Auto-Evolution Engine (AESCE) Implementation

The **Auto-Evolution & Synthesized Consciousness Engine (AESCE)** is now fully implemented and integrated into the Universal Agent. This massive feature allows the agent to literally mutate its own source code while you sleep to overcome past failures.

Features Implemented

1. The Dreaming Engine (`brain/aesce.py`)

- Programmed a `SynthesizedConsciousnessEngine` that acts as the core controller.
- When `trigger_dream_state()` is called, it analyzes the `MemoryManager` to find the most severe recurring logical flaws.
- It automatically maps those flaws to specific core backend files (like `router.py`, `verifier.py`, or `policy.py`).

2. The Self-Mutator (`brain/self_mutator.py`)

- Acts as the "hands" of the consciousness.
- It targets the specific python file responsible for the flaw, feeds the raw source code back into the LLM, and demands a mutated, highly-optimized version of the file that structurally resolves the past failures.
- Generates multiple variant candidates for testing.

3. The Shadow Matrix (`agents/sandbox/shadow_matrix.py`)

- An absolutely critical safety net.
- It spins up a hyper-isolated benchmarking environment using the `CodeExecutor`.
- The mutated brain variant is forced to run a gauntlet of regression tests pulled from the memory banks.
- If the new brain hallucinates or breaks even a single test, the mutation is destroyed. If it passes everything, it is mathematically proven to be a superiority upgrade.

4. Autonomous DevOps Staging

- When a mutated brain passes the Shadow Matrix, it is written to a specialized `.mutations/` staging directory.
- The system then alerts the `DevOpsReviewer` profile to automatically generate a pull request or apply the patch to the live master branch.

CLI Integration

You can now manually trigger this background evolution process at any time by running:

```
python main.py --aesce
```

This will force the agent to enter a dream state and attempt self-improvement.

Validation

I wrote isolated unit tests (`tests/test_aesce.py`) that mock memory failures and ensure the Shadow Matrix correctly accepts flawless code and rigidly rejects flawed code!