Objektorientierte und funktionale Programmierung mit Python (DLBDSOOFPP01_D)

Aufgabenstellung: Entwurf eines Dashboars für mein Studium

Name: Gökhan Sen / Matrikelnummer: IU14140216

Phase 01 – Konzept und Machbarkeitsprüfung

1. Ziel der Phase 01

Ziel der Phase 01 ist es, die grundlegende Konzeption und Umsetzbarkeit eines Studien-Dashboards zu prüfen. Das Dashboard soll den Fortschritt im Studiengang „Angewandte Künstliche Intelligenz“ strukturiert darstellen und zentrale Kennzahlen (KPIs) zur Studiendauer und zur Durchschnittsnote transparent machen.

Im Vordergrund stehen dabei (a) ein konsistentes objektorientiertes Datenmodell, (b) eine nachvollziehbare Tool- und Technologieauswahl sowie (c) der Nachweis, dass die wichtigsten technischen Bausteine (lokale Datenbank, GUI-Grundinteraktion, UML-Modellierung) grundsätzlich funktionieren.

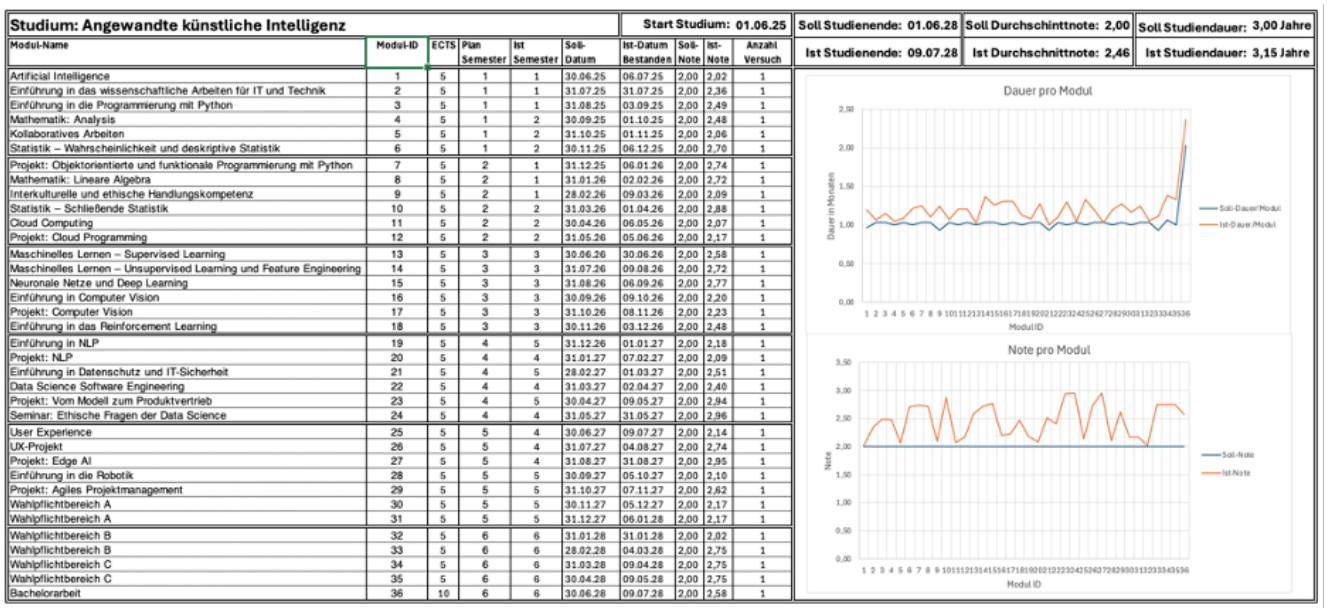
Phase 01 dient damit als belastbare Grundlage für Phase 02 (Implementierung der Kernlogik und Persistenz) und Phase 03 (Vervollständigung, Erweiterungen wie Suchen/Ändern, Validierungen und Abschlussabgabe).

2. Excel-Dashboard als Vorlage

Das Excel-Dashboard wird in Phase 01 ausschließlich als Vorlage und als Dokumentationshilfe genutzt. Es dient dazu, die Zielsetzung (welche Daten werden benötigt?) sowie die gewünschte Darstellung (Tabellenübersicht und Diagramme für Abweichungen) greifbar zu machen. Die spätere Umsetzung erfolgt vollständig in Python; Excel ist nicht Bestandteil der Zielanwendung.

Die Tabelle bildet pro Modul u. a. Modul-ID, ECTS, Plan-Semester sowie Soll-/Ist-Werte (Datum, Note, Versuche) ab. Die Diagramme visualisieren die Abweichungen zwischen Soll- und Ist-Dauer bzw. Soll- und Ist-Note. Damit ist früh erkennbar, bei welchen Modulen die größten Abweichungen auftreten.

Wichtig: Die Excel-Vorlage ist bewusst pragmatisch. Einige Kennzahlen (z. B. Durchschnittsnoten oder Studiendauer) werden in der späteren Anwendung nicht als manuelle Eingabe geführt, sondern aus den gespeicherten Daten abgeleitet. Die Excel-Datei dient in Phase 01 daher primär der Kommunikation des Zielbildes und nicht der finalen Datenhaltung.



Studium: Angewandte künstliche Intelligenz						Start Studium: 01.06.25			
Modul-Name	Modul-ID	ECTS	Plan Semester	Ist Semester	Soll-Datum	Ist-Datum Bestanden	Soll-Note	Ist-Note	Anzahl Versuch
Artificial Intelligence	1	5	1	1	30.06.25	06.07.25	2,00	2,02	1
Einführung in das wissenschaftliche Arbeiten für IT und Technik	2	5	1	1	31.07.25	31.07.25	2,00	2,36	1
Einführung in die Programmierung mit Python	3	5	1	1	31.08.25	03.09.25	2,00	2,49	1
Mathematik: Analysis	4	5	1	2	30.09.25	01.10.25	2,00	2,48	1
Kollaboratives Arbeiten	5	5	1	2	31.10.25	01.11.25	2,00	2,06	1
Statistik – Wahrscheinlichkeit und deskriptive Statistik	6	5	1	2	30.11.25	06.12.25	2,00	2,70	1
Projekt: Objektorientierte und funktionale Programmierung mit Python	7	5	2	1	31.12.25	06.01.26	2,00	2,74	1
Mathematik: Lineare Algebra	8	5	2	1	31.01.26	02.02.26	2,00	2,72	1
Interkulturelle und ethische Handlungskompetenz	9	5	2	1	28.02.26	09.03.26	2,00	2,09	1
Statistik – Schließende Statistik	10	5	2	2	31.03.26	01.04.26	2,00	2,88	1
Cloud Computing	11	5	2	2	30.04.26	06.05.26	2,00	2,07	1
Projekt: Cloud Programming	12	5	2	2	31.05.26	05.06.26	2,00	2,17	1
Maschinelles Lernen – Supervised Learning	13	5	3	3	30.06.26	30.06.26	2,00	2,58	1
Maschinelles Lernen – Unsupervised Learning und Feature Engineering	14	5	3	3	31.07.26	09.08.26	2,00	2,72	1
Neuronale Netze und Deep Learning	15	5	3	3	31.08.26	06.09.26	2,00	2,77	1
Einführung in Computer Vision	16	5	3	3	30.09.26	09.10.26	2,00	2,20	1
Projekt: Computer Vision	17	5	3	3	31.10.26	08.11.26	2,00	2,23	1
Einführung in das Reinforcement Learning	18	5	3	3	30.11.26	03.12.26	2,00	2,48	1
Einführung in NLP	19	5	4	5	31.12.26	01.01.27	2,00	2,18	1
Projekt: NLP	20	5	4	4	31.01.27	07.02.27	2,00	2,09	1
Einführung in Datenschutz und IT-Sicherheit	21	5	4	5	28.02.27	01.03.27	2,00	2,51	1
Data Science Software Engineering	22	5	4	4	31.03.27	02.04.27	2,00	2,40	1
Projekt: Vom Modell zum Produktvertrieb	23	5	4	5	30.04.27	09.05.27	2,00	2,94	1
Seminar: Ethische Fragen der Data Science	24	5	4	4	31.05.27	31.05.27	2,00	2,96	1
User Experience	25	5	5	4	30.06.27	09.07.27	2,00	2,14	1
UX-Projekt	26	5	5	4	31.07.27	04.08.27	2,00	2,74	1
Projekt: Edge AI	27	5	5	4	31.08.27	31.08.27	2,00	2,95	1
Einführung in die Robotik	28	5	5	5	30.09.27	05.10.27	2,00	2,10	1
Projekt: Agiles Projektmanagement	29	5	5	5	31.10.27	07.11.27	2,00	2,62	1
Wahlpflichtbereich A	30	5	5	5	30.11.27	05.12.27	2,00	2,17	1
Wahlpflichtbereich A	31	5	5	5	31.12.27	06.01.28	2,00	2,17	1
Wahlpflichtbereich B	32	5	6	6	31.01.28	31.01.28	2,00	2,02	1
Wahlpflichtbereich B	33	5	6	6	28.02.28	04.03.28	2,00	2,75	1
Wahlpflichtbereich C	34	5	6	6	31.03.28	09.04.28	2,00	2,75	1
Wahlpflichtbereich C	35	5	6	6	30.04.28	09.05.28	2,00	2,75	1
Bachelorarbeit	36	10	6	6	30.06.28	09.07.28	2,00	2,58	1

Soll Studienende: 01.06.28	Soll Durchschinttnote: 2,00	Soll Studiendauer: 3,00 Jahre
Ist Studienende: 09.07.28	Ist Durchschnittnote: 2,46	Ist Studiendauer: 3,15 Jahre

Abbildung 1: Excel-Übersicht (Vorlage für Phase 01)

3. Datenmodell (UML)

Das UML-Klassendiagramm bildet die fachliche Struktur des Dashboards ab und trennt bewusst zwischen stabilen Stammdaten (Modulkatalog) und den tatsächlichen Studienergebnissen (Belegungen). Diese Trennung ist wichtig, da Module in der Praxis flexibel belegt werden können: Ein Modul kann didaktisch für ein bestimmtes Plansemester vorgesehen sein, aber real in einem anderen Semester absolviert werden.

Das Modell besteht aus fünf zentralen Klassen: Student, Studiengang, Semester, Modul und ModulBelegung. Student enthält die personenbezogenen Stammdaten. Studiengang beschreibt die Ziele und enthält berechnete KPIs. Semester strukturiert den Studienverlauf, Modul bildet den Modulkatalog ab, und ModulBelegung hält die Ist-Daten zu tatsächlich absolvierten Leistungen.

- Student enthält die personenbezogenen Stammdaten (Vorname, Nachname, Matrikelnummer sowie optional Geburtsdatum und Adresse) als Eigentümer des Dashboards.
- Studiengang enthält die Soll-Ziele (z. B. sollStudienende, sollStudiendauerJahre, sollDurchschnittsnote) und stellt abgeleitete KPIs als Operationen (/IstStudienende(), /IstStudiendauerJahre(), /IstDurchschnittsnote(), /NoetigeRestDurchschnittsnote()) bereit.
- Modul ist der Modulkatalog mit eindeutiger modulId, titel und ects sowie Planinformationen (planSemesterNr, sollBestandenAm).
- ModulBelegung verknüpft ein Modul mit dem tatsächlich absolvierten Semester-Kontext (istBelegung) und speichert nur Ist-Werte: istBestandenAm, istNote, anzahlVersuche.
- Constraints (Empfehlungen): Semester.nummer ist eindeutig je Studiengang; Modul.modulId ist eindeutig im Modulkatalog.

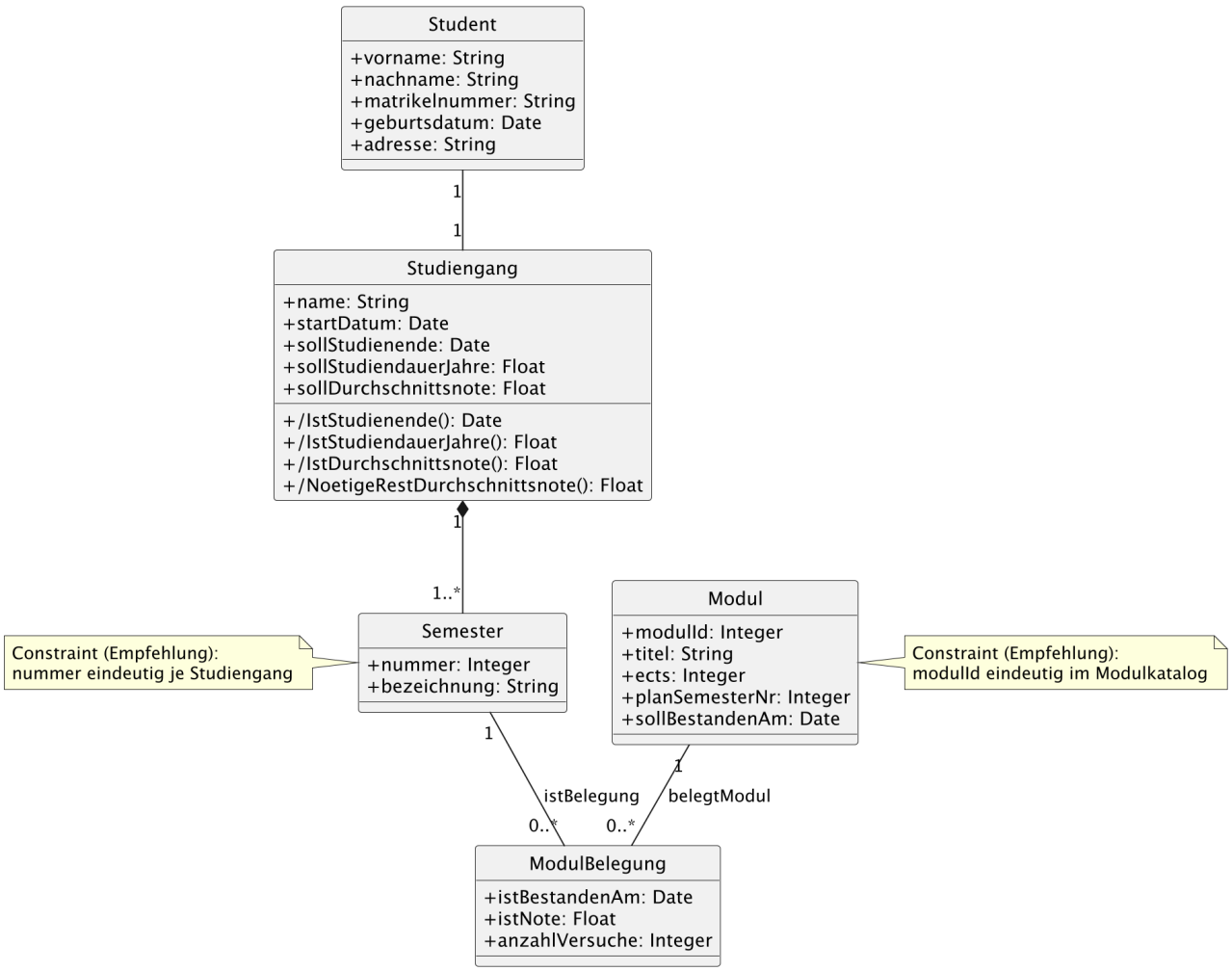


Abbildung 2: UML-Klassendiagramm

4. Machbarkeitsprüfung (Tests)

In Phase 01 wird die Machbarkeit über kleine Testprogramme nachgewiesen. Dabei geht es nicht um eine vollständige Anwendung, sondern um die Bestätigung, dass die gewählten Werkzeuge und die geplante Architektur grundsätzlich funktionieren: lokale SQL-Datenhaltung, minimale GUI-Interaktion sowie UML-Erstellung und Versionierung.

4.1 Entwicklungsumgebung und Bibliotheken

Als Entwicklungsumgebung wird PyCharm genutzt, da es für objektorientierte Python-Projekte eine sehr gute Unterstützung bietet (Projektstruktur, Debugging, virtuelle Umgebungen, Git-Integration). Für Phase 01 werden bevorzugt Standardbibliotheken eingesetzt (z. B. sqlite3, tkinter), um Installationsaufwand und Abhängigkeiten gering zu halten. Für UML wird PlantUML in PyCharm über die IDE-Integration verwendet.

4.2 Lokale Datenspeicherung

Die Datenhaltung soll lokal und ohne Serverabhängigkeit erfolgen. Dafür wird eine SQLite-Datenbank als Datei genutzt. SQLite ist leichtgewichtig, benötigt keine separate Installation eines Datenbankservers und unterstützt dennoch relationale Modellierung (Tabellen, Schlüssel, Joins). Das Schema orientiert sich am UML-Modell: Modulkatalog (Modul) und Ergebnisdaten (ModulBelegung) sind getrennt, wodurch Plan- und Ist-Daten konsistent verwaltet werden können.

4.3 Quellcodeverwaltung

Der Quellcode wird mit Git versioniert und in GitHub verwaltet. Damit sind Änderungen nachvollziehbar (Commits), Zwischenstände sind abgesichert und die Entwicklung bleibt reproduzierbar. Für die Abgabe ist dies hilfreich, da sowohl Modell (PlantUML-Datei) als auch Testprogramme und Dokumentation als konsistenter Projektstand bereitgestellt werden können.

4.4 Benutzerinteraktion

Für die spätere Anwendung ist eine GUI mit Eingabefeldern vorgesehen, um Modulbelegungen komfortabel zu erfassen und auszuwerten. In Phase 01 genügt eine einfache Test-GUI, die exemplarisch zeigt, dass Eingaben entgegengenommen, validiert und in der lokalen Datenbank gespeichert werden können. Die Auswahl einer GUI in Phase 01 reduziert Projektrisiken, da typische Integrationspunkte (Eingabemaske, Button-Events, Datenbankzugriff) frühzeitig getestet werden.

Dateneingabe und CRUD-Umfang (Phase 1)

Eingabeweg (User Interaction)

In Phase 1 erfolgt die Dateneingabe testweise über eine GUI-Eingabemaske. Die GUI dient als Machbarkeitsnachweis für eine spätere vollständige Anwendung mit Benutzeroberfläche. Ergänzend werden Konsolentests genutzt, um die Initialisierung und Überprüfung (z. B. Datenbankdatei, Tabellenanlage) reproduzierbar auszuführen und zu dokumentieren.

Welche Daten können eingegeben werden?

Die Eingabe orientiert sich am UML-Modell und unterscheidet zwischen Modulkatalog (Stammdaten) und ModulBelegung (Ist-Daten). In Phase 1 werden insbesondere folgende Daten testweise erfasst:

• Modul (Stammdaten): modulId, titel, ects (planSemesterNr und sollBestandenAm sind konzeptionell Teil des Moduls).

• ModulBelegung (Ist-Daten): istBestandenAm, istNote, anzahlVersuche.

Damit ist die Grundlage geschaffen, die abgeleiteten KPI-Operationen /IstStudienende(), /IstStudiendauerJahre() und /IstDurchschnittsnote() aus den gespeicherten Belegungen abzuleiten.

CRUD: Was ist in Phase 1 möglich?

Für Phase 1 ist der Umfang bewusst reduziert, da Fokus auf der grundsätzlichen Machbarkeit liegt:

• Create (Anlegen): vorhanden – ModulBelegungen können gespeichert werden; Module werden über modulId eindeutig referenziert.

• Read (Lesen): vorhanden – gespeicherte Einträge können angezeigt/abgerufen werden (z. B. „letzte Einträge“).

• Update (Ändern): nicht Bestandteil von Phase 1 – wird in Phase 2/3 ergänzt (z. B. Auswahl über belegung_id und UPDATE-Statement).

• Delete (Löschen): nicht Bestandteil von Phase 1 – wird in Phase 2/3 ergänzt (z. B. DELETE-Statement mit Bestätigungsdialog).

Die Abgrenzung verhindert unnötige Komplexität in der frühen Phase und hält den Machbarkeitsnachweis fokussiert.

Ausblick (Phase 2/3)

In Phase 2/3 ist geplant, die GUI um Suche/Filter (z. B. nach modulId oder belegung_id), Auswahl eines Datensatzes sowie Änderungs- und Löschfunktionen zu erweitern. Damit werden nachträgliche Korrekturen und eine vollständige CRUD-Unterstützung ermöglicht.

4.5 Begründung der Tool-/Versionsauswahl

Die Auswahl der Tools orientiert sich an Stabilität, Verfügbarkeit und geringem Einrichtungsaufwand. PyCharm ist als etablierte IDE für Python-Projekte geeignet und unterstützt GitHub sowie virtuelle Umgebungen. SQLite und tkinter sind Teil der Python-Standardbibliothek und damit auf den typischen Zielsystemen ohne zusätzliche Installation nutzbar. PlantUML ermöglicht, das UML-Modell als Textdatei zu versionieren, wodurch Änderungen am Modell transparent nachvollziehbar werden.

Bei der Versionswahl gilt in Phase 01 das Prinzip: „Getestet = begründet“. Entscheidend ist, dass die verwendete Python-Version und die eingesetzten Bibliotheken im lokalen Setup erfolgreich laufen und die geforderten Kernfunktionen abdecken. Zusätzlich wird darauf geachtet, möglichst wenige externe Abhängigkeiten einzuführen, um spätere Setup-Probleme (z. B. auf anderen Rechnern oder bei der Abgabe) zu vermeiden.

4.6 Ausführung der Tests

Die Machbarkeit wurde anhand von Beispieldaten und Testfunktionen geprüft. Dabei wurden die grundlegenden Bausteine erfolgreich ausgeführt: (1) Python/Tooling-Check, (2) Erzeugen und Nutzen einer lokalen SQLite-Datenbankdatei, (3) einfache GUI-Eingabe und Persistenz, (4) Erstellung/Preview des UML-Klassendiagramms in PlantUML. Auf Basis dieser Tests wird die Umsetzung des Dashboards in den Folgephasen als realistisch und technisch umsetzbar eingestuft.