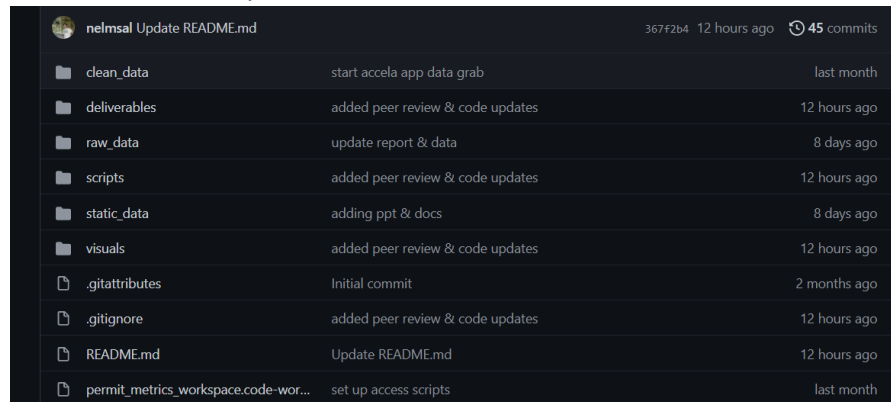


## File Organization

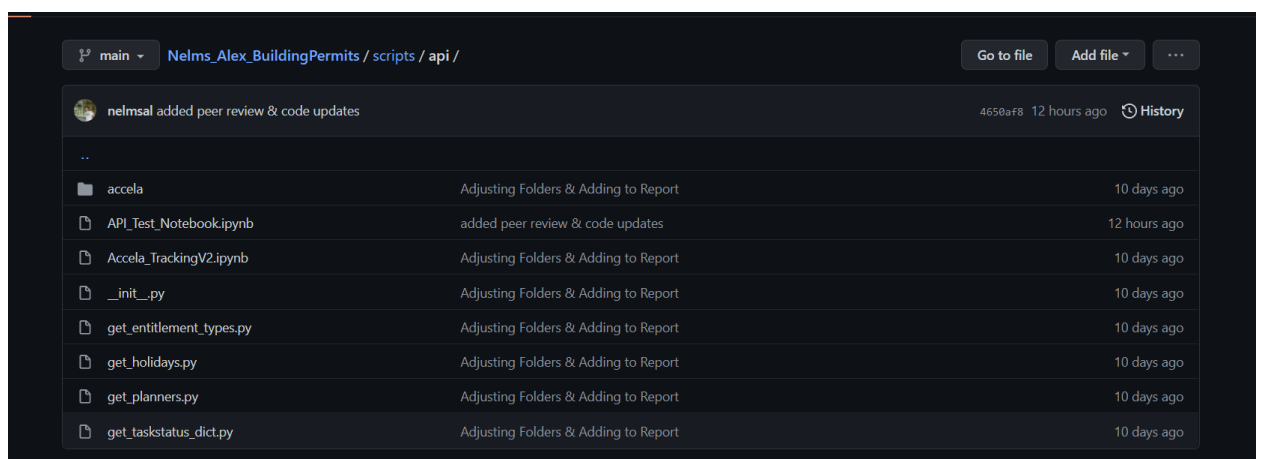
1. Is it easy to find what you are looking for?
  - a. Yes, the files are neatly organized, with each file understandable as to what is happening. I can tell the scripts are the Jupyter notebook files, the visuals are the graphics, the deliverables are the documents and presentations for the course, and the data folders are easy to find



A screenshot of a GitHub repository's file list for user 'nelmsal'. The repository is named 'Update README.md' and has 367F2b4 commit hash, 12 hours ago, and 45 commits. The file list includes folders and files with their commit messages and timestamps.

File/Folder	Commit Message	Timestamp
clean_data	start accelera app data grab	last month
deliverables	added peer review & code updates	12 hours ago
raw_data	update report & data	8 days ago
scripts	added peer review & code updates	12 hours ago
static_data	adding ppt & docs	8 days ago
visuals	added peer review & code updates	12 hours ago
.gitattributes	Initial commit	2 months ago
.gitignore	added peer review & code updates	12 hours ago
README.md	Update README.md	12 hours ago
permit_metrics_workspace.code-wor...	set up access scripts	last month

2. Are the files well-named?
  - a. The files are well named. For example, in the scripts under the API file, I can see all the different functions that Alex has written out per script.



A screenshot of a GitHub repository's file list for user 'nelmsal', specifically the 'scripts / api' directory. The repository is named 'Nelms\_Alex\_BuildingPermits' and has 4650af8 commit hash, 12 hours ago, and a 'History' link. The file list includes folders and files with their commit messages and timestamps.

File/Folder	Commit Message	Timestamp
..		
accela	Adjusting Folders & Adding to Report	10 days ago
API_Test_Notebook.ipynb	added peer review & code updates	12 hours ago
Accela_TrackingV2.ipynb	Adjusting Folders & Adding to Report	10 days ago
_init_.py	Adjusting Folders & Adding to Report	10 days ago
get_entitlement_types.py	Adjusting Folders & Adding to Report	10 days ago
get_holidays.py	Adjusting Folders & Adding to Report	10 days ago
get_planners.py	Adjusting Folders & Adding to Report	10 days ago
get_taskstatus_dict.py	Adjusting Folders & Adding to Report	10 days ago

3. Is the README helpful?
  - a. Honestly, there is not much else to say but a well-organized file collection with a nice readme. It lays out what the Author is looking to answer, and how he is going to answer it with the deliverables, along with the background motivation. Once he has his final deliverables done, I can imagine this will be filled out and updated.

### Question

1. How long does it takes to obtain a residential building construction permit for the cities of San Francisco, Berkeley, & Walnut Creek?
  - with how many Total Days, Working Days, Re-Submittals, & Public Hearings needed?
2. How much does the approval and permitting time vary depending on the: A. Building Characteristics, B. The Community's Demographics & Economic Factors, or C. Public Body Approval?

### Final Deliverable

1. A Dashboard measuring changes in permitting time & location
2. A Report analyzing the statistical & spatial differences in permits

### Background

Building Construction Permits can be notoriously difficult to receive due to the lengthy administrative & political system and policies. The public hearing bodies that approve permits & provide transparency to the process, have become bottlenecks for approval. The San Francisco Bay area is empirically known for its long permitting process & notorious unapprovals. The slow permit process has been one of the factors leading up to the area's housing crisis.

The permitting process has largely gone under-researched. Because local government typically has suffered from siloed data, overworked permitting systems, and data illiteracy, it is difficult to get accurate measurements of the permitting process. Without this data, it is difficult to use statistical or spatial analysis to isolate the leading factors of the permitting process.

Because of my work with Walnut Creek, I have already created Python scripts that pull permit timelines from Accela, process each task into time measurements, then aggregate them. Luckily, many West Coast cities contract Accela to manage their permitting system through software and structured data. So expanding the scope of this analysis into other Bay Area cities wouldn't be as difficult.

## Code

1. Is it easy to understand what the code is doing?
  - a. Yes, for the most part the code is easy to understand what is happening. A few titles or comments here and there wouldn't hurt though, and maybe some titles for the charts just if someone has to pick this up, they can do a quick skim and see what the visuals are about. For example, what are the two lines representing in the building permits close out date.

```

date_group(date_col).plot(ax=ax, kind="line")
#date_group(date_col, period="Y").plot(ax=ax, color='black', kind="bar")

date_col = 'date_close'
date_group(date_col).plot(ax=ax, color='red', kind="line")

ax.set_xlim([date_start, None])
ax.set_ylim([0, None])

plt.show()

```



2. Is it modular? Do they repeat code?
  - a. Its very Modular, Alex has well written functions that he uses instead of writing numerous code. He has this set up through different function files.

```

1]: from matplotlib.legend import Legend
    from matplotlib.legend_handler import HandlerBase

    class TextHandlerB(HandlerBase):
        def create_artists(self, legend, text ,xdescent, ydescent,
                           width, height, fontsize, trans):
            tx = Text(width/2.,height/2, text, fontsize=fontsize,
                      ha="center", va="center", fontweight="bold")
            return [tx]

    Legend.update_default_handler_map({str : TextHandlerB()})

    def set_label(
        bins,
        zero = True,
        binder = '-.'
    ):
        binder = ' {} '.format(binder)

        if zero == True:
            bins = [0] + bins

        bins = list(zip(bins[:-1],bins[1:]))

        def to_string(pair):
            paired = str(pair[0]) + binder + str(pair[1])
            return paired

        bins = [to_string(bin) for bin in bins]
        bins = [
            '0' if bin == '0' + binder + '0'
            else bin
            for bin in bins
        ]
        return bins

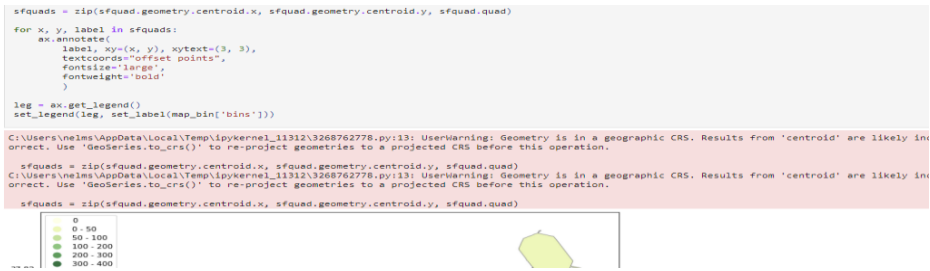
    def set_legend(leg, new_texts, location=2):

        leg._loc = location

        for lbl,new_text in zip(leg.get_texts(), new_texts):
            label_text = lbl.get_text()
            #lower = label_text.split()[0]
            #upper = label_text.split()[2]
            #new_text = f'{float(lower):,.0f} - {float(upper):,.0f}'
            lbl.set_text(new_text)

```

3. Are things well-named? Overall “Code-Smells”
  - a. There aren’t any real comments or titles of sections as to what is being performed, but when you actually go through the code line by line, you can see how he is cleaning it and filtering out unneeded information.
4. Are you confident this code is error-free? Is it hard to tell?
  - a. For the most part I believe this code is error free, except for some warnings that just need to be looked at a second time



5. If you inherited this project tomorrow, would you be able to succeed with it?
- In terms of how the code is written and organized, I believe that yes, I will be able to succeed with it. It is a lot better organized than my code tbh, and Alex is a pro at this.