cognient

Generative AI Handbook

Ideas and tips for how and where to start with generative AI in your organisation

Author: Jonas Ulander, Senior Consultant, Cognient

Contributors: Ola Ekman, Senior Consultant, Cognient; Ola Ekman, Senior Consultant, Cognient; Ola Ekman, Senior Consultant, Cognient

Generative AI will reshape every industry. But when?

Generative AI is the most significant technological shift since the internet. It changes how we write, how we code, how we design, how we make decisions. The question is not whether it will reshape your industry. The question is when, and how fast.

Some organisations are already deploying it in production, cutting costs, and building capabilities that will be very difficult to match later. Others are still in pilot mode, waiting for more clarity before they act. The gap between these two groups will widen every quarter.

This handbook is for those who want to move from pilot mode to production. It covers the landscape, the basics, applications, and how to build responsibly. It is written for people who need to make decisions about generative AI, not for those who want to read about it for fun.

Landscape

Understanding the landscape is prerequisite to making good decisions. This section covers what generative AI is and why it matters, the main types of models and what they can do, and the key factors that determine which model is right for your use case.

Why we are at a tipping point

Foundation models have crossed a threshold. Models like GPT-4, Claude, and Gemini can now reliably perform tasks that previously required specialised machine learning models or human expertise. Writing, coding, analysis, summarisation, translation, and reasoning all fall within their capabilities. The quality is high enough that organisations are deploying them in production, not just in pilots.

Infrastructure has matured. Cloud providers offer managed access to these models through APIs, with enterprise-grade security, compliance, and data handling. The tooling around prompt engineering, retrieval augmentation, and evaluation has developed rapidly. Organisations can now build on this infrastructure without starting from scratch.

Adoption is accelerating. Early movers are already seeing measurable impact in customer service, software development, and internal knowledge management. The competitive dynamics are shifting as those who have deployed successfully pull ahead of those who have not.

Costs are falling. The cost per token has dropped significantly over the past two years and continues to fall. Models that were prohibitively expensive for production use a year ago are now viable for many applications.

Open models are closing the gap. Open-weight models from Meta, Mistral, and others are now competitive with proprietary models for many tasks. This gives organisations more options for deployment, data handling, and customisation.

Where AI is headed

- Integration into everyday tools
- Agentic systems that can plan and act
- Multimodal models (text, image, audio, video)
- Specialised models for specific domains
- Open-source alternatives maturing

Matching the AI motives

Pre-trained models (LLMs)

- Text generation
- Summarisation
- Translation
- Code generation
- Conversation
- Reasoning
- Multimodal (text + image)

Fine-tuned models

- Classification
- Sentiment analysis
- Entity extraction
- Domain-specific generation
- Tone and style matching

Embedding models

- Semantic search
- Similarity matching
- Clustering
- Recommendation
- Anomaly detection

Multimodal models

- Image generation
- Image understanding
- Speech-to-text
- Text-to-speech
- Video generation

Agentic systems

- Tool use
- Multi-step reasoning
- Planning
- Autonomous execution

Basics

Understanding the basics of generative AI helps you make better decisions about what to build, what to buy, and what to avoid. This section covers how these models work, what they can and cannot do, and the key concepts you need to know before you start building.

Known strengths of generative AI

Text generation. Drafting documents, emails, reports, and any other written output. The quality is high enough for most business writing tasks.

Summarisation. Condensing long documents into shorter versions while preserving the key information. Useful for meeting notes, research, and document review.

Translation. Converting text between languages with high accuracy. Useful for multilingual teams and customer-facing content.

Code generation. Writing, debugging, and explaining code across many programming languages. Useful for developers and non-developers alike.

Conversation. Maintaining context across multi-turn interactions. Useful for chatbots, assistants, and interfaces where users need to iterate.

Reasoning. Following multi-step instructions, comparing options, and drawing conclusions from provided information. Useful for analysis and decision support.

Multimodal (text + image). Understanding images and generating text about them. Useful for document processing, visual inspection, and accessibility.

Known weak points of generative AI

Factual accuracy. Models can generate plausible-sounding but incorrect information. They do not have access to the world unless you provide it.

Mathematical reasoning. Models struggle with precise calculations and multi-step mathematical reasoning. Tool use can mitigate this.

Long-term memory. Models have no memory between sessions by default. Each conversation starts from scratch unless you provide context.

Consistency. Outputs can vary between runs even with the same input. Temperature settings control this, but perfect consistency is not guaranteed.

Recent information. Models trained on data up to a cutoff date do not know what happened after that. Retrieval augmentation can address this.

Proprietary knowledge. Models do not know your organisation, your products, your customers, or your internal processes. You must provide this context.

Adversarial robustness. Models can be manipulated by carefully crafted inputs to bypass safety guidelines or produce unintended outputs.

Bias. Models reflect patterns in their training data, including biases. Outputs should be reviewed for fairness in sensitive applications.

Open AI vs open question

Open AI

- Open-weight models
- Open-source tooling
- Open datasets
- Open research

Open question

- Will open models close the gap with proprietary models?
- How will licensing evolve?
- Who will control the infrastructure layer?
- How will regulation shape access?

Applications

Generative AI can be applied across many functions. This section covers the main categories of use cases, from customer-facing applications to internal tools and infrastructure. Each section includes examples and key considerations.

1. Market and competitive intel

Examples

- Competitive analysis
- Market research
- Trend monitoring
- Sentiment analysis
- Lead generation
- Pricing intelligence

Tools

- Perplexity
- Deep research modes
- Custom GPTs
- Web search integrations
- Scraping + LLM pipelines

Key considerations

- Verify sources before acting on outputs
- Combine with proprietary data for competitive advantage
- Automate the collection, human review the conclusions

2. Software development

Examples

- Code generation
- Debugging
- Code review
- Documentation
- Test generation
- Refactoring

Tools

- Cursor
- Claude Code
- GitHub Copilot
- Codex
- Custom agents

Key considerations

- Review all generated code before production
- Security and licensing implications
- Maintainability of AI-generated code

3. Production

Examples

- Marketing
- Sales
- Customer support
- Internal knowledge
- Documentation
- Training

Tools

- Custom GPTs
- Knowledge bases
- Retrieval augmentation
- Workflow automation
- Chatbots

Key considerations

- Start with internal use before customer-facing
- Ground responses in your own data
- Measure impact before scaling

Delivering at scale

Examples

- Document processing
- Data extraction
- Classification
- Translation
- Quality control
- Reporting

Tools

- Batch APIs
- Structured output
- Embeddings
- Fine-tuned models
- Evaluation frameworks

Key considerations

- Cost per unit matters at volume
- Latency requirements vary by use case
- Monitor quality continuously, not just at launch

5. Expert services

Examples

- Legal
- Medical
- Financial
- Technical
- Scientific
- Regulatory

Tools

- Domain-specific models
- Retrieval over expert knowledge
- Citation tracking
- Human review workflows

Key considerations

- Liability remains with the human expert
- Accuracy requirements are high
- Models should augment, not replace, expert judgment

6. Customer experience

Examples

- Chatbots
- Voice assistants
- Personalisation
- Recommendations
- Feedback analysis

Tools

- Conversational AI
- Sentiment analysis
- Recommendation engines
- Speech-to-text
- Text-to-speech

Key considerations

- Disclosure: customers should know they are interacting with AI
- Escalation to human must be easy
- Tone and brand consistency matter

7. Customer leadership

Examples

- Dashboards
- Reporting
- Forecasting
- Scenario analysis
- Decision support

Tools

- Data + LLM pipelines
- Natural language query
- Automated reporting
- Alerting systems

Key considerations

- Ground decisions in data, not model intuition
- Human judgment remains the final layer
- Transparency in how conclusions are reached

Responsible AI

Deploying AI responsibly is not optional. It is the difference between systems that build trust and systems that create liability. This section covers the principles that matter in practice, the risks you need to manage, and the governance structures that make responsible deployment possible.

Deploying your AI in a responsible and effective way

Governance and accountability

- Clear ownership of AI systems and their outcomes
- Defined escalation paths when things go wrong
- Regular review of deployed systems against their intended purpose
- Documentation of decisions, assumptions, and limitations

Transparency

- Users should know when they are interacting with AI
- Outputs should be explainable to the people who rely on them
- Training data and model choices should be documented where possible

Fairness and bias

- Test outputs across different groups before deployment
- Monitor for bias in production, not just at launch
- Correct for bias in training data where it affects outcomes

Privacy and data handling

- Know what data leaves your organisation and where it goes
- Prefer models that do not retain your data by default
- Anonymise or pseudonymise sensitive data before processing

Security

- Protect against prompt injection and adversarial inputs
- Limit what the model can access and what it can do
- Audit logs for all AI-assisted decisions in sensitive domains

Human oversight

- Humans should review outputs in high-stakes domains
- Automation should augment judgment, not replace it
- Escalation to human must be easy and fast

Guardrails and human oversight

Guardrails keep models within boundaries.

- Input filtering: block harmful or out-of-scope requests before they reach the model
- Output filtering: check responses before they reach the user
- Topic boundaries: define what the model should and should not discuss
- Tone and brand: keep responses consistent with your organisation's voice
- Rate limits: prevent abuse and control costs

Human oversight is the final layer.

- Review high-stakes outputs before they are acted on
- Sample low-risk outputs periodically to catch drift
- Escalation paths must be clear and fast
- Humans should be able to override any AI decision

Why both matter.

- Guardrails alone are not enough: models can still produce harmful outputs within boundaries
- Human oversight alone is not scalable: you cannot review everything at production volume
- Together they form a system that can be audited, improved, and trusted

Implementation checklist

- Input filtering
- Output filtering
- Topic boundaries
- Tone and brand
- Rate limits
- Escalation path
- Human review
- Audit logs

Transparency and automation at the heart of LLMOps

Transparency

- Log every prompt and response
- Track which model version produced which output
- Record retrieval results and context used
- Make decisions auditable by humans

Automation

- Automated evaluation runs on every change
- Regression tests against known inputs
- Alerts when quality metrics drop below thresholds
- Cost tracking per feature and per user

Why both matter

- Transparency without automation does not scale
- Automation without transparency creates black boxes
- Together they form a system that can be improved and trusted

Quality assurance and testing systems in a gen AI world

Traditional software testing checks whether the system does what it was designed to do. Unit tests, integration tests, and end-to-end tests verify that inputs produce expected outputs.

Generative AI adds new dimensions.

- Output quality is subjective and context-dependent
- Models can drift without any code change
- Small prompt changes can have large effects
- Cost and latency vary with usage patterns

Evaluation frameworks.

- Define what "good" means before you build
- Use automated evaluation for scale
- Use human review for edge cases and calibration
- Track quality over time, not just at launch

Regression testing.

- Maintain a set of known inputs and expected outputs
- Run evaluations on every prompt change and model update
- Alert when quality drops below thresholds

Monitoring in production.

- Log every prompt and response
- Track latency, cost, and error rates
- Sample outputs for human review
- Collect feedback from users where possible

Readiness

Deploying generative AI successfully depends on more than choosing the right model. This section covers the practical questions you need to answer before you start building: what data you have, what infrastructure you need, and what skills your team must develop.

Practical guidance to reap the benefits of gen AI

Start with a clear problem.

- Define the task before you choose the model
- Measure the baseline you are trying to improve
- Set success criteria before you build, not after

Ground in your own data.

- Retrieval augmentation connects models to your documents
- Quality of retrieval determines quality of output
- Keep your knowledge base current

Build evaluation into the process.

- Define what "good" means before you build
- Use automated evaluation for scale
- Use human review for edge cases and calibration

Start small, measure, iterate.

- Ship a narrow version first
- Measure real usage and real outcomes
- Expand only after the core works reliably

Plan for cost and latency.

- Model choice affects both
- Caching and batching can reduce cost significantly
- Latency requirements vary by use case

Keep humans in the loop where it matters.

- Review high-stakes outputs before they are acted on
- Sample low-risk outputs periodically
- Escalation paths must be clear and fast

Promoting AI stories in enterprise-grade gen AI tools

Why this matters.

- Adoption spreads when people see what others have built
- Internal stories reduce the gap between pilot and production
- Shared examples save time and prevent duplicate work

What to share.

- Problem solved
- Approach taken
- Tools used
- Outcome measured
- Lessons learned

How to share.

- Internal showcases
- Documentation
- Demos
- Communities of practice

Design a path to scale successful POCs

Pilot.

- Ship a narrow version to real users
- Keep the scope small enough to learn quickly

Evaluate.

- Measure against the baseline you set before you started
- Collect both quantitative and qualitative feedback

Harden.

- Add guardrails, monitoring, and error handling
- Address the gaps that the pilot revealed

Ship.

- Deploy to production with monitoring in place
- Start with a limited rollout if the risk is high

Measure.

- Track usage, quality, cost, and user feedback
- Compare against the baseline

Iterate.

- Fix what broke
- Improve what worked
- Decide what to scale next

Identify opportunities to add AI to the development process

Code review.

- Automated first pass before human review
- Catch common issues and style violations

Documentation.

- Generate from code and comments
- Keep it current as the code changes

Test generation.

- Unit tests from function signatures
- Edge cases from specs

Debugging.

- Explain error messages
- Suggest fixes for common patterns

Refactoring.

- Suggest improvements
- Apply changes with review

Code generation.

- Boilerplate
- Scaffolding
- Implementation from specs

Design.

- Architecture options
- Trade-off analysis

Planning.

- Estimate effort
- Identify risks

Requirements.

- Draft from conversations
- Clarify ambiguities

Modernise and significantly automate core business operations

Why this matters.

- Legacy systems slow down every other initiative
- Manual processes scale poorly and create errors
- The gap between what your competitors can do and what you can do widens every quarter

Where to start.

- Map the processes that consume the most time
- Identify the steps that are rule-based and repetitive
- Start with one process, not all of them at once

What to automate.

- Data entry and extraction
- Document processing
- Report generation
- Routine decisions with clear criteria

What to keep human.

- Exceptions that require judgment
- Relationships that depend on trust
- Decisions with high stakes and low data

Set risk appetite with your leaders

Why this matters.

- Without a shared risk appetite, every decision becomes a negotiation
- Teams need to know what they can ship without asking permission
- Leaders need to know what they are accepting when they say yes

What to define.

- Which data can leave the organisation
- Which models are approved for production use
- Which use cases require human review before deployment
- Which use cases are off limits

How to start.

- Write a one-page policy that covers the basics
- Review it with legal, security, and the teams that will use it
- Update it as you learn what works and what does not

Why this matters.

- Without a shared risk appetite, every decision becomes a negotiation
- Teams need to know what they can ship without asking permission
- Leaders need to know what they are accepting when they say yes

What to define.

- Which data can leave the organisation
- Which models are approved for production use
- Which use cases require human review before deployment
- Which use cases are off limits

How to start.

- Write a one-page policy that covers the basics
- Review it with legal, security, and the teams that will use it
- Update it as you learn what works and what does not

Why this matters.

- Without a shared risk appetite, every decision becomes a negotiation
- Teams need to know what they can ship without asking permission
- Leaders need to know what they are accepting when they say yes

What to define.

- Which data can leave the organisation
- Which models are approved for production use
- Which use cases require human review before deployment
- Which use cases are off limits

How to start.

- Write a one-page policy that covers the basics
- Review it with legal, security, and the teams that will use it
- Update it as you learn what works and what does not

Resources

- Anthropic Prompt Engineering Guide
- Anthropic Claude Code Best Practices
- Anthropic Claude Code Sub-agents
- Anthropic Claude Code Hooks
- Anthropic Claude Code Skills
- Anthropic Claude Code Plugins
- Anthropic Claude Code Agent Teams
- Anthropic Claude Code Ralph Wiggum Plugin
- Anthropic Claude Code Ralph Wiggum Technique
- Anthropic Claude Code Ralph Wiggum Explained
- Anthropic Claude Code Ralph Wiggum Guide
- Anthropic Claude Code Ralph Wiggum Loop
- Anthropic Claude Code Ralph Wiggum Cursed Version
- Anthropic Claude Code Ralph Wiggum Cursed Version Explained
- Anthropic Claude Code Ralph Wiggum Cursed Version Guide
- Anthropic Claude Code Ralph Wiggum Cursed Version Loop
- Anthropic Claude Code Ralph Wiggum Cursed Version Technique
- Anthropic Claude Code Ralph Wiggum Cursed Version Plugin
- Anthropic Claude Code Ralph Wiggum Cursed Version Best Practices
- Anthropic Claude Code Ralph Wiggum Cursed Version Tips
- Anthropic Claude Code Ralph Wiggum Cursed Version Tricks
- Anthropic Claude Code Ralph Wiggum Cursed Version Hacks
- Anthropic Claude Code Ralph Wiggum Cursed Version Secrets
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Explained
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Guide
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Loop
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Technique
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Plugin
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Best Practices
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Tips
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Tricks
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Hacks
- Anthropic Claude Code Ralph Wiggum Cursed Version Revelations Secrets