Agentic AI Finance Assistant - POC Documentation

# 1. Overview

This document details the implementation of a Proof of Concept (POC) using Agentic AI on Amazon Bedrock. The goal was to build a Finance Assistant agent using Meta's LLaMA model on Bedrock that calculates income tax, provides financial summaries, and answers tax-related questions using a knowledge base (KB).

# 2. Architecture Overview

The system uses the following components:  
- Amazon Bedrock: Agent Orchestration and LLM inference (Meta LLaMA)  
- AWS Lambda: Logic execution (tax calculation)  
- Amazon S3: File storage  
- Titan Embeddings: Used for Knowledge Base  
- Knowledge Base: For answering factual finance/tax queries  
- Agent Action Group: Configured for tool invocation

# 3. Use Cases

- Calculate income tax for a given income.  
- Provide a clean financial summary.  
- Answer factual queries from the KB (e.g., tax slabs, Section 80C).

# 4. Agent Action Group Configuration

Action Group includes tools such as:  
- calculateTax: Lambda function to compute tax.  
- financialSummary: Lambda function to generate summaries.  
- knowledgeLookup: Used for KB interaction.  
  
Each tool is triggered by identifying user intent and matching it to a predefined action in the group.

# 5. Implementation Details

- Lambda Function for Tax Calculation (Python): Parses income input, applies slabs, and returns estimated tax.  
- Lambda Function for Summary: Uses tax output and formats summary with suggestions.  
- Knowledge Base: Indexed with Titan Embeddings. Example content: tax slabs, deductions.  
- Agent: Hosted on Amazon Bedrock. Uses Meta LLaMA. Returns message with `messageVersion: 1.0` and response wrapped in `TEXT` block for proper parsing.

# 6. Sample User Interactions

User: What is my tax for ₹8,00,000?  
Agent: Your estimated tax is ₹72,500.  
  
User: What are the slabs for FY 2023–24?  
Agent: The income tax slabs for FY 2023–24 are... (details fetched from KB)

# 7. Debugging Challenges & Fixes

- Fixed Lambda input parsing issues (income format, missing fields).  
- Resolved null toolResult errors.  
- Adjusted Action Group schema.  
- Wrapped final response in proper format to ensure LLaMA understanding.

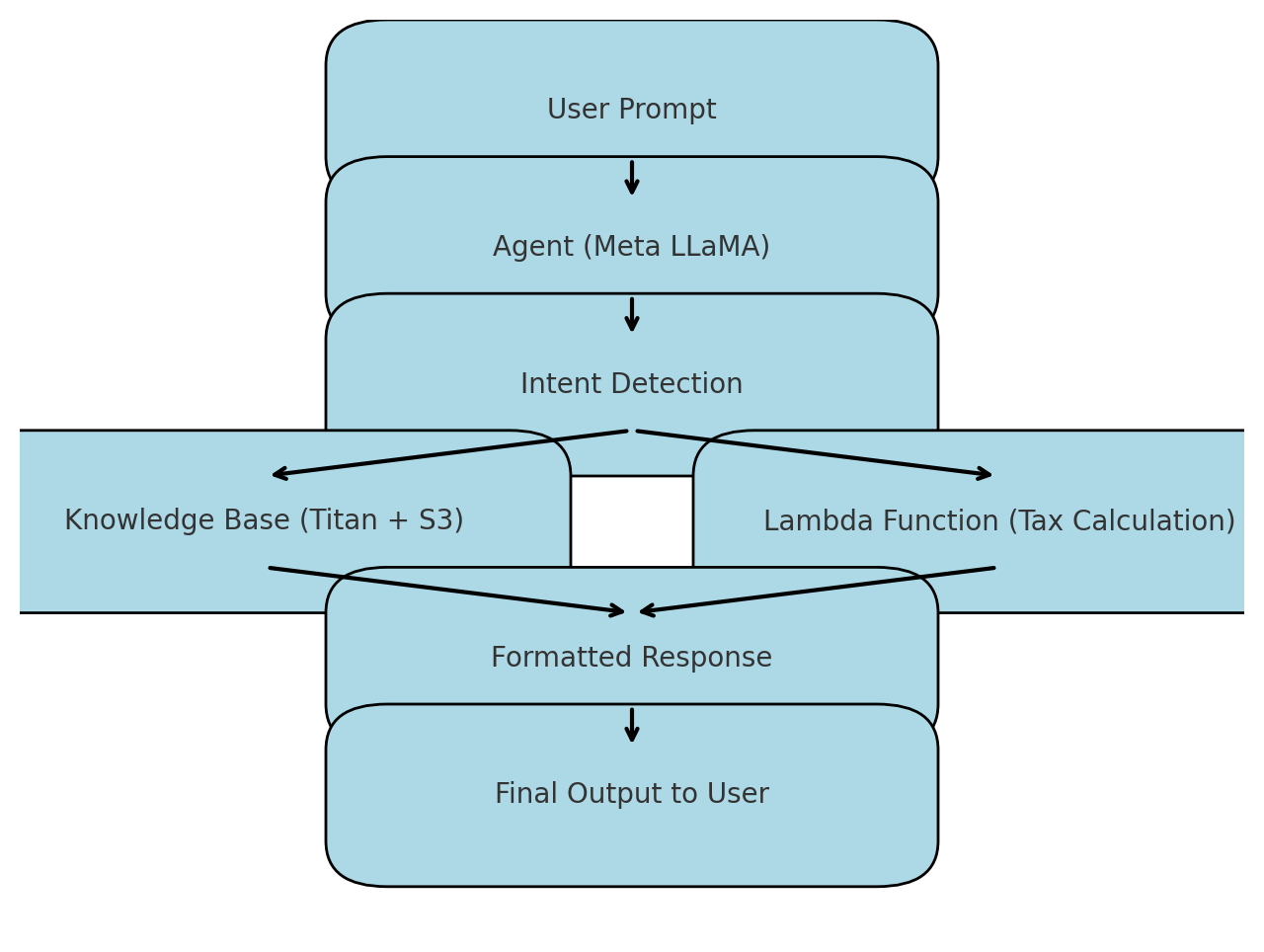
# 8. Key Learnings

- Combining Lambda and Knowledge Base allows modular and scalable GenAI logic.  
- Dynamic tool invocation based on user prompt enabled fluid, intelligent conversations.  
- Proper schema, formatting, and API wiring critical for Bedrock-based agent success.

# 9. Conclusion

This POC demonstrates the potential of serverless GenAI orchestration with Amazon Bedrock, Meta’s LLaMA, and a clean, modular cloud-native architecture.

# 10. Architecture Flowchart



# 11. Demo Screenshot

A screenshot of a phone

AI-generated content may be incorrect.