# **Gemma-Kavach**

## **Day 1: Setup up backend server**

I built a **production-ready backend server** using **FastAPI** to serve the **multimodal Gemma 3n model** from Google. This server supports:

* 🔤 **Text-based queries** via /generate
* 🖼️ **Image + prompt reasoning** via /ask\_image
* 🎤 **Voice/audio transcription + reasoning** via /ask

The server enables secure, real-time, and **offline-first multimodal inference**, designed to support use cases like **crisis response** and **privacy-preserving assistants**.

**🧠 Key Learnings**

1. **Stick to Official Google Docs**  
   Trying to outsmart the official guidance led to unnecessary errors. Following Google’s documentation for Gemma 3n (especially around loading AutoProcessor and AutoModelForImageTextToText) ensured compatibility and feature completeness.
2. **Correct Library Installation is Crucial**
   * Mismatched or outdated transformers, accelerate, or torch versions caused slowdowns and even runtime errors.
   * Up-to-date, version-pinned installations prevent debugging nightmares.
3. **Model Parameters Directly Impact Speed**
   * Failing to disable torch.compile, or setting the wrong attn\_implementation, can **drastically slow down token generation**.
   * Using torch\_dtype=torch.bfloat16, disable\_compile=True, and attn\_implementation="eager" gives you a huge speed boost for inference.
4. **Every Developer Should Know How to Serve Models Locally**  
   Hosting your own inference backend is critical for:
   * 💼 **Enterprise deployments** with data security needs
   * 🔐 **Offline/private use** (like in crisis zones or field missions)
   * 🛠️ Learning how open-source LLMs actually integrate with production apps

## **Day 1: Feature Planning: Gemma Kavach Vision**

Simple Objective: Let people upload a video and get analysis and otherwise I real time stream.