FULL STACK, HTML

Jan 29, 2025

AGENDA

- Application Stack
- Representations of Front-End design
- Wireframes, Examples
- HTML
  - Basic Guidelines
  - Outline
  - Elements
  - Forms

WHAT WE'VE COVERED SO FAR

- ✓ Tracking code changes and maintaining a repository
- ✓ Interacting with the Operating system
- ✓ Introduction to building portable applications

Now we'll move on to learning the basics of some programming languages and concepts that will enable us to build a full stack application

APPLICATION STACK

Front-End

- The presentation layer
- What the user sees and interacts with
- What software tools do I use to build it and present it to the users

Back-End/Database

- The underlying database and code to interact with it
- Authentication, User Management

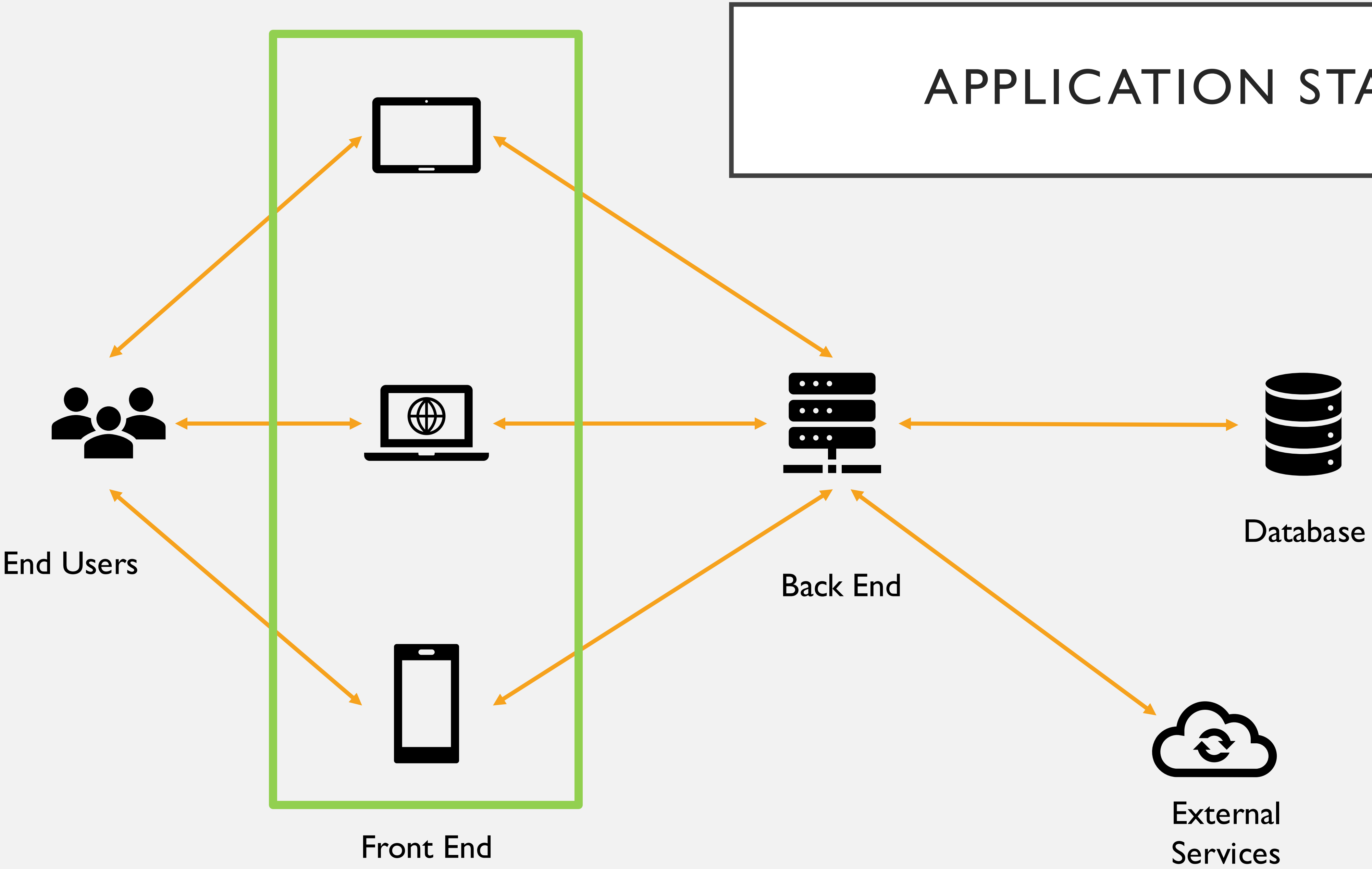
Back-End/Integration Layer

- Connects the Front-End to the Database

Full Stack Developer

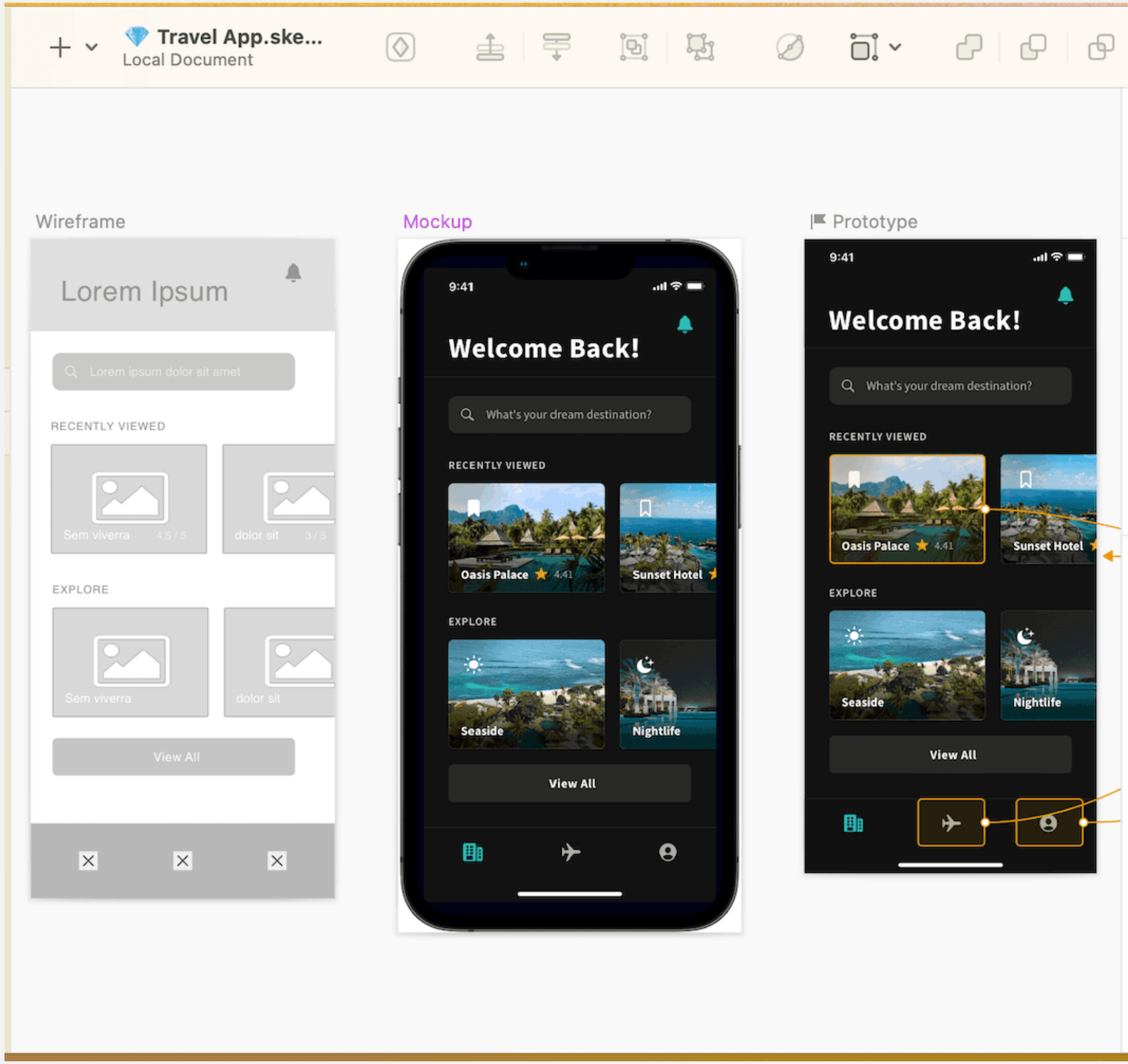
- "A Full-Stack Developer is someone who is able to work on all portions of an application – front-end, backend, integration."

APPLICATION STACK



REPRESENTATIONS OF FRONT-END DESIGN

- **Wireframe**
  - Blueprint that focuses on the structure and navigation of the UI
- **Mockup**
  - High fidelity representation of the design to depict the final appearance of the app
- **Prototype**
  - Early model of you application that will show the UI and the functionality of the app as it would be in the final version.
  - Can be high-fidelity or low-fidelity



https://www.sketch.com/blog/wireframe-vs-mockup-vs-prototype/

WIREFRAMES

How a team designs a web site

- The wireframes tell the story

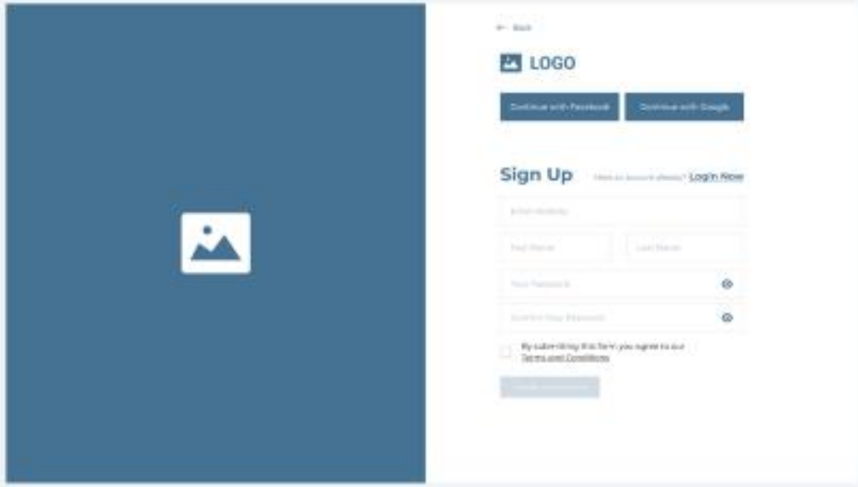
Contains important layout and content, not style

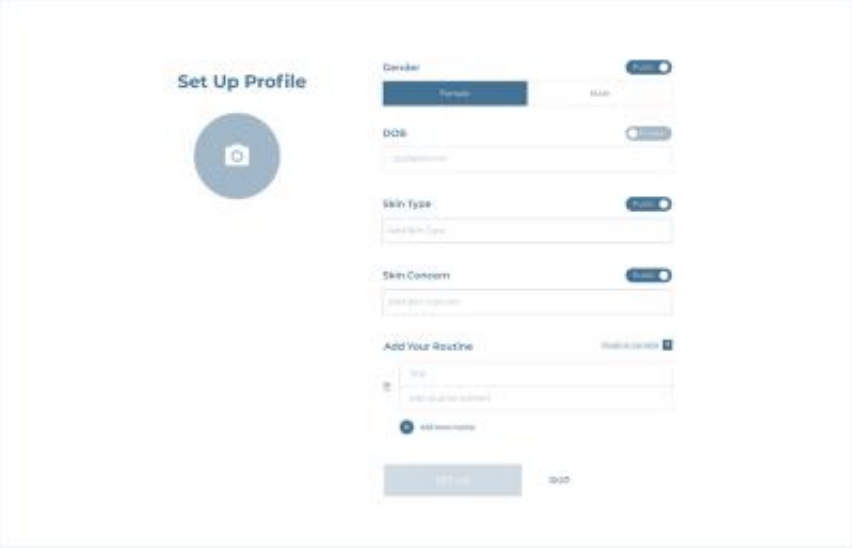
No color; no graphics, no images

Includes

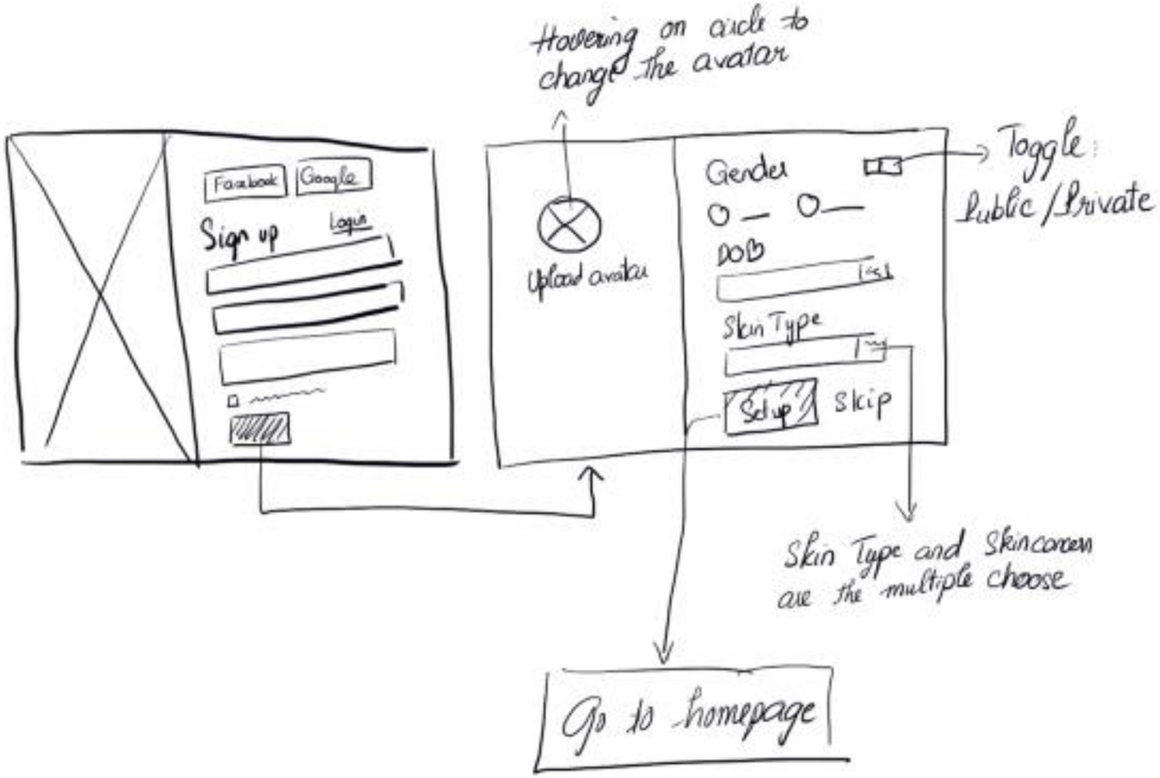
- Information design (describes what's being presented)
- Navigation design (describes links and nav menus)
- Interface design (drop down menus, buttons, form fields)

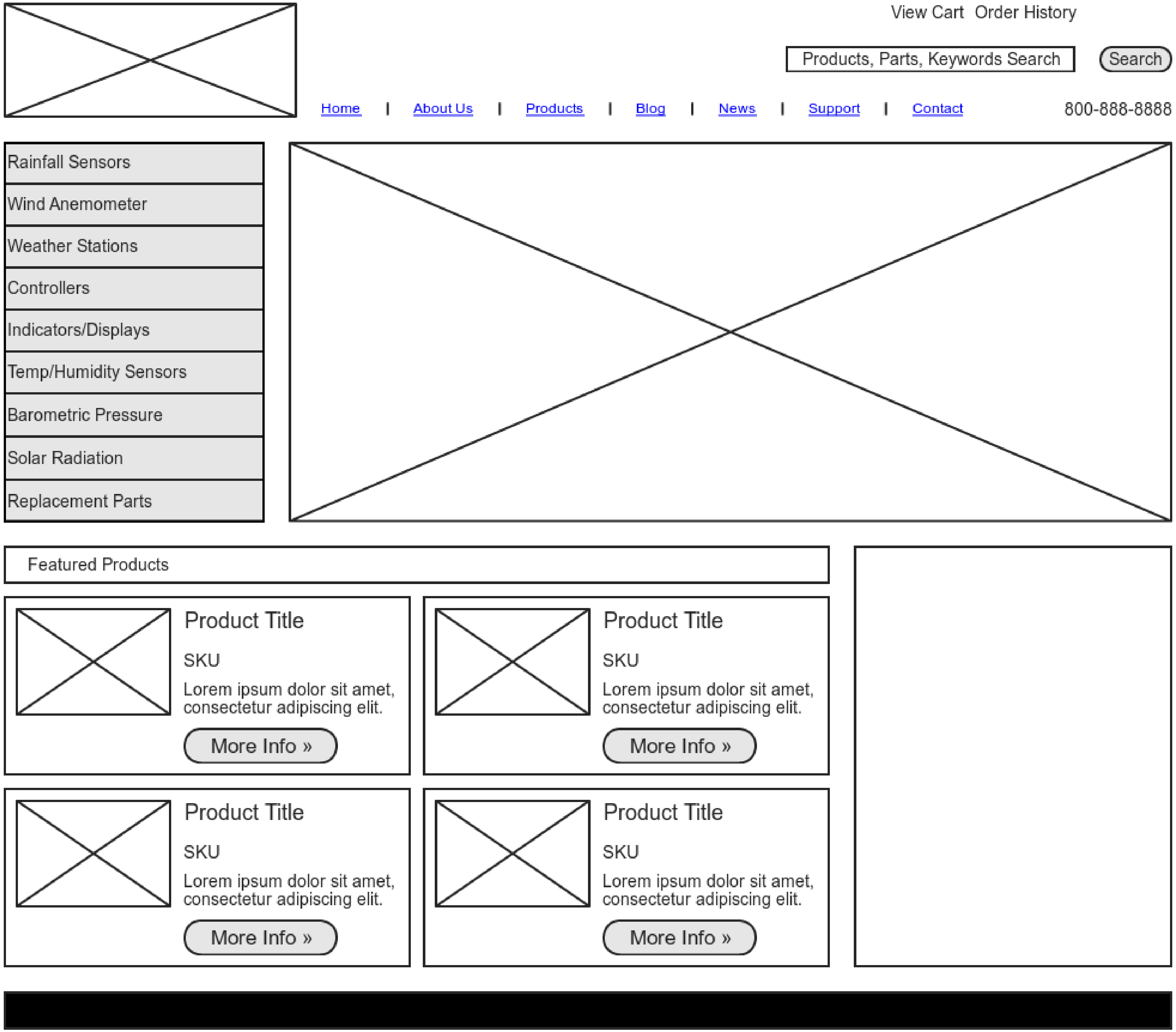
High Fidelity Wirefrmae





Low Fidelity Wireframe



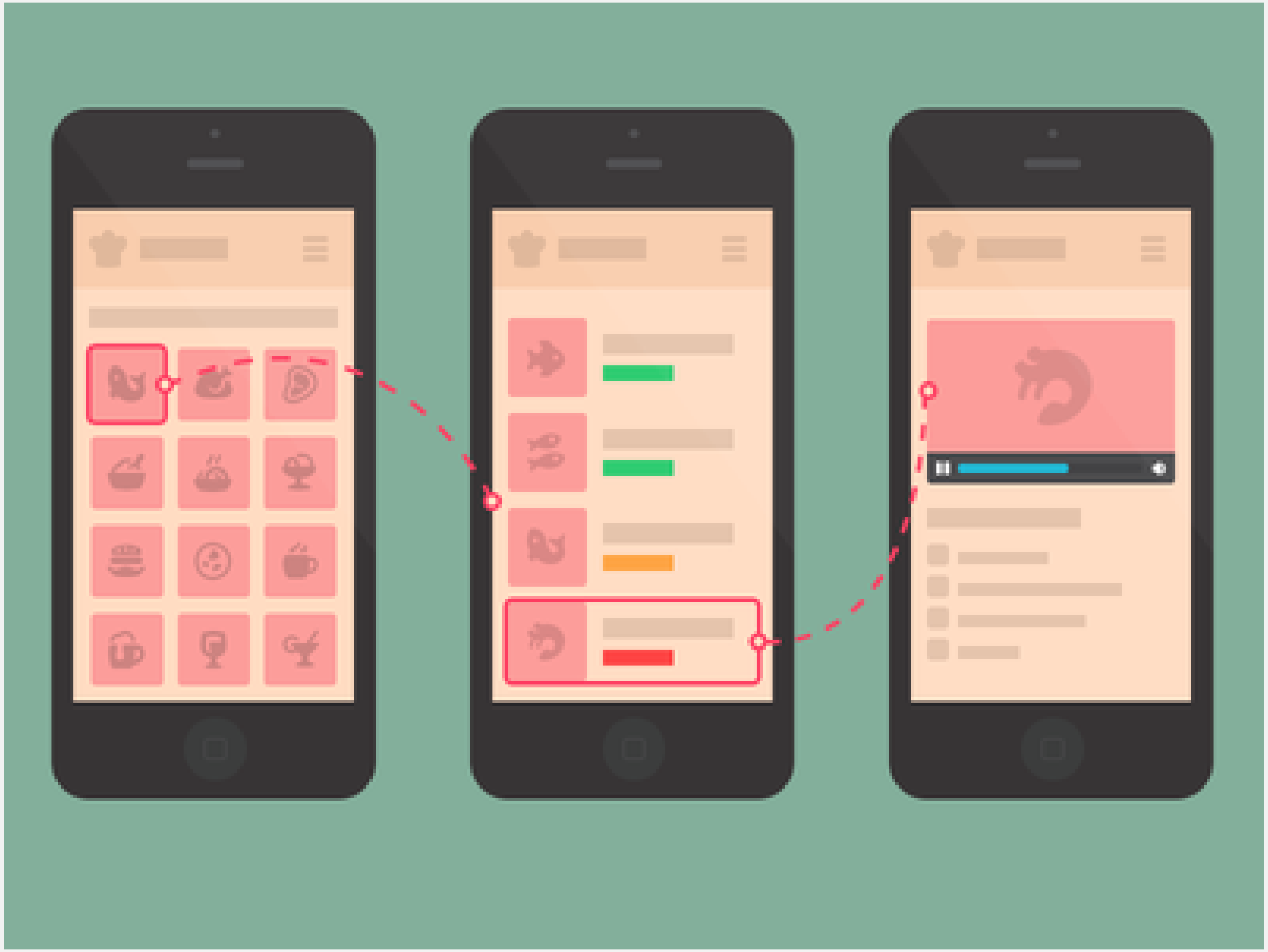


Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.



https://dribbble.com/shots/1058089-Mobile-Cooking-App-Wireframe-free#

WIREFRAMES

- Free Wireframe Tools
- https://www.designforfounders.com/free-wireframe-tools/

HTML

HyperText Markup Language

The foundation of all web pages

Marking up text with tags

Tags look like

<tag>content</tag>

Latest standard is HTML5 - focuses purely on the structure and layout of the page

Some markups might not be recognized by different browsers

SOME BASIC GUIDELINES

- HTML keywords are NOT case sensitive.
- Anything in quotes *might be* case sensitive
- Spaces: Many spaces = one space.
- Always use end tags.
- Nesting elements: From <tag> to </endtag>
- Good habit: Quote all attribute values.

Tip: always enter tags in lower case.

Such as attribute values:

```
<img src="picture.jpg">
```

BASIC OUTLINE OF AN HTML PAGE

```
<!DOCTYPE html>
<html>
   <head>
     <!-- head elements go here -->
   </head>
   <body>
     <!-- body elements go here -->
   </body>
</html>
```

ELEMENTS

- Paragraph
  - **`<p>`** text goes here **`</p>`**
- Headings
  - **`<h1>`** big heading **`</h1>`**
  - **`<h2>`** smaller heading **`</h2>`**
  - **`<h3>`** even smaller heading **`</h3>`**
- Divisions
  - **`<div>`** contains multiple paragraphs **`</div>`**
- Lists
  - **`<ol>`** an ordered list **`</ol>`** A,B,C or 1,2,3
  - **`<ul>`** an unordered list **`</ul>`** bullets

MORE ELEMENTS

Horizontal Lines

- <hr size="10" width=”10px” float=”[left,right,center]" >

Line Break

- <br> </br>

Blanks

- &nbsp

Comments

- <!-- xxxx -->

GO PLAY

- https://www.w3schools.com/html/default.asp
- https://www.tutorialspoint.com/online_html_editor.php
- https://html.com
- https://codepen.io

TERMINOLOGY

Cache: Browser caches pages. Beware when reloading. SHIFT + Reload or CTRL + Reload will force.

"**Deprecated**" = no longer supported by the standard.

It’s either a **TAG** or it’s **TEXT**

Basic **attributes** for every tag

- **id="xxxx” or name=“xxxx”** – identifies it
- **class="classname"** – ties it into a group for style
- **style="xxxxx"** – where xxx is a list of style elements

LINKING

- Hyperlink
  - Takes the web user to another document
  - Can be on the same site, or ANYWHERE
  - Implemented via `<a>` tag ("a" is for "anchor")
  - Uses the `href="xxxxx"` attribute
  - Browser identifies Link via underscore and color
    - Avoid `<u>` tag
  - Cursor changes shape on "mouseover"
  - Status Bar shows URL of the link on "mouseover"

LINKING

- **Absolute versus Relative URL**
  - `Protocol://host.domain.tld/fullpath/file.html` - absolute
  - `file.html` - relative
  - Relative location set by current page OR `<base>` tag in `<head>` section
  - `<base href=http://host.domain.tld/path>`
  - No file name - uses "`index.html`" or "`default.html`”
- **URL**
  - Directory path – filename plus extension
- **Linking to a "marker"**
  - Defined by **`id="xxxx"`** attribute

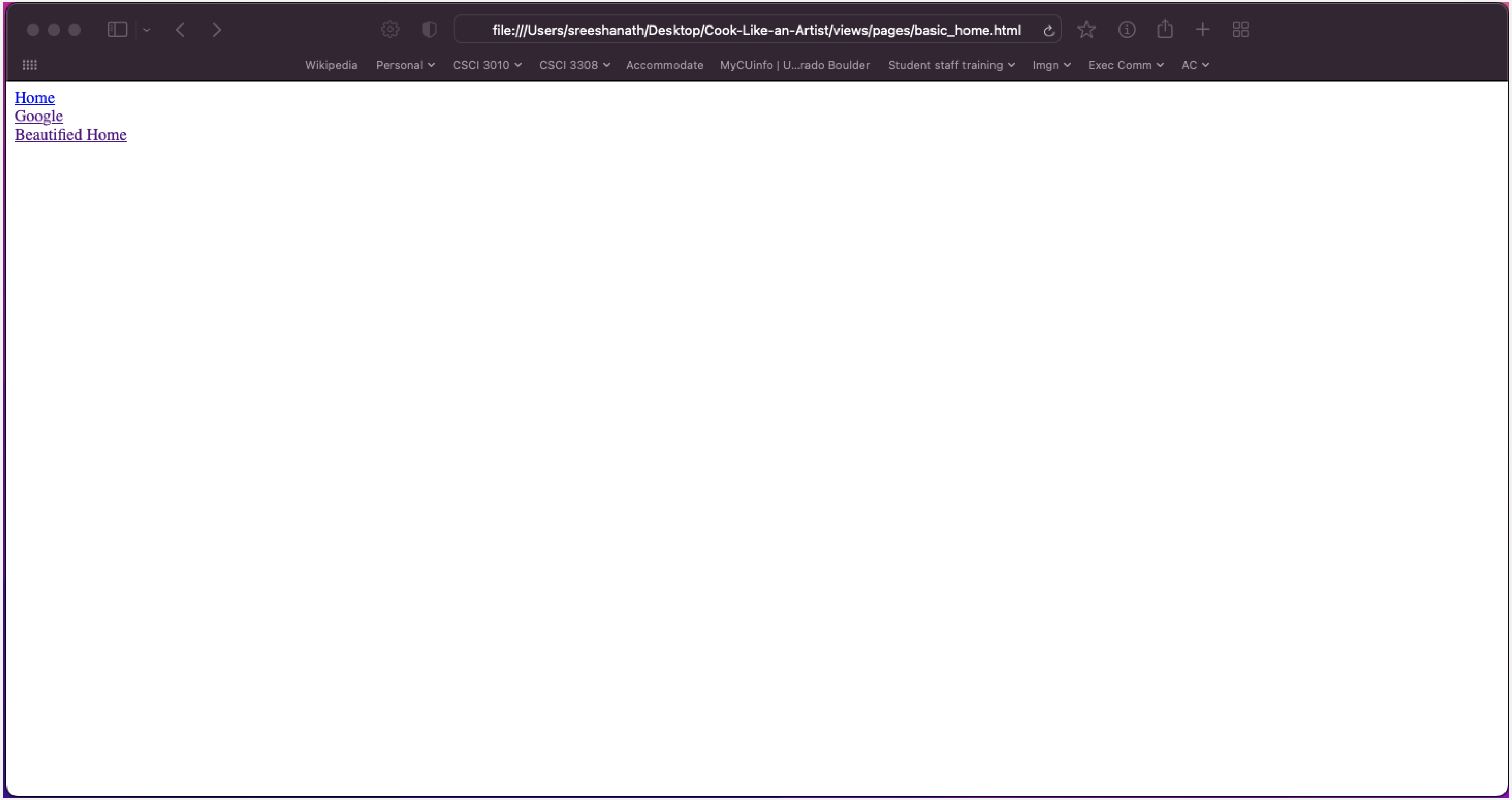
LINKING

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Cook Like an Artist</title>
  </head>
  <body>
    <a href="#">Home</a> <br>
    <a href="https://www.google.com">Google</a> <br>
    <a href="./home.html">Beautified Home</a> <br>
  </body>
</html>
```

Self

External Resource. Always use Absolute Path.

Page/Resource within the website. Always use Relative Path



file:///Users/sreeshanath/Desktop/Cook-Like-an-Artist/views/pages/basic_home.html
Wikipedia
Personal
CSCI 3010
CSCI 3308
Accommodate
MyCUinfo | U...rado Boulder
Student staff training
Imgn
Exec Comm
AC

Home
Google
Beautified Home

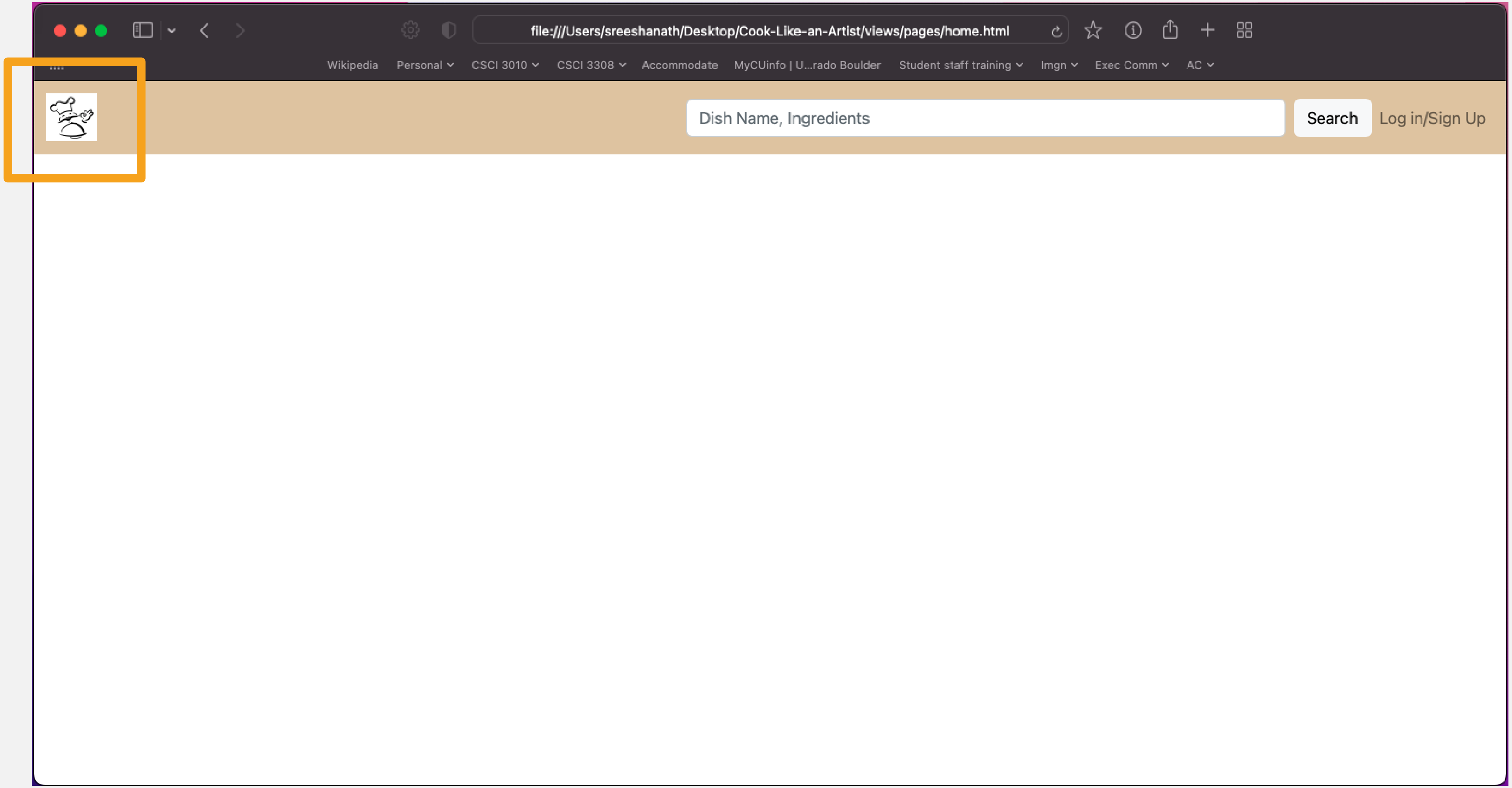
IMAGES

<img> tag attributes

- **alt=“xxxxx” -** text to display when image is not available
- **src=“path to file”** – location of image to be displayed

<img> as "link" (i.e., a "button")

```
<a href="./home.html”>
    <img src="../../resources/img/logo.jpeg” alt=“Logo” style="width:50px;">
</a>
```


file:///Users/sreeshanath/Desktop/Cook-Like-an-Artist/views/pages/home.html
Wikipedia
Personal
CSCI 3010
CSCI 3308
Accommodate
MyCUinfo | U...rado Boulder
Student staff training
Imgn
Exec Comm
AC
Dish Name, Ingredients
Search
Log in/Sign Up

TABLES

Used Primarily for LAYOUT options

Rows and Columns, "cells"

The browser will SIZE the table large enough to hold the cells’ contents

Every row gets the same number of "cell positions", whether you define or use them

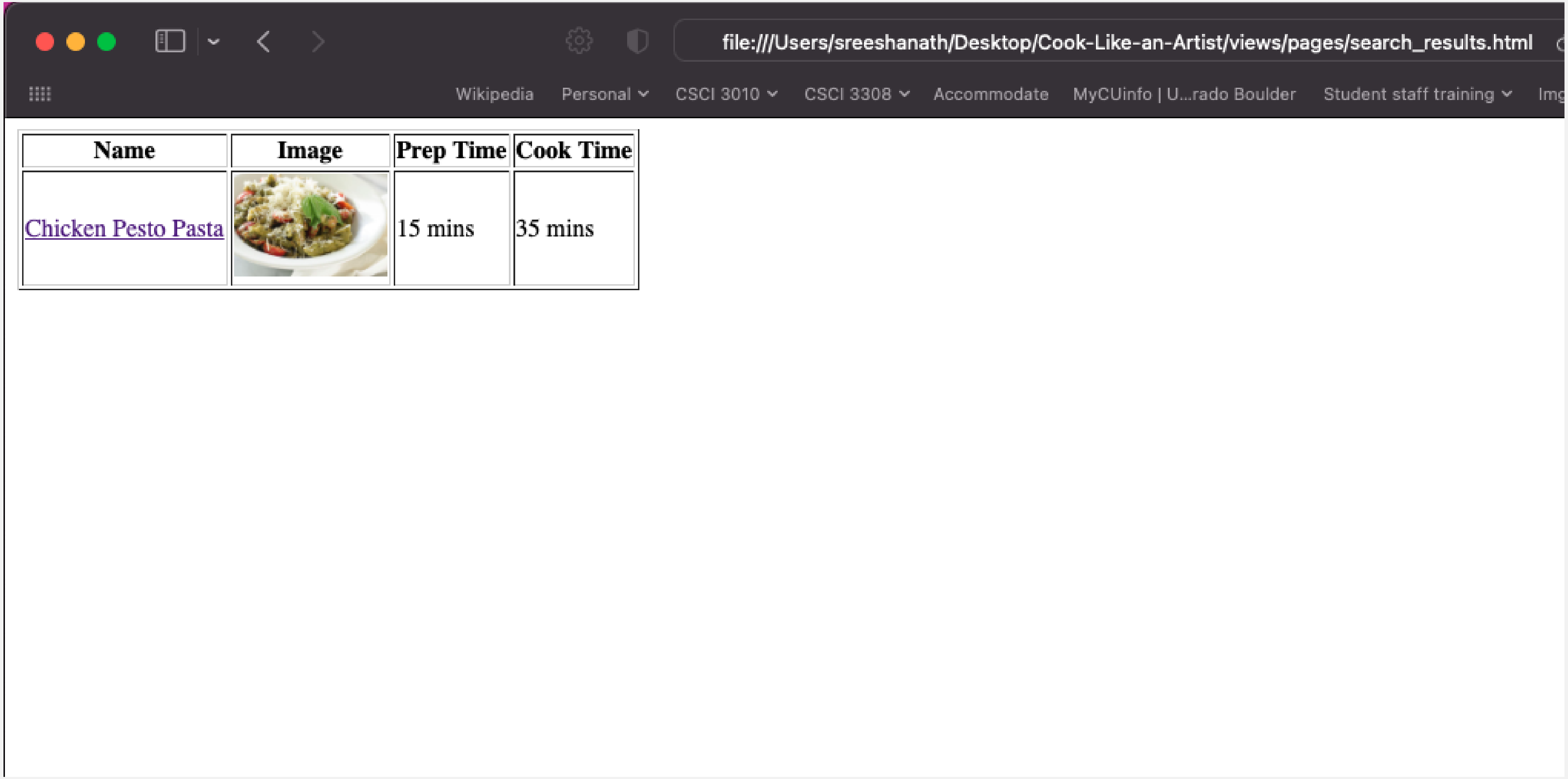
Table Tags:

- <table> defines the table
- <tr> defines a table row
- <th> defines the table’s header cell content
- <td> defines the table’s data cell content
- <caption> puts some text above, outside the table

TABLE

```
<table border="1">
<tr>
  <th>Name</th>
  <th>Image</th>
  <th>Prep Time</th>
  <th>Cook Time</th>
</tr>
<tr>
  <td>
    <a href="https://www.thepioneerwoman.com/food-
    cooking/recipes/a93847/one-pot-chicken-pesto-
    pasta/">Chicken Pesto Pasta</a>
  </td>
  <td>
    <img src="https://hips.hearstapps.com/amv-prod-
    tpw.s3.amazonaws.com/wp-content/uploads/2017/06/one-pot-
    chicken-pesto-pasta-01.jpg?resize=980:*" alt="Chicken
    Pasta" width="100">
  </td>
  <td>15 mins</td>
  <td>35 mins</td>
</tr>
</table>
```

Deprecated

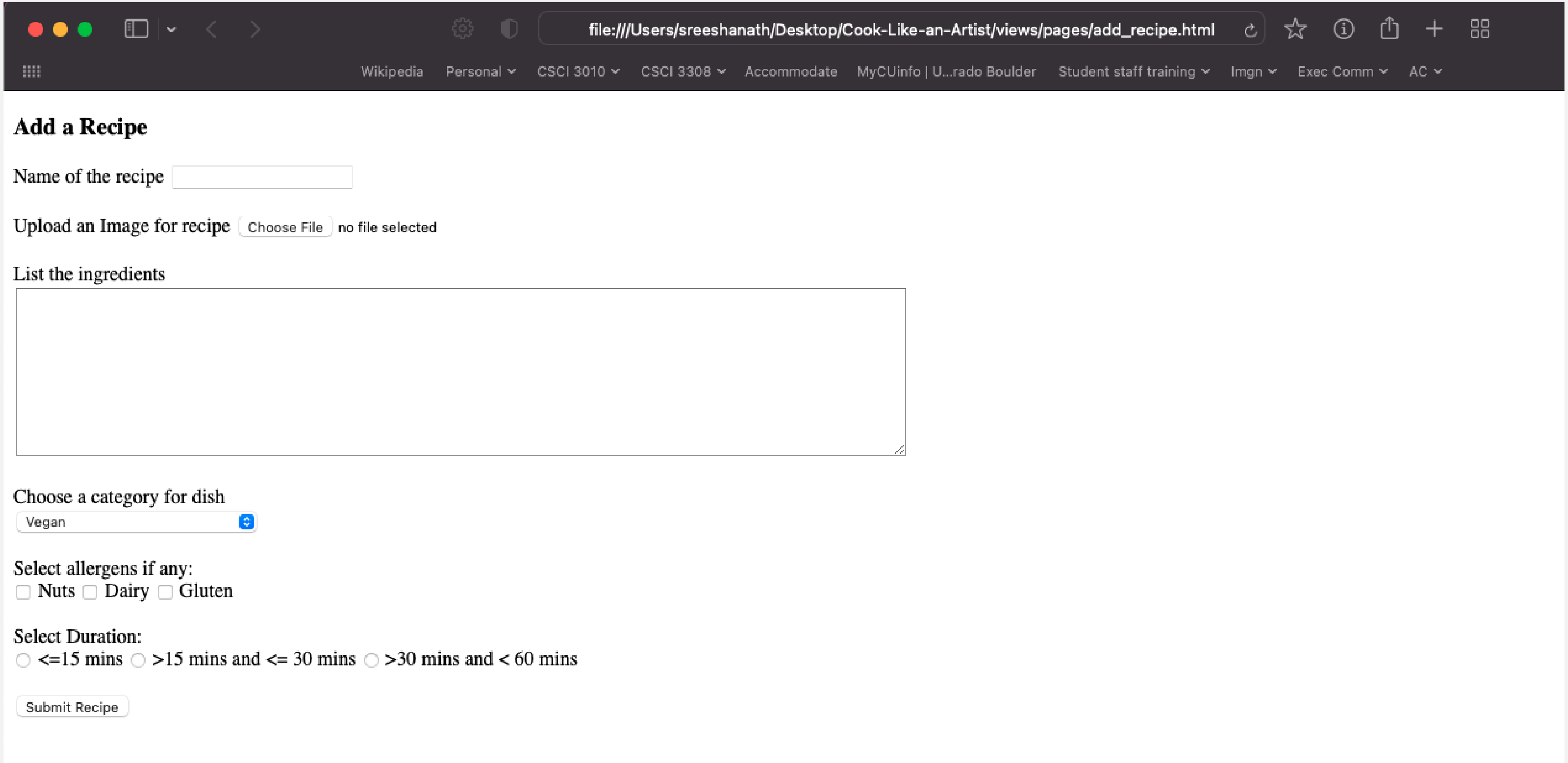


file:///Users/sreeshanath/Desktop/Cook-Like-an-Artist/views/pages/search_results.html

Name	Image	Prep Time	Cook Time
Chicken Pesto Pasta		15 mins	35 mins

HTML FORMS

- How are they used?
  - Use the browser’s window as a data entry screen
  - Collect information from the user
  - Pass it to the web server via http
  - Invoke a server-side script
  - Passes form data as input to the script



file:///Users/sreeshanath/Desktop/Cook-Like-an-Artist/views/pages/add_recipe.html
Wikipedia
Personal
CSCI 3010
CSCI 3308
Accommodate
MyCUinfo | U...rado Boulder
Student staff training
Imgn
Exec Comm
AC

Add a Recipe

Name of the recipe

Upload an Image for recipe Choose File no file selected

List the ingredients

Choose a category for dish
Vegan

Select allergens if any:
□ Nuts □ Dairy □ Gluten

Select Duration:
○ <=15 mins ○ >15 mins and <= 30 mins ○ >30 mins and < 60 mins

Submit Recipe

TWO IMPORTANT ATTRIBUTES OF FORMS

ACTION

- `<form action=“http://URL”>` name of a program on the web server
  - URL specifies the location of the executable file on the web server
- `<form action=“mailto:mailrecipient”>` prompts user to send an email

METHOD

- `<form method=“GET”>` or `<form method = “POST”>`
  - POST when you have large amount of data being sent, encryption available, a two-step process
- GET for small amounts, no security – all in one step

HTML FORM ELEMENTS

- The `<input>` tag –
  - Specifies an input field on a form
- `type` attribute – tells us what kind of control
  - text
  - radio
  - checkbox
  - submit
  - reset

HTML FORMS - EXAMPLES

<form> elements

- Text Box

```
<input type="text" name="recipe" id="recipe” size=”20”>
```

- Radio Button(s)

```
<label for="duration">Select Duration: </label><br>
<input type="radio" name="duration"> &lt;=15 mins
<input type="radio" name="duration"> &gt;15 mins and &lt;= 30 mins
<input type="radio" name="duration"> &gt;30 mins and &lt; 60 mins
```

HTML FORMS - EXAMPLES

- Checkbox

```
<label for="allergens">Select allergens if any:</label><br>
<input type="checkbox" name="allergens"> Nuts
<input type="checkbox" name="allergens"> Dairy
<input type="checkbox" name="allergens"> Gluten
```

HTML FORMS - EXAMPLES

- List boxes/Dropdown Menu

```
<label for="category">Choose a category for dish</label><br>
<select name="food_category" id="food_category">
  <option value="Veg">Vegan</option>
  <option value="Veggie">Vegetarian(contains egg, dairy)</option>
  <option value="NonVeg">Non-vegetarian</option>
  <option value="Keto">Keto-friendly</option>
</select>
```

HTML FORMS - EXAMPLES

- Submit Button

```
<input type="submit" value="Submit Recipe">
```

- Reset Button

```
<input type = “reset” />
```

- For a multi-line text box

```
<textarea rows="10" cols="100" name="ingredients" id="ingredients">
</textarea>
```

```
<form action="#" method="POST">
    <label for="recipe">Name of the recipe</label>
    <input type="text" name="recipe" id="recipe"><br><br>
    <label for="recipe_img">Upload an Image for recipe</label>
    <input type="file" name="recipe_img" id="recipe_img"><br><br>
    <label for="ingredients">List the ingredients</label><br>
    <textarea rows="10" cols="100" name="ingredients"
    id="ingredients"></textarea><br><br>
    <label for=”food_category">Choose a category for dish</label><br>
    <select name="food_category" id="food_category">
        <option value="Veg">Vegan</option>
        <option value="Veggie">Vegetarian(contains egg, dairy)</option>
        <option value="NonVeg">Non-vegetarian</option>
        <option value="Keto">Keto-friendly</option>
    </select><br><br>
    <label for="allergens_nuts">Select allergens if any:</label><br>
    <input type="checkbox" name="allergens” id="allergens_nuts”> Nuts
    <input type="checkbox" name="allergens” id=“allergens_dairy”> Dairy
    <input type="checkbox" name="allergens” id=“allergens_gluten”> Gluten<br><br>
    <label for="duration15">Select Duration: </label><br>
    <input type="radio" name="duration” > &lt;=15 mins
    <input type="radio" name="duration"> &gt;15 mins and &lt;= 30 mins
    <input type="radio" name="duration"> &gt;30 mins and &lt; 60 mins <br><br>
    <input type="submit" value="Submit Recipe">
</form>
```

CASCADING STYLE SHEETS

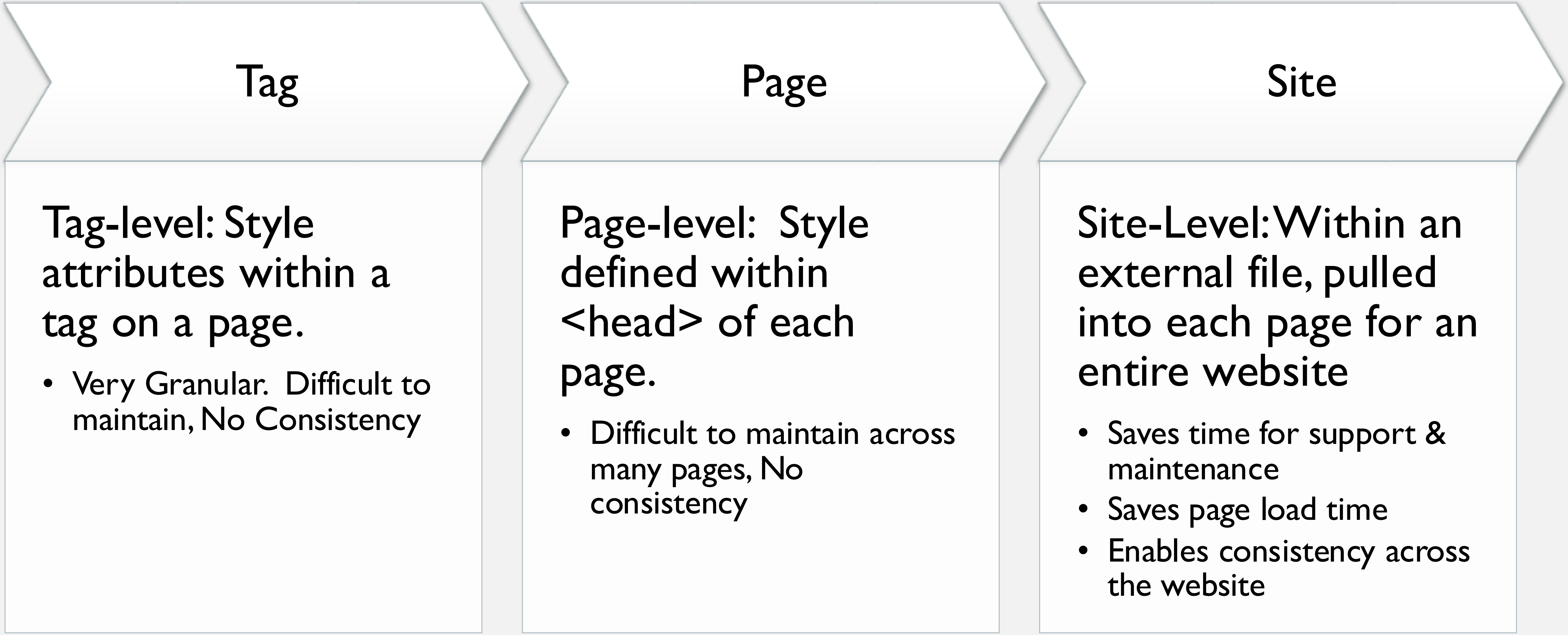
AGENDA

- What is style? How do we apply it to HTML?
- Cascading style sheet (CSS)
- Guidelines for style sheet
- Examples
- Types of identifiers
- Bootstrap

WHAT IS STYLE?

- Objects in an HTML document have “style” attributes
  - Font, font size, font color assigned to headings, paragraphs
  - Background color
  - Size and shape of images
  - Hyperlinks, colors, behaviors
  - Placement of objects on the page

APPLYING STYLE



Tag

Tag-level: Style attributes within a tag on a page.

- Very Granular. Difficult to maintain, No Consistency

Page

Page-level: Style defined within <head> of each page.

- Difficult to maintain across many pages, No consistency

Site

Site-Level: Within an external file, pulled into each page for an entire website

- Saves time for support & maintenance
- Saves page load time
- Enables consistency across the website

CASCADING STYLE SHEETS

- **What is a Style Sheet?**
  - The style sheet is an EXTERNAL document (.css) containing the rules of style to be applied to your document
  - You “link” to it to bring it into one or more pages

- **Why do we use them?**
  - Consistency from Page to Page
  - Faster, Easier Page Construction

GUIDELINES FOR A STYLE SHEET

- Should not contain any HTML tags
- Saved as file with a type/extension of ".css"
- Style Sheet –
  - Reference it and bring it into your document:

```
<head>
     <link rel="stylesheet" type="text/css”
     href="mystyle.css">
</head>
```

CASCADING STYLE SHEETS

- **Hierarchy of Applying Style**
  - Child tags Inherit Style from Parent tags
  - Detail-Level Overrides High-Level

- **Question**
  - If I have a style defined for a tag at the page level and at the tag level, which one do you think would take precedence?

CASCADING STYLE SHEETS

- **Using the <style> tag**
  - Entered in <head> section
  - Applies to entire document
- **Style Rules are Not HTML**
- **Rule Syntax**
  - selector – the tag that the rule applies to
  - { } Braces to contain the rule
  - Property:value pair
  - Ends with semicolon

EXAMPLES

```
<style type="text/css">
  body {
    font-size: 16pt;
    color: blue;
    background-color: pink;
  }
  h1{
    font-size: 24pt;
    color: black;
  }
  p {
    margin-left: 10%;
    margin-right: 10%;
  }
</style>
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<h1>Heading</h1>
```

```
/* CSS */
h1 {
   color: blue;
}
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<h1 id=“majorheading”>Heading</h1>

/* css */
#majorheading {
        color: blue;
}
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<h1 class=“demo_class”>Heading</h1>
<p class=“demo_class”>Paragraph</p>

/* css */
.demo_class {
        color: blue;
}
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<h1 class=“demo_class”>Heading</h1>

/* css */
h1.demo_class {
        color: blue;
}
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<h1>Heading 1</h1>
<h2>Heading 1</h2>

/* css */
h1, h2 {
      color: purple;
      text-align: center;
}
```

TYPES OF IDENTIFIERS FOR STYLE RULES

- element (tag)
- # id
- .class
- Element.class
- Group (multiple selectors)
- Nested (selector within a selector)

```
<!-- HTML -->
<div>
      <h1>Heading 1</h1>
</div>

/* css */
div > h1 {
      color: red;
}
```

OVERRIDING STYLES

- “!important” override indicator
  - For a rule that must NOT be overridden in cascade
  - “!important” must follow the rule

```
<p style=“color: green !important;”>
```

SOME STYLE PROPERTIES

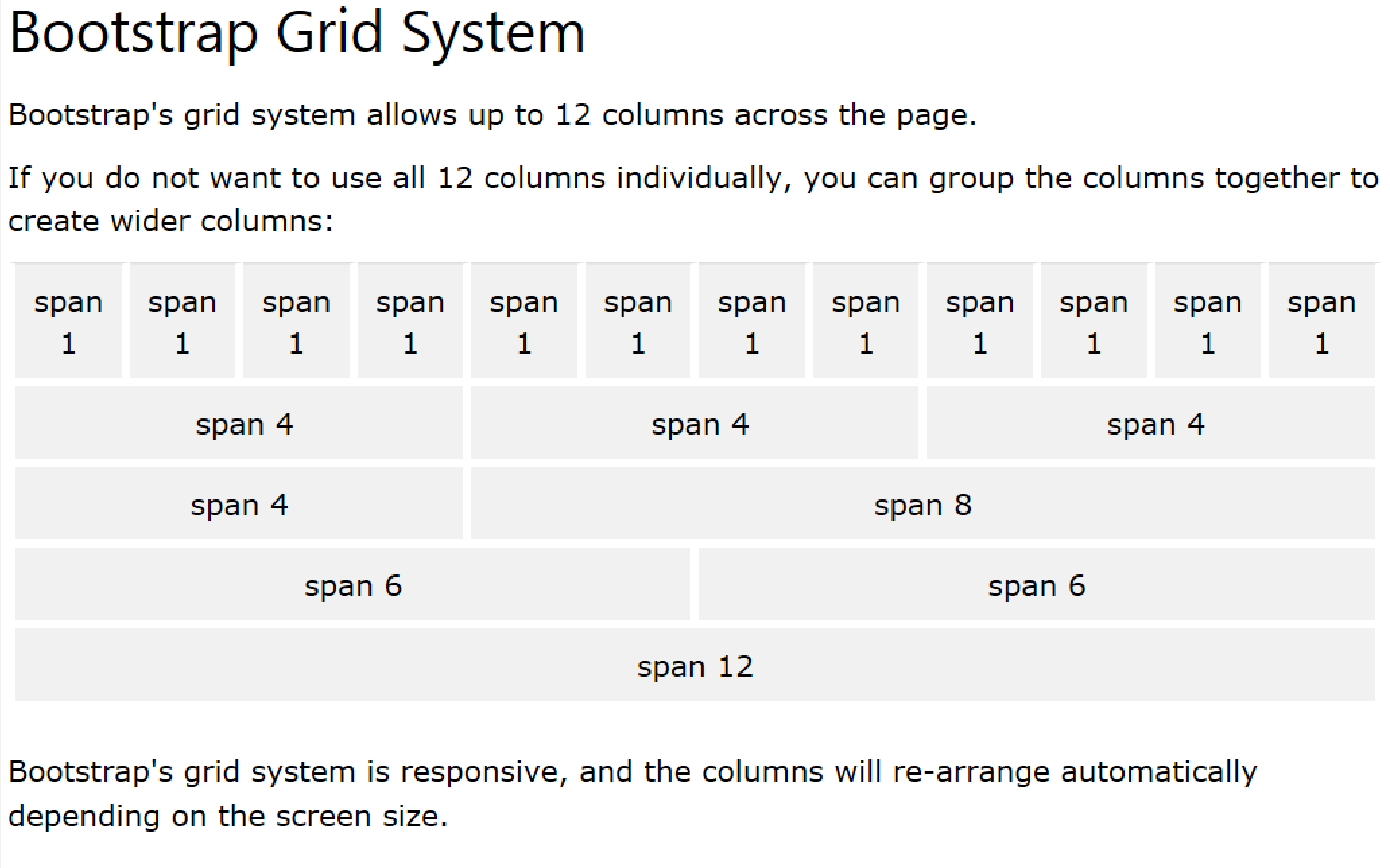
- Font Size
- Font Family
- Font-weight
- Text-transform
- Word-spacing
- Letter-spacing
- Text-align
- Text-indent
- Color
- Background-color
- Background-image
- Box properties

BOOTSTRAP

- A style library
- Easy formatting using templates
- Easy transition from computer-based viewing to phone-based viewing i.e., responsive, mobile-first websites
- https://getbootstrap.com/docs/5.3/getting-started/introduction/

BRINGING BOOTSTRAP INTO YOUR PAGES

Include	an external CSS sheet from a CDN (Content Delivery Network)
Include	a jQuery (Java Script) library (only needed for certain JS plug-ins)
Include	a client java script engine from a CDN



Bootstrap Grid System

Bootstrap's grid system allows up to 12 columns across the page.

If you do not want to use all 12 columns individually, you can group the columns together to create wider columns:

span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1 | span 1

span 4 | span 4 | span 4

span 4 | span 8

span 6 | span 6

span 12

Bootstrap's grid system is responsive, and the columns will re-arrange automatically depending on the screen size.

QUESTIONS?