

Kotak Securities AI Workshop

Use Cases and Tool Schemas

Compiled: January 17, 2026

Contents

1. Use Case: Email Routing & SPOC Identification (Aaron Pereira)
2. Use Case: New Joiner Onboarding Assistant (Aaron Pereira)
3. Use Case: Speech-to-Text Analytics for Contact Center (Jimmy Joshi)
4. Tool Schemas: Query Intake & Resolution System (Jyotbir Lamba)

1. Email Routing & SPOC Identification

Submitted by: Aaron Pereira (Product, KSL)

Date: January 17, 2026

Problem Statement:

In Account Opening, we get a lot of emails from different teams (Customer Service, Offline Channels, Ops team, Vendor teams) with different types of issues. These often get directed to the incorrect SPOC and the first few days get wasted in understanding the issue and finding the correct SPOC to address the issue. The urgency of the issue, problem to solve is sometimes lost due to the long mail thread.

Recommended Architecture (from workshop feedback):

- SearchEmail - Use enums for common filters (date, sender type)
- GroupByIssueType - Pre-defined issue categories as enum
- AskUserForInput - Enum for question_type if structured
- CheckSOP - Return boolean + SOP_id, not full content
- RouteToSPOC - Enum for team/department options

2. New Joiner Onboarding Assistant

Submitted by: Aaron Pereira (Product, KSL)

Date: January 17, 2026

Problem Statement:

Onboarding a new joiner with set defined processes and a roadmap. It's a very vague process on whom to approach for asset allocation, HR policies, who to approach for which accesses, etc. Standard questions can be solved by feeding HR policies to the LLM for helping to solve basic queries and explaining policies that a new joiner may need to learn.

3. Speech-to-Text Analytics for Contact Center

Submitted by: Jimmy Joshi (Product, KSL)

Date: January 17, 2026

Problem Statement:

Contact centers generate large volumes of customer service calls that contain critical information such as customer issues, resolutions, decisions, risks, and follow-up actions. Currently, extracting these insights from call recordings or transcripts is manual, time-consuming, inconsistent, and prone to human error, leading to missed actions, poor visibility, and limited scalability.

Expected Capabilities:

The AI agent should analyze a call transcript and automatically produce: 1. Call Summary (2-3 concise sentences) 2. Key Discussion Points (clear bullet points) 3. Decisions Made 4. Action Items (task, owner, and deadline if mentioned) 5. Risks or Concerns Identified 6. Next Steps

Output Guidelines:

- Responses must be concise, structured, and professional
- The system must not infer or invent information
- If a specific item is not discussed in the call, explicitly state "Not mentioned"
- Use clear bullet points for readability and consistency

Business Outcome:

This solution will improve operational efficiency, ensure no critical customer commitments are missed, enable better compliance and quality monitoring, and provide leadership with actionable insights derived directly from customer conversations—at scale.

4. Tool Schemas: Query Intake & Resolution System

Submitted by: Jyotbir Lamba (Product, KSL)

Date: January 17, 2026

Overview:

Formal tool schemas covering: • Channel routing • Input normalization (text and voice) • Intent classification with enums and confidence thresholds • Knowledge base lookup • Auto-response handling • Smart escalation • Focus-time queue management (rule-based) The schemas are designed to be flat, enum-driven, and deterministic wherever possible to ensure token efficiency and predictable behavior.

Complete Tool Schemas:

1 ■■■ ChannelRouter

```
{
  "tool_name": "ChannelRouter",
  "description": "Identifies the source channel of the incoming query",
  "parameters": {
    "raw_input": {
      "type": "string",
      "required": true,
      "description": "Raw incoming payload from any channel"
    },
    "channel": {
      "type": "string",
      "enum": ["email", "whatsapp", "slack", "teams", "voice"],
      "required": true,
      "description": "Detected source channel"
    },
    "sender_id": {
      "type": "string",
      "required": true,
      "description": "Identifier of the sender (user or system)"
    }
  }
}
```

■ Deterministic

■ No LLM involvement

2 ■■■ TextNormalizer

```
{
  "tool_name": "TextNormalizer",
  "description": "Normalizes text-based inputs into clean plain text",
  "parameters": {
    "input_text": {
      "type": "string",
      "required": true,
      "description": "Raw email or message text"
    },
    "channel": {
      "type": "string",
      "enum": ["email", "whatsapp", "slack", "teams"],
      "required": true
    }
  }
}
```

■ Deterministic

■ Used only for text channels

3 ■■■ SpeechToText

```
{
  "tool_name": "SpeechToText",
  "description": "Converts voice calls into text",
  "parameters": {
    "audio_url": {
      "type": "string",
      "required": true,
      "description": "Location of recorded call audio"
    }
  }
}
```

```

    },
    "language": {
      "type": "string",
      "default": "en",
      "description": "Language of the call"
    }
  }
}

```

■ ML-based (non-LLM)

■ Output feeds into IntentClassifier

4 ■ IntentClassifier

```

(LLM)
{
  "tool_name": "IntentClassifier",
  "description": "Classifies the intent of a normalized query",
  "parameters": {
    "query_text": {
      "type": "string",
      "required": true,
      "description": "Normalized plain text query"
    },
    "channel": {
      "type": "string",
      "enum": ["email", "whatsapp", "slack", "teams", "voice"],
      "required": true
    },
    "intent": {
      "type": "string",
      "enum": [
        "SOP_CLARIFICATION",
        "PRODUCT_QUERY",
        "BUG_REPORT",
        "DATA_REQUEST",
        "ESCALATION",
        "GENERAL"
      ],
      "required": true
    },
    "confidence": {
      "type": "number",
      "required": true,
      "description": "Confidence score between 0 and 1"
    },
    "confidence_threshold": {
      "type": "number",
      "default": 0.8,
      "description": "Minimum confidence for auto-resolution"
    }
  }
}

```

■ Primary LLM usage point

■ Enum-constrained intents (token efficient)

5 ■ KnowledgeBaseLookup

```

{
  "tool_name": "KnowledgeBaseLookup",
  "description": "Searches SOP and product documentation for answers",
  "parameters": {
    "query_text": {
      "type": "string",
      "required": true
    },
    "intent": {
      "type": "string",
      "enum": ["SOP_CLARIFICATION", "PRODUCT_QUERY"],
      "required": true
    },
    "max_results": {
      "type": "number",
      "default": 3
    }
  }
}

```

■ Retrieval-only

■ No LLM required

6 ■ AutoResponder

```
{
  "tool_name": "AutoResponder",
  "description": "Sends standardized responses back to the originating channel",
  "parameters": {
    "response_text": {
      "type": "string",
      "required": true
    },
    "channel": {
      "type": "string",
      "enum": ["email", "whatsapp", "slack", "teams"],
      "required": true
    },
    "recipient_id": {
      "type": "string",
      "required": true
    }
  }
}
```

■ Deterministic

■ Can optionally use templated responses

7 ■ EscalationManager

(LLM-assisted summary)

```
{
  "tool_name": "EscalationManager",
  "description": "Routes unresolved or critical queries to PM/Engineer",
  "parameters": {
    "query_text": {
      "type": "string",
      "required": true
    },
    "intent": {
      "type": "string",
      "enum": ["BUG_REPORT", "ESCALATION", "DATA_REQUEST"],
      "required": true
    },
    "priority": {
      "type": "string",
      "enum": ["HIGH", "MEDIUM", "LOW"],
      "required": true
    },
    "summary": {
      "type": "string",
      "required": true,
      "description": "LLM-generated concise summary"
    },
    "assignee_role": {
      "type": "string",
      "enum": ["PM", "ENGINEER"],
      "required": true
    }
  }
}
```

■ LLM only for summarization

■ Routing remains deterministic

8 ■ FocusTimeQueue

(Rule-based, No LLM)

```
{
  "tool_name": "FocusTimeQueue",
  "description": "Queues or releases queries based on focus hours and criticality",
  "parameters": {
    "query_id": {
      "type": "string",
      "required": true
    },
    "is_focus_hours": {
      "type": "boolean",
      "required": true
    },
    "is_critical": {
      "type": "boolean",
      "required": true
    },
    "action": {
      "type": "string",
      "enum": ["QUEUE", "BYPASS"],
      "required": true
    }
  }
}
```

```
}  
}
```

- Correctly excludes LLM
- Maximum token savings

- Final Architecture Insight (Validation)

LLM usage limited to:

IntentClassifier

Escalation summary generation

Everything else deterministic or retrieval-based

Enums everywhere → low hallucination risk

Defaults reduce prompt size

Focus logic outside LLM → best practice

If you want next:

Token cost estimation per query

Prompt design for IntentClassifier

OpenAPI spec from these schemas

Java interface + DTO mapping

Comparison vs MCP / LangChain tools