# Project Report: Deploying BrainRot Chatbot with LLaMA 2-13B

## 1. Environments

- Python / Conda (“mop-ai” env):  
 - Contains LLaMA-2 model code, FastAPI, PyTorch, Transformers, bitsandbytes.  
 - GPU-enabled for 4-bit quantized inference.  
- Node.js / Conda (“mychatbot-nodejs” env):  
 - Contains React + Vite front‑end and Express static-server.  
 - Uses Axios or fetch to communicate with FastAPI.

## 2. Python API Service

1. Dependencies:  
 ```bash  
 conda activate mop-ai  
 pip install fastapi uvicorn pydantic transformers torch bitsandbytes accelerate  
 ```  
2. api.py:  
 - Imports load\_model\_and\_tokenizer() & generate\_response() from chatbot.py.  
 - Defines Pydantic Query schema (text: str).  
 - Wraps user message in System prompt template for MindEase persona.  
 - Exposes POST /api/chat returning {"response": …}.  
3. Run:  
 ```bash  
 uvicorn api:app --host 0.0.0.0 --port 8000  
 ```

## 3. React + Vite Front-End

1. Scaffold:  
 ```bash  
 cd react-chat-bot/mychatbot  
 npm install  
 ```  
2. Proxy configuration in vite.config.ts to forward /api → FastAPI.  
3. Chat UI (src/App.tsx):  
 - useState for history, sendMessage() posts to /api/chat, updates history.  
 - Auto-scroll and Enter-to-send.  
4. Styling (src/App.css): message bubbles and layout.  
5. Dev & Build:  
 ```bash  
 npm run dev # http://localhost:5173  
 npm run build  
 node server.js # Serve build on port 3000  
 ```

## 4. Prompt Template Integration

- System prompt embedded via wrap\_prompt(), so model sees MindEase guidelines first.  
- Ensures empathetic, safe, concise responses tailored to mental health support.  
- Uses Alpaca-style [INST]/[SYS] tokens matching fine-tuned LLaMA-2 format.

## 5. Next Steps & Enhancements

- Persist chat history to localStorage or a database.  
- Add WebSocket support for token streaming.  
- Introduce authentication for private chat logs.  
- Deploy FastAPI on GPU-enabled cloud instance; host React on Netlify/Vercel.

## 6. Conclusion

A clean, two‑service architecture:  
1. FastAPI + PyTorch for GPU‑backed LLM inference with LoRA‑tuned LLaMA‑2.  
2. React + Vite + Express for a modern chat interface with session history.  
This separation ensures scalability, maintainability, and ease of future enhancements.