

# KI-gestützte Git-Commit-Nachrichten

Dieses Python-Skript sollte in einem Verzeichnis platziert werden, das in Ihrem System-PATH enthalten ist, wie z.B. ~/bin.

```
import subprocess
import os
from openai import OpenAI
from dotenv import load_dotenv
import argparse

load_dotenv()

def gitmessageai(push=True, only_message=False):
    # Stage alle Änderungen
    subprocess.run(["git", "add", "-A"], check=True)

    # Hole den Diff der gestagten Änderungen
    diff_process = subprocess.run(["git", "diff", "--staged"], capture_output=True, text=True, check=True)
    diff = diff_process.stdout

    if not diff:
        print("Keine Änderungen zum Commit vorhanden.")
        return

    # Bereite die Eingabeaufforderung für die KI vor
    prompt = f"""
```

Generiere eine prägnante Commit-Nachricht im Conventional Commits-Format für die folgenden Code-Änderungen.  
Verwende einen der folgenden Typen: feat, fix, docs, style, refactor, test, chore, perf, ci, build oder  
Falls zutreffend, füge einen Bereich in Klammern hinzu, um den betroffenen Teil der Codebasis zu beschreiben.  
Die Commit-Nachricht sollte 70 Zeichen nicht überschreiten.

Code-Änderungen:

```
{diff}
```

Commit-Nachricht:

```
"""
```

```
# Sende die Eingabeaufforderung an die DeepSeek API
api_key = os.environ.get("DEEPSEEK_API_KEY")
if not api_key:
    print("Fehler: DEEPSEEK_API_KEY Umgebungsvariable nicht gesetzt.")
    return

client = OpenAI(api_key=api_key, base_url="https://api.deepseek.com")

try:
    response = client.chat.completions.create(
        model="deepseek-chat",
        messages=[
            {"role": "user", "content": prompt}
        ],
        max_tokens=100
    )
    if response and response.choices:
        commit_message = response.choices[0].message.content.strip()
        commit_message = commit_message.replace('\n', ' ')
    else:
        print("Fehler: Keine Antwort von der API.")
        return
except Exception as e:
    print(f"Fehler beim API-Aufruf: {e}")
    return

# Überprüfe, ob die Commit-Nachricht leer ist
if not commit_message:
    print("Fehler: Leere Commit-Nachricht generiert. Commit wird abgebrochen.")
```

```

        return

    if only_message:
        print(f"Vorgeschlagene Commit-Nachricht: {commit_message}")
        return

    # Commit mit der generierten Nachricht
    subprocess.run(["git", "commit", "-m", commit_message], check=True)

    # Pushe die Änderungen
    if push:
        subprocess.run(["git", "push"], check=True)
    else:
        print("Änderungen lokal committed, aber nicht gepusht.")

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Generiere eine Commit-Nachricht mit KI und committe d
    parser.add_argument('--no-push', dest='push', action='store_false', help='Committe Änderungen lokal
    parser.add_argument('--only-message', dest='only_message', action='store_true', help='Nur die KI-ge
    args = parser.parse_args()
    gitmessageai(push=args.push, only_message=args.only_message)

```

Fügen Sie dann in Ihrer ~/.zprofile-Datei Folgendes hinzu:

```

alias gpa='python ~/bin/gitmessageai.py'
alias gca='python ~/bin/gitmessageai.py --no-push'
alias gm='python ~/bin/gitmessageai.py --only-message'

```