Github url: <https://github.com/MSK-hash/Final.git>

Saikiran\_Mamidala-500209412

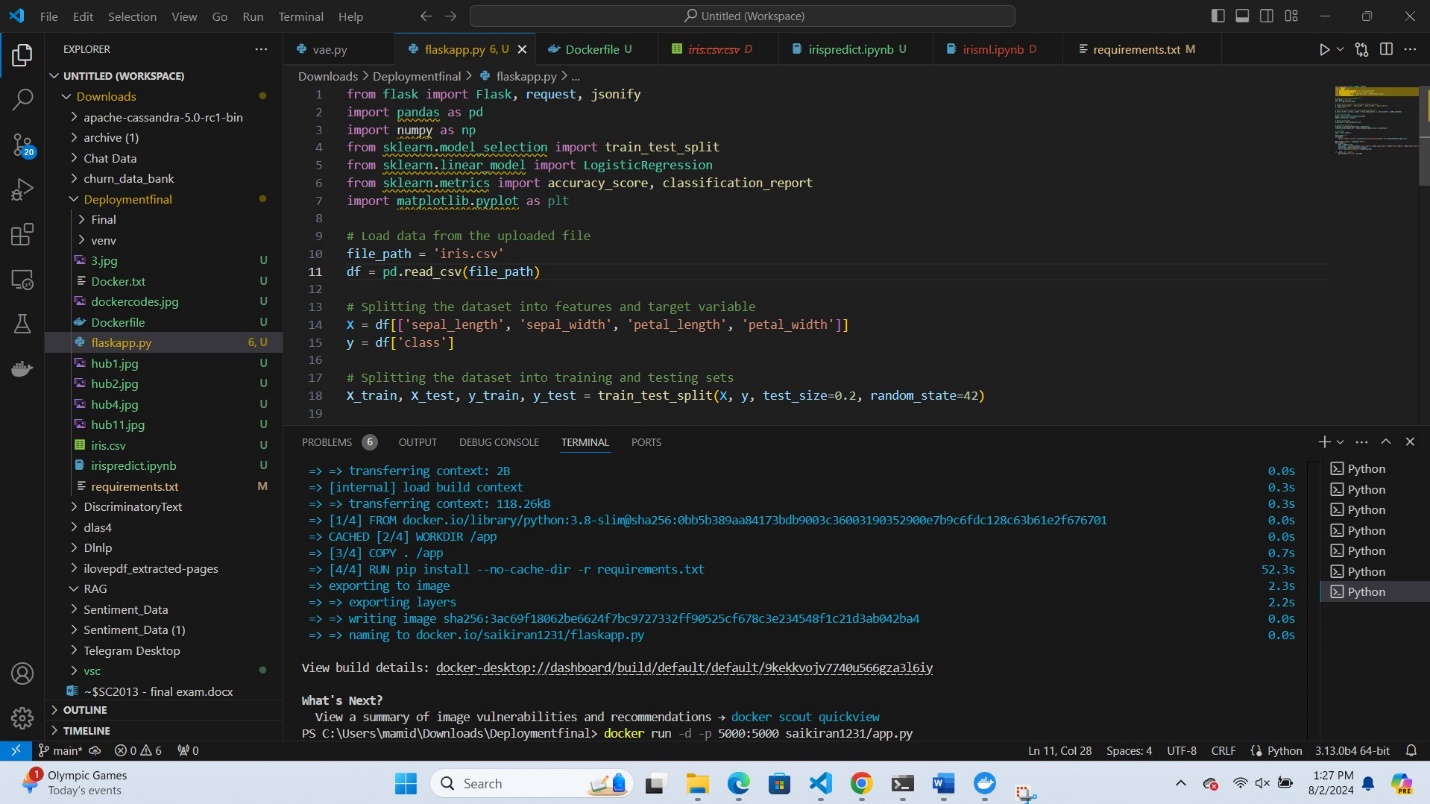
Provide the Docker commands used along with screenshots:

docker build -t saikiran1231/app.py .

docker run -d -p 5000:5000 saikiran1231/app.py

A screenshot of a computer screen

Description automatically generated



This Python script is a Flask web application that loads a machine learning model to classify iris flowers based on their features. The application provides two main functionalities:

1. Displaying the model's accuracy and classification report.
2. Predicting the class of an iris flower based on user-provided features.

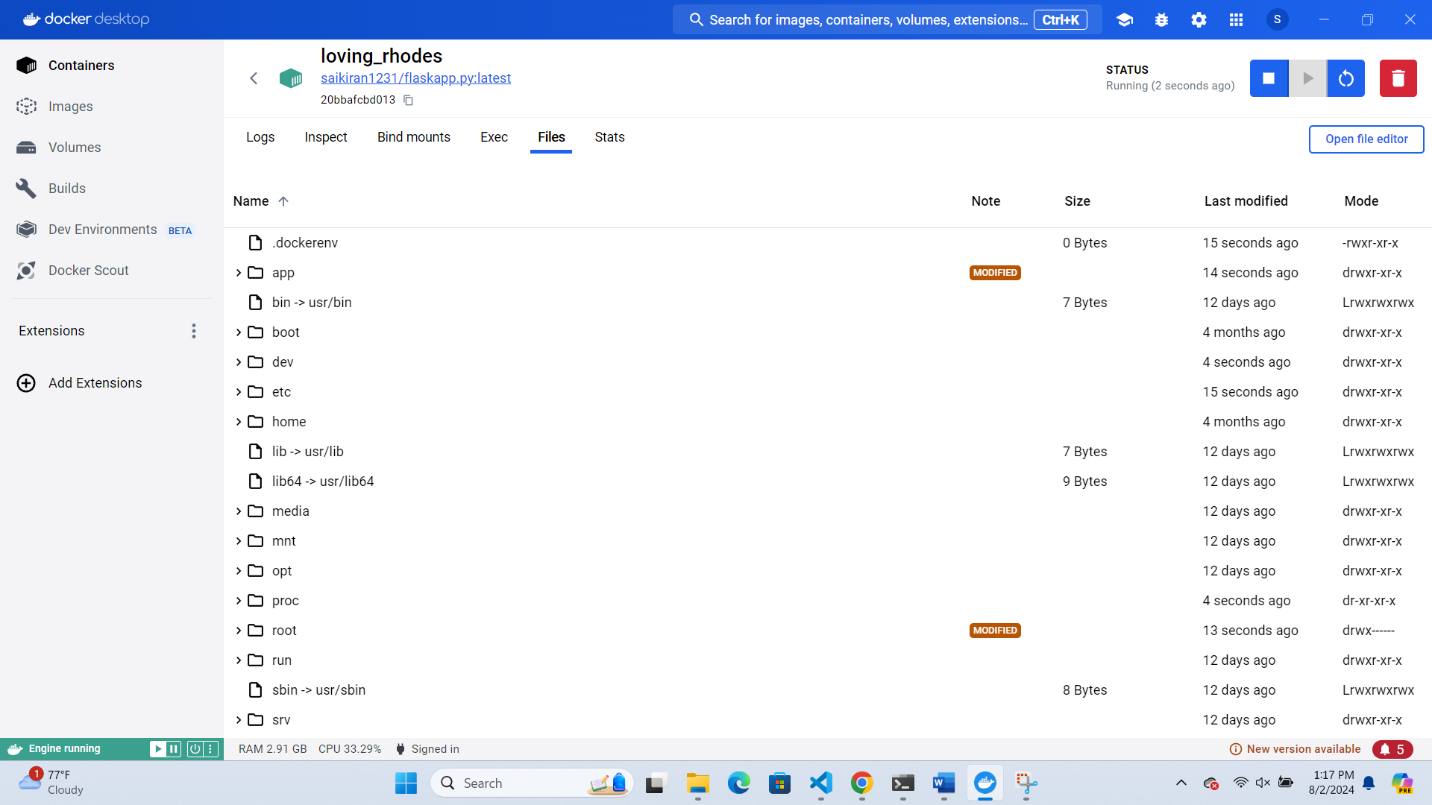
**Detailed Explanation**

1. **Import Libraries**:
   * The script imports necessary libraries for data manipulation (pandas), numerical operations (numpy), machine learning (sklearn), visualization (matplotlib), and web development (flask).
2. **Load Dataset**:
   * The Iris dataset is loaded from a file named iris.csv using pandas.
3. **Data Preparation**:
   * The dataset is split into features (X) and the target variable (y).
   * The features include sepal\_length, sepal\_width, petal\_length, and petal\_width.
   * The target variable is the class of the iris flower.
4. **Split Data**:
   * The dataset is divided into training and testing sets using train\_test\_split from sklearn. This ensures that the model can be trained and evaluated on separate data.
5. **Train Model**:
   * A Logistic Regression model is instantiated and trained on the training data (X\_train, y\_train).
6. **Make Predictions**:
   * The trained model is used to make predictions on the test data (X\_test).
7. **Evaluate Model**:
   * The accuracy of the model is calculated using accuracy\_score.
   * A detailed classification report is generated using classification\_report.
8. **Flask Web Application**:
   * The Flask app is initialized and configured to run on host 0.0.0.0 and port 80.
9. **Home Route (/)**:
   * The home route (/) returns the model's accuracy and classification report in HTML format.
10. **Prediction Route (/predict)**:
    * The prediction route (/predict) accepts POST requests with JSON data containing the features of an iris flower.
    * It extracts the features from the JSON data, reshapes them into the required format, and uses the trained model to predict the class of the iris flower.
    * The predicted class is returned as a JSON response.

Screenshots of your application running in the container:

1. **Dockerfile**:
   * The Dockerfile defines the instructions to build the Docker image, including the base image (Python 3.9-slim), setting the working directory, copying the application files, installing dependencies, and exposing the necessary port.
2. **Building the Docker Image**:
   * The Docker image is built using the command docker build -t iris-classifier ., which packages the application and its dependencies into an image.
3. **Running the Docker Container**:
   * The application is run inside a Docker container using the command docker run -p 4000:80 iris-classifier. This maps port 80 inside the container to port 4000 on the host machine, allowing access to the application via http://localhost:4000.
4. **Accessing the Application**:
   * Once the container is running, users can access the application in their web browser. The home page displays the model's accuracy and classification report. Additionally, the /predict endpoint can be used to make predictions by sending POST requests with iris flower features.

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

Docker in Hub:

**Pull the Docker Image from Docker Hub**

First, ensure that you have Docker installed on your machine. Then, pull your Docker image from Docker Hub

**Run the Docker Container**

Run a container from the pulled Docker image

By containerizing the application with Docker, we ensure a reliable and reproducible deployment process, making it easier to manage and scale the application in different environments, including local development, staging, and production.

A computer screen shot of a computer

Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generatedA computer screen with a white background

Description automatically generated

A computer screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated