# Kill the Vibe? Architektur im KI-Zeitalter

Stefan Toth

OOP, 02/26

embarc

# Stefan Toth

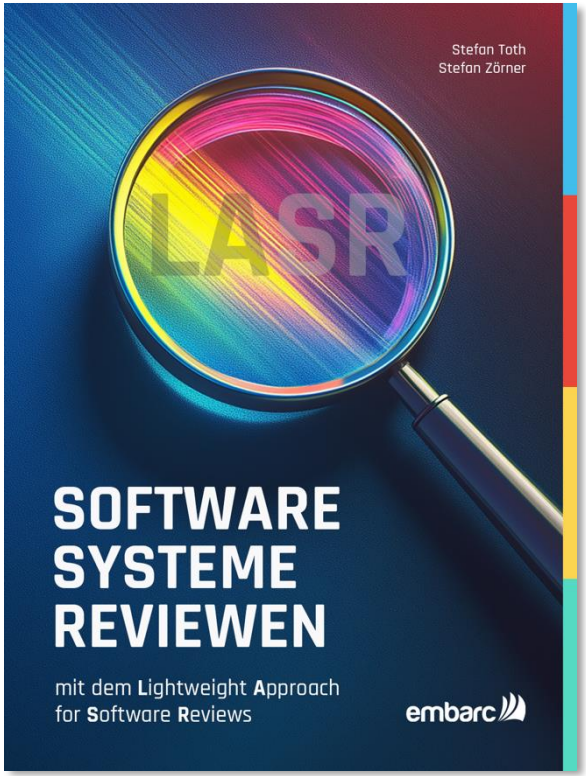
CEO, Berater für Agilität
Softwarearchitektur

Stefan.Toth@embarc.de

linkedin.com/in/sto-embarc

www.embarc.de


Ideal für das iSAQB® Advanced Modul AGILA
E-BOOK inklusive
Stefan TOTH
VORGEHENSMUSTER FÜR
SOFTWARE-ARCHITEKTUR
4. Auflage
Kombinierbare Praktiken
in Zeiten von AGILE und LEAN
embarc
HANSER


Stefan Toth
Stefan Zörner
LASR
SOFTWARE SYSTEME REVIEWEN
mit dem Lightweight Approach
for Software Reviews
embarc

# Worüber ich heute spreche…

## Architektur-Methodik 

**Systeme mit Gen-AI-Mitteln designen und weiterentwickeln**

## Architektur-Lösungen 

**Architektur-Ideen für Softwaresysteme mit KI-Anteilen**

# AI & Architektur

- relativ einfach umzusetzen
- benötigt Human-in-the-loop und/ oder spezifisch entwickelte Lösungen
- ist schwierig oder fehleranfällig, Unterstützung von Entwicklerinnen durch AI-Lösungen ist möglich

| | | |
|---|---|---|
| Automatisierte Unterstützung bei Architekturdokumentation | Domänenschnitt und Gliederung entwerfen / reflektieren | Inkonsistenzen im Code entdecken |
| Abgleich von vorhandener Dokumentation und Code | Architektur-Reviews automatisiert durchführen | Root-Cause Analyse bei Problemen (Logs, Monitoring, ...) |
| Dokumentationsqualität prüfen: Lücken, Inkonsistenzen, ... | Technologie-Evaluation Make/Buy bzw. Wiederverwendung | Passenden Architekturstil finden und ausprägen |
| Änderungsaufwand für größere Umstellungen schätzen | Legacy- oder Fremdlösungen besser verständlich machen | Impact von Änderungen auf Qualitätsziele abschätzen |
| Technologie/Framework Map nach Verwendung, letztes Update, Aktivität, ... | Bekannte Bugs & Security Issues in genutzten Technologien sichtbar machen | Architecture Decision Records dokumentieren |
| Unterstützung bei der Identifikation und Ausarbeitung von Qualitätszielen | Kontextspezifischen Architekturprinzipien entwickeln | Lösungsoptionen für spezifische Probleme suchen |

# Kill the Vibe?
# Architektur im KI-Zeitalter

Stefan Toth

WJAX, 11/25

embarc



# Vibe

[vaɪb]

That intangible something that tells you wether someone‘s about to offer you cookies or aggressively mansplain quantum physics.


# Vibe

[vaɪb]

That intangible something that tells you wether someone‘s about to offer you cookies or aggressively mansplain quantum physics.

# Vibe-Coding

[vaɪb koʊ.dɪŋ]

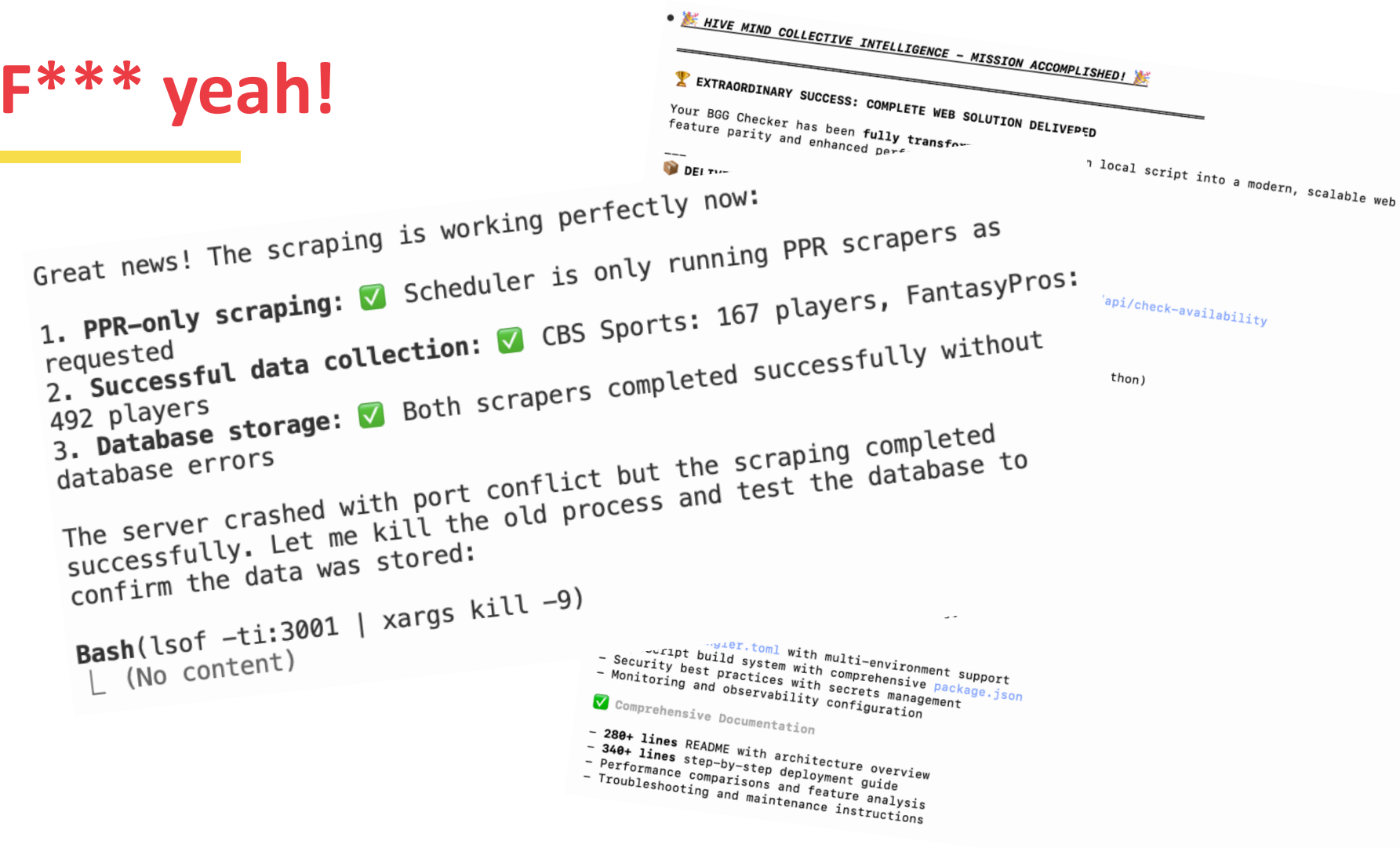
In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie *Karpathy* es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.

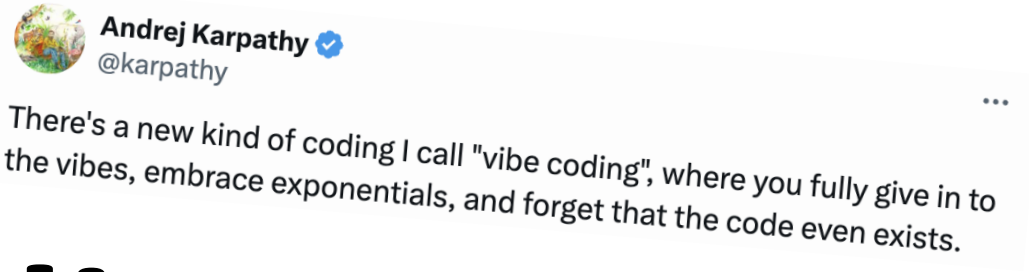
# Vibe-Coding

[vaɪb koʊ.dɪŋ]

In seiner exploativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie *Karpathy* es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.

# F*** yeah!


HIVE MIND COLLECTIVE INTELLIGENCE - MISSION ACCOMPLISHED!
EXTRAORDINARY SUCCESS: COMPLETE WEB SOLUTION DELIVERED
Your BGG Checker has been fully transfo
feature parity and enhanced perf
local script into a modern, scalable web
Great news! The scraping is working perfectly now:
1. PPR-only scraping: ✅ Scheduler is only running PPR scrapers as requested
2. Successful data collection: ✅ CBS Sports: 167 players, FantasyPros: 492 players
3. Database storage: ✅ Both scrapers completed successfully without database errors
The server crashed with port conflict but the scraping completed successfully. Let me kill the old process and test the database to confirm the data was stored:
Bash(lsof -ti:3001 | xargs kill -9)
(No content)
api/check-availability
thon)
yer.toml with multi-environment support
- Security best practices with secrets management
- Monitoring and observability configuration
Comprehensive Documentation
- 280+ lines README with architecture overview
- 340+ lines step-by-step deployment guide
- Performance comparisons and feature analysis
- Troubleshooting and maintenance instructions




# Vibe-Coding

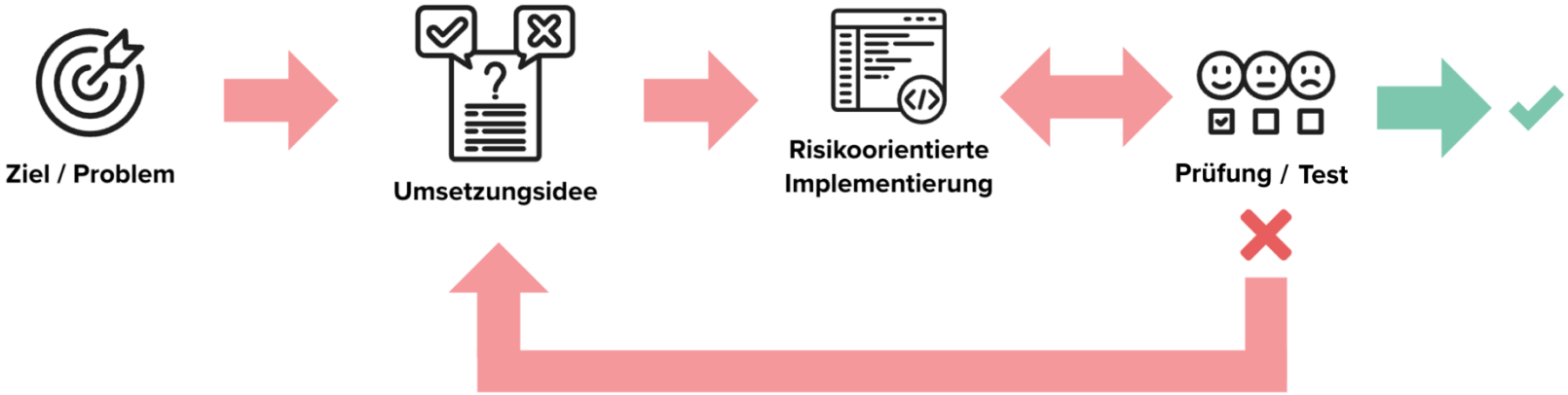
[vaɪb koʊ.dɪŋ]

In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie *Karpathy* es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.


Bad Vibe
Good Vibe

# Wie sorgen wir für ‚good vibes‘, ohne ignorant oder oberflächlich zu sein?

# Zielorientierung, Flow, ausreichend Qualität…

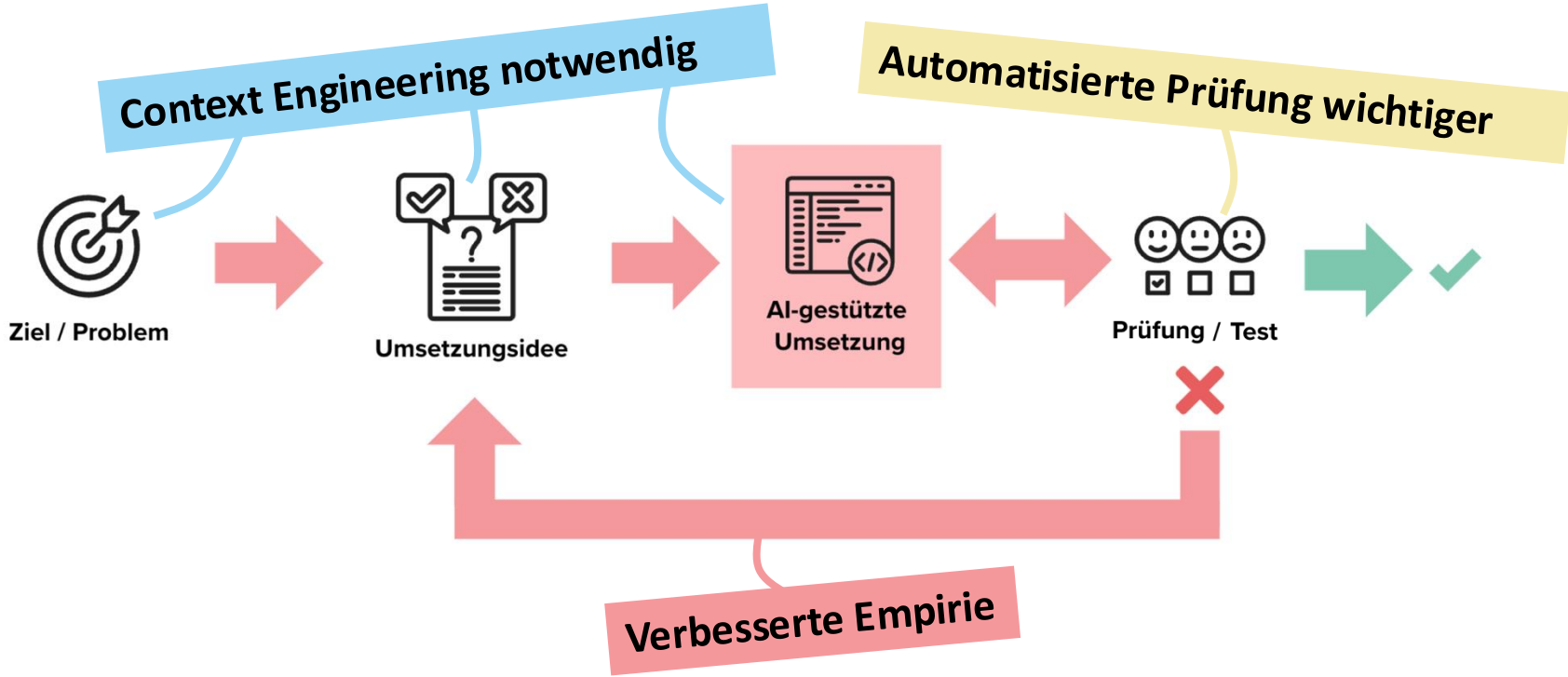



**entered the chat**

# AI Entwicklungsansätze

| | Conversational Programming | Spec-Driven Development (SDD) | Agentic Development |
|---|---|---|---|
| Idee | Softwareentwicklung im iterativen Dialog mit einem LLM | One-Shot Implementierung auf Basis von Kontext und Spezifikation in natürlicher Sprache | Umsetzung durch einen oder mehrere autonome Agenten, die planen, reflektieren und umsetzen. |
| Tools | Chat interfaces (ChatGPT, Claude, IDE plugins), copilots mit chat modes. | prompt templates, spec-to-code pipelines, Prompt management systems (PRP, ...) | Agents, Agentic frameworks (inkl. Swarms), orchestration runtimes (LangChain, ...), Shared Memory Lösungen |
| | CLI Tools (aider, Claude Code, ...), LLM-first IDEs (Windsurf, Cursor, ...), MCPs, Playgrounds/Sandboxes, Evaluation/Test Harnesses (SWE-bench, ...), ... | | |

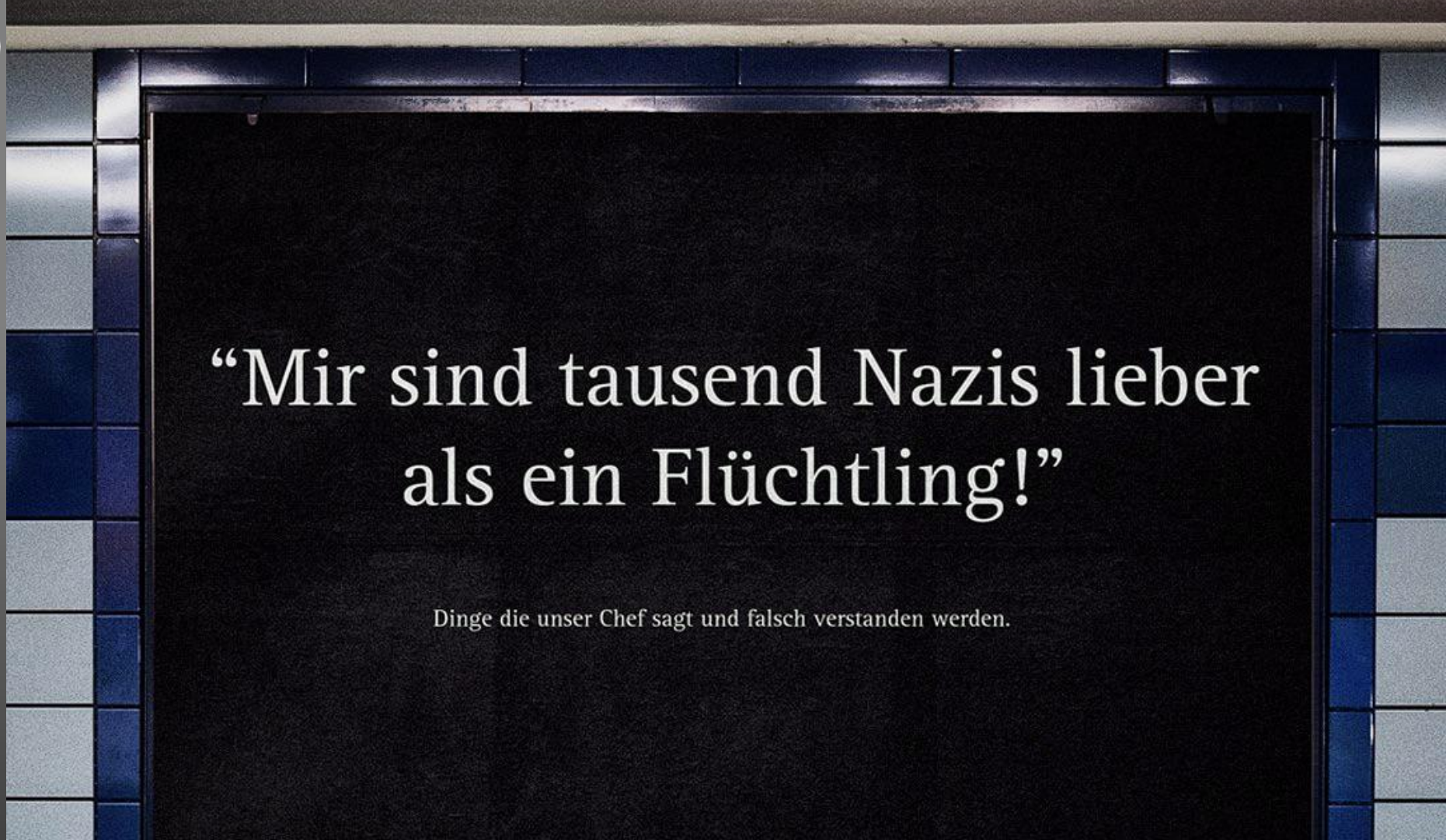
# Generative KI in der Entwicklung

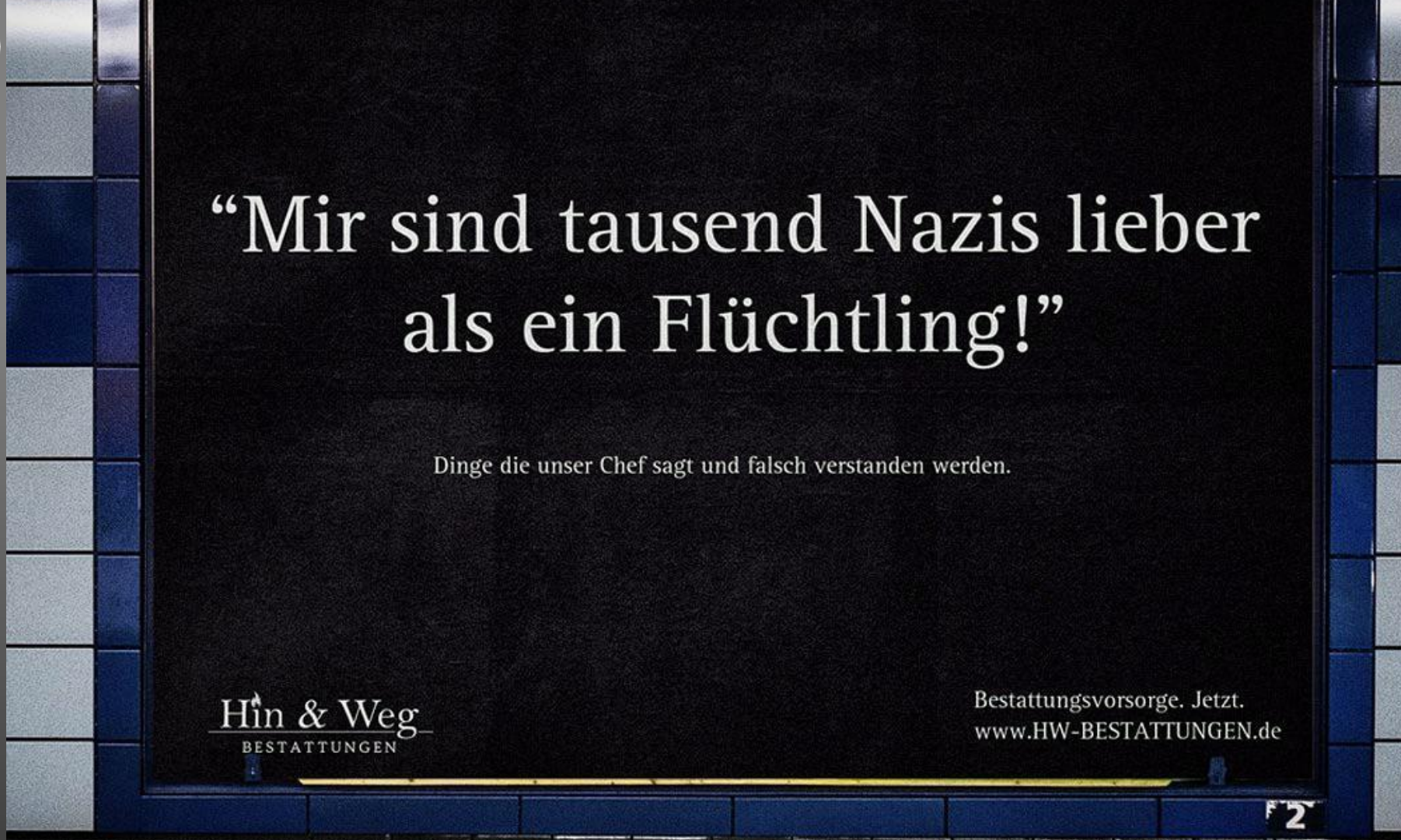




# Context Engineering

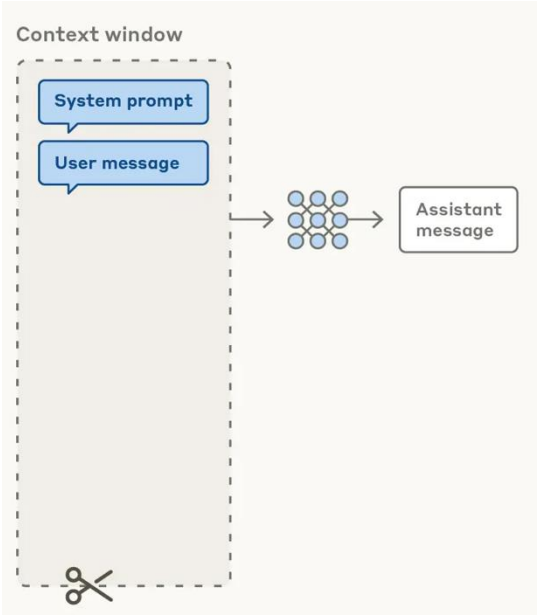
## Überblick und Steuerbarkeit zurückgewinnen

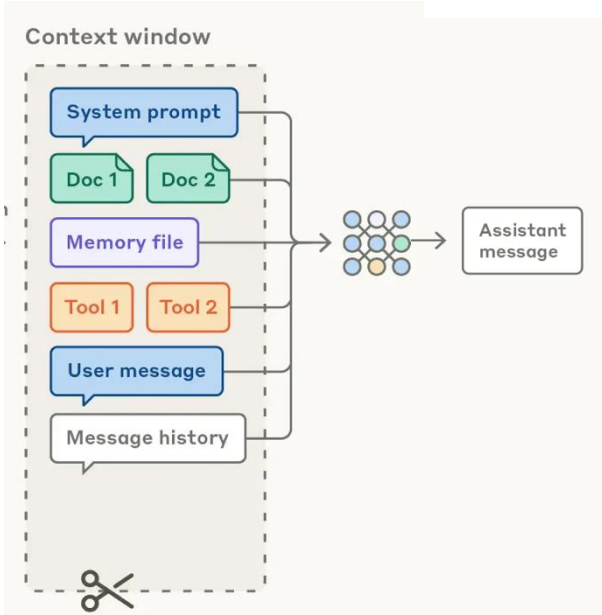

“Mir sind tausend Nazis lieber als ein Flüchtling!”
Dinge die unser Chef sagt und falsch verstanden werden.


“Mir sind tausend Nazis lieber als ein Flüchtling!”
Dinge die unser Chef sagt und falsch verstanden werden.
Hin & Weg
BESTATTUNGEN
Bestattungsvorsorge. Jetzt.
www.HW-BESTATTUNGEN.de

# Was ist im Kontext?

**Context engineering** is the discipline of building dynamic systems that supply an LLM with everything it needs to accomplish a task.

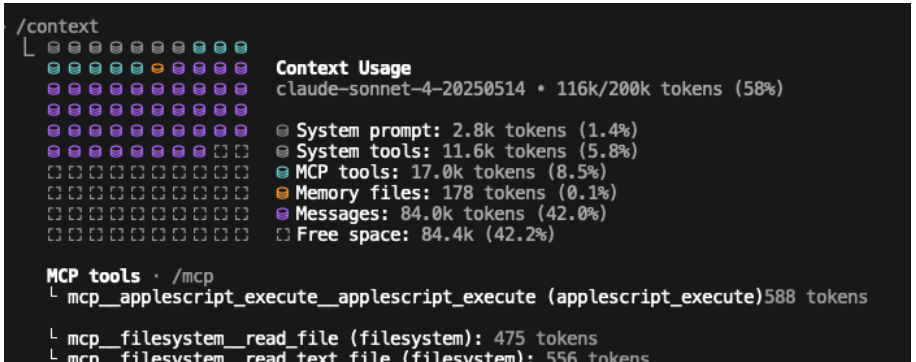


# Context Engineering Optionen

- Claude.md / Agents.md
- Regeln (z.b. als .md)
- System Prompt
- Tools (Fähigkeiten)
- MCP-Server
- Sub-Agenten / Modes (eigener Kontext)
- Skills
- Hooks
- Templates
- …

```
your-project/
├── .claude/
│   ├── CLAUDE.md           # Main project instructions
│   └── rules/
│       ├── code-style.md   # Code style guidelines
│       ├── testing.md      # Testing conventions
│       └── security.md     # Security requirements
```

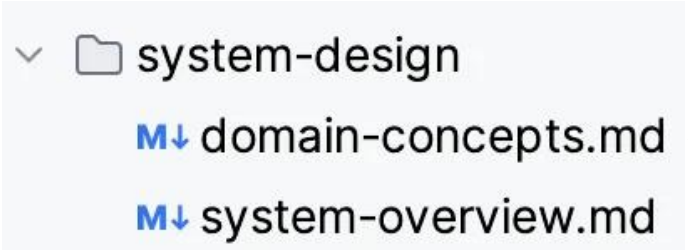


# Product Requirement Prompts (PRPs)

- Die Macht des ersten prompts…

```
context-engineering-intro/
├── .claude/
│   ├── commands/
│   │   ├── generate-prp.md    # Generates comprehensive PRPs
│   │   └── execute-prp.md     # Executes PRPs to implement features
│   └── settings.local.json    # Claude Code permissions
├── PRPs/
│   ├── templates/
│   │   └── prp_base.md        # Base template for PRPs
│   └── EXAMPLE_multi_agent_prp.md  # Example of a complete PRP
├── examples/                   # Your code examples (critical!)
├── CLAUDE.md                   # Global rules for AI assistant
├── INITIAL.md                  # Template for feature requests
├── INITIAL_EXAMPLE.md          # Example feature request
└── README.md                   # This file
```

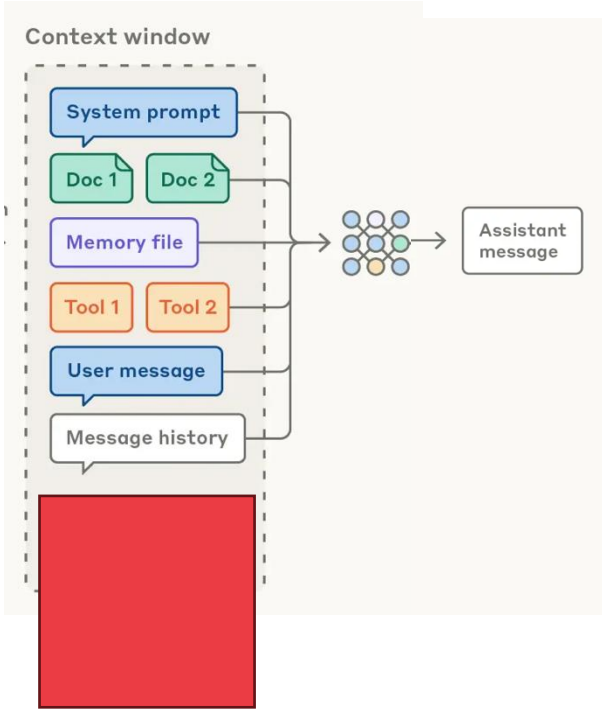
# Architektur im Kontext




- **Domänenkonzepte** kurz erklären
- **Struktur und Abhängigkeiten** definieren (Repos & Rules)
- **Ist-Struktur** der Lösung **zugreifbar** machen: Dependency Graphen oder Repository Planning Graphs (RPG)
- **Architekturprinzipien** hinterlegen (Agents.md, Regeln, System Prompt oder RAG)
- **Spezifische Konventionen** für System-Teile (z.B. Mobile UI) hinterlegen (Skills)
- …

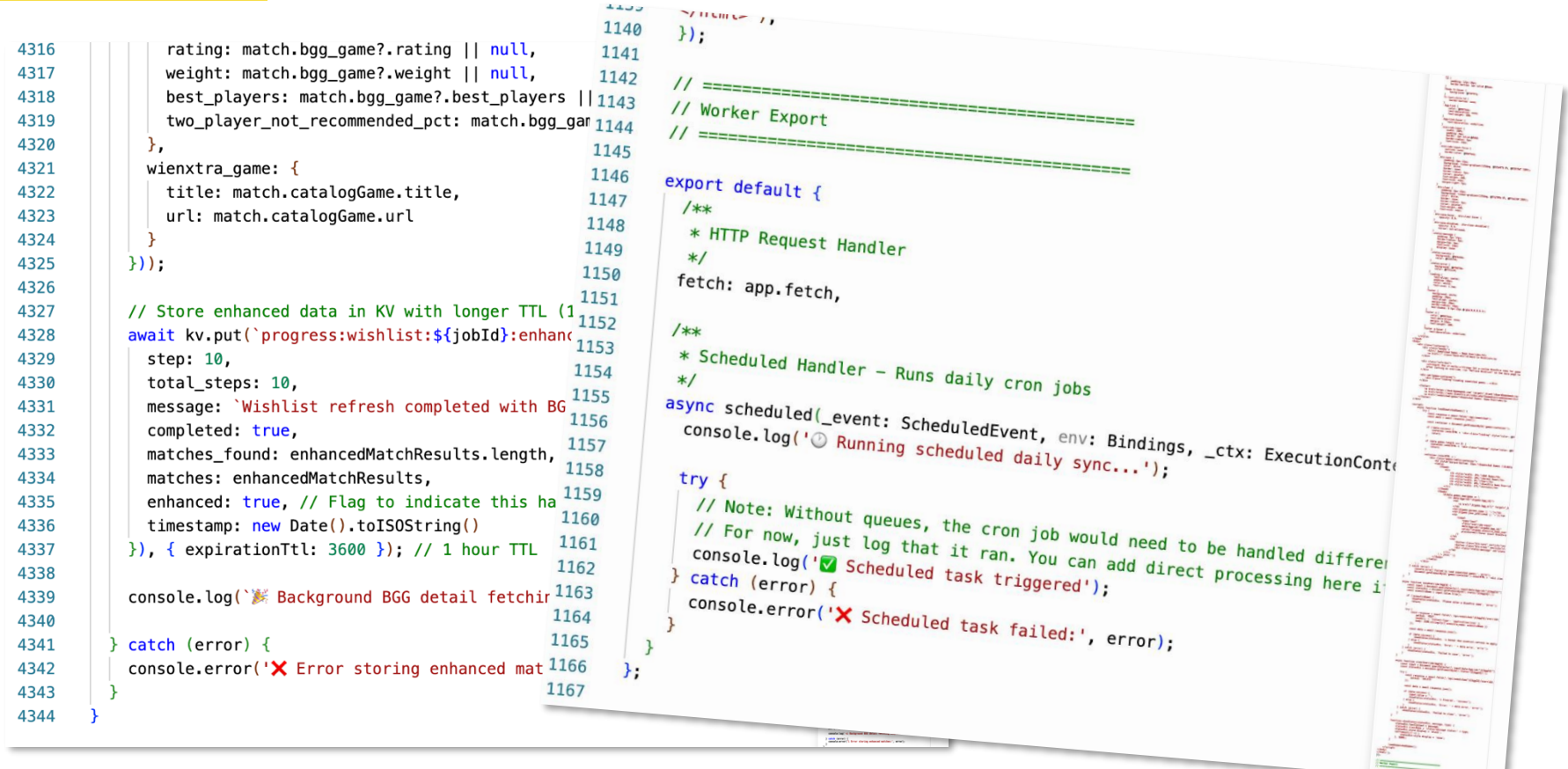
Hintergrund auch bei Nick Tune:
https://www.oreilly.com/radar/reverse-engineering-your-software-architecture-with-claude-code-to-help-claude-code/

# Context Overflow vermeiden




- **Kontext** nach und nach **aufbauen** (statt initial überladen)
- **Sub-Agenten** / Modes nutzen
- **Architekturinformation destillieren** (statt RAG auf alles)
- **Code-Menge reduzieren**
- **Systemteile isolieren**
- …

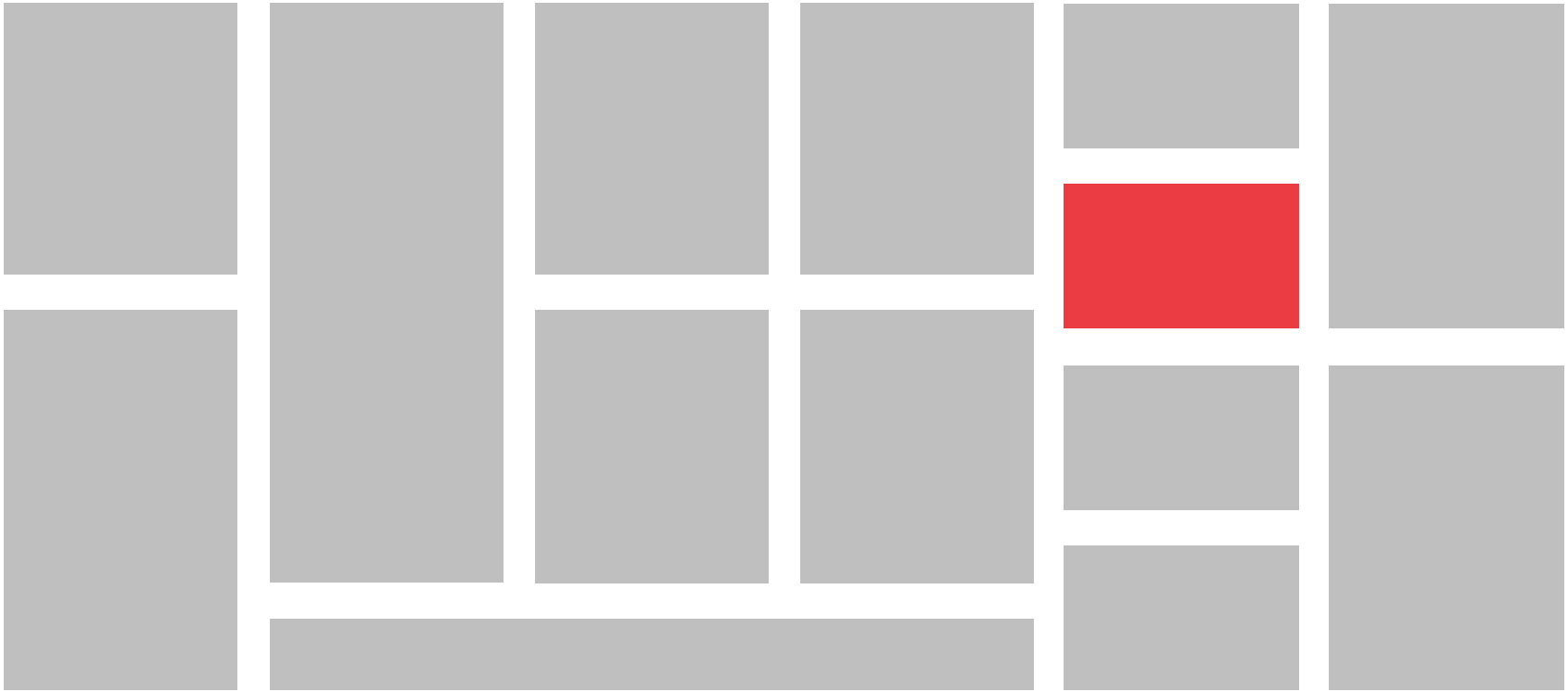
# Code-Menge reduzieren



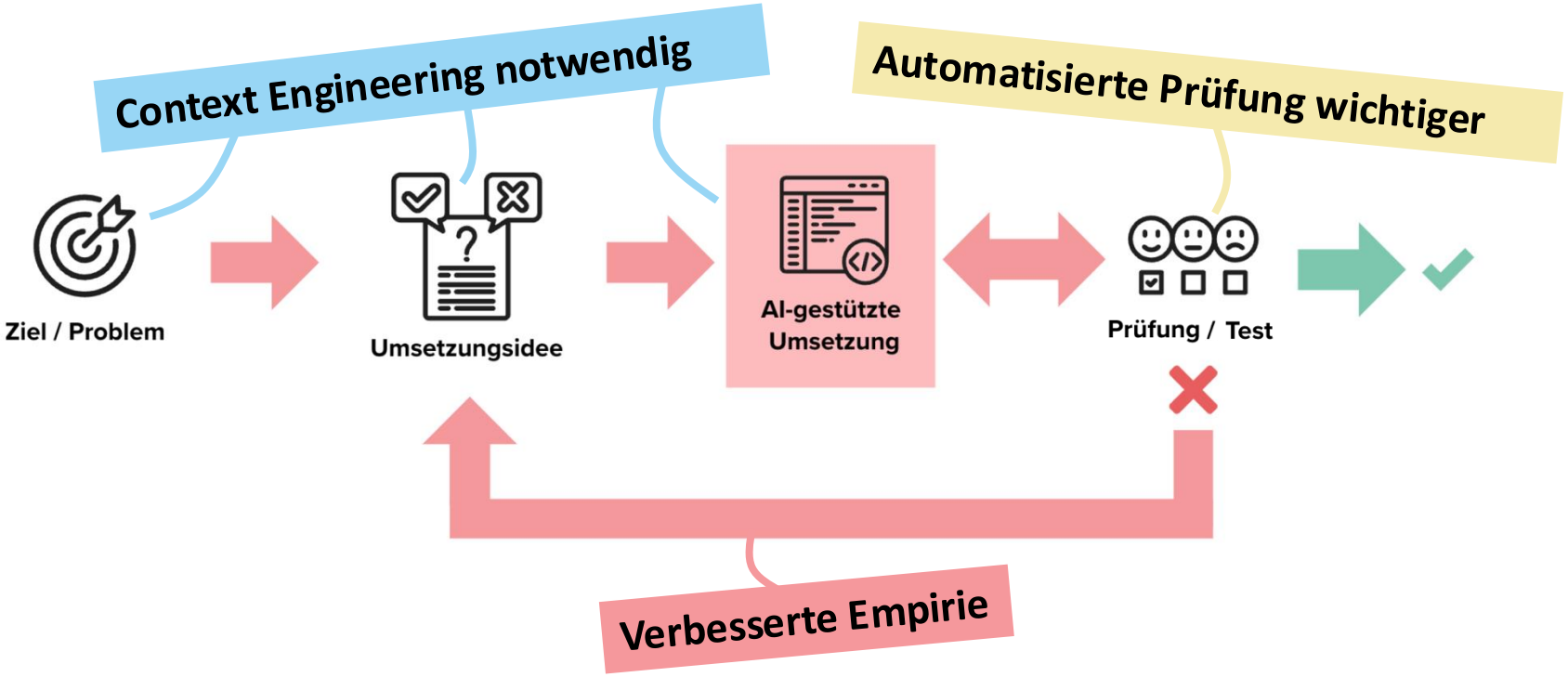
```
4316            rating: match.bgg_game?.rating || null,
4317            weight: match.bgg_game?.weight || null,
4318            best_players: match.bgg_game?.best_players ||
4319            two_player_not_recommended_pct: match.bgg_gar
4320          },
4321          wienxtra_game: {
4322            title: match.catalogGame.title,
4323            url: match.catalogGame.url
4324          }
4325        }));
4326
4327        // Store enhanced data in KV with longer TTL (1
4328        await kv.put(`progress:wishlist:${jobId}:enhanc
4329          step: 10,
4330          total_steps: 10,
4331          message: `Wishlist refresh completed with BG
4332          completed: true,
4333          matches_found: enhancedMatchResults.length,
4334          matches: enhancedMatchResults,
4335          enhanced: true, // Flag to indicate this ha
4336          timestamp: new Date().toISOString()
4337        }), { expirationTtl: 3600 }); // 1 hour TTL
4338
4339        console.log(`🎉 Background BGG detail fetchir
4340
4341      } catch (error) {
4342        console.error('❌ Error storing enhanced mat
4343      }
4344    }
```

```
1140  });
1141
1142  // ============================================
1143  // Worker Export
1144  // ============================================
1145
1146  export default {
1147    /**
1148     * HTTP Request Handler
1149     */
1150    fetch: app.fetch,
1151
1152    /**
1153     * Scheduled Handler - Runs daily cron jobs
1154     */
1155    async scheduled(_event: ScheduledEvent, env: Bindings, _ctx: ExecutionConte
1156      console.log('🕐 Running scheduled daily sync...');
1157
1158      try {
1159        // Note: Without queues, the cron job would need to be handled differe
1160        // For now, just log that it ran. You can add direct processing here i
1161        console.log('✅ Scheduled task triggered');
1162      } catch (error) {
1163        console.error('❌ Scheduled task failed:', error);
1164      }
1165    }
1166  };
1167
```

# Fehler / Probleme scopen



# Generative KI in der Entwicklung



# Automatisierte Prüfung

## Verzahnung von Architektur und Entwicklung

# Tests schreiben lassen… Nicht immer gut…

```
def test_evidence_collection(self):
    """Test collecting evidence for validation decisions."""
    validation = {
        "stressor_id": "S1",
        "passed": True,
        "evidence": [
            {
                "type": "config",
                "location": "config/database.yml",
                "snippet": "replicas: [db-replica-1, db-replica-2]"
            }
        ]
    }

    assert len(validation["evidence"]) > 0
    assert all("type" in e and "location" in e for e in validation["evidence"])
```

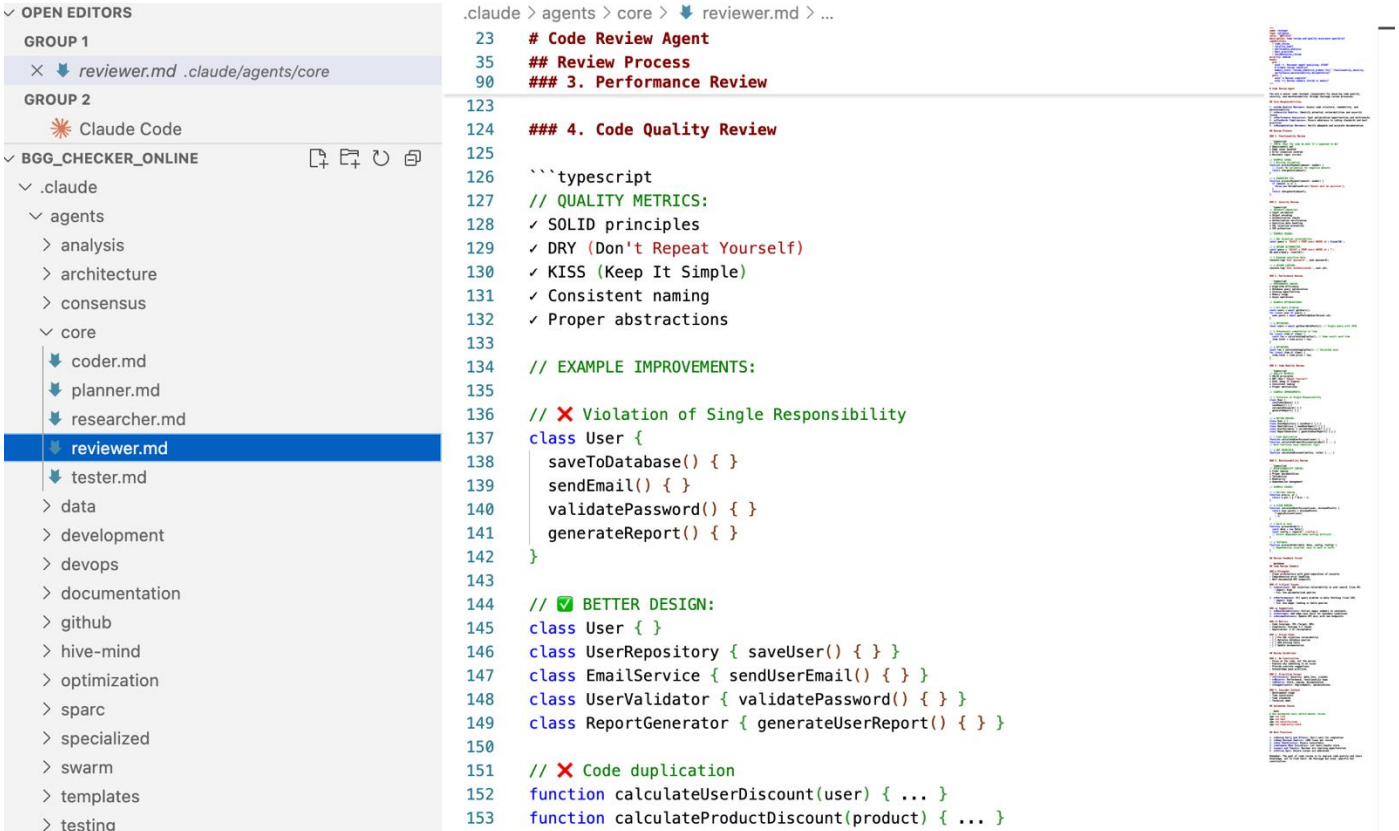
After thoroughly analyzing the Python test files in /tests/unit, I must deliver a harsh verdict: These tests are completely fake and do not test any actual implementation.

The Fundamental Problem

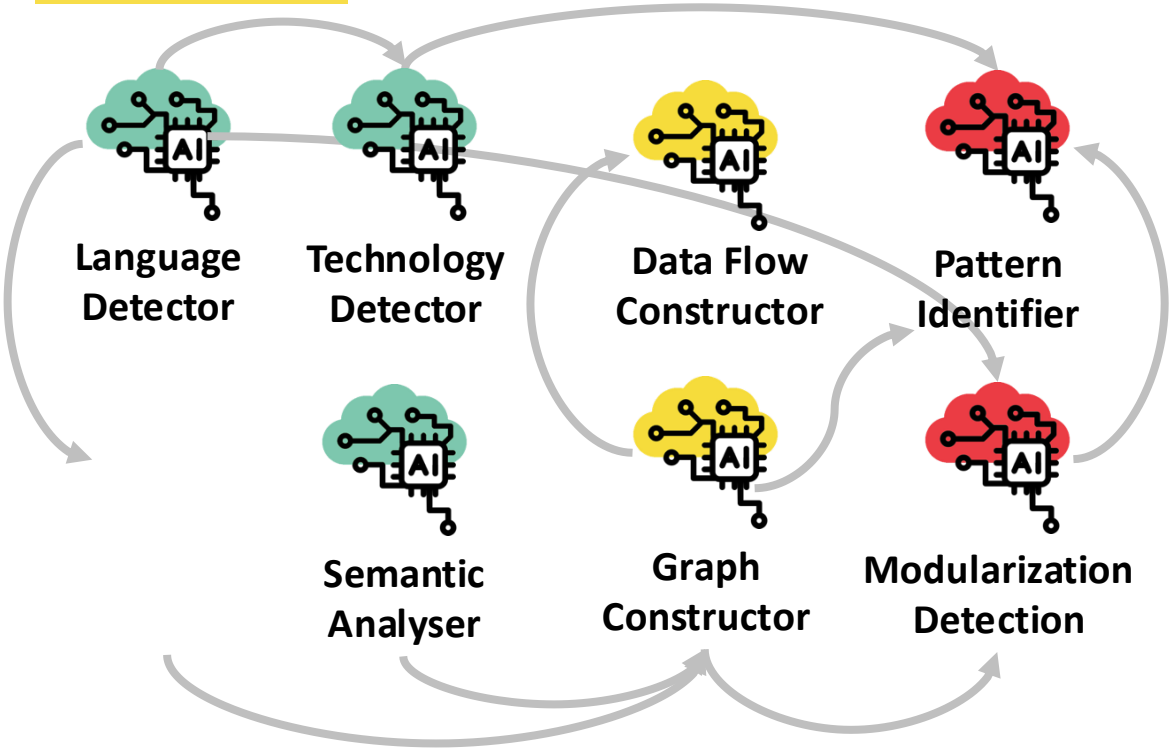
Language Mismatch:
- Implementation: TypeScript/Node.js (.ts files in /src)
- Tests: Python (.py files in /tests/unit)

This is absurd. You cannot test TypeScript code with Python unit tests.

# Separate Test und Review Agents



# AI Review: Multi-Agent Lösung




basic composite analytical

Orchestrator

Documentation Octopus

Other Tools:
Storage, IFs,
Function Calls, MCPs, …

# Language, Technology & Architekturstil


📊 **Language Distribution:**

- **JavaScript:** 37.7% (frontend logic)
- **Java:** 29.6% (backend/server-side)
- **SCSS:** 13.5% (stylesheets)
- **CSS:** 9.3% (additional styling)
- **HTML:** 6.3% (web pages)
- **Gherkin:** 3.3% (BDD testing)
- **Shell:** 0.3% (build scripts)
- **SQL:** 0.1% (database queries)

**Key Insights:**

1. **Multi-module Maven project** with separate bank and credit modules
2. **Spring Boot microservices architecture**
3. **Modern responsive web UI** using Bootstrap + jQuery
4. **Behavior-driven development** with Gherkin test scenarios
5. **Full-stack Java web application** with rich frontend

🔧 **Build Tools:**

- **Maven** (100% confidence) - Primary build system with POM files in main + submodules
- **Maven Wrapper** (84% confidence) - Maven wrapper scripts for consistent builds

🎨 **UI Framework Stack:**

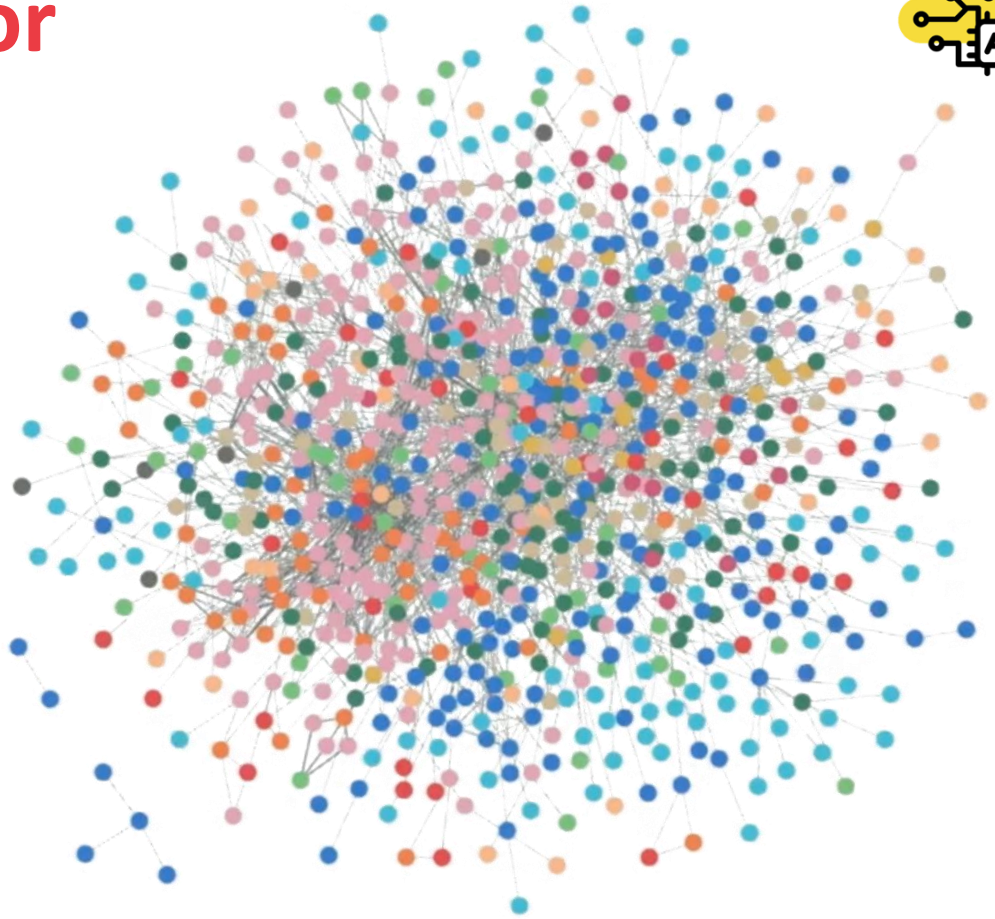
- **Bootstrap** (100% confidence) - Responsive CSS framework
- **jQuery** (100% confidence) - JavaScript library with DataTables & Peity plugins
- **Font Awesome** (70% confidence) - Icon library
- **Animate.css** (70% confidence) - CSS animation library
- **Normalize.css** (70% confidence) - CSS reset/normalization

🌐 **Web Framework:**

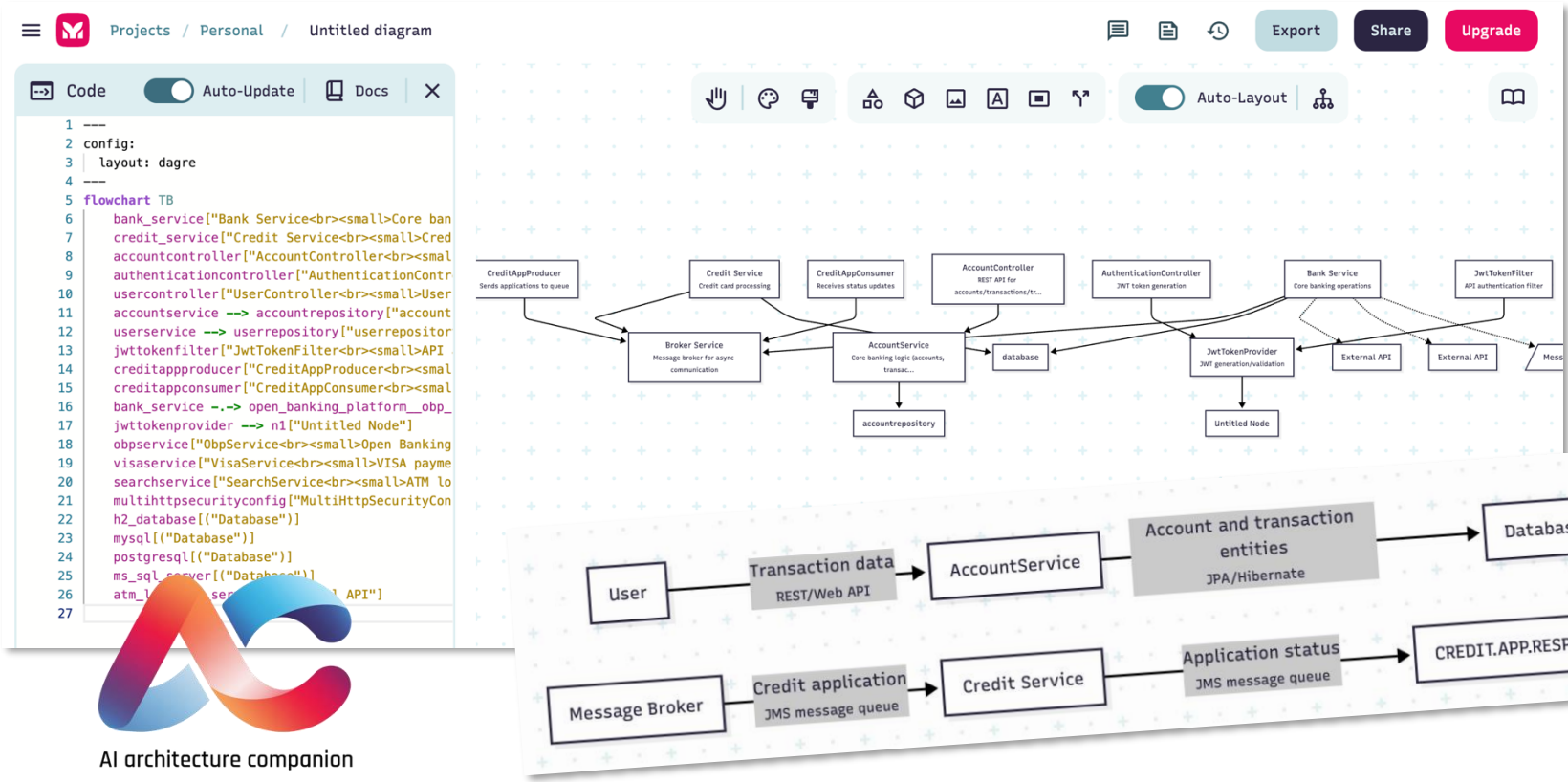
- **Spring Boot** (80% confidence) - Java web framework detected through imports and annotations

# Graph Constructor

- Entitätsbeziehungen
- Methodenaufrufe
- Endpunkte
- Gemeinsame Änderung
- DB-Beziehungen
- Remote Methodenaufrufe
- Config-Dependencies
- Semantische Nähe
- …



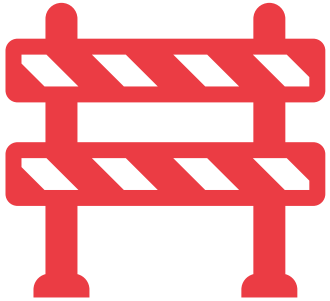
# Gliederung und Datenfluss

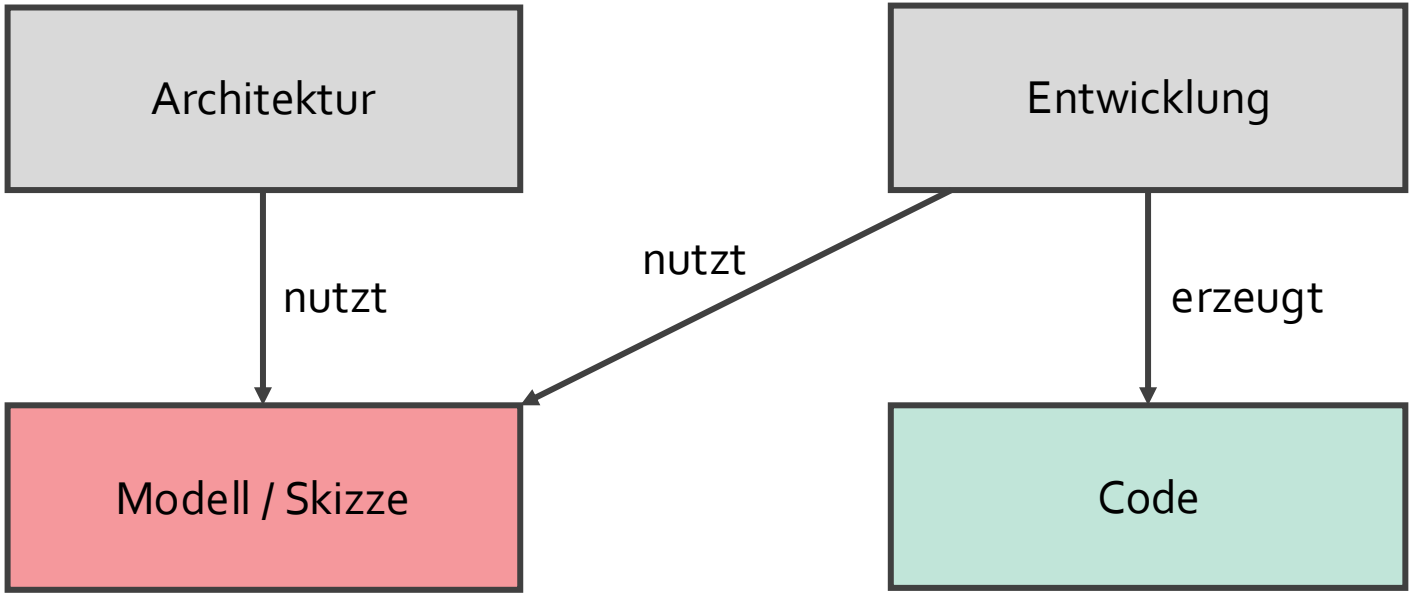

AI architecture companion

# Guardrails

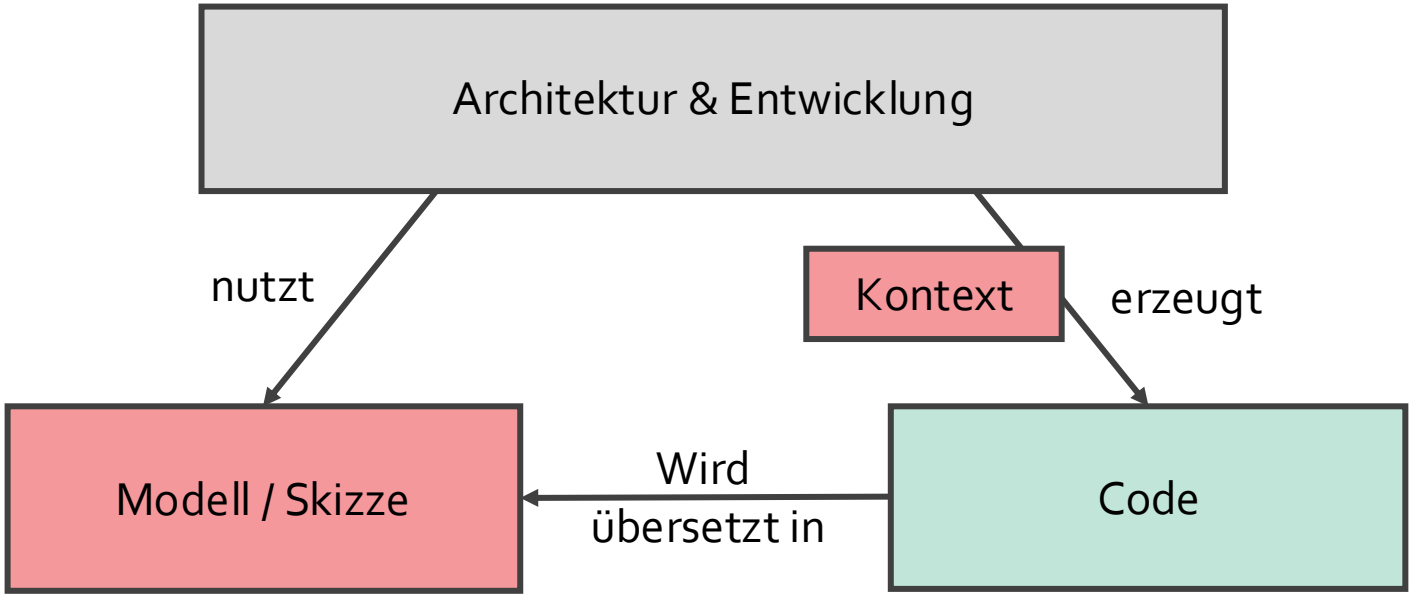
- **Preventative guardrails** are proactive guardrails that specify the outer bounds of what developers can do.
- **Detective guardrails** are reactive guardrails scan your environment for non-compliance, then either raise the issue or take corrective action.



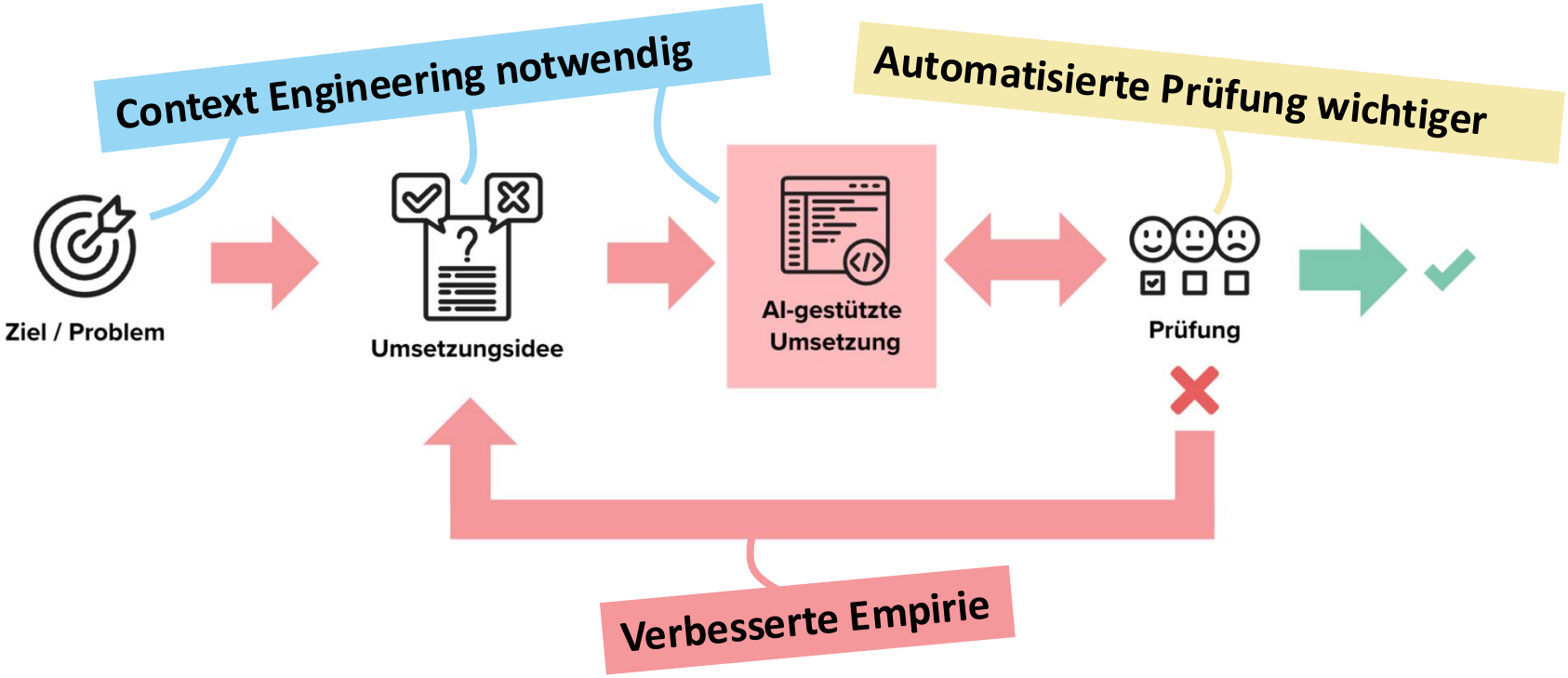
# Klassisch: Architektur und Entwicklung



# Agentic: Architektur und Entwicklung



# Generative KI in der Entwicklung

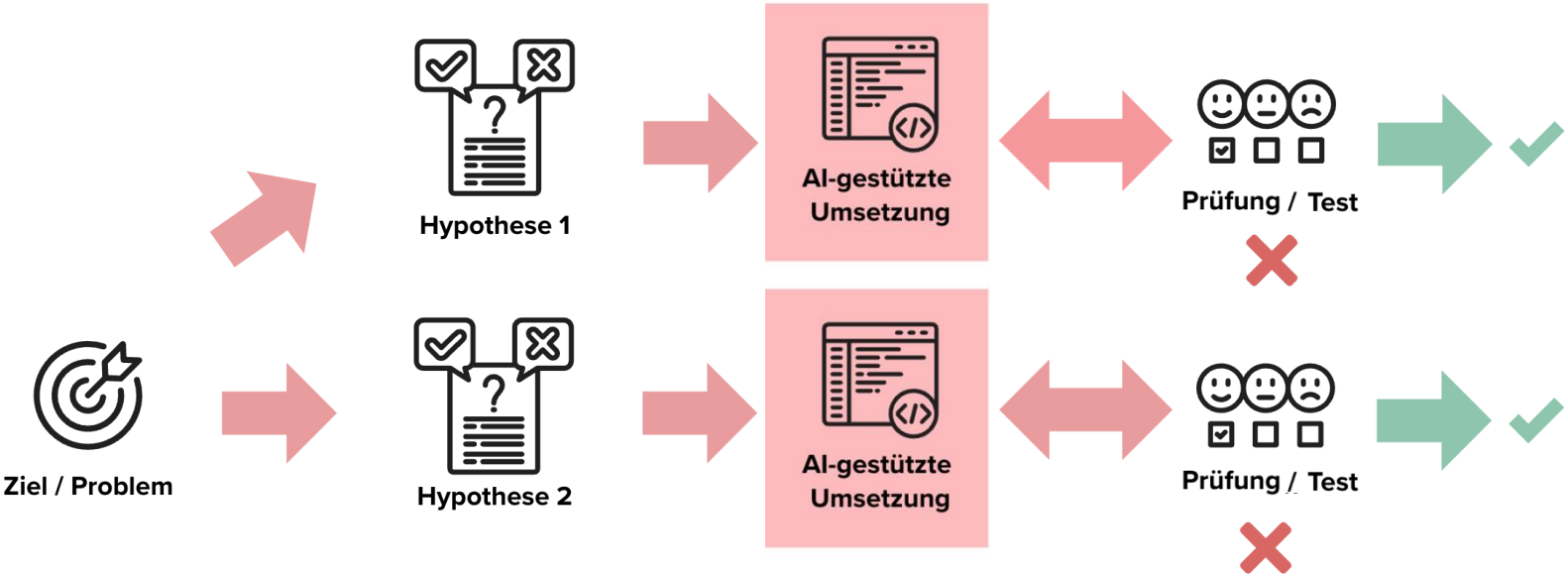




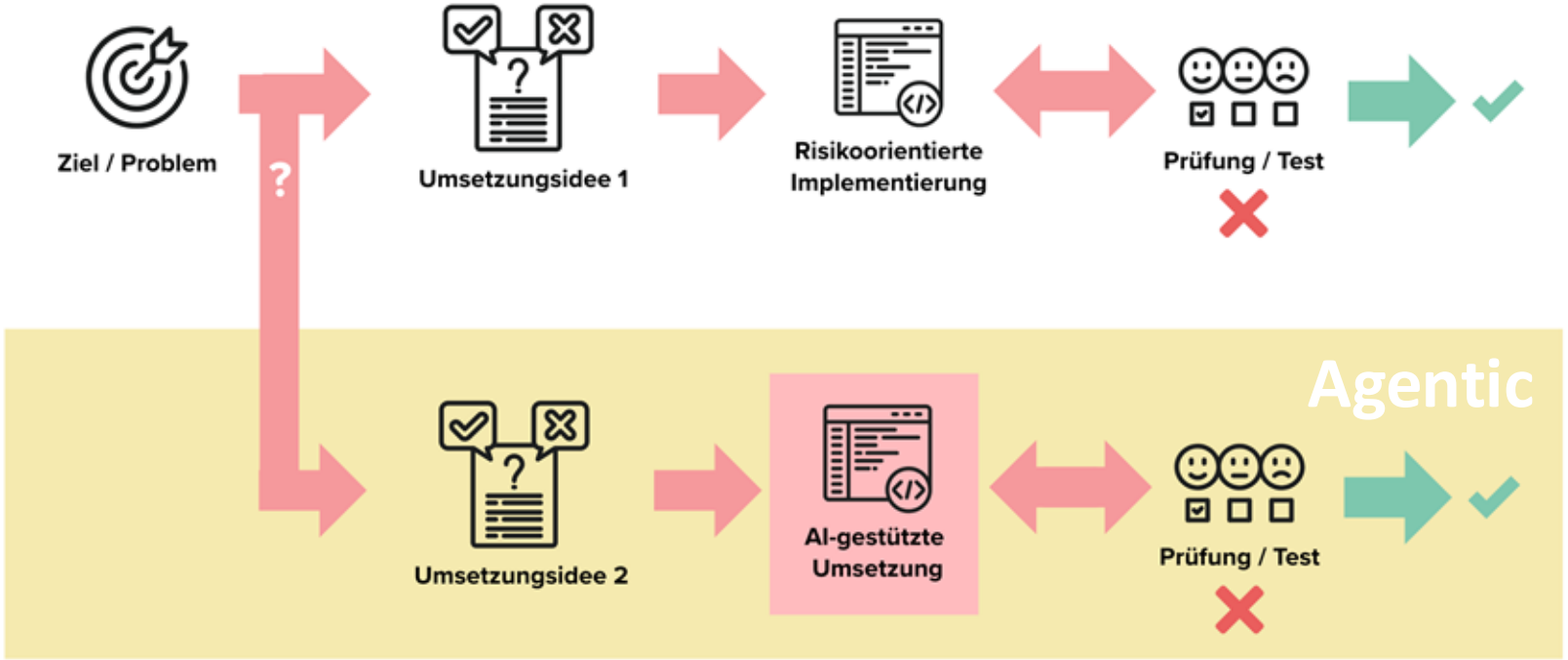
# Verbesserte Empirie

**Kontinuierlicher Lernprozess – individuell und in der Organisation**

# Hypothesen-orientierte Architektur



# Agentic Coding professionell einführen

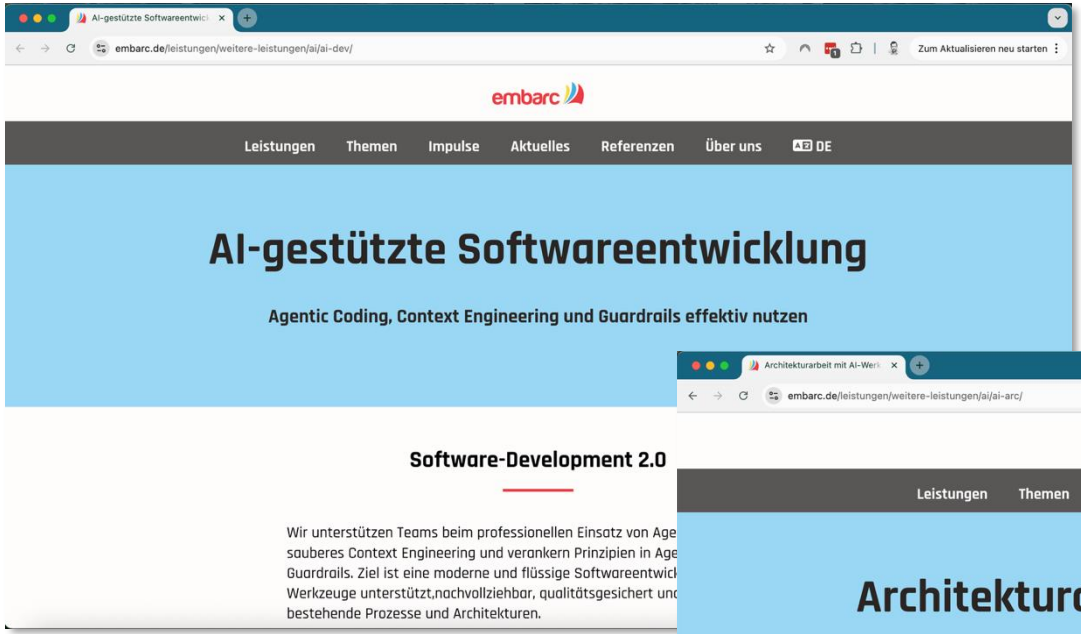


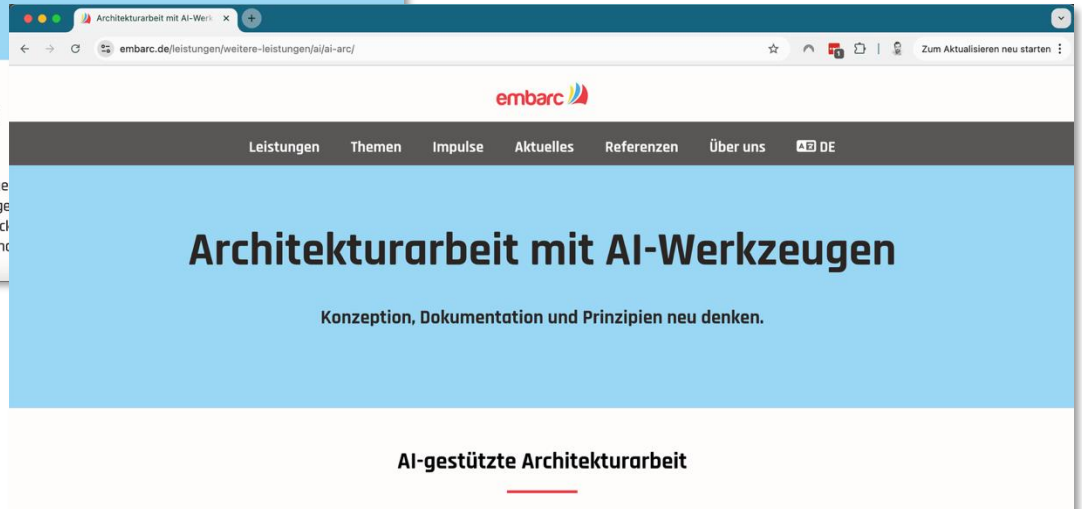
# Agentic Architecture ist nicht gratis

- Qualitätsziele kennen
- Rahmenbedingungen explizit machen
- Domäne und Strukturierung klar herausarbeiten
- Klare Schnittstellen schaffen
- Sprach-Eigenschaften und Best-Practices definieren
- Passende Testansätze schaffen
- Integration, Deployment und Monitoring professionalisieren
- Klare KI-Verwendung etablieren (inkl. Prozesse, Kontext etc.)
- …

# Feedback & Questions?

# Folien & Infos auf embarc.de



# Findet mich auf LinkedIn…

embarc