# Final Project Proposal: Context-Adaptive Personal AI Clone

## 1. Executive Summary

This project aims to build a "Personal Digital Twin"—a sophisticated WhatsApp automation system capable of replying to messages on your behalf. Unlike standard chatbots, this system will mimic your distinct linguistic style, adapt its tone based on the relationship with the contact (e.g., professional vs. casual), and maintain a "long-term memory" of people and events using a Knowledge Graph.

The system prioritizes **user control** via a "Human-in-the-Loop" override mechanism and **operational security** through advanced human behavior simulation to mitigate ban risks associated with unofficial WhatsApp APIs.

## 2. Solution Architecture

The system utilizes a **Microservices Architecture** to decouple the fragile WhatsApp connectivity layer from the robust AI reasoning engine.

### 2.1 High-Level Stack

* **Interface (Frontend):** **Next.js 16** (React) – A web dashboard for configuration, QR authentication, and real-time monitoring.
* **Connectivity (Gateway):** **Node.js** + **whatsapp-web.js** – Emulates a browser session to connect to your personal WhatsApp account.
* **Orchestration (Brain):** **LangGraph.js** – Manages the decision loops, state persistence, and agentic workflows.
* **Intelligence (Model):** **Claude Sonnet 4.5** – Selected for its superior reasoning and style mimicry capabilities.
* **Memory (Database):** **FalkorDB** – A high-performance, low-latency Graph Database for storing relationship context.
* **State & Queues:** **Redis** – Handles message queues and "Paused/Active" states.

## 3. Core Component Specifications

### 3.1 The Intelligence Layer: Hybrid Model Strategy

Based on the comparative analysis, we will employ a **Tiered Architecture** to balance cost, speed, and quality.

* **Tier 1: The Router (Claude Haiku 4.5)**
  + **Role:** Traffic Controller.
  + **Task:** Analyzes incoming messages to classify them as "Phatic" (simple greetings like "lol", "ok") or "Substantive" (requiring information/memory).
  + **Benefit:** Handles ~40% of traffic instantly at extremely low cost ($1.00/1M tokens).
* **Tier 2: The Reasoner (Claude Sonnet 4.5)**
  + **Role:** The Ghostwriter.
  + **Task:** Handles substantive queries. It retrieves context from the Graph, selects the correct persona style, and generates the reply.
  + **Optimization:** Utilizes **Prompt Caching** to store your heavy "Persona Style Guide" and "Graph Schema," reducing latency and cost by ~90% for repeated requests.

### 3.2 The Memory Layer: FalkorDB (GraphRAG)

We selected **FalkorDB** over Neo4j for its superior latency in agentic workflows and lower resource overhead.

* **Schema Design:**
  + **Nodes:** User (You), Contact, Entity (Person, Company, Event), Topic.
  + **Edges:** (:User)-->(:Contact), (:Contact)-->(:Topic).
* **GraphRAG Workflow:**
  1. **Extraction:** Background agents process chat history to build the graph (e.g., "John mentions he works at Google" -> Creates (John)-->(Google)).
  2. **Retrieval:** When John asks "How's work?", the bot queries the graph to see *your* work context and *John's* relationship to it, avoiding generic responses.

### 3.3 The Connectivity Layer: Operational Security

To use your personal number safely, we must rigorously simulate human behavior.

* **Human Typing Simulation (HTS):**
  + The bot will **not** reply instantly.
  + **Algorithm:** Delay = (Message\_Length \* 300ms) + Cognitive\_Pause (2s-5s).
  + **Visual Cue:** Triggers the "Typing..." status on WhatsApp during this calculated delay.
* **Rate Limiting Circuit Breaker:**
  + If message volume exceeds human norms (e.g., >10 msgs/min), the bot auto-pauses to prevent flagging.

### 3.4 Human-in-the-Loop (The "Override")

The system ensures you never lose control of your chats.

* **Auto-Pause Mechanism:** The Node.js client listens for the message\_create event. If it detects a message sent *from your phone* (fromMe: true), it immediately sets a Redis key PAUSE:{ContactID} for a configurable duration (e.g., 60 minutes).
* **Dashboard Toggle:** A "Master Switch" on the web dashboard to kill all bot activity instantly.

## 4. Implementation Roadmap

### Phase 1: Infrastructure & Connectivity (Weeks 1-2)

* **Goal:** Establish a stable link to WhatsApp and a basic web UI.
* **Deliverables:**
  + Node.js server running whatsapp-web.js.
  + Next.js 16 dashboard streaming the QR Code via Socket.io.
  + Basic "Echo Bot" functionality to test sending/receiving.

### Phase 2: The Brain & Style Engine (Weeks 3-4)

* **Goal:** Connect LLMs and implement style customization.
* **Deliverables:**
  + Integration with Anthropic API (Sonnet 4.5).
  + "Persona Editor" in the UI to define styles (e.g., "Professional" vs. "Bestie").
  + Implementation of the **Haiku Router** to filter simple messages.

### Phase 3: Memory & GraphRAG (Weeks 5-6)

* **Goal:** Give the bot long-term memory.
* **Deliverables:**
  + Deploy **FalkorDB** (Dockerized).
  + Develop the **LangGraph** workflow for "Ingestion" (extracting facts from chat) and "Retrieval" (querying facts for replies).
  + Connect the graph context to the Sonnet 4.5 prompt.

### Phase 4: Polish & Safety (Weeks 7-8)

* **Goal:** Human mimicry and production readiness.
* **Deliverables:**
  + Implement **Human Typing Simulation (HTS)** algorithms.
  + Finalize the **Override/Auto-Pause** logic.
  + End-to-end testing with a pilot group of contacts.

## 5. Technical Requirements Summary

| **Component** | **Technology Choice** | **Reason** |
| --- | --- | --- |
| **Frontend** | **Next.js 16** | Stable Turbopack for fast dev, Server Actions for backend logic. |
| **Backend** | **Node.js** | Required for whatsapp-web.js (browser automation). |
| **AI Orchestration** | **LangGraph.js** | Native cyclic graph support for complex agent loops. |
| **Main LLM** | **Claude Sonnet 4.5** | Best-in-class reasoning and persona adherence. |
| **Graph DB** | **FalkorDB** | Low latency, high performance for RAG, simple Redis protocol. |
| **Vector DB** | **FalkorDB (Native)** | FalkorDB handles vector search natively, simplifying the stack. |
| **Hosting** | **Docker / VPS** | Must run 24/7 (e.g., Hetzner, DigitalOcean) to maintain WhatsApp socket. |

## 6. Conclusion

This architecture moves beyond simple "chatbots" to create a true **AI extension of yourself**. By leveraging **FalkorDB** for deep context and **Claude Sonnet 4.5** for nuanced expression, the system will be able to handle complex personal relationships with a high degree of fidelity. The strict adherence to **Human Typing Simulation** and **Override Protocols** ensures the system remains a helpful tool rather than a liability.