 1.2em;
}
```

There is wide latitude in specifying font-size. Check out the Guide, "Working with Fonts"

```
.featured-group .main {
  font-size: 2.2em;
  text-transform: uppercase;
}
```

```
.featured-group .summary {
  font-size: 1.2em;
}
```

Styling the <hr> tag

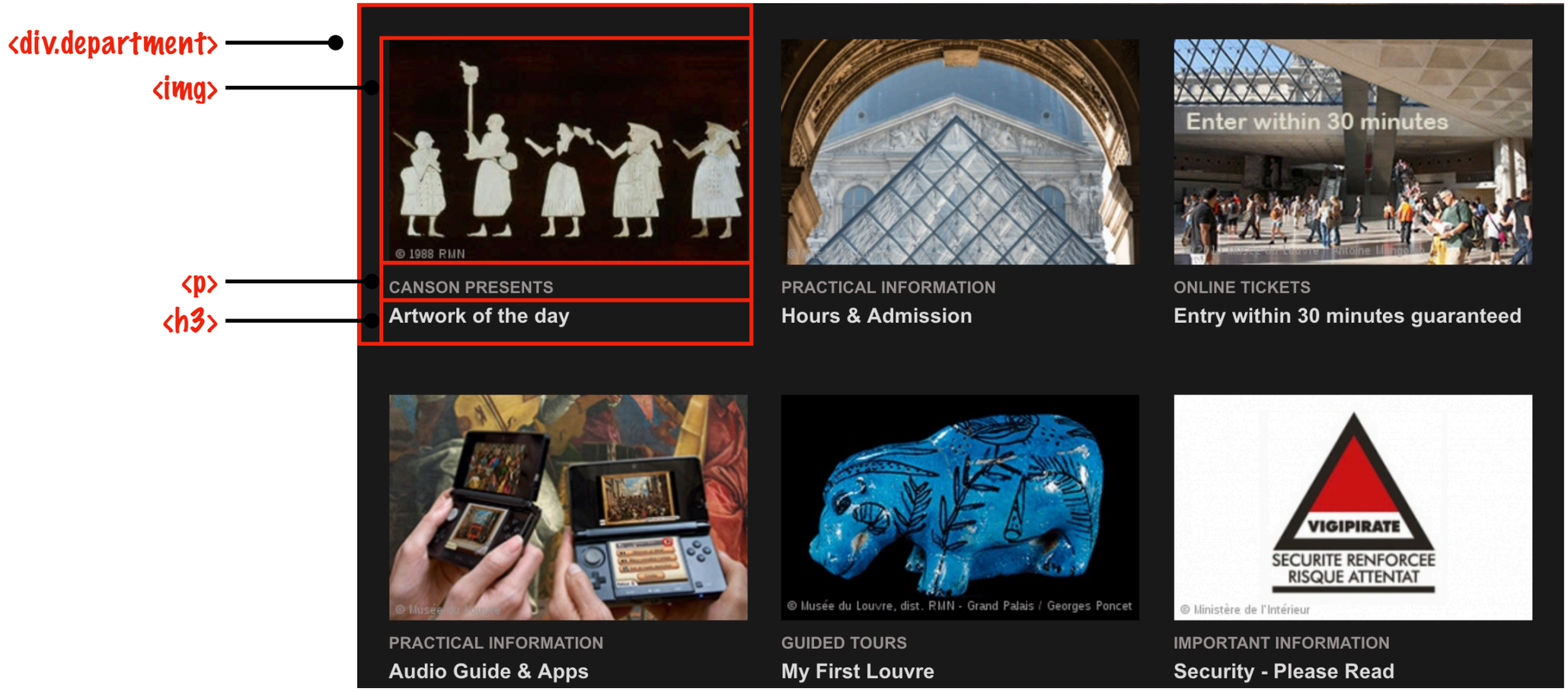
```
.featured-group hr {
  border: .5px solid darkolivegreen;
  padding: 0;
}
```

Earlier, we gave all children of .featured-group a padding. For the horizontal line, we want to remove that

```
.featured-group .details {
  font-family: Arial, Helvetica, sans-serif;
  padding-right: 26.5px;
}
```

FINISHING UP WITH DEPARTMENT

Let's look at the image to identify "boxes" within our final `<div>`, `departments`.



Here, we have 6 repeating `<div>` elements, all with a `class` of `"department"`. Here's the HTML for each of them:

```
<div class="department">
    <img src="./images/louvre-frise-d039un-panneau-mosaique_2.jpg" />
    <p>canson presents</p>
    <h3>Artwork of the day</h3>
  </div>
```

```
<div class="department">
    <img src="./images/louvre-hours-admission.jpg" />
    <p>practical information</p>
    <h3>Hours & Admission</h3>
  </div>
```

```
<div class="department">
    <img src="./images/louvre-entry-within-30-minutes-guaranteed-278-175-px.jpg" />
    <p>online tickets</p>
    <h3>Entry within 30 minutes guaranteed</h3>
  </div>
```

```
<div class="department">
    <img src="./images/louvre-img-mea-audioguides-du-musee.jpg" />
    <p>practical information</p>
    <h3>Audio Guide & Apps</h3>
  </div>
```

```
<div class="department">
    <img src="./images/louvre-statuette-hippopotame-278-175-px.jpg" />
    <p>guided tours</p>
    <h3>My First Louvre</h3>
  </div>
```

```
<div class="department">
    <img src="./images/louvre-vigipirate-278-175-px.jpg" />
    <p>important information</p>
    <h3>Security – Please Read</h3>
  </div>
```

FINISHING THE CSS

Now, we can finalize the CSS

```
#departments {
  background-color: #1d1919;
}
```

Dark grey color that will be the broad padding

We'll use this styling for all 6 of the department elements.

```
.department {
  float: left;
  padding-left: 26.5px;
  padding-top: 26.5px;
  padding-bottom: 26.5px;
  font-family: Arial, Helvetica, sans-serif;
}
```

```
.department img {
  margin-bottom: 12px;
}
```

```
.department p {
  text-transform: uppercase;
  color: #9c9393;
  font-size: .8em;
  font-weight: 600;
  padding-bottom: 8px;
}
```

```
.department h3 {
  font-weight: 900;
}
```

SUMMARY

We've covered a lot in this Guide. Because students often get confused (and discouraged) by CSS, let me reiterate: you're not expected to be a master of CSS. If you were able to follow along with this Guide, great! If not, try it again. You'll find things more comprehensible if you read the Guides referred to in the notes.

Above all, don't give up! You probably didn't do too well the first time you tried anything both new and complex. You'll make it. And it's worth it — you want to have a basic understanding of CSS before we get into the much-more-fun topics of JavaScript, React, and Redux.