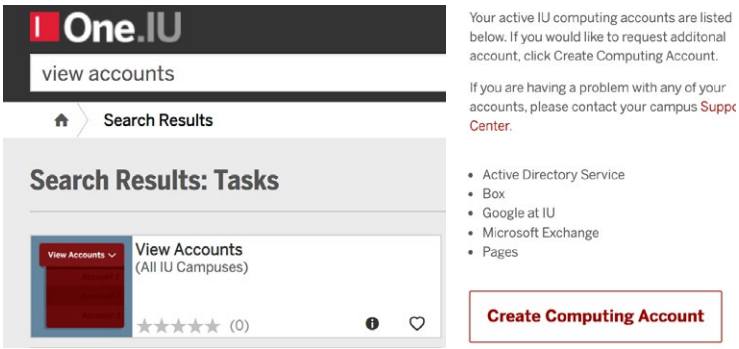
YOUR WEBSITE

With this handout, you will follow the steps to add your C226 content, from your three previous projects, to a basic HTML page, and use CSS styles to create colors, typography and structure. You will publish your site using IU's Pages service.

YOUR PAGES ACCOUNT

1. You will have to have an active Pages account to publish your work. If you think you might not have one, check by going to **one.iu** (**https://one.iu.edu/**) and searching for "View Accounts." You should see "Pagesy" listed on your active accounts.

2. If you don't yet have an account, click the "Create Computing Account" button, and on the next page, click the radio button next to "Pages." Then click "Create Account." Your account will be active within the next 24 hours.



PREP TIME

1. Download the **C226 Website project** compressed folder from Canvas > Files > Course Material or Canvas > Files > Assignments. Move the compressed folder to your Desktop (for easy access). Unzip the folder by double-clicking. Trash the .zip file.

2. Preview the contents of the folder: it includes an *index.html* document, a *css* folder that has in it a file called *style.css* that will specify the presentation aspects of your page. There are also two empty folders: *images* (you will put your Picture Story images in this folder) and *resources* (where you can put your text, among other things).

3. Copy the images you want to use from your Picture Story and Magazine Design (the JPEG spreads) to the folder's images folder. *Make sure* the file names include the .jpg extension, and that you do not have any capital letters or spaces in the name. Instead of spaces, you can use hyphens, like this: ***ernie-pyle.jpg***

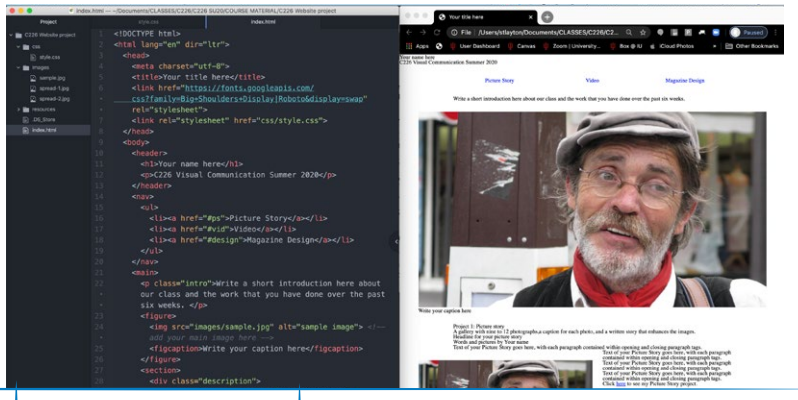
4. Open your story in Word (or whichever program you used to write it) and copy all the text.

5. Launch Atom (or your code editor). Go to File > Open, and navigate to the c226-website folder. You should see the files in the Project sidebar. Close any other window (hover over the right edge of the tabs and click the "x").

QUICK TIP! *Under the Atom menu, choose "Preferences," and click the "Editor" tab. Scroll down until you see "Soft Wrap," and check that box. This will keep all your text visible on the screen (i.e., no horizontal scrolling).*

6. Make Atom take up about half your screen, and launch a browser (like Chrome) on the other side. Click on the *index.html* file in Atom's project sidebar, and drag the tab from Atom to the tab on your browser. The *index.html* file will display there. The configuration should look like this:

Every time you make a change to your code, save the file in Atom (Command-S), and reload the page (Command-R) in the browser to see the effects of these changes.



ADDING YOUR CONTENT

1. Within the <title> tag, add your name.

```
<title>Ernie Pyle</title>
```

2. Inside the <h1> tag (on Line 10), write "C226 / " and your name.

```
<h1>C226 / Ernie Pyle</h1>
```

3. On Line 21, write a brief summary of our class. Remember — you will be publishing this site, so anyone on the Web will be able to see it. Make sure the entire page includes real text.

4. Inside the <h2> tag on Line 31, write *your* headline, replacing "Add your headline here." Use the same one you used for your Picture Story post, or come up with a new one.

```
<h2>Welcome to my life, tattoo</h2>
```

5. Inside the first <p> tag, add your name instead of "Yourname Here".

```
<p>Words and pictures by Ernie Pyle</p>
```

6. Swipe over the line of text in the second <p> tag, and paste your copied text (Command-V) from Step 4 on the previous page.

7. Each of the paragraphs in the story should be contained inside a <p> tag. Add opening and closing paragraph tags to each of your paragraphs until you reach the end of your story.

```
<p>Tattooing is an art that has been practiced for a very long time. While the meaning
and purpose differentiates across cultures, tattoos in the United States have recently
gravitated from a form of rebellion to a form of self-expression.</p>
```

8. We will now add an image. Images are added to HTML via the <img> tag. This is an empty tag, meaning it cannot contain content. It calls in the image to the page as an **attribute**, telling the browser specifically which image you want to display. Return to the first line after the <main> tag (line 20) and replace the img attribute you see there with your image — which should be in the images folder. Be sure to have the correct spelling of your image:

```
<img src="images/myimage.jpg">
```
(the file name will be the file name of your image, of course.)

This tells the browser to find the image inside the *images* folder and display it.

9. Next we'll add a caption to this image. For this, notice that our <img> tag is within a <figure> tag — the <figure> will open before the <img> tag and close after it. Before the <figure> closes, but after the <img> element, we will add a <figcaption> tag, which will contain the text of the caption. Make sure your caption is your real caption from your Picture Story! Here is how this chunk of code will look:

```
<figure>
   <img src="images/myimage.jpg">
   <figcaption>Write the text for your caption here.</figcaption>
</figure>
```

10. Use the model in the previous step to add one or two additional images (depending on the length of your story).

11. Save and reload your page on the browser side — you should now see your content.

NOTE: If you see a broken link instead of your image, double-check that you have spelled the filename correctly, and specified the path for the browser to locate the image.

12. Go to our class website — **https://sites.mediaschool.indiana.edu/mschc226-stlayton-fall21/** — and find your Picture Story.

13. Copy the link from the URL window.

14. Return to Atom, and paste the copied link into your href (replacing the hashtag symbol) for the hyperlink at the end of the first section element.

```
<p>Click <a href="http://sites.mediaschool.indiana.edu/mschc226-stlayton-
fall20/2020/10/01/explore-the-markets-of-paris/" target="_blank">here</a> to see my
Picture Story project.</p>
```

15. Move to the second section tag in your HTML code, the one that begins the Video description. Repeat the steps from the previous page where you added your own content, changing headlines and adding real text for the paragraph(s).

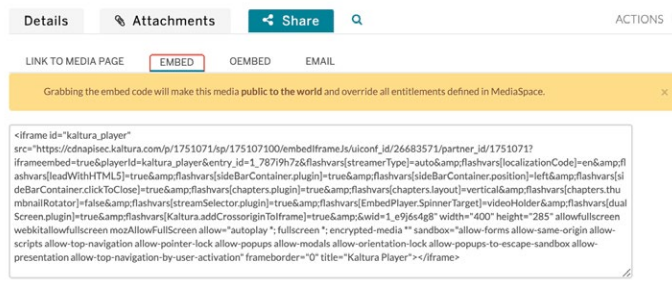
16. The template includes a sample video (actually one of our lectures), but of course you will want your own Video project to appear here. For this, we will bring in your Video from Kaltura by using an <iframe> tag.

a. GO to **https://iu.mediaspace.kaltura.com/my-media** and click on your video.

b. CLICK the “Share” button below the video, and click the “Embed” button.

c. COPY the <iframe> tag that appears there.

d. PASTE your copied <iframe> tag over the <iframe> tag you see in the HTML code. (The <iframe> tag has lots of characters, so it will be quite lengthy.)



Details
Attachments
Share
ACTIONS
LINK TO MEDIA PAGE
EMBED
OEMBED
EMAIL
Grabbing the embed code will make this media public to the world and override all entitlements defined in MediaSpace.
<iframe id="kaltura_player" src="https://cdnapisec.kaltura.com/p/1751071/sp/175107100/embedIframeJs/uiconf_id/26683571/partner_id/1751071?iframeembed=true&playerId=kaltura_player&entry_id=1_787i9h7z&flashvars[streamerType]=auto&flashvars[localizationCode]=en&flashvars[leadWithHTML5]=true&flashvars[sideBarContainer.plugin]=true&flashvars[sideBarContainer.position]=left&flashvars[sideBarContainer.clickToClose]=true&flashvars[chapters.plugin]=true&flashvars[chapters.layout]=vertical&flashvars[chapters.thumbnailRotator]=false&flashvars[streamSelector.plugin]=true&flashvars[EmbedPlayer.SpinnerTarget]=videoHolder&flashvars[dualScreen.plugin]=true&flashvars[Kaltura.addCrossoriginToIframe]=true&&wid=1_e9j6s4g8" width="400" height="285" allowfullscreen webkitallowfullscreen mozAllowFullScreen allow="autoplay *; fullscreen *; encrypted-media *" sandbox="allow-forms allow-same-origin allow-scripts allow-top-navigation allow-pointer-lock allow-popups allow-modals allow-orientation-lock allow-popups-to-escape-sandbox allow-presentation allow-top-navigation-by-user-activation" frameborder="0" title="Kaltura Player"></iframe>

17. Save and reload. You should now see your video in place of our placeholder video.

18. For the last of the three <section> elements, you will be adding your magazine spreads. The code is already prepared for this — and will even display your spreads side by side thanks to the CSS (once it is correctly linked ... more on that in the next section of this handout). You can even use the same file names as in my code if you like!

```
<div class="spreads">
    <img src="images/spread-1.jpg" alt="opening spread">
    <img src="images/spread-2.jpg" alt="second spread">
</div>
```

C226 / Your name here
Visual Communication Summer 2020
Picture Story
Video
Magazine Design
Write a short introduction here about our class and the work that you have done over the past six weeks.
Write your caption here
Project 1: Picture story
A gallery with nine to 12 photographs,a caption for each photo, and a written story that enhances the images.
Headline for your picture story
Words and pictures by Your name
Text of your Picture Story goes here, with each paragraph contained within opening and closing paragraph tags.
Write your caption here
Text of your Picture Story goes here, with each paragraph contained within opening and closing paragraph tags.
Text of your Picture Story goes here, with each paragraph contained within opening and closing paragraph tags.
Text of your Picture Story goes here, with each paragraph contained within opening and closing paragraph tags.
Click here to see my Picture Story project.
Project 2: Video
A two-to-three-minute video story with a soundtrack, including at least one interview, music and a voiceover.
0:00 / 50:19
Headline for your video
Text of the short blurb for your video goes here, within opening and closing paragraph tags.
Click here to see my Video project.
Project 3: Design
A re-purposing of the Picture Story as two magazine spreads (four pages total) using Adobe InDesign.

19. Don’t forget to add real text for this third section, too.

20. At the bottom of your HTML file, find the <footer> tag. It includes two <p> tags: one to allow a user to email you, the second to allow the user to see our class website. For these to actually do something when they are clicked on, we will add an <a> tag within the <p> tag. Of course, we will have to tell the browser where we want to go, too. We do this through an **href** attribute inside the <a> tag.

```
<p>Visit our <a href="https://sites.mediaschool.indiana.edu/mschc226-stlayton-fall2020/">class website</a>!</p>
```

21. You will build the hyperlink to your email in a similar way, but instead of telling the browser to go to a website, you will be telling it to access your mail program.

```
<p><a href="mailto:youremailhere@indiana.edu">Contact me!</a></p>
```

Our HTML document is mostly complete — you should now be able to save the file in Atom, and reload the page in your browser to see your content. Of course, this is part of our DESIGN unit, so we now must move on to changing the presentation of our page!

STYLING YOUR CONTENT

1. Go to the *style.css* file aside the css folder. It already includes a **reset**, which strips away most browser defaults from the presentation. After that chunk of code ends, on line 48, there are a number of other style rules written into the document — but none of these are currently affecting your page. Why not?

2. Return to the *index.html* document to check the link in the <head> ... for this to work, the code needs to exactly match the file name which it doesn't. Repair the code to match the name of the file, save and reload the page. You should now see a slight change to your layout. The way to tell it's working? The background will be bright yellow.

3. Start by changing the background color of the page. To do this, you will apply a background-color rule to the <html> element. You can use a saved color name (like beige), or use RGB or hexadecimal values for your color.

```
html {
    background-color: yellow;
}
```

For a complete list of saved color names, go to: **https://www.w3schools.com/colors/colors_names.asp**

RECOMMENDED: Open your main image in Photoshop and see if you can sample a color from that image to use as your background color. (It's also perfectly fine to use white if you want ...

Three ways to make "white":

```
background-color: white;
background-color: rgb(255,255,255);
background-color: #FFFFFF;)
```

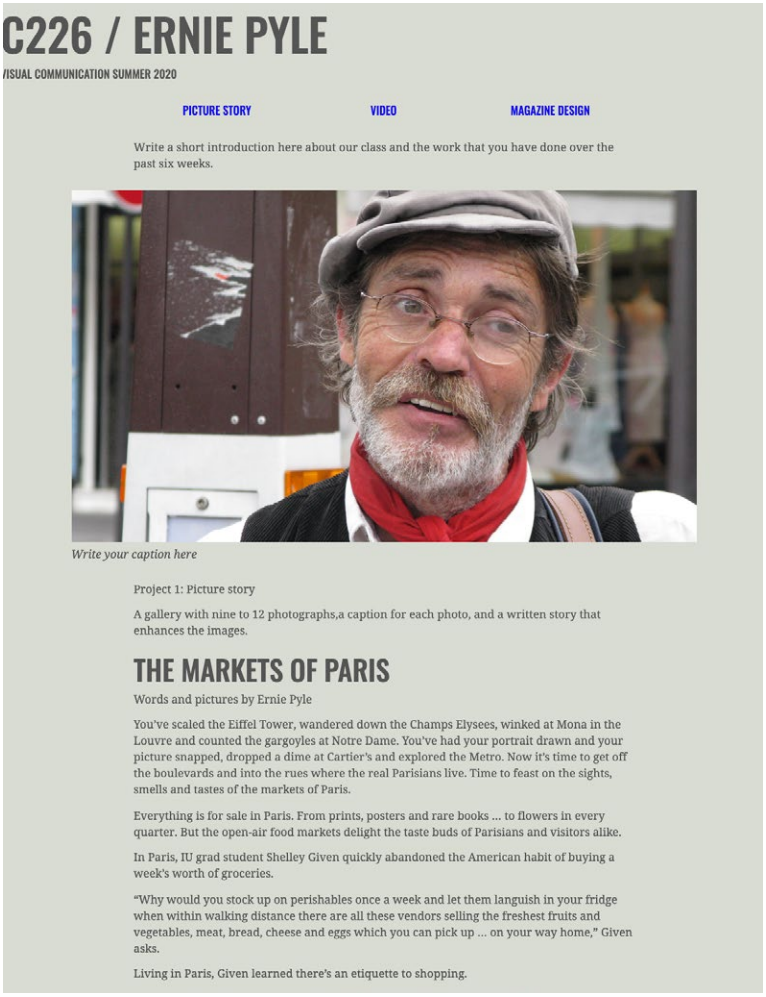
4. Reload your page. The color of the background will change from yellow to ... whatever you changed it to.

The CSS file already has some basic structure added in ... controlling the width and margin of the <main> element, some of the styling of the <nav> (the page menu), a few declarations for the visual content (images, figures and iframe) and a few margin/padding rules.

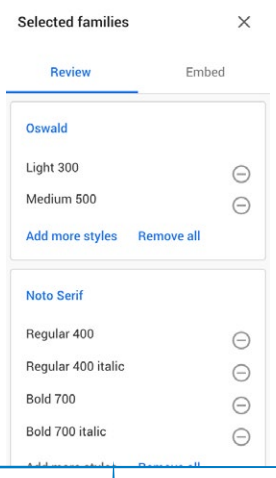
You may alter these if you like, but your focus for this project will be on creating customized styles for typography and display. Note that the CSS has three different "sections" set apart by commented headings: structure, typography and visuals.

5. So that we can read the text easier, we will add some space beneath each paragraph, using a variation of the margin rule. We will also add a rule so that there is more space between each line — the equivalent of leading.

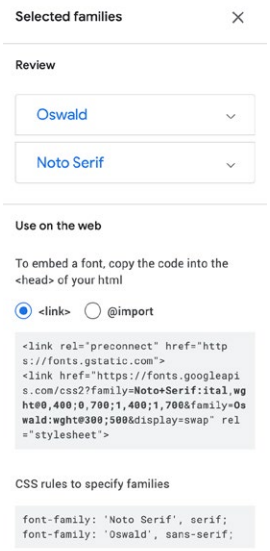
```
p {
    margin-bottom: 18px;
    line-height: 1.3;
}
```



6. Let's try customizing our typography to better reflect our content. We will be using Google fonts. Open a new tab on the browser side of your screen, and go to **https://fonts.google.com/**



7. Scroll through the available font families. Find one you like for the body copy, and another for your headings. Click a typeface family to review it, and if you would like to use one or more of its styles, click the blue "Select this style" button. It is now added to your collection in the sidebar at the right of the screen.



8. When you have chosen two fonts — one for body, one for display — make sure the "View Selected Families" button is blue at the top so you can see the sidebar (left). This panel includes two vital pieces of code.

9. Swipe over an copy the <link> tag and paste it into your index.html document, just after the <title> tag (line 6). It's recommended to call this in before your own CSS file.

12. Back in Google Fonts, copy the CSS font-family rule for the body copy font (i.e. your story text).

13. In your *style.css* file, add a rule for the <body> element, within the "structure" section. Paste the copied **font-family** rule inside the curly braces so all the text on your page appears in this font. (The color rule shown here will make the text easier on the eyes than black type, and we will make the text a little larger than the browser default of 16 pixels.)

```
body {
    font-family: 'Roboto Slab', serif;
    font-size: 18px;
    color: #555555;
}
```

14. Repeat this last step (copy from Google Fonts, paste into your CSS) for the typeface you want for your headings. This time, create a rule for the <h1> and <h2> elements, with selectors separated by commas. Change the color of the text by adding a color property. (You can also sample a color from Photoshop for this.)

```
h1,h2 {
    font-family: 'Oswald', sans-serif;
    color: darkolivegreen;
}
```

15. The previous rule was an example of a *multiple* selector, but we can also set individual styles for each of these elements — and we will do so for their size. Our reset turned all our text the same size, and now, because of the body rule (step 13), all our type is 18 pixels. But we want our headings to be clearly larger, so we will use CSS to adjust the size of the h1 and h2 elements, individually. We will use **ems** as our unit — essentially, this allows us to specify a value for the size in comparison to the size the elements would be without a specific rule:

```
h1 { font-size: 4em;}
h2 { font-size: 2.5em; }
```

(Use your own design sensibility to determine the sizes, of course.)

16. Next, we will center-align some of our content. We will write a rule for the header, footer and h2 elements to center them within the layout.

```
header, footer, h2 {
    text-align: center;
}
```

17. We'll also center the byline, which we can target by adding, in the *index.html* file, a class of "byline" to that <p> element.

```
<p class="byline">Words and pictures by Ernie Pyle</p>
```

We can now reference this particular paragraph in CSS. The dot between "p" and "byline" means that this selects *every paragraph with a class of "byline."* This rule centers the text, and also makes it appear in all caps.

```
p.byline {
    text-align: center;
    text-transform: uppercase;
    font-weight: bold;
}
```

18. Finally, we will change the color of our hyperlinks, both in normal state and when the user hovers over the link. You can use saved colors, or sample a hexadecimal color from Photoshop (see Page 6).

```
a { color: red; }
a:hover {color: darkred;}
```

On your own:

• Add some style touches to the project descriptions that begin each section, using this selector:

```
div.description
```

(This has already been set up in the HTML.)

• Try using a declaration for the **figcaptions**, so they will clearly differentiate from the rest of the text — try making the bold, or italic, or both, through the font-weight and font-style properties.

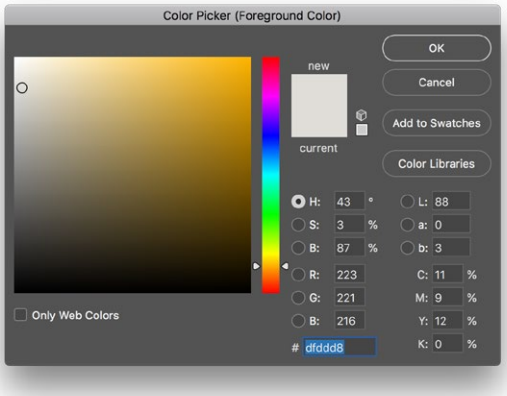
• Try adding a **box-shadow** beneath the magazine pages, using the following property:

```
box-shadow: 4px 4px 4px rgba(0,0,0,.5);
```

The four values are, in order: horizontal offset (from left), vertical offset (from top), amount of blur and color. RGBA color includes an opacity value (between 0 and 1, invisible to opaque) as the last value.

SOME NEW THINGS TO TRY

CUSTOM COLORS — You can use *any color you like* for web elements — not just those 140 saved color names. Try opening your image in Photoshop and sampling one of its colors with the Eyedrop tool. You can use the **rgb** values (red, green, blue) or its **hexadecimal** value for this. Photoshop's Color Picker — double-click the small color square at the bottom of the tool panel to call it up — will tell you these values. In the example shown here, the RGB value is **223,221,216** while the hex value is **dfddd8**. (The hex value requires a # sign in CSS.)



Here is how you would reference these colors in CSS:

RGB: `html {background-color: rgb(223,221,216);}`

Hex: `html {background-color: #dfddd8;}`

■ **TRY:** Write a CSS rule for the html element using a hex value for a color in your main image.

COLOR BOXES — You can use the **background-color** property to add a color for the box of an HTML element. (Remember the CSS Box model — every HTML element has a box!) We will try it with the <header> at the top, and the <footer> at the bottom.

```
header {
    background-color: rgb(223,221,216);
    padding: 10px;
    color: white;
}
```

This rule would make the background of the header element a color (our C226 shade of blue), and the padding is needed to add space between the elements (the h1 and p) and the edge of the box. Finally, the white color will affect all the text elements that are contained within the header — the h1 and p again.

```
footer {
    background-color: rgba(0,0,0,.2);
    padding: 30px;
}
```

For the footer, I've added a little more padding, and used an **rgba** color. The "a' in rgba stands for alpha, and it is how you can add a color with a reduced opacity — .2 means 20 percent. An rgb color of (0,0,0) is black, so this rule will slightly darken the box around the footer.

■ **TRY:** Write a CSS rule for the html element using a hex value for a color in your main image.

ADDING IMAGES — We have our main image at the top of the page, but let's add a second ... and write some code that would change the size and placement of that image.

Where you place the image within the HTML does matter. Go to your **index.html** document and add a second image about halfway the story (using your own image name and caption, of course).

```
<figure class="left">
    <img src="images/song-3.jpg">
    <figcaption>Copy and paste the
    appropriate caption.</figcaption>
</figure>
```

When I reload the page, the image will appear where I placed it, but will take up the full width of the <main> element, interrupting the text.

Try writing a rule for just this figure ... note that in my HTML code above, I have added a class to the figure tag. I can reference just this figure — and not the other one — by using, as my selector, figures that have a class of "left."

```
figure.left {
    width: 50%;
    float: left;
    margin-right: 30px;
    margin-bottom: 15px;
}
```

The width rule means that the figure will be just half the width of its container — the <main> element. The float rule will bring text from the story adjacent to the image, and I have used margins to add space between the figure and the text at its right and bottom.

■ **TRY:** Adding a third image, creating a new class, and floating it on the right rather than the left.

■ **TRY:** Adjusting the pixel width of the main element in CSS to see the effect on the layout.

PUBLISHING YOUR PAGE

Once you have created your Pages account and it is active, you can upload your files and publish your website. Your URL will be

http://username.pages.iu.edu

(where "username" is your username).

Once your account is set up and active, you are ready to upload files to the Pages server.

CONNECT TO IU'S VPN

To access the IU pages/Mercury server from off campus, you will need to download and install a VPN client. This is software that allows you to establish a connection between it and the VPN server, and offers access to VPN services.

1. Go **here** to download Pulse Secure.

2. Follow the instructions to **install and configure** Pulse Secure.

3. To connect, click the "connect" button.

4. When prompted, enter "push" for the secondary password. (This is Duo authentication.)

For more information on the IU VPN: **https://kb.iu.edu/d/ajrq**

For instructions on downloading, installing and using Pulse Secure: **https://kb.iu.edu/d/aygt**

CONNECT VIA CYBERDUCK

1. Cyberduck is an FTP program and is in the dock of the Macs in Franklin Hall. You can download Cyberduck yourself from **https://cyberduck.io/**

2. Hit Command-N to open a new connection. In the dialog box that drops down, choose "SFTP" in the top dropdown.

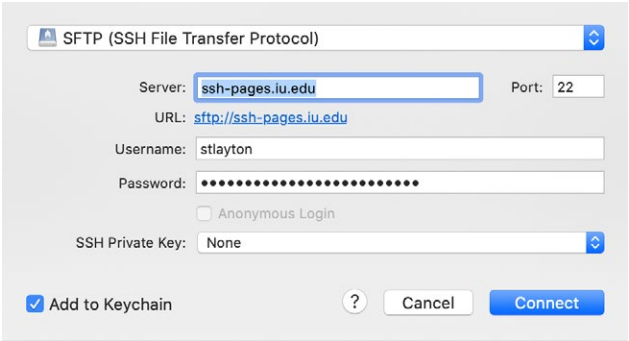
3. In the server name field, type
ssh-pages.iu.edu

4. Add your IU username and pass phrase, and click the "connect" button.

5. Authenticate with your Duo app.

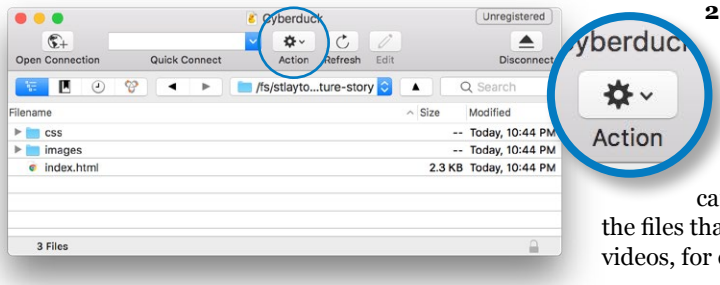
6. Double-click "web" to open that folder. You can move the relevant files into this directory, or create a new folder (c226) if there's something there already.

NOTE> Adding a "c226" folder will lengthen the URL to ***username.pages.iu.edu/c226***



UPLOADING FILES

1. Once connected, with a destination folder opened, click the Cog button, and pull down to "Upload."



2. Select the files and folders you want to upload from your project folder. Be sure to include your html document (*index.html*), your *images* folder and your *css* folder. DO NOT upload the folder itself, but DO upload its contents.

3. There is a 2 GB limit to the amount of space you can have on this server, so be sure to only include the files that are vital to your site, avoiding any large PSDs or videos, for example.

4. At any time, you can upload additional files, upload newer versions of files that are already there, or delete files — all from an FTP program like Cyberduck.

TROUBLESHOOTING

Check your URL to make sure that you have successfully uploaded your files and published your site.

• If your IU username is *erniepyle*, and your files reside at the first level of the "web" folder in your Pages folder, the url for your site will be:

erniepyle.pages.iu.edu

(note: your name is NOT Ernie Pyle and your username is NOT erniepyle).

• If you created a subdirectory (additional folder) within your "web" folder for your C226 project and named that folder "c226," the url for your site will be:

erniepyle.pages.iu.edu/c226

Is something not working correctly? Here are the most common issues:

• Forbidden — can't see anything but this short message

PROBLEM: Browser is looking for an index.html file at your URL and not finding one.

PROBABLE CAUSE: Did you upload your entire project folder rather than its content? If you were to upload the folder called "C226 Website project", that name would become part of your URL:

erniepyle.pages.iu.edu/C226%20Website%20project

SOLUTION: Re-upload only the files from that folder — *index.html*, the *css* folder and the *images* folder — rather than the folder itself. You can also simply move the files from inside that folder to the root level of the "web" folder by dragging them within the Cyberduck window.

• Not Found — can't see anything but this short message

PROBLEM: Browser is looking for a folder in your URL that does not exist.

PROBABLE CAUSE: A typo — check your URL to make sure it matches your IU username and any folder name.

SOLUTION: Type in the correct URL in the window at the top of the browser.

• Images not appearing

PROBLEM: Even though images worked fine while previewing before uploading, they do not appear on the published site; instead, there are only broken link icons and alt ext.

PROBABLE CAUSE: Three possibilities: file name of the image and the file name in the code do not EXACTLY match, capitalization in the file name, or changed structure in the relative path from the index.html file to the image.

SOLUTION 1: First, verify that the name of your file and the src attribute in your HTML code match EXACTLY. If the code calls for *example.jpg* and the file is named *example.JPG*, change the file name to all lowercase.

SOLUTION 2: If the names do match but you have used either spaces or CAPITAL LETTERS in the file names, change both the file name and the code to eliminate both. **Use only lowercase and no spaces in file names!**

SOLUTION 3: Make sure the structure of your project on Pages matches the code. If your code specifies "*images/example.jpg*", there MUST be an *images* folder that includes a file named *example.jpg* parallel to your *index.html* file! Re-upload your *images* folder to restore the relative path, and your images should display.

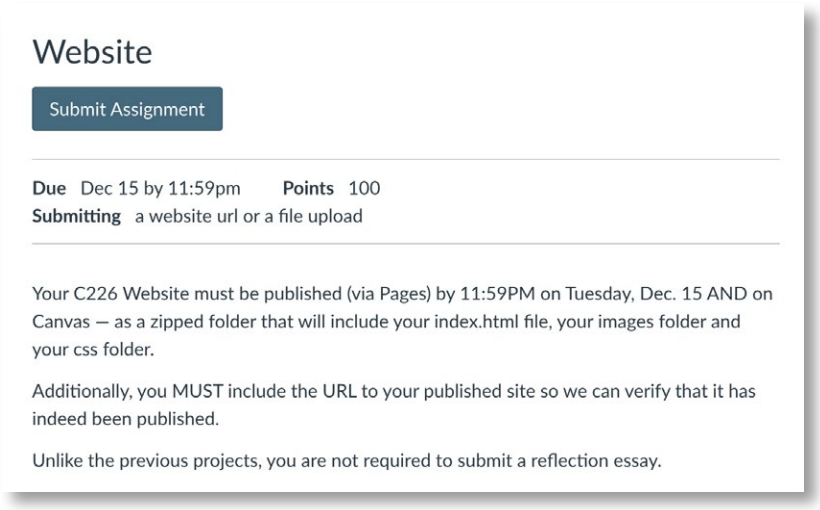
NOTE: This third solution would also be the way to correct the fact that your CSS does not show on your site.

SUBMITTING THE PROJECT

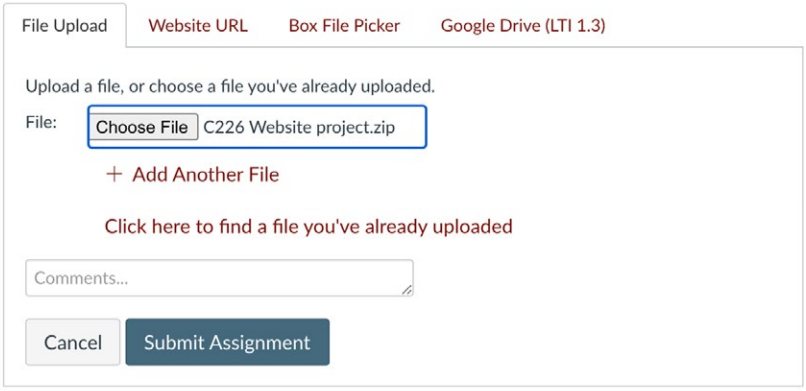
You will need to publish your website on Pages AND submit yourproject folder, as a ZIP file, to Canvas.

1. Once you have finished all your code and published your site, find your project folder on your computer and compress (ZIP) it so that it can be uploaded to Canvas.

2. Open the Canvas assigment (Website) and click the "Submit Assignment" button.

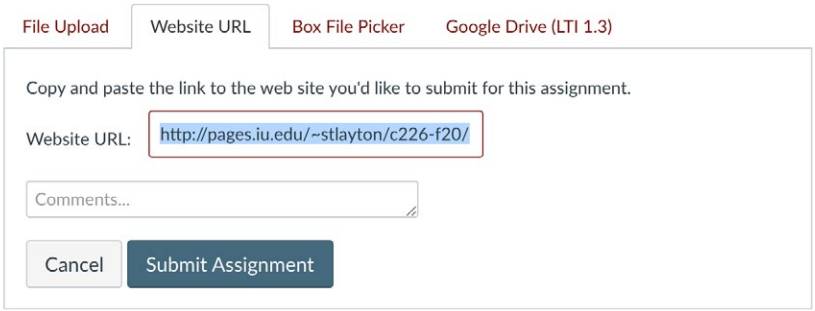


3. Upload your zip folder — make SURE you are not uploading the original zipped folder that you BEGAN working on this week!



4. Click the tab that says "Website URL" and add in your link here. You can type it yourself, or open another tab in your browser, go to your website, and copy the link from the URL window, then paste it into Canvas.

5. Click "Submit Assignment."



That's it! You are now officially finished with this project — AND with C226!

There is no requirement for a reflection essay with this assignment.