Please complete this challenge as quickly as you can in Python 3.

Please provide code files for each step of the coding challenge to show the evolution of your solution.

1. Create a function that takes a location and finds the nearest SEPTA Regional Rail train station using this data set: https://www.pasda.psu.edu/kmz/SEPTARegionalRailStations2016.kmz
    a. The return format of the train station must be in GeoJSON.
    b. Bonus: Give walking directions to the train station.
2. Build an HTTP API that exposes that function.
3. Ensure your API does not search for the same location more than once at a time, even if multiple instances are running concurrently.
4. Make your API as cost-effective to operate as possible, given you will charge for each location searched. Explain what factors you took into consideration and what mitigations you put into place.
5. Make your API return sensible responses for anyone using from any location in the world. Explain what factors you took into consideration and what improvements you put into place.
6. Make a single reasonably-sized instance of your API able to serve millions of requests per day. Explain what factors you took into consideration and what improvements you put into place.
7. Protect your API against malicious users. Explain what factors you took into consideration and what mitigations you put into place.
8. Your API gets really popular and people request that you add the DC metro to it, which you decide to do. Add them using this dataset:
    a. https://opendata.dc.gov/datasets/metro-stations-regional/explore?location=38.934919%2C-77.042966%2C11.58#:~:text=Generate%20new%20download-,with,-latest%20data
    b. Your API must remain backwards-compatible.
9. Bonus: Design a deployment for your API in the orchestration system of your choice, taking into account all relevant production deployment best practices.
    a. Extra Bonus: Provide a working deployment configuration.