CODE-REVOLUTION: WIE GITHUB COPILOT DAS PROGRAMMIEREN NEU ERFINDET (SIMON KEßLER)

Innovative Tools verändern die Programmierlandschaft nachhaltig

AGENDA DER PRÄSENTATION

- Einführung in GitHub Copilot
- Die Modi von GitHub Copilot
- Aktuelle Features von GitHub Copilot
- Prompting Methoden
- Zukunftsaussichten und Grenzen von GitHub Copilot

WARUM KI?

- Die Telekom hat durch den Einsatz von KI 4000 Vollzeitäquivalente eingespart.
- „Für die aktuellen Aufgaben werden wir weniger Mitarbeiter benötigen, aber dafür mehr für andere Tätigkeiten“, erklärte Amazon-CEO Andy Jassy.
- Schon 30 Prozent der Angestellten verwenden ChatGPT und ähnliche Tools.
- McKinsey schätzt, dass bis 2045 generative KI 50% der vorhandenen Arbeitsplätze automatisieren wird.

EINFÜHRUNG IN GITHUB COPILOT

WAS IST GITHUB COPILOT?

Echtzeit-Code-Vorschläge

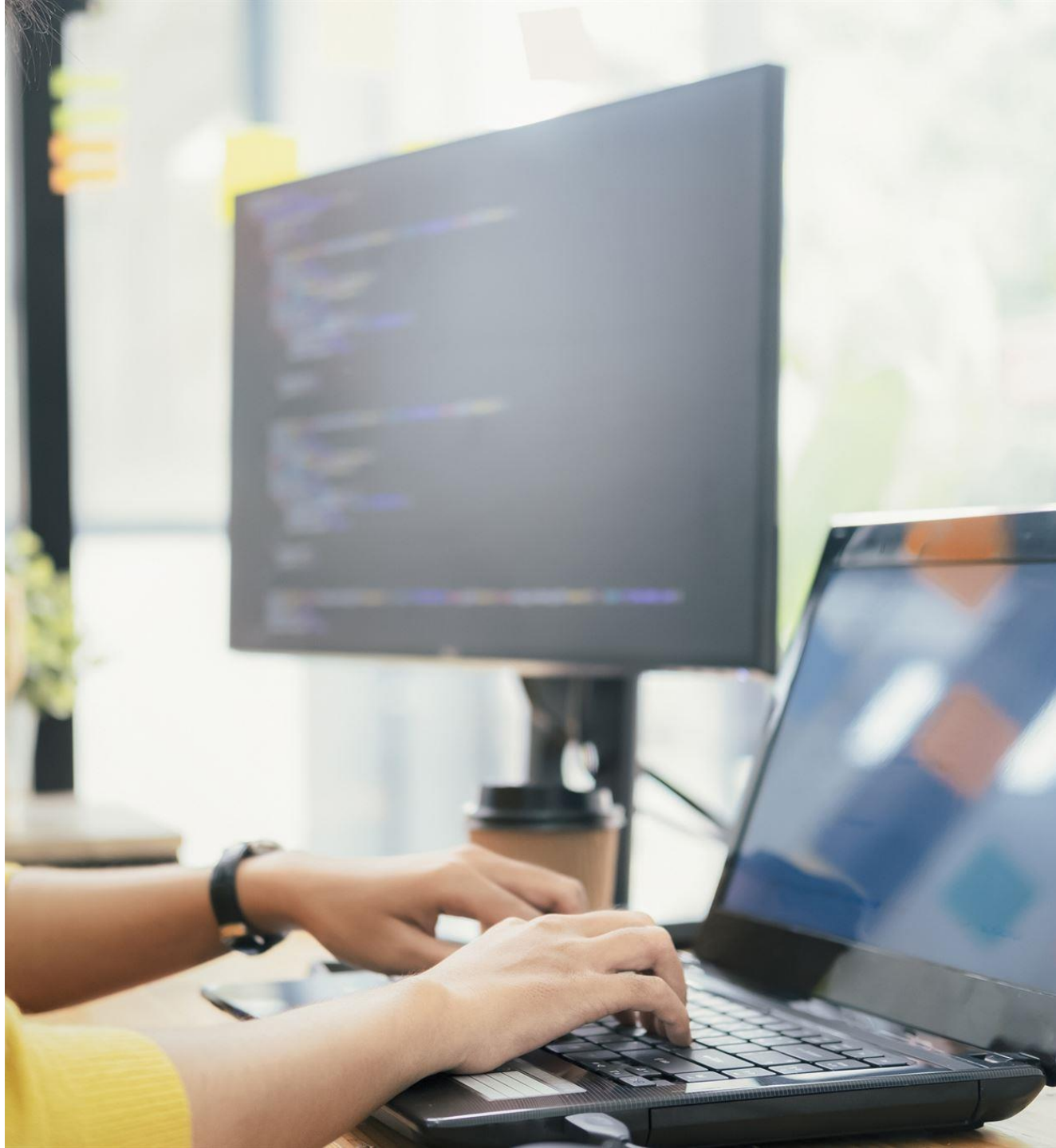
GitHub Copilot bietet Entwicklern sofortige Vorschläge für Code in Echtzeit, was die Effizienz erhöht.

Maschinelles Lernen

Das Tool verwendet maschinelles Lernen, um den Kontext des Codes zu verstehen und relevante Snippets zu generieren.

Entwicklung beschleunigen

Die Verwendung von GitHub Copilot beschleunigt den Entwicklungsprozess erheblich, indem es den Programmieraufwand reduziert.



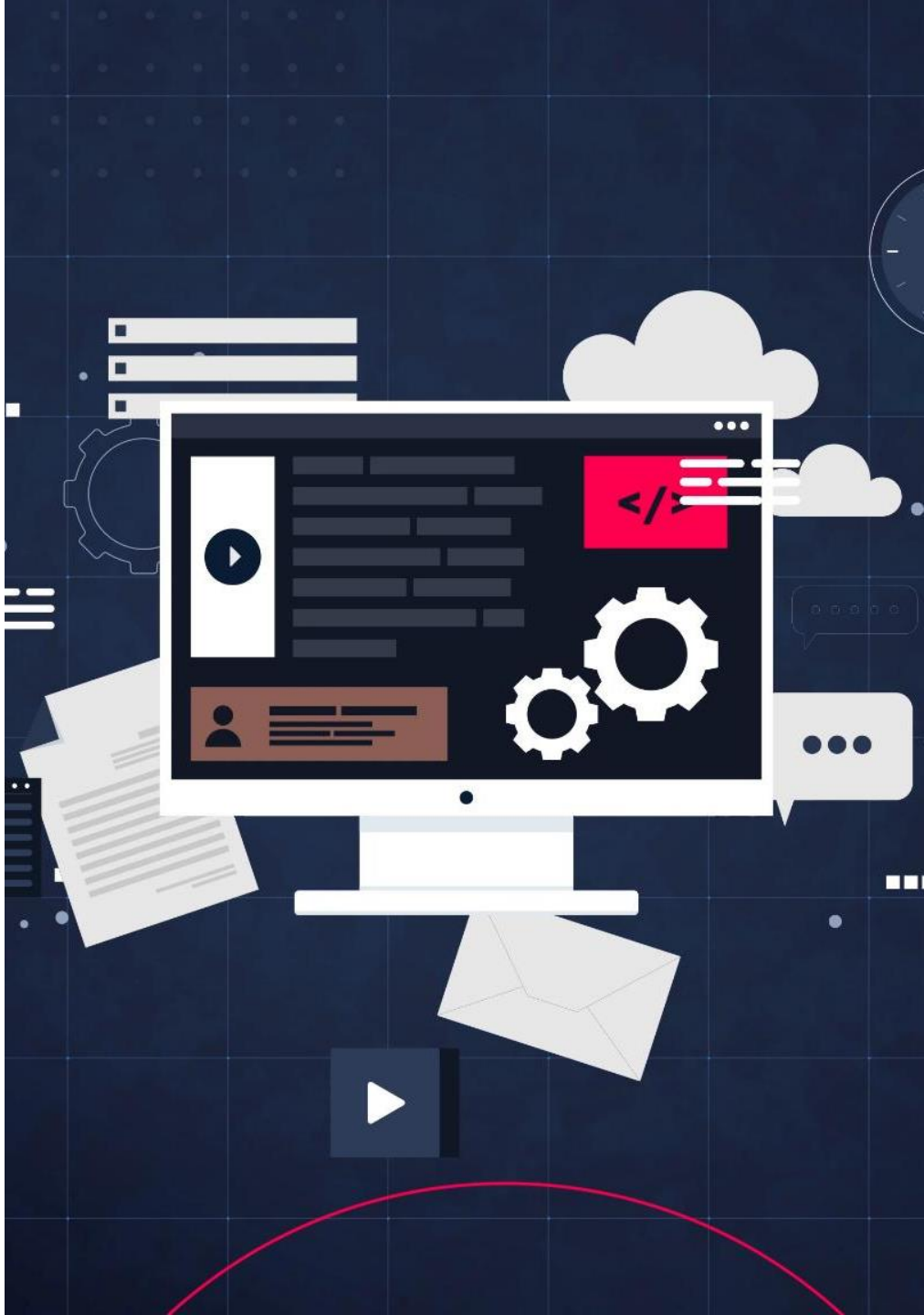
ZIELE UND VISIONEN HINTER COPILOT

Produktivitätssteigerung

GitHub Copilot zielt darauf ab, die Produktivität von Entwicklern zu steigern, indem es Routineaufgaben automatisiert und vereinfacht.

Fokus auf Kreativität

Mit der Unterstützung von Copilot können Programmierer sich auf die kreativeren Aspekte der Softwareentwicklung konzentrieren und innovativere Lösungen finden.



INTEGRATION IN DEN PROGRAMMIERALLTAG

Nahtlose Integration

GitHub Copilot integriert sich mühelos in verschiedene Entwicklungsumgebungen und verbessert die Benutzerfreundlichkeit für Entwickler.

Wertvolles Entwicklerwerkzeug

GitHub Copilot ist ein unverzichtbares Werkzeug, das Entwicklern hilft, effizienter zu arbeiten und ihre Produktivität zu steigern.

Optimierung des Workflows

Mit GitHub Copilot wird der Workflow optimiert, indem alltägliche Programmieraufgaben automatisiert und Vorschläge bereitgestellt werden.

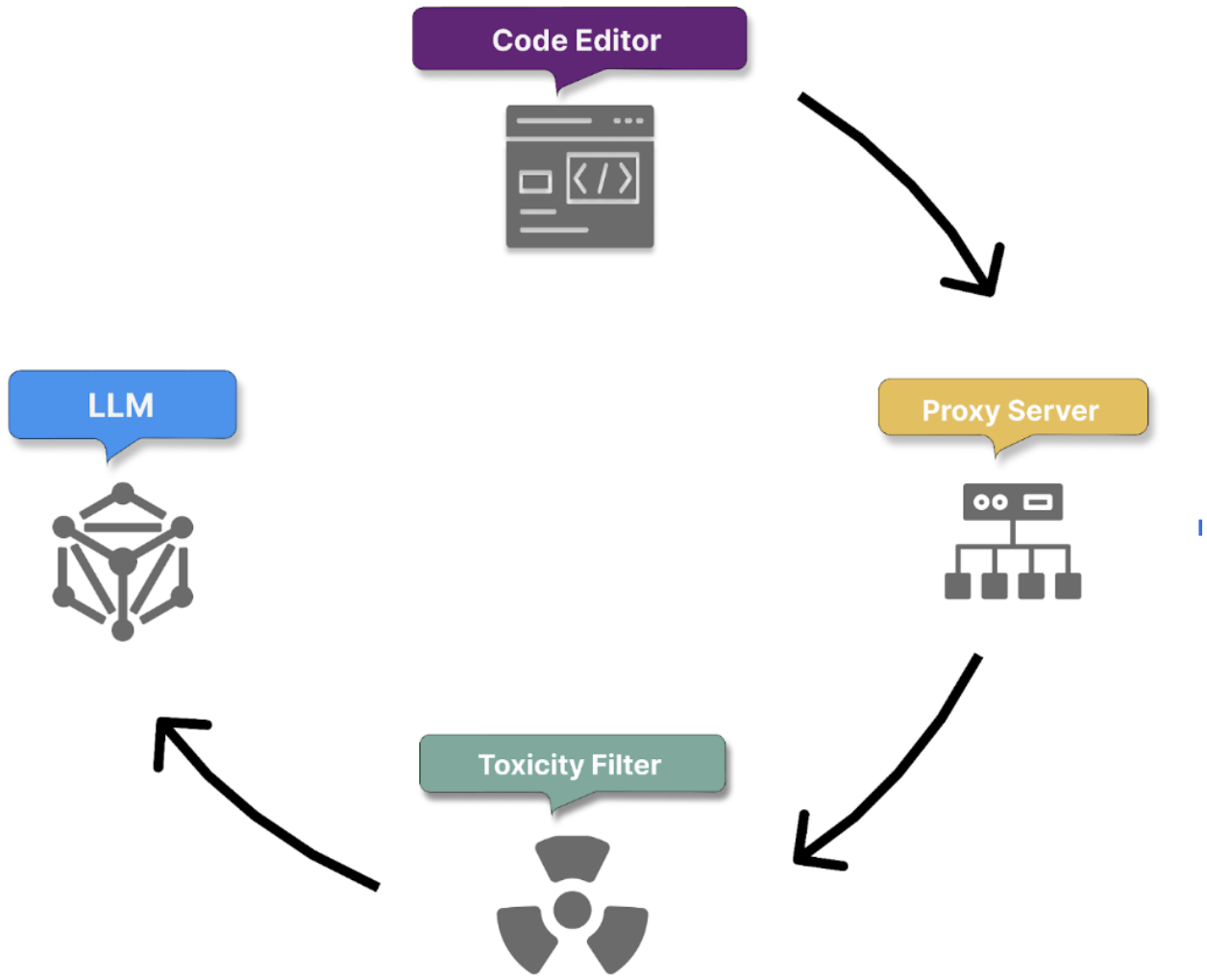
INBOUND FLOW

Sichere Übertragung und Kontextermittlung

Proxy Filter

Toxicity filtering

Code Generierung mit LLM



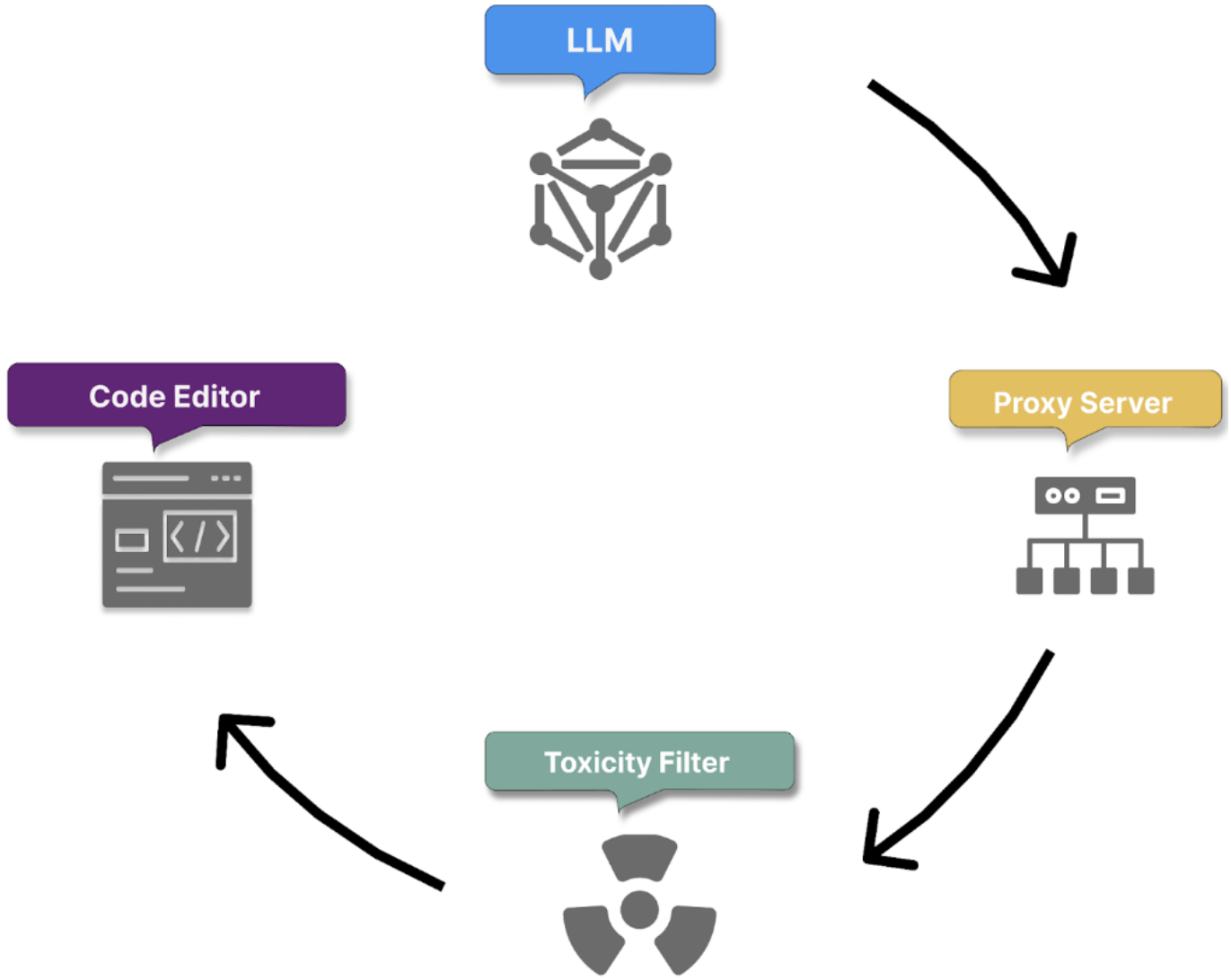
OUTBOUND FLOW

Post-processing und response validation

Code quality (common bugs, SQl injections,...)

Matching public code

Toxicity filtering



DIE MODI VON GITHUB COPILOT

DIE MODI VON GITHUB COPILOT

Ask Mode

Der Ask-Modus funktioniert wie ein schneller Berater im Editor, bei dem du Code markieren und Fragen stellen kannst, ohne dass Änderungen vorgenommen werden – er liefert Erklärungen, Vorschläge und Programmierhilfe direkt im Kontext deines Projekts.

Edit Mode

Im Edit-Modus kannst du natürlichsprachliche Anweisungen geben, um Code in ausgewählten Dateien zu ändern, wobei Copilot die Änderungen vorschlägt und du die volle Kontrolle über die Überprüfung und Annahme der Vorschläge behältst.

Agent Mode

Der Agent-Modus übernimmt komplexere Aufgaben selbstständig und kann mehrere zusammenhängende Aktionen ausführen – wie das Erstellen ganzer Funktionen, das Refaktorieren von Code oder das Lösen von GitHub Issues – wobei er wie ein eigenständiger Entwickler arbeitet, der mehrere Dateien analysiert und zielgerichtet verändert.

AKTUELLE FEATURES VON GITHUB COPILOT

CODE-VERVOLLSTÄNDIGUNG UND -GENERIERUNG

Hauptfunktion von Copilot

GitHub Copilot bietet eine intelligente Code-Vervollständigung, die den Programmierprozess effizienter macht.

Generierung von Code

Copilot generiert neuen Code basierend auf den vorherigen Eingaben des Entwicklers und unterstützt die Kreativität.

Sinnvolle Ergänzungen

Die Vorschläge von Copilot sind kontextabhängig, was die Relevanz der Ergänzungen erhöht und Zeit spart.

KONTEXTABHÄNGIGE VORSCHLÄGE UND OPTIMIERUNGEN

Kontextanalyse

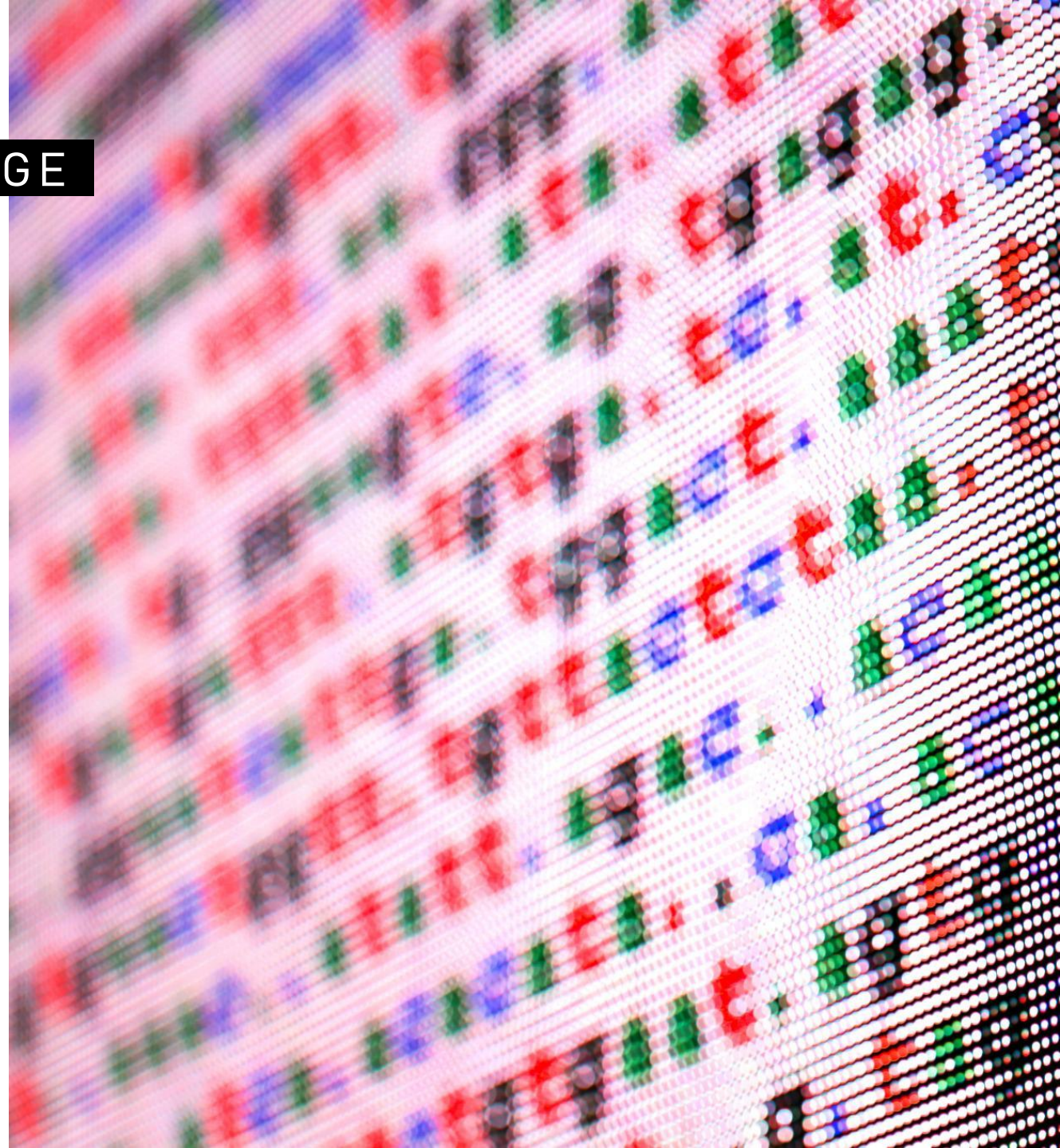
GitHub Copilot analysiert den aktuellen Code-Kontext, um relevante und maßgeschneiderte Vorschläge zu generieren.

Maßgeschneiderte Vorschläge

Die maßgeschneiderten Vorschläge von Copilot helfen Entwicklern, effizienter an spezifischen Programmieraufgaben zu arbeiten.

Code-Optimierung

Durch die Vorschläge von Copilot können Entwickler ihren Code optimieren und die besten Ansätze für ihre Projekte wählen.



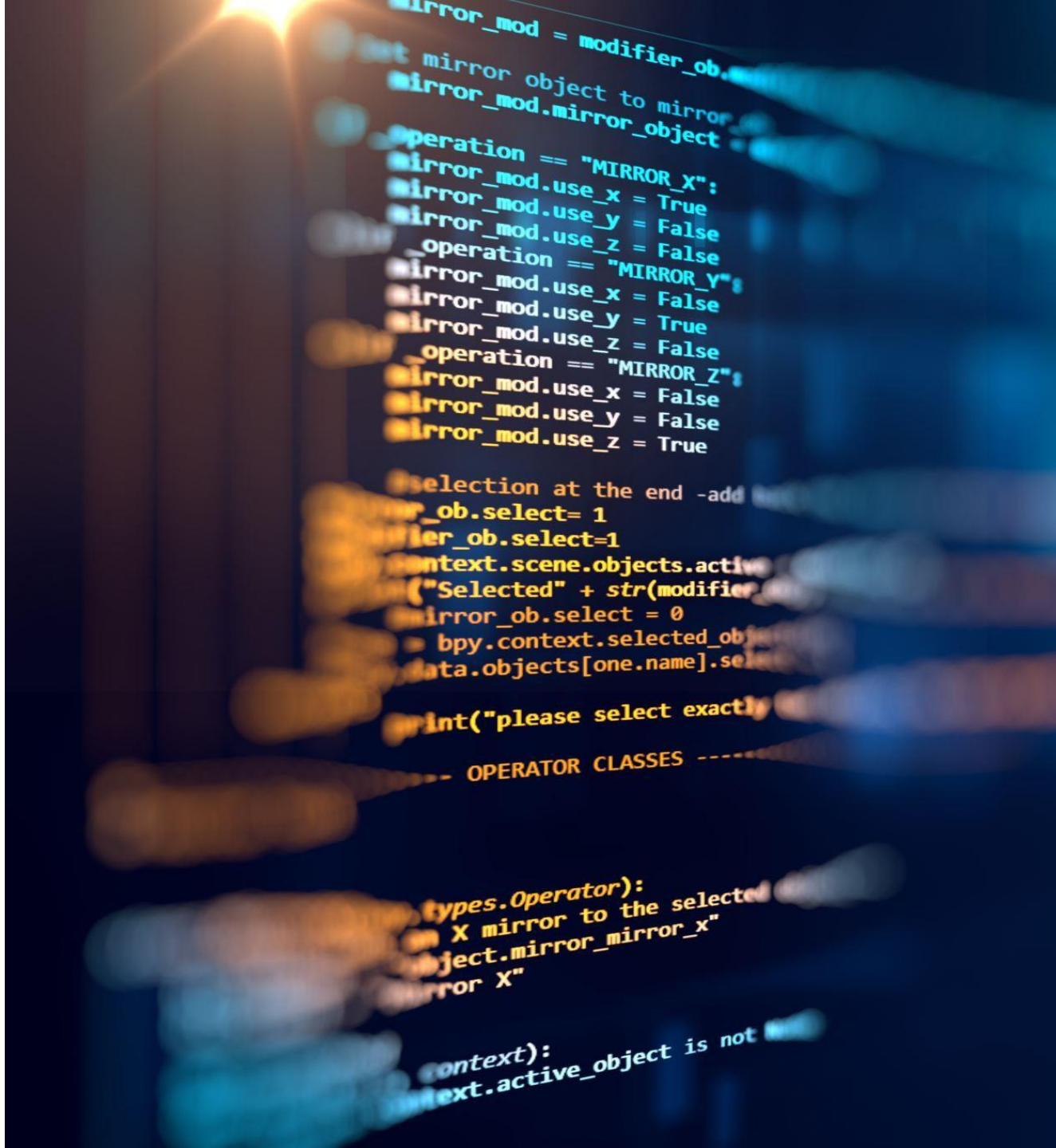
UNTERSTÜTZTE PROGRAMMIERSPRACHEN UND FRAMEWORKS

Vielfalt der Programmiersprachen

GitHub Copilot unterstützt eine breite Palette von Programmiersprachen, einschließlich Python, JavaScript und Ruby, was es vielseitig macht.

Vielseitige Tools für Entwickler

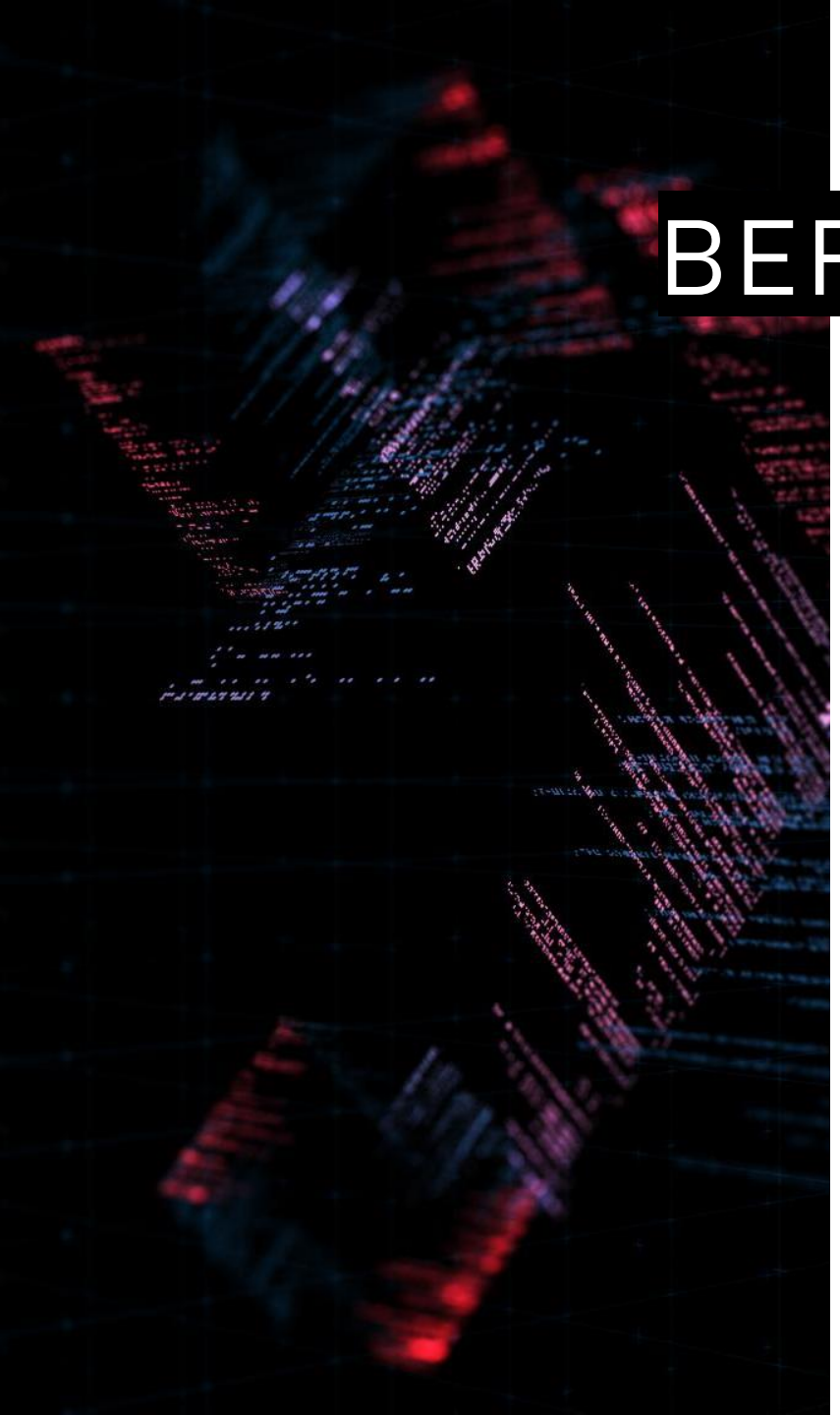
Dank der Unterstützung vieler Frameworks ist GitHub Copilot ein unverzichtbares Werkzeug für Entwickler, die verschiedene Technologien nutzen.



ÜBERSICHT ÜBER KI-MODELLE

GPT-4.1
GPT-4o
GPT-4.5
Claude Sonnet 3.5
Claude Sonnet 3.7
Claude Sonnet 3.7 Thinking
Claude Sonnet 4
Claude Opus 4
Gemini 2.0 Flash
Gemini 2.5 Pro
o1
o3
o3-mini
o4-mini

https://docs.github.com/en/copilot/using-github-copilot/ai-models/choosing-the-right-ai-model-for-your-task



BEFEHLE UND VARIABLEN

Command	Beschreibung
/doc	Fügt einen Dokumentationskommentar zu diesem Symbol hinzu
/explain	Erläutert, wie der Code in deinem aktiven Editor funktioniert
/fix	Lösung für Probleme im ausgewählten Code vorschlagen.
/help	Kurzübersicht und Verwendungsgrundlagen für GitHub Copilot
/optimize	Laufzeit des ausgewählten Codes analysieren und verbessern.
/tests	Erstellt Komponententests für den ausgewählten Code

INSTRUCTIONFILES IN GITHUB COPILOT

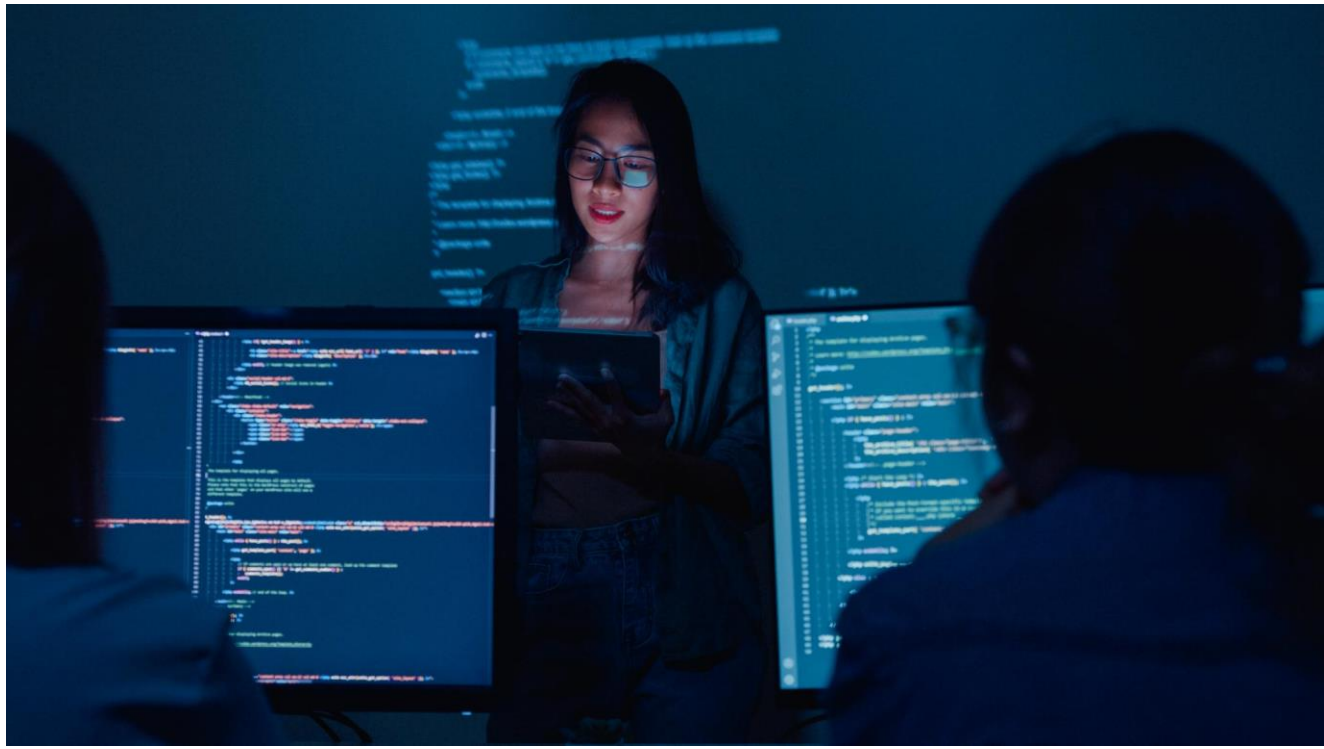
Dient dazu die bestimmte Informationen oder Befehle an die Chat-Anfrangen anzuhängen.

.github/copilot-instructions.md

We use Bazel for managing our Java dependencies, not Maven, so when talking about Java packages, always give me instructions and code samples that use Bazel.

We always write JavaScript with double quotes and tabs for indentation, so when your responses include JavaScript code, please follow those conventions.

Our team uses Jira for tracking items of work.



COPILOT-CHATS MIT MCP

Verbesserte Interaktion

Copilot-Chats mit MCP verbessern die Interaktion zwischen Mensch und KI durch besseres Kontextverständnis.

Genauere Antworten

MCP ermöglicht dem Modell, relevantere und präzisere Antworten zu liefern.

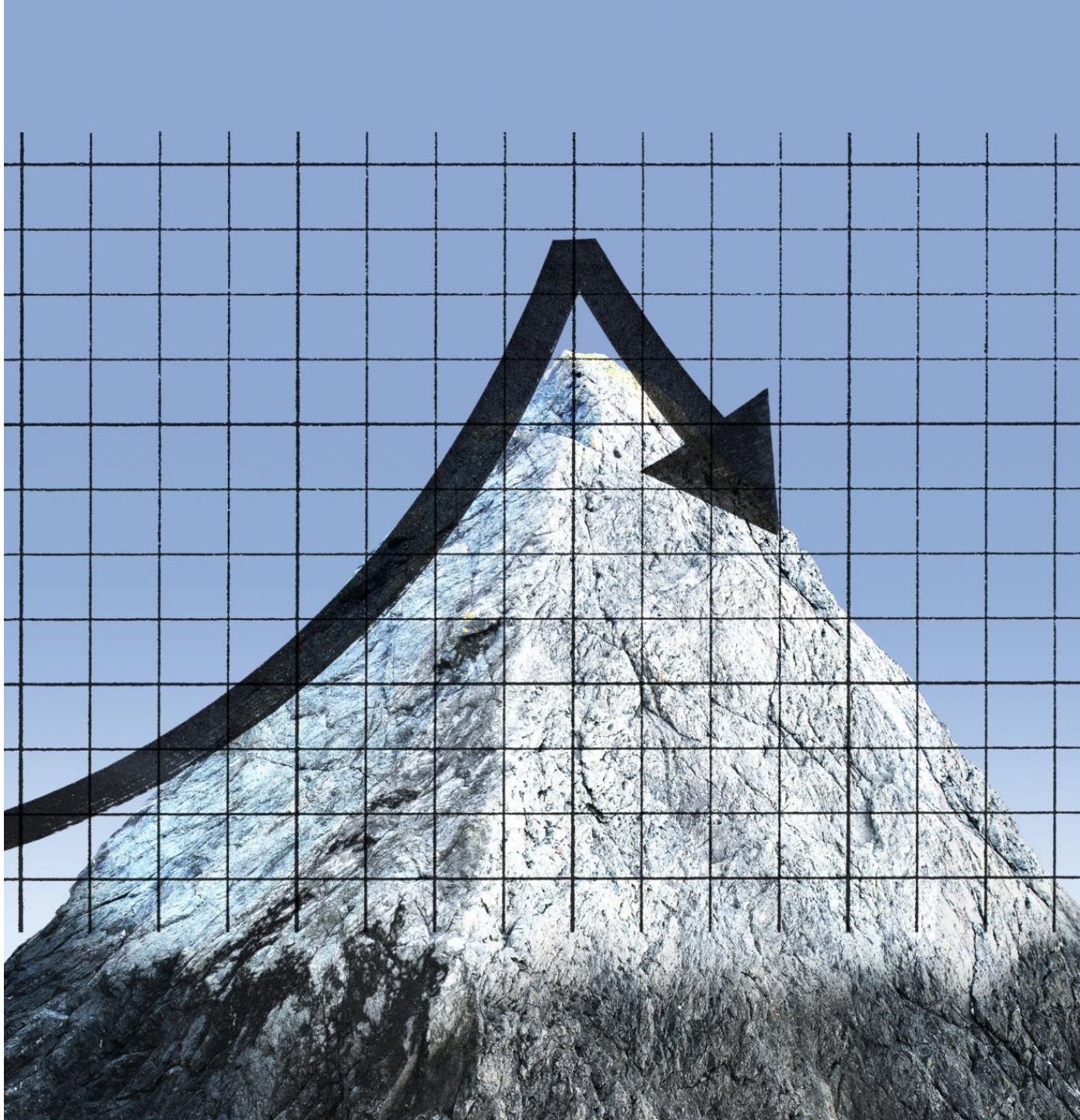
Effizientere Dialoge

Effizientere und produktivere Dialoge, besonders in komplexen oder fachspezifischen Bereichen.

CUSTOM MODEL

Trainiert auf Repositories des Unternehmens

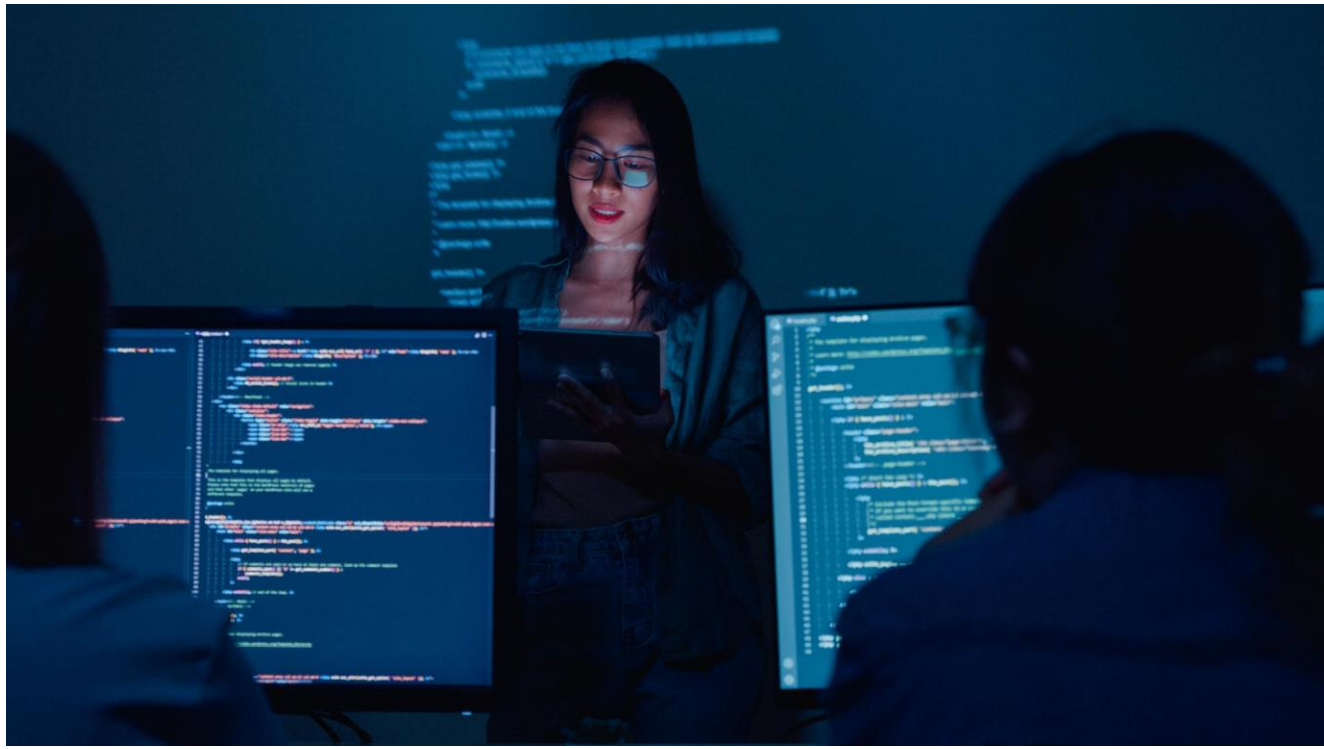
Dadurch deutlich verbesserte Code Vorschläge



KNOWLEDGE BASES

Unternehmensspezifischer Kontext

Organisationsbesitzer können eine Wissensdatenbank erstellen, die Markdown-Dokumentation in einem oder mehreren Repositorys zusammenführt. Anschließend können Organisationsmitglieder diese Wissensdatenbank als Kontext für Copilot Chat in GitHub verwenden.



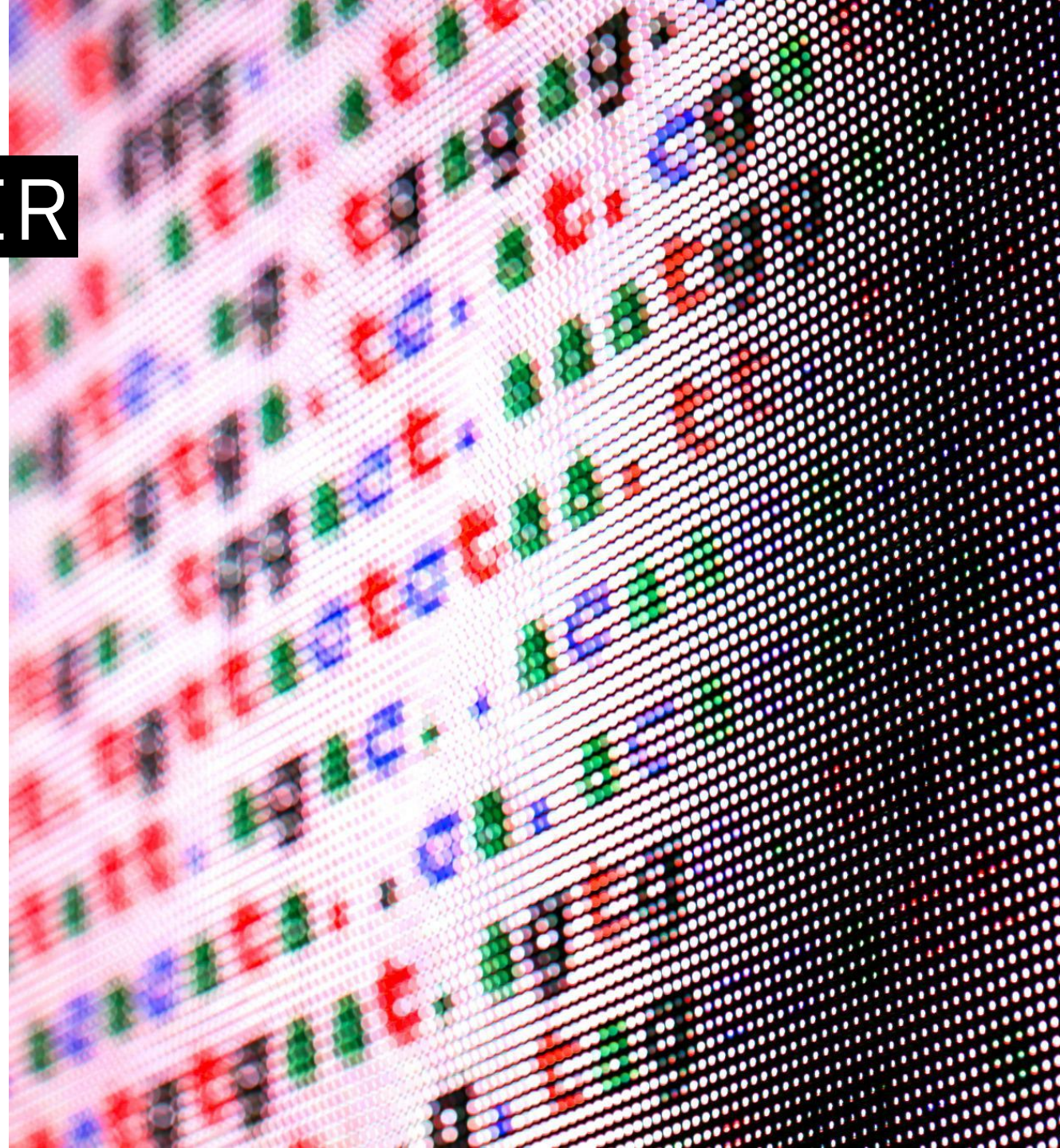
PUBLIC CODE FILTER

Filter für öffentlich zugänglichen Code

Können entweder komplett deaktiviert werden oder es werden Warnungen angezeigt.

Ermöglicht Schutz vor Copyright Problemen

Falls der Filter aktiviert ist, übernimmt Github die Verantwortung für Copyright Klagen



PROMPTMETHODEN: ZERO-SHOT, ONE-SHOT UND FEW-SHOT LEARNING

DEFINITION UND UNTERSCHIEDE DER PROMPTMETHODEN

Zero-shot Learning

Zero-shot Learning ermöglicht es Modellen, ohne Trainingsbeispiele neue Aufgaben zu bewältigen, indem es auf vorhandenes Wissen zurückgreift.

One-shot Learning

One-shot Learning verwendet ein einziges Beispiel, um ein Modell zu trainieren, was eine hohe Effizienz beim Lernen ermöglicht.

Few-shot Learning

Few-shot Learning erfordert mehrere Beispiele, um ein Modell zu trainieren, was eine flexible Anpassung an neue Aufgaben ermöglicht.

ZERO-SHOT LEARNING

```
1    # I need a function to convert Celsius to Fahrenheit
2
3    def celsius_to_fahrenheit(celsius):
4
5        fahrenheit = (celsius * 9/5) + 32
6        return fahrenheit
7
```

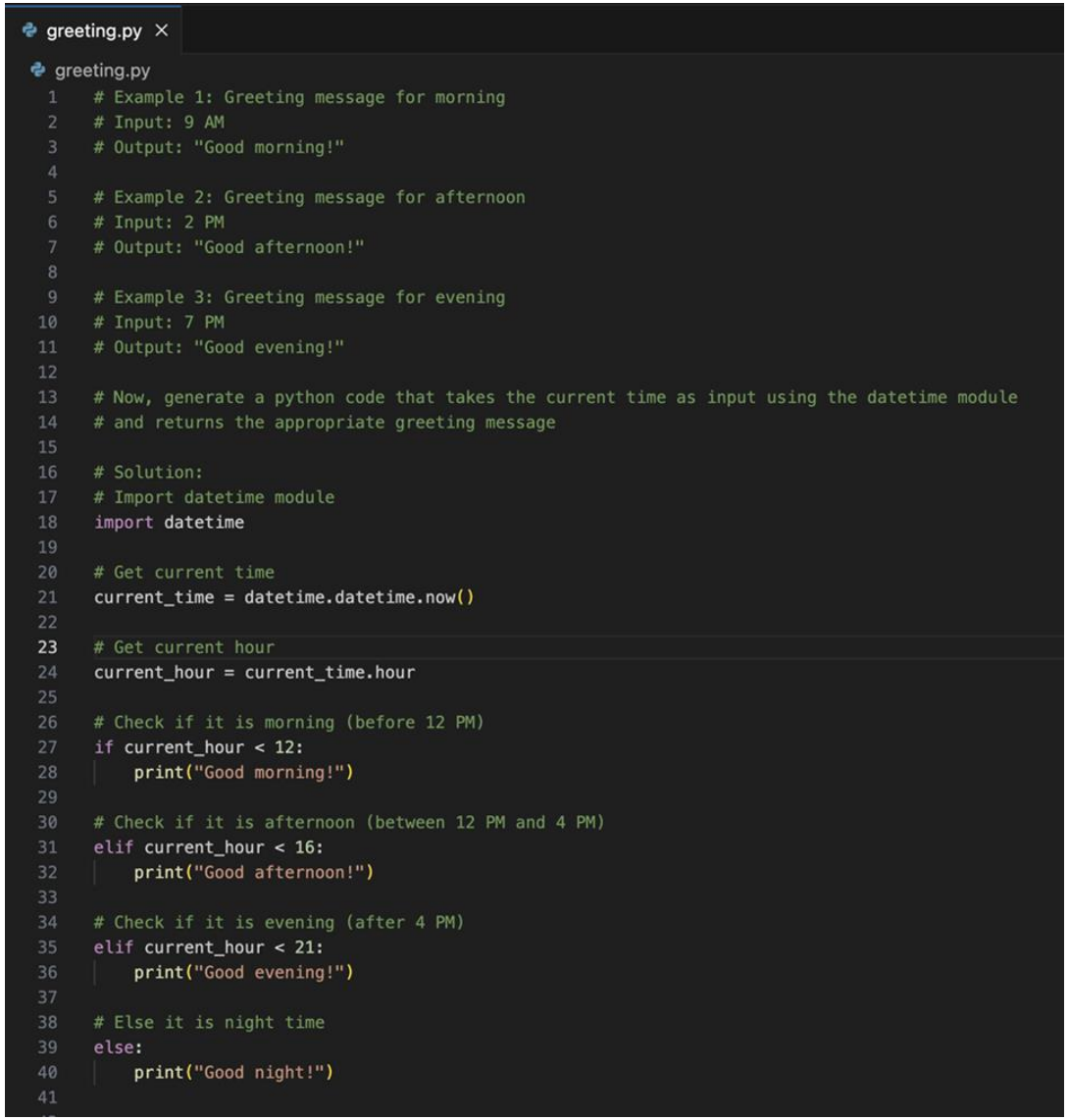
ONE-SHOT LEARNING

```python
# Example: Function to convert Fahrenheit to Celsius
def fahrenheit_to_celsius(fahrenheit):
    celsius = (fahrenheit - 32) * 5/9
    return celsius

# Now I need a function to convert Celsius to Fahrenheit

def celsius_to_fahrenheit(celsius):
    fahrenheit = celsius * 9/5 + 32
    return fahrenheit
```

FEW-SHOT LEARNING



VORTEILE UND HERAUSFORDERUNGEN JEDER METHODE

Zero-shot Learning

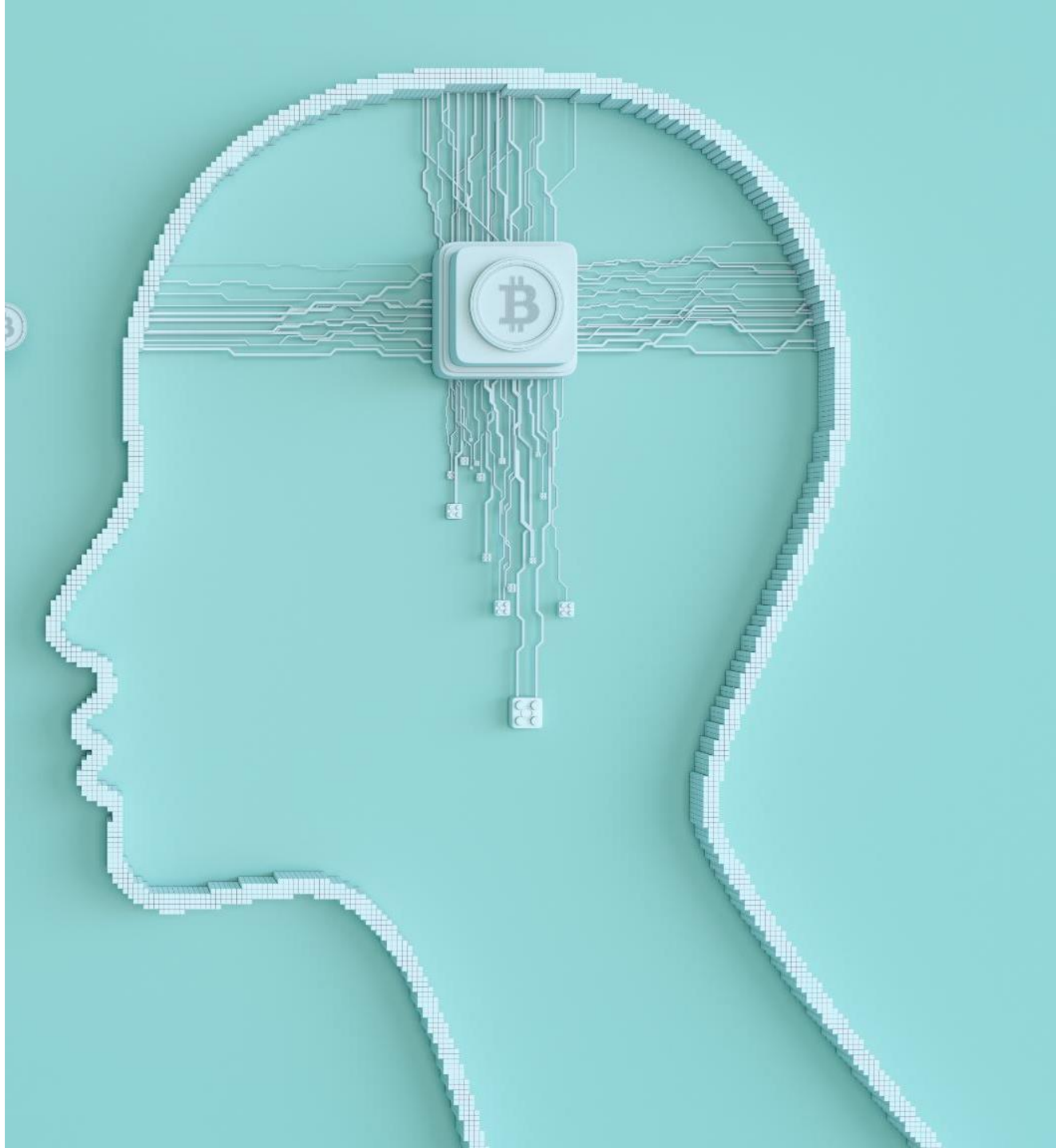
Zero-shot Learning bietet Flexibilität, indem es ermöglicht, neue Aufgaben ohne vorherige Beispiele zu bewältigen, leidet jedoch unter Genauigkeitsproblemen.

One-shot Learning

One-shot Learning erzielt präzisere Ergebnisse, da es nur ein Beispiel benötigt, setzt jedoch eine gute Datenqualität voraus.

Few-shot Learning

Few-shot Learning kombiniert die Vorteile von One-shot und Zero-shot Learning, benötigt jedoch mehr Daten für effektive Ergebnisse.



ZUKUNFTSAUSSICHTEN UND GRENZEN VON GITHUB COPILOT

ZUKUNFTSAUSSICHT

Coding-Agenten übernehmen die Programmierung

Zukünftig könnten intelligente Coding-Agenten eigenständig große Teile der Programmierung übernehmen und Entwickler bei Routineaufgaben entlasten.

Stärkerer Einfluss auf die Softwarearchitektur

Durch den Einsatz von GitHub Copilot könnten KI-Systeme künftig stärker Einfluss auf die Softwarearchitektur nehmen, indem sie Architekturmuster empfehlen, optimale Strukturen vorschlagen und Entscheidungen auf Grundlage bewährter Architekturprinzipien unterstützen.

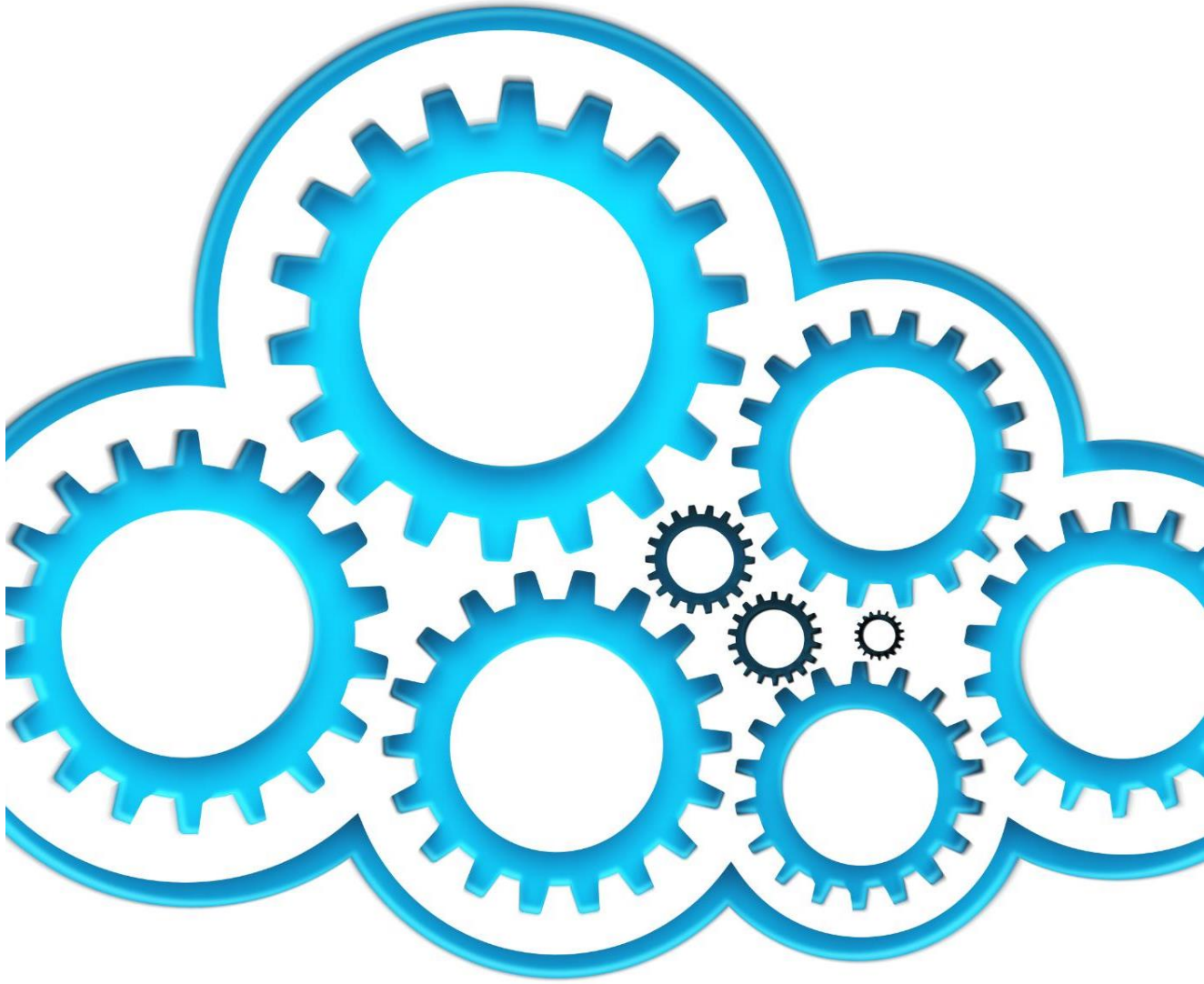
GRENZEN

Qualität der Vorschläge

Die Qualität der Vorschläge ist entscheidend für die Benutzerzufriedenheit und die Effektivität des Tools. Mängel in diesem Bereich können die Benutzererfahrung stark beeinträchtigen.

Überwachung und Kontrolle notwendig

Menschliche Kontrolle und Überwachung ist notwendig, da KI Fehler machen kann.



GEFAHREN

Verlust der Fähigkeiten

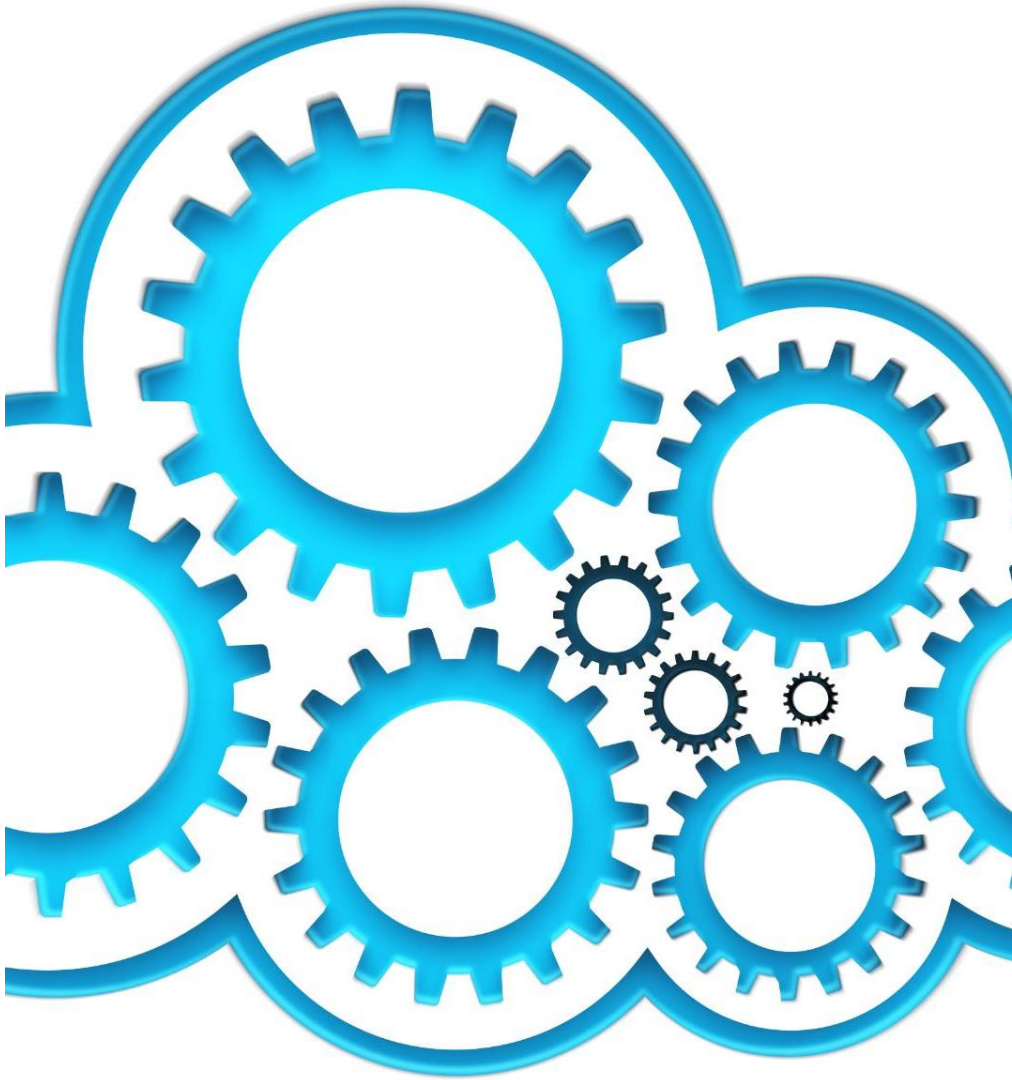
Programmierer verlassen sich komplett auf KI und verlernen selbst zu coden.

Verstärkung von Vorurteilen (Bias)

Trainingsdaten können implizit bestehende Vorurteile enthalten, die von KI-Systemen übernommen oder sogar verstärkt werden.

Fehleranfälligkeit und mangelnde Transparenz

KI-Systeme können unerwartete Fehler produzieren, und es ist oft schwierig nachzuvollziehen, auf welcher Grundlage Entscheidungen getroffen werden (Black-Box-Problematik).



ETHIK UND VERANTWORTUNG IN DER NUTZUNG VON KI IM PROGRAMMIEREN

Ethische Fragestellungen

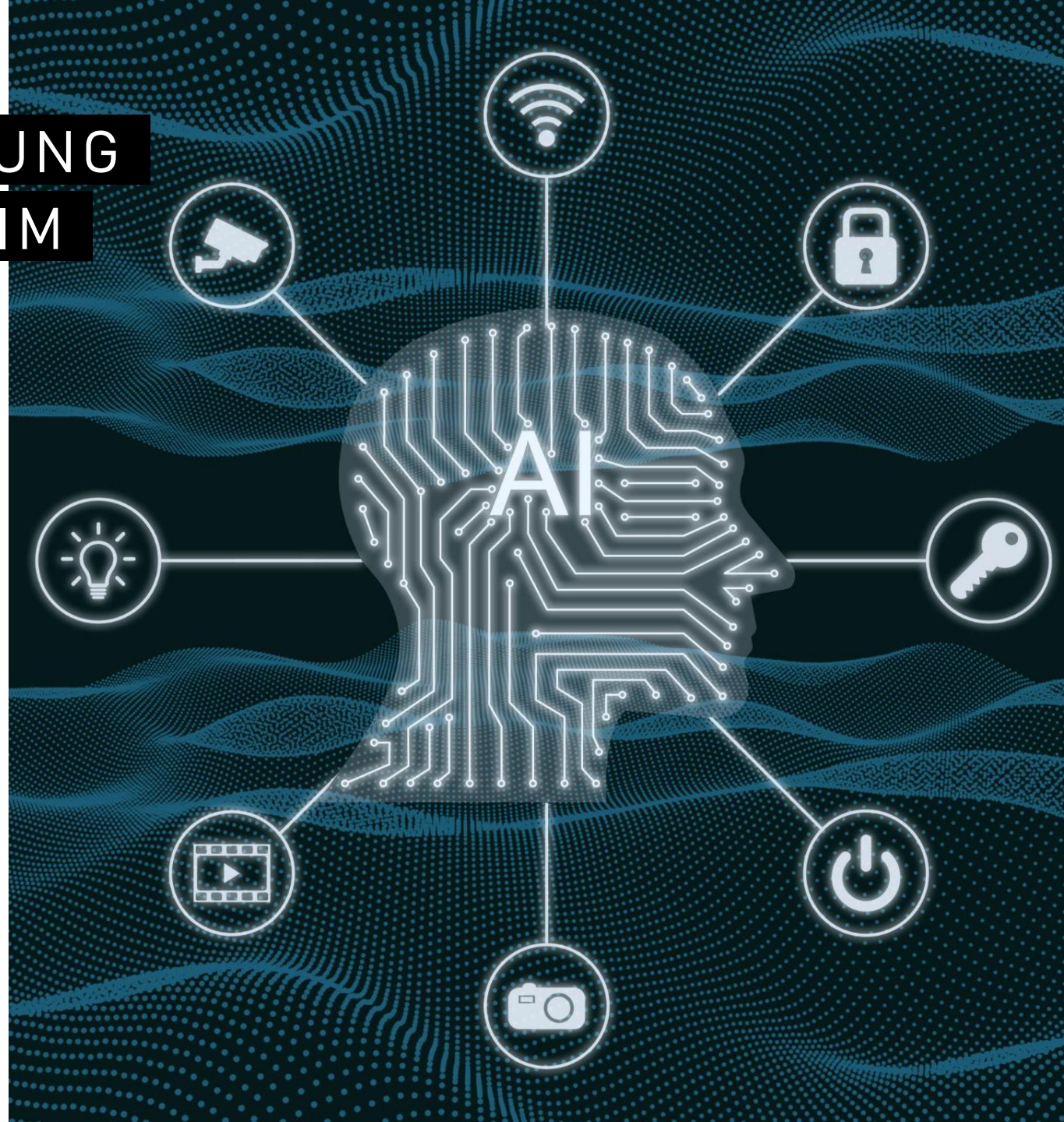
Die Nutzung von KI wirft wichtige ethische Fragen auf, die bedacht werden müssen, um negative Auswirkungen zu vermeiden.

Sichere Anwendungen

Es ist entscheidend, KI-Technologien sicher zu implementieren, um Benutzer und Daten zu schützen.

Verantwortungsbewusste Nutzung

Die verantwortungsvolle Nutzung von KI ist notwendig, um Gerechtigkeit und Fairness sicherzustellen.



SCHLUSSFOLGERUNG

Revolution der Programmierung

GitHub Copilot könnte die Programmierung revolutionieren, indem es Entwicklern ermöglicht, schneller und kreativer zu arbeiten.

Effizienz und Kreativität

Durch intelligente Funktionen unterstützt Copilot Entwickler dabei, effizientere Lösungen zu finden und ihre Kreativität auszuleben.

Herausforderungen und Ethik

Es ist wichtig, die Herausforderungen und ethischen Fragen, die durch AI-gestützte Programmierung entstehen, sorgfältig zu betrachten.