Execute Task: 1. Set up project structure and core infrastructure 1. Set up project structure and core infrastructure

**Kiro**

Including Steering Documents

GenRules.md

I'll implement task 1 to set up the project structure and core infrastructure for the GITTE federated learning system. Let me start by updating the task status and then implementing each sub-task.

Task: *1. Set up project structure and core infrastructure* from tasks.md

Status: In Progress

Now let me implement the sub-tasks one by one:

**1. Create directory structure following 4-layer architecture**

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

Created

\_\_init\_\_.py

**2. Set up Docker Compose with PostgreSQL, MinIO, Ollama containers**

Created

docker-compose.yml

Created

Dockerfile

Created

requirements.txt

**3. Create Makefile with dev, test, migrate, seed, run targets**

Created

Makefile

**4. Initialize configuration management system with feature flags**

Created

config.py

Created

.env.example

Created

.gitignore

Now let me create a basic main application entry point and some utility scripts:

Created

main.py

Created

seed\_database.py

Created

alembic.ini

Created

env.py

Created

script.py.mako

Now let me create a basic test structure:

Created

conftest.py

Created

test\_config.py

Now let me create a README file to document the project structure:

Created

README.md

Now let me run the tests to verify the configuration system works correctly:

Command

python -m pytest tests/test\_config.py -v

PS E:\Forschung\KIRO\_GITTE> python -m pytest tests/test\_config.py -v

================================================================================= test session starts =================================================================================

platform win32 -- Python 3.10.11, pytest-8.4.1, pluggy-1.6.0 -- E:\Program Files\Python310\python.exe

cachedir: .pytest\_cache

rootdir: E:\Forschung\KIRO\_GITTE

plugins: anyio-4.6.2.post1, langsmith-0.3.45, cov-6.2.1

collected 9 items

tests/test\_config.py::test\_config\_initialization PASSED [ 11%]

tests/test\_config.py::test\_feature\_flags\_initialization PASSED [ 22%]

tests/test\_config.py::test\_environment\_variable\_override PASSED [ 33%]

tests/test\_config.py::test\_feature\_flag\_environment\_override PASSED [ 44%]

tests/test\_config.py::test\_get\_feature\_flag PASSED [ 55%]

tests/test\_config.py::test\_text\_localization PASSED [ 66%]

tests/test\_config.py::test\_config\_validation\_development PASSED [ 77%]

tests/test\_config.py::test\_config\_validation\_production\_fails PASSED [ 88%]

tests/test\_config.py::test\_config\_validation\_production\_passes PASSED [100%]

================================================================================== 9 passed in 0.05s ==================================================================================

Great! All tests pass. Now let me test the basic Streamlit application:

Command

python -c "from src.ui.main import main; print('✅ Streamlit app imports successfully')"

PS E:\Forschung\KIRO\_GITTE> python -c "from src.ui.main import main; print(' Streamlit app imports successfully')"

Streamlit app imports successfully

Perfect! Now let me verify the Docker Compose configuration is valid:

Command

docker-compose config --quiet

PS E:\Forschung\KIRO\_GITTE> docker-compose config --quiet

time="2025-08-10T16:38:44+02:00" level=warning msg="E:\\Forschung\\KIRO\_GITTE\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid po

tential confusion"

The Docker Compose configuration is valid (the version warning is just about the version field being deprecated but still works). Let me test the Makefile:

Command

make help

PS E:\Forschung\KIRO\_GITTE> make help

make : Die Benennung "make" wurde nicht als Name eines Cmdlet, einer Funktion, einer Skriptdatei oder eines ausführbaren Programms erkannt. Überprüfen Sie die Schreibweise des

Namens, oder ob der Pfad korrekt ist (sofern enthalten), und wiederholen Sie den Vorgang.

In Zeile:1 Zeichen:1

+ make help

+ ~~~~

+ CategoryInfo : ObjectNotFound: (make:String) [], CommandNotFoundException

+ FullyQualifiedErrorId : CommandNotFoundException

The make command isn't available on Windows. Let me create a PowerShell equivalent script for Windows users:

Created

dev.ps1

Let me test the PowerShell script:

Command

.\scripts\dev.ps1 help

PS E:\Forschung\KIRO\_GITTE> .\scripts\dev.ps1 help

In E:\Forschung\KIRO\_GITTE\scripts\dev.ps1:98 Zeichen:50

+ ... Write-Host "âŒ Unknown command: $Command" -ForegroundColor Red

+ ~~~~~~~~~~~~~~~~~~~~~~

Die Zeichenfolge hat kein Abschlusszeichen: ".

In E:\Forschung\KIRO\_GITTE\scripts\dev.ps1:73 Zeichen:20

+ function Show-Logs {

+ ~

Die schließende "}" fehlt im Anweisungsblock oder der Typdefinition.

+ CategoryInfo : ParserError: (:) [], ParseException

+ FullyQualifiedErrorId : TerminatorExpectedAtEndOfString

There's an encoding issue with the PowerShell script. Let me fix it:

14 edits to file

dev.ps1

Now let me test the PowerShell script again:

Command

.\scripts\dev.ps1 help

PS E:\Forschung\KIRO\_GITTE> .\scripts\dev.ps1 help

Available commands:

help - Show this help message

dev - Start development environment

test - Run test suite

migrate - Run database migrations

seed - Seed database with initial data

run - Run the application locally

build - Build Docker images

up - Start all services with Docker Compose

down - Stop all services

logs - Show logs from all services

clean - Clean up containers and volumes

Usage: .\scripts\dev.ps1 <command>

Perfect! Now let me test the test command:

Command

.\scripts\dev.ps1 test

PS E:\Forschung\KIRO\_GITTE> .\scripts\dev.ps1 test

Running test suite...

======================================================================== test session starts =========================================================================

platform win32 -- Python 3.10.11, pytest-8.4.1, pluggy-1.6.0 -- E:\Program Files\Python310\python.exe

cachedir: .pytest\_cache

rootdir: E:\Forschung\KIRO\_GITTE

plugins: anyio-4.6.2.post1, langsmith-0.3.45, cov-6.2.1

collected 9 items

tests/test\_config.py::test\_config\_initialization PASSED [ 11%]

tests/test\_config.py::test\_feature\_flags\_initialization PASSED [ 22%]

tests/test\_config.py::test\_environment\_variable\_override PASSED [ 33%]

tests/test\_config.py::test\_feature\_flag\_environment\_override PASSED [ 44%]

tests/test\_config.py::test\_get\_feature\_flag PASSED [ 55%]

tests/test\_config.py::test\_text\_localization PASSED [ 66%]

tests/test\_config.py::test\_config\_validation\_development PASSED [ 77%]

tests/test\_config.py::test\_config\_validation\_production\_fails PASSED [ 88%]

tests/test\_config.py::test\_config\_validation\_production\_passes PASSED [100%]

========================================================================= 9 passed in 0.04s ==========================================================================

Excellent! Now let me update the README to include Windows-specific instructions:

Accepted edits to

README.md

Now let me create a final verification by checking that all the required directories and files are in place:

Searched workspace

Verify the complete project structure has been created correctly

Perfect! Let me also verify the 4-layer architecture structure:

Searched workspace

Verify the 4-layer architecture directories are properly created

Read file(s)

\_\_init\_\_.py

\_\_init\_\_.py

\_\_init\_\_.py

\_\_init\_\_.py

Excellent! Now let me mark the task as completed:

Task: *1. Set up project structure and core infrastructure* from tasks.md

Status: Completed

**Task 1 Implementation Complete! ✅**

I have successfully implemented all sub-tasks for **Task 1: Set up project structure and core infrastructure**:

**✅ Completed Sub-tasks:**

1. **Created directory structure following 4-layer architecture**:
   * src/ui/ - UI Layer (Streamlit interfaces and display logic)
   * src/logic/ - Logic Layer (Business logic, workflow orchestration, decision making)
   * src/services/ - Service Layer (External service integration, data transformation, caching)
   * src/data/ - Data Layer (Data persistence, schema management, migrations)
   * Additional directories: config/, tests/, migrations/, scripts/
2. **Set up Docker Compose with PostgreSQL, MinIO, Ollama containers**:
   * Complete docker-compose.yml with all required services
   * Dockerfile for the main application
   * Proper networking and volume configuration
   * Environment variable configuration
3. **Created Makefile with dev, test, migrate, seed, run targets**:
   * Complete Makefile with all required targets
   * Additional PowerShell script (scripts/dev.ps1) for Windows users
   * Both provide identical functionality across platforms
4. **Initialized configuration management system with feature flags**:
   * Comprehensive config/config.py with all configuration sections
   * Environment variable override support
   * Feature flags system for runtime behavior control
   * Text localization support
   * Configuration validation for production environments

**🧪 Verification:**

* All tests pass (9/9 tests in tests/test\_config.py)
* Streamlit application imports successfully
* Docker Compose configuration is valid
* PowerShell development script works on Windows
* Project structure follows the specified 4-layer architecture

**📋 Requirements Satisfied:**

* **1.1, 1.2, 1.3**: 4-layer architecture implemented and enforced
* **13.1, 13.2**: Docker Compose and development infrastructure complete
* **17.1, 17.2**: Configuration management with environment variable overrides

The core infrastructure is now ready for implementing the remaining tasks. The system provides a solid foundation with proper separation of concerns, comprehensive configuration management, and cross-platform development support.