Objektorientierte und funktionale Programmierung mit Python (DLBDSOOFPP01_D)

Konzeptdokument

Studiengang: Softwareentwicklung

Datum: 11. Juni 2025

Name: Yannic Dykow

Matrikelnummer: 3220807

1. Zieldefinition

Hauptziele:

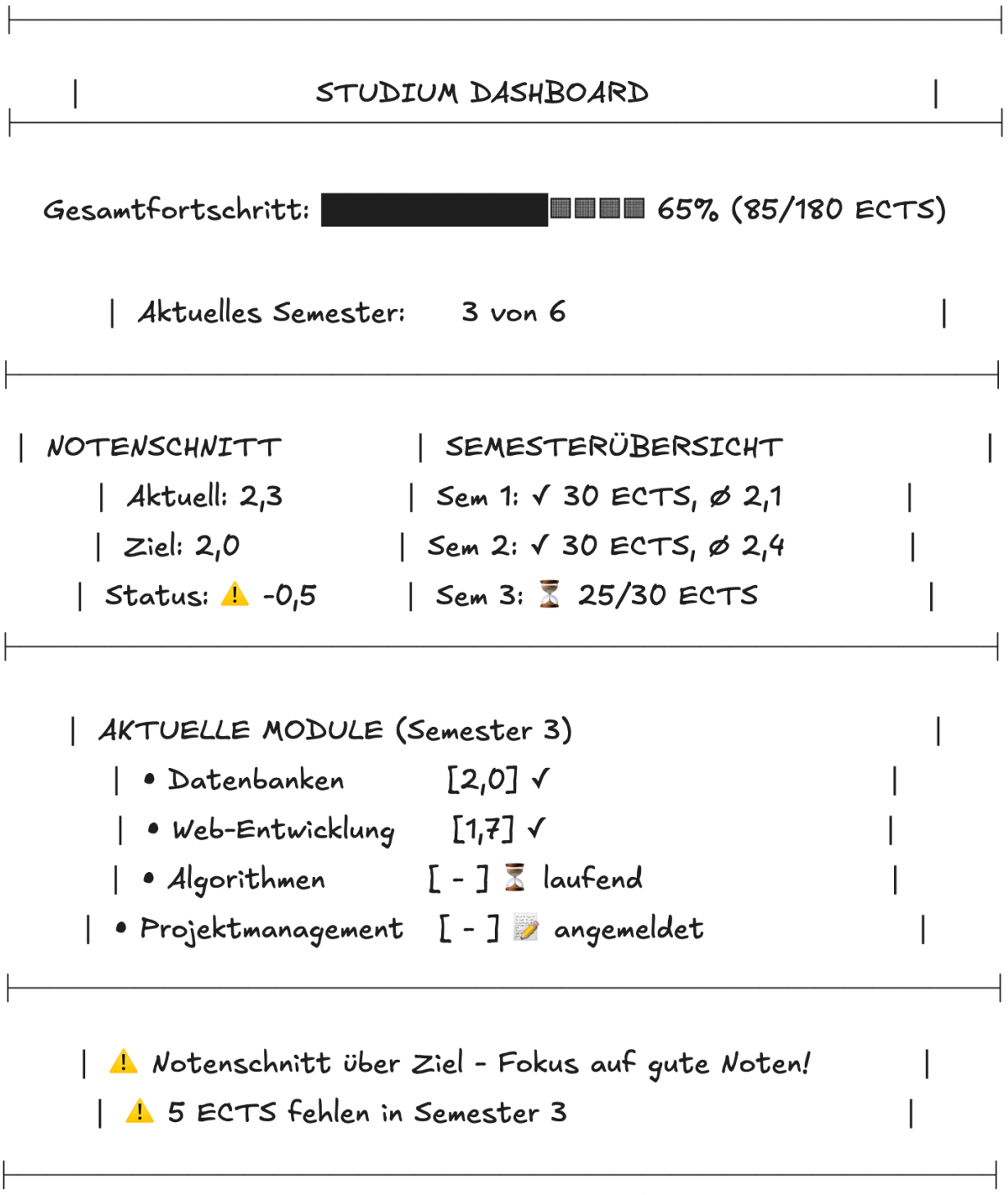
- Zeitliches Ziel: Abschluss des Studiums in 6 Semestern (Regelstudienzeit)
- Qualitätsziel: Notendurchschnitt von 2,0 oder besser

Nebenziel:

- Fortschrittsziel: Mind. 80% der geplanten ECTS pro Semester erreichen

Diese Ziele ermöglichen eine klare Überwachung des Studienfortschritts sowohl zeitlich als auch qualitativ.

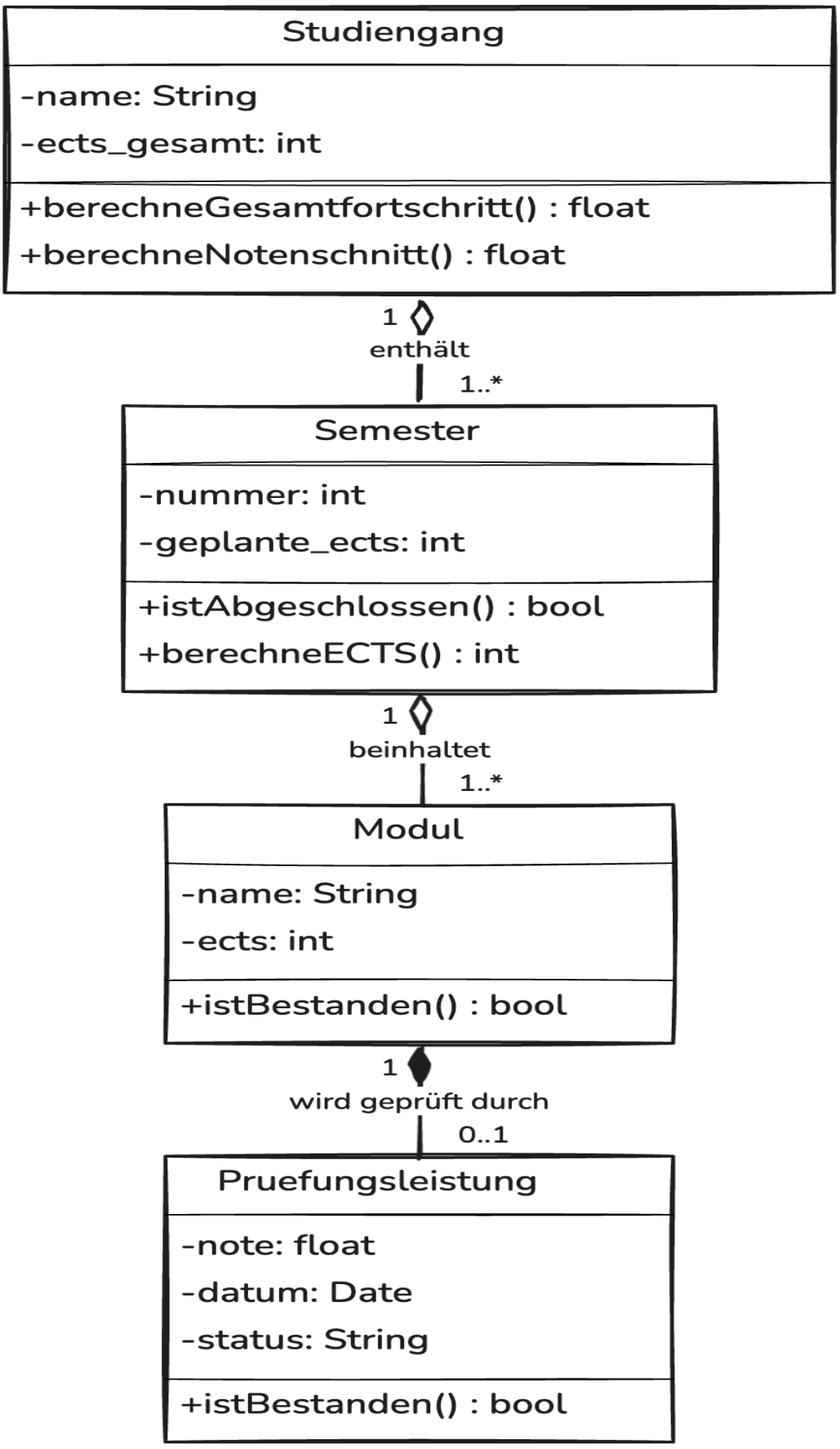
2. Dashboard-Design



Das Dashboard zeigt alle wesentlichen Informationen kompakt an:

- Gesamtfortschritt als Balken und Prozentangabe
- Aktueller vs. Ziel-Notenschnitt mit Warnung bei Abweichung
- Semesterübersicht mit ECTS-Fortschritt
- Detailansicht des aktuellen Semesters

3. UML-Klassendiagramm



Kritische Diskussion der Modellierung:

Beziehungen:

- Studiengang-Semester: Aggregation gewählt, da Semester theoretisch wiederverwendbar sind
- Semester-Modul: Aggregation, da Module in mehreren Semestern vorkommen können
- Modul-Prüfungsleistung: Komposition, da Prüfung ohne Modul keinen Sinn macht

Kritikpunkt an eigener Modellierung: Die Trennung zwischen Modul und Prüfungsleistung könnte problematisch sein, da ein Modul mehrere Prüfungsversuche haben kann. Alternative wäre eine 1..* Beziehung.

Verzicht auf Vererbung: Bewusst keine Vererbung verwendet, da keine klaren "ist-ein" Beziehungen vorliegen. Unterschiede (z.B. Pflicht-/Wahlmodul) werden über Attribute gelöst.

4. Technische Machbarkeit

Entwicklungsumgebung

- Python mit VS Code: Kostenlos, gute Python-Unterstützung
- Bibliotheken: tkinter (GUI), json (Datenspeicherung), datetime

Datenspeicherung

- JSON-Dateien: Einfach für Prototyp, keine DB-Installation nötig

Durchgeführte Tests:

Test 1 - JSON-Speicherung:

```python
import json
# Speichern
data = {"module": [{"name": "Mathe", "ects": 5, "note": 2.3}]}
with open("test.json", "w") as f:
    json.dump(data, f)
# Laden
with open("test.json", "r") as f:
    loaded = json.load(f)
print(loaded)
```

Test 2 - Einfaches tkinter-Fenster:

```python
import tkinter as tk
root = tk.Tk()
root.title("Dashboard Test")
label = tk.Label(root, text="Semester: 3")
label.pack()
# Progress bar simuliert
progress = tk.Canvas(root, width=200, height=20)
progress.create_rectangle(0, 0, 130, 20, fill="green")
progress.pack()
root.mainloop()
```

Geplante Umsetzung

Einfache Struktur mit 3 Python-Dateien:

- `models.py`: Entity-Klassen
- `gui.py`: Dashboard-Oberfläche
- `main.py`: Hauptprogramm

Begründung: Für einen Prototyp ausreichend, ohne überflüssige Komplexität.