📌 Project: AI Music Generator with LSTM

1️⃣ Objective

We use an **LSTM (Long Short-Term Memory)** neural network to generate music by training on .midi files.
The model learns the sequence of musical notes and predicts new notes to create original compositions.

2️⃣ Required Modules

You (and your students) must install these Python modules in the virtual environment (.venv):

```
pip install streamlit
pip install tensorflow
pip install pretty_midi
pip install music21
pip install numpy
pip install matplotlib
pip install joblib
```

Module purposes:

- **streamlit** → Web interface for interaction.
- **tensorflow** → Deep learning framework to build and train the LSTM model.
- **pretty_midi** → For reading/writing MIDI files.
- **music21** → Music analysis & note processing.
- **numpy** → Numerical computations.
- **matplotlib** → Visualization of loss curves/training.

- **joblib** → Saving/loading preprocessed data or model artifacts.

3️⃣ Folder Structure

Example:

```
PythonProject/
├── app_streamlit.py      # Streamlit frontend
├── train.py              # Model training script
├── generate.py           # Music generation script
├── data/                 # MIDI training files
├── model/                # Saved trained model
├── notes.npy             # Preprocessed note sequences
└── .venv/                # Virtual environment
```

4️⃣ Steps to Run the Project

Step 1 — Prepare Data

- Collect .midi files and put them inside the data/ folder.
- Preprocess them into note sequences using pretty_midi or music21.

Step 2 — Train the Model

Run:

```
python train.py --data data/ --save model/music_model.h5 --epochs 50 --batch 64 --seq_len 50 --lr 0.001
```

Arguments:

- --data → Path to MIDI dataset folder
- --save → Path to save trained model
- --epochs → Number of training epochs
- --batch → Batch size
- --seq_len → Number of notes per training sequence
- --lr → Learning rate

After training:

- music_model.h5 is saved.
- notes.npy stores note mappings for generation.

Step 3 — Generate Music

Run:

python generate.py --model model/music_model.h5 --notes notes.npy --output output.mid

This generates a **new MIDI file** based on learned patterns.

Step 4 — Run with Streamlit UI

Run:

streamlit run app_streamlit.py

- Opens a **web interface** in the browser.

- Allows users to **train** the model or **generate music** easily without command line.
- Plays generated .midi or .wav output directly in the app.

5️⃣ Execution Flow

1. **Data Loading** → Read MIDI files.
2. **Preprocessing** → Convert to sequences.
3. **Model Building** → LSTM layers + Dense output layer.
4. **Training** → Optimize model weights.
5. **Generation** → Predict next notes.
6. **Output** → Save as .midi or .wav.
7. **User Interaction** → Done through Streamlit UI.

6️⃣ Extra Features You Can Teach

- Add **Tempo control** in generated music.
- Export directly as **.wav** file from MIDI.
- Let user **upload their own MIDI file** to train.
- Add **real-time note visualization**.