Name: Chinmay Arvind
Batch Code: LISUM18
Submission Date: March 12, 2023

Week 5: Cloud Deployment
Stage 1: Model building with Python, TensorFlow, and MNIST dataset

In [2]:
```
# Source code adapted from: https://docs.ray.io/en/latest/serve/tutorials/serve-ml-models.html
# Importing necessary libraries
import pickle
import tensorflow as tf
```

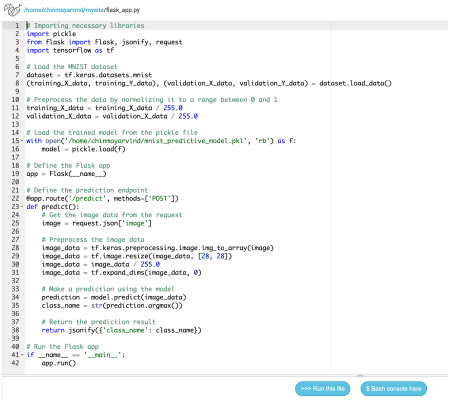
In [3]:
```
# Load MNIST dataset
dataset = tf.keras.datasets.mnist
(training_X_data, training_Y_data), (Validation_X_data, Validation_Y_data) = dataset.load_data()

# Preprocessing the data by normalizing it to a range between 0 and 1
training_X_data = training_X_data / 255.0
Validation_X_data = Validation_X_data / 255.0

# Defining the Neural Network's architecture
neural_net_model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(5, activation='softmax')
])

# Compiling the model with the poisson loss and training the model
neural_net_model.compile(optimizer='sgd', loss='poisson', metrics=['accuracy'])
neural_net_model.fit(training_X_data, training_Y_data, epochs=20)

# Running an evaluation and checking how well the model performs on the validation data
neural_net_model.evaluate(Validation_X_data, Validation_Y_data)

# Saving the model into a pickle file by writing to it
with open('mnist_predictive_model.pkl', 'wb') as filename:
    pickle.dump(neural_net_model, filename)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 1s 0us/step
Epoch 1/20
```

```
2023-03-12 15:12:31.292994: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operat
ions:  SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
```

```
1875/1875 [==============================] - 3s 2ms/step - loss: 7.3794 - accuracy: 0.0879
Epoch 2/20
1875/1875 [==============================] - 3s 2ms/step - loss: 7.3725 - accuracy: 0.0951
Epoch 3/20
```

Stage 2: Deployment of model to Cloud
Step 1: Create pythonanywhere account
Step 2: Create a web app with flask and upload flask_app.py code to it
Step 3: Finally, open webpage where the model has been deployed


/home/chinmayarvind/mysite/flask_app.py

```
# Importing necessary libraries
import pickle
from flask import Flask, jsonify, request
import tensorflow as tf

# Load the MNIST dataset
dataset = tf.keras.datasets.mnist
(training_X_data, training_Y_data), (validation_X_data, validation_Y_data) = dataset.load_data()

# Preprocess the data by normalizing it to a range between 0 and 1
training_X_data = training_X_data / 255.0
validation_X_data = validation_X_data / 255.0

# Load the trained model from the pickle file
with open('/home/chinmayarvind/mnist_predictive_model.pkl', 'rb') as f:
    model = pickle.load(f)

# Define the Flask app
app = Flask(__name__)

# Define the prediction endpoint
@app.route('/predict', methods=['POST'])
def predict():
    # Get the image data from the request
    image = request.json['image']

    # Preprocess the image data
    image_data = tf.keras.preprocessing.image.img_to_array(image)
    image_data = tf.image.resize(image_data, [28, 28])
    image_data = image_data / 255.0
    image_data = tf.expand_dims(image_data, 0)

    # Make a prediction using the model
    prediction = model.predict(image_data)
    class_name = str(prediction.argmax())

    # Return the prediction result
    return jsonify({'class_name': class_name})

# Run the Flask app
if __name__ == '__main__':
    app.run()
```

>>> Run this file $ Bash console here