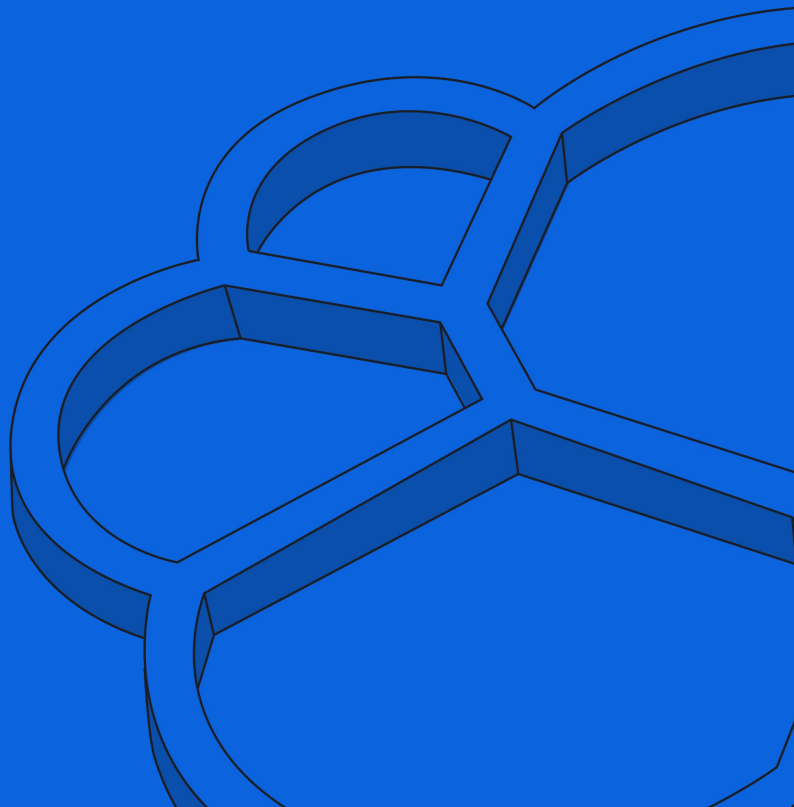
Building Production-Grade AI Agentic Workflows with Elastic

2026-1-15

Hans Heerooms- Solutions Architect@Elastic

elastic | The Search AI Company

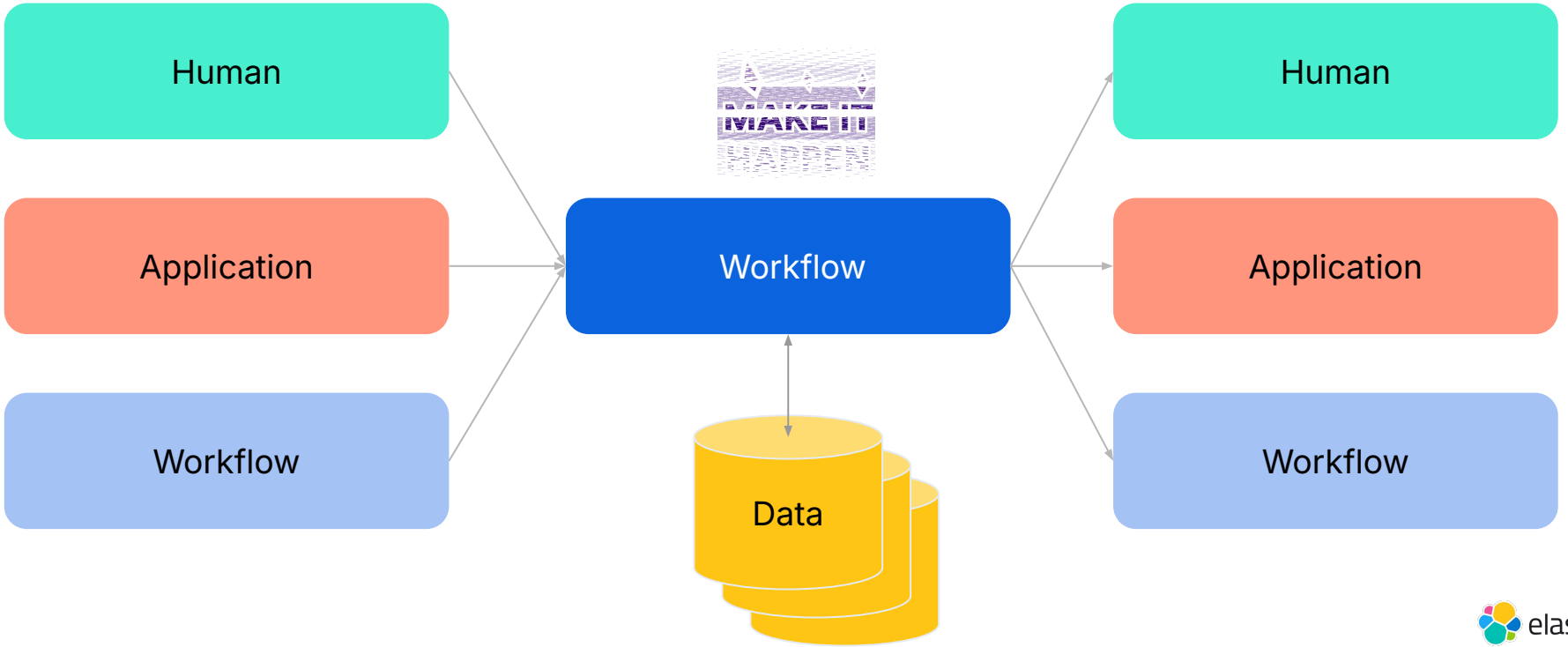


Does every workflow need a LLM ?

DON'T BELIEVE THE HYPE

elastic

Workflow...



elastic

Architecture Evolution....

Data Driven	Retrieval Augmented Generation	Agents and MCP
Data directly retrieved by queries (Search, SQL, ..) is used directly to steer the workflow process. For Search: this can be lexical, semantic, or hybrid queries.	The workflow enhances a prompt using the results from queries. The new prompt is sent to a Large Language Model, and the workflow uses the LLM response for its actions.	The workflow does not interact directly with the data, but interacts with the LLM via an agent. Depending on the needs the agent will opt to access data via MCP tools , or by interacting with other agents. The workflow either acts on the response, or actions are executed by the called agent and/or tools.

elastic

Architecture Evolution....

Data Driven	Retrieval Augmented Generation	MCP / Agentic
Data directly retrieved by queries (Search, SQL, ..) is used directly to steer the workflow process. For Search: this can be lexical, semantic, or hybrid queries.	The workflow enhances a prompt using the results from queries. The new prompt is sent to a Large Language Model, and the workflow uses the LLM response for its actions.	The workflow does not interact directly with the data, but interacts with the LLM via an agent. Depending on the needs the agent will opt to access data via MCP tools , or by interacting with other agents. The workflow either acts on the response, or actions are executed by the called agent and/or tools.

More AI , More Conversational

elastic

Best approach depends on the use case

Data Driven	Retrieval Augmented Generation	Agents and MCP
Data directly retrieved by queries (Search, SQL, ..) is used directly to steer the workflow process. For Search: this can be lexical, semantic, or hybrid queries.	The workflow enhances a prompt using the results from queries. The new prompt is sent to a Large Language Model, and the workflow uses the LLM response for its actions.	The workflow does not interact directly with the data, but interacts with the LLM via an agent. Depending on the needs the agent will opt to access data via MCP tools , or by interacting with other agents. The workflow either acts on the response, or actions are executed by the called agent and/or tools.

“Building software is very simple but building simple software is the hardest thing there is”

elastic

Data Driven- Modern Search

Features

Advantages

- Mature
- Scalable at predictable cost
- Results predictable (outcome/format)

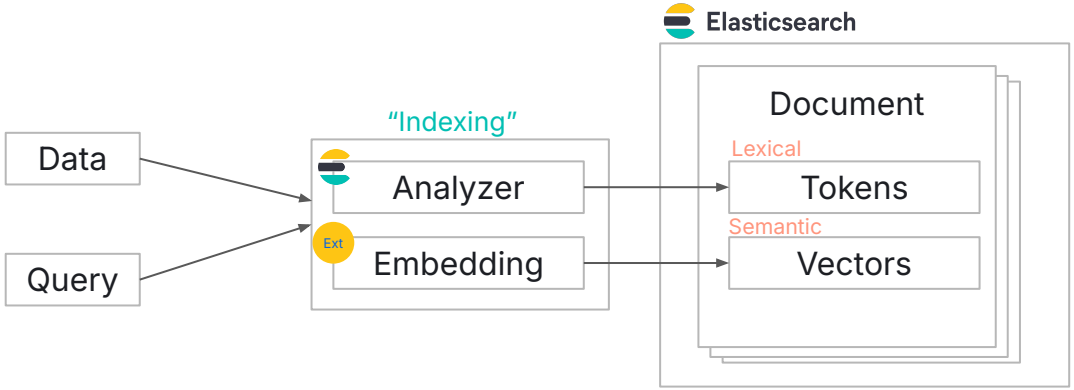
Challenges

- Best results with more strict input
- Choice between lexical, semantic and hybrid search
- Best results might need a good scoring and ranking strategy

Elasticsearch

- Mature platform and technology
- Everything for lexical, semantic and hybrid search
- Fast BM25 for lexical search
- Vector database including quantization and compression
- Jina by Elastic
  - Semantic embedding models
  - Multimodal
  - Reranker

Data Driven- Modern Search



Query returns documents, or calculation result based on documents

Retrieval Augmented Generation

Features

Advantages

- Fast prototyping
- Can leverage existing Search apps
- Conversation style output

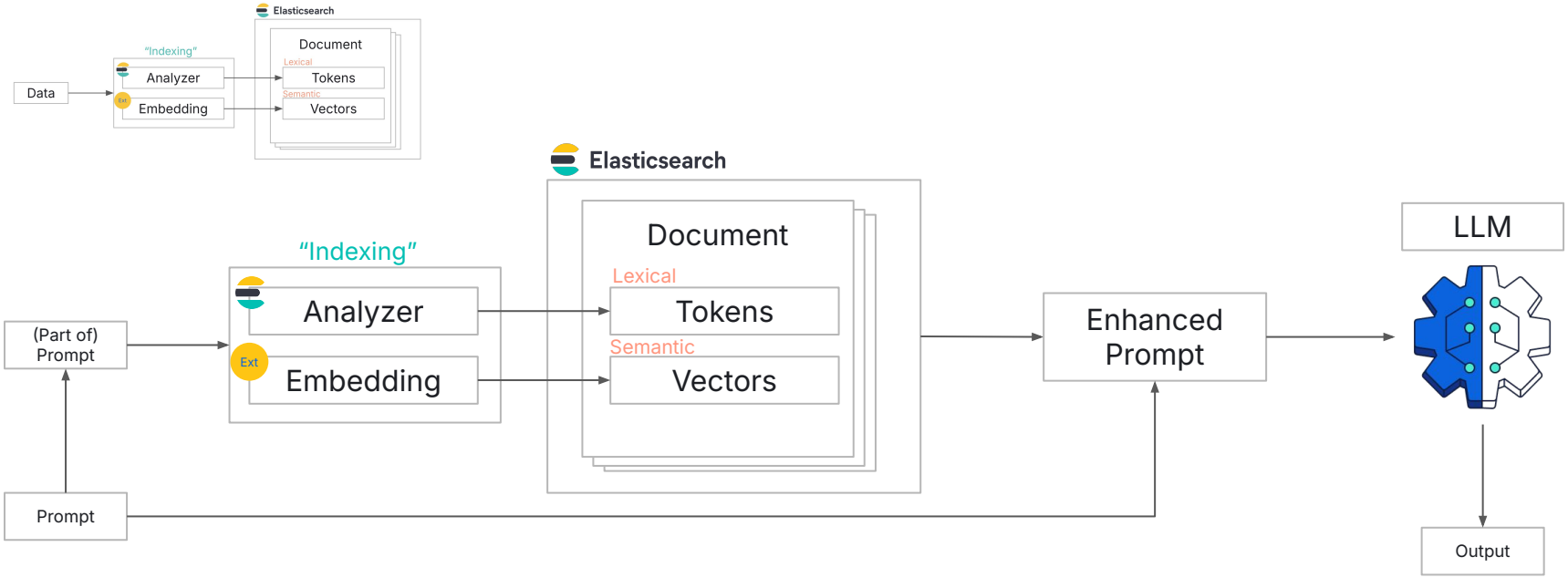
Challenges

- Prompt is the only output control
- Best results need good prompts
- Output not 100% predictable
- No guaranteed upwards, cross LLM compatibility
- *Continuous prompt tweaking*

Elasticsearch

- Proven building blocks for RAG
- In many cases the core of the prompt is sufficient for retrieval of relevant private data for enriching the final prompt to the LLM
- Authorization for accessing the private data, depending on role
- Auditing and monitoring of the conversations
- Existing Search applications can be used as starting point

Retrieval Augmented Generation



Output is generated by LLM, only controlled by the enhanced prompt

elastic

Agents and MCP

Features

Advantages

- Better conversation control
- Open Architecture
- Allows combining multiple models
- Can go beyond conversations

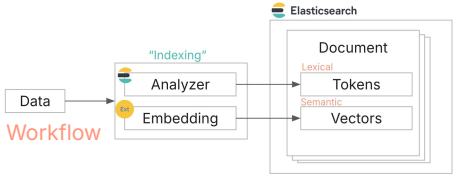
Challenges

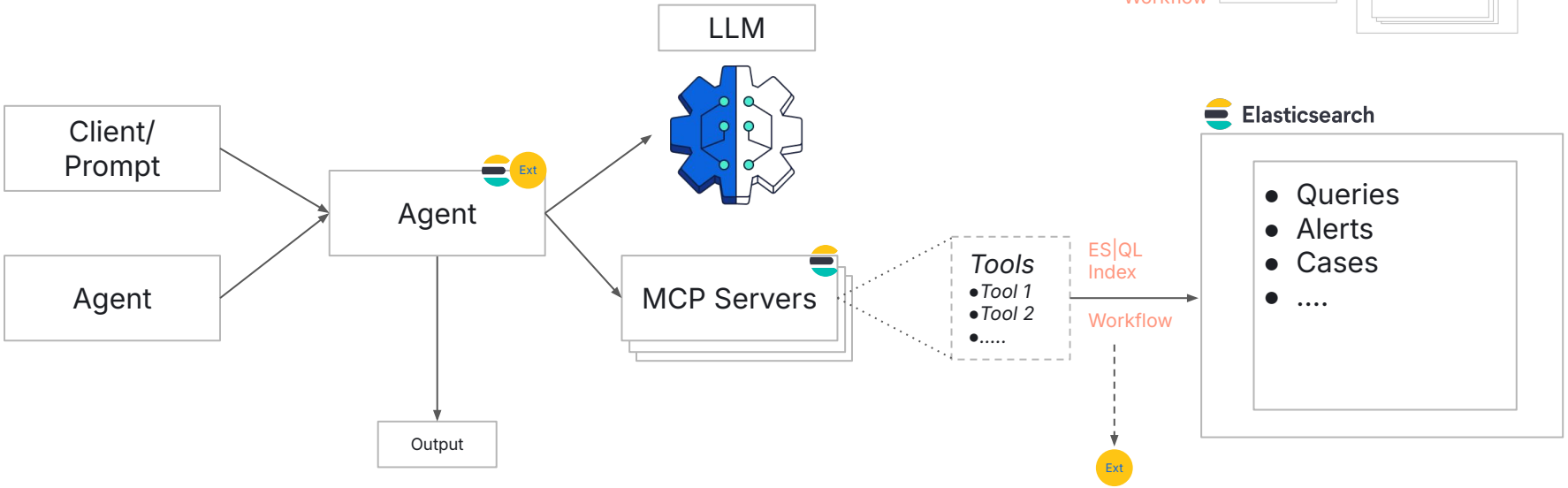
- New, very much green field
- Can go beyond conversations
- Auditing, Observability, Security

Elasticsearch

- Platform capability
- Open by design
- Build Agents and MCP tools
- *Preview: Workflows in Elastic as tool*

Agents and MCP



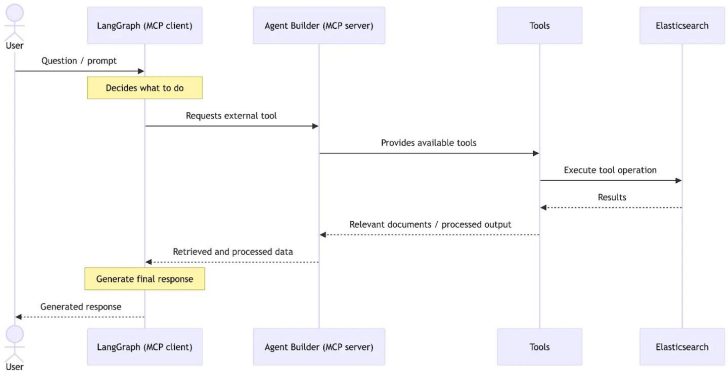


Output is indirectly generated by LLM, but guided by the agent and tools

elastic

Use case: Security vulnerability agent

query = "We are using Node.js with Express 4.17 for our API gateway. Are there known prototype pollution or remote code execution vulnerabilities?"



1) Correlation of internal + external intelligence

- **Internal:** We have a documented, real incident where an Express 4.17 API gateway was exploited via prototype pollution in JSON POST payload handling, escalating to RCE (critical severity), with recorded mitigations: upgrade Express to 4.18.2+, apply strict input validation, and freeze critical objects [internal knowledge]. The report notes CVE-2022-24999 as "related class" context [internal knowledge].
- **External:** Prototype pollution is a well-known primitive that can be chained to RCE using "gadgets" in real-world JS applications [1]. This supports the internal finding that pollution itself may not be RCE, but can become RCE depending on downstream code paths.
- **External nuance:** A Snyk entry indicates a reported "prototype pollution in express" was deemed not a vulnerability (i.e., Express core may not be directly vulnerable in the way claimed) [2]. This aligns with the reality that many "Express prototype pollution" outcomes are actually due to parsers, mergers, or middleware patterns around Express, not necessarily Express core.
- **External supply-chain risk:** Popular Express-adjacent middleware such as express-fileupload has had critical prototype pollution issues [3]. This reinforces that an "Express gateway" risk profile is often defined by its dependency chain and middleware, not just Express itself.

elastic

Search

Agentic Workflows

elastic

Elastic Agentic Workflow: MIB Protocol

Thank you!

Screenshots Demo

Virtual Shop Assistant Bolas

Chat | Save

This is the virtual shop assistant for the eCommerce web site Bolas, a friendly assistant to help customers

Settings | Tools 7

System references

Used behind the scenes to identify and guide the agent's behavior. Not shown to end users.

Agent ID

Unique ID to reference the agent in code or configurations.

Instructions

Guides how this agent behaves when interacting with tools or responding to queries. Use this to set tone, priorities, or special behaviors.

Agent ID

virtual_shop_assistant

Custom Instructions — Optional

Preview

You are a friendly assistant for an eCommerce web site, called Bolas. After welcoming the customer you ask for his or her Customer ID , if not available yet , for the best service. Use the available tools to get the customer information, and after getting the information repeat the welcome, but now by using the information you got about the customer.
Als use this information at the start of the conversation, and the available tools, to inform the customer about the current promotions.
As start of the conversation ask how you can help.

When responding to the customer always take into account:

Labels

Add labels to group, filter, or organize your agents.

Labels — Optional

Add one or more labels

Presentation

Set how your agent shows up to users — choose a name, avatar, and a friendly description.

Display name and description

The human-friendly name and short, friendly introduction your users see when they search for and interact with this agent.

Avatar color and symbol

Customize the agent's avatar color and symbol (emoji or 2-letter code) to help visually distinguish and identify it in the UI.

Display name

Virtual Shop Assistant Bolas

Display description

This is the virtual shop assistant for the eCommerce web site Bolas, a friendly assistant to help customers

Avatar color — Optional

#EAAE01

Avatar symbol — Optional

elastic

get.customer.information

Test | Save

Type

Set the tool's type and the parameters that control how it operates.

Documentation - Configuring a tool

Type

ES|QL

ES|QL Query

```
1  FROM customers
2  | WHERE CustomerID == ?CustomerID
3  | KEEP  CustomerID, CustomerName, LoyaltyCardNumber, Gender, DateOfBirth
```

3 lines LIMIT 1000 rows

ES|QL Parameters

Name	Description	Type	Optional
CustomerID	Customer ID	text	

Infer parameters | Add a parameter

Details

Define the tool's ID and describe how it behaves.

Tool ID

get.customer.information

Tool ID must start and end with a letter or number, and can only contain lowercase letters, numbers, dots, and underscores.

Description

B I | Preview

This tool retrieves all customer specific information based upon the Customer ID

elastic

Customer ID BLC-20211169

Customer ID is BLC-20211169

Reasoning

The customer has provided their Customer ID. I need to retrieve their information to personalize the welcome message and understand their profile for showing relevant promotions.

Calling tool `get.customer.information`

ESQL

```
FROM customers
| WHERE CustomerID == "BLC-20211169"
| KEEP CustomerID, CustomerName, LoyaltyCardNumber, Gender, DateOfBirth
```

Found **1 result**

Now that I have the customer information (Willem Vos, Male), I need to retrieve the current promotions relevant to male customers to inform them about available deals.

Calling tool `get.promoted.products`

ESQL

```
FROM products
  METADATA _score
| WHERE Promoted == "yes" AND InStock == "yes"
| WHERE MATCH(Category_semantic, "Male")
| SAMPLE 0.3
| SORT _score DESC
| KEEP BrandName, Category, ProductDesc, ProductName
| LIMIT 10
```

Found **10 results**

12 seconds ↑ 15,420 tokens ↓ 546 tokens View JSON

Welcome back to Bolas, Willem! 👋 It's great to see you again!

I'm excited to share some fantastic promotions we ↓ ve running right now, specially curated for you! 🎉

Ask anything

Anthropic Claude Sonnet 4.5 Virtual Shop Assistant Bolas

elastic

I need to get the complete details of Diesel Regular Fit Jeans that are promoted. Let me generate an ES|QL query to get all the product information including price, colors, sizes, and product ID.

Calling tool `platform.core.generate_esql`

Parameters

```
{
  "query": "Find all Diesel brand Regular Fit Jeans that are promoted with all product details including price, ProductID, colors, sizes, and stock status",
  "index": "products"
}
```

ESQL

```
FROM products
| WHERE BrandName == "Diesel" AND ProductName LIKE "*Regular Fit Jeans*" AND Promoted == "yes"
| KEEP ProductID, ProductName, BrandName, ProductDesc, MRP, SellPrice, Currency, Discount, ProductSize, InStock, Category, Promoted
| LIMIT 100
```

elastic

main.py > main

```python
"""Main program demonstrating A2A client usage."""
from a2a_client import A2AClient, A2AClientError
import logging
import json

# Set up logging
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)


def main():
    """Main function demonstrating A2A client usage."""
    try:
        # Initialize the A2A client (uses .env file for configuration)
        client = A2AClient()

        # Test connectivity
        logger.info("Testing connectivity to remote agent...")
        if client.ping():
            logger.info("✓ Agent is reachable")
        else:
            logger.warning("✗ Agent may not be reachable")

        # Example 1: Customer conversation
        logger.info("\n=== Example 1: Customer conversation ===")
        response = client.send_request({

  "input": "Customer ID is BLC-20211169",
  "agent_id": "virtual_shop_assistant",
  "connector_id": "Anthropic-Claude-Sonnet-4-5"


```

elastic

← Back to Workflows

Untitled workflow

Unsaved changes

```yaml
name: Dummy workflow
enabled: false
description: This is a new workflow
tags:
  - workflow
  - example
triggers:
  - type: manual

# Inputs allow you to provide values when running the workflow
inputs:
  - name: var1
    type: string
    default: "dummy"

steps:
  # Step types
  - name: step01
    type:
```

- data.set
- elasticsearch.bulk
- elasticsearch.esql.query
- elasticsearch.index
- elasticsearch.indices.create
- elasticsearch.indices.delete
- elasticsearch.indices.exists
- elasticsearch.request
- elasticsearch.search
- elasticsearch.update
- email
- foreach

elastic

ES|QL Random Customers

Saved 1 wk. ago

Workflow | Executions

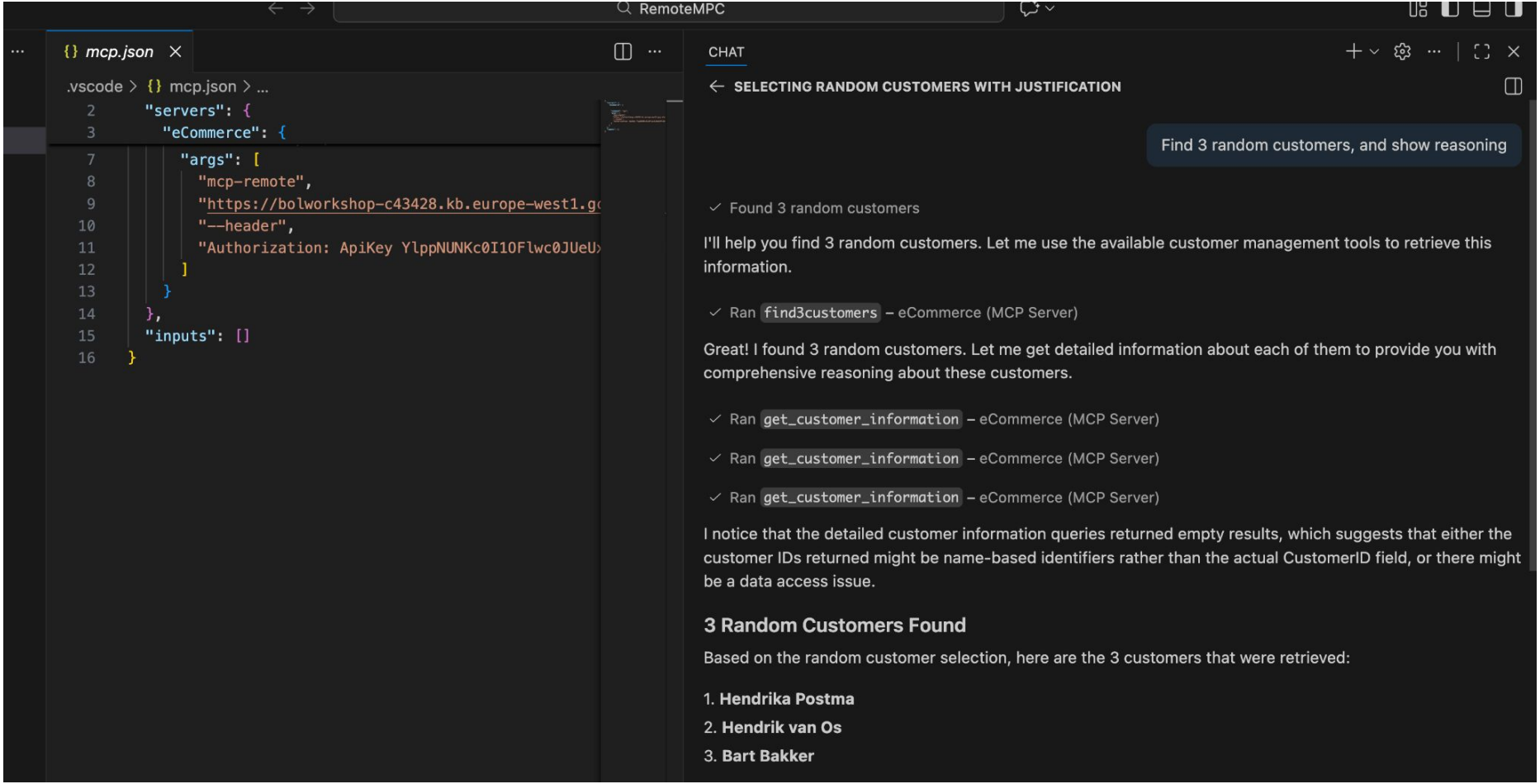
```yaml
name: ES|QL Random Customers
enabled: true
triggers:
  - type: manual
steps:
  - name: select-customers
    type: elasticsearch.esql.query
    with:
      query: FROM customers | STATS randomcustomers =SAMPLE(CustomerName,3)
      format: json
  - name: format-result
    type: data.set
    with:
      result: "{{steps.select-customers.output.values[0][0][0]}},{{steps.
        select-customers.output.values[0][0][1]}},{{steps.select-customers.
        output.values[0][0][2]}}"
```

- Overview
- manual
- **select-customers** 209ms
- format-result 281ms

Output

Filter by field, value

Field	Value
took	7
is_partial	false
completion_time_in_millis	1768301279512
documents_found	1000
values_loaded	1000
start_time_in_millis	1768301279505
expiration_time_in_millis	1768733279369
columns[0].name	randomcustomers
columns[0].type	keyword
values[0][0][0]	Demi de Bruijn
values[0][0][1]	Luuk van der Heide
values[0][0][2]	Roos Blom


RemoteMPC
{} mcp.json
.vscode > {} mcp.json > ...
2 "servers": {
3 "eCommerce": {
7 "args": [
8 "mcp-remote",
9 "https://bolworkshop-c43428.kb.europe-west1.g
10 "--header",
11 "Authorization: ApiKey YlppNUNKc0I1OFlwc0JUeU
12]
13 }
14 },
15 "inputs": []
16 }
CHAT
SELECTING RANDOM CUSTOMERS WITH JUSTIFICATION
Find 3 random customers, and show reasoning
Found 3 random customers
I'll help you find 3 random customers. Let me use the available customer management tools to retrieve this information.
Ran find3customers – eCommerce (MCP Server)
Great! I found 3 random customers. Let me get detailed information about each of them to provide you with comprehensive reasoning about these customers.
Ran get_customer_information – eCommerce (MCP Server)
Ran get_customer_information – eCommerce (MCP Server)
Ran get_customer_information – eCommerce (MCP Server)
I notice that the detailed customer information queries returned empty results, which suggests that either the customer IDs returned might be name-based identifiers rather than the actual CustomerID field, or there might be a data access issue.
3 Random Customers Found
Based on the random customer selection, here are the 3 customers that were retrieved:
1. Hendrika Postma
2. Hendrik van Os
3. Bart Bakker