Capgemini

Architecting AI agents in the public sector

A detailed guide to game-changing multi-agent platforms for technical leads

What's inside

Introduction

The public sector is facing a significant skilled labor shortage, which is worsening due to ongoing demographic shifts. To address this, increasing automation is essential, enabling public organizations to handle tasks more efficiently with fewer people.

AI agents sound futuristic – and they are. Think of fully automated processes and intelligent assistants that solve problems on their own. That is exactly what AI agents bring to the table. So why should you care? Because AI agents take your automation to a whole new level.

Automation without AI is like using a typewriter in the age of computers; it is better than handwriting, but it misses out on the transformative potential and efficiency that modern technology offers.

Modern automation platforms provide a strong foundation: you build workflows, transfer data, and automate repetitive tasks. But the real transformation happens when you integrate AI agents into those workflows – agents that analyze data, make decisions, and optimize processes in real time with a level of precision that is impossible to achieve manually.

Research recent by the Capgemini Research Institute suggests that public sector organizations are awake to this potential: 90% of those surveyed plan to implement agentic AI in the next 2-3 years.[1] But the field is evolving rapidly. The concept of agentic AI is still emerging and often complex. Many solutions are experimental, fragmented, or deeply technical – which makes orientation difficult. That is why we focus on clarity in this point of view: We illustrate typical architectures, compare core technologies, and show how AI agents can work in practice, across sectors and scenarios.

While there is a growing ecosystem of commercial platforms – from open source to enterprise-grade – our goal here is to show what is possible and not to describe single solutions. From citizen interactions with government services to internal workflows and non-public use cases, the agentic approach reshapes how we think about automation.

In this point of view, we will shed light on what makes AI agents so powerful, how they work, and how you can smartly integrate them into your automation strategy. Whether it is for communication, data processing, or creative tasks, the possibilities they offer are endless.

We will explore real-world use cases across sectors, including the public sector, where AI agents unlock new potential in citizen communication and administrative efficiency. With rising administrative workloads, limited staff resources, and growing expectations for digital services, AI agents offer concrete solutions: faster response times, scalable service delivery, and intelligent case handling.

Public sector environments differ from enterprise use: They require maximum data sovereignty, transparent decision logic, and integration into existing systems and responsibilities. Unlike private companies, public administrations must also make sure every automated decision is legally accountable and explainable. Their fragmented IT systems require AI to integrate smoothly, and citizen data must be protected under national laws, making public sector AI deployment far more complex than in corporate settings.

This point of view is designed to support public sector technical leads in navigating these complexities. It offers clarity, practical guidance, and architectural insights to help you lead successful AI-driven automation initiatives.

[1] Capgemini. Data foundations for government: From AI ambition to action. https://www.capgemini.com/insights/research-library/data-mastery-in-government/. 2025. (Visited 28.07.2025).

Why AI agents matter

Supercharging processes

What happens when you combine automation, artificial intelligence, and autonomous agents? Let us break it down:

- Automation refers to systems that follow predefined rules to complete tasks without manual effort.
- AI enhances automation by enabling systems to learn from data, recognize patterns, and make data-driven decisions.
- AI agents go a step further – they not only execute tasks but also analyze the context, adapt their behavior, and continuously optimize outcomes.

When these three elements work together, processes do not just repeat; they evolve. They learn, adapt, and continuously improve. That is the real supercharge: systems that do not just follow instructions but anticipate needs and think smartly.

In the public sector, this is especially important for sustainability and performance. Generative AI (GenAI) – including foundation models and on-premise deployments – allows you to implement intelligent automation that meets the highest standards of security, transparency, and efficiency. This means not only faster processes but also better public services.

What are foundation models?

Foundation models are large-scale AI systems trained on vast and diverse datasets, designed to be adaptable across a wide range of tasks. Their main subcategories include:

- **Large Language Models** (LLMs) for understanding and generating text for broad capabilities.
- **Small Language Models** (SLMs) for lightweight VLMs (Vision Language Models) for integrating visual and textual data.
- Other specialized types like Audio Foundation Models and Multimodal Models that combine multiple data modalities.

Language, data and context: the keys to automation

Language is not just English, German, or Chinese. It also comes in the form of application programming interfaces (APIs), machine languages, robot commands, or control sequences. Yet it is not just the language itself that unlocks true automation; it is the combination of language, data, and context.

Modern artificial intelligence can understand, process, and apply these different forms of language in context. It extracts meaning from data and adapts its actions based on what is relevant here and now. That is how you turn language into real-world action: by weaving it together with data and context to make processes smarter and more responsive.

Large Language Models like Microsoft's Azure OpenAI Service, OpenAI's GPT-5, Google's Gemini, Amazon's Bedrock, and Mistral AI's open-source models are not limited to natural language. They are equally capable of interpreting programming code, database queries, API calls, and much more.

When a model cannot only communicate in human language but also operates in machine and control languages, its functionality expands dramatically.

At that point, it is no longer just about writing text or holding conversations. It is about controlling systems, triggering processes, analyzing data, and making decisions – all in real time.

This multilingual capability is the foundation of AI agents. Because they understand various types of languages – not just natural ones – they can communicate with different systems, process data, and orchestrate complex workflows autonomously. Yet these agents do not replace humans. They enhance human capabilities, creating what we call "human-AI chemistry".

What is an AI agent?

An AI agent is an intelligent software system that autonomously perceives its environment, reasons about it, and acts to achieve specific goals. It combines decision logic, dynamic adaptability, and domain-specific intelligence to operate within complex, evolving environments. Hence, an AI agent understands tasks, executes them, and continuously improves its performance. These agents do not just follow commands – they make decisions on their own, delegate tasks among each other, and exchange intermediate results.

At the core of every AI agent are three fundamental capabilities:[2]

Perception: The agent gathers and analyzes data from various sources – including APIs, databases, log files, text, images, or real-time sensor input.

Processing: Using algorithms and models – especially Large Language Models (LLMs) – the agent evaluates and processes this data. This goes far beyond basic pattern recognition and includes logical reasoning, contextual understanding, and creative problem-solving.

Action: Based on its analysis, the agent performs actions autonomously. This could involve sending messages, triggering workflows, or even developing new strategies.

The real game-changer is the agent's ability to understand and apply different "languages". APIs, code, database queries, and natural language become tools it can actively use. This capability is often referred to as "function calling" – the targeted activation of specific functions or processes using either language or code inputs.

[2] Dave Andre. Die Anatomie Weines KI-Agenten: Wahrnehmung, Kognition und Handlung. https://www.allaboutai.com/de-de/ki-agenten/anatomie/. 2025. (Visited 05.04.2025).

The anatomy of an AI agent

To complete tasks autonomously, an AI agent calls upon several interlinked components working together. Figure 1 illustrates the anatomy of an AI agent.

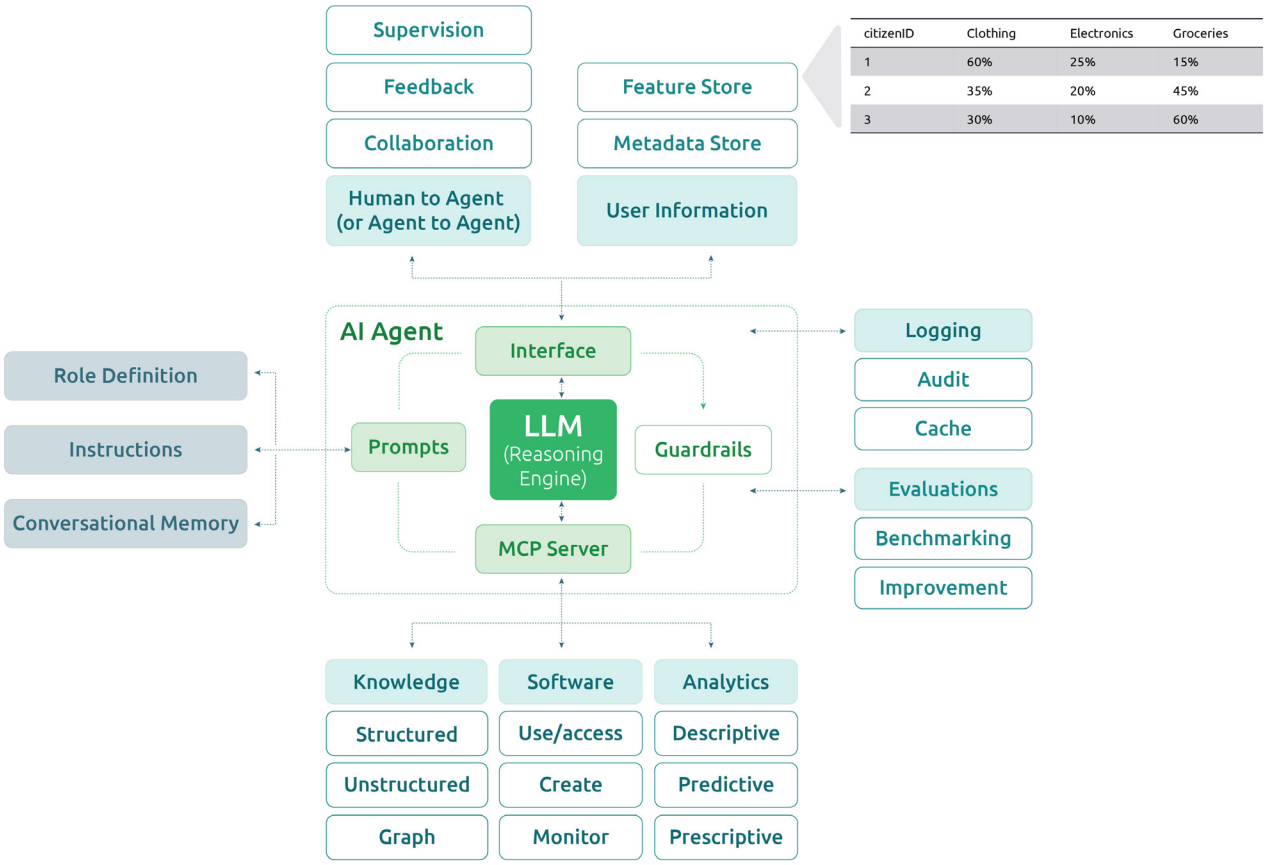


Figure 1: The anatomy of an AI agent [3,4]

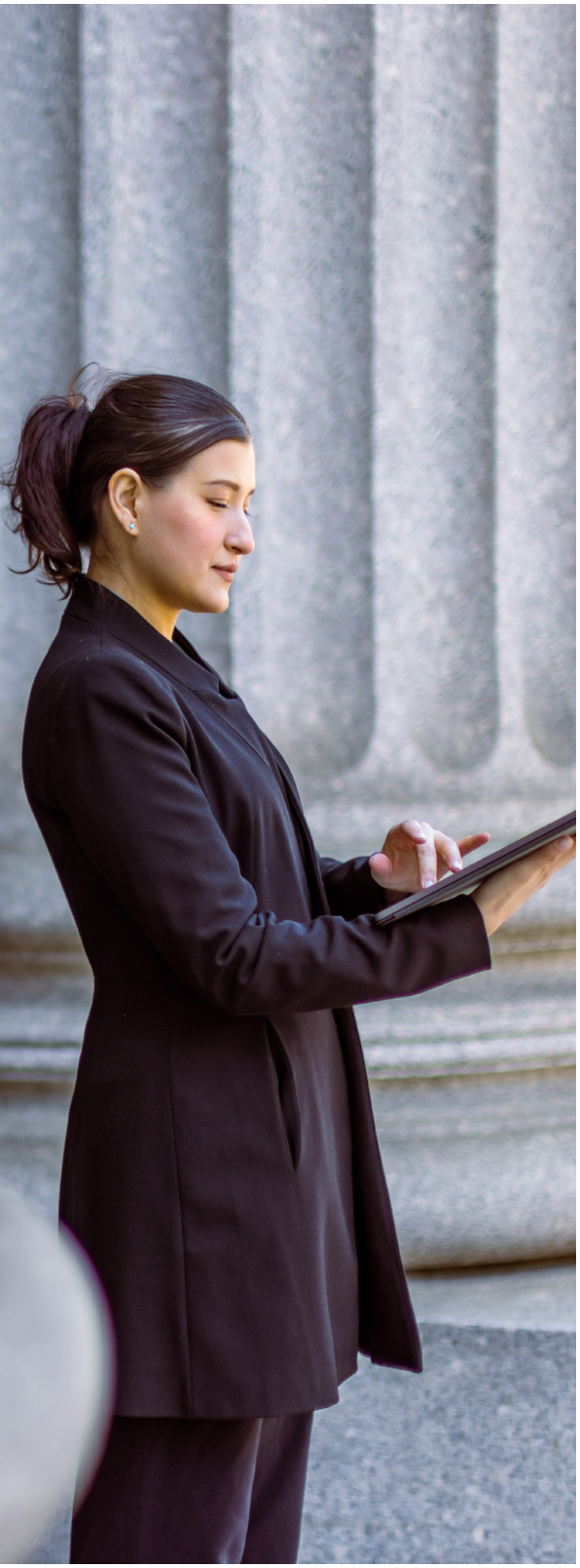
Here is how they function:[3]

An **AI agent** perceives its environment and captures relevant data such as text, speech, images, or structured information. Its knowledge base includes stored general knowledge, domain-specific knowledge, and specialized knowledge acquired over time. Agents dynamically expand their **knowledge base** through interaction and learning.

Based on the perceived data and the available knowledge, the agent makes decisions using rule-based systems, machine learning, or neural networks. It then executes actions – such as calling APIs, querying data, or interacting with **users** – by leveraging **tools** through interfaces like API endpoints, user interfaces, or other communication channels.

One key enabler is **MCP (Model Context Protocol)** – an open protocol that connects enterprise data to AI systems in a maintainable and scalable way.[4] By offering standardized APIs, the MCP server acts as a broker between agents and tools. This standardization is critical: it allows different agents to access the same tools without

[3] AWS. Was sind KI-Agenten? https://aws.amazon.com/de/what-is/ai-agents/. 2023. (Visited 06.04.2025).
[4] Amos Gyamfi. The Top 7 MCP-Supported AI Frameworks. https://medium.com/@amosgyamfi/the-top-7-mcp-supported-ai-frameworks-a8e5030c87ab. Apr. 2025. (Visited 06.04.2025).



requiring custom integration. It also simplifies the construction of complex workflows and significantly eases the connection between agents and their tools.

Agents can use these standardized APIs to trigger specific functions, query databases, send messages, or interact with web services. More advanced agents can even generate and orchestrate additional agents dynamically – representing the highest level of agent intelligence (see “Level 5: fully autonomous”, page 8).

All AI agents are capable of learning from experience and adapting to changing conditions, which makes them improve continuously over time. This improvement is driven by feedback and oversight obtained through supervised, unsupervised, or reinforcement learning methods.

Handling user information

Another crucial element of AI agents is how they handle user information. Two key components here are metadata stores and feature stores, which help agents better understand context and generate personalized responses.

Metadata Store: Stores general information about users or processes, such as user activities, interaction history, or configuration settings.

Feature Store: Manages structured attributes in the form of tables. In the public sector, this could mean tracking how a citizen typically interacts with government services – for example: walk-in = 55%, telephone = 30%, online portal = 15%. Based on this behavioral profile, agents can prioritize communication channels, personalize reminders, or even pre-select the most effective service path.

Technically, each individual (typically referenced by a citizenID or caseID) is linked to a dynamic profile. The agent continuously updates these values, detects usage patterns and preferences, and uses them to generate tailored responses or guide process flows more efficiently.

Another typical example is from e-commerce: For each customer (typically referenced by a customerID), the agent tracks preferences – for example, clothing = 60%, electronics = 25%, groceries = 15%. The agent dynamically updates these values, detects patterns and preferences, and uses them to generate personalized recommendations or targeted marketing content.

The evolution of AI: levels of autonomy in agentic AI

The autonomy of AI agents can be divided into levels of increasing maturity, as Figure 2 shows.

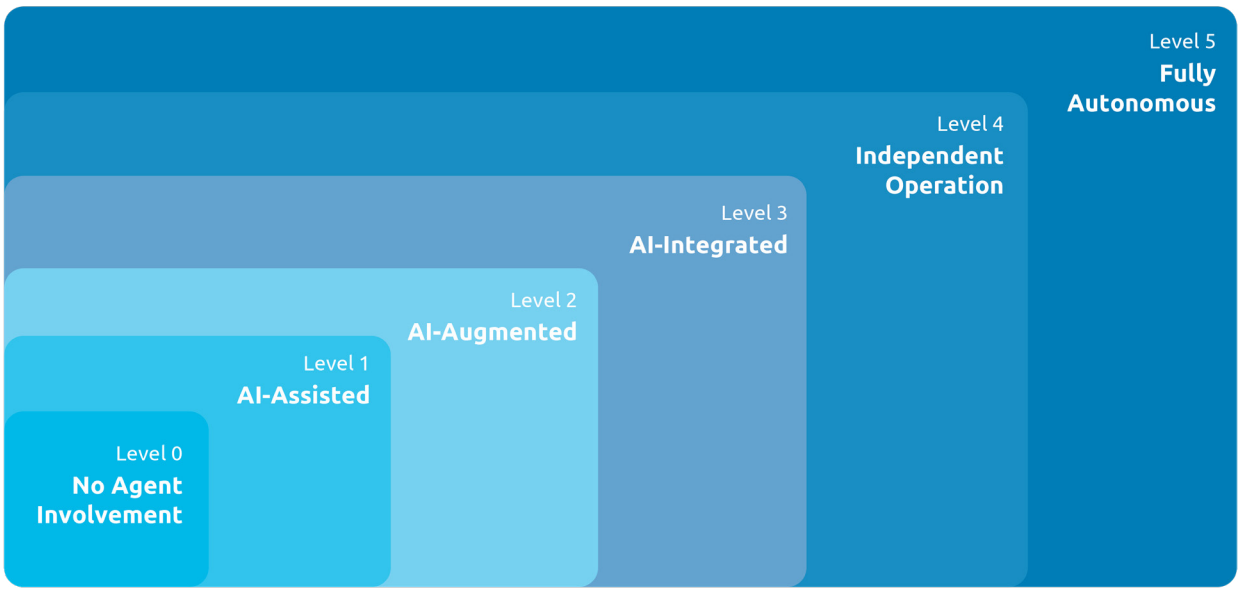


Figure 2: Six levels of AI maturity

These levels range from Level 0 (no AI involvement) to Level 5 (fully autonomous). Each level represents increasing sophistication, starting with basic AI assistance, advancing through decision support and process integration, and culminating in independent, collaborative AI agents capable of self-improvement.

Level 0: no agent involvement

At Level 0, there is no AI involvement at all. All tasks and decisions are handled entirely by humans, without automation or AI assistance.

Level 1: AI-assisted

Level 1 introduces AI-assisted capabilities, where basic rule-based tools support predefined workflows. While automation begins to play a role, humans remain heavily involved in all decision-making processes.

Level 2: AI-augmented

Moving to Level 2, AI becomes decision-augmenting, offering recommendations and insights that enhance human judgment. Although AI contributes to optimizing workflows and improving outcomes, humans still retain full control over decisions.

Level 3: AI-integrated

At Level 3, AI is integrated into processes, with semi-autonomous agents managing more complex and cross-functional tasks. Human oversight is reduced, as routine operations are increasingly handled by AI systems.

Level 4: independent operation

Level 4 marks the shift to independent AI operation, where multiple AI agents collaborate in real time to coordinate tasks and make decisions. This is often referred to as “swarms”, in which agents function as an intelligent collective. Various frameworks, such as CrewAI, Microsoft AutoGen, LangGraph[5] and OpenAI’s Agents SDK[6] – which replaces OpenAI Swarm[7] – enable the implementation of swarms. Human involvement is minimal, limited mostly to strategic oversight and intervention in exceptional cases.

Level 5: fully autonomous

Finally, Level 5 represents full autonomy, where AI systems operate as self-evolving ecosystems. These agents not only manage business functions independently but also learn and adapt their strategies over time. Human input is only required for high-level governance, compliance, and ethical oversight.

[5] LangChain. LangGraph. https://github.com/langchain-ai/langgraph. 2024. (Visited 06.04.2025).
[6] **OpenAI. OpenAI Agents SDK.** https://github.com/openai/openai-agents-python. 2025. (Visited 11.04.2025).
[7] OpenAI. Swarm. https://github.com/openai/openai-agents-python. 2024. (Visited 06.04.2025).

Case study

Automating ticket creation with context awareness

What practical value does a multi-agent architecture deliver in real-world applications?

A compelling example comes from the German Federal Employment Agency (Bundesagentur für Arbeit, BA). In collaboration with Capgemini, they implemented AI agents to automate IT service ticket creation within ALLEGRO, an internal system supporting social benefit processes for over 40,000 employees.

Rather than relying on rigid, rule-based automation, the solution uses a team of AI agents that work together to transform change requests and user stories into structured Jira tickets. These agents extract relevant information, break down tasks, generate complete tickets, and review outputs for consistency and duplicates – all within the agency’s infrastructure and in full compliance with data protection standards.

This approach not only streamlines operations and improves output quality, but also demonstrates how AI agents can support – rather than replace – public sector employees by taking over repetitive tasks and allowing them to focus on more complex, value-adding responsibilities.

Capgemini. KI-Agenten für die Verwaltung: Wie die Bundesagentur für Arbeit große Sprachmodelle einsetzt. https://www.capgemini.com/de-de/news/kundenprojekte/ki-agenten-fuer-die-verwaltung/. 2025. (Visited 28.07.2025)

Multi-agent architectures

Multi-agent architectures integrate several specialized agents into a coordinated system.[8] These systems consist of autonomous agents that take on different tasks, communicate with each other, and handle complex workflows. One particularly exciting aspect is the combination of language models with specialized tools and sub-agents, enabling intelligent orchestration across diverse functions.

This is precisely where frameworks like LangChain come into play. LangChain exemplifies the multi-agent approach by allowing the creation of agent chains. These are hierarchically structured workflows in which different agents perform specific tasks such as information retrieval, data processing, or decision-making. The flexible configuration of agent chains allows for the implementation of complex scenarios far beyond simple question and answer interactions.[9]

The design of a multi-agent architecture varies depending on the use case. Figure 3 shows the different possibilities:

Single agent: One agent handles all tasks. Technically, this is not a multi-agent setup, but it serves as the foundational building block for more complex systems. This approach is simple but often inefficient in complex scenarios, as the agent must manage every process on its own.

Network architecture: Multiple, equal-status agents work in parallel and share information. This setup is flexible and scalable but can lead to inconsistencies in data processing.

Supervisor architecture: A central agent coordinates multiple subordinate agents. This model is particularly effective for delegating tasks to specialized agents and aggregating their results efficiently.

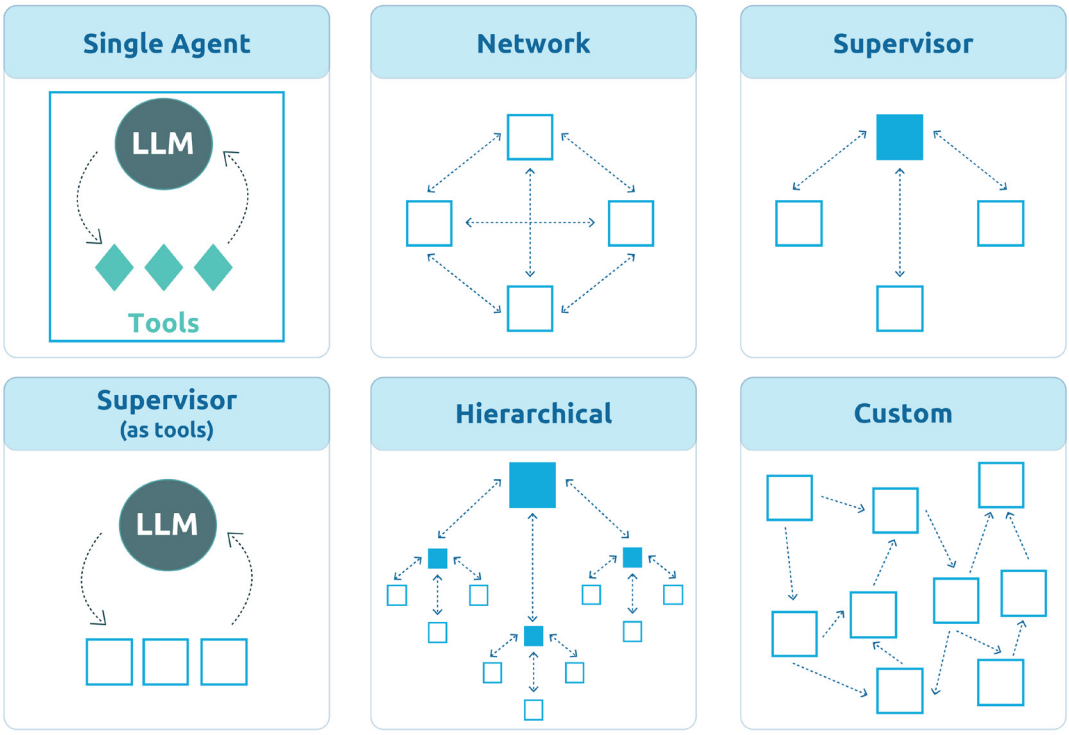


Figure 3: Overview of multi-agent architectures[8]

[8] LangChain. Multi-agent Systems. https://langchain-ai.github.io/langgraph/concepts/multi_agent/. 2025. (Visited 06.04.2025).
[9] n8n Docs. Demonstration of key differences between agents and chains. https://docs.n8n.io/advanced-ai/examples/agent-chain-comparison/. 2024. (Visited 06.04.2025).

Supervisor as tools: In this model, specialized agents function as tools or modules that are invoked by a main agent. This approach is well-suited for modular systems with clearly defined functional domains.

Hierarchical: Multiple layers of agents are organized in a pyramid structure. Higher-level agents delegate tasks to lower levels, making this architecture especially efficient for complex processes with many subcomponents.

Custom: Tailor-made architecture specifically designed for the needs of a particular use case. This model offers maximum flexibility but also requires the highest development effort.

A new dimension of agent autonomy in multi-agent architectures is emerging through agent-to-agent (A2A) communication protocols. These enable AI agents to coordinate tasks without human mediation. A notable example is Google's A2A protocol, which facilitates structured, scalable interactions between autonomous agents – an excellent addition for building multi-agent systems.

Collaboration in agent spaces across organizations

Imagine a shared agent environment in which agents operated by different public organizations can interact and coordinate tasks seamlessly across boundaries.

By automating routine processes, these agents relieve both citizens and public employees from manual work, freeing up time and resources for more meaningful activities.

Example 1: arranging a school trip

You are a vocational schoolteacher planning a one-week educational trip to Copenhagen for your class as part of an EU-funded project. You task the AI agents with a clear goal:

"Plan an educational trip for 20 students to Copenhagen from October 12 to 19, 2025. Include travel by train from Berlin, accommodation in a hostel or youth center, and at least two educational activities per day (e.g. visiting a Danish vocational school, a technology museum, or a startup tour). Total cost including entrance fees, meals, tickets, accommodation, and train travel: no more than €9,000. Eligible for Erasmus+ funding."

As Figure 4 shows, several specialized agents within Google Agentspace work together seamlessly in this scenario:

Research Agent: Gathers information on educational institutions, public programs, and EU partners.

Traffic Agent: Checks suitable train connections and group travel offers using mobility APIs.

Accommodation Agent: Finds available lodging near the destination and compares prices and locations.

Funding Agent: Calculates the total cost, verifies funding eligibility under Erasmus+ guidelines, and generates the application.

Program Planning Agent: Assembles suitable daily educational activities.

All results are collected by the Orchestrator Agent, which reconciles schedules, availability, and budget constraints to determine the optimal plan. Finally, the Booking Agent handles reservations, generates invoices, and compiles a full travel dossier for administrative or funding purposes.

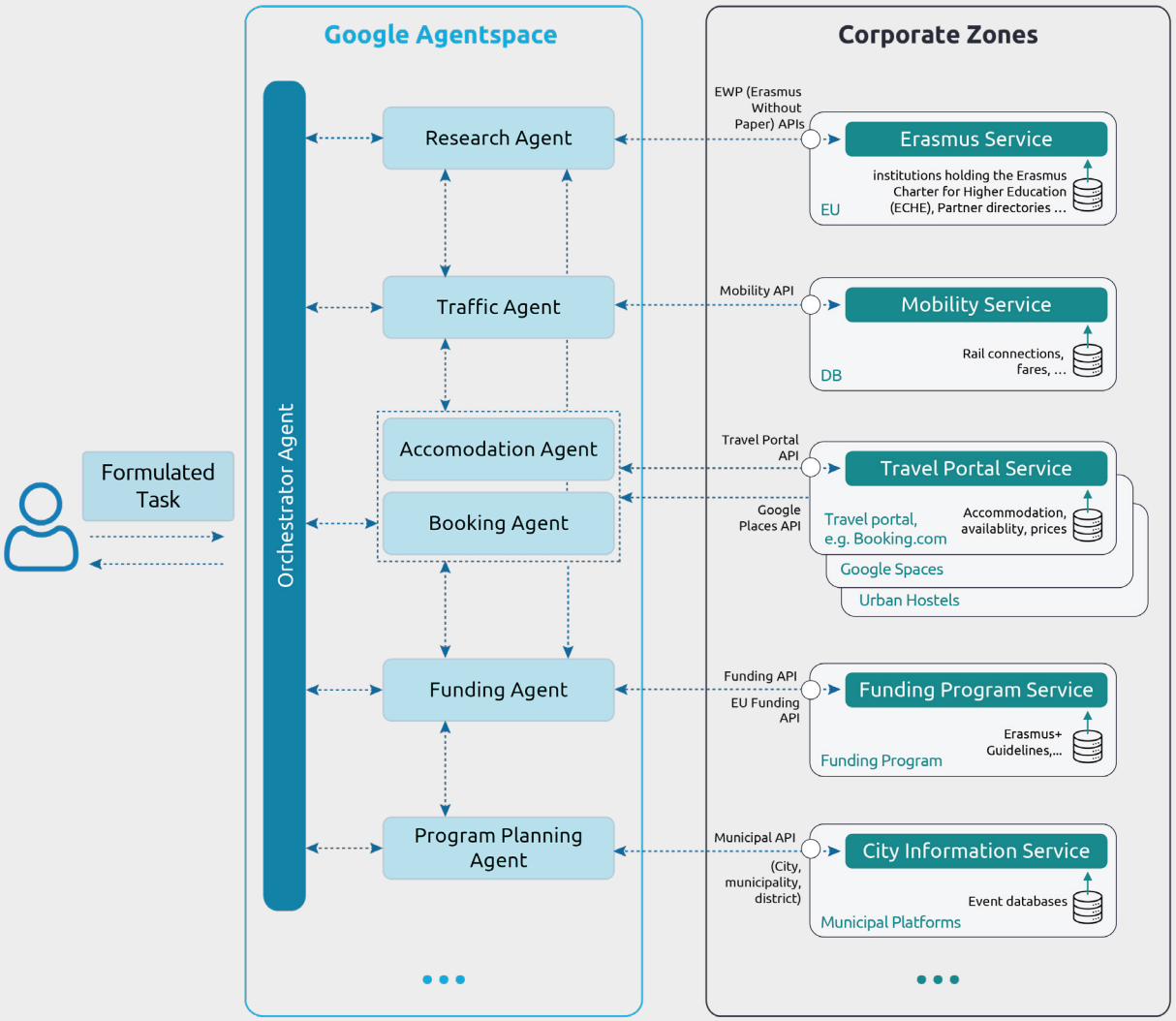


Figure 4: Cross-organizational agent interaction in Google Agentspace – example of AI-driven educational trip planning

Example 2: AI-supported response to a parliamentary inquiry

You are working in a federal ministry department for labor market policy. A parliamentary inquiry arrives, consisting of 15 detailed questions regarding youth unemployment programs, funding distribution by region, and measurable outcomes of recent initiatives. The deadline for response is tight, and the information needed is spread across internal databases, previous reports, statistical portals, and policy documents.

Instead of manually coordinating multiple units, an **AI agent swarm** is deployed to handle the task efficiently.

The **Document Retrieval Agent** searches for previous reports, legislative texts, and knowledge bases to extract relevant passages and figures.

The **Data Aggregation Agent** pulls the latest statistics from government dashboards and harmonizes them by region and timeframe.

The **Policy Context Agent** cross-references funding guidelines and program goals to ensure statements are compliant with current legal frameworks.

The **Drafting Agent** formulates initial answer texts in administrative language, adapting tone and style to parliamentary conventions.

The **Validation Agent** checks for completeness, factual consistency, and alignment with internal positions.

Finally, the **Orchestrator Agent** assembles all responses, structures the document, and prepares it for final human review and ministerial approval.

This approach accelerates turnaround times, reduces manual workload, and improves consistency across departments – while maintaining full transparency and human oversight.

On-premise vs. cloud

AI agents are not limited to hyperscale cloud infrastructures; they can also be deployed in sovereign public cloud and on-premise infrastructures.

Several of the systems presented offer full on-premise deployment options, including:

n8n: Open-source platform, fully deployable on-premise. Offers maximum flexibility for integrations and full control over data flows.

Make: Available as an enterprise version with private hosting or dedicated instances (i.e. a private version of software used by one organization only) for large organizations.

Relevance AI: Explicitly offers private cloud and single-tenant deployments, ideal for handling sensitive processes.

Elastic Stack (Elasticsearch, Kibana): Open-source by default, deployable entirely on private servers or isolated networks.

While AI agents can be deployed on-premise to meet specific regulatory or security requirements, such environments often face inherent limitations. These include restricted hardware scalability, limited support for large-scale or state-of-the-art AI models, and challenges in maintaining high availability and connectivity. As AI workloads become increasingly complex and data-intensive, these constraints can hinder innovation and responsiveness.

Sovereign cloud platforms offer a powerful alternative for public sector organizations by combining cloud-native scalability with strict data sovereignty and compliance. Solutions like the

Delos Cloud[10] enable you to deploy AI agents in sovereign environments with full control over data flows, model behavior, and integration. Similarly, STACKIT, the cloud platform by Schwarz Group, provides a European-based infrastructure that ensures GDPR compliance and operational sovereignty while supporting modern AI workloads.

These platforms bridge the gap between the flexibility of hyperscale service providers and the control of on-premise deployments. They empower public sector organizations to run advanced AI agents securely and efficiently – without compromising on performance, compliance, or trust.

Platform overview: automation meets agent intelligence

The variety of automation and agent platforms available is vast. On the automation side, traditional platforms like Make, Zapier, or n8n focus on rule-based workflows, connecting services and streamlining repetitive processes. On the agent side, modern agent platforms such as Relevance AI or LangChain use AI agents that act autonomously, make decisions, and continuously improve.

But as the landscape of these platforms evolves, the lines between the two kinds are increasingly blurring. Many providers now offer hybrid solutions – like n8n's "AI Agent Node" or Zapier's "AI Actions" – that integrate intelligent agents into automation flows. These developments unlock new possibilities but also demand deeper technical understanding, especially for complex or custom use cases. Figure 5 shows how these tools and platforms interact.

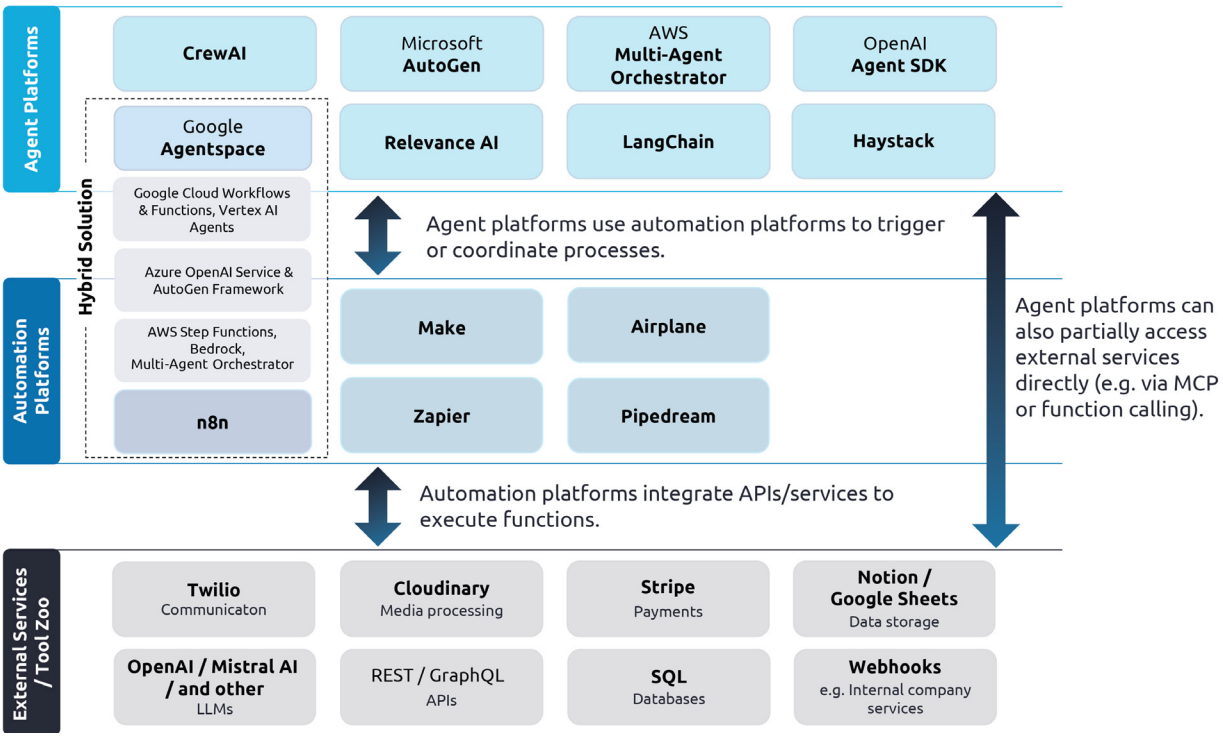


Figure 5: Overview of platforms and how they interact

[10] Delos Cloud is a cloud-based platform designed for the digitalization of the German public sector, providing data storage, processing, and management services.

This is what AI really looks like

Real-world uses cases in the public sector

Government agencies dealing with high volumes of incoming requests can benefit from AI-powered solutions. Examples include:

- Citizen service centers that automatically respond to inquiries about forms, deadlines, or opening hours.
- School administrations that provide consistent answers to recurring questions about enrollment or school holidays.
- Immigration offices that generate multilingual responses regarding residence permits and document requirements.
- Building departments that reply to questions about application statuses or formal requirements.
- Social services offices that handle document-related follow-ups automatically.

The use cases in this section are examples of solutions with relevance for the public sector. They are based on specific solutions and vendors available today, but they can easily be transferred to other automation platforms. The workflows remain nearly identical, especially when it comes to connecting external APIs.

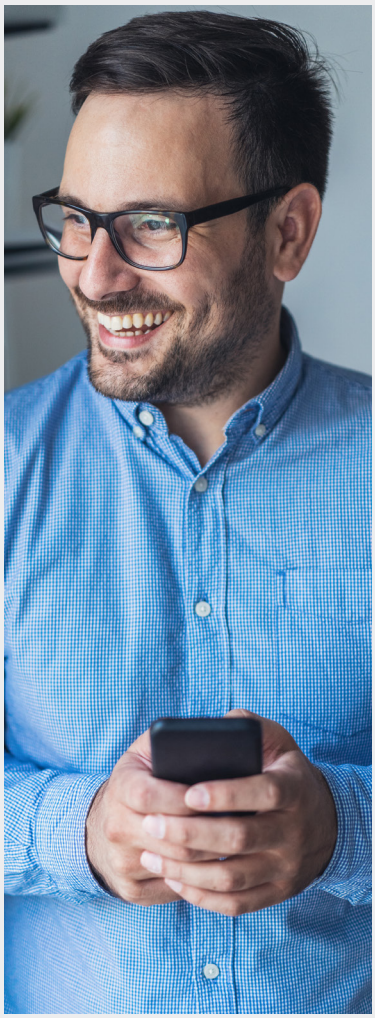
Automating social media posts with ChatGPT and Zapier

Social media is no longer just a playground for influencers. Public sector organizations are increasingly recognizing the strategic value of digital visibility and social media is gaining traction as a tool for engagement and outreach.

Amid growing talent shortages and demographic shifts, a strong presence on platforms like LinkedIn, Instagram, and Facebook is crucial to being perceived as a modern and attractive employer. At the same time, authentic communication fosters social cohesion and provides a stage for success stories, innovations, and citizen-centric services. A positive external image is not just a nice-to-have – it is a strategic must-have for building trust and attracting new talent.

Creating, publishing, and documenting social media content can be automated, with ChatGPT at the center of the process. ChatGPT is not just a creative content generator but also the trigger for an entire automation chain.

Zapier acts as a bridge between ChatGPT – where a custom GPT must be created – and the connected tools (see Figure 6).



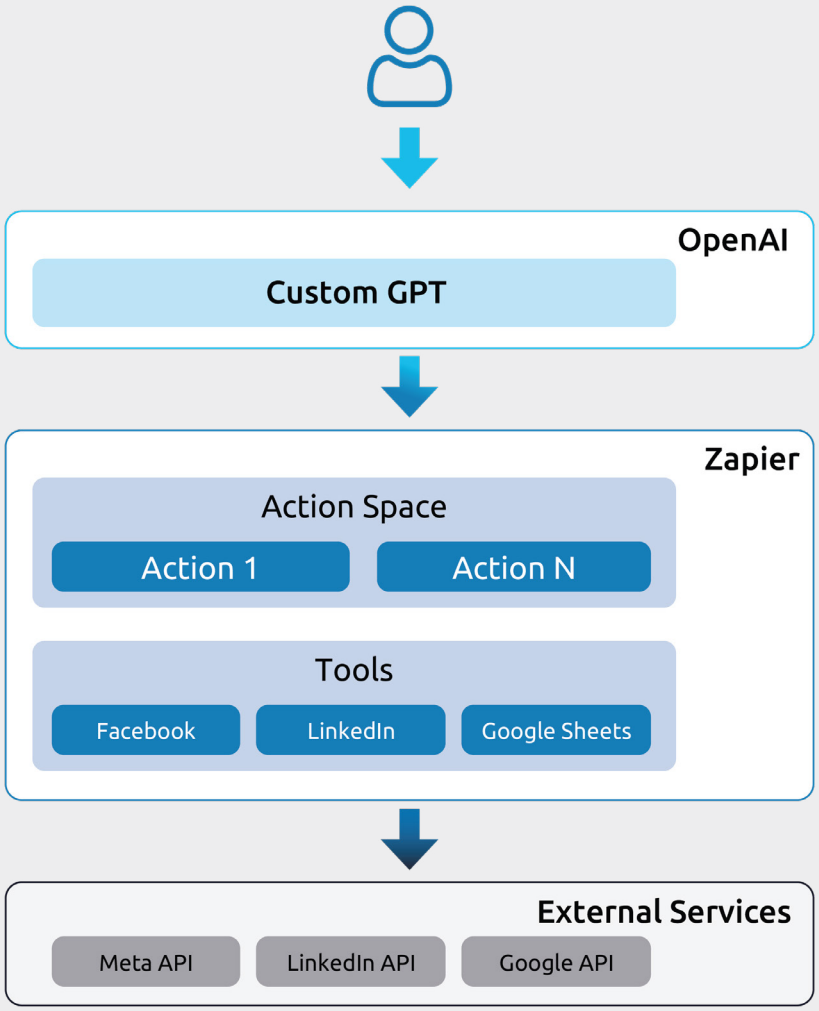


Figure 6: Triggering Zapier actions via a custom GPT

ChatGPT passes data and results from user interactions to Zapier, which then performs follow-up actions using tools connected via APIs. A common use case is the automated creation and publishing of social media posts: when you enter the topic **"AI in talent development"**, your custom GPT will generate two tailored posts. Zapier will then automatically publish them on your chosen platforms and log the post URLs in a Google Sheet. With this approach, you not only save valuable time but also ensure consistent and high-quality results across your social media channels.

Handling citizen emails

Government agencies are dealing with high volumes of incoming requests. AI can help to classify and process emails and act accordingly. Zapier manages the data flow and uses GPT functions such as text generation, analysis, or decision-making. Here's how:

Analyze the request: Zapier extracts the content of the email and passes relevant data such as subject line and message body to your custom GPT.

Generate a reply: your GPT creates a professional and personalized response tailored to the inquiry.

Send or log the response: Zapier automatically sends the GPT-generated reply back to the customer or stores it in systems like Jira, HubSpot, or Salesforce.

Using the "**AI by Zapier**"[11] application, you can incorporate language models into any workflow. This application can take on various tasks, such as generating reports from tables (Google Sheets, Airtable), providing automated FAQ responses, summarizing content, or crafting personalized messages.

[11] Zapier. https://zapier.com/blog/ai-by-zapier-guide/. (Visited 06.04.2025).

Automating meter readings via WhatsApp

This real-world use case shows how everyday administrative processes can become more efficient and citizen-friendly. In the German city of Hof, the local utility company uses WhatsApp to allow residents to digitally submit their electricity meter readings. The channel is low-threshold, which means it is easy to access and requires no special apps or technical knowledge. It is also direct and already part of users' daily routines – ideal for modern civic communication.

Such a solution can be built using the Twilio API combined with automation platforms like Zapier, Make, or n8n. WhatsApp handles user input, while the automation platform manages the backend orchestration and processing – from data capture to storage.

Here is a possible process flow: a citizen receives a QR code by mail. After scanning it, an automation platform (e.g. Zapier, Make, or n8n) is triggered and sends a WhatsApp message via Twilio, asking the citizen to submit a photo of their electricity meter.

The citizen replies with the image. The platform triggers an AI image recognition service that extracts and validates the meter reading. The reading is then automatically stored in a database or Enterprise Resource Planning (ERP) system. Finally, the citizen receives confirmation via WhatsApp and later an invoice via email.

The advantages are obvious: no login is required – just WhatsApp. The entry barrier for users is low, and there are minimal media disruptions compared to manually filled web forms. The entire solution can be implemented using platforms like Zapier, Make, or n8n, along with APIs such as Twilio. An on-premise setup, for instance using n8n, not only enables rapid and flexible integration, but also ensures full data sovereignty – fostering trust. It can also be scaled seamlessly across an entire country if needed.

Figure 7 provides a schematic overview of the architecture for sending WhatsApp messages.

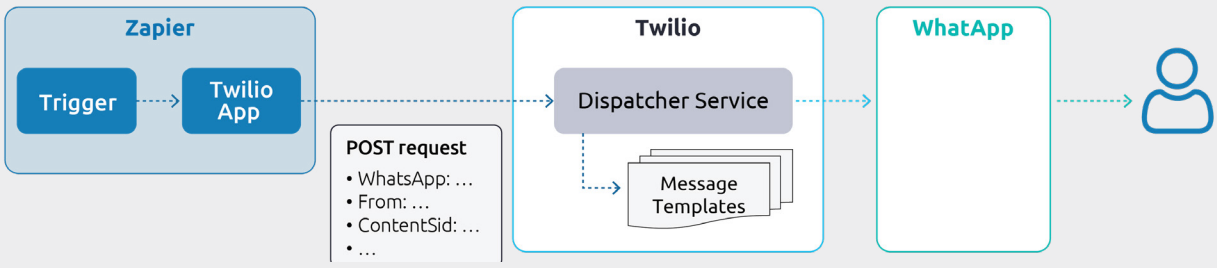


Figure 7: Automated WhatsApp message delivery via Twilio

Automating services with Relevance AI

Service automation can significantly improve the citizen application process for government services – benefiting both citizens and employees. It can, for example, detect missing attachments early and enable faster, less effort-intensive responses. It can also help bridge gaps in digital public services, especially when processes still rely on mail or PDF-based applications. It can do this by analyzing unstructured text and automatically forwarding the application to the appropriate person. The same applies to complex submissions involving advanced concepts such as applications for building power plants or wind turbines.

AI agents can respond to questions about waste collection schedules, identifier (ID) renewals, childcare registration, or benefit entitlements – tasks that are often repetitive, time-sensitive, and resource-intensive. Help desks can also analyze incoming tickets more quickly and 24x7 hours a day. Standard issues such as password resets or Virtual Private Network (VPN) access are resolved quickly and consistently.

Finally, agent platforms can take over the intelligent processing of unstructured data – content that cannot be handled by fixed rules, such as free text, chat logs, support inquiries, and complex form inputs.

For example, Relevance AI provides a powerful agent layer: agents that understand, interpret, classify, and trigger targeted actions or follow-up questions. These agents can delegate to sub-agents or collaborate across contexts.

In human resources (HR) departments, Relevance AI helps to pre-qualify applications by filtering relevant profiles, drafting replies, and forwarding suitable candidates to hiring teams. It also sends automated status updates to applicants.

In the higher education sector, it answers student requests related to course registration, exam procedures, or financial aid consistently and in line with official guidelines.

These examples illustrate how Relevance AI adds the missing intelligence layer for language-based, unstructured processes in government services, educational institutions, and IT operations. For complex, legally sensitive, or emotionally charged inquiries, the "Escalate to Human" function is triggered automatically. This hands the request to a human caseworker – along with all analysis steps, classified content, and response drafts – so they can respond quickly, accurately, and empathetically.

Figure 8 illustrates how the system works.

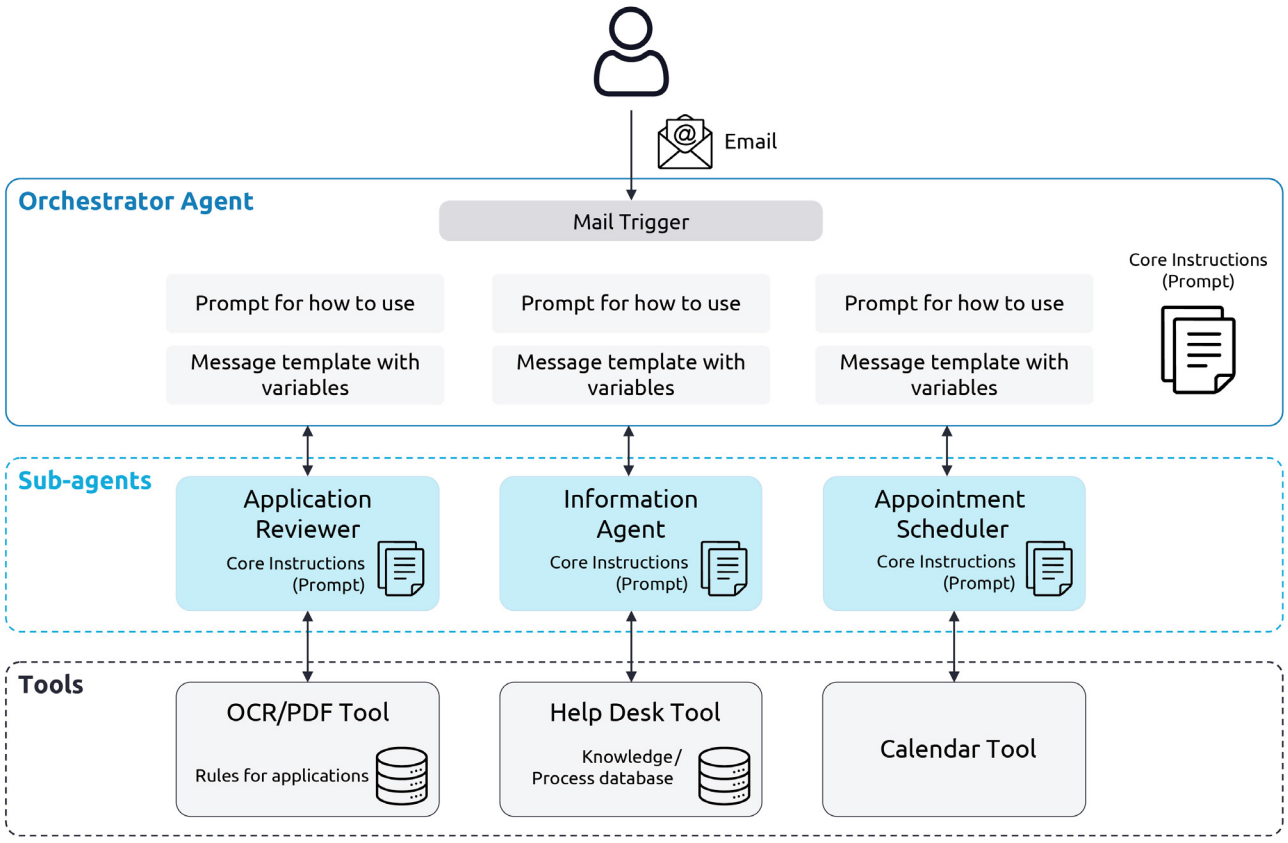


Figure 8: Relevance AI architecture for service automation

In the HR scenario, a central orchestrator agent processes incoming emails. The main component is the Core Instruction – a central prompt that defines how the agent reasons, communicates, and makes decisions. The orchestrator analyzes the subject line, content, and attachments, classifies the request (e.g. as an application, information request or appointment inquiry), and forwards it to the appropriate sub-agent. Following the "divide and conquer" principle, specialized sub-agents with clearly defined roles take over individual parts of the task. Each sub-agent receives its input through a message template containing variables such as the subject, message body, attachments, or user context, so it knows exactly which content to process.

For example, the Application Reviewer analyzes incoming applications, checks them for completeness (including required fields, signatures, and IBAN), and formulates follow-up questions if needed. It extracts data from PDF attachments, evaluates them using rule-based logic or LLMs, and generates clear and understandable replies.

Meanwhile, the Information Agent handles general inquiries regarding opening hours, departmental responsibilities, or the current status of a request. To do this, it accesses internal administrative processes and knowledge databases.

The Appointment Scheduler scans employee calendars for available time slots and sends appointment confirmations via email. If required information is missing, it automatically follows up with the requester to complete the scheduling process.

Monitoring and dashboards

Automation accelerates processes, but without oversight, it can quickly turn into a black box. That is why monitoring is a core element of any production-grade AI or workflow architecture – particularly for public sector organizations, for whom transparency is a priority.

One of the most powerful and open-source solutions available is the Elastic Stack, which consists of Elasticsearch for data indexing, Logstash for data processing and transport, and Kibana for data visualization. These tools are widely regarded as the de facto standard for real-time dashboards and operational monitoring for many organizations.

The use cases for automated monitoring, such as continuous data visualization, are highly diverse.

Whether you are tracking support tickets in customer service or applications in real time, it plays a key role in keeping processes transparent, identifying bottlenecks early, and enabling data-driven decisions.

Take real-time application monitoring as an example. Suppose you are running an automation that processes incoming applications, such as grant applications, vacation requests, or service tickets. These applications are submitted via web forms or email and processed through a workflow using tools like Zapier, n8n, or Make.

Figure 9 shows an architecture using Zapier to demonstrate how this process can be monitored in real time.

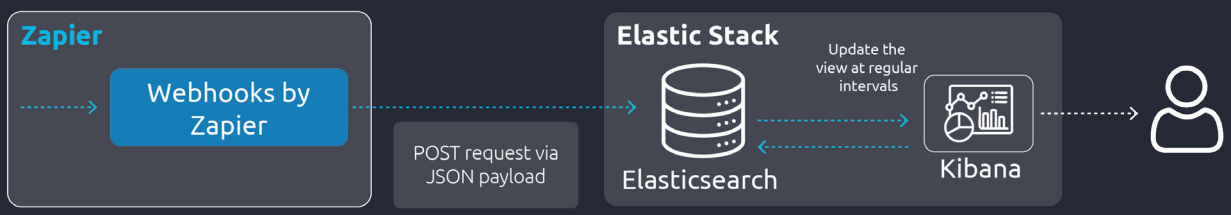


Figure 9: Automated dashboards with Elastic Stack

This approach always results in full transparency: How many applications have been received? Which categories are most common? Where are processes getting stuck?

A major advantage of the Elastic Stack is that it is open-source, free to use, and can be hosted on your own servers – making it ideal for organizations with strict data protection requirements. At the same time, the system is highly flexible: you can integrate almost any data source and visualize the key metrics that matter for your operations – whether you are in public administration, IT support, or industrial production.

Critical reflectio and limitations

AI agents and automation platforms offer enormous efficiency gains but also bring challenges that must be consciously addressed.

Complexity and implementation effort

Building a complex agent architecture requires significant effort in planning, configuration, testing, and monitoring. While systems like n8n, Relevance AI, or Make are technically flexible, they demand a deep understanding of processes, data flows, and security requirements – especially in the public sector.

Costs and resources

Although many tools offer open-source or free-tier models, running productive environments often generates substantial costs for server infrastructure, scaling, maintenance, and licensing of advanced features. On-premise setups ensure data sovereignty but also increase operational and staffing demands – adding extra pressure to public sector organizations already facing labor shortages.

Risks and error sources

Automated systems are only as reliable as their inputs and configurations. Misconfigured agents can produce faulty responses, mishandle sensitive data, or escalate processes unintentionally if proper monitoring is lacking. Without clearly defined error handling and human escalation pathways, residual risks remain.

Ethical and legal considerations

Deploying AI agents in public communication or administration touches on ethical dimensions: transparency towards citizens, non-discriminatory decision-making, and the protection of sensitive personal data. Compliance, fairness, and accountability must be ensured at the highest level.

Applying "responsible AI" (in other words, developing and deploying AI systems that are lawful, ethically sound, and technically robust throughout their lifecycle) allows public sector organizations to meet these requirements. In the public sector, this means not only protecting data through systems that are secure by design. It also means maintaining public trust in how government institutions use automation.

Depending on the specific use case, the **EU AI Act** may also apply – particularly where decisions affect individuals' rights or require human oversight. In Germany, the **Administrative Procedures Act (Verwaltungsverfahrensgesetz)** explicitly prohibits the automation of human discretion granted by law. Other countries take a different stance: the **United Arab Emirates**, for example, have just introduced legal provisions enabling fully autonomous administrative decisions under certain conditions.

Despite these challenges, the potential of AI agents and automation platforms remains enormous. They create space for more meaningful, value-driven work, improve service quality, and offer a unique opportunity to make administrative and business processes future-proof – **as long as they are implemented thoughtfully and responsibly.**

Key risks at a glance

Area	Potential risk
Complexity	High setup and maintenance effort for reliable operations
Costs and resources	Significant operational costs for hosting, scaling, and licenses
Error sources	Faulty configurations or missing monitoring can cause malfunctions
Ethical considerations	Risks related to transparency, bias, and data protection
Legal compliance	Strict requirements for GDPR, data sovereignty, and accountability

From agentic vision to action

Six steps public sector leaders should take now

Agentic AI marks a fundamental shift in how automation is designed, deployed, and governed. For technical leads and architects, the challenge lies in translating this potential into secure, observable, and deeply integrated systems. The essential steps below outline how to move from concept to implementation, providing a practical roadmap for building agentic AI into real-world, public sector workflows.

1. Build a robust data foundation

For AI agents to make autonomous and context-aware decisions, you need a strong, well-managed data infrastructure. This foundational step is critical: AI agents can only perform well if the data they rely on is high-quality, accessible, and trustworthy.[12] Here are some important steps in building this foundation:

- **Create a unified semantic data model**
  Make sure systems use the same terms (such as citizenID, caseID, and documentType) so AI agents interpret data the same way.
- **Provide clean, reliable data through APIs:**
  Use API gateways or data layers that give agents up-to-date, verified information – minimizing the risk of hallucinations or outdated logic.

[12] Capgemini. Data foundations for government: From AI ambition to execution. https://www.capgemini.com/insights/research-library/data-mastery-in-government/. 2025 (Visited 03.07.2025).

- **Track data sources and changes:** Use metadata tools to see where data comes from, how it changes, and how agents use it. This is essential for troubleshooting and audits.
- **Establish a process for validating data:** Set up systems to catch problems like missing fields, unexpected formats or strange patterns – especially when using event-based systems like Kafka or webhooks.
- **Train agents safely using synthetic or test data:** Create safe, isolated environments for testing agents, to protect live systems and personal data while you fine-tune behavior.
- To make sure your AI agents operate within a secure and compliant framework:
- Use model proxies or gateways like PromptLayer or Azure API Management to control access.
- Enforce request validation, the masking of personal info, and limits on LLM use.
- Choose hosting models that meet the compliance needs of your environment (e.g. sovereign cloud or zero-trust on-prem).

2. Assess automation readiness at the system level

- Keep documentation on each AI agent's capabilities, risks, and backup plans.
- Start by identifying where agentic automation can slot into your existing stack. Focus on processes that are already partly digitized and can connect via APIs, webhook endpoints, or message queue systems like Kafka or RabbitMQ. Common examples include:
- Incoming emails with structured attachments (PDFs, forms)
- Workflows triggered via service portals
- Systems with reliable CRUD APIs or webhook capabilities
- Apply event-driven architecture principles to pinpoint where agents can listen, act, and pass along results autonomously. Be clear early on about what the system is responsible for and how to avoid repeated actions that could cause errors.

3. Select a suitable agentic runtime architecture, then design for context

- Choose an execution model that works within your operational constraints. When choosing, think about things like speed, how much it needs to scale, where your data lives, and how easy it is to monitor.
- **n8n** is ideal for on-premise or air-gapped environments. It is visual, low-code and supports LLMs through "AI Agent Node" or HTTP integrations.
- **Relevance AI and LangChain** are modular, Python-first frameworks suitable for building LLM-based, multi-agent systems that can remember things, use tools, and work together.
- **OpenAI Agent SDK, AutoGen, and CrewAI** are best for more advanced automation, like agents that make decisions, reason step by step and choose what to do next on their own.
- AI agents need both short-term and long-term context. Saving their thought processes helps make results more consistent, as well as clear and easy to review.
- **Short-term memory**
  Keep recent conversation history using tools like a token buffer, sliding window, or summary buffer.
- **Long-term memory**
  Use databases like Redis or Pinecone to store important facts, user preferences, or behavior over time. These are helpful for profiling and personalization.

4. Engineer prompts and interfaces systematically

Rather than treating prompts as one-off tricks, build them like real software components:

- **Use templates with variables** (like user metadata or prior actions) so prompts can adapt to different situations.
- **Set up tools** the agent can use, using OpenAI function-calling, LangChain tools, or the ReAct pattern.
- **Track prompt versions** just like you do with code, using GitOps or Config-as-Code pipelines.
- **Add rules and settings** (like schema checks or temperature limits) to make the AI's behavior more predictable.

Pro tip: Build shared prompt libraries and internal APIs for common tasks like classification, summarization, or eligibility checks.

5. Identify and prioritize use cases strategically

Not every use case is suitable for agentic automation. Use a decision matrix based on four key dimensions to prioritize where you implement it:

Criterion	Description	Best fit for agentic AI
Routine intensity	How frequently does the task occur (daily/ weekly vs. rarely)?	High-frequency tasks (e.g. email replies)
Criticality	What is the legal, financial, or reputational risk if something goes wrong?	Low to moderate (e.g. FAQs, reminders)
Creativity	Does the task require subjective judgement, empathy, or interpreting policy?	Low creativity (e.g. form checking)
Structurability	Can the process be described via rules, data, or schemas?	Highly structured (e.g. validations)

Map potential use cases across these four dimensions and implement the ones in the "high routine / low creativity / high structurability / low criticality" quadrant first. Examples include:

- Classifying emails and sending templated replies.
- Checking if someone meets basic eligibility requirements.
- Reading documents and filling out forms.
- Scheduling appointments and sending reminders.

You can also choose use cases by matching them to the right technical patterns and tools. Avoid starting with highly creative or policy-critical tasks. Instead, begin with low-risk use cases where you can test safely, measure success, and iterate quickly.

Pattern type	Typical use case	Tool stack example
Retrieval-augmented generation (RAG)	Dealing with inquiries about documents	LangChain + Pinecone + LLM
Function-calling agent	Running API-based workflows	OpenAI/Gemini + Zapier/n8n + REST APIs
Agent swarms	Coordinating across departments	AutoGen + LangGraph + Redis memory
On-prem NLP	Handling compliance-focused emails and documents	spaCy + Haystack + n8n + Postgres

6. Monitor, test, and improve your system all the time

Agentic systems are dynamic, but you need to be able to see what they are doing. Show key stats on dashboards – both technical ones, like speed and errors, and operational ones, like the number of use cases and successful completions.

- Set up request tracing and logging (using tools like Elastic Stack and OpenTelemetry).
- Keep track of details about each prompt, such as how many tokens were used, model version, and response time.
- Have back-up plans for when an AI fails to reach a confidence threshold or you need a human to step in.
- Use red teaming to regularly test the system for robustness, bias, and unusual cases.
- Run controlled experiments and measure impact by tracking both **how well the system works** and **how it helps users:**
- Time saved per case or task.
- Accuracy of decisions, like classification or routing.
- Model drift and hallucination rate.
- User satisfaction (internal and external).
- How often the system needs to hand back to a human.

Create "Agent Scorecards" to continuously check performance, improve behavior over time and make sure results are clear and traceable.

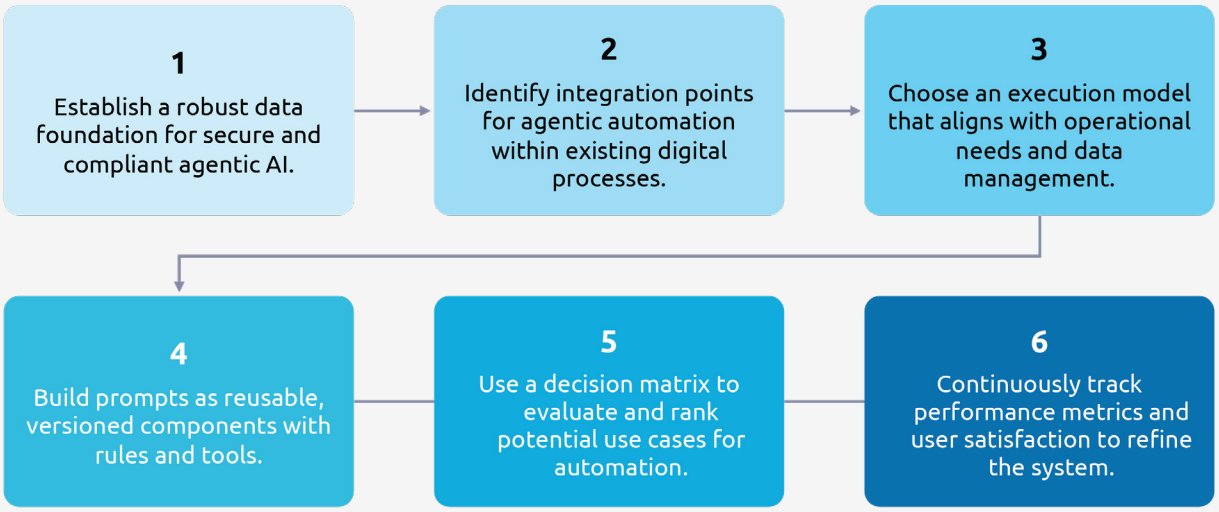


Figure 10: Roadmap for public sector leaders to implement agentic AI

Agentic AI is more than a strategic concept; it marks a new software paradigm. Architects and engineers are central to its success, responsible for building systems that are not only intelligent, but also secure, observable, and compliant. But true impact in the public sector comes not just from using AI. It comes from engineering AI systems that are deeply embedded in real-world public sector workflows.

Summary and outlook:

What we have learnt from the rise of AI agents

AI agents are no longer a future vision – they are already reshaping how we automate work, respond to complexity, and deliver services across domains. From basic copilots to dynamic multi-agent systems, this point of view has shown how agentic architectures unlock real potential: increasing scalability, smarter workflows, and tailored interactions that go far beyond traditional automation.

Especially in the public sector, the impact is tangible. Whether it is answering citizen requests, coordinating inter-agency communication, or handling policy documentation, agents help reduce workload, increase consistency, and free up human capacity for complex, value-adding tasks.

But implementing AI agents is not a plug-and-play process. It requires thoughtful design, legal awareness, technical expertise, and a strong sense of responsibility. That is why we have not just shown what is possible, but also what to watch out for – from data protection to explainability.

The key takeaway?

You do not need to build a fully autonomous system overnight. Start small, stay strategic, and build your agent ecosystem step by step – with the right tools, use cases, and partners. The technology is ready. The time is right. It is up to us to use it wisely.

How Capgemini can help

Designing human-in-the-loop agent systems

We make sure AI agents augment – not replace – your workforce by embedding human validation into decision-making processes where discretion or accountability are essential.

Implementing transparent and auditable logic

We build our architectures with explainable models and traceable workflows, so public institutions can meet legal requirements for transparency, documentation, and review.

Upholding full data sovereignty

We deploy AI agent platforms on sovereign clouds (e.g. Delos Cloud, StackIT) or on-premise infrastructures, so systems stay compliant with GDPR and national data protection laws.

Integrating with legacy systems and public IT landscapes

We specialize in connecting agentic workflows to existing administrative systems – including specialized procedures, registers, and document management systems – without disrupting current operations.

Analyzing your current IT and data landscape

We assess your infrastructure, identify integration opportunities, recommend the most suitable agent platform – open source, cloud-native, or sovereign – and implement it fully.

Developing domain-specific agents

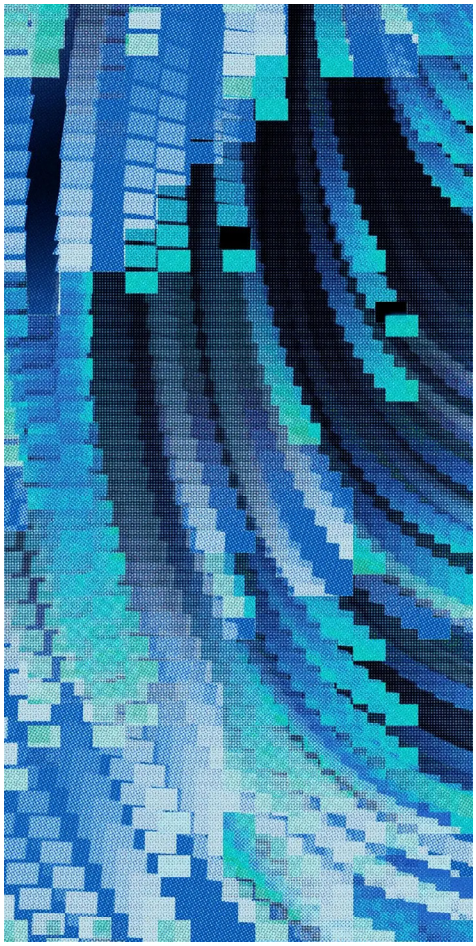
We train and configure AI agents for government use cases, like citizen communication, eligibility checks, or document classification – all tailored to your language, tone, and process logic.

Building trust through responsible AI governance

We help establish governance frameworks that align with the EU AI Act. These cover risk classification, oversight structures, red teaming, and continuous monitoring of AI behavior.

Training your teams for hybrid collaboration

We help public employees to work effectively with AI through targeted upskilling programs, workshops, and sandbox environments for experimenting safely.



Read more from Capgemini

Resonance AI framework (Capgemini)

Rise of agentic AI: How trust is the key to human-AI collaboration (Capgemini Research Institute)

Think big, start small: unleashing the transformative power of Gen AI and agentic AI across government (Capgemini)

Data foundations for government: from AI ambition to action (Capgemini Research Institute)

Connecting the dots: data sharing in the public sector (Capgemini Research Institute)

Authors

Would you like to start a conversation with us? Then feel free to get in touch:

Eldar Sultanow

Business Enterprise Architect Director
Capgemini Germany
eldar.sultanow@capgemini.com

Lars Santesson

Chief Technology Officer Public Sector
Capgemini Germany
lars.santesson@capgemini.com

Ceyda Icöz

Business Analyst
Capgemini Germany
ceyda.icoez@capgemini.com

About Capgemini

Capgemini is an AI-powered global business and technology transformation partner, delivering tangible business value. We imagine the future of organizations and make it real with AI, technology and people. With our strong heritage of nearly 60 years, we are a responsible and diverse group of 420,000 team members in more than 50 countries. We deliver end-to-end services and solutions with our deep industry expertise and strong partner ecosystem, leveraging our capabilities across strategy, technology, design, engineering and business operations. The Group reported 2024 global revenues of €22.1 billion.

Make it real | www.capgemini.com