






SWT Waterfall Project

Tech Stack

Javascript (Frontend)	
Flask (Backend)	
SQLite (DB)	

Backend extensions:

Jinja (Template Engine)	
SQLAlchemy (ORM)?	

Flask

Python version	3.10.0
-----------------------	---------------

1. Git Repository aus flask branch pullen: „git pull origin flask“.
Falls noch kein lokales Git Repository erstellt, bitte zu „*Git einrichten*“ scrollen, dann hier weitermachen.
2. Python installieren, Version 3.10.
(Check if python is installed via: „python --version“)
3. Python Virtual Environment erstellen. Geeignetes Verzeichnis dafür suchen.
 - a. „python -m venv .venv“
4. Python Virtual Environment aktivieren
 - a. Windows: „.venv\Scripts\activate.bat“.
Falls Skripte deaktiviert sind, zuerst: „Set-ExecutionPolicy Unrestricted -Scope Process“.
 - b. Linux: „source .venv/bin/activate“.
5. Notwendige Packages installieren: „python -m pip install -r requirements.txt“.
6. IntelliJ öffnen, Projekt öffnen, main.py öffnen, rechtsklick, „Run ‘main.py‘“, URL in Konsole anklicken.
7. Ta-da!

Git

Git einrichten (Bash):

1. Git Bash installieren.
2. GitBash in geeignetem Verzeichnis öffnen.
3. Lokales git initialisieren: „*git init*“.
4. Remote Repository hinzufügen: „*git remote add origin* <https://github.com/kr1pt0n05/Softwaretechnik-Waterfall-Project.git>“.

Änderungen auf GitHub pushen:

1. *git add .*
2. *git commit -m „Deine Änderungen kurz beschreiben“*
3. *git push -u origin main*

Beachte, zuerst immer Repositories synchronisieren: „*git pull*“.

Andere Nützliche Befehle:

git status
git ls-files
git branch
git checkout -b BRANCH
git rm -r --cached .

SQLite

Referenz:

```
import sqlite3

app = Flask(__name__)

@app.route("/")
def index():
    con = sqlite3.connect("sqlite3.db")
    cur = con.cursor()
    cur.execute("SELECT * FROM temperatures")
    temps = cur.fetchall()
    cur.execute("SELECT id FROM temperatures ORDER BY id DESC LIMIT 1")
    id = cur.fetchone()
```

Jinja

Ist bereits in Flask integriert. Erlaubt Python Code in html-files auszuführen. Jina läuft im Backend, d.h. der Python Code wird serverseitig ausgeführt und nicht im Frontend des Nutzers.

Beispiel siehe: index.html

SQLAlchemy

Statt lästige SQL Befehle schreiben zu müssen, können wir unsere Daten auch in Python Klassen verpacken und dann in unsere Datenbank schieben.

Die Library übersetzt unsere Python Klasse quasi in SQL.

Ordnerstruktur

/flask

```
|
|
|— main.py      # Backend Logik, Routes & Views
|
|— static/      # Hier kommen .css, .js & Bilder rein
|  |— index.js
|  |— styles.css
|  └─ images/   # Hier Bilder rein
|
└─ templates/  # Hier kommen .html rein
    └─ layout.html # Base layout template
```

└─ **index.html** **# Example template**