HAW Kiel
Hochschule für Angewandte Wissenschaften Kiel
Kiel University of Applied Sciences

ASA - Angewandte Softwarearchitektur
ASA - Applied Softwarearchitecture

Allgemeine Informationen	
Modulkürzel oder Nummer	ASA
Eindeutige Bezeichnung	AngewSWArch-01-BA-M
Modulverantwortlich(e)	Prof. Dr. Prochnow, Steffen (steffen.prochnow@haw-kiel.de)
Lehrperson(en)	M.Sc. Petersen, Eike (eike.petersen@haw-kiel.de) Prof. Dr. Prochnow, Steffen (steffen.prochnow@haw-kiel.de)
Wird angeboten zum	Sommersemester 2026
Moduldauer	1 Fachsemester
Angebotsfrequenz	Regelmäßig
Angebotsturnus	In der Regel im Sommersemester
Lehrsprache	Deutsch
Empfohlen für internationale Studierende	Nein
Ist als Wahlmodul auch für andere Studiengänge freigegeben (ggf. Interdisziplinäres Modulangebot - IDL)	Ja

Studiengänge und Art des Moduls (gemäß Prüfungsordnung)
Studiengang: B.Eng. - E - Elektrotechnik (PO 2017, V3) Modulart: Wahlmodul Fachsemester: 4, 6
Studiengang: B.Eng. - E - Elektrotechnik (PO 2023, V4) Modulart: Wahlmodul Fachsemester: 4, 6
Studiengang: B.Eng. - Ming - Medieningenieur/-in (PO 2018, V1 + PO 2021, V2) Modulart: Wahlmodul Fachsemester: 4, 6
Studiengang: B.Sc. - INF - Informatik (PO 2021,V1) Schwerpunkt: Anwendungsentwicklung Modulart: Verpfl. Wahlmodul, PVO §3 Fachsemester: 4, 6
Studiengang: B.Sc. - INF - Informatik (PO 2021,V1) Modulart: Wahlmodul Fachsemester: 4, 6

Kompetenzen / Lernergebnisse
Kompetenzbereiche: Wissen und Verstehen; Einsatz, Anwendung und Erzeugung von Wissen; Kommunikation und Kooperation; Wissenschaftliches Selbstverständnis/Professionalität.

Lernziele: Anhand von bestehenden und selbst entwickelten Architekturen werden Aspekte der Softwarearchitektur und deren Nutzen erarbeitet. Durch das angebotene Labor, werden die Techniken anhand von Beispielen vertieft und deren Anwendung geübt.

Fachkompetenzen:
Die Studierenden üben die Anwendung der Grundlagen von Softwarearchitektur anhand von neuen und bestehenden Softwareprojekten. Dabei wird der Fokus auf die praxisbezogene Arbeit mit bekannten Entwurfsmustern gelegt.

Sozialkompetenzen:
Die Studierenden müssen im Team Teilaufgaben erarbeiten und dabei ihre eigenen Positionen gegenüber anderen Teammitgliedern vertreten.

Selbstkompetenz:
Die Studierenden können Ziele für den Arbeitsprozess definieren und ihn so nachhaltig gestalten. Sie erkennen ihre eigenen Stärken und Schwächen in der Teamarbeit. Sie lernen selbstständig Teile der Vorlesung zu erarbeiten.

Systemische Kompetenz:
Die Studierenden können Architekturen von Softwaresystemen entwerfen, analysieren und bewerten.

Angaben zum Inhalt

Lehrinhalte	Clean Code: Was macht guten Code aus? UML: Kurze Zusammenfassung von UML. Design Patterns: Entwurfsmuster und Ihre Verwendung in OpenSource Software. Architekturmuster: Einstieg in die gebräuchlichsten Architekturmuster. (Daten-)Sicherheit: Speichern von Passwörtern und Authentifizieren von Applikationen. Netzwerkprotokolle: Daten in einer sinnvollen Weise zwischen Applikationen austauschen. Kosten / Nutzen: Entscheidungen für eine Architektur aufgrund von verschiedenen Faktoren treffen. Effizienz: Code auf seine Effizienz überprüfen und optimieren. Natur --> Architektur: Aus bestehenden Dingen in der Welt Architekturen ableiten.
Literatur	- E. Gamma, R. Helm, R. E. Johnson, J. Vlissides: Design Patterns. Elements of Reusable Object-Oriented Software (ISBN-13: 978-0201633610) - Robert C. Martin: Clean Code: A Handbook of Agile Software Craftsmanship (ISBN-13 978-0132350884)

Lehrformen der Lehrveranstaltungen

Lehrform	**SWS**
Labor	2
Lehrvortrag	2

Arbeitsaufwand

Anzahl der SWS	4 SWS
Leistungspunkte	5,00 Leistungspunkte
Präsenzzeit	48 Stunden
Selbststudium	102 Stunden

Modulprüfungsleistung	
Voraussetzung für die Teilnahme an der Prüfung gemäß PO	Keine
ASA - Portfolioprüfung	Prüfungsform: Portfolioprüfung Gewichtung: 100% wird angerechnet gem. § 11 Absatz 2 PVO: Nein Benotet: Ja Anmerkung: Die Portfolioprüfung setzt sich zusammen aus: - Spezifikation einer Software mit Architekturmuster - Implementierung einer Software auf der Basis von Architekturmuster

Sonstiges	
Empfohlene Voraussetzungen	Einführung in die Objektorientierte Programmierung (OOP), Programmieren (PRG)