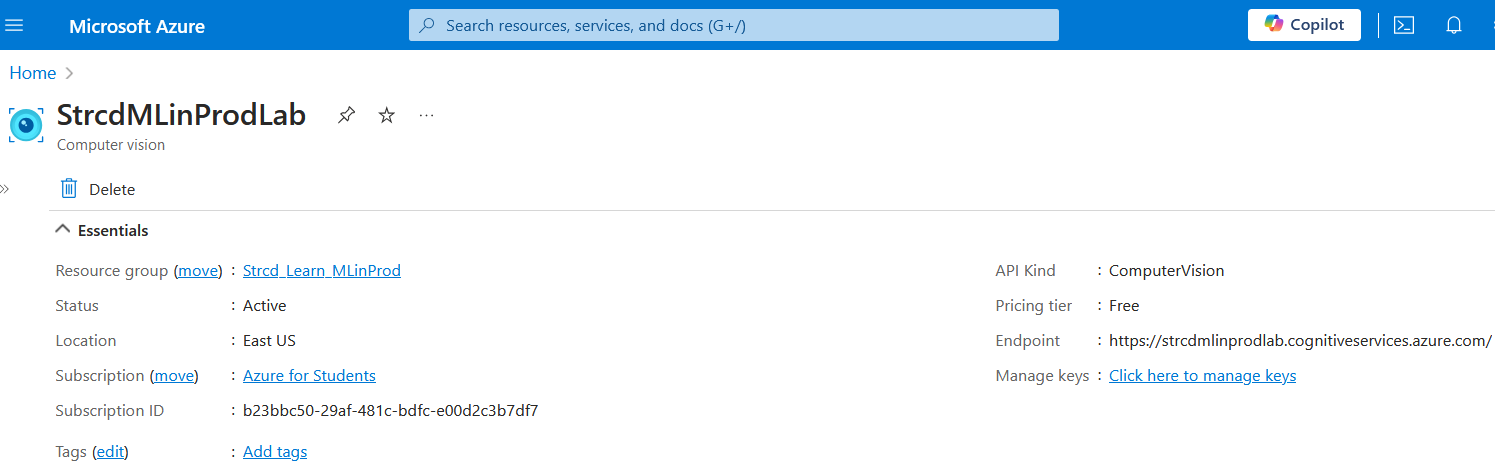
**Lab 1: Calling, Building, and Securing APIs**

**Ahmad Furqan (AndrewID: afurqan)**

1. **Create an account and connect to the Azure Vision API.**

****

1. **Explain to the TA why hard-coding credentials is a bad idea. Commit your code to GitHub without committing your credentials.**
2. **Security Risks:** exposes credentials to anyone with access to the repository, leading to potential unauthorized use of sensitive services. Attackers actively scan public repositories for exposed credentials, which they can exploit for malicious activities, increasing security risks.
3. **Inflexibility:** difficult to change, requiring source code modifications and redeployment, which increases the risk of errors. They also hinder environment-specific configurations, making it challenging to manage different credentials for development, testing, and production environments.
4. **Lack of Scalability:** becomes unmanageable as applications grow, leading to inconsistent and hard-to-maintain code across multiple services. In collaborative settings, this practice risks exposing sensitive information when team members accidentally check in code containing hard-coded credentials.

I put my credential to config.json file and load them into analyze.py

Config.json

{

    "api\_key": "my\_api\_key",

    "endpoint": "my\_endpoint”

}

Analyze.py

import json

def load\_config():

    with open('config.json') as f:

        return json.load(f)

config = load\_config()

endpoint = config.get('endpoint')

key = config.get('api\_key')

credentials = CognitiveServicesCredentials(key)

client = ComputerVisionClient(

    endpoint=endpoint,

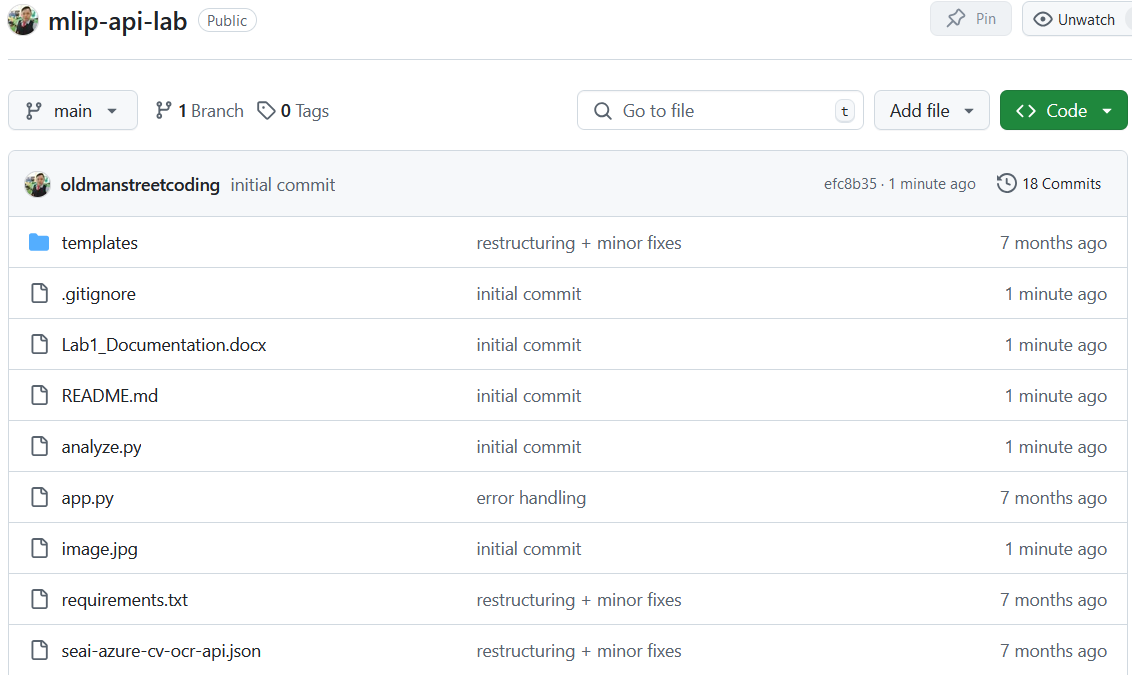
    credentials=credentials

)

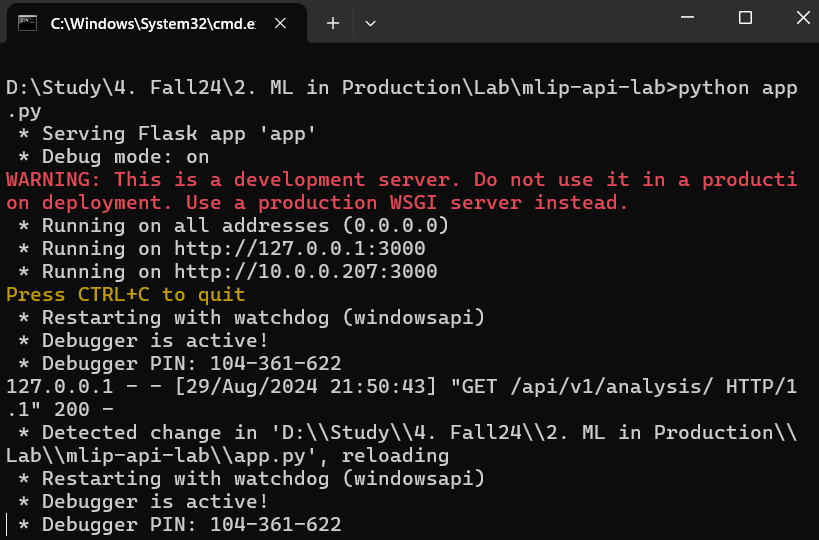
.gitignore

# exclude config file

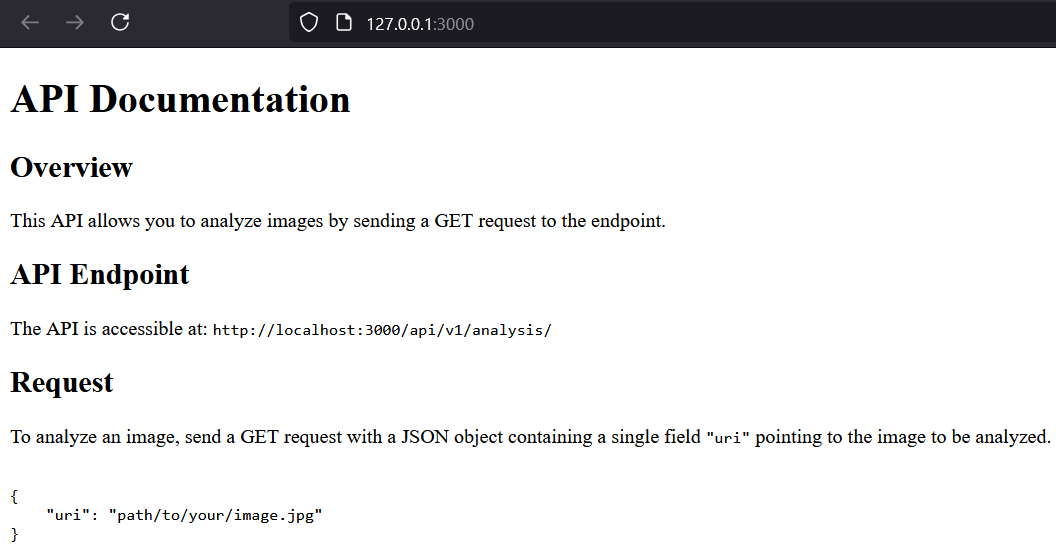
config.json



1. **Run the API endpoint with the starter code and demonstrate that it works with an example invocation (e.g., using curl)**



It will automatically open our default browser and show API documentation



Invocation using Postman

