उत्तम! अभिनंदन! 🎉

हा रिस्पॉन्स यशस्वीरित्या आल्याचा अर्थ असा आहे की तुमचा संपूर्ण प्रोजेक्ट (मॉडेल ट्रेनिंग पासून API टेस्टिंग पर्यंत) आता व्यवस्थित काम करत आहे.

तुम्ही यशस्वीरित्या खालील गोष्टी पूर्ण केल्या आहेत:

* मॉडेल तयार करून ट्रेन करणे.
* मॉडेलला .joblib फाईलमध्ये सेव्ह करणे.
* Flask वापरून एक वेब API तयार करणे.
* API ला स्थानिक पातळीवर (locally) चालवणे.
* Postman वापरून API ला यशस्वीरित्या टेस्ट करणे.

**## या रिस्पॉन्सचा अर्थ**

* "prediction": "Poisonous": तुमच्या मॉडेलने दिलेल्या इनपुटसाठी **'Poisonous' (विषारी)** अशी भविष्यवाणी केली आहे.
* "probability": तुमच्या मॉडेलला **९८.४३%** खात्री आहे की ते विषारी आहे आणि फक्त **१.५७%** शक्यता आहे की ते खाण्यायोग्य आहे.

**## आता पुढे काय? (Next Steps)**

तुमचा सर्वात महत्त्वाचा आणि तांत्रिक भाग यशस्वीरित्या पूर्ण झाला आहे. आता तुम्हाला तुमच्या प्रोजेक्ट डॉक्युमेंटनुसार 1111111111111111111111111111111 खालील गोष्टींवर लक्ष केंद्रित करायचे आहे:

1. **क्लाउड डिप्लॉयमेंट (Cloud Deployment):** तुमचा हा API आता AWS, GCP, किंवा Azure सारख्या क्लाउड प्लॅटफॉर्मवर होस्ट करा. 2
2. **डॉक्युमेंटेशन (Documentation):** HLD, LLD, आणि आर्किटेक्चर डॉक्युमेंट्स तयार करा. 3
3. **प्रोजेक्ट रिपोर्ट (Project Report):** एक सविस्तर प्रोजेक्ट रिपोर्ट (DPR) तयार करा. 4
4. **डेमो व्हिडिओ (Demo Video):** तुमच्या प्रोजेक्टचा कमीत कमी ५ मिनिटांचा एक डेमो व्हिडिओ रेकॉर्ड करा. 5
5. **लिंक्डइन पोस्ट (LinkedIn Post):** तुमच्या प्रोजेक्टबद्दल एक पोस्ट लिंक्डइनवर शेअर करा. 6

तुमचे काम उत्कृष्ट झाले आहे!

तुमचा Flask API क्लाउडवर डिप्लॉय करण्यासाठी अनेक पर्याय आहेत. आपण **Google Cloud Run** वापरून हे कसे करायचे ते पाहूया, कारण हा एक सोपा आणि प्रभावी मार्ग आहे. Cloud Run तुमच्या कंटेनराइज्ड ॲप्लिकेशनला थेट चालवते आणि तुम्हाला सर्व्हर व्यवस्थापित करण्याची गरज नसते. ☁️

## ## पूर्वतयारी (Prerequisites)

डिप्लॉयमेंट सुरू करण्यापूर्वी, तुमच्याकडे खालील गोष्टी तयार असल्याची खात्री करा:

1. **Google Cloud Platform (GCP) खाते:** तुमच्याकडे एक सक्रिय GCP खाते आणि एक प्रोजेक्ट तयार असावा.
2. **Google Cloud SDK (gcloud):** तुमच्या कॉम्प्युटरवर gcloud कमांड-लाइन टूल इन्स्टॉल आणि कॉन्फिगर केलेले असावे.
3. **Docker:** तुमच्या कॉम्प्युटरवर Docker Desktop इन्स्टॉल केलेले असावे.
4. **तुमचे प्रोजेक्ट फाइल्स:** एका फोल्डरमध्ये खालील फाइल्स तयार असाव्यात:
   * app.py (तुमचा Flask API कोड)
   * mushroom\_model.joblib
   * model\_columns.joblib

## ## क्लाउड डिप्लॉयमेंटच्या पायऱ्या (Steps for Cloud Deployment)

### ### पायरी १: requirements.txt फाईल तयार करा

तुमच्या प्रोजेक्ट फोल्डरमध्ये requirements.txt नावाची एक नवीन फाईल तयार करा. ही फाईल तुमच्या ॲप्लिकेशनला लागणाऱ्या सर्व Python लायब्ररींची यादी करते. **Gunicorn** हा एक महत्त्वाचा प्रॉडक्शन वेब सर्वर आहे जो Flask च्या अंगभूत सर्वरपेक्षा अधिक कार्यक्षम आहे.

Plaintext

# requirements.txt

Flask

pandas

scikit-learn

joblib

gunicorn

### ### पायरी २: Dockerfile तयार करा 🐳

तुमच्या प्रोजेक्ट फोल्डरमध्ये Dockerfile (कोणतेही एक्सटेन्शन नाही) नावाची एक नवीन फाईल तयार करा. ही फाईल तुमच्या ॲप्लिकेशनचा एक कंटेनर (एक स्वतंत्र पॅकेज) कसा तयार करायचा याच्या सूचना देते.

Dockerfile

# Dockerfile

# Python 3.9 ला बेस इमेज म्हणून वापरा

FROM python:3.9-slim

# वर्किंग डिरेक्टरी सेट करा

WORKDIR /app

# आवश्यक फाइल्स कंटेनरमध्ये कॉपी करा

COPY requirements.txt .

COPY app.py .

COPY mushroom\_model.joblib .

COPY model\_columns.joblib .

# requirements.txt मधील लायब्ररी इन्स्टॉल करा

RUN pip install --no-cache-dir -r requirements.txt

# Gunicorn वापरून ॲप चालवण्यासाठी कमांड

# हे 8080 पोर्टवर चालेल, जे Cloud Run साठी आवश्यक आहे

CMD ["gunicorn", "-b", "0.0.0.0:8080", "app:app"]

### ### पायरी ३: डॉकर इमेज तयार करा (Build the Docker Image)

तुमच्या टर्मिनलमध्ये प्रोजेक्ट फोल्डरवर नेव्हिगेट करा आणि खालील कमांड चालवून तुमच्या ॲप्लिकेशनची डॉकर इमेज तयार करा. mushroom-api हे तुमच्या इमेजचे नाव आहे.

Bash

docker build -t mushroom-api .

### ### पायरी ४: GCP APIs सक्षम करा

तुमच्या GCP प्रोजेक्टमध्ये Artifact Registry आणि Cloud Run APIs सक्षम करण्यासाठी खालील कमांड्स चालवा:

Bash

gcloud services enable artifactregistry.googleapis.com

gcloud services enable run.googleapis.com

### ### पायरी ५: डॉकरला GCP शी ऑथेंटिकेट करा

तुमची डॉकर इमेज GCP वर अपलोड करण्यासाठी डॉकरला परवानगी द्या:

Bash

gcloud auth configure-docker <REGION>-docker.pkg.dev

**टीप:** <REGION> च्या जागी तुमच्या जवळची GCP रिजन टाका (उदा. asia-south1).

### ### पायरी ६: Artifact Registry मध्ये एक रिपॉझिटरी तयार करा

तुमची डॉकर इमेज साठवण्यासाठी GCP वर एक जागा (रिपॉझिटरी) तयार करा.

Bash

gcloud artifacts repositories create my-api-repo --repository-format=docker --location=<REGION> --description="Mushroom API repository"

### ### पायरी ७: डॉकर इमेजला टॅग करा

तुमच्या स्थानिक डॉकर इमेजला GCP रिपॉझिटरीमध्ये अपलोड करण्यासाठी योग्य नावाने टॅग करा.

Bash

docker tag mushroom-api <REGION>-docker.pkg.dev/<PROJECT\_ID>/my-api-repo/mushroom-api

**टीप:** <PROJECT\_ID> च्या जागी तुमचा GCP Project ID टाका.

### ### पायरी ८: डॉकर इमेज GCP वर अपलोड करा (Push)

टॅग केलेली इमेज Artifact Registry वर अपलोड करा.

Bash

docker push <REGION>-docker.pkg.dev/<PROJECT\_ID>/my-api-repo/mushroom-api

### ### पायरी ९: Cloud Run वर डिप्लॉय करा 🚀

आता तुमची इमेज GCP वर आहे. खालील कमांड वापरून ती Cloud Run वर डिप्लॉय करा. --allow-unauthenticated हे तुमच्या API ला सार्वजनिक (public) बनवते.

Bash

gcloud run deploy mushroom-api-service \

--image=<REGION>-docker.pkg.dev/<PROJECT\_ID>/my-api-repo/mushroom-api \

--platform=managed \

--region=<REGION> \

--allow-unauthenticated

ही कमांड चालवल्यानंतर, GCP तुमच्यासाठी एक सार्वजनिक URL तयार करेल.

### ### पायरी १०: लाईव्ह API टेस्ट करा

डिप्लॉयमेंट पूर्ण झाल्यावर, तुम्हाला टर्मिनलमध्ये एक URL मिळेल (उदा. https://mushroom-api-service-xxxxx-an.a.run.app). ही URL कॉपी करा आणि Postman मध्ये वापरून तुमच्या लाईव्ह API ला टेस्ट करा!

अभिनंदन! तुमचा API आता क्लाउडवर यशस्वीरित्या डिप्लॉय झाला आहे.