

Stappenplan: Applicatie met Streamlit Refactoren

1. Lokaal voorbereiden en stapsgewijze refactoring

1.1 Virtual environment & dependencies

1.1.1 Maak of vernieuw je virtuele omgeving (.venv):

```
python3 -m venv .venv  
source .venv/bin/activate
```

1.1.2 Installeer externe pakketten:

```
python3 -m pip install openai requests python-dotenv streamlit
```

1.1.3 Leg je dependencies vast:

```
python3 -m pip freeze > requirements.txt
```

1.2 Tests schrijven vóór je refactor

1.2.1 Installeer pytest:

```
pip install pytest
```

1.2.2 Maak map tests/ en voeg basistests toe voor elke module (core en io).

1.2.3 Draai tests om te garanderen dat de huidige code werkt:

```
pytest -q
```

1.3 Projectstructuur met Streamlit UI

- Maak entry-point voor de Streamlit-app: app.py in de root.
- Verplaats UI-logica (Streamlit widgets) naar app.py en laat deze core-functies aanroepen.
- Herstructureer de mappen zo:
 - core/ (business-logic en AI-toetsing)
 - io/ (config-loader, file I/O)
 - app.py (Streamlit interface)
 - tests/ (unit-tests)

1.4 Identificeer en refactor “risicozones”

- Configuratie-loading via config_loader.py.
- Logging centralisatie in log_definitie.py.
- Scheid business-logic (core) van I/O (io).

1.5 Code-stijl & statische checks

1.5.1 Installeer en run Black, isort, flake8, mypy:

```
pip install black isort flake8 mypy
```

1.5.2 Na elke refactoringset draai je:

```
black .  
isort .  
flake8 .  
mypy .
```

1.6 Continue validatie en Streamlit smoke-tests

1.6.1 Draai unit-tests bij iedere wijziging:

```
pytest -q
```

1.6.2 Smoke-test de Streamlit-app handmatig:

```
streamlit run app.py
```

1.6.3 Controleer dat UI en backend correct samenwerken.

2. Versiebeheer & CI via GitHub

2.1 Git initialiseren

2.1.1 Ga naar project-root en initieer git:

```
cd /pad/naar/DefinitieAgentTotaal
git init
git add .
git commit -m "Initial setup with Streamlit UI and tests"
```

2.2 Branch-strategie voor refactoring

- main: stabiele versie met core en I/O.
- refactor/config-loader
- refactor/logging-setup
- refactor/streamlit-ui

2.3 GitHub Actions voor automatisering

2.3.1 Maak .github/workflows/ci.yml met:

```
name: CI

on: [push, pull_request]

jobs:
  test-and-lint:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Setup Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.x'
      - name: Install dependencies
        run: |
          python3 -m venv .venv
          source .venv/bin/activate
          pip install -r requirements.txt
      - name: Lint & type-check
        run: |
          black --check .
          isort --check .
          flake8 .
          mypy .
      - name: Run tests
        run: pytest -q
      - name: Smoke-test Streamlit import
        run: python -c "import streamlit; print(streamlit.__version__)"
```

2.3.2 Commit & push deze workflow.

2.4 Verder optimaliseren

- Maak branches per refactor-stap: kleine, beheerbare wijzigingen.
- Open Pull Requests voor review en feedback.
- Gebruik GitHub Issues voor bugs en features.
- Overweeg deployment (Docker, Streamlit sharing).