

# Picard Cemetery GIS & Website

## PRACTICUM JOURNAL & USER GUIDE

TREVOR GWIN

### Completed Website

The website hosting the completed Picard Cemetery GIS is:

<https://picardcemetery.neocities.org/>

This is a free web hosting service created as an unofficial spiritual successor to Yahoo's Defunct GeoCities platform. It is my hope they continue to succeed as an organization and the website remains hosted for many years to come.

|

# Introduction

## Description

My senior practicum project is creating a website where families(or anyone else) can lookup burial sites in the Picard Cemetery, and see details about what is engraved on the headstone and the location of the headstone within the cemetery. The problem being solved being a lack of an easy to browse online database of the headstones in the cemetery and their locations.

## Approach

My approach to this project was to solve it in a manner that satisfies three main criteria:

- Useful
- Portable
- Updatable

*Useful:* It was very important to me that the result of my work be useful and usable in the real-world. I enjoyed working with the cemetery during past Geomatics Club community service events and while the scope of the project was constrained to the single semester of the course I am happy to report that the website is fully functional and contains records, photographs, and maps for all of the roughly one-thousand persons laid to rest in the cemetery.

*Portable:* It was my goal from the start to make the website simple enough that it could be hosted inexpensively on any simple web-hosting service. I avoided using any Esri products because I knew they would be somewhat locked-in to their platform and anything I host will be lost to time after I graduate, and my ArcGIS student licenses expire. The GIS aspect of the website is simple but needs not be any more complicated than it is to serve its purpose. The maps are simply pre-generated jpeg files showing a graphic of a pin placed at the (approximate) location for each headstone.

*Updatable:* The websites files are generated automatically using Python, a well-established programming language that is free and open source for anyone to use. I wanted to keep the software used to a minimum both for reasons of complexity and licensing. The only software needed beyond Python 3 are two plug-ins easily installed in Python's in-built software repository and the well-established "ImageMagick" open-source image manipulation software

## Deliverables

### The completed website

The website is delivered complete and fully functional. All files will be provided, and the website is also already hosted online with the platform Neocities.

### "Website Generator" Python Script & Supporting Files

My custom Python script reads a CSV file and with a single click generates the entire website in under one second. The script is relatively basic and written in a well organized manor with many comments to serve any future caretakers of the project.

## “Map Generator” Python Script & Supporting Files

The second Python script included takes a jpeg image of the headstone, extracts the GPS coordinates from the metadata of the image file, and uses them to edit a pin onto an aerial image of the cemetery showing the location of the headstone.

## Contents

Introduction .....	1
Description .....	1
Approach .....	1
Deliverables .....	1
The completed website .....	1
“Website Generator” Python Script & Supporting Files .....	1
“Map Generator” Python Script & Supporting Files .....	2
Scope of Work .....	4
Introduction .....	4
Background .....	4
Scope of Work .....	4
Quality Control & Assurance .....	4
Project Deliverables .....	5
Project Budget .....	5
Raw Data .....	6
Headstone Photographs & GPS Data .....	7
Aerial Imagery: .....	7
Computed & Generated Data .....	8
Aerial Image “Touch-Up” in GIMP Image Editor .....	8
Name, Birthdate, and Deathdate data entry from headstones .....	8
Headstone Location GIS Mapping .....	9
Webpage Generation .....	9
Completed Documents .....	10
Completed Website .....	10
Image Attributes CSV File .....	10
Map Generator Python Script .....	10
Website Generator Python Script .....	11

## Scope of Work

### Introduction

#### Background

Picard Cemetery is a small cemetery located near the town of Dorris, CA. In 2021 the Oregon Tech Geomatics club visited the site and performed monument locating, surveying, photography, and aerial photography.

For my Project I wish to continue to build off of this work by establishing monuments that will going forward allow easy grace-site plot establishment with simple measuring tape, as well as build a GIS system of the cemetery and it's individual plots. This project came-to-be from my experience helping with the Geomatics Club's work at the site, and my desire to get more hands-on practice using modern Land-Surveying Equipment while also leveraging the UAS equipment and photography available.

### Scope of Work

#### *GIS-level plot of individual burial sites.*

Due to the ambiguity of where the boundaries for individual sites lay my goal is to create a "GIS-level" plot of the sites consisting of my "educational-guess" at the boundary locations between the headstones. I will perform this work by tracing over a geo-referenced orthorectified photograph of the cemetery.

#### *Photography of Individual Gravesites w/ Georeferencing*

As part of the GIS I will prepare georeferenced photographs of each headstone. I hope to use a Topcon HiperSR GNSS rover, a FC-5000 data-collector, and a dedicated high-quality digital camera. I will need to experiment with the equipment to see what is possible, my preferred workflow and secondary workflow option are as follows:

- A) Plug the camera into the tablet and take photos tagged with the highly-accurate GPS coordinates from the rover.
- B) Record continuous observations with the rover and carry it with me as I take the photographs separately with the camera. Then later geo-tag the photographs with the GPS observation taken at the nearest time the photograph was. Ideally I would find and/or create a software option to automate this process.

#### *Monument Establishment for Burial Plot layout and alignment*

From what I currently understand the burial plots are laid out somewhat informally. I plan to work with the caretaker of the cemetery to determine going forward where the rows of headstones should be placed, and then establishment monuments at certain intervals among this rows so that going forward the cemetery can measure off of them and layout future burial sites in a neat row.

### Quality Control & Assurance

*California Department of Consumer Affairs: Cemetery and Funeral Bureau*

[https://www.cfb.ca.gov/about\\_us/mission.shtml](https://www.cfb.ca.gov/about_us/mission.shtml)

Seemingly more concerned with business practices and aesthetic upkeep of cemeteries.

## Project Deliverables

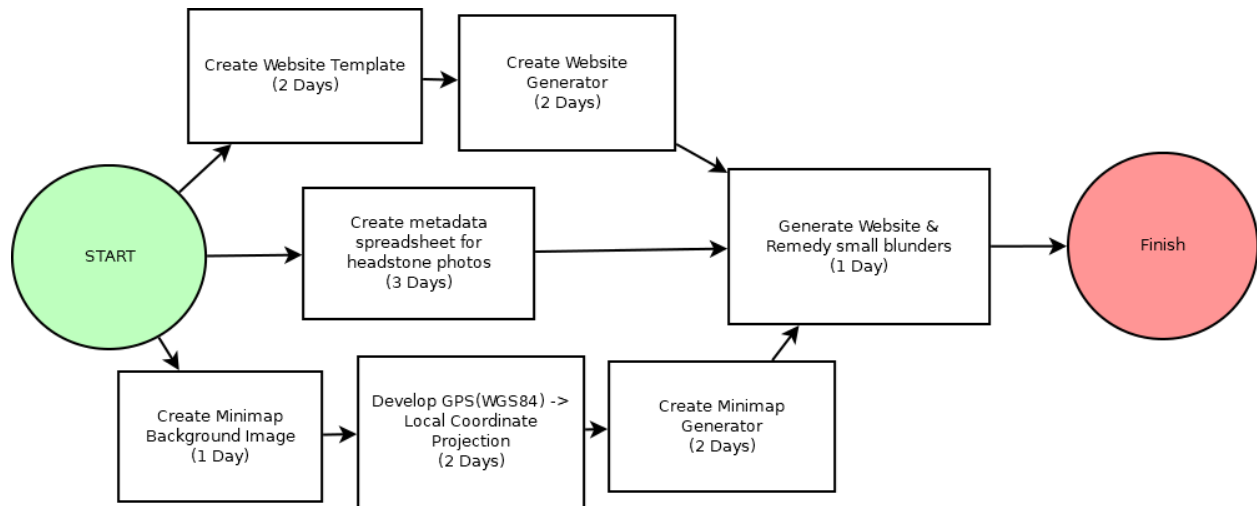
- GIS of Existing Burial Plots
  - Orhtorectified Imagery Used
  - Individual Burial Sites represented in Polygon Feature Class
  - Ability to pull up Grave-Site close up images
  - Initial Product in ArcGIS Online
  - Potential for ArcGIS independent product on Neocities or another webhost.
- Close-up Photography of headstones
  - High-Resolution images taken with Digital Camera
  - Geo-Referenced with GPS Coordinates precise to 10-20cm
- CAD Design for future burial site locations.
- 15-30 Control Points w/ monuments placed strategically to facilitate burial plot on-the-ground layout with basic Construction Tools and Methods.

## Project Budget

A total budget of \$3,401 broken down as follows:

	Cost	Qty		
<b>Drone Imagery Acquisition</b>				
2 Hours of Drone Operator	\$ 200.00	2	\$	400.00
Journey to Cemetery	\$ 1.50	48.00	\$	72.00
<b>Existing Plots from Digitalization of UAS Imagery</b>				
2 Hours Drafting Technician	\$ 90.00	2	\$	180.00
1/2 Hour Office Technician (Overview of work)	\$ 200.00	0.5	\$	100.00
<b>Headstone Photographes w/ Geo-Referencing</b>				
Survey Field Party	\$ 200.00	2	\$	400.00
Journey to Cemetery	\$ 1.50	48.00	\$	72.00
<b>CAD Design for Future Burial Sites</b>				
Survey Office Technician	\$ 200.00	5	\$	1,000.00
Drafting Technician (Clean-up)	\$ 90.00	1	\$	90.00
<b>Monument Setting</b>				
Professional Surveyor	\$ 200.00	2	\$	400.00
Hubs	\$ 0.50	30	\$	15.00
Journey to Cemetery	\$ 1.50	48.00	\$	72.00
<b>GIS Design</b>				
Survey Office Technician	\$ 200.00	3	\$	600.00

## Critical Path



## Timecard Totals

Task	Rate	Hours	Total
Digitizing Burial Sites	\$ 90.00	1.25	\$ 112.50
Renaming, Rotating, and Processing Headstone Pictures	\$ 90.00	1	\$ 90.00
Geo-referencing UAS Orhto to GCPs	\$ 300.00	1.75	\$ 525.00
Digitizing Burial Sites	\$ 90.00	1.5	\$ 135.00
Digitizing Burial Sites	\$ 90.00	2	\$ 180.00
Geo-referencing Photos to headstones	\$ 90.00	2.25	\$ 202.50
Web Design	\$ 200.00	7	\$ 1,400.00
Headstone Reading to Spreadsheet via Data entry service	\$60 Flat Rate		\$ 60.00
Projection Development	\$ 300.00	2	\$ 600.00
Minimap Generator Programming	\$ 200.00	3	\$ 600.00
Website Generator Programming	\$ 200.00	3	\$ 600.00
		<b>Total</b>	<b>\$4,505.00</b>

After tallying up the timecards, the total billable cost of the project came out to \$4,505. This is more expensive than the initial estimate, however I believe this is mostly to do with my relative inexperience in the work I was doing. Some aspects of the project I was effectively learning as I went which greatly inflated the amount of time spent compared to how long it would take me to do future projects. Nonetheless it was an interesting exercise in thinking about billing.

## Background & Research

### Historical Data

For this specific project I decided to submit the headstone photos to a data entry service in order to obtain usable data in the format I needed.

### Approach to Complete Project

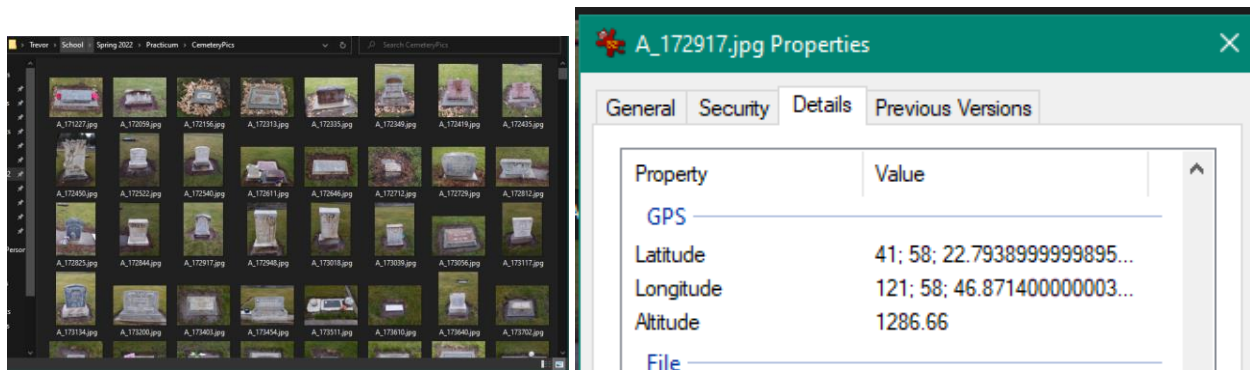
My approach to this project is to acquire raw data in the form of headstone photographs and attributes obtained by reading the individual headstones. This data will then be processed in order to generate the online GIS website.

## Raw Data

The only raw-data used consists of the Headstone Photographs and the Orthorectified Aerial Photo. The Three Dimensional Model was used to produce an image displayed on the homepage but this was more “for-fun” than necessary for the functionality of the project.

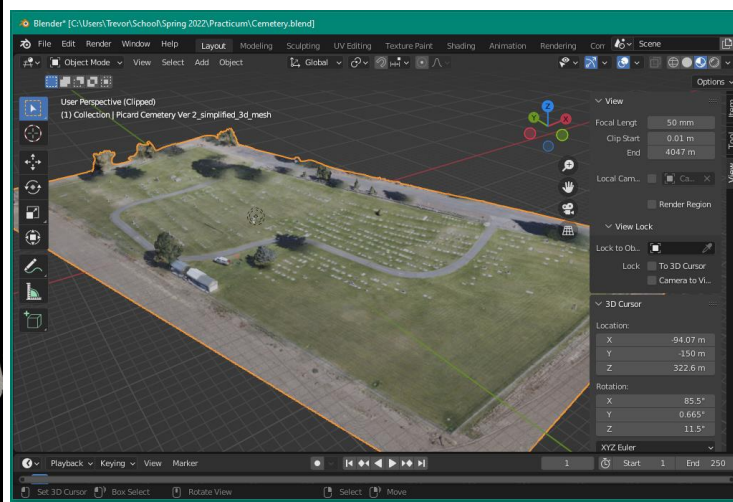
## Headstone Photographs & GPS Data

The Geomatics club went to the cemetery and using a combination of an Android Smartphone and Garmin GPS units with cameras approximately 800 Geotagged images were captured.



## Aerial Imagery:

I experimented with the open-source OpenDroneMap Software, but ultimately settled for using the products generated by Prof. Jack Walker with Pix4D because of their higher resolution. I imported the Three-Dimensional model into Blender but only ended up using it to generate a cosmetic aerial photo for the homepage of the website.

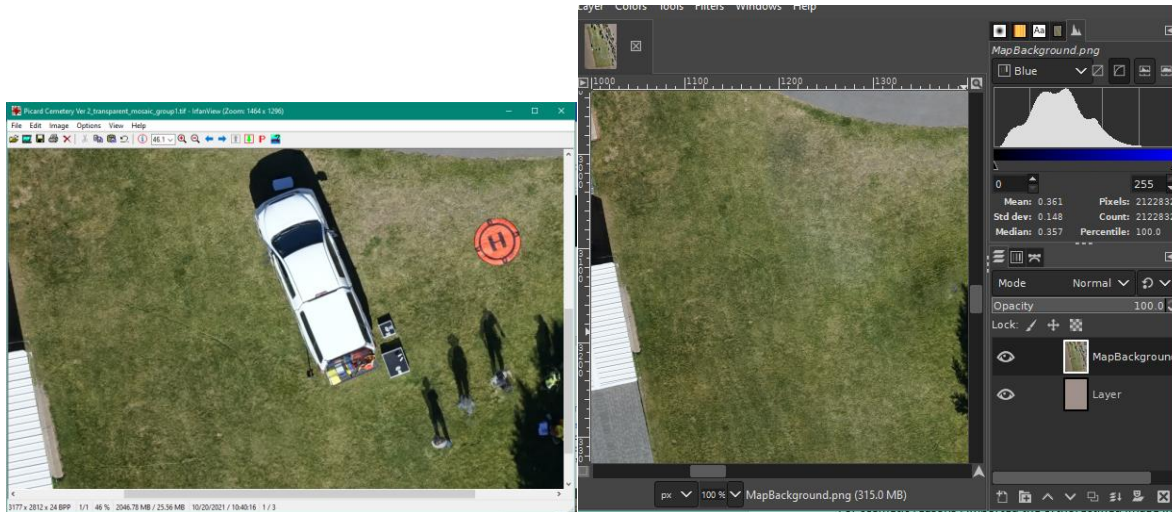




## Computed & Generated Data

### Aerial Image “Touch-Up” in GIMP Image Editor

For cosmetic reasons I imported the orthorectified image into the GIMP image editor and used the “Band-Aid” tool to cover various blemishes such as the monument location aids, the survey truck, and a total station that was setup in the middle of the cemetery. The tool worked great, it covered the truck by taking nearby grass imagery and blending it in.



### Name, Birthdate, and Deathdate data entry from headstones

I was not provided with any data on whom is buried in the cemetery. I went off the headstone images to obtain metadata. I entered several records manually but with approximately 1,000 records needing to be entered I decided to use an online data entry service in order to deliver a complete product instead of a proof-of-concept with only some of the data. The result of this was a table of data associating each image with metadata on who is buried at the site pictured, data that was vital to building the website’s sorted and searchable directory of burials.

	A	B	C	D	E	F	G	H	I	J
1	First Name	Last Name	Birth Mon	Birth Day	Birth Year	Death Mo	Death Day	Death Yea	Image File Name	
2	Perry A	Langer	1	6	1907	11	26	1996	A_171227.jpg	
3	Iva Mae	Langer	2	11	1909	7	7	2002	A_171227.jpg	
4	Richard L	Stevenson	1	25	1924	2	28	1948	A_172059.jpg	
5	Donald L	Craig			1911			1912	A_172156.jpg	
6	W. E. "Bob"	Marshall			1918				A_172313.jpg	
7	Renda J	Marshall			1920			1993	A_172313.jpg	
8	W O	Marshall	3	7	1859	1	29	1917	A_172349.jpg	
9	James A	Thompson			1876			1941	A_172419.jpg	
10	Rose May	Jones			1891			1969	A_172435.jpg	
11	Mary	Langer	1	10	1866	5	19	1931	A_172450.jpg	
12	Henry F	Langer	4	16	1892	2	10	1911	A_172450.jpg	
13	Albert	Langer			1863			1944	A_172522.jpg	
14	Mrs C	Ogden	12	5	1878	1	17	1917	A_172540.jpg	
15	Hilary	White			1930				A_172611.jpg	
16	Rita	White			1939				A_172611.jpg	
17	Gloria Der	White			1950			1951	A_172611.jpg	
18	Verma Lor	White	10	15	1930	7	7	2002	A_172611.jpg	
19	Shirlee Lo	Barbee	1	12	1930	11	30	1998	A_172646.jpg	

## Headstone Location GIS Mapping

Without doing too much re-inventing of the wheel, I wanted the GIS aspect of the website to be as simple as possible whilst still serving its function, and to be independent of any external platforms that may cease to continue hosting any maps. With this goal in mind, I settled on the approach of simply pre-generating a map for each headstone as a jpeg image. This makes the website as unbreakable as possible, as long as the image remains hosted the map will work indefinitely.

I developed a Map Generator in Python that reads each headstone image, extracts the GPS data from it, and using a custom and re-configurable Plate Carrée projection calculates the local coordinates on a jpeg image of the cemetery and using the free open-source ImageMagick software package adds a pin at that location. The pin image is a Public Domain work from OpenClipArt.org.

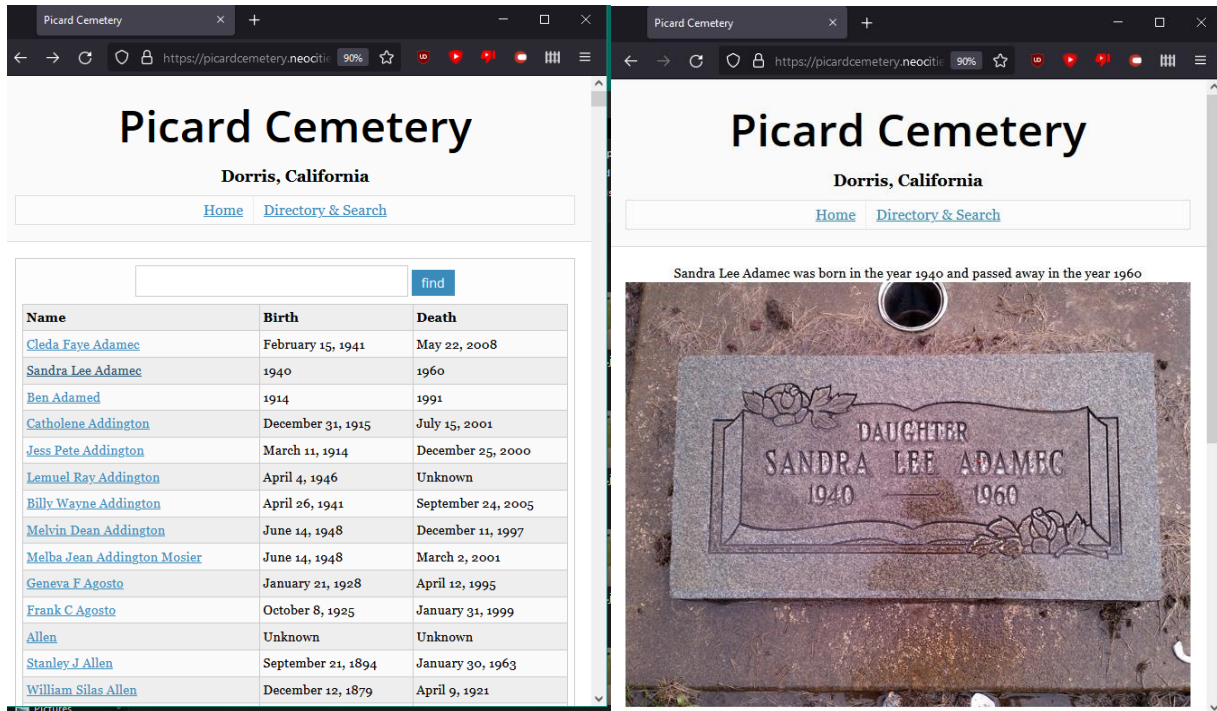
Should the aerial image used ever need replacing, the script is designed to easily adapt. The only values that would need to be modified are four clearly labeled variables specifying the GPS coordinates of two points, along with the local coordinates in the image that correspond to their location.



```
def projectToLocal(Coord):  
  
    #Coordinates for generating the Map Projection  
    NW_Corner_GPS=(121.9807459,41.9736235)  
    NW_Corner_Local=(309,396)  
  
    SE_Corner_GPS=(121.9790586,41.9718114)  
    SE_Corner_Local=(3673,5201)
```

## Webpage Generation

I am by no stretch of the imagination a professional web developer. Because of this, my goal was to use simple HTML to create the website. I created several “template” html files, and then wrote a Python script that reads them and fills in the relevant information from the spreadsheet of attributes for each image. This process takes one-click, as it is all contained in a single Python script. The script runs in under one second and generates the Homepage, Directory page, and the approximately 1,000 “obituary” pages for each person buried at each burial plot.



## Completed Documents

### Completed Website

The website is as of writing hosted online at:

<https://picardcemetery.neocities.org/>

This is a free webhosting service created as an unofficial spiritual successor to Yahoo's Defunct GeoCities platform. It is my hope they continue to succeed as an organization and the website remains hosted for many years to come.

### Image Attributes CSV File

This file is a spreadsheet listing each person buried in the cemetery, along with an associated image file. This file can be updated, and the Website Generator script re-ran as needed going forward.

### Map Generator Python Script

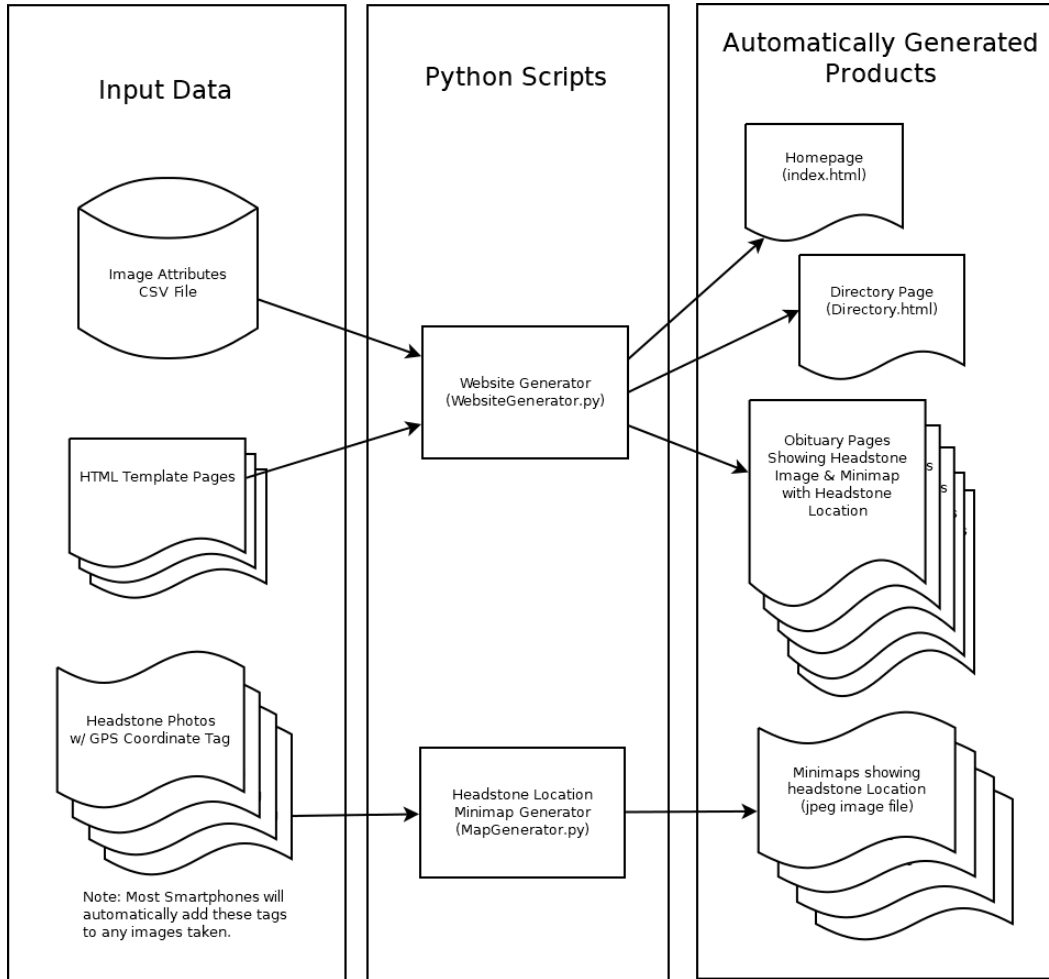
The map generator script includes the pin image and background image needed, it takes in Geotagged Images and produces images of maps of where to find the headstone in the photograph. It does depend on the following easy to install software packages:

- Python 3
  - Available in an easy to use Windows Installer at: <https://www.python.org/downloads/>
- Python Libraries via "PIP" installer
  - Pandas
  - Wand (ImageMagick Wrapper)

- ImageMagick
  - Available in an easy to use Windows Installer at: <https://imagemagick.org/script/download.php>

### Website Generator Python Script

This script is responsible for generating all of the html files that make up the website. The workflow of it works as follows:



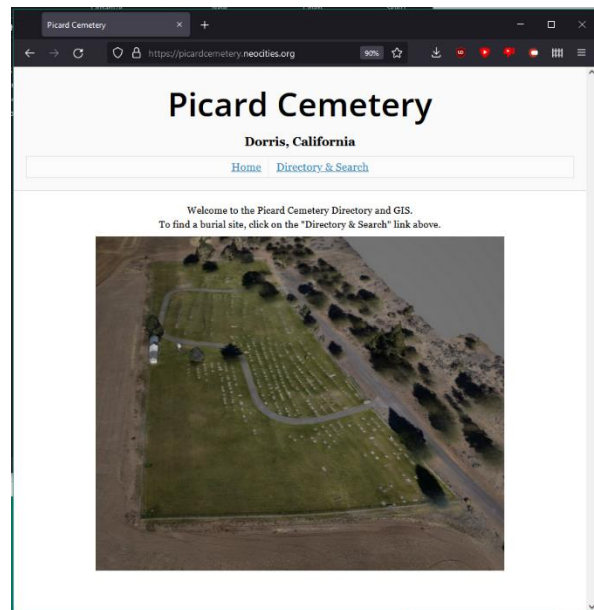
## The 1800's Cemetery you can visit in 2020's Fashion using 1990's Technology:

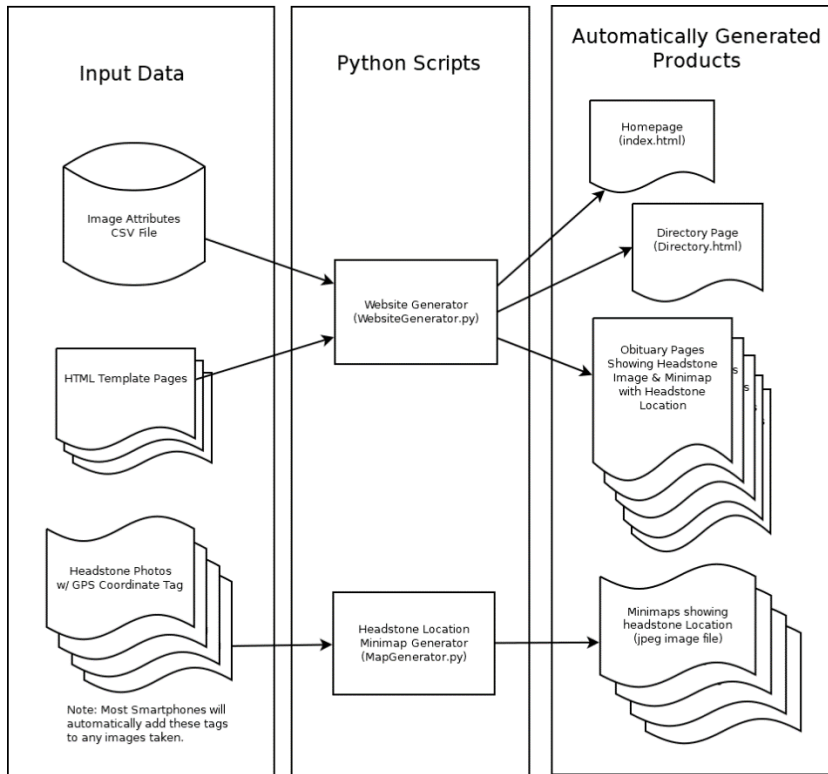
### Trevor Gwin's Picard Cemetery Directory and GIS.

For as long as I can recall using the internet has been a daily activity for me. I was born in the year 1995 and in my 27 years on this planet I've spent more time than perhaps one should exploring the digital spaces that make up the world-wide-web. In 2021 myself and others from the Oregon Tech Geomatics club visited and surveyed an old cemetery, the Picard Cemetery, in Dorris California. We found old monuments and set new ones, took two sets of aerial imagery, and as a trio of three descended upon the cemetery with GPS-enabled cameras to capture photography of over 800 headstones. As 2022 rolled around the project sat on the backburner for a bit, so when I had the opportunity to decide on my Senior Practicum Project, I decided to pick the cemetery without really knowing what exactly my project would be.

As the semester got started, I had to whittle down all my various ideas into an actionable project. Whilst my degree is in Land Surveyor my Minor is in GIS and I thought a GIS project would be the most real-world useful for those with loved ones buried in the cemetery. I find myself being a jack of many trades, a master of some of them. Whilst pondering over my various varied skillsets I decided developing a simple but functional website that serves as a directory and GIS for locating headstones would be the way to go. Technology get's a bad reputation for being fickle and prone to breaking. With this project I wanted to try and defeat this line of thinking by making the GIS in the form of a basic HTML based website.

Websites on today's internet are very complicated beasts. Web design is not my profession and I have absolutely no intentions of ever making it so. Instead of trying to jerry-rig a modern website together, I instead looked to the internet of the late 1990's and early 2000's. I wanted to use my skillsets with modern GIS, and modern Python programming, combined with my knowledge of basic web-design of a bygone era to create a rock-solid product that functions reliably for years to come whilst managing to not look too out of place on the modern web.





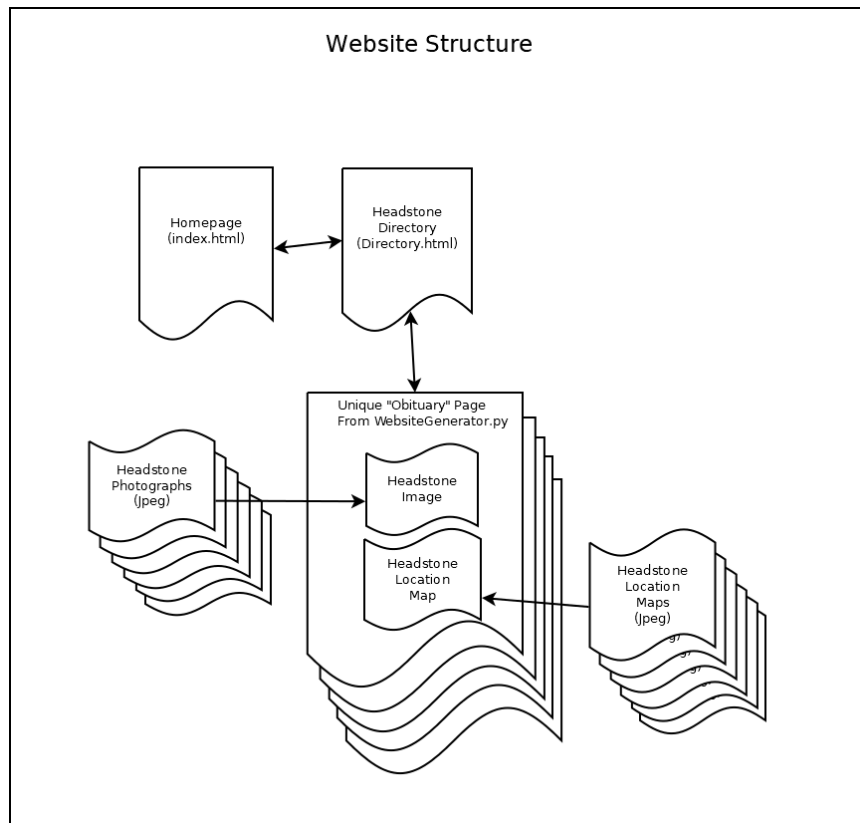
My approach was to program a “Website Generator” system. A spreadsheet with information on each burial plot I read into a Python script, and a page for each burial site is generated.

To add the “Geographic” aspect to our Information System, I also programmed a script that generates many maps, each showing the location within the cemetery of a given headstone.

ArcGIS Online maps may not stay forever, especially ones made by temporary student license holders. Google Maps embeds may change their API rules and break themselves.

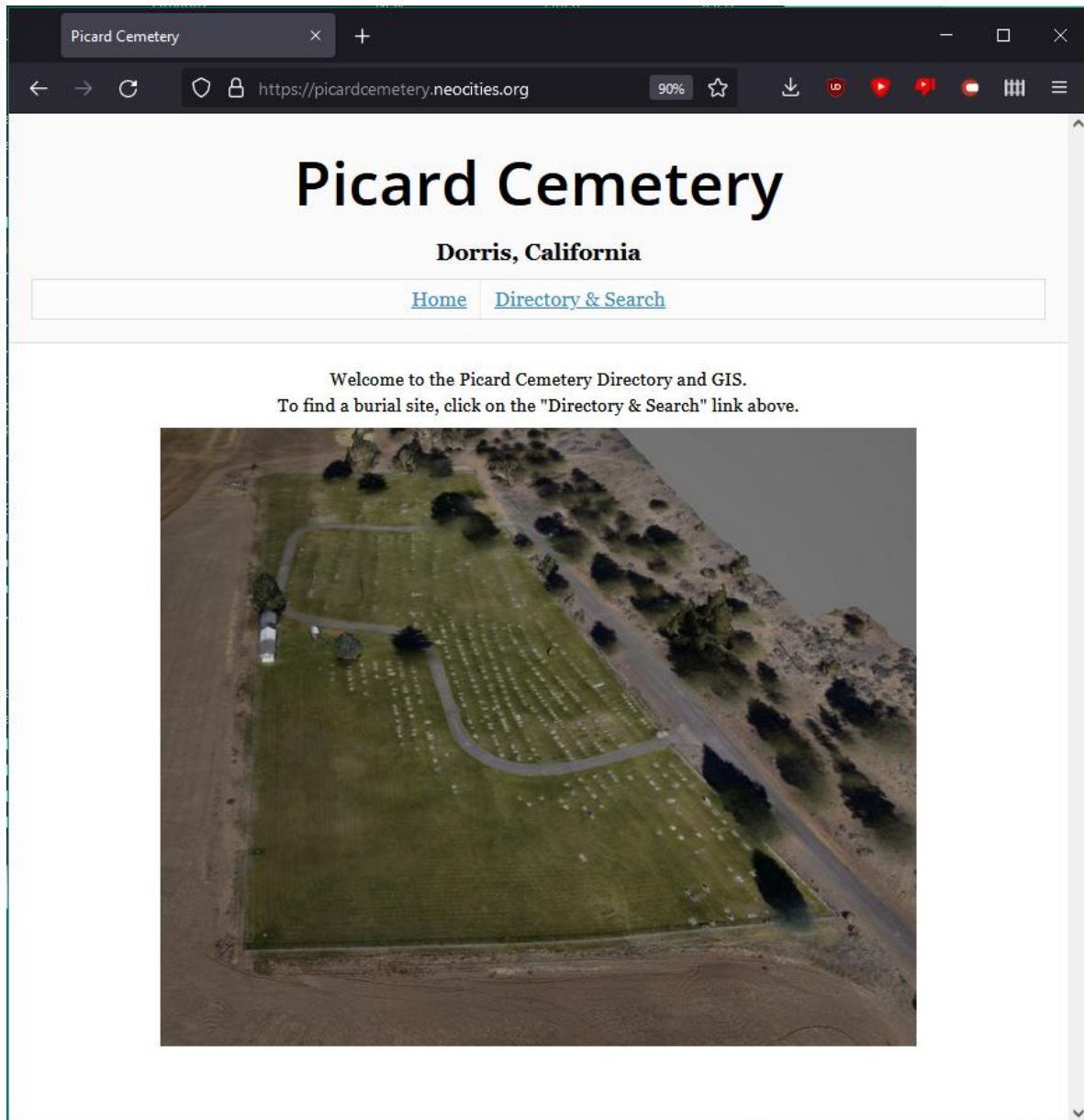
Jpeg images, however, are forever! At least as much as anything could hope to be in the fickle and changing digital world.

The resulting website structure is simple but remarkably functional. A pleasant homepage, a helpful directory of burials sorted by last name with a convenient search-bar, and for every person buried a tasteful web-page showing a photograph of the headstone and a map showing where to find it.



I am happy to have created a functional GIS that whilst not packed with features serves it's purpose and makes up for what it lacks in shininess with practical simplicity that is made to last. Please visit my website and see for yourself! It is graciously hosted free of charge by the fine folks behind Neocities, a spiritual successor to Yahoo's GeoCities service that gave so many people the opportunity to publish their two-cents or whatever else they were up to on the World Wide Web.

**<https://PicardCemetery.Neocities.org>**



```

1  from wand.image import Image, COMPOSITE_OPERATORS
2  from wand.display import display
3  from wand.drawing import Drawing
4
5  from os import listdir
6  from os.path import exists
7
8  #For testing access
9  NWtest=(121.9807459,41.9736235)
10 SEtest=(121.9790586,41.9718114)
11
12 def projectToLocal (Coord):
13
14     #Coordinates for generating the Map Projection
15     NW_Corner_GPS=(121.9807459,41.9736235)
16     NW_Corner_Local=(309,396)
17
18     SE_Corner_GPS=(121.9790586,41.9718114)
19     SE_Corner_Local=(3673,5201)
20
21     XScaleFactor=(NW_Corner_Local[0]-SE_Corner_Local[0]) / (NW_Corner_GPS[0]-
22     SE_Corner_GPS[0])
23     YScaleFactor=(NW_Corner_Local[1]-SE_Corner_Local[1]) / (NW_Corner_GPS[1]-
24     SE_Corner_GPS[1])
25
26     #print (XScaleFactor,YScaleFactor)
27
28     #For testing, use NW_Corner_GPS as our input coordinates
29     X=Coord[0]
30     Y=Coord[1]
31
32     Xp= ( (X-NW_Corner_GPS[0])*XScaleFactor ) + NW_Corner_Local[0]
33
34     Yp= ( (Y-NW_Corner_GPS[1])*YScaleFactor ) + NW_Corner_Local[1]
35
36     return (Xp,Yp)
37
38 def drawPin (Coord,ImageOutput):
39     pin=Image(filename="MapPin.png")
40     pin.rotate(90)
41     with Image(filename="MapBackground.png") as img:
42         with img.clone() as i:
43             with Drawing() as draw:
44                 draw.composite(operator='atop',left=Coord[0],top=Coord[1],width=250,
45                 height=250, image=pin)
46                 draw(i)
47                 i.resize(round(img.width*0.2),round(img.height*0.2))
48                 i.rotate(270)
49                 i.compression_quality = 75
50                 i.save(filename=ImageOutput)
51
52 def CoordFromImage (ImageFile):
53     exif = {}
54     with Image(filename=ImageFile) as image:
55         exif.update((k[5:], v) for k, v in image.metadata.items())
56         if k.startswith('exif:')
57     X=exif["GPSLongitude"]
58     #Convert from DMS to Decial Degrees
59     X=X.split(',')
60     X= ( int(X[0].split('/')[0])/int(X[0].split('/')[1]) ) + ( int(X[1].split('/')[0])/
61     int(X[1].split('/')[1]) )/60 + ( int(X[2].split('/')[0])/int(X[2].split('/')[1]) )/
62     60/60
63
64     Y=exif["GPSLatitude"]
65     #Convert from DMS to Decial Degrees

```



```
61     Y=Y.split(',')
62     Y= ( int(Y[0].split('/')[0])/int(Y[0].split('/')[1]) ) + ( int(Y[1].split('/')[0])/
int(Y[1].split('/')[1]) )/60 + ( int(Y[2].split('/')[0])/int(Y[2].split('/')[1]) )/
60/60
63     return (X,Y)
64
65
66 def drawMapFromGeotaggedImage (ImageName):
67     drawPin( projectToLocal( CoordFromImage(ImageName) ), ImageName)
68
69
70 # get the path/directory
71 folder_dir = "Generated/Obituaries/Images"
72 for file in listdir(folder_dir):
73     # check if the image ends with jpg
74     if (file.endswith(".jpg")):
75         if exists("Generated/Obituaries/Maps/map_"+file):#If the map already exists
skip it.
76         print("Generated/Obituaries/Maps/map_"+file+" ALREADY EXISTS! Skipping.")
77     else:
78         Coord=projectToLocal( CoordFromImage(folder_dir+"/"+file) )
79         drawPin(Coord,"Generated/Obituaries/Maps/map_"+file)
80         print("Generated/Obituaries/Maps/map_"+file+" Made!")
81
82
```

```

1  import pandas as pd
2  import re
3  from datetime import date
4
5  #Read in our CSV file and sort it by Last Name
6  data = pd.read_csv("Image Attributes.csv").sort_values("Last Name")
7
8  #Function to Pad date numbers with zeroes to make them compliant with the ISO date
9  #format, and thus accepted by the datetime functions
10 def toISO(month,day,year):
11     if(month<10):
12         month="0"+str(round(month))
13     else:
14         month=str(round(month))
15
16     if(day<10):
17         day="0"+str(round(day))
18     else:
19         day=str(round(day))
20
21     year=str(round(float(year)))
22
23     return year+"-"+month+"-"+day
24
25 #Function to read in a file as a (multi-line)string
26 #We use this to read in html templates
27 def fileRead(fileName):
28     return open(fileName,"r").read()
29
30 #Create a blank string to add table items to
31 #Despite the name of the variable, the variable is a normal string, not a Directory or
32 #a Table
33 DirectoryTableItems=""
34
35 for index, row in data.iterrows():
36     #Begin constructing a composite key
37     #This will be used for the name of the obituary webpage file
38     compKey=""
39
40     #Begin constructing a pleasant sentence to be written on the obituary webpage for
41     #the person
42     pleasantSentence=""
43
44     FirstName=row["First Name"]
45     #If there is no first name, just write a blank string to the variable instead of nan
46     if pd.isna(FirstName):
47         FirstName=""
48
49     #If there is no last name, just write a blank string to the variable instead of nan
50     LastName=row["Last Name"]
51     if pd.isna(LastName):
52         LastName=""
53
54     #Add name or placeholder to pleasant sentence and compKey
55     if FirstName == "" and LastName == "":
56         pleasantSentence+="The Person laid to rest here was born "
57         compKey+="Unnamed_"
58     else:
59         if FirstName != "":
60             pleasantSentence+=FirstName+" "
61             compKey+= re.sub('\W','',FirstName) + "_" #Remove any character that isn't
62             #a letter, number, or underscore
63         if LastName != "":
64             pleasantSentence+=LastName+" "
65             compKey+= re.sub('\W','',LastName) + "_" #Remove any character that isn't a

```

```

        letter, number, or underscore
62
63 #Add Birth Day to pleasant sentence and compKey
64
65 Image=row["Image File Name"]
66
67
68 #Attempt to convert the Birth Day to ISO format. If the day, month, and/or year and
missing we expect this to fail
69 BirthMonth=row["Birth Month"]
70 BirthDay=row["Birth Day"]
71 BirthYear=row["Birth Year"]
72
73
74 print(Image,FirstName,LastName,BirthMonth,BirthDay,BirthYear)
75
76 #If we have Month, Day, and Year:
77 if not pd.isna(BirthMonth) and not pd.isna(BirthDay) and not pd.isna(BirthYear):
78     BirthISO=toISO(BirthMonth,BirthDay,BirthYear)
79     pleasantSentence+= "was born "+str(date.fromisoformat(BirthISO).strftime("%B
    %e, %Y")) + " "
80     compKey+=BirthISO+"_"
81     BirthText=str(date.fromisoformat(BirthISO).strftime("%B %e, %Y"))
82 elif not pd.isna(BirthYear):
83     BirthYear=str( round(row["Birth Year"]) ) #Round and convert it to string,
otherwise it prints with a decimal place which looks silly.
84     pleasantSentence+="was born in the year "+str(BirthYear)+" "
85     compKey+=str(BirthYear)+"_"
86     BirthText=str(BirthYear)
87 else:
88     pleasantSentence+="was born on an unknown date "
89     compKey+="BirthUnknown"+"_"
90     BirthText="Unknown"
91
92 #Attempt to convert the Death Day to ISO format. If the day, month, and/or year and
missing we expect this to fail
93 DeathMonth=row["Death Month"]
94 DeathDay=row["Death Day"]
95 DeathYear=row["Death Year"]
96
97 #If we have Month, Day, and Year:
98 if not pd.isna(DeathMonth) and not pd.isna(DeathDay) and not pd.isna(DeathYear):
99     DeathISO=toISO(DeathMonth,DeathDay,DeathYear)
100    pleasantSentence+= "and passed away "+ str(date.fromisoformat(DeathISO).strftime
    ("%B %e, %Y")) + "."
101    compKey+=DeathISO+"_"
102    DeathText=str(date.fromisoformat(DeathISO).strftime("%B %e, %Y"))
103 elif not pd.isna(DeathYear):
104     DeathYear=str( round(DeathYear) ) #Round and convert it to string,
otherwise it prints with a decimal place which looks silly.
105     pleasantSentence+="and passed away in the year "+str(DeathYear)+"."
106     compKey+=str(DeathYear)+"_"
107     DeathText=str(DeathYear)
108 else:
109     pleasantSentence+="and passed away on an unknown date."
110     compKey+="DeathUnknown"+"_"
111     DeathText="Unknown"
112
113
114 #Add the image name to the compKey if there is one
115 if pd.isna(row["Image File Name"]):
116     compKey+="NoImg"
117 else:
118     compKey+=row["Image File Name"][:-4]#remove the last 4 characters to remove the
    extention

```

```

119     #Create an Imaged embed to add to the HTML page
120     imgEmbed='<br>\n'
121
122     #If there is neither a first or last name, change the first namne to something that
123     #will produce a clickable link
124     if FirstName==" " and LastName==" ":
125         FirstName="Unknown/Illegible"
126
127     #Create Table entry for the person
128     DirectoryTableItems+=str('<tr>\n')
129     DirectoryTableItems+=str('<td><a href="'+Obituaries/'+compKey+'.html'+'">'+
130     FirstName+" "+LastName+'</a></td>\n')
131     DirectoryTableItems+=str('<td>'+BirthText+'</td>\n')
132     DirectoryTableItems+=str('<td>'+DeathText+'</td>\n')
133     DirectoryTableItems+=str('</tr>\n')
134
135     #Make obituary webpage for the person
136     #Read in template pages
137     header=fileRead("Template/Header.html").replace('href=""','href="../')#Add an 'up
138     #one level' to the links since we're within the Obituaries Directory now
139     obituaryPage=fileRead("Template/Obituary.html")
140
141     #Replace template placeholders with generated values
142     obituaryPage=obituaryPage.replace("<!--header-->",header).replace(
143     "<!--pleasantSentence-->",pleasantSentence+"<br>").replace("<!--imageEmbed-->",
144     '",",header).replace(
159 "<!--directoryEntries-->",DirectoryTableItems)
160 #Write Directory page to file
161 open('Generated/Directory.html', 'w').write(DirectoryPage)
162
163 #Create Homepage
164 #Read in template pages
165 header=fileRead("Template/Header.html")
166 HomePage=fileRead("Template/index.html")
167 #Replace template placeholders with generated values
168 HomePage=HomePage.replace("<!--header-->",header)
169 #Write Directory page to file
170 open('Generated/index.html', 'w').write(HomePage)

```