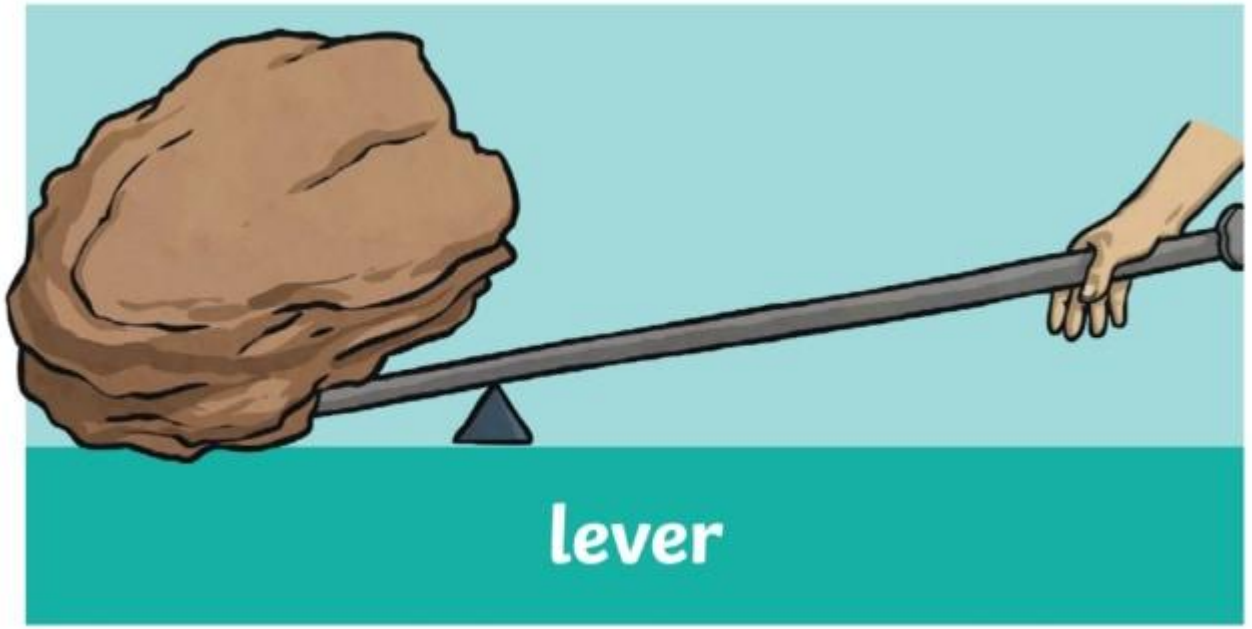
Preparing for the AI Revolution:

From Writing Code to Orchestrating Cognition

Krishnakumar N.
Principal Engineer
Core AI, Microsoft

Prologue

Understanding Leverage



Leverage

Lever is a simple machine that follows the principle of Conservation of Energy. It doesn't create energy, but it transforms a small force into a large force

Leverage: Using a small input of - special tool helps to achieve a disproportionately large output

Example:

Mechanical Leverage

Financial Leverage

Operational Leverage

Time Leverage

Part 1

Becoming the Architect of Intelligence

Why developers must pivot from syntax to strategy

The Core Shift: From **How** to **What**

The Pre-AI Era Question

"HOW will I write this feature?"

The engineer's value was in manually translating business needs into code syntax and implementation details.

The AI Era Question

"WHAT is the objective of this system?"

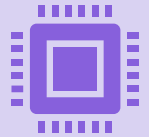
The engineer's value is in defining the purpose, constraints, and success criteria for the AI.

Key to Success in AI era is to transform from an How Engineer to a What Engineer

From Writer to Editor-In-Chief

The "Gartner" Prediction
Gartner predicts that by 2028, **90% of software engineers** will shift from "coding" to "orchestrating."

The role of the developer isn't dying, but it is fast-forwarding.



The era of "selling syntax" is over; the era of "selling solutions" has begun.

AI is rapidly commoditizing the "middle" work—writing boilerplate, unit tests, and refactoring.

The opportunity is that AI gives us the leverage to become "10x Architects" overnight if we pivot correctly

The Danger Zone (The Mid-Level Trap)

Mid-level engineers often pride themselves on knowing the "How"—obscure syntax, complex library implementations, and writing perfect boilerplate.

The Hard Truth: AI is now better at the "How" than you are. It knows every library, every syntax, and every regex pattern.

The Shift: If you stay stuck in the "How," you become obsolete. You must move to the "What" (Product) and the "Why" (Architecture).

The developer’s skill upgrade 101

Master IDE-integrated AI

(Cursor, VS Code Copilot etc).AI takes over mundane coding tasks, freeing developers to focus on higher-level challenges and innovation. Learn "Context Management." AI is only as good as the context you give it.

Emphasis on Strategic Thinking

Prioritize system design and strategic problem-solving. Your value is now your *taste* and *judgment*, not your typing speed.

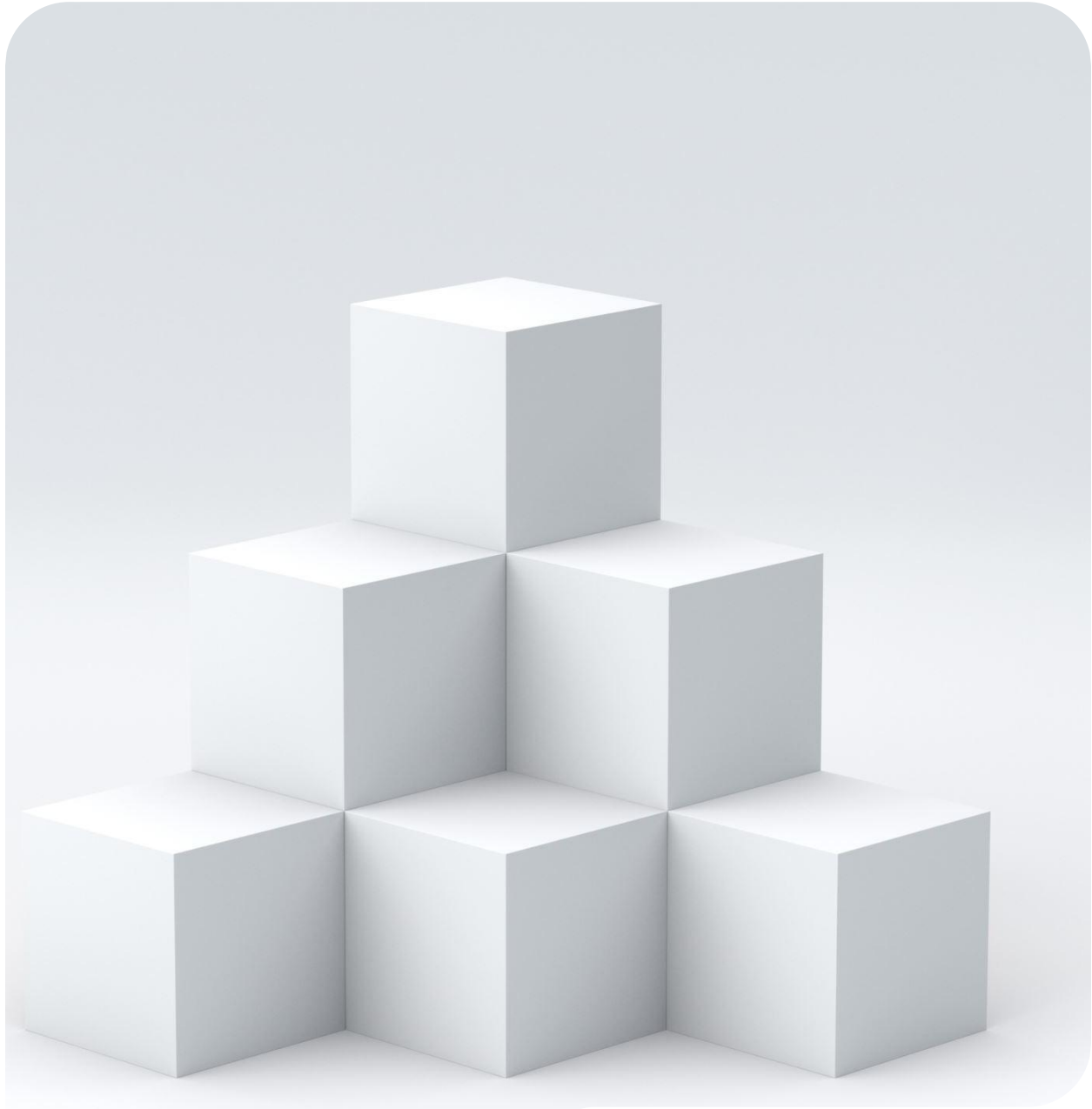
Focus on Conceptual Skills

The engineer who can translate vague business requirements into a technical spec that an AI can execute is the most valuable person in the room.

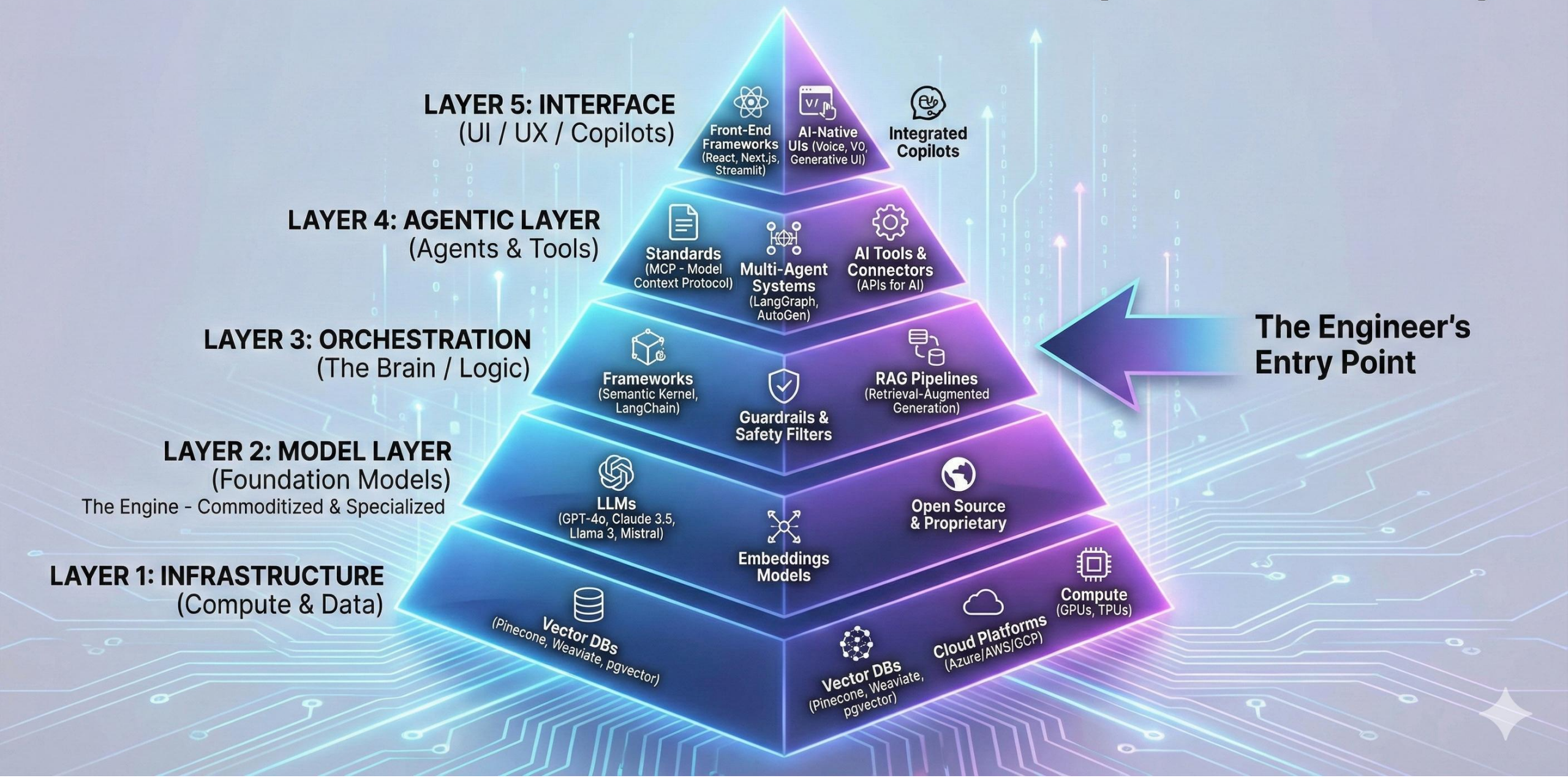
Part 2

Understanding the AI Developer Stack Pyramid

Build Intelligence, Layer by Layer



THE AI DEVELOPER STACK PYRAMID (2025 EDITION)



The AI Pyramid

LAYER	COMPONENT	EXAMPLES	DEVELOPER OPPORTUNITY
5. Interface (Top)	**UI / UX / Copilots**	React, Streamlit, Voice, V0, Next.js	**High.** Building the "glass" the user touches.
4. Agentic Layer	**Agents & Tools**	MCP (Model Context Protocol), LangGraph, AutoGen	**CRITICAL.** Defining *what* the AI can do (Tools).
3. Orchestration	**The Brain / Logic**	Semantic Kernel, LangChain, Guardrails	**CRITICAL.** Wiring the brain to the body.
2. Model Layer	**Foundation Models**	GPT-4o, Claude 3.5, Llama 3, Mistral	**Low.** (Unless you are an ML Researcher).
1. Infra (Bottom)	**Compute & Data**	Vector DBs (Pinecone/Weaviate), Azure/AWS, GPUs	**Medium.** Backend/Platform Engineering.
Layer	Component	Examples	Developer Opportunity

Layer 1: Infrastructure & Data (The Bedrock)

Purpose: The Compute, Data, and Pipeline Foundation.
Provides clean, fast, and scalable context to the AI models (Layers 2-5).

Opportunity : BIG
Vast area for growth, especially for existing infrastructure and data skills

AI Leverage: Minimal disruption for the engineer who quickly pivots to become the AI's supervisor,.

Developers Role	Key Responsibility in Layer 1
Cloud Engineer	Manage, Provision, and **Scale** specialized GPU clusters for cost-effective inference.
Data Engineer	Ingest and manage data. Focus shifts from ETL (Extract-Transform-Load) to **Embedding, Chunking, and RAG Pipeline Management.**
Solutions Architect	Builds the system blueprint: **On-Premise vs Cloud** decisions, scale modeling, and **Cost** optimization.
AI Ops / SRE	**Infra as Code (IaC)**, Reliability, Monitoring, and building **self-healing** systems for data pipelines.

Layer 2: AI Models (Foundation)

Purpose: Training the actual Large Language Models (LLMs).

Opportunity : No-Go Zone !

Unless you have a PhD in Math and access to 10,000 H100 GPUs, do not try to compete here. Don't build your own LLM from scratch. Treat this layer like a Utility—like Electricity or AWS

AI Leverage: Peripheral roles for data ingestion, validation and Infra management and OPS roles,.

Developers Role	Key Responsibility in Layer 1
Model Validator	Setting up automated **Evaluation (Eval) metrics** and benchmarks to ensure model quality before deployment.
Data Scientist / Engineer	Ingest and manage data. Focus shifts from ETL (Extract-Transform-Load)
AI Ops / SRE	**Infra as Code (IaC)**, Reliability, Monitoring, and building **self-healing** systems for data pipelines.

Layer 3: Orchestration (The Brain)

Purpose: To turn a single API call (the raw LLM) into a multi-step, reliable, and governed business workflow.
Coordinated, context-aware, and safe sequences of actions for the Agentic Layer (Layer 4).

Opportunity : The differentiator!

Learn how to implement **RAG .**This is how you connect your company's private data (PDFs, SQL databases) to the public AI.
Stop writing features. Start writing **Planners and Guardrails**.

AI Leverage: Peripheral roles for data ingestion, validation and Infra management and OPS roles,.

Developers Role	Key Responsibility in Layer 1
AI Engineer (Hot role !!)	Using **Semantic Kernel** (C#/.NET) or **LangChain** (Python) to define sequential/parallel steps, manage context, and handle multi-agent collaboration.
RAG Specialist	Designing and optimizing the **Retrieval-Augmented Generation (RAG)** pipeline
AI Architect (Hot role !!)	Defining the core orchestration framework based on the tech stack

Layer 4: The Agentic Layer (The "Hands")

Purpose: To endow the AI with the capability to perform real-world actions by calling APIs, services, and running code.

Autonomous, goal-driven execution, user workflow automation.

Opportunity : The Gold rush !

Learn to build MCPs, Agents, RAGs, Tools, Agent to Agent communications

Build the future of Engineering

AI Leverage: Stop writing code that executes logic. Start writing **Descriptions and Permissions** for the Tools that *enable the AI to write and execute the logic itself*. You are building the **Autonomous API Ecosystem**

Developers Role	Key Responsibility in Layer 1
Multi-Agent Orchestrator **(Hot role !!)**	Designing systems where multiple specialized agents (e.g., a "Planner Agent," a "Coder Agent," a "Test Agent") work together to achieve a complex goal.
Backend Engineer	Convert legacy applications to robust APIs and MCPs
Security Engineer (Emerging role !!)	Defining the agent's persona, its capabilities, and the guardrails for its actions to ensure a safe and effective user experience.

Layer 5: Interface (UI/UX / Copilots)

Purpose: To deliver the orchestrated output (from Layer 3/4) to the user in a seamless, interactive, and AI-native way.

Intuitive Copilots, conversational interfaces, and dynamically personalized user experiences.

Opportunity : Big!

Rapid prototyping. Vision to a running application in a day !

Build the next gen experiences

AI Leverage: The best frontend engineers are pivoting from writing code to curating, auditing, and designing the unique UX challenges of non-deterministic systems.

Developers Role	Key Responsibility in Layer 1
Dev-Innovator	Drives rapid prototyping of features and internal tools
UX/UI Designer:	Designs **conversational flows** and UI elements that build **user trust**
Frontend Developer	Uses tools for **90% code generation**, focusing their time on **curation, refinement, and unique human-centric UX**.

Action Plan

Stop Competing with the Model (Layer 2):

Don't try to build the LLM. **Do** pivot into **MLOps** to make the LLM reliable, fine-tuned, and scalable.

Focus on the High-Leverage Layers (Layers 3 & 4):

This is the engineering sweet spot. Your systems thinking is needed to define **Guardrails** (safety/compliance) and build the **Tools** that allow agents to act on your company's behalf. **This is where you earn your Architect title.**

Harness the Accelerator (Layers 1 & 5):

Use AI to eliminate boring work. Master the **Vector Database** (Layer 1) to give the AI accurate memory, and leverage **AI Code Generation** (Layer 5) to launch features 50% faster.

Epilogue – The AI Leverage

How AI Magnifies Human Effort:

Time Amplification: An employee who uses AI to draft a report can complete the task in 1 hour instead of 8 hours. The same 1 hour of human effort is "leveraged" to produce 8 times the output.

Scalability: A small team can use AI-powered chatbots to handle 10,000 customer inquiries per day. The cost of adding more capacity (more chatbots) is much lower than hiring 100 new employees to handle the same volume.

Insight Magnification: AI can analyze petabytes of customer data to identify a tiny but highly profitable trend that a team of human analysts might spend months or years trying to find. The small input of computing power is leveraged to yield massive strategic value.

AI creates a powerful competitive advantage,

Thank you !!