# Abstract: Entwicklung eines Objektorientierten Studien-Dashboards

## 1. Einleitung und Zielsetzung

Dieses Portfolio-Projekt dokumentiert die Entwicklung eines CLI-basierten Studien-Dashboards zur Verwaltung des akademischen Fortschritts im Studiengang B.Sc. Cyber Security. Das zentrale Ziel war die praktische Anwendung objektorientierter Programmierkonzepte (OOP) in Python, insbesondere **Kapselung**, **Komposition** und **Aggregation**. Das Dashboard ermöglicht die Verwaltung von Studiengängen, Semestern, Modulen und Prüfungsleistungen und visualisiert den Studienfortschritt (ECTS-Punkte, Notendurchschnitt). Die Anwendung basiert auf einer **3-Schichten-Architektur** und verarbeitet reale Studiendaten aus dem Cyber Security-Programm.

## 2. Technische Umsetzung

Die Entwicklung erfolgte iterativ in drei Phasen: Konzeption (UMLDiagramme), Reflexion (theoretische Auseinandersetzung mit OOP) und Implementierung.

**Architektur:** Die Anwendung folgt einer 3-Schichten-Architektur: Domänenschicht (Kernlogik und Geschäftsobjekte), Persistenzschicht (Datenspeicherung in JSON/CSV/Pickle) und Präsentationsschicht (CLIbasierte Benutzeroberfläche).

**OOP-Konzepte:** Die Umsetzung der vom Tutor geforderten Prinzipien stand im Mittelpunkt: **Kapselung** durch konsequente Verwendung von @property und @setter-Dekoratoren zur Kontrolle des Attributzugriffs.

**Komposition:** Ein Studiengang besitzt Semester-Objekte, die im Konstruktor erstellt werden.

**Aggregation:** Ein Semester enthält ModulObjekte, die unabhängig existieren und dynamisch hinzugefügt werden.

**Enums:** Verwendung von Enumerationen für Abschluss, ModulStatus und Pruefungsart zur Verbesserung der Typsicherheit. Die Implementierung erfolgte in Python 3.14 unter ausschließlicher Verwendung der Standard Library ohne externe Bibliotheken

## 3. Herausforderungen und Lernerfolge

Die größte Herausforderung bestand in der korrekten Unterscheidung und Implementierung von **Komposition und Aggregation**. Die praktische Umsetzung im Code erforderte eine tiefgehende Auseinandersetzung mit Objektlebenszyklen. Die Entscheidung, Semester im Konstruktor der Studiengang-Klasse zu erstellen (Komposition), während Module von außen an ein Semester übergeben werden (Aggregation), hat mein Verständnis für Objektbeziehungen nachhaltig geschärft.

Ein weiterer bedeutender Lernerfolg war die disziplinierte Einhaltung der **3-Schichten-Architektur**. Die strikte Trennung von Logik, Datenhaltung und Darstellung ermöglichte unabhängige Weiterentwicklung. Die Verwendung von @property-Dekoratoren statt direkter Attributzugriffe führte zu einem saubereren, pythonischeren Code. Die Verarbeitung realer Studiendaten (21 Module aus dem Cyber SecurityProgramm) und die

Implementierung präziser Berechnungen (ECTSgewichteter Notendurchschnitt) erforderten sorgfältige Datenmodellierung. Die Entwicklung von Test-Skripten zur Demonstration der OOP-Konzepte half, die Korrektheit der Implementierung zu validieren.

## 4. Reflexion und Ausblick

Dieses Projekt war eine wertvolle Brücke zwischen theoretischem Wissen und praktischer Anwendung. Besonders stolz bin ich auf die vollständige Umsetzung des Tutoren-Feedbacks und die professionelle Code-Qualität. Die saubere 3-Schichten-Architektur zeigt, dass ich die theoretischen Konzepte verstanden und praktisch anwenden kann. Das Dashboard ist nicht nur ein akademisches Projekt, sondern ein nützliches Werkzeug für meinen weiteren Studienverlauf. Zukünftige Erweiterungen sind denkbar: Eine grafische Benutzeroberfläche (Tkinter/ PyQt), Datenbankanbindung (SQLite/PostgreSQL), Prognosen für den Abschlussdurchschnitt oder eine Flask-basierte Webanwendung. Die modulare Architektur erleichtert alle diese Erweiterungen. Zusammenfassend hat dieses Projekt meine Fähigkeiten in der objektorientierten Softwareentwicklung maßgeblich erweitert und eine solide Grundlage für zukünftige, komplexere Aufgaben geschaffen.