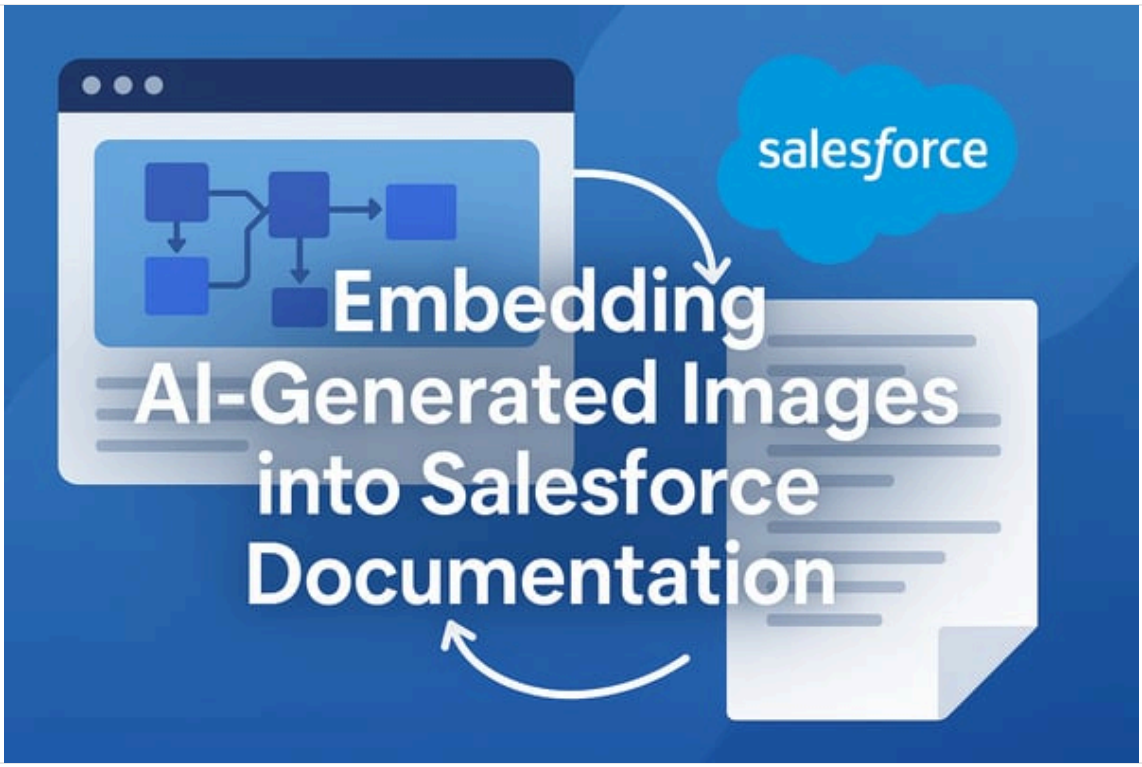
Using AI to Create Visuals for Salesforce Documentation

Published August 21, 2025 30 min read



Embedding AI-Generated Visuals in Salesforce Documentation

Overview: AI-Generated Visuals in Technical Documentation

AI-generated visuals are emerging as a powerful aid for technical writers, enabling the rapid creation of architecture diagrams, flowcharts, and other illustrations from natural language descriptions. In the context of Salesforce architecture and documentation, these visuals can dramatically improve how complex systems are communicated. By converting descriptions of CRM systems, data flows, or integration processes into diagrams, large language models (LLMs) like GPT-4 can help document

designers explain Salesforce solutions more clearly (Source: medium.com). This accelerates the documentation process and enhances reader understanding, as visual diagrams often convey structure and sequence better than text alone.

Generative AI image models (such as OpenAI's DALL·E 3) also offer the ability to produce custom illustrations or conceptual art to accompany technical content. For example, members of the Salesforce community have started to leverage AI for visuals in their publications. Salesforce architect *Bob Buzzard* (Keir Bowden) has experimented with using DALL·E 3 to generate illustrative images for blog posts on Salesforce topicsbobbuzzard.blogspot.com. These AI-created images can add visual interest or metaphors to documentation (e.g. a DALL·E image symbolizing a "Ship Happens" continuous deployment scenariobobbuzzard.blogspot.com), making technical guides more engaging. Overall, AI-generated visuals – whether schematic diagrams or creative graphics – support Salesforce documentation by saving time and providing clearer, up-to-date visual explanations of complex ideas.

AI Tools for Generating Salesforce Diagrams

Modern LLMs and generative models can produce two broad types of visuals for documentation:

- **Diagram Code Generation with LLMs (GPT-4):** GPT-4 and similar LLMs can translate a plain-language description of an architecture or process into diagram markup languages like PlantUML or Mermaid. PlantUML and Mermaid are text-based syntax systems for creating diagrams (UML charts, flowcharts, sequence diagrams, etc.) that can be rendered into images. Using GPT-4 to generate these definitions allows a writer to get a first draft of a diagram without manual drawing (Source: medium.com)(Source: bool.dev). For instance, a writer might prompt: *"Generate a sequence diagram of a Salesforce login flow with a user, the Salesforce server, and an authentication service. Use Mermaid syntax."* The LLM can output a Mermaid sequence diagram code reflecting that description. One example in a related context is asking ChatGPT to explain how DNS works in a sequence diagram; the AI produced a correct Mermaid diagram defining user, browser, DNS resolver, etc., saving significant manual effort (Source: bool.dev). GPT-4 can similarly produce UML diagrams (class diagrams, flowcharts, deployment diagrams) by describing Salesforce components and relationships – for example, outlining a Salesforce data model with Accounts, Contacts, and a custom object could yield PlantUML code for a class diagram. The key benefit is speed and accuracy in capturing the described structure. Writers may use a library of prompt templates to ensure consistent output format (e.g. always requesting the diagram in a specific syntax and style).

- **Generative Image Models (DALL·E, Stable Diffusion):** For visuals that are less about precise architecture and more about conceptual or decorative illustrations, image-generation models can be employed. A technical writer could prompt DALL·E with a request for an architecture schematic or an **infographic-style image**. For example: *"An isometric illustration of a Salesforce cloud architecture*

with databases and integration arrows, in a flat icon style." The model might produce a unique image to use in an overview section. These tools can also create **cover images** for documentation pages (as Bob Buzzard did, using an AI image to represent an "evil AI assistant" scenario (Source: bobbuzzard.blogspot.com)). However, purely generative images have limitations: they might not precisely follow Salesforce's official iconography or could introduce inaccuracies (e.g., a DALL·E image might depict generic cloud servers rather than Salesforce-specific components). Thus, they are usually used to supplement documentation (for visual appeal or conceptual diagrams), while structured diagrams (like UML-style charts) are better handled with text-to-diagram tools for precision.

By combining GPT-4 for diagram code and models like DALL·E for creative illustrations, Salesforce documentation teams can cover both precise technical diagrams and engaging conceptual visuals. In all cases, human oversight is essential to verify that the AI-generated visuals correctly represent the Salesforce features or architecture being documented.

Workflow Overview: From Prompt to Published Diagram

Embedding AI-generated visuals into a Git-backed documentation site involves a series of steps that integrate LLM outputs with documentation tools. Below is a **step-by-step workflow** illustrating how a technical writer can go from an idea to an auto-updated diagram in a Salesforce documentation portal (assuming a docs-as-code approach using Docusaurus, PlantUML/Mermaid, Git, and CI/CD):

1. **Draft the Diagram Prompt:** The process begins with the writer formulating a prompt for the LLM describing the desired diagram. For example, suppose the writer needs an architecture diagram of a Salesforce integration. They might write a prompt like: *"Draw a system architecture diagram in PlantUML. Show a Salesforce Sales Cloud org, a middleware API gateway, and an external billing system. Indicate data flows from Salesforce to the API gateway and to the billing system."* The prompt can be refined with directives (e.g., *"use PlantUML deployment diagram syntax with nodes and components"*). Using a consistent prompt template helps ensure the AI's output is in the correct format (for instance, always starting with `@startuml` and ending with `@enduml` for PlantUML).

2. **Generate Diagram Code with the LLM:** The writer feeds the prompt to GPT-4 (via ChatGPT or an API integration). The LLM responds with diagram code in the requested syntax. For instance, GPT-4 might return:

   plantuml

   Copy

```
@startuml actor User node SalesforceOrg <<Salesforce>> { component \"Sales Cloud\" as SC } node APIGateway { component \"Middleware API\" as API } node BillingSystem { component \"Billing DB\" as DB } User --> SC: uses; SC --> API: sends data; API --> DB: billing info; @enduml
```

This code is a textual representation of the diagram. In practice, ChatGPT's output quality is high – it converts natural language descriptions into valid PlantUML syntax that can be rendered (Source: medium.com)(Source: medium.com). If Mermaid was requested instead, a similar text block in Mermaid syntax would be produced. The technical writer reviews this output, ensuring it matches the intended architecture (and asking the LLM to adjust if something is missing or incorrect, using iterative prompts). Notably, LLMs can refine diagrams through dialogue: for example, *"Add an arrow showing the API Gateway calling back to Salesforce"* would prompt GPT-4 to insert the additional relationship in the code. This iterative refinement leverages the LLM's memory to incrementally improve the diagram (Source: medium.com)(Source: medium.com).

3. **Incorporate Diagram Code into Documentation:** Once satisfied, the writer integrates the diagram into the documentation source. There are two common methods:

    - **Embedding as Diagram Source (Mermaid in Markdown):** If using Docusaurus (or similar static site generators) with Mermaid, the writer can paste the code directly into a Markdown/MDX file inside a triple-fenced code block annotated with `mermaid`. For example:

        ```
        \`\`\`mermaid
        ```

        ```
        graph TD; SC[Salesforce Sales Cloud] --> API[Middleware API Gateway]; API --> DB[Billing System];
        ```

        Copy

```
``Docusaurus v2/v3 supports rendering Mermaid diagrams from such Markdown code blocks v
```

Tags: salesforce, generative ai, technical documentation, llm, diagramming, dall-e, salesforce architect

About Cirra

About Cirra AI

Cirra AI is a specialist software company dedicated to reinventing Salesforce administration and delivery through autonomous, domain-specific AI agents. From its headquarters in the heart of Silicon Valley, the team has built the **Cirra Change Agent** platform—an intelligent copilot that plans, executes, and documents multi-step Salesforce configuration tasks from a single plain-language prompt. The product combines a large-language-model reasoning core with deep Salesforce-metadata intelligence, giving revenue-operations and consulting teams the ability to implement high-impact changes in minutes instead of days while maintaining full governance and audit trails.

Cirra AI's mission is to **"let humans focus on design and strategy while software handles the clicks."** To achieve that, the company develops a family of agentic services that slot into every phase of the change-management lifecycle:

- **Requirements capture & solution design** – a conversational assistant that translates business requirements into technically valid design blueprints.
- **Automated configuration & deployment** – the Change Agent executes the blueprint across sandboxes and production, generating test data and rollback plans along the way.
- **Continuous compliance & optimisation** – built-in scanners surface unused fields, mis-configured sharing models, and technical-debt hot-spots, with one-click remediation suggestions.
- **Partner enablement programme** – a lightweight SDK and revenue-share model that lets Salesforce SIs embed Cirra agents inside their own delivery toolchains.

This agent-driven approach addresses three chronic pain points in the Salesforce ecosystem: (1) the high cost of manual administration, (2) the backlog created by scarce expert capacity, and (3) the operational risk of unscripted, undocumented changes. Early adopter studies show time-on-task reductions of 70-90 percent for routine configuration work and a measurable drop in post-deployment defects.

Leadership

Cirra AI was co-founded in 2024 by **Jelle van Geuns**, a Dutch-born engineer, serial entrepreneur, and 10-year Salesforce-ecosystem veteran. Before Cirra, Jelle bootstrapped **Decisions on Demand**, an AppExchange ISV whose rules-based lead-routing engine is used by multiple Fortune 500 companies. Under his stewardship the firm reached seven-figure ARR without external funding, demonstrating a knack for pairing deep technical innovation with pragmatic go-to-market execution.

Jelle began his career at ILOG (later IBM), where he managed global solution-delivery teams and honed his expertise in enterprise optimisation and AI-driven decisioning. He holds an M.Sc. in Computer Science from Delft University of Technology and has lectured widely on low-code automation, AI safety, and DevOps for SaaS platforms. A frequent podcast guest and conference speaker, he is recognised for advocating "human-in-the-loop autonomy"—the principle that AI should accelerate experts, not replace them.

Why Cirra AI matters

- **Deep vertical focus** – Unlike horizontal GPT plug-ins, Cirra's models are fine-tuned on billions of anonymised metadata relationships and declarative patterns unique to Salesforce. The result is context-aware guidance that respects org-specific constraints, naming conventions, and compliance rules out-of-the-box.
- **Enterprise-grade architecture** – The platform is built on a zero-trust design, with isolated execution sandboxes, encrypted transient memory, and SOC 2-compliant audit logging—a critical requirement for regulated industries adopting generative AI.
- **Partner-centric ecosystem** – Consulting firms leverage Cirra to scale senior architect expertise across junior delivery teams, unlocking new fixed-fee service lines without increasing headcount.
- **Road-map acceleration** – By eliminating up to 80 percent of clickwork, customers can redirect scarce admin capacity toward strategic initiatives such as Revenue Cloud migrations, CPQ refactors, or data-model rationalisation.

Future outlook

Cirra AI continues to expand its agent portfolio with domain packs for Industries Cloud, Flow Orchestration, and MuleSoft automation, while an open API (beta) will let ISVs invoke the same reasoning engine inside custom UX extensions. Strategic partnerships with leading SIs, tooling vendors, and academic AI-safety labs position the company to become the de-facto orchestration layer for safe, large-scale change management across the Salesforce universe. By combining rigorous engineering, relentlessly customer-centric design, and a clear ethical stance on AI governance, Cirra AI is charting a pragmatic path toward an autonomous yet accountable future for enterprise SaaS operations.
