

Context Engineering Fundamentals

The art of optimizing AI input for superior output

Compaction

Maximize information density within token limits

Structured Note-taking

Organize knowledge for AI consumption

Sub-agent Architectures

Specialized prompting for complex tasks

Made with GAMMA

Why Context Engineering Matters

Transforms Claude

From a tool to a collaborative developer partner

Ensures Consistency

High-quality outputs across all interactions

Reduces Friction

Fewer iteration cycles and context switching



Made with GAMMA



Compaction Techniques

Fitting more context without losing clarity

Use Concise Names

Variable names and abbreviations Claude understands

Reference Documents

Compressed project knowledge libraries

Context Libraries

Reusable prompt components

Strip Verbosity

Essential patterns only

Example Result: 60-70% token reduction while maintaining full specification

Made with GAMMA



Advanced Compaction Patterns

Architecture Specs

API:REST|Auth:JWT|DB:Postgres|Cache:Redis:60s

Code Patterns

ErrorHandling:try-catch-log-rethrow

Validation:zod-schemas

Dependencies

Core:Next14,TS5,Tailwind

DB:Prisma,PG14

Key Principle: Create consistent abbreviation language Claude can decode

Made with GAMMA

Structured Note-taking

Organizing project context for optimal AI comprehension

1

Architecture

MVC with service layer, state management patterns

2

Conventions

TypeScript strict mode, component patterns, error handling

3

File Structure

API endpoints, shared utilities, atomic components

4

Critical Rules

No direct DB access, loading states, input validation

Made with GAMMA

Context Layering Strategy

Building context from general to specific prevents information overload

Layer 1: Foundation

Project type, stack, design pattern

Layer 2: Technical Details

API design, authentication, data layer

Layer 3: Task Context

Current work, dependencies, constraints



Made with GAMMA

Sub-agent Architectures

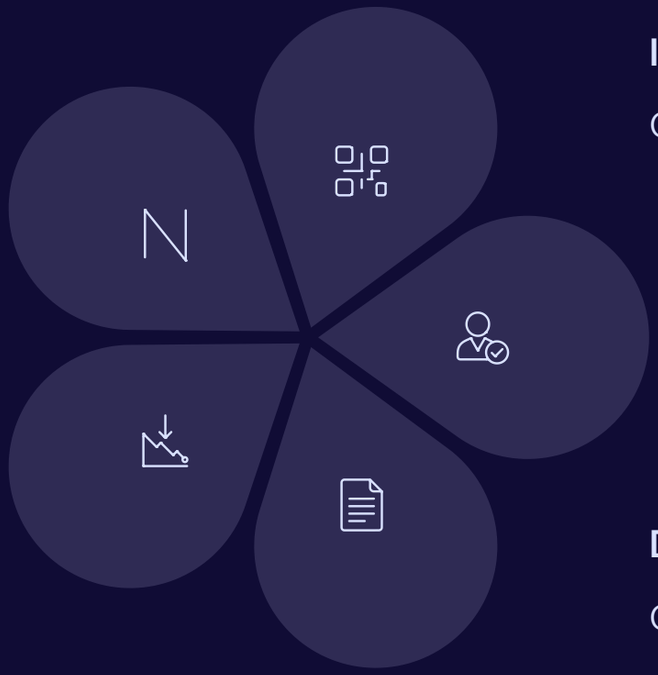
Breaking complex tasks into specialized expert prompts

Architect Agent

Design and structure

Specialist Agents

Performance, security, DevOps



Implementation Agent

Code generation

Review Agent

Quality and security

Documentation Agent

Comments and docs

Made with GAMMA



Maintaining Project Conventions

Solving the consistency challenge through anchoring and reinforcement

Convention Anchoring

Define error formats, API responses, naming patterns

Reference Reinforcement

Link to existing code examples and patterns

Drift Prevention

Include examples, reference files, periodic style checks

Made with GAMMA



Best Practices & Advanced Tips

1 **Front-load Critical Information**

Most important context first for maximum impact

2 **Use Consistent Terminology**

Build a project glossary for clarity

3 **Prefer Examples Over Explanations**

Show, don't tell—demonstrate patterns

4 **Layer Complexity Gradually**

Start simple, add detail as needed

Pro Tip: Create a context-engineering.md file in each project with compacted contexts, successful prompts, and learned patterns—treat it as living documentation.

Made with GAMMA

Thank You

Questions & Discussion

- Thank you for your attention
- Hope these context engineering techniques help improve your AI workflows
- Feel free to reach out with questions or share your own experiences

Created by Julkar Naen Nahian

Made with GAMMA