You got it, Prince! Here's a clean, outline for your presentation—highlighting how your RAG-enhanced approach slashes cost and boosts efficiency compared to legacy static prompt methods. I’ll mimic a typical internal deck structure with enterprise tone, clean segmentation, and punchy visuals.

**🧠 Slide 1: Executive Summary**

**Title**: Cost Optimization via RAG-Driven Domain Classification  
**Subtitle**: Mapping Reports at Scale Using GPT-4 Turbo + LangChain

* ✅ Migrated from 20K-token static prompts to efficient RAG-based invocations
* 💸 Per Excel file cost reduced by **~90%**
* ⚙️ RAG enables **dynamic, reusable retrieval** across teams and mappings
* 📊 Direct impact: Scalable classification of Excel rows with controlled token budget

**📉 Slide 2: Previous Architecture – Static Prompt Method**

**Description**:  
Every row in Excel triggers an LLM call with a full 20,000-token document embedded in the prompt. No retrieval, no memory reuse.

|  |  |
| --- | --- |
| Metric | Value |
| Tokens per row | 20,400 |
| Rows per file | 10 |
| Total tokens/file | 204,000 |
| Cost estimate | $2.12 / file |
| Model used | GPT-4-turbo |

⚠️ **Challenges**:

* High token cost
* Repetitive overhead
* Context cap risk
* No scalability

**⚙️ Slide 3: Optimized Architecture – RAG Invocation**

**Description**:  
RAG retriever built once from domain reference document (20K tokens). Excel rows query the retriever with lightweight text → targeted domain/subdomain output.

|  |  |
| --- | --- |
| Metric | Value |
| Tokens per row | ~1,400 |
| Rows per file | 10 |
| Total tokens/file | 14,000 |
| Cost estimate | $0.22 / file |
| Model used | GPT-4-turbo + LangChain RetrievalQA |

✅ **Benefits**:

* ~90% cost reduction
* Fast processing per row
* Reusable across teams
* No prompt size bottleneck

**💰 Slide 4: Cost Comparison Snapshot**

|  |  |  |  |
| --- | --- | --- | --- |
| Approach | Tokens/file | Cost/file | 🔻 Savings |
| Static Prompt | 204,000 | $2.12 | — |
| RAG Invocation | 14,000 | $0.22 | **~$1.90 / file (~90%)** |

📌 Annual savings scale with volume:  
1000 files/year → **$1,900+ saved**

**🧩 Slide 5: Architecture Diagram (Suggested Visual Elements)**

* Left: 📄 Domain file → Chunking + Embedding → FAISS retriever
* Right: 📊 Excel row → Query → RetrievalQA → Domain/Subdomain output
* Middle: 🔄 Shared retriever node used by all rows

I can mock up this visual for you, or you can slide it into PowerPoint using boxes and arrows for quick rendering.

Let me know if you want this exported into PowerPoint structure, add JPMorgan-standard fonts/colors, or embed this into a readout email for internal sharing. You nailed the optimization—let’s make the messaging just as sharp ⚡📈.