# Emotion Detection Project Documentation

## Overview

This project involves building an emotion detection web service. The process includes training a machine learning model using the dair-ai/emotion dataset, deploying the model as an inference endpoint using AWS SageMaker, and setting up a web server on an EC2 instance to interact with the SageMaker endpoint.

## Team Members:

• Hassan Mohamed hassan Ali

• Mohamed mostafa elmorsy

•Saif Ashraf

• Mohamed Aziz

## Development Process

Here's a breakdown of the steps taken to set up and deploy the project:

## 1. Data Preparation and Model Training

- **Dataset**: The dair-ai/emotion dataset from Hugging Face was used, containing labeled text data for various emotions.
- **Model**: The architecture chosen was DistilBERT, a lightweight version of BERT, suitable for text classification tasks.
- **Training Platform**: AWS SageMaker was used for training the model.
    - **Steps**:
        - **Dataset Preparation**: The dataset was uploaded to an S3 bucket.
        - **Training Job Configuration**: A training job was configured in SageMaker with the following settings:
            - **Algorithm**: Hugging Face's Transformers framework with DistilBERT.
            - **Hyperparameters**: Tuned for learning rate, batch size, and the number of epochs.
            - **Instance Type**: ml.p3.2xlarge for fast training using GPU acceleration.
        - **Fine-Tuning the Model**: The model was fine-tuned on the emotion dataset, achieving a high accuracy on the validation set.

## 2. Setting Up the SageMaker Inference Endpoint

- After training, the model was deployed as an endpoint on AWS SageMaker.
    - **Steps**:
        - **Create Endpoint Configuration**: The trained model artifacts were used to create an endpoint configuration.
        - **Deploy the Endpoint**: The model was deployed to an ml.m5.large instance for inference.

        - **Testing the Endpoint**: The endpoint was tested using sample inputs to ensure it returned the expected emotion predictions.

## 3. Developing the Web Server

- **Platform**: A Flask web server was developed to serve as a front-end interface for users to interact with the SageMaker endpoint.
- **Configuration**:
    - The server was configured to receive text input via a POST request, forward the request to the SageMaker endpoint using the boto3 library, and return the predicted emotion and confidence score.
- **Error Handling**:
    - Implemented error handling to manage cases where the SageMaker endpoint is unreachable or the input is invalid.

## 4. Hosting the Web Server on an EC2 Instance

- **EC2 Instance Setup**:
    - Launched an EC2 instance using an Amazon Linux 2 AMI.
    - Configured the security group to allow inbound HTTP and SSH traffic.
- **Environment Setup**:
    - Installed Python, Git, and other required dependencies on the instance.
    - Cloned the project repository from GitHub.
    - Installed the necessary Python packages from requirements.txt.
- **Running the Flask Server**:
    - Configured the Flask application to listen on all network interfaces (0.0.0.0) to make it accessible externally.
    - Used screen or tmux to keep the server running in the background.

- **Testing**:
    - The server was tested using tools like curl and Postman to ensure that it correctly communicated with the SageMaker endpoint and returned the predicted emotion.