Building winning AI with Claude on Vertex AI

A guide to starting, scaling, and succeeding based on real-world examples from Anthropic and Google Cloud

ANTHROP\C

Foreword

Today's innovative businesses need two things to stay ahead: powerful AI models and solid infrastructure.

That's why Anthropic is proud to be a Google Cloud partner and offer our frontier Claude models on Vertex AI, Google Cloud's fully managed, unified AI development platform. Through this partnership, you gain access to leading AI models that solve real-world problems, alongside a comprehensive platform that makes evaluating, building, and deploying AI solutions straightforward.

We've seen customers achieve remarkable results—like 20-30% faster feature development cycles, 96% reduction in audit times, and millions of dollars in operational savings—all by thoughtfully implementing Claude on Vertex AI.

In this guide, we share lessons and real-world examples, drawing from thousands of customers building with Claude on Vertex AI, including Palo Alto Networks, Quora, Replit, and more. We also provide practical guidance on developing AI strategies and implementation roadmaps for teams at earlier stages of their AI journey.

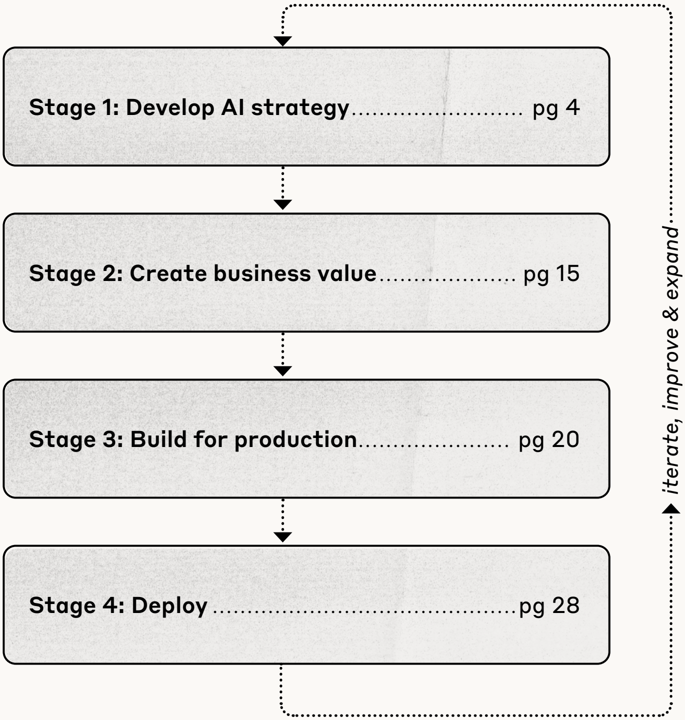
Michael Gerstenhaber, VP of Product, Anthropic

"Anthropic and Google Cloud share the same values when it comes to developing AI—it needs to be done in both a bold and responsible way. This expanded partnership with Anthropic, built on years of working together, will bring AI to more people safely and securely, and provides another example of how the most innovative and fastest growing AI startups are building on Google Cloud."

Thomas Kurian, CEO, Google Cloud

Table of contents

Here's the framework we'll be following:



Stage 1: Develop your AI strategy

The winning formula for building AI is methodical: build strong foundations, identify high-impact use cases, and scale what works. In this section, we'll walk through a typical implementation roadmap and the steps for designing your first AI pilot.

Already completed a successful pilot? Skip ahead to see which Claude model is a good fit as you move to production.

CREATING AN AI IMPLEMENTATION ROADMAP

Getting your organization to adopt any new technology requires a thoughtful approach to evolving your teams, processes, and tooling—and AI is no different. AI implementations often unfold across four key phases, with both technical depth and operational breadth increasing as you move through them.

Most organizations take about a year to progress through these phases and achieve broad enterprise adoption. However, we've seen many companies that use Claude on Vertex AI condense this implementation process into a few months, if not weeks, thanks to the integrated solution.

"Thanks to the quality of Anthropic's technology and the ease of implementing it through Vertex AI, we switched our chat feature to Claude 3.5 Sonnet and improved performance in a single weekend."

Guy Gur-Ari, Co-founder, Augment

Build your roadmap around these phases:

Phase 1: Foundation building (months 1–3)
Start by creating the infrastructure and organizational framework for sustainable AI adoption. During these crucial first months, focus on the following:

- Establishing a governance structure that balances innovation with risk management
- Defining comprehensive technical requirements for AI security, data privacy, and compliance
- Building and empowering a core team with clear mandates and diverse skills across engineering, data science, and product management

Your goal is to prevent the three primary causes of AI project failure: misaligned governance, technical debt, and talent gaps. Consider creating an AI review board, defining ethical guidelines, and setting clear processes for model evaluation and incident response. You'll also want to start aligning your systems, operations, and security policies with best practices for large language model operations (LLMOps). Set up an AI development platform like Vertex AI to provide a reliable and scalable environment for prototyping pilots and eventually deploying to production.

Phase 2: Pilot implementation (months 4–6)
With a solid foundation in place, you can move to the experimental phase and start building applications. Here, you'll focus on:

- Launching well-designed pilots that align with business objectives and balance impact with risk
- Developing rigorous success criteria to measure both technical and business outcomes

- Establishing leadership and organizational trust in AI outputs
- Creating systematic feedback loops to capture learnings and enable rapid iteration

Success requires balancing ambitious goals against practical limitations, allowing teams to learn and adapt quickly.

Phase 3: Strategic scaling (months 7-12)

Once your pilots prove successful, it's time to scale. In this phase, you'll expand what works to more teams and use cases, refine your processes based on what you've learned, and build your team's AI skills through training.

Invest in change management strategies and best practice documentation to effectively replicate early successes. At this point, you'll also have a clearer understanding of opportunities to improve your teams' AI skills and identify hiring gaps.

Phase 4: Broad adoption (months 13+)

The final phase focuses on embedding AI capabilities deeply into the organization's operations by:

- Refining systems and operations based on data and experience
- Building more complex AI applications to improve efficiency and customer experiences
- Scaling successful patterns across the organization
- Identifying additional use cases across departments, with new pilot implementations

You'll know your organization has achieved broad adoption when you see team members instinctively turn to AI-driven solutions to address inefficient processes or develop new customer experiences.

PILOT DESIGN

With this roadmap in mind, let’s take a closer look at what a successful pilot entails.

We’ve noticed that the most successful AI teams start small, choosing use cases that demonstrate value quickly before scaling up. While it’s tempting (and sometimes there’s pressure) to tackle the biggest opportunities first, a more measured approach typically yields better results. An ideal pilot strikes a balance: significant enough to demonstrate business value, yet contained enough to deliver results quickly.

Look for friction points in your business that AI could address. Prioritize use cases with enthusiastic sponsors, readily available data, and manageable compliance requirements. This will encourage rapid iteration and experimentation.

This checklist of ideal pilot characteristics can help you narrow things down:

- **Well-suited to LLM capabilities**, such as processing unstructured data, content classification, or format transformation.
- **Meaningful success metrics** that measure technical performance as well as impact on key business indicators, e.g., reduced processing time, increased throughput, or improved accuracy.
- **Clear return on investment**, which will help you build organizational confidence and broader adoption.
- **Business critical**, but low security risk, giving you room to establish governance frameworks and build institutional knowledge without jeopardizing critical operations.

- **Abundant data** to ensure clear test results, in the format needed, with the necessary permissions.
- **Minimal disruption** to existing processes, by having AI-enhanced processes running alongside existing workflows until performance and reliability are proven.
- **Scalable and replicable** if successful, allowing you to quickly expand to more business units.

Here are some use cases teams commonly choose for their initial pilots due to ease of implementation and significant business impact:

Example use cases		
External, revenue-oriented	**Internal, cost- and risk-oriented**	
Better customer experiences	**Increased team productivity & creativity**	**Optimize business processes**
Advanced chatbots: *Address customer issues and queries in real-time*	**Code generation:** *Rapidly create and optimize programming code*	**Fraud detection:** *Proactively identify and mitigate fraudulent activities*
Agent assist: *Deliver enhanced support levels without increasing headcount*	**Data analysis:** *Interpret complex datasets and generate insights*	**Document processing:** *Parse and summarize text to identify key information*

Graduation criteria

Getting AI buy-in from your organization hinges on your ability to demonstrate success in clear, quantitative ways. That's why you need to establish measurable graduation criteria for transitioning from pilot to production. Let's take a closer look at what your measurement plan should cover and best practices for success metrics.

A comprehensive set of graduation criteria spans these three dimensions:

Performance thresholds

- Accuracy metrics
- Speed improvements & latency metrics
- Cost efficiencies

Operational readiness

- System stability
- Support infrastructure
- Team capability

Risk management infrastructure

- Security compliance
- Data protection
- Operational controls

Running these metrics through the well-known SMART framework can help ensure your goals are effective:

1. **Specific**—for example, "target 95% accuracy in customer inquiry classification and reduce average resolution time from 45 to 30 minutes"
2. **Measurable**—for example, "less than 0.1% of outputs flagged for bias across 10,000 interactions" for more abstract criteria like ethical AI deployment

3. **Achievable**—for example, “build an AI support chatbot in the next quarter”
4. **Relevant**—for example, aligned with operational efficiency, revenue growth, or customer satisfaction goals
5. **Time-bound**—for example, “achieve in the next six weeks”

Going back to our example pilot use cases, here’s how their success metrics would differ:

Success metrics for different use cases				
Ticket routing	**Content moderation**	**Customer chatbot**	**Code generation**	**Data analysis**
↑ Routing accuracy rate	↓ False positive rate	↓ Cost per conversation	↓ Time spent on routine coding	↓ Time to insight
↓ Rerouting rate	↓ False negative rate	↑ Conversation completion rate	↓ Bugs and errors in code	↑ Decision accuracy
↓ Time to resolution	↑ Per-category accuracy	↓ Average time to resolution	↓ Project completion time	↑ Ability to handle larger and more diverse datasets
↓ Queue processing time	↓ User churn	↑ First contact resolution rate	↑ Adherence to coding standards	↑ Customer satisfaction with data-driven products or services
↓ Cost per ticket	↓ Appeal volume	↓ Escalations to human agents	↑ Developer productivity	↓ Routine task elimination
↑ CSAT	↓ Cost per review	↑ Percentage of users engaging with chatbot	↑ Code reuse	↓ Time saved per analysis
↑ Volume handling	↑ Community health	↑ CSAT	↑ Test pass rate	↑ Query complexity handling

SECURITY, DATA PRIVACY, AND COMPLIANCE

As you design your initial pilot, you'll want to start examining your technical systems and policies through an AI lens to ensure they address everything from data privacy to model security and compliance with regulatory requirements. With Vertex AI, you can build confidently using Google Cloud's robust built-in security, privacy, and compliance measures to securely scale your applications in production.

Key components to consider:

Data protection: Establish guidelines for data that can be used with LLMs and procedures for handling sensitive information. Review these areas:

- Encryption standards
- Access controls
- Privacy measures
- Continuous data quality management

Compliance management: Create processes to log LLM interactions for security audits and compliance checks. Research the following:

- Regulatory requirements
- Industry standards
- Internal policies

Monitoring & auditing: Determine who can deploy models and what approvals are needed. Enterprise controls, such as Vertex AI Model Garden's organization policy, provide the right access controls to make sure only approved models can be accessed.

Adopt tools that allow you to monitor your application's accuracy, fairness, and explainability. Set up these processes:

- Activity logging
- Performance tracking
- Compliance reporting

Network security: Consider the privacy of your connections. Employ private networking options like Private Service Connect to establish secure and isolated connections for your AI workloads and data traffic within your virtual private cloud, minimizing exposure to the public internet.

Stage 2: Create business value

After building a secure foundation and aligning with leadership on an initial pilot, it's time to build. Your goal at this stage is to demonstrate fast and clear business value that fosters confidence in the technology and encourages further experimentation.

The model you choose can greatly impact your ability to deliver results like increased operational efficiency, improved decision-making, and new revenue. Here's a closer look at when to use Anthropic's frontier models.

WHEN TO USE CLAUDE

Claude shines in applications that require state-of-the-art intelligence, sophisticated reasoning, code generation, or multistep instruction following. Our customers use Claude for everything from production-ready code, complex agent workflows, comprehensive web apps, end-to-end automation, and more. Whatever the use case, you also get Claude's natural conversation abilities and Anthropic's commitment to responsible AI.

CASE STUDY: REPLIT

Software development platform Replit wanted to create an AI agent that would enable users to build functional applications within minutes. After evaluating multiple AI models, the team chose Claude on Vertex AI for its code generation capabilities and cloud integration, including Cloud Run, Cloud SQL, and BigQuery.

Results: Replit Agent, powered by Claude Sonnet, now supports more than 100,000 applications through Cloud Run. Learn more

"Everything goes back to customer experiences. Having a better model and better technology means you can do things faster and cheaper, which customers love. Working with Anthropic and Google Cloud makes our agent more reliable and successful."

Michele Catasta, President, Replit

CASE STUDY: AES

Global energy company AES needed a more efficient way to conduct thousands of internal safety audits for their power-generating assets without sacrificing accuracy. The team used Claude to develop an AI-powered auditing system made up of different agents that handled document processing, task breakdown, and report generation.

Results: By leveraging Claude's natural language processing capabilities on Vertex AI, AES increased audit accuracy by

10-20%, accelerated audit results from two weeks to one hour, and reduced audit costs by 99% . Additionally, thanks to the integration with Vertex AI, they were able to accelerate the process of approving and deploying API calls. Learn more

"I love the accuracy of Anthropic's Claude models, and the security and advanced AI tools that Google Cloud provides."

Dr. Sean Otto, Senior Director of Data Science & Analytics, AES

SELECTING THE RIGHT CLAUDE MODEL FOR YOUR USE CASE

Choosing the right model depends on your specific needs. Here are the top factors to consider:

1. Balance of capabilities: Consider trade-offs between intelligence, speed, and cost.

2. Task complexity: For strong performance at scale, choose models that balance intelligence and cost. More complex tasks require more powerful models while simpler tasks or high-throughput scenarios can be done with lighter-weight, more compact models.

3. Response time: When near-instant responsiveness is essential, go with lightweight models for their speed.

4. Cost: Cost is generally a trade-off against capabilities. The more complex the task, the more you'll need to spend on a model to ensure consistently high performance.

5. Context window: Select models with at least a 200K token context window to process long documents and complex interactions.

Anthropic offers three Claude models—Opus, Sonnet, and Haiku—as well as an agentic coding tool, Claude Code, to assist with a wide range of needs. These solutions feature varying levels of intelligence and cost. Here's how these differences come into play based on use case:

Agentic coding

While not a model, Claude Code can search and read code, edit files, write and run tests, commit and push code to GitHub, and use command line tools.

Claude Sonnet can automate tasks throughout the software development lifecycle, from initial planning to bug fixes, maintenance, and large-scale refactoring. It can also be relied upon for instruction following, tool selection, error correction, and advanced reasoning in tool-dependent agentic workflows.

Coding

- Claude Sonnet performs well in tasks such as code migration, code fixes, and translations.
- Claude Haiku can deliver quick, accurate code suggestions and completions for real-time development workflows.

Content generation and analysis

- Claude Sonnet understands nuance and tone in content, generating more compelling content and analyzing on a deeper level.

Data extraction and labeling

- Claude Sonnet can extract detailed data from visual elements such as charts or graphs.

- Claude Haiku is optimized for rapid data extraction and automated labeling tasks.
- Claude Opus can process images to generate text output, and analyze complex visual content, such as charts, graphs, technical diagrams, and reports.

These are just a few of the nuances between the models. Claude can also power customer-facing agents, document Q&A, content moderation, end-to-end task automation, and many more use cases.

CASE STUDY: PALO ALTO NETWORKS

Palo Alto Networks, the world's largest cybersecurity company, recently set out to improve its developer productivity by using AI to mitigate critical issues in early product development stages. After evaluating multiple solutions, the team chose Claude on Vertex AI because it excelled at coding tasks while maintaining high accuracy and security standards.

Results: With Claude on Vertex AI, Palo Alto Networks achieved a 20-30% increase in feature development velocity and reduced new developer onboarding time from months to weeks. Learn more

"At Palo Alto Networks, we combine different Claude models to maximize developer productivity. We use Claude 3.5 Haiku for real-time code completion as our developers type, providing immediate assistance without workflow disruption. For more complex tasks like architecture discussions and sophisticated code generation, we leverage Claude 3.5 Sonnet. This combination keeps our developers in their flow state while ensuring they have the right level of AI assistance for each specific task."

Dr. Sean Otto, Senior Director of Data Science & Analytics, AES

Armed with the right model, you'll be well on your way to proving AI's impact on business value. Studies show that early LLM adopters have seen remarkable results:

- Customer support teams are responding **20-35%** faster to inquiries.
- Engineering teams are reducing coding time by **15%**.
- Content creators are working **30-50%** faster.
- Back office operations are **20-50%** more efficient.[1]

Perhaps the most striking, top performers are attributing over **10%** of earnings to their generative AI implementations.[2]

1. Bain & Company Technology Report 2024
2. McKinsey State of AI report 2024

Stage 3: Build for production

With your business objectives clear and the right Claude model selected, you're ready for Stage 3: building your first AI workflow that turns potential into reality.

Think of AI workflows as newly hired employees who need training, evaluation, and feedback to reach their full potential. Whether you're building customer-facing features or internal tools, here's everything to know about ensuring your Claude implementation runs reliably and scales smoothly on Vertex AI.

PROMPT ENGINEERING

Strong prompts are at the heart of your AI application. Think of them as input instructions for a model to produce the outputs you want. Take time to refine—or "engineer"—these prompts to ensure they produce the most relevant and accurate results.

Let's start by breaking down the main elements of a successful prompt:

1. Task and role content
2. Background data, documents and images
3. Detailed task description and rules
4. Conversation history (if applicable) or user input
5. Immediate task description or request
6. Output formatting
7. Pre-filling the response (if required)

Here's what our prompt might look like if we wanted Claude to help us classify customer support tickets.

System	1	The assistant will be acting as a customer support ticket classification system. The task is to classify the ticket according to the rules.
User	2	You will classify a customer support ticket into one of the following categories: `<categories>{{categories_list}}</categories>`
	3	Here are some important rules for the classification system: `<rules>{rules}</rules>`
	4	Here is the support ticket that you need to classify: `<ticket>{{ticket}}</ticket>`
	5	You should respond with the correct classification for the ticket in the requested format

	6	Put your response in the following format:
		`<response>`
		    `<category>Your classification choice goes here</category>`
		`</response>`

Assistant (prefill)	7	`<response><category>`

Pro tips:

- **Don't give up too soon.** Teams tend to give up on prompting prematurely and pivot to costlier solutions. A few hours of prompt engineering can often fix issues without requiring a bespoke model or fine-tuning that incurs extra costs to train and maintain.
- **Be clear and specific.** Remember, Claude is like a new employee who has no context about your organization, task requirements, and specific needs.
- **Iterate.** For example, try reformatting the prompt or adding files before instructions to see if responses improve.
- **Break down complex tasks into simpler prompts.** Smaller prompts, such as chain prompts or aggregate responses, can help you improve accuracy and debug issues more quickly.
- **Control the length of responses** to optimize processing time and reduce latency.

Facing a "blank page" problem? The Anthropic Console has tools to help you generate strong starting prompts and improve existing ones. Within Vertex AI, you can also use the Gen AI evaluation service to access prompt templates.

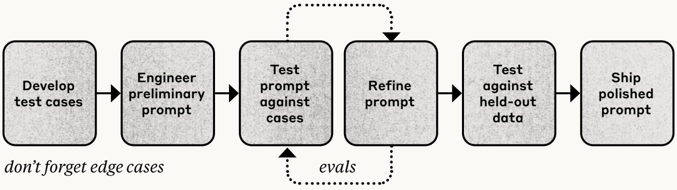
Check out these resources for more best practices:

- Anthropic's prompt engineering documentation and use case library (tip: chat with Claude directly in the search bar to get help with specific issues)
- Anthropic's code examples and interactive tutorials on core prompting techniques and production-level prompting
- Google Cloud's prompting strategies and prompt gallery.

EVALUATION

Once you've developed your prompt, you're ready to create evaluation tests that measure its performance against the success criteria that you've defined.

Don't rush this step—spending time here will result in a more effective production system. Without robust evaluation tests, you won't know if the changes you make have a positive or negative impact, and you will lack the signals you need to continue making future model upgrades.



Remember, effective evaluation tests are detailed, specific, and completely automatable. Consider using LLMs as a judge. Prioritize a higher volume of tests, even if the quality of individual tests is lower. You'll need a comprehensive suite of tests that assess multiple dimensions.

For example, our support ticket classifier should have tests that examine how the model handles:

- **Key issue types**: technical problems, billing questions, feature requests, etc.
- **Edge cases**: tickets that mention multiple issues or contain minimal information, requiring the model to demonstrate its reasoning capabilities
- **Customer personas**: varying communication styles, different levels of technical knowledge

Anthropic and Google Cloud both offer tools to help you develop comprehensive tests:

- Start with the Anthropic Console to leverage automated test case features optimized for Claude models. Its side-by-side tool lets you quickly compare the outputs of two or more prompts. Improve output by grading response quality on a 5-point scale and using the feedback to iterate on new prompt versions.
- Use Vertex AI's Gen AI evaluation service once you're ready to integrate evaluation into a broader workflow and deploy in a production environment. It will guide you in preparing datasets, running evaluations, and assessing results on an ongoing basis.

Learn more about building strong evaluations with our guide and course.

"Working with Google Cloud and Anthropic's Claude models helps us serve millions of Latin Americans. With Claude, we get a model that's tailored to the unique needs of Latin Americans. With Vertex AI, we get best-in-class AI infrastructure and tools to deploy to millions of users.."

Nicolás Loeff, CTO, BrainLogic AI

OPTIMIZATION

The results of your evaluation tests will help you determine what improvements are needed next. Here are two iteration strategies that should be part of your toolkit.

1. Few shot examples

This technique teaches Claude to perform the task by providing examples within the prompt. Providing examples is one of the most effective ways to improve output quality if you are already following other prompting best practices. Some tips:

- Provide examples directly in context or use Retrieval-Augmented Generation (RAG) for dynamic example insertion.
- Include edge cases in your examples.
- (Optional) Consider including examples showing how not to perform the task.
- If your dataset doesn't contain enough examples, just ask Claude to create more!

"Sometimes people think that generative AI is very costly, it's very expensive, but if you can use it correctly, like some fine tuning for example, that we do use in Bedrock, it gets cheaper than using traditional models."

Renan de Padua, Head of Generative AI, iFood

2. Chain of Thought (CoT)

This technique gives Claude space to "think out loud" when facing complex tasks like research, analysis, or problem-solving. CoT prompting encourages Claude to break down problems step by step, generating more accurate and nuanced outputs.

Pro tip: Due to how LLMs generate responses, CoT prompting is only effective if the model is given space to think out loud before it produces its final answer. Providing a rationale after the model has already given its answer generally does not improve its response over the baseline.

Benefits of letting Claude think

- **Accuracy:** Stepping through problems reduces errors, especially in math, logic, analysis, and complex tasks.
- **Coherence:** Structured thinking leads to more cohesive, well-organized responses.
- **Debugging:** Seeing Claude's thought process helps pinpoint unclear elements in your prompts.

If you're using the latest version of Claude Sonnet, you can turn on an extended thinking mode to have the model think longer for more complex problems and see exactly how it analyzes problems. We've seen improved performance in math, physics, instruction-following, coding, and other tasks when extended thinking is enabled.

However, keep in mind that increased output length may impact latency. Not all tasks require in-depth thinking. Use CoT judiciously to ensure the right balance of performance and latency. The latest Claude Sonnet model lets you specify how many tokens you want it to use for thinking.

Let's go back to our ticket routing example. Here's what the prompt looks like after adding few shot examples and CoT tags:

System	❶	The assistant will be acting as a customer support ticket classification system. The task is to classify the ticket according to the rules.
User	❷	You will classify a customer support ticket into one of the following categories: `<categories>{{categories_list}}</categories>`
	❸	Here are some important rules for the classification system: `<rules>{rules}</rules>`
	❹	Use the following examples to help you classify the ticket: `<examples>{{examples_list}}</examples>`
	❺	Here is the support ticket that you need to classify: `<ticket>{{ticket}}</ticket>`
	❻	You should respond with the correct classification for the ticket in the requested format
	❼	Think about your answer first before you respond in `<scratchpad>` tags. Consider all of the information provided and create a concrete argument for your classification.
	❽	Put your response in the following format: `<response>` `<scratchpad>Your thinking here</scratchpad>` `<category>Your classification choice goes here</category>` `</response>`
Assistant (prefill)	❾	`<response><category>`

❶ Task context
❷ Background data, documents & images
❸ Detailed task description & rules
❹ Examples
❺ Conversation history or user input
❻ Immediate task description or request
❼ Thinking step by step (CoT if applicable)
❽ Output formatting
❾ Pre-filled response (if any)

MORE COMPLEX APPLICATIONS

Much like the evolution of web applications from static pages to sophisticated distributed systems, AI implementations follow a similar journey of increasing capability and complexity. Successfully deploying AI in production environments typically follows a natural progression through these levels of technical maturity.

Basic: Getting started

The journey begins with straightforward implementations that focus on direct interactions between users and the AI model:

- Simple chat interactions with clear input/output patterns
- Basic prompt engineering to improve responses
- Direct model responses without complex processing
- Single-turn conversations without extensive context

This level is ideal for initial proof-of-concept projects and building team familiarity with AI capabilities. If you're not already on Vertex AI, consider adopting it at this stage to ensure you're building securely in an integrated environment to evaluate, deploy, and manage Claude-powered applications.

Intermediate: Adding intelligence

As organizations gain confidence and experience, they typically advance to more sophisticated implementations that enhance the AI's capabilities:

- Structured prompts and templates for consistent outputs
- Integration of basic tools for extended functionality
- Knowledge retrieval systems (RAG) for improved accuracy

- Multi-turn conversations with context management
- Basic workflow integration

At this stage, many organizations begin seeing significant business value as AI handles more complex tasks and integrates with existing processes.

Advanced: Building AI agents

The most sophisticated implementations transform AI from a simple query-response tool into an intelligent agent that can execute complex tasks autonomously:

- Multiple tool integration for diverse capabilities
- Complex multi-step workflows
- Agent-based systems with decision-making ability
- Advanced memory and context management
- Sophisticated error handling and self-correction

Common examples include agents that can browse the web, interact with APIs, or operate computer systems to accomplish tasks. Pro tip: Try Vertex AI's Agent Builder to build and deploy generative AI agents that use your organization's data.

UNDERSTANDING TOOLS AND AGENTS

As you progress in your AI journey, two concepts will become increasingly important: tools and agents. These capabilities represent the frontier of AI implementation, enabling systems that can not only understand requests but take action to fulfill them.

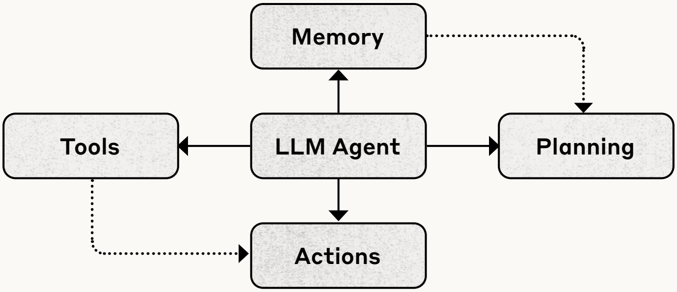
Tools

Tools give Claude the ability to do more than just respond with text. With tools, Claude can take action by connecting to your systems and applications to complete real tasks. Tools are the defined functions that the AI can call to perform specific actions. You can build tools that extend models with the following:

- **Retrieval**: Search and access information from databases, knowledge bases, or the internet
- **API integration**: Connect to external services like APIs, data sources, and business systems
- **Data processing**: Perform calculations, analyze data, or reformat content
- **Content creation**: Generate specialized content like images, code, or structured documents
- **Memory**: Store and retrieve information from previous interactions
- **Control flow**: Enable the model to implement logic, loops, and conditional operations
- **System interaction**: Interact with operating systems, file systems, and applications

Agents

An agent is an AI system that integrates an LLM with tools, decision-making framework, and memory systems to take action in both real-world and digital environments.
They are useful for open-ended tasks with unpredictable steps and scenarios that require both conversational engagement and action, such as customer support and coding.



Combining agents and tools with your company's systems opens up the opportunity for significantly more powerful AI applications that are capable of completing complex tasks such as:

- **Customer support automation**, where agents manage routine inquiries, access customer records, process refunds, and update tickets without human intervention.
- **Software development acceleration**, where coding agents solve GitHub issues, implement code changes across multiple files, and run tests automatically.
- **Personalized sales outreach**, where agents research prospects, access CRM data, tailor pitches, and schedule follow-ups.
- **Lead qualification**, where agents interact with potential customers, access product information, and use pricing systems to qualify leads around the clock and route them to sales.
- **Document processing**, where agents extract information from documents, validate data, and input data into internal systems

These are just some of the possibilities that you can unlock with advanced AI systems. Learn more about building effective agents with Claude and Vertex AI.

Stage 4: Deploy your application on Vertex AI

After building your AI application through careful prompt engineering, evaluation, and optimization, the final step is to bring it to your users. Whether you've created a basic interaction system or an advanced AI agent, proper deployment ensures your solution delivers value safely and reliably in real-world conditions.

Do

- Progressively roll out your application.
- Set up infrastructure for A/B testing.
- Design user-friendly ways for human feedback.
- Update your offline evaluations and prompts based on production data.
- Minimize latency to improve response time.
- Implement automated testing gates and rollback procedures.
- Run responses through safety filters.

Don't

- Replace your previous system right away.
- Treat offline evaluations as static.
- Make decisions based on a single evaluation test.
- Attempt implementation without expert guidance—partners like Anthropic can help!

CASE STUDY: AUGMENT

Augment set out to create an expert AI coding assistant that helps developers understand and work with intricate codebases. The team chose Claude on Vertex AI for its code understanding capabilities, combined with Vertex AI's enterprise-grade security and infrastructure reliability.

Results: Thanks to Claude on Vertex AI, Augment was able to switch their production site to Claude in a single weekend and implement secure protocols with SOC 2 Level 2 certification. Since then, their AI agent has processed millions of lines of code across hundreds of customer codebases. Learn more

"Everything stays inside Google Cloud. This makes our security story very clean and simple to explain to users."

Guy Gur-Ari, Chief Scientist, Augment

LLMOPS BEST PRACTICES

Over the years, distinct best practices and principles have emerged for deploying and managing LLMs, given their large size, complex training requirements, and higher computational demands.

Here are six critical areas to focus on:

1. Robust monitoring and observability

Comprehensive monitoring is critical to successful deployment. In addition to tracking basic metrics like response times and error rates, monitor LLM-specific concerns like token usage, semantic accuracy, and response relevance against established baselines. Capture and analyze user interactions and feedback to continuously refine performance. The key is to create a system that lets you see how your models are performing in production, so you can catch issues before they impact users.

Use tools like Vertex AI Model Monitoring to facilitate monitoring of your models.

2. Systematic prompt management

Like code, prompts need version control, testing, and proper documentation. Set up a central repository where teams can collaborate on prompts, track changes, and document design decisions. Implement a testing framework to validate prompts across different scenarios and maintain clear documentation of each prompt's purpose and expected behavior.

Tools like Vertex AI Studio can help you systematically track prompt versions and performance. Implement strict versioning control with clear rollback procedures and compatibility tracking.

3. Security and compliance by design

Build proper access controls, content filtering, and data privacy measures from the outset. Specify who can deploy models and how models are monitored for compliance. Set clear policies for authenticating and authorizing both access and deployment.

To protect your data, implement minimization, anonymization, and privacy-preserving techniques. Develop pre- and post-processing filters to prevent harmful content generation and sensitive data leakage. Ensure compliance with relevant regulations like GDPR, HIPAA, or industry-specific requirements through comprehensive logging of LLM interactions, systematic controls, and documentation. Leverage Private Service Connect for secure, private connections to the Vertex AI API

4. Scalable infrastructure and cost management

Design your LLM infrastructure with scalability in mind, but balance this with cost efficiency. Implement multi-level caching strategies for responses, embeddings, and computations. Consider serverless deployment to automatically manage scaling without maintaining infrastructure.

Deploy different models based on task complexity. Optimize prompt design and response generation to minimize token usage while maintaining output quality. Dynamically adjust compute resources based on demand patterns to optimize costs. Maintain a central model registry, like Vertex AI Model Registry, to track models, versions, deployment status, and performance metrics.

Consider Provisioned Throughput for production grade, large workloads.

5. Continuous quality assurance

Automate testing of model outputs for quality and consistency. Monitor for hallucinations and validate responses against specific business requirements. Establish feedback loops with end-users and maintain clear processes for addressing quality issues when they arise. For critical applications, incorporate human review, using techniques such as random sampling and edge case analysis. When updates are made to models or prompts, automatically test them for regressions against previous benchmarks. Leverage Vertex AI Model Evaluation tools to automatically assess and validate your models

Each of these practices supports the others – for instance, good monitoring helps inform both quality assurance and cost management, while systematic prompt management contributes to better security and quality. The key is implementing them as part of a coherent strategy rather than as isolated initiatives.

6. Continuous integration and continuous deployment (CI/CD)

Accelerate and improve release reliability by automating data preparation, code integration, testing, deployment, and application monitoring. Use tools like Vertex AI Pipelines to orchestrate the entire development lifecycle. Implement automated testing gates to validate model performance, security, and compliance before advancing to the next deployment stage. Establish gradual rollout strategies to minimize risk when deploying new models or prompt versions. Develop automated rollback procedures for models that fail to meet performance or safety thresholds.

With these best practices in place, you can be confident about your application's security, compliance, and performance. Each of these practices supports the others—for instance, good monitoring helps inform both quality assurance and cost management, while systematic prompt management contributes to better security and quality. The key is implementing them as part of a coherent strategy rather than as isolated initiatives—which is why a platform like Vertex AI for integrated prompt management, evaluation, and CI/CD capabilities is critical for successful LLM deployments.

> **"The adoption of GenAI has helped us develop thousands of experiences and itineraries for 80% less cost than it would if we were to have our writers manually curate those. Furthermore, our writers are now freed up to go focus on finding the next in-destination thing and continue to inspire travelers everywhere."**
>
> Chris Whyde, Senior VP of Engineering at Lonely Planet

Anthropic + Google Cloud

Together, Anthropic and Google Cloud provide a powerful AI solution for startups and enterprises. You benefit from the combined strengths of Anthropic's powerful models and Google Cloud's fully managed, unified AI development platform for building and scaling generative AI applications.

Choosing to work with Anthropic and build on Claude means your AI applications are not only capable but also reliable, safer, aligned with human values, and directly integrated with your existing cloud infrastructure. With Claude on Vertex AI, your teams can focus on AI innovation, not infrastructure.

Reach out to the Anthropic sales team to learn how they can partner with you on a successful AI strategy. To get started with Claude on Vertex AI, contact anthropic-gtm@google.com.

"Claude delights our customers worldwide with its intelligence, versatility, and human-like conversational abilities. It powers millions of daily interactions, covering everything from professional to creative purposes. Combining Claude with Vertex AI's enterprise-grade security and scalability helps us ensure every user interaction is fast and reliable."

Spencer Chan, Product Lead, Poe by Quora

APPENDIX

Essential AI Terminology

Tokens: The fundamental units of text that language models process, typically representing word fragments or individual characters (averaging about 4 characters per token in English).

Sampling: The probabilistic process by which an AI model selects its next output token from a distribution of possible choices, with parameters like temperature controlling the randomness of selections.

Pretraining: The initial phase of AI model development where the model learns general language understanding and capabilities from massive datasets before any specialized training occurs.

Fine-tuning: The process of further training a pre-trained model on specific datasets to enhance its performance for particular tasks or domains.

Supervised Learning: A training approach where the model learns from labeled examples, systematically mapping inputs to their correct outputs based on human-provided training data.

Preference Model: A component of modern AI systems that ranks potential outputs based on desired characteristics, helping align the model's behavior with intended outcomes.

Reinforcement Learning: A training method where the model learns optimal behavior through a system of rewards and penalties, improving its performance through trial and error.