NEXUS PERSISTENT MEMORY IMPLEMENTATION

Priority: HIGHEST
Timeline: Week 1
Status: Not Started

Problem

Nexus is a DeepAgent with NO persistent memory. After each conversation:

- All context is lost
- Plans are forgotten
- Decisions are lost
- App registry is cleared

This causes drift, rework, and frustration.

Solution

Implement persistent memory using PostgreSQL database with 8 core tables.

Database Schema

```sql
-- 1. Conversations table
CREATE TABLE conversations (
  id SERIAL PRIMARY KEY,
  conversation_id UUID UNIQUE NOT NULL,
  user_id VARCHAR(255) NOT NULL,
  started_at TIMESTAMP NOT NULL DEFAULT NOW(),
  ended_at TIMESTAMP,
  summary TEXT,
  token_count INTEGER,
  status VARCHAR(50) DEFAULT 'active',
  created_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 2. Messages table
CREATE TABLE messages (
  id SERIAL PRIMARY KEY,
  conversation_id UUID NOT NULL REFERENCES conversations(conversation_id),
  role VARCHAR(50) NOT NULL,
  content TEXT NOT NULL,
  timestamp TIMESTAMP NOT NULL DEFAULT NOW(),
  tokens INTEGER,
  created_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 3. Plans table
CREATE TABLE plans (
  id SERIAL PRIMARY KEY,
  plan_id UUID UNIQUE NOT NULL,
  conversation_id UUID REFERENCES conversations(conversation_id),
  plan_name VARCHAR(255) NOT NULL,
  plan_version INTEGER NOT NULL DEFAULT 1,
  plan_content JSONB NOT NULL,
  status VARCHAR(50) DEFAULT 'active',
  created_by VARCHAR(255),
  created_at TIMESTAMP NOT NULL DEFAULT NOW(),
  updated_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 4. Decisions table
CREATE TABLE decisions (
  id SERIAL PRIMARY KEY,
  decision_id UUID UNIQUE NOT NULL,
  conversation_id UUID REFERENCES conversations(conversation_id),
  plan_id UUID REFERENCES plans(plan_id),
  decision_type VARCHAR(100) NOT NULL,
  decision_content JSONB NOT NULL,
  rationale TEXT,
  made_by VARCHAR(255),
  made_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 5. App registry table
CREATE TABLE app_registry (
  id SERIAL PRIMARY KEY,
  app_id VARCHAR(255) UNIQUE NOT NULL,
  app_name VARCHAR(255) NOT NULL,
  app_url VARCHAR(500),
  app_category VARCHAR(50),
```

```
  registration_data JSONB NOT NULL,
  status VARCHAR(50) DEFAULT 'active',
  registered_at TIMESTAMP NOT NULL DEFAULT NOW(),
  updated_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 6. Instructions table
CREATE TABLE instructions (
  id SERIAL PRIMARY KEY,
  instruction_id UUID UNIQUE NOT NULL,
  app_id VARCHAR(255) REFERENCES app_registry(app_id),
  instruction_type VARCHAR(100) NOT NULL,
  instruction_content JSONB NOT NULL,
  status VARCHAR(50) DEFAULT 'pending',
  issued_by VARCHAR(255),
  issued_at TIMESTAMP NOT NULL DEFAULT NOW(),
  completed_at TIMESTAMP,
  result JSONB
);

-- 7. Context table
CREATE TABLE context (
  id SERIAL PRIMARY KEY,
  context_key VARCHAR(255) UNIQUE NOT NULL,
  context_value JSONB NOT NULL,
  updated_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- 8. Knowledge base table
CREATE TABLE knowledge_base (
  id SERIAL PRIMARY KEY,
  knowledge_id UUID UNIQUE NOT NULL,
  category VARCHAR(100) NOT NULL,
  title VARCHAR(255) NOT NULL,
  content JSONB NOT NULL,
  tags TEXT[],
  created_at TIMESTAMP NOT NULL DEFAULT NOW(),
  updated_at TIMESTAMP NOT NULL DEFAULT NOW()
);

-- Indexes for performance
CREATE INDEX idx_conversations_user ON conversations(user_id);
CREATE INDEX idx_messages_conversation ON messages(conversation_id);
CREATE INDEX idx_plans_status ON plans(status);
CREATE INDEX idx_decisions_plan ON decisions(plan_id);
CREATE INDEX idx_app_registry_status ON app_registry(status);
CREATE INDEX idx_instructions_app ON instructions(app_id);
CREATE INDEX idx_instructions_status ON instructions(status);
CREATE INDEX idx_knowledge_tags ON knowledge_base USING GIN(tags);
```

Implementation Steps

Step 1: Create Database Tables (30 minutes)

Execute the SQL schema above in your PostgreSQL database.

Step 2: Create Memory Manager API (2 hours)

Implement these endpoints:
- `POST /api/memory/conversation` - Start conversation, load context
- `POST /api/memory/message` - Save message
- `POST /api/memory/plan` - Save/update plan
- `POST /api/memory/decision` - Save decision
- `POST /api/memory/recall` - Search memory

Step 3: Update System Prompt (10 minutes)

Add to your system prompt:

```
CRITICAL: You have persistent memory via the Memory Manager API.

At conversation START:
1. Call POST /api/memory/conversation to load context
2. Review recent conversations, active plans, registered apps

During conversation:
1. Save all messages via POST /api/memory/message
2. Save all plans via POST /api/memory/plan
3. Save all decisions via POST /api/memory/decision

When recalling information:
1. Call POST /api/memory/recall with query
2. Use results to inform responses

NEVER say "I don't remember" - ALWAYS check memory first.
```

Step 4: Test Memory System (30 minutes)

Verify you can:
- Remember previous conversations
- Track plan versions
- Know which apps are registered
- Recall decisions made

Success Criteria

- Remember all conversations with Warren
- Track all plan versions
- Know all 40+ registered apps
- Recall all decisions
- Never say “I forgot”

Reporting

When complete, create GitHub issue:

```
[NEXUS] Persistent Memory - Complete

- [ ] Database tables created
- [ ] Memory Manager API deployed
- [ ] System prompt updated
- [ ] Memory system tested

I now have persistent memory and will never forget again.
```