1: Feature-Extraktion (3 Metriken)

Ziel: Farbbasierte, neuronale und semantische Features

- `color_histogram.py` : HSV-Histogramm (OpenCV)
- `cnn_embedding.py` : ResNet50-Features (Keras)
- `third_metric.py` :
  - Option 1: Gabor-Filter + PCA
  - Option 2: Autoencoder-Bottleneck-Vektor
  - Option 3: CLIP-Embedding (OpenAI, nur Text-Bild optional)

Output: `.npy` -Dateien mit Feature-Vektoren pro Bild

2: Relationale Datenbank (Metadata & Pfade)

Ziel: SQLite-DB für Pfad, Größe, Auflösung, Vektorpfad

- `image_db.py` : `create_table()` , `insert()` , `query_by_id()`
- Nutzung von `sqlite3`
- Zusatz: Einbindung von EXIF-Daten (Kameramodell etc.)

3: ANN-Suche mit Faiss (Skalierung)

Ziel: Ähnliche Bilder schnell finden (top-k Suche in Sekunden)

- `faiss_index.py` : Index trainieren, laden, durchsuchen
- Optimierung: Auf Embedding-Vektoren reduziert

4: Kombination von Metriken

Ziel: Ähnlichkeit auf Basis mehrerer Metriken berechnen

`metric_combination.py` :

```
def combine_scores(score1, score2, alpha=0.5):
    return alpha * score1 + (1 - alpha) * score2
```

5: Dimensionalität & Visualisierung

Ziel: Position aller Bilder im 2D/3D-Raum darstellen

- `umap_projection.py` :
  - UMAP, t-SNE, optional T-MAP
  - interaktive Plotly-Grafik mit Hover-Image

6: Streamlit-GUI mit Useroptionen

Ziel: Upload → Anzeige Top-5 ähnlicher Bilder

- `streamlit_app.py` :
  - Upload-Button
  - Dropdown: Auswahl der Metrik(en)
  - Vorschau ähnliche Bilder
  - Optional: zwei Bilder gleichzeitig hochladen (kombinierte Suche)

7: Bewertung & Optimierung

Ziel: Laufzeitanalyse + Qualität der Metriken

- `evaluate_metrics.py` :
  - Precision@5, Recall, ggf. Manuelles Rating
- `runtime_profiling.py` :
  - Nutzung von `cProfile` , Visualisierung mit Snakeviz

8: Tests + Clean Code + Dokumentation

- Unit-Tests:
  - `pytest` für alle Funktionen (Metriken, DB, Generator)
- `README.md` + PDF:
  - Motivation, Designskizze, Beschreibung Metriken & Ergebnisse