Technical Plan — Glass-Box Conversational NLP System

Project Goal:
Develop a conversational NLP model with the long-term objective of achieving a fully interpretable glass-box architecture. Initial phases focus on fine-tuning a transformer while tracing neuron and attention-level computations, ultimately enabling deterministic rule extraction and a meta-layer for transparent inference.

1. Model Selection & Training

- **Architecture:** Transformer-based conversational model (encoder-decoder or decoder-only)
- **Pre-trained Base:** Medium-scale model (e.g., GPT-2 medium) for tractability
- **Fine-Tuning Dataset:** Domain-specific conversational corpus
- **Training Strategy:**
    - Modular design: embeddings, attention layers, feed-forward layers, and outputs separated
    - Save intermediate activations for later tracing and analysis

2. Phase 1 — CPU & OS-Level Prototyping

1. **Model Execution:** Fine-tune and run the transformer entirely on CPU
2. **Profiling & Tracing:**
    - Capture neuron activations, attention weights, residuals per layer
    - Record memory usage and execution flow at OS-level
    - Use tools such as `cProfile`, `perf`, or PyTorch hooks
3. **Initial Rule Extraction:**
    - Identify consistent patterns in activations and attention heads
    - Aggregate recurring patterns into IF-THEN style rules
4. **Validation:**
    - Confirm traced outputs reproduce model predictions

  - Test sensitivity to small input perturbations

Output: CPU-level dataset of activation traces, attention patterns, and preliminary internal rules.

3. Phase 2 — CUDA/GPU Profiling & Scaling

1. **Port Pipeline:** Move activation/attention logging to GPU using CUDA/LibTorch
2. **Hardware-Level Analysis:**
   - Profile floating-point operations, memory transfers, and per-layer computations
   - Identify high-impact neurons and attention pathways
3. **Scale Model:** Fine-tune larger transformers efficiently while applying tracing methodology

Output: High-resolution GPU-level traces for accurate, reproducible rule extraction.

4. Deterministic Rule Extraction

- Translate consistent neuron and attention patterns into **interpretable rules**:
  - Map high-impact activations to output contributions
  - Aggregate rules across layers for deterministic, human-readable explanations
- Goal: approximate or replicate transformer outputs using a **rule-guided mechanism**

5. Integration with Glass-Box Principles

- Design a **meta-layer** or module inspired by ChurnBot's Meta-EBM:
  - Weight sub-module outputs deterministically based on extracted rules
  - Enable partial-to-full explainability for predictions
- Long-term target: a hybrid transformer-glass-box system where every output can be **traced through neuron/attention contributions and deterministic rules**

6. Evaluation & Iteration

- **Functional Testing:** Verify conversational quality and coherence
- **Interpretability Metrics:**
  - % of outputs explained by rules
  - Consistency of neuron/attention contributions across similar inputs
- **Iteration:** Refine rule extraction, layer weighting, and module sensitivity

7. Documentation & Artifacts

- Maintain an organized branch in the project GitHub with:
  - Model architecture diagrams
  - CPU/GPU activation and attention traces
  - Extracted internal rules
  - Profiling reports and experimental logs
  - Fine-tuning hyperparameters and sensitivity analyses

Key Insight / Glass-Box Roadmap

- **Phase 1 (CPU):** Build a functional transformer, trace computations, and extract preliminary rules
- **Phase 2 (GPU/CUDA):** Scale tracing and profiling to full-size models, capturing high-resolution activation patterns
- **Phase 3 (Meta-Layer Integration):** Apply deterministic rule-guided weighting to outputs for partial explainability
- **Phase 4 (Full Glass-Box NLP):** Combine transformer outputs with internal rules to achieve fully interpretable, reproducible, and explainable predictions

💡 **Takeaway:** Starting with CPU-level tracing creates a foundation for interpretable analysis. Over time, extracted rules and a meta-layer can convert the black-box transformer into a **true glass-box NLP system**