This paper presents a detailed explanation of how I conducted this digital humanities project through what tools. In sum, I used Cascading Style Sheets (CSS) to make sidebar icons and image alignment to aesthetically advance my webpage. I also used Javascript functions such as intersection observer to display the change of sidebar icons as the viewer scrolls down different sections of the webpage; the "onclick" function also helped me present caption text with an embedded link to Wikidata when I click on one specific image of Chinese or Greek bronzes. I also used a historical geodata curator like Locolligo to transcribe my spreadsheet data to a visualized map with interconnected links to Artstor or Google Maps street view of museums.

For each of the specific processes from the sidebar icons, I used Font Awesome, a content distribution network (CDN) that provides access to icons for websites and other apps, as it helped visualize the toolbar so that it wouldn’t be just text by itself. This process is quite simple, you will need to first create a stylesheet with the link that can directly retrieve the icons you want from the CDN, and then insert your selected icon’s name to the class in your targeted text box. These icons should be prominent and simple enough for the viewers to understand the textual content you want to symbolize, especially for visual learners who catch images and symbols better than textual information, or for foreign language speakers who are not so familiar with the text but can understand the universal, generic symbol.

Moreover, in order to advance the effect, I also added a Javascript function to “highlight” the sectional icon when the viewers scroll down to each specific section in coordination. For example, when the viewer scrolls down to the section “Geographical Location,” the sidebar icon will be highlighted in black accordingly along with the viewer’s movement. This function helps the viewer locate their position on the website to get a sense of what they’ve read so far and how many sections are left. This is particularly helpful when the website page is too long to scroll down to the top or bottom to locate the web page position.

To accomplish this function, I created an “event listener” to observe the content status within the webpage window. To further detect the movement of the web page scrolling down or up to another section, I need to query select each section's name. In this case, all the sections are under scrutiny and will catch every time the viewer scrolls on my webpage. However, to highlight the section icons that were created earlier through CDN, I would need to create another function within the “event listener” function, using additional braces for the intersection observer function. Within this function, every time the entries intersect with the viewport, it will always iterate through the icons. However, we only want the viewer to see the highlighted icon when scrolling to the specific section. In order to do so, we need to first 'deselect' and cancel it, removing the constant highlighting. Then we can trigger the section when the function observes the viewer scrolling at the specific section by targeting it in the side icon list created in the earlier observing function. In summary, this function deselects the constant intersection first in order to highlight the specific section later.

I also have to say this is one of the hardest obstacles I’ve encountered in this project, as we haven’t really covered Javascript functions that much in class, and personally I do not have much experience with it either. It took me quite a long time to figure out how to make the function accurate, especially at the very last moment before the project was due. And this block of function only covers a minor part of my web page, which further makes it seem unworthy. Yet I definitely learned a lot about website making and how details are important and intricate for engineers in the process of creation. I definitely appreciate the knowledge and experience I gained from figuring out how to use this function.

Some of the other javascript functions I use are “onclick” and “hiddenmodel.” The “onclick” function can open the image sources of the bronze piece in a modal, and the second function will hide a modal when the viewer does not intend to get a closer look at the image by setting its display property to “none.” Moreover, since I also want to display text to describe the image, I would need to use the “element.alt” to change the “captionText” according to the specific name of each image of the bronze piece. Moreover, within the element alternative text, I further insert the Wikidata source as an embedded link below. Yet due to time constraints, I was only able to play with it for the first bronze piece on the web page of the *Boxer at Rest*. This insertion will provide the data link if the viewer wants to take a closer investigation of a specific bronze piece.

One of the other methods I used to make the webpage more aesthetically satisfying was the alignment of images for visual comparisons of Chinese and Greek bronze objects. In order to make three images align beautifully in one row horizontally, I created CSS classes in columns and rows to each adjust the image size and ratio and not to stretch distortedly by using styles such as “justify-content” and “columns gap.”

I also used Locolligo and Peripleo to present a map with all the data of where those Chinese and Greek bronze objects are currently located, exhibited, and collected at the moment. This map helps the viewer to get a sense of which region in the United States is most fascinating with bronze objects from antiquity, which is visually more effective and aesthetically satisfying than hardcore cold data. In order to transcribe data into map coordinates with descriptions, I first exported my spreadsheet as utf-8 csv files. Then by using the Locolligo site, I transcribed the csv files with assigned columns to a JSON file. Following the steps that we went over in class, I was able to display all my data from the spreadsheet to the Peripleo site embedded in my “Geographical Location” section on my web page. Moreover, Locolligo also provides connections with other sources such as Google Maps, where I was also able to put web resources of the exact location of those museums in street view. The assigned columns in Locolligo also allow me to insert the link to the Artstor site with more detailed information (such as size, time period, and provenance) to bronzes. In other words, I was able to connect Artstor and Google Maps street view within my Peripleo map. Unfortunately, due to the default setting of Locolligo transcription, I was not able to make color differences to distinguish the location of Chinese and Greek bronzes.

At last, I also put this reflection paper as a downloadable source for people who want to get a sense of how I finished this project and what are the specific processes. Again I use the Font Awesome CDN to symbolize the download function with the textual caption marking that this is my reflection paper.