W3WI_110 - Entwicklung verteilter Systeme

Dozent: Prof. Dr. Michael Eichberg
Kontakt: michael.eichberg@dhbw.de, Raum 149B
Version: 23SEB (SoSe 2025 - 1. Rev.)

Ausgewählte Inhalte gem. MHB - Web-Programmierung

Kerninhalte

- HTML, CSS, JavaScript als clientseitige Web-Technologien und aktuelle APIs (z.B. HTML5 und verwandte Technologien)
- Übertragungsprotokolle und APIs zwischen Client und Server (z.B. HTTP, HTTPS, WebSockets, XMLHttpRequest, Fetch API)
- Kommunikation zwischen einzelnen Komponenten Web-basierter Anwendungen
- Optimierung von Webseiten für verschiedene Zielsysteme

Zusatzinhalte

- Vertiefung von Frameworks
- Dynamische serverseitige Erzeugung von Webseiten

Ausgewählte Inhalte gem. MHB - Verteilte Systeme

- Terminologie, Konzepte, Architekturen, Anforderungsprofile und Architekturmodelle für verteilte Systeme
- Entwurfs- und Implementierungsansätze
- Vergleich unterschiedlicher Middleware-Konzepte
- Synchrone und asynchrone Kommunikation, entfernter Methodenaufruf
- Asynchrone Kommunikation und Messaging-Systeme
- Sicherheitsaspekte in verteilten Systemen

Prüfungsleistung - Portfolio

- das Modul hat 55 VL
- Verteilte Systeme hat 22VL, Web-Programmierung hat 33 VL

Zwei Bestandteile:

1. (**30** von 120 Punkten) - Vorträge (Hinweise zur Präsentationen: https://delors.github.io/allg-vortraege/folien.de.rst.html)

   Die Präsentationen müssen am Abend vorher im Moodle hochgeladen werden!
2. (**90** von 120 Punkten) - Projekt bzw. Programmieraufgabe in Teams von 4 Personen
   1. Projekt (Funktionsumfang, Code, Dokumentation, etc.)
   2. Abschlusspräsentation / Vorführung
   3. Code Reviews

Gruppenarbeit = Gruppennote

Das Projekt ist als Gruppenarbeit ausgelegt und alle Gruppenmitglieder erhalten für den Projekteil die gleiche Note.

Sollte eine individuelle Benoten gewünscht sein, dann ist dies **vor Abgabe des Projekts** explizit zu kommunizieren, damit das weitere Vorgehen besprochen werden kann.

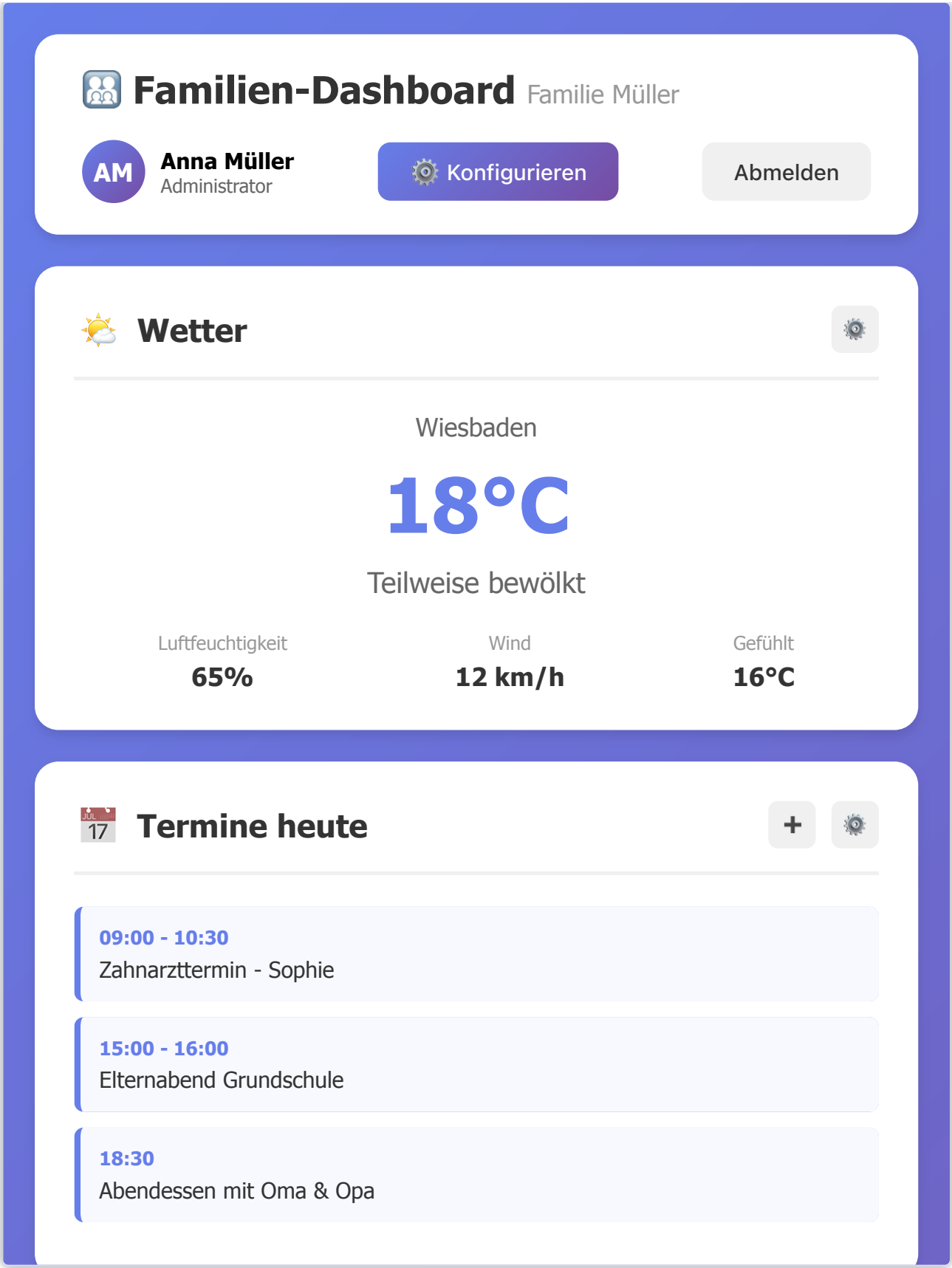
Projekt/Programmieraufgabe

Projektübersicht

Entwickeln Sie ein webbasiertes, responsives Familien-Dashboard, das als zentrale Informationsplattform für Familienmitglieder dient. Die Anwendung soll verschiedene konfigurierbare Widgets bereitstellen und durch ein Rollenkonzept unterschiedliche Zugriffsrechte ermöglichen - ggf. auf Widget-Level.

Anwendungsbeispiele

- Sie möchten - z. B. in der Küche - ein Dashboard auf einem Tablet anzeigen lassen, das den Familienkalender, die Schulpläne der Kinder, den Wochenplan für das Au Pair, das Wetter und eine To-Do-Liste anzeigt.
- Sie möchten, dass alle Familienmitglieder gegenseitig lesenden Zugriff auf den/die Kalender der Familienmitglieder haben; die Kalender werden ggf. in externen Diensten (Google Calendar, Apple Calendar etc.) geführt.
- Sie möchten den Schulplan der Kinder als Widget auf dem Dashboard anzeigen lassen
- Ein Au-Pair soll lesenden Zugriff auf ihren/seinen Wochenplan haben, aber keine administrativen Rechte besitzen.
- Die To-Do-Liste kann von allen Familienmitgliedern bearbeitet werden.
- ...



Beispiel eines Familien-Dashboards erzeugt mit Claude Code.

Anforderungen

Funktionale Anforderungen

Widget-System

- Implementierung eines modularen Widget-Systems mit mindestens 3 der folgenden Widgets:
  - Stundenplan (für Kinder und Au-Pairs) (MUSS)
  - Gemeinsamer Terminkalender (ggf. als Integration externer Kalenderdienste)
  - Wetteranzeige (mit Standortauswahl)
  - To-Do-Liste
  - Notizen/Pinnwand
- Widgets sollen hinzugefügt, entfernt und auf dem Dashboard positioniert werden können
- Jedes Widget muss individuell konfigurierbar sein (z.B. Datenquelle, Darstellungsoptionen)

Rollenkonzept

Familien-Administrator-Rolle:

Volle Konfigurations- und Verwaltungsrechte

- Widgets hinzufügen/entfernen/konfigurieren
- Benutzer verwalten und Rollen zuweisen

  (D. h. Registrierung und Authentifizierung von Nutzern durchführen und Nutzer zu Familiengruppen zuweisen.)
- Widget-Berechtigungen festlegen

Nutzer-Rolle:

Eingeschränkte Rechte

- Dashboard ansehen und sein persönliches Layout anpassen
- Zugriff nur auf freigegebene Widgets
- Interaktion mit Widget-Inhalten (z.B. Termine einsehen, nicht aber Dashboard umgestalten)

System-Administrator-Rolle:

Verwaltung der Anwendung als solches

- Benutzer- und Familiengruppenverwaltung
- Systemweite Einstellungen und Wartung

Nicht-funktionale Anforderungen

- Responsive Design
  - Optimierte Darstellung für Desktop, Tablet und Smartphone
  - Touch-optimierte Bedienung für mobile Endgeräte
- Verteilte Architektur
  - Klare Trennung von Frontend und Backend
  - RESTful API für die Kommunikation
  - Zustandsverwaltung (State Management) im Frontend
  - Datenpersistenz im Backend

- Erweiterbarkeit
  - Modulare Architektur ermöglicht einfaches Hinzufügen neuer Widgets
  - Klar definierte Schnittstellen zwischen Komponenten
  - Plugin-Architektur für Widget-Entwicklung wäre wünschenswert
- Technische Anforderungen

  Entwickeln Sie eine verteilte Anwendung mit folgender Architektur:
  - Frontend als Single Page Application (SPA) ggf. unter Einsatz von responsive UI-Frameworks (z.B. Bootstrap, Tailwind CSS)
  - Backend mit RESTful API; der Technologiestack ist frei wählbar (z.B. Node.js/Express, Python/Flask, Java/Spring Boot)
  - Datenbank zur Persistierung (z.B. PostgreSQL, MongoDB, MySQL)
- Architektur-Dokumentation
  - Erstellen Sie Architekturdiagramme (z.B. mit C4-Modell)
  - Dokumentieren Sie Design-Entscheidungen
  - Begründen Sie die Wahl von Technologien und Architekturmustern
- Datenintegration
  - Integration mindestens einer externen API (z.B. Wetter-API)
  - Synchronisation von Kalenderdaten
  - Optional: Import/Export von Daten

◆ Bemerkung

Einsatz von KI-Tools

Nutzung von KI-Assistenten (z.B. GitHub Copilot, ChatGPT, Claude Code oder OpenCode) zur Code-Generierung ist erlaubt.

KI kann zur Unterstützung der folgenden Aufgaben eingesetzt werden:

- Boilerplate-Code generieren
- Code-Optimierung und Refactoring
- Debugging und Fehleranalyse
- Erstellung von Tests
- Dokumentation

Pflichten bei KI-Nutzung

- Jedes Teammitglied muss jeden Teil der Anwendung erklären können
- Die Architektur und Design-Entscheidungen müssen vom Team begründet werden
- Der Technologiestack muss vom Team begründet werden
- Im Projektbericht: Dokumentation, wo und wie, welche KI eingesetzt wurde
- Reflexion über Vor- und Nachteile des KI-Einsatzes im Projekt

Bewertungskriterien für das Projekt

Kategorie	**max. 90 Punkte**
Code Reviews	max. 10
Abschlusspräsentation	max. 05
Vorführung	max. 10
Funktionsumfang	max. 15
Dokumentation (Entwickler und Benutzer)	max. 05
Qualität des Codes und der Tests (HTML, CSS und JavaScript)[1]	max. 40
Qualität des Buildprozesses[2]	max. 05

Es ist ein Dokument einzureichen aus dem hervorgeht:

1. welche KI Tools wofür eingesetzt wurden. (*Fehlanzeige erforderlich!*)
2. wer an welchem Teil mitgewirkt hat. (*Ohne dieses Dokument erfolgt keine Bewertung.*)

[1] Es ist neben dem Code auch ein kurzes Video 10 bis max. 15 Minuten einzureichen, dass in die Struktur und die Codebasis einführt. Dieses Video geht in die Benotung ein! Bitte nur im Notfall über Moodle bereitstellen.

[2] Werden Tests ausgeführt und wird am Ende ein Container gebaut?

Ablauf - W3WI-110 - Entwicklung verteilter Systeme 23SEB

- Scheduled: 16. May 2025 at 13:15 to 17:30, CEST
- Scheduled: 19. May 2025 at 13:15 to 17:30, CEST
- Scheduled: 22. May 2025 at 13:15 to 17:30, CEST
- Scheduled: 6. Jun 2025 at 13:15 to 17:30, CEST

  Kurzpräsentation der Projekte (kein Code - Powerpoint ist ausreichend; 5 Minuten pro Team). Kein unmittelbare Bewertung - dient "lediglich" zur Steuerung.
- Scheduled: 13. Jun 2025 at 13:15 to 17:30, CEST
- Scheduled: 16. Jun 2025 at 13:15 to 17:30, CEST
  - **Grundlagen der Virtualisierung** (Terminologie: z.B. Bare Metal Virtualisierung, Hypervisor Level..; Sicherheitsmodelle ggf. von CPU an.)- 2 Stud.
  - **Virtualization Platforms** (Proxmox und Openstack) - 2 Stud.
  - **Container Technologies** (Docker, Firecracker, Linux Containers (LXC)) - 2 Stud.
  - **Container Orchestrators** (Kubernetes, Docker Swarm) - 2 Stud.
- Scheduled: 24. Jun 2025 at 13:15 to 17:30, CEST
  - **Web- and Distributed Application Testing** (Diskussion und Präsentation von Werkzeugen für das Frontend- und Backend Testing) - 4 Stud.
    (Jeder Studierende soll sich sein eigenes Thema suchen! D. h. es werden danach im Prinzip vier Einzelvorträge gehalten.)
  - **gRPC** 🔗 und **gRPC-web** 🔗 - 2 Stud.
  - **Apache Thrift** 🔗 - 2 Stud.
- Scheduled: 30. Jun 2025 at 13:15 to 17:30, CEST
  - **Grundlagen von outdoor/indoor Positionierungssystemen** (GPS) - 1 Stud.
  - **Distributed Hash Tables** - 1 Stud.
  - **Paxos** 🔗 - 2 Stud.
  - **Raft Consensus Algorithm** 🔗 - 2 Stud.
  - **Gossip Protokoll** 🔗 - 2 Stud.
- Scheduled: 4. Jul 2025 at 13:15 to 17:30, CEST

  Gegenseitige Code Reviews und Präsentation der Ergebnisse der Code-Reviews.

  (Jedes Team führt ein Review durch (45 Minuten) und wird auch einmal reviewed. Danach erstellen alle Teams einen Bericht über das Projekt, dass sie reviewed haben. Dafür stehen ca. 45 Minuten zur Verfügung. Die Berichte werden danach präsentiert (ca. 10 Minuten). Die Präsentation und die Berichte werden als Teil der Gruppenleistung bewertet.)
- Scheduled: 7. Jul 2025 at 13:15 to 17:30, CEST

  Gruppenindividuelle Betreuung bei Fragen und Problemen bzgl. des Projekts.
- Scheduled: 11. Jul 2025 at 13:15 to 17:30, CEST

  Abschlusspräsentationen (Vorstellung des Tools und Vorstellung wie die Komponenten genutzt werden kann - d. h. Code zeige) und Vorführung der Projekte.

  Die Projektabgabe ist am 9.7.2025 um 23:59 Uhr.

Code Reviews - 4.7.2025

1. Durchführung eines Code Reviews (Frontend, Backend, Buildscripte, Projektstruktur, Dokumentation,...) eines anderen Projektes (2 * 45 Minuten)

   Die Code Reviews erfolgen in zwei Runden, damit jede Gruppe ein Review bekommmt; die Gruppen sollten sich aufteilen, damit alle Teile reviewt werden!

3. Erstellung eines Reports, der konstruktive Vorschläge enthält (30-45 Minuten).
4. Präsentation der Reports

- Gruppe 1: Planning Poker
  Nico Wrede, Max Meinel, Okan Sönmez, Johannes Kling
- Gruppe 2: CoCreate
  Ramona Korten, Monika Pjano, Paulina Klaus, Lisa Molter
- Gruppe 3: Chat App
  Jonas Stammer, Felix Erhard, Luca Bäck, Raphael Plett
- Gruppe 4: Chat App
  Iven Stahl, Christian Zweigert, Ibrahim Tikce, Nils Teschke
- Gruppe 5: Tool für Umfragen
  Jonathan Wieder, Mika Jun, Leon Priemer, Sergio Meli
- Gruppe 6: Planning Poker
  Tom Weber, Tarnbir Singh, Jan Müller, Dilmand Sado