

# Optimizing AI Economics through Intelligent Routing

A technical breakdown of the Cost-Control Smart Model Router—reducing API costs by up to 99.6% without sacrificing response quality.



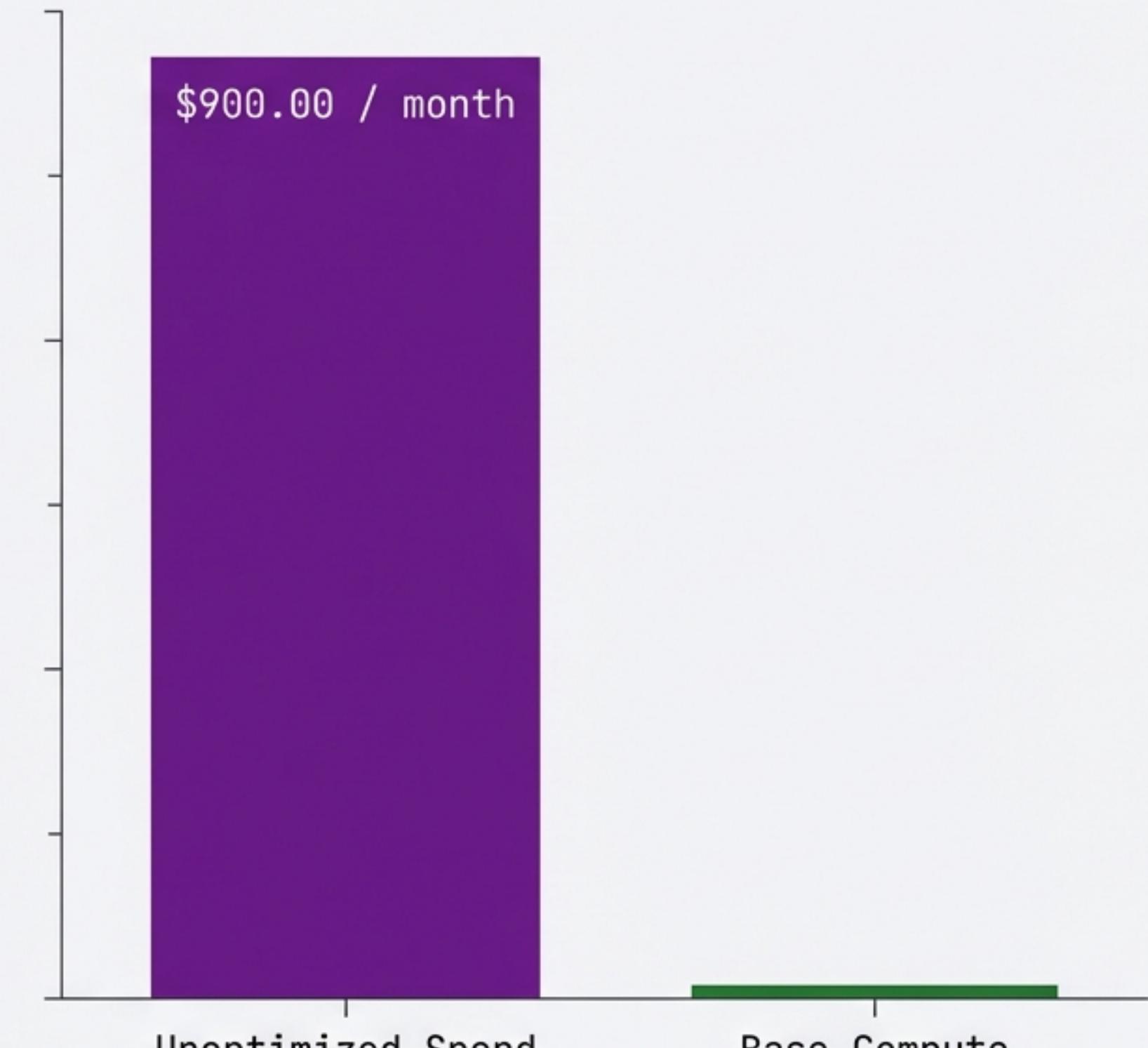
Python 3.13 Compatible

FastAPI + Streamlit Architecture

MIT License

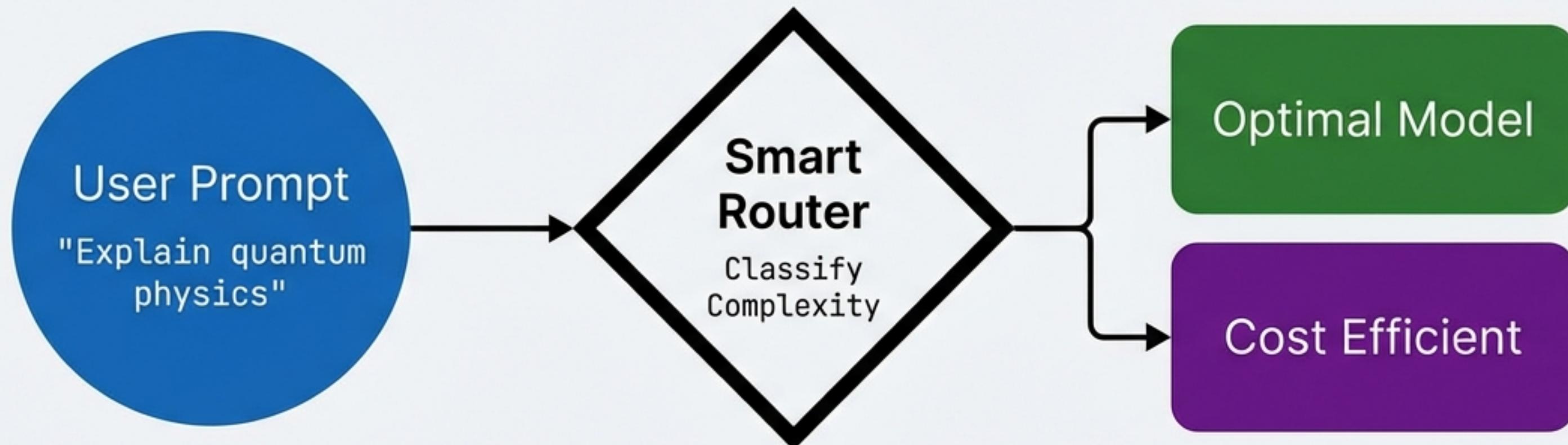
# Stop Treating Every User Prompt Like a PhD Thesis

Most applications default to a “One-Size-Fits-All” strategy, utilizing expensive models like GPT-4o for trivial tasks. This inefficiency scales linearly with user growth, resulting in massive waste on simple queries like greetings or factual lookups.



Scenario based on 1,000 requests/day at average \$0.03/request.

# The Solution: A Pre-Inference Decision Layer



**Core Value Proposition:** Reduce API costs by up to 70-99% while maintaining response quality.

# A Full-Stack System for Intelligent Traffic Control

A Python-based routing system integrating backend logic with frontend analytics.



## Smart Routing

Automatically classifies prompts and routes to optimal models based on difficulty.



## Cost Optimization

Dynamic selection of cheaper models for simple tasks to minimize burn rate.



## Multi-Provider Support

Native integration for OpenAI (GPT-4o), Google (Gemini), and local models.



## Real-time Analytics

Track savings, latency, and token usage via a built-in Streamlit dashboard.



## Auto-Discovery

System automatically detects and configures available API keys and models.

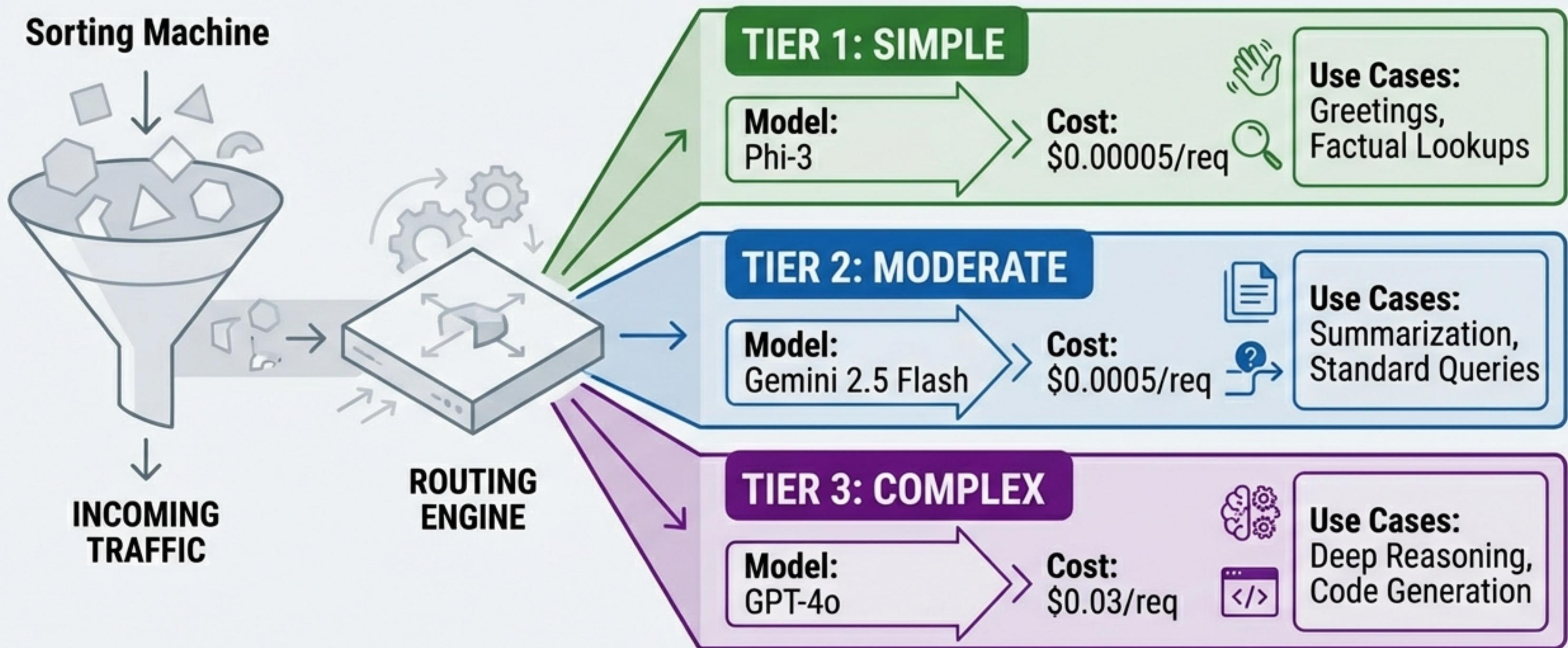


## Secure & Scalable

Built for enterprise deployment with secure key management and scalable architecture.

# The Tiered Routing Logic

The router sorts incoming traffic into three distinct buckets based on complexity analysis.



# Flexible Classification Engines

Administrators can toggle the classification mechanism via the environment configuration.



## Rule-Based Mode

**Mechanism:** Keyword matching and heuristics.

**Benefit:** Zero latency, zero cost.



## LLM-Based Mode

**Mechanism:** Lightweight model semantic analysis.

**Benefit:** High nuance and accuracy for ambiguity.

.env

```
1 # Classifier Configuration
2 CLASSIFIER_TYPE=rules
3 # Options: 'rules' or 'llm'
```

# The Economics of Smart Routing

By routing 90% of traffic to cheaper models, costs collapse while utility remains constant.

## WITHOUT ROUTER

**\$900.00 / month**

100% Traffic to GPT-4o

## WITH SMART ROUTER

**\$3.18 / month**

Optimization Breakdown:

- 600 Simple Requests (Phi-3): \$0.03
- 300 Moderate Requests (Gemini): \$0.15
- 100 Complex Requests (GPT-4o): \$3.00

**TOTAL SAVINGS: 99.6%**

# Real-Time Visibility via Streamlit

The screenshot displays a Streamlit application interface with a dark theme. On the left, a sidebar menu includes "JetBrains Mono", "Settings", and "API Configuration". The main area features a large green-to-purple gradient box at the top with the text "Cost Savings" and "\$896.82 saved" in white, followed by "vs. 100% GPT-4o". Below this is a card for "Model: GPT-4o" containing a "Quantum physics" section with explanatory text and a "Explain quantum physics" button. At the bottom is a "Model Distribution" card showing a green bar for Llama-3-70B (85%) and a purple bar for GPT-4o (15%). A blue arrow points from the right towards the "Explain quantum physics" button, with the text "Interactive Prompt Testing & Analytics" overlaid.

JetBrains Mono

Settings

API Configuration

Cost Savings  
\$896.82 saved

vs. 100% GPT-4o

Model: GPT-4o

Quantum physics is used to understand atomic systems. Explain quantum physics can assimilate quantum numbers, which combines information from different areas of quantum mechanics. It provides experience with atomic nuclei and their interaction with particles and photons: physical and empirical.

Explain quantum physics

Model Distribution

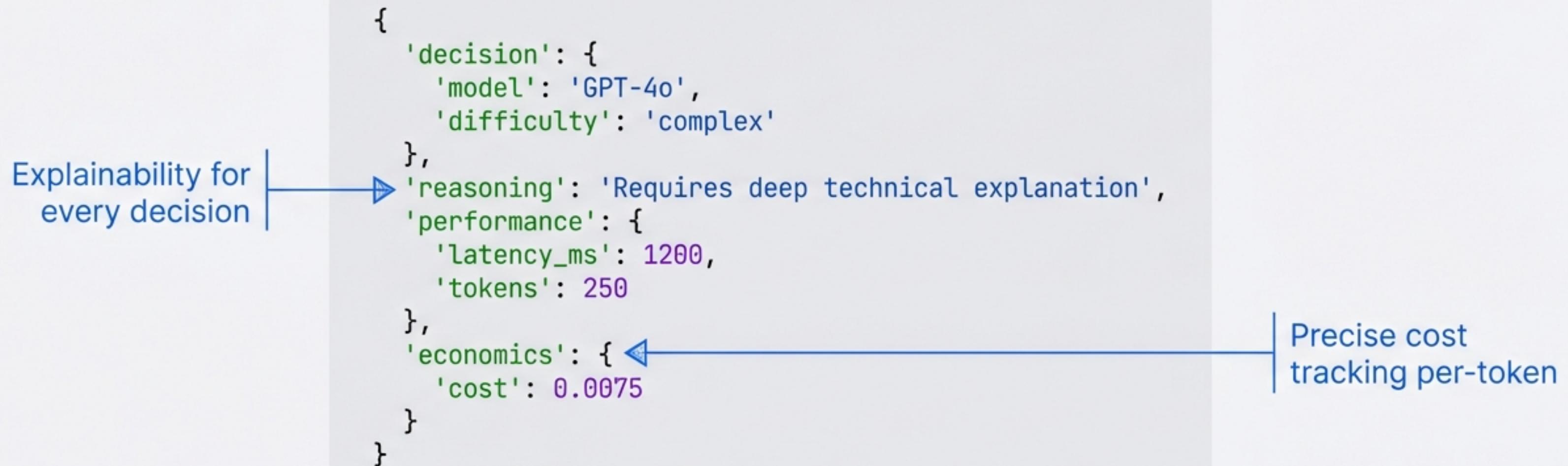
85%  
Llama-3-70B

15%  
GPT-4o

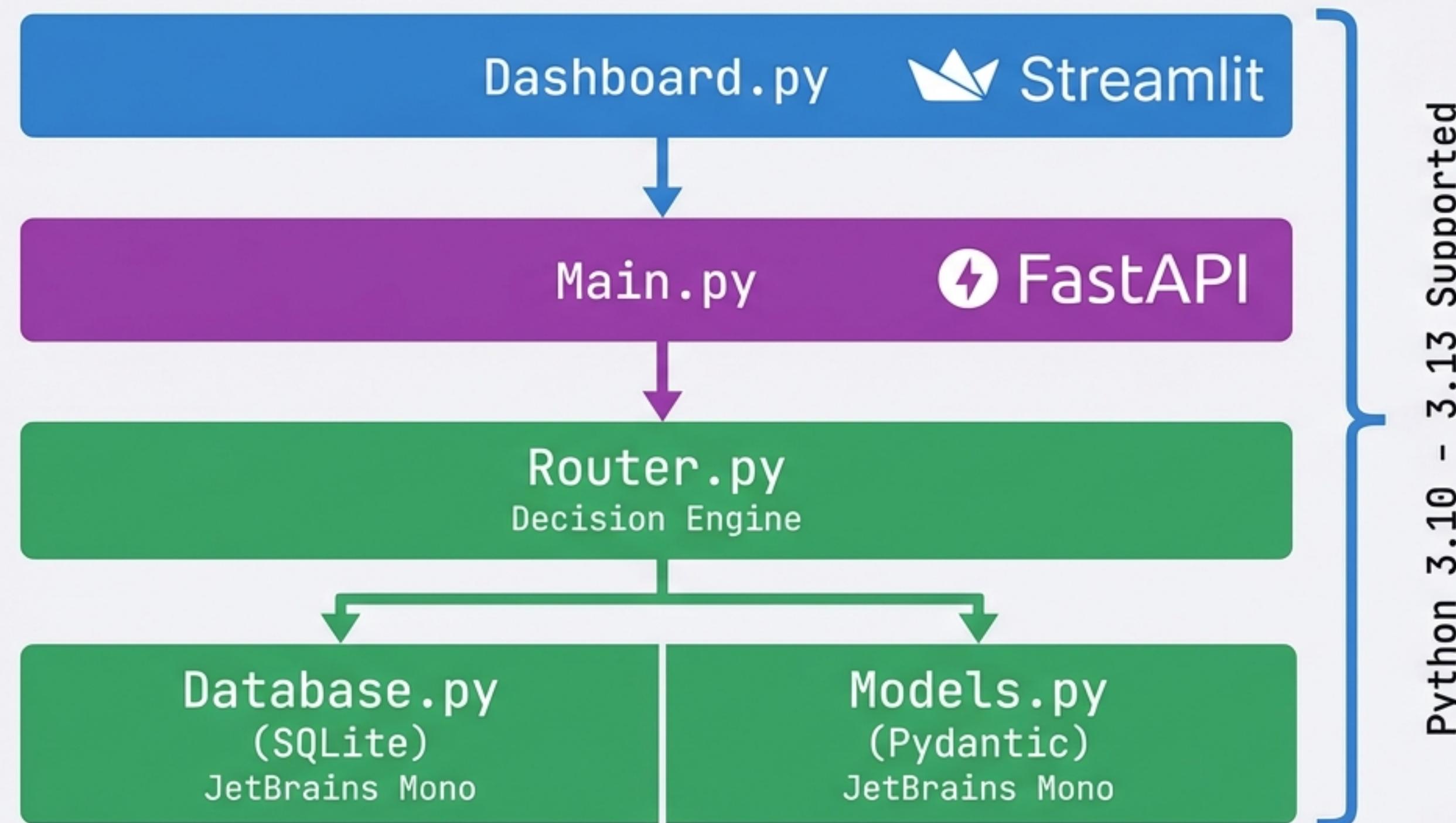
Interactive Prompt Testing & Analytics

# Granular Analytics & Transparency

The system provides detailed metadata for every request, ensuring routing logic is fully auditable.

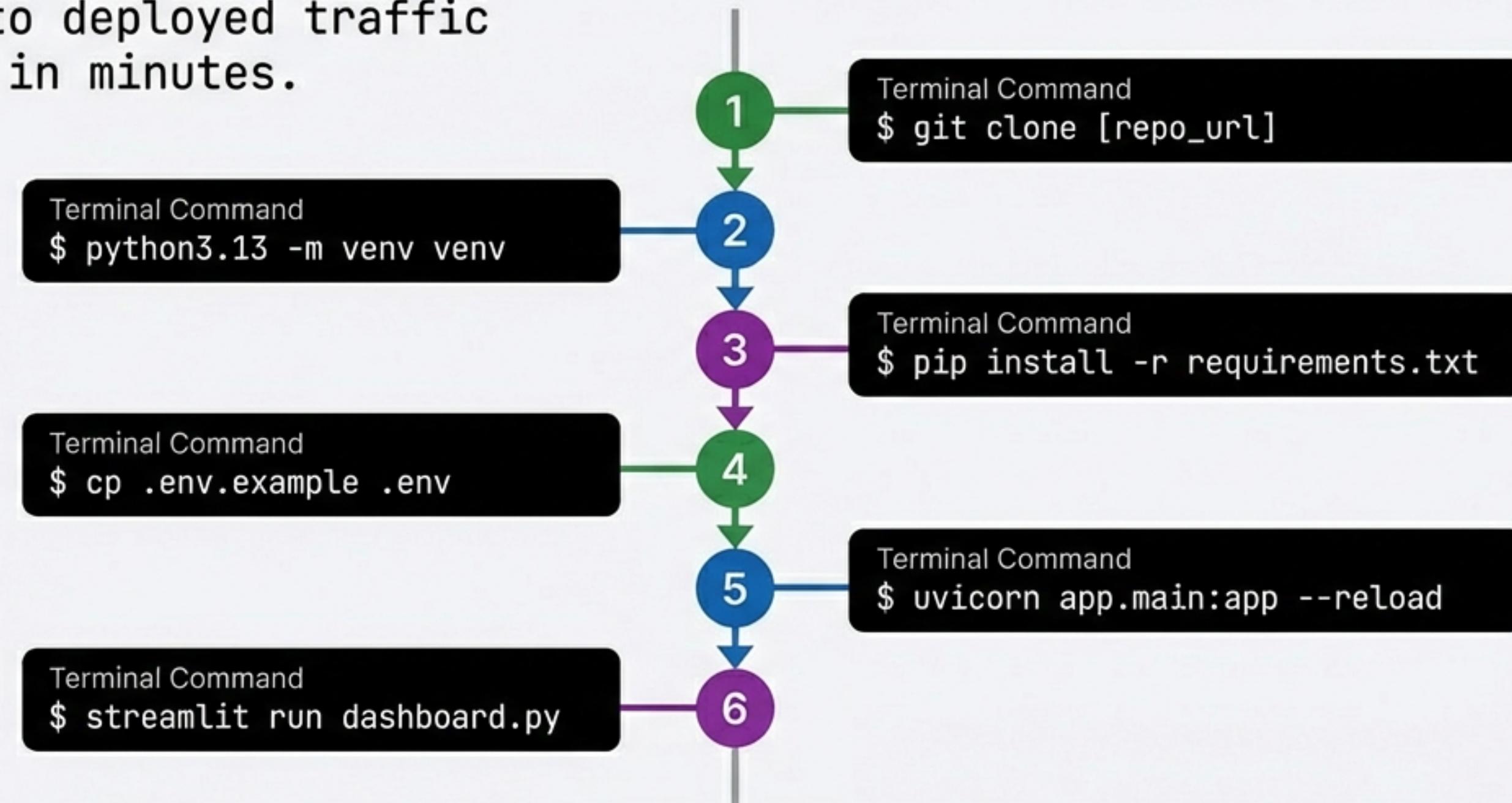


# Built on a Modern Python Stack



# Deployment in Six Steps

From zero to deployed traffic controller in minutes.



Access Points: Dashboard @ localhost:8501 | API @ localhost:8000

# Designed for Extensibility

Adding new providers (e.g., Claude) is achieved through clean class inheritance.

## 1. Define Client (app/llm/providers.py)

```
1 class ClaudeClient(LLMClient):  
2     async def generate(self, prompt, ...):  
3         # Implementation  
4         return response, cost, tokens  
5 
```

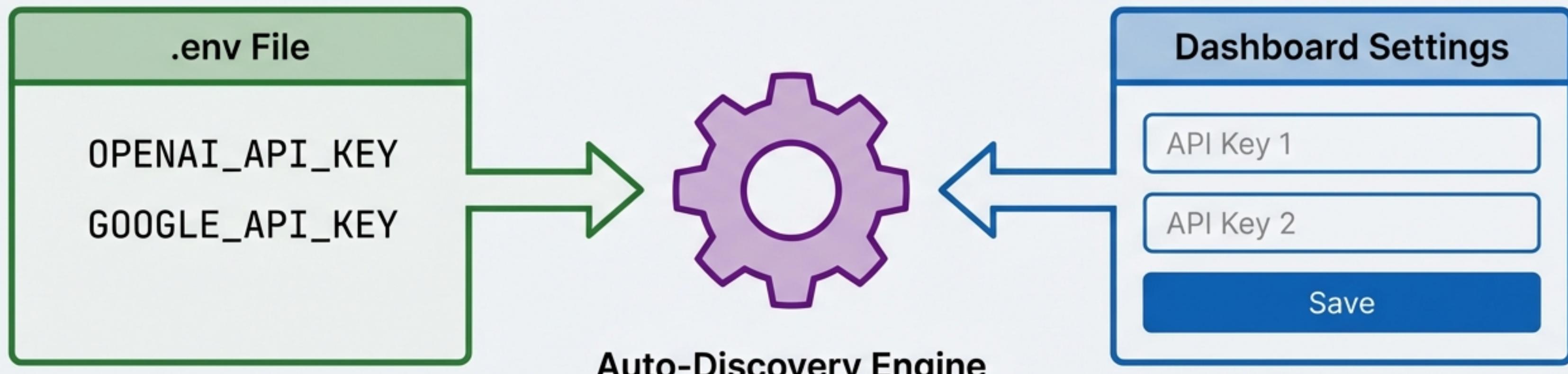
→  
Adding new providers...

## 2. Register Model (app/router.py)

```
1 self.clients = {  
2     'simple': Phi3Client(),  
3     'moderate': GeminiClient(),  
4     'complex': ClaudeClient() # Added  
5 } 
```

# Configuration & Auto-Discovery

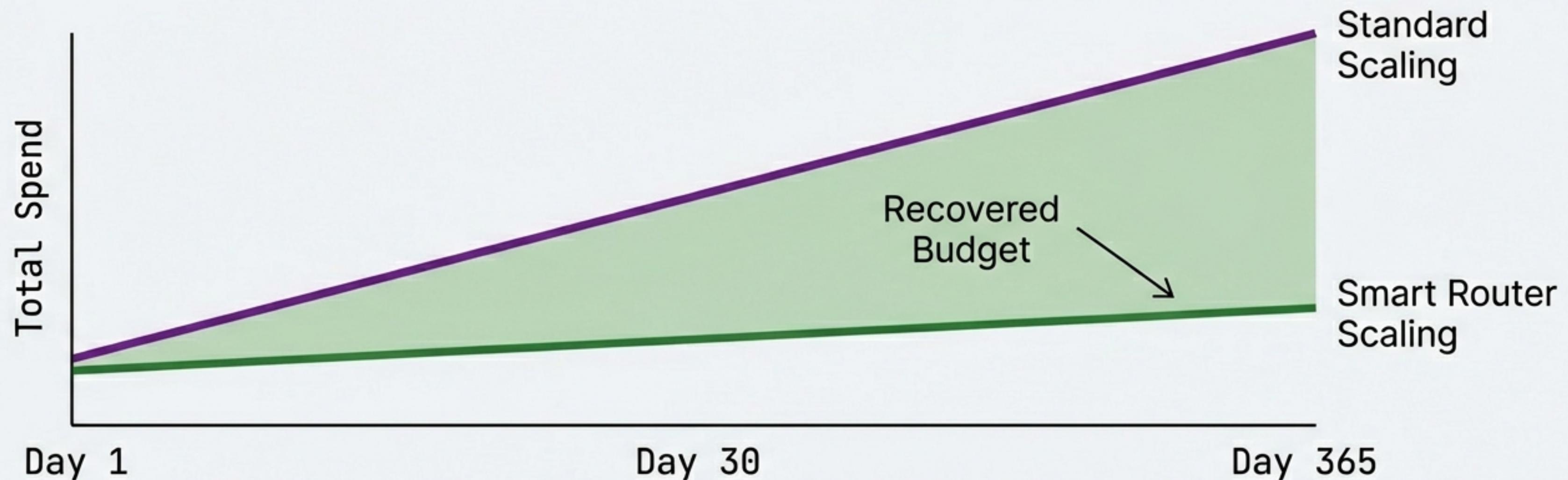
The system automatically identifies available models based on provided keys.



System detects capabilities and  
enables Smart Classification.

# Small Optimizations, Enterprise Impact

Per-request savings compound significantly over time.



“What gets measured gets managed. What gets routed gets saved.”

# Route Smart, Build Faster

The Cost-Control Smart Model Router is open source and ready for contribution.



A GitHub repository card for the 'Cost-Control-Smart-Model-Router' project. The card features a blue folder icon with a white GitHub logo inside. To the right of the icon, the repository name 'adi-7192 / Cost-Control-Smart-Model-Router' is displayed. Below the name, there are three status indicators: 'MIT License' with a green shield icon, 'Python' with a blue Python logo, and 'Open Source' with a purple globe icon.

adi-7192 /  
**Cost-Control-Smart-Model-Router**

MIT License   Python   Open Source

- Fork the repo to customize routing logic.
- Submit Pull Requests for new providers.
- Deploy your own traffic controller.

```
> git clone https://github.com/adi-7192/Cost-Control-Smart-Model-Router
```