**Project Name:** Data Visualization: Auto Insurance

**Project Objective:** Provide users an interactive means to explore data themselves. For this project, allow users to discover auto insurance information based on provided policy holder data.

**Project Steps:**

1. Reading .csv into database… Completed via PostgreSQL
2. Flask API development completed via Visual Studio Code
   1. Education endpoint utilized
   2. Return all the rows (array) for the education column, providing a total count of policyholder\_education\_level types
   3. Possibly add more endpoints later (i.e., more js libraries, charts)
3. Create one pie chart by reading Flask API via Visual Studio Code (utilizing D3 and chart.js)
   1. Can address policyholder ID later… Completed pie chart visualization per agreed approach

**API / Datasets:**

* CSV: insurance data from Kaggle: insurance\_claims.csv
  + File reflects policyholder data across 39 categories for 1,000 individuals
  + Includes demographic, policy, claim, and vehicle information
* API: build from .csv file
  + Dump .csv into database via postgres, then use flask to make it into an API

**Database:** SQL

**Front end or back end preference:** Front end

**Project type:** A dashboard page with multiple charts that update from the same data

**New JS tool used:** chart.js

**Visualization must include:**

* Python-flask powered API, HTML/CSS, JS, and one DB (SQL)
* User-driven interaction: Dropdowns from which shoppers select their vehicle and residence location… With this detail nearby policyholder information can be provided through textboxes
  + Narrowed project scope to visualization of one endpoint in a pie chart (education level)

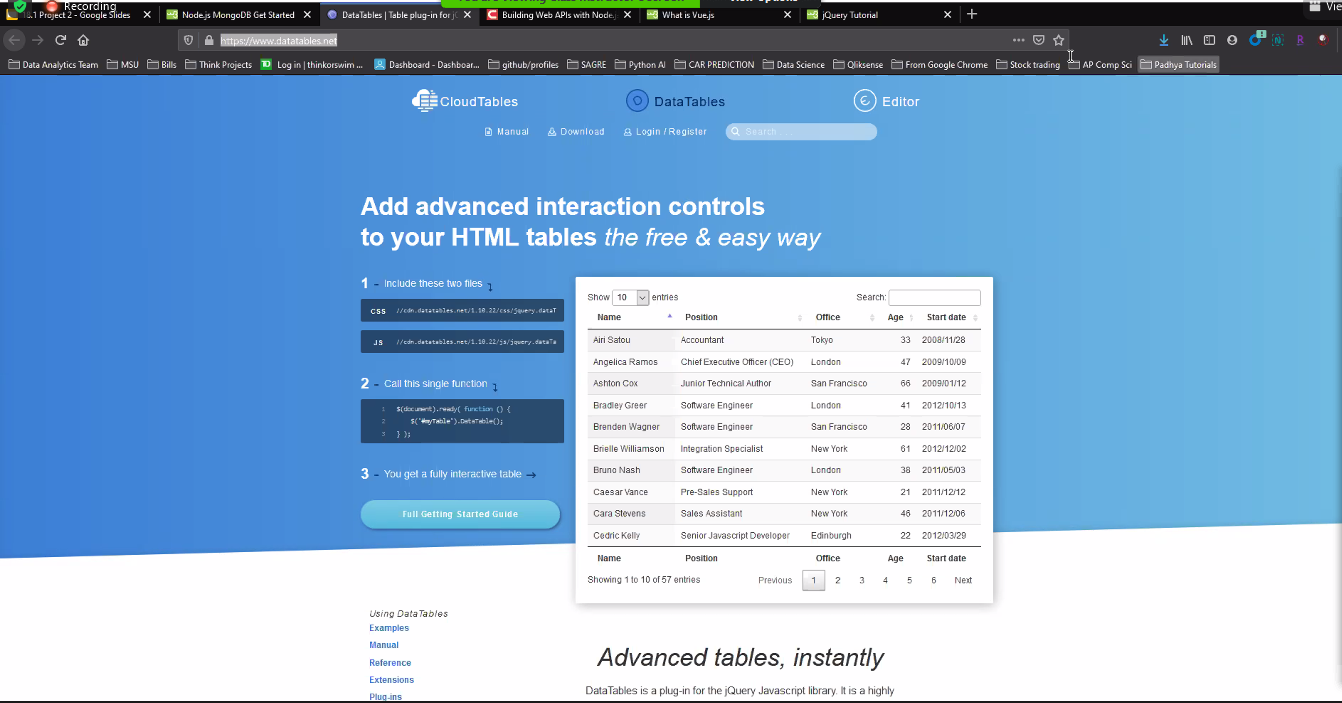
**Outstanding concerns / to-do’s:**

* Heroku deployment
  + Followed documentation provided, still need to set up sql database with Heroku
* Back-end (api folder) app.py file
  + Still need to hide postgres password
* Heroku deployment
  + Followed documentation provided, still need to set up sql database with Heroku
  + Create initdb.py for my project
  + Involves sqlalchemy and Setting up sql and Heroku… create new db from Heroku?
* *Use kaggle for domains / datasets*
* *D3 plus Plotly for third option*
* *Three analyses or three types of reports for item #6*
* *Start determining topic (domain), then find dataset, then sketch visual ideas*
* *Ideas:*
  + *Auto insurance shopping*

*Do’s and Don’ts:*

* *Don’t need huge efforts… no Amazon expected!*
* *Focus should be to make you familiar with D3, dashboards, etc… whatever you are working on*

*Helpful:*



* HTML, makes columns sortable
* [www.awesomeopensource.com](http://www.awesomeopensource.com)
  + A source for project ideas

This project is about visualization. Don’t worry about multiple data sets.

Can satisfy all requirements with my csv spreadsheet; create flask api

Pull together sketches re. visualization ideas

Open csv, get familiar, then sketch ideas on paper and share

What do I want project to look like to the user?

I have data (csv), front end, then how to make it happen with sufficient amount of technologies

Don’t need multiple datasets

Thinking of visualization will help in setting up API… think back and forth across each

Hint: return a list of rows, each field is a column, then I can filter the columns… see Slack

Probably have one endpoint, then filter

Have an endpoint for an entire row plus for each column an endpoint that only returns that column (ten columns would require eleven endpoints)

Entire row endpoint is more important than column endpoints

Maybe just start with education column endpoint…

Then could do the entire row endpoint

Project steps:

1. Reading csv into database
2. Flask api development via Visual Studio Code
   1. Education endpoint… see Kourt’s (Jess) reference, should really help
   2. Return all the rows (array) for the one education column… for total count of policyholder\_education\_level types
   3. Full row endpoint
   4. Possibly more columns
3. Create one pie chart by reading flask api via Visual Studio Code (via D3 first, then via chart.js)
   1. chart.js does not require a download / install
   2. Can work on policyholder ID later; build visualization now
   3. I have enough to start work on my pie chart
4. Then add more endpoints, more js libraries, and more charts

Treat the four steps above as separate assignments, don’t force together

Use policy number as primary key… update QDD with this

Connect postgres on local to Heroku

Briefly, I need to:

1.  Create first Flask API endpoint  
2.  Create first chart  
3.  Migrate to Heroku  
4.  Add more endpoints and more charts

Print out 4-step plan and place on wall for reference… when working on an individual step, work 45 min and take a break, etc… again try to tackle a piece at a time

Re. #2 above, if no filter give all ed level for all policyholders

A better idea is to accept a parameter such as policy number, education number

We’ve used several APIs in class… basically api is a website that hits certain URLs and give info back

So, each endpoint is a particular function within the api, like the methods in goodreads

Full row API gives a full row, all info, for a particular policy holder

At first, do all row (no filters)… the ‘full row” endpoint

No filters will give array… entire table this is second step; first step is array of policy number and education (will be JS as an array: policy number followed by education)

First: If I don’t give a parameter (pol number), give me all education (or whole table)

Dump csv into database via postgres, use flask to make it into an api

Start with one pie chart idea from one csv endpoint… education

From csv to database, then flask api, then which plotting tool can query my api

Have something for Wednesday… Sketches, pie chart, etc

Python based flask api; can then go to visualization work from there

First, plan how do I want to show dashboard on front end… Consider user, what they will see/want

1. Take data, dump into postgres, get ready to use in Heroku deployment
2. Still dump csv into database via postgres, use flask to make it into an api
3. Scaling… if a free api, could use

refer to previous homework

could use a zip code api that translates to city/state so users could see “neighbor” rates

be familiar with data and know my visualization tool goals

focus on front end for users… what does it look like for users?

Could use google maps api to look against user inputs

Based only on csv, think of a visualization

Could create a map of claims with zip codes (leaflet map) with markers

Could put amount of cost, chloroplast of state averages

Not really a rate quote model anymore, more of a reflection of area data using markers

Could do pie chart and other types around the data set

Think of types of plots, what to show with the data, etc.

The flask will be some work, as the Heroku part

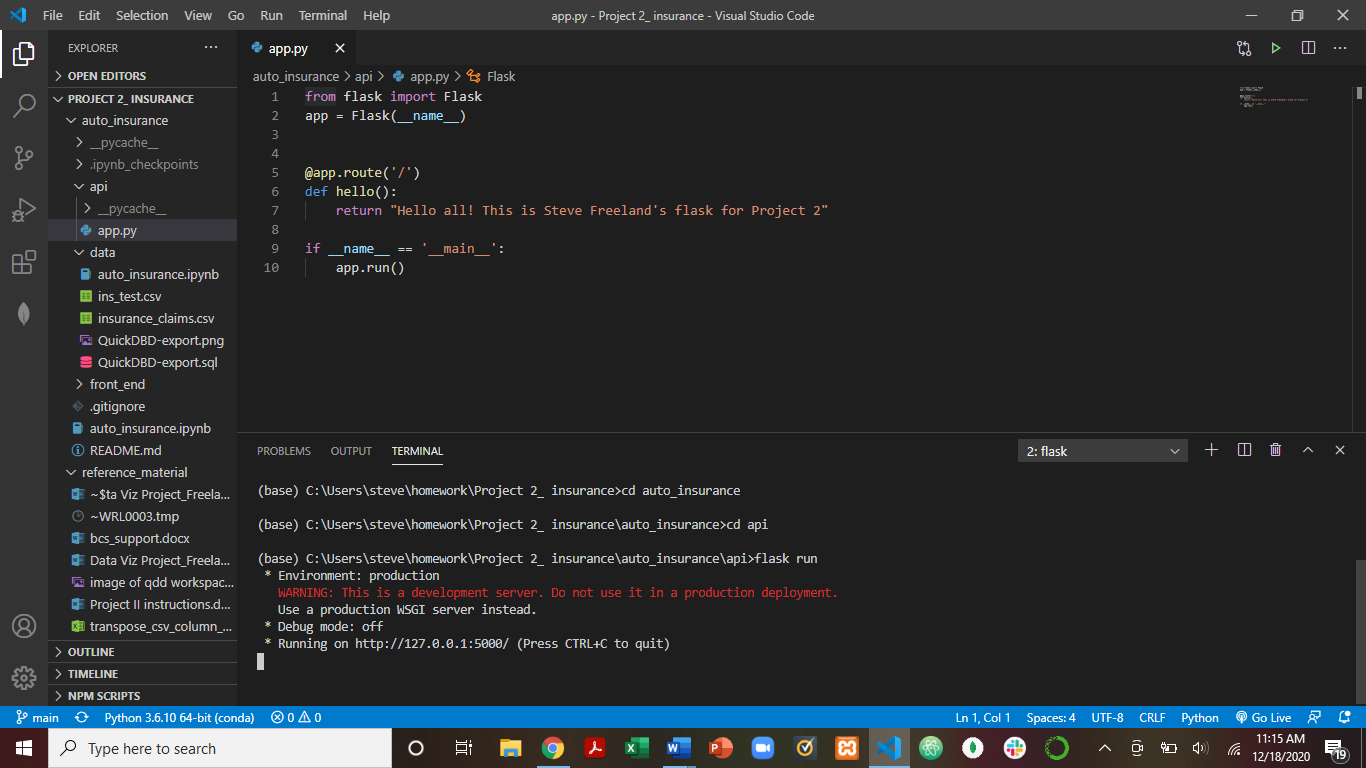
Keep it simple, at end of day should be able to deploy… doesn’t have to be super complicated

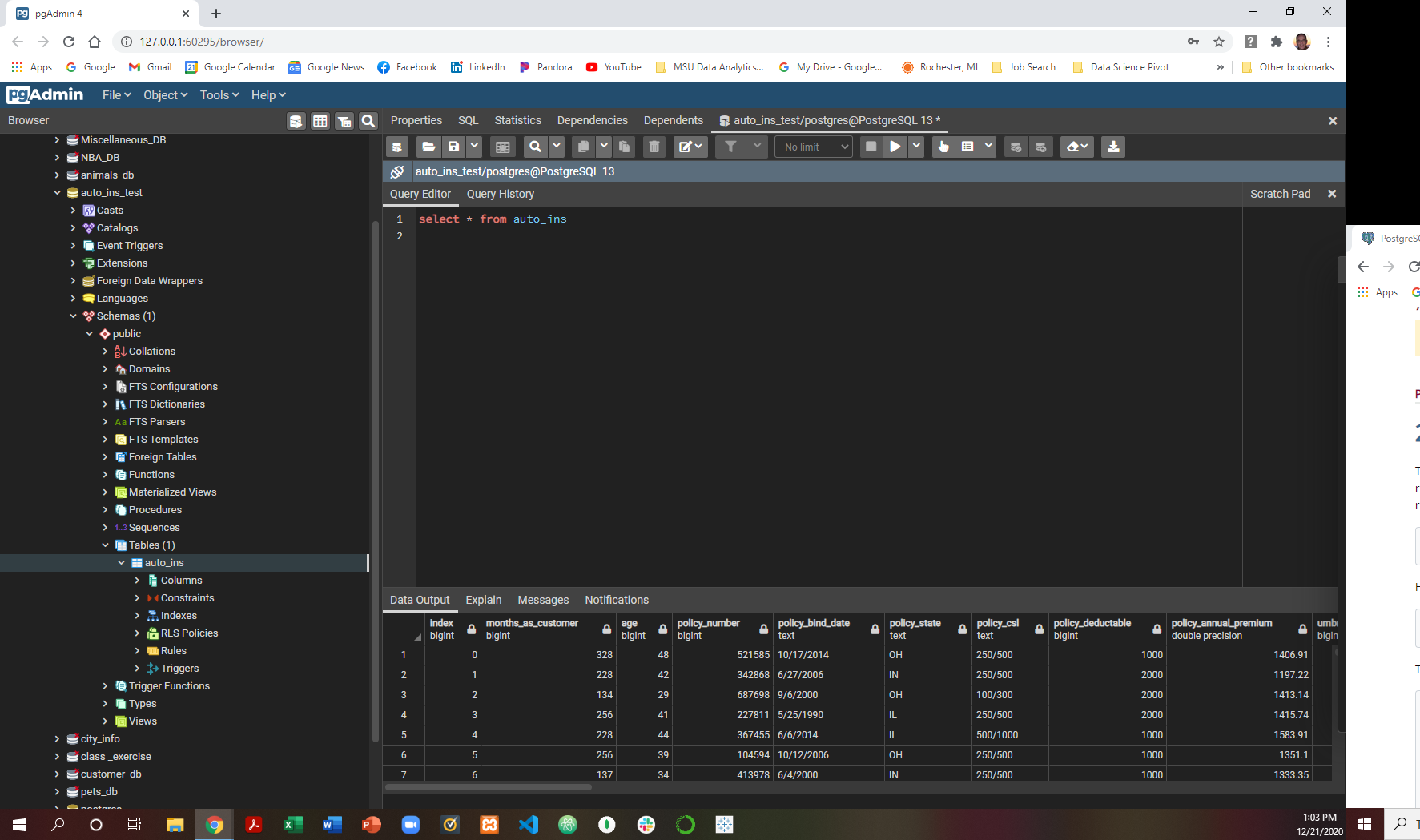
Scalable if a free insurance industry API was available

Share example of dashboard

To run flask…

* Use anaconda prompt





10-minute presentation template:

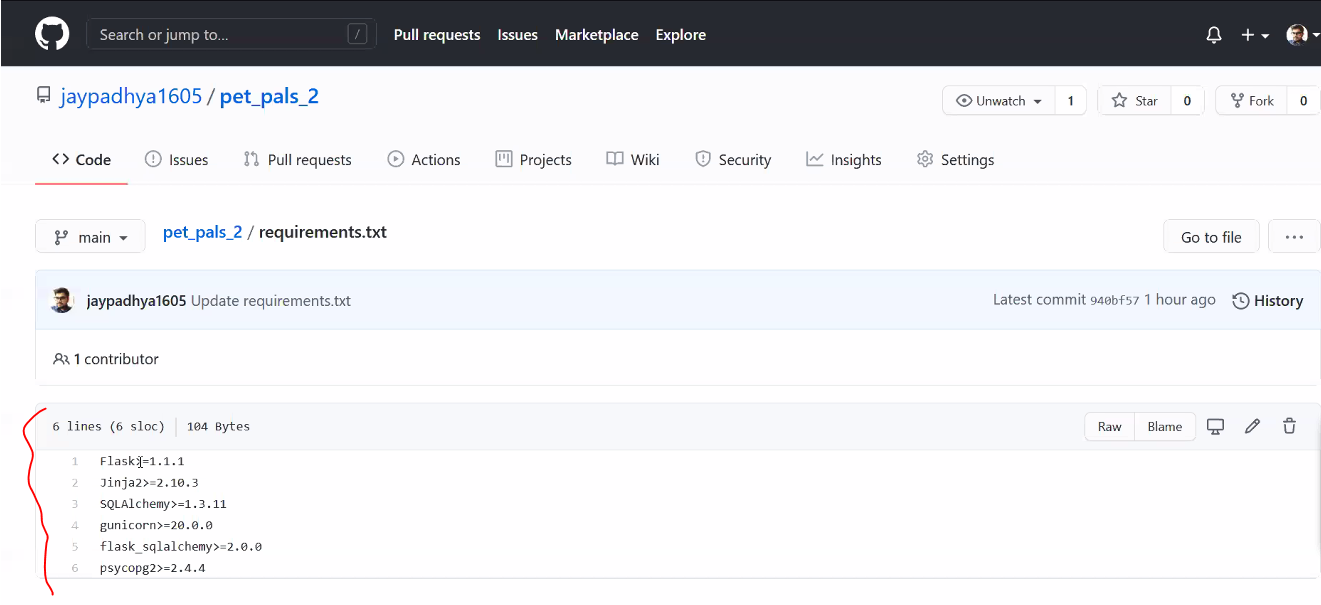
What we did

Approach

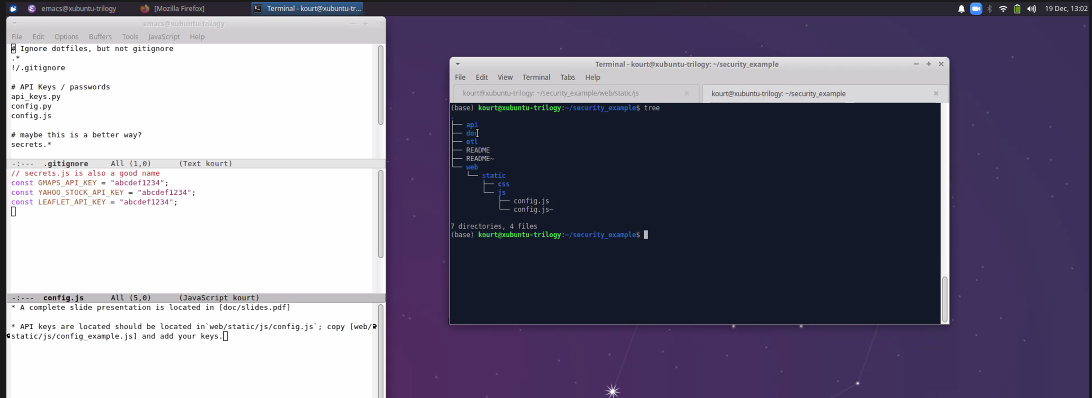
How did we generate flask api?

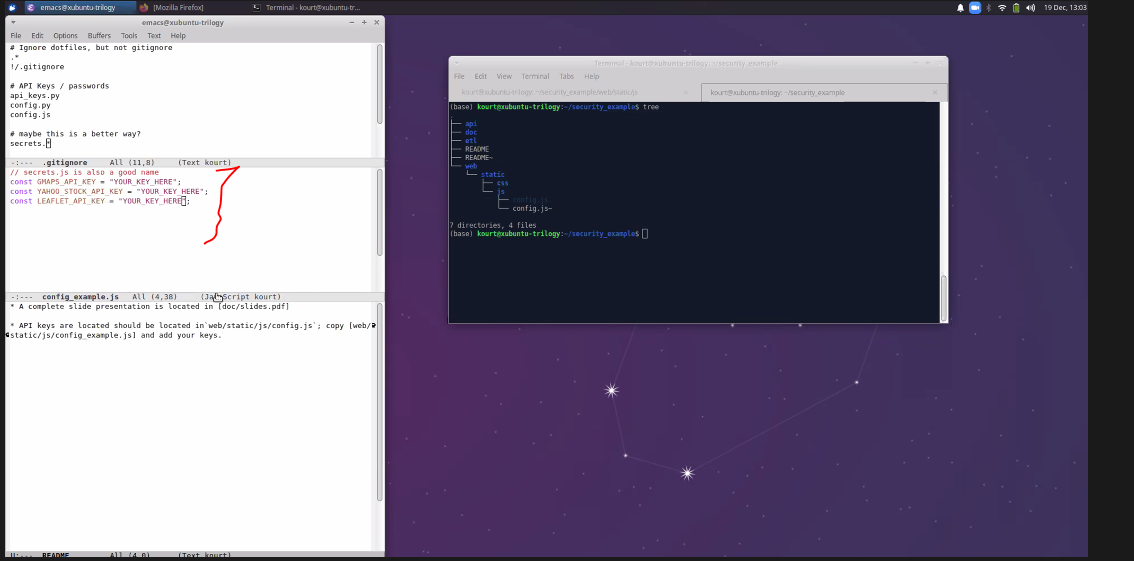
Etc

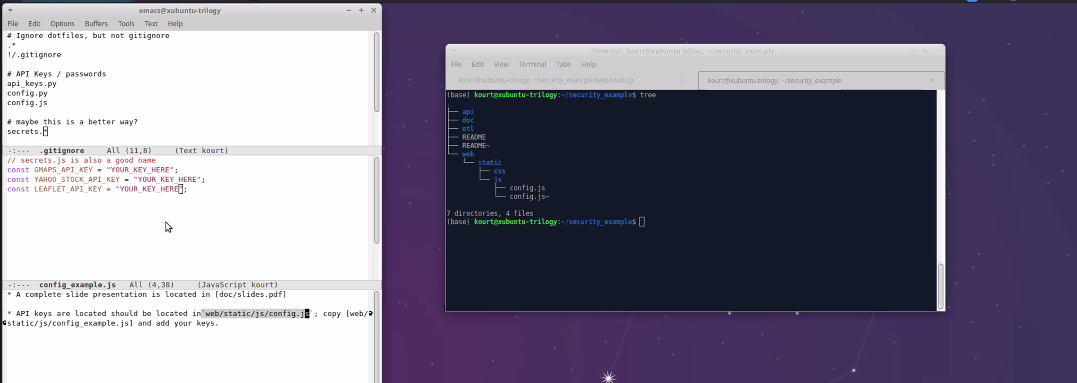
Map project template



* Security:







* Make sure gitignore includes all API keys, not just those in Python