Website Creation Process

This outline will cover the website creation method for static site deployment using Github Pages. It will detail the initial setup process and proper steps needed to get started in deployment and website creation, but web development knowledge is required for the core content and will not be covered, however the full code for this project is provided at https://github.com/ktndwn/WGU. Reference documentation for web development can be found at either https://developer.mozilla.org/en-US/docs/Web/HTML or https://www.w3schools.com/. Creation will require a Windows 10 or MacOS computer with Firefox or Chrome, an internet connection and the ability to properly navigate the installation and creation of files.

Initial Setup

This section will cover where the website will live on the internet, installation of the tools required and in no part require any payment from the user. Let's begin with creating an account on Github, there are many other alternatives that offer more features or are easier to navigate, but they generally require payment or are beyond the technical ability of the average user. First, simply navigate to https://github.com and create a free account. Once an account is created and you are logged in to the Github homepage, create a new repository using the button at the top left and name the repository WGU (or anything else of your liking). Make sure the repository is set to public, click the checkbox that says initialize this repo with a readme and click Create Repository. Once in your repo page, navigate to the repo settings using the top tabbed menu and scroll down until you see the Github Pages section. Select master branch for source and the page should automatically refresh. If you scroll back down to the Github Pages section, it will display your new website url (https:// *your username* .github.io/WGU) for your repo. Currently this page

is linking straight to your readme markdown file that was created when the repo was made, but soon we will deploy an html file for the repo to use. Be mindful when entering this url as it is case sensitive and might not return the website if wgu is used in place of WGU. If you own a domain name that you would like to use in place of the provided url, there are numerous setup guides associated with individual domain name companies that you can research, but for now we'll leave it as the provided url.

HTML Creation

Now we will setup the actual html website files needed for this task. Create a project folder where the html files will reside on your computer. Name it TDT1 (or whatever you'd like). Navigate to https://code.visualstudio.com/ and download VSCode and install it. We will use this program to code the html files needed for our website. Once installed, click on File and then New Window, click the Explorer icon at the top left and click Open Folder, navigate to your created project folder and click Select Folder. Now right click anywhere in the TDT1 section of the Explorer pane and select New File. Name this file "index.html", this will be our homepage and is specifically named index.html to be recognized as the root webpage by the Github Pages url. We'll start coding and the first step is to fill in boilerplate code to begin with and luckily, VSCode's built-in auto-complete system is designed to help you throughout your coding process, simply type "html:5", select it from the drop down list and hit enter. The boilerplate code should fill in automatically. In the body section of your html file. Enter the word "Testing", save and test out any changes you make to this file by simply opening the index.html file with Firefox or Chrome and locally viewing your website. Using html, you can write content, display images, embed videos and link to other html pages using the documentation to MDN or w3schools listed

above. Since the focus of the instructional content is multimedia, a DSLR camera is choosen to be the banner image for the website. Create other html files to accommodate the required guidelines such as docs.html, graphics.html and multimedia.html and link as necessary.

Styling

While html is fairly useful on it's own, it's very dry and offers little for styling options built-in. For this, we'll need to link a filetype called css. So let's do just that. Create a new file called "style.css" and you should automatically see the new file open within VSCode. Navigate back to the index.html file by using the tab system at the top or the explorer panel on the left. We will now link the index.html with the newly created style.css file. In the head section of the index.html file, you have to paste these three pieces of code after the title:

```
<link href="https://fonts.googleapis.com/css?family=Montserrat:800&display=swap"
rel="stylesheet">

<link href="https://fonts.googleapis.com/css?family=Khula:300&display=swap" rel="stylesheet">

<link rel="stylesheet" href="style.css">
```

The first two pieces of code are font references and link to Google's font repository site, which is used by millions of websites around the world, if you have your own preferences, feel free to use those. The last piece of code tells the index.html file to use the style.css file you created. Html files load from top to bottom so we're telling it load the font files first and then use the css file. Using css we can assign fonts, change text sizes to fit what's most appropriate, resize images, change colors and add some menu animations for more visual flair. Make sure colors don't clash

with background images assigned. From this point on, the website creation process assumes knowledge of web development and you can use the MDN and w3schools reference documentation listed above and go back and forth between the html and css files to fill out the required content.

Deployment

Once you are happy with the content of your website and want to see it live on the web, open your Github repo page (https://github.com/ *your username* /WGU) and select "Upload files". At this point, navigate to your project folder on your computer, select and drag all the files in the TDT1 folder into the upload box on Github and select Commit changes. Double check to see if the index.html is in the root directory of the repo, otherwise it will not work. Deployment of your website is not instantaneous and can take anywhere between one minute to one hour (or sometimes longer). If you make any changes to your code, you will have to repeat the upload process again overwriting any previously saved files. If you ever have issues with updating and have waited the appropriate amount of time, try double checking if the index.html is in the root of the repository, opening a private tab or deleting browser cache.