FastAPI + LangChain + Groq + LangServe Translator API

# 1. Project Overview

This application exposes an API that translates text into a target language using:  
  
- FastAPI → to create and serve the API.  
- LangChain → to build a pipeline (prompt + LLM + parser).  
- Groq → as the backend LLM provider (using Gemma-2-9b-it).  
- LangServe → to expose LangChain chains as API routes automatically.  
- dotenv → to securely load the API key.

# 2. Code Walkthrough

## Imports

FastAPI, LangChain modules, Groq integration, dotenv for environment variables, and LangServe to expose chains.

## Load API Key

Loads the Groq API key from .env file.

## Define the Model

Initializes Groq's Gemma-2-9b-it model using ChatGroq.

## Create Prompt Template

Defines the system and user messages for translation.

## Output Parser

Uses StrOutputParser to return clean string outputs.

## Build the Chain

Connects prompt → model → parser into a pipeline.

## FastAPI App Setup

Creates FastAPI app with metadata.

## Expose Chain as API

Adds LangChain runnable chain as a /chain endpoint using add\_routes.

## Run the App

Runs FastAPI with Uvicorn at localhost:8000.

# 3. How to Run

1. Install dependencies:  
 pip install fastapi uvicorn langchain langchain-groq langserve python-dotenv  
  
2. Create .env file with your Groq API key:  
 Groq\_key=your\_groq\_api\_key\_here  
  
3. Run the server:  
 python main.py  
  
Server runs at http://localhost:8000

# 4. API Usage

POST Request → http://localhost:8000/chain

## Request Body:

{  
 "input": {  
 "language": "French",  
 "text": "Hello, how are you?"  
 }  
}

## Response:

{  
 "output": "Bonjour, comment allez-vous ?"  
}

# 5. Interactive API Docs

Swagger UI: http://localhost:8000/docs  
ReDoc: http://localhost:8000/redoc

# 6. Directory Structure

project/  
│── main.py # FastAPI app  
│── .env # API key storage  
│── requirements.txt # dependencies