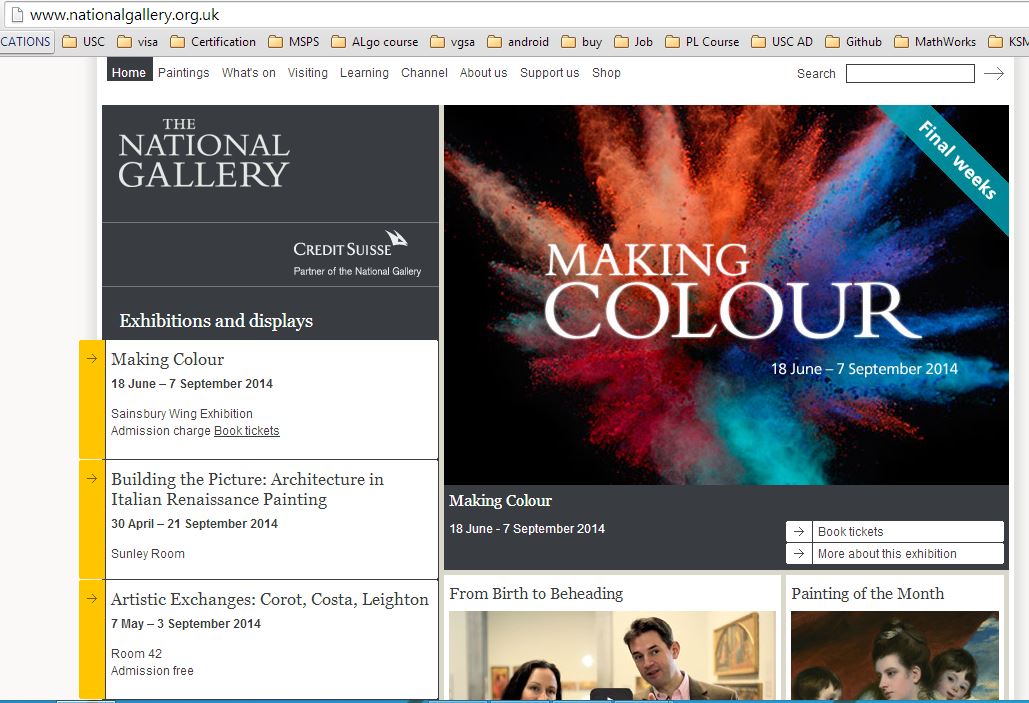
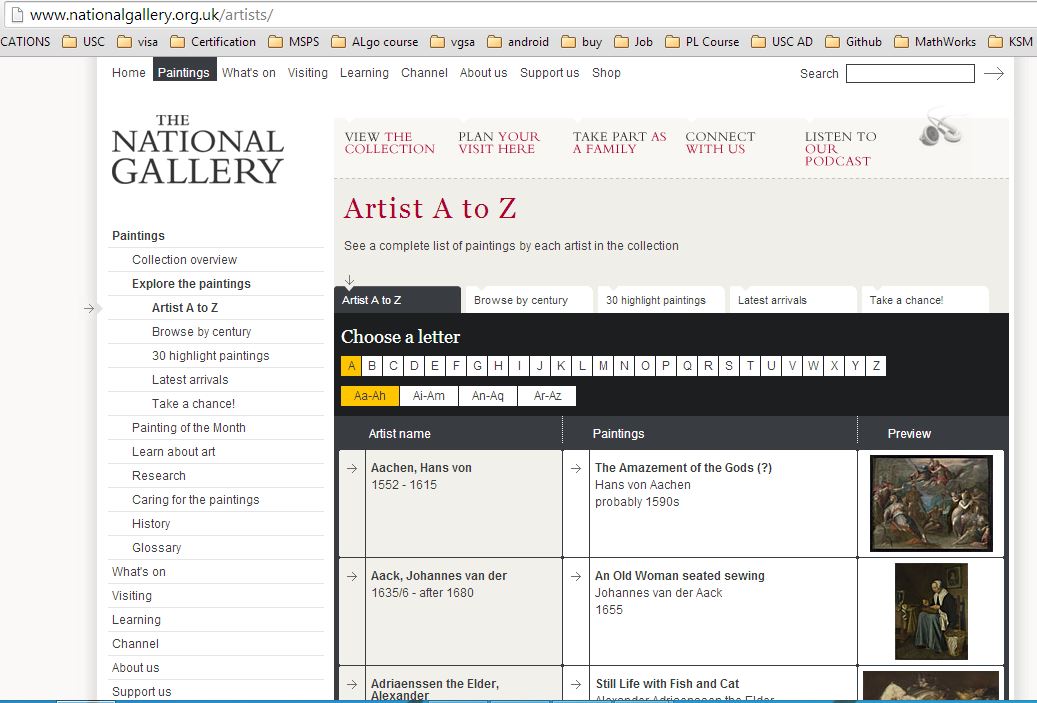
CSCI 548 – HomeWork 1

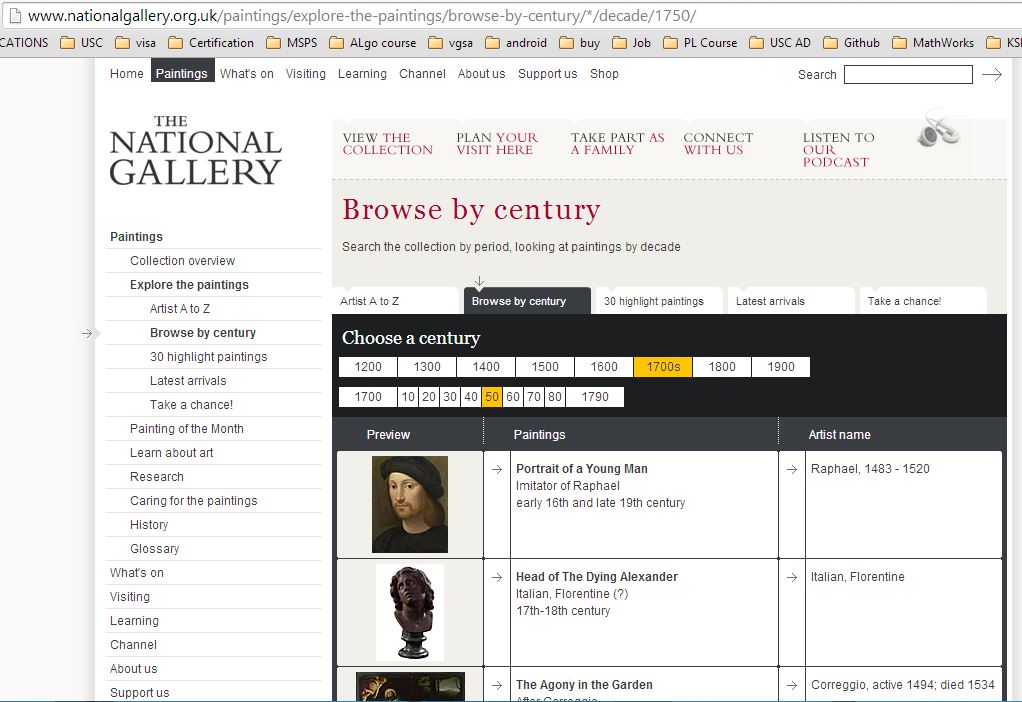
1. I, Nidhi Praveen Jain, declare that the submitted work is original and adheres to all University policies and acknowledge the consequences that may result from a violation of those rules.
2. Screenshots:
3. National Gallery Home page



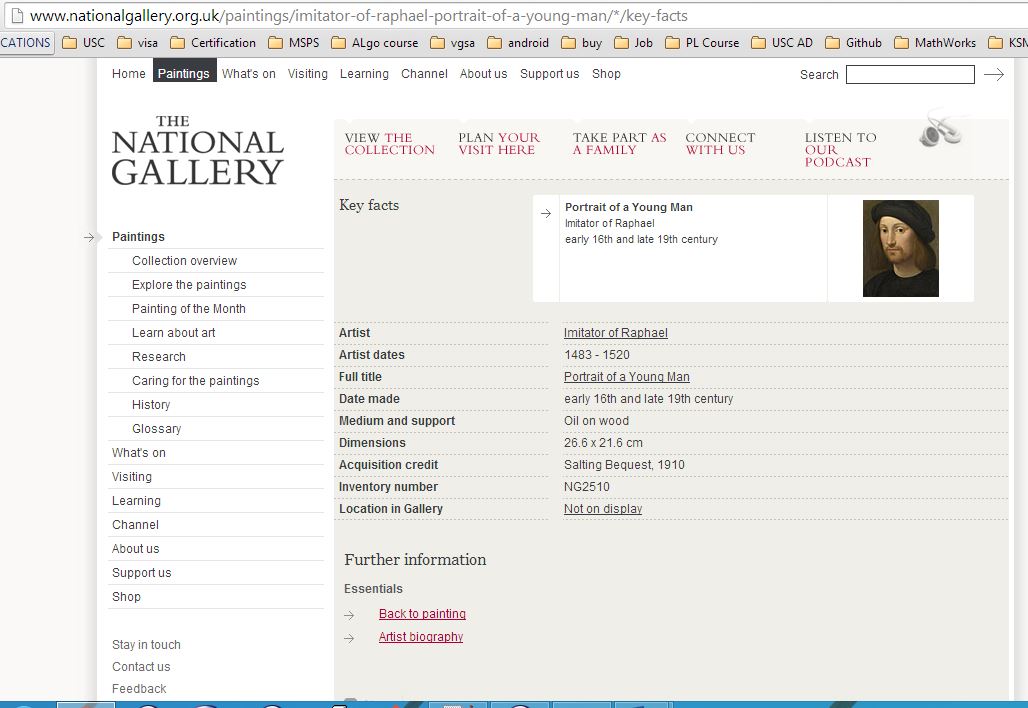
1. Paintings Page Overview



1. Paintings categorized by century



1. Individual Painting Key Facts Page



1. I have extracted the data in the form of JSON file. One individual JSON object looks like:

{

"Artist": "Giuseppe Maria Crespi",

"Image URL on Website": "http://www.nationalgallery.org.uk/upload/img/crespi-saint-jerome-desert-NG6345-ft.jpg",

"Artist dates": "1665 - 1747",

"Date made": "1710-20",

"Dimensions": "87 x 66.6 cm",

"Full title": "Saint Jerome in the Desert",

"Acquisition credit": "Bought, 1963",

"Location in Gallery": "Not on display",

"Medium and support": "Oil on canvas",

"Inventory number": "NG6345"

}

1. The website I am using has paintings categorized based on centuries. So initially, while working on the website, I was considering the centuries 1700, 1800 and 1900 since they covered all the paintings of those times. But, this provided only 150 odd paintings.

Thus, I had to try various other websites and tools. With few websites, the data was displayed as text and needed too much of substring functions.

Few other websites had Javascript disabled and so I decided to switch from JSoup to HTMLUnit, since JSoup does not parse Javascript. But, the website had dynamic data generated which I was unable to capture using HTMLUnit.

Another website displayed 36 paintings on the default page and would increase the number of paintings when “View More” button was clicked but the URL never changed. This made it difficult to extract data from, since having the button clicked through any of these softwares would be tedious or impossible.

I then decided to go with my original website and realized that the data was indeed more than 1000 paintings with a lot of duplicates but not all of them were. They just looked as if all the data is similar. After using ArrayList and checking for duplicates, I could successfully use JSoup to extract the data as needed.

1. I used JSoup tool for scraping the data from the museum website. JSoup is a JAVA HTML parser and I am very comfortable using JAVA so that was one of the reasons why I looked into tools related to JAVA. JSoup allows easy scraping and parsing of data using DOM and CSS. If the website is well structures, we can use ‘#’ and ‘.’ CSS selectors to easily extract any data from the HTML source.
2. I did not extract and display the image. The image is difficult to display in a JSON and so I considered not extracting it.

I have extracted the painting URL and I am using it to get more details but have not displayed in the JSON because it does not account for as a painting attribute.

There is a text description of each of the painting and artist biography in the form of text. The entire text could be extracted and displayed but that would make the JSON file very long. Extracting particular details from the text is difficult to anticipate using a tool.