Depression Indicator App - Project Documentation

# 1. Problem Theme

Mental health issues, particularly depression, are one of the most pressing global health challenges. Early detection through survey-based data can play a crucial role in intervention and treatment.

# 2. Problem Statement

The aim is to develop an AI-based solution that predicts whether an individual is likely to be experiencing symptoms of depression based on survey responses.

# 3. Aim of the Project

Our goal is to build a deep learning model using PyTorch and provide an easy-to-use interface through Streamlit to predict depression indicators.

# 4. About the Dataset

The dataset consists of survey responses from individuals, containing both categorical and numerical features related to lifestyle, social interactions, and psychological well-being. Data required cleaning to remove invalid categories and encoding categorical values.

# 5. Our Approach

1. Data Cleaning & Preprocessing  
2. Exploratory Data Analysis (EDA)  
3. Encoding categorical features  
4. Converting inputs into tensors  
5. Using PyTorch DataLoader for efficient batch processing  
6. Building and training the model  
7. Deploying via Streamlit Cloud.

# 6. Deep Learning Framework

We used PyTorch as the core deep learning framework for model implementation, training, and evaluation.

# 7. Model Architecture

The model consists of a fully connected feedforward neural network with multiple hidden layers. Activation functions like ReLU were used, along with dropout layers for regularization.

# 8. Streamlit UI

We built a user-friendly UI using Streamlit, where users can input survey data and get a real-time prediction. The app is deployed on Streamlit Cloud for accessibility.  
  
**App Link**: https://ishaaq09-depression-indicator-app-app-qngzxe.streamlit.app/

# 9. Problems Faced During Development

- Data Cleaning: Encountered invalid categories in categorical features. Initially encoded directly, but had to restart and clean data properly.  
- Tensor Conversion: Faced issues converting input data into tensors during preprocessing.  
- Dataloader Confusion: Initially had no idea how to use PyTorch Dataloaders, but learned on the go.  
- Model Deployment: Streamlit required compatible library versions, which caused some dependency conflicts.  
- Debugging Model Training: Faced NaN values during training due to improper normalization of inputs.

# 10. Lessons Learned

- Importance of clean data preprocessing before model training.  
- How to properly use PyTorch tensors and Dataloaders.  
- Deployment considerations when working with cloud platforms like Streamlit.  
- Iterative debugging and testing improves project robustness.

# 11. Future Improvements

- Collect more diverse datasets to improve generalization.  
- Experiment with advanced architectures like LSTMs or Transformers for sequence data.  
- Integrate explainable AI (XAI) methods to interpret model predictions.  
- Improve UI/UX of the Streamlit app with better visuals and insights.

# 12. References

- PyTorch Documentation  
- Streamlit Documentation  
- Kaggle community discussions on survey-based datasets  
- Research papers on mental health prediction using ML