**ipso!**

**Meeting Room Reservation Projektdokumentation**

**Von Even, Yannick und Tunahan**

Inhalt

[1. Projektaufgabe 3](#_Toc194588639)

[1.1 Lernziele 3](#_Toc194588640)

[1.2 Aufgabenbeschreibung 3](#_Toc194588641)

[2. Zielsetzung 3](#_Toc194588642)

[3. Projektanforderungen 3](#_Toc194588643)

[3.1. Frontend (Angular/React) 3](#_Toc194588644)

[3.2. Backend (ASP.NET Core WebAPI) 4](#_Toc194588645)

[3.3. Datenbank (MongoDB) 4](#_Toc194588646)

[4. Projektstruktur 4](#_Toc194588647)

[4.1. Yannick 5](#_Toc194588648)

[4.2. Even 5](#_Toc194588649)

[4.3. Tunahan 5](#_Toc194588650)

[4.4. Gemeinsame Aufgaben 5](#_Toc194588651)

[5. Projektplanung und Zeitmanagement 5](#_Toc194588652)

[5.1 GANTT digramm 6](#_Toc194588653)

[5.2 Frontend-Entwicklung 6](#_Toc194588654)

[5.3 Backend-Entwicklung 6](#_Toc194588655)

[5.4 Datenbankintegration 6](#_Toc194588656)

[5.5 Testing & Qualitätssicherung 7](#_Toc194588657)

[6. Abschluss und Abgabe 7](#_Toc194588658)

# 1. Projektaufgabe

## 1.1 Lernziele

* Erstellung eines Projektplans
* Nutzung der Technologien: Angular/React, ASP.NET Core WebAPI und MongoDB
* Sammeln von Erfahrungen in der Teamarbeit

## 1.2 Aufgabenbeschreibung

Die MeetingPoint AG-Service möchte eine datenbankbasierte Web-Lösung für die Verwaltung und Reservierung interner Besprechungsräume implementieren.

# 2. Zielsetzung

Ziel dieses Projekts ist die Entwicklung einer einfachen Webanwendung, die es Nutzern ermöglicht, Besprechungsräume zu reservieren. Die Anwendung besteht aus:

* **Frontend:** Angular/React
* **Backend:** ASP.NET Core WebAPI
* **Datenbank:** MongoDB
* **Funktionalität:** CRUD-Operationen für Besprechungsräume und Reservierungen

# 3. Projektanforderungen

## 3.1. Frontend (Angular/React)

Das Frontend kommuniziert mit der Backend-API und bietet eine Benutzeroberfläche zur Verwaltung von Besprechungsräumen und Reservierungen.

**Funktionen:**

* Anzeige verfügbarer Besprechungsräume und aktueller Reservierungen
* CRUD-Operationen für Besprechungsräume (Erstellen, Lesen, Aktualisieren, Löschen)
* Listen- oder Kalenderansicht für Reservierungen
* Validierung von Eingaben (z. B. Verhinderung von Überschneidungen bei Reservierungen)
* Responsive UI und interaktive Elemente

## 3.2. Backend (ASP.NET Core WebAPI)

Entwicklung einer RESTful API zur Verwaltung von Besprechungsräumen und Reservierungen.

**Funktionen:**

* CRUD-Endpunkte für Besprechungsräume
* CRUD-Endpunkte für Reservierungen
* Datenvalidierung (z. B. Vermeidung von Terminüberschneidungen)
* Speicherung und Abruf von Daten aus MongoDB
* Logging von Laufzeitfehlern und API-Anfragen

## 3.3. Datenbank (MongoDB)

**Collections:**

* **MeetingRooms:** Enthält Raumname, Kapazität, Ausstattung und Verfügbarkeit
* **Reservations:** Enthält Start- und Endzeit, Raum-ID, Benutzer-ID und Zweck der Reservierung

**Funktionen:**

* Sicherstellen korrekter CRUD-Operationen
* Verhinderung von Überschneidungen bei Reservierungen
* Sicherstellen, dass nur verfügbare Räume gebucht werden können

# 4. Projektstruktur

Das Projekt wird in spezifische Bereiche aufgeteilt, wobei Teammitglieder jeweils bestimmte Verantwortlichkeiten übernehmen:

* **Frontend:** Entwicklung mit Angular/React
* **Backend:** Entwicklung mit ASP.NET Core WebAPI
* **Datenbank:** Design und Integration von MongoDB
* **Testing & Qualitätssicherung**

## 4.1 Yannick

* **Frontend (Angular/React)**:
  + Entwicklung der Benutzeroberfläche.
  + Implementierung von Formularen für Räume und Reservierungen.
  + Integration der API und Validierung von Benutzereingaben.

## 4.2. Even

* **Backend (ASP.NET Core WebAPI)**:
  + Entwicklung der API zum Verwalten der Besprechungsräume und Reservierungen.
  + Implementierung von CRUD-Endpunkten und Datenvalidierung.
  + Anbindung an die MongoDB-Datenbank.

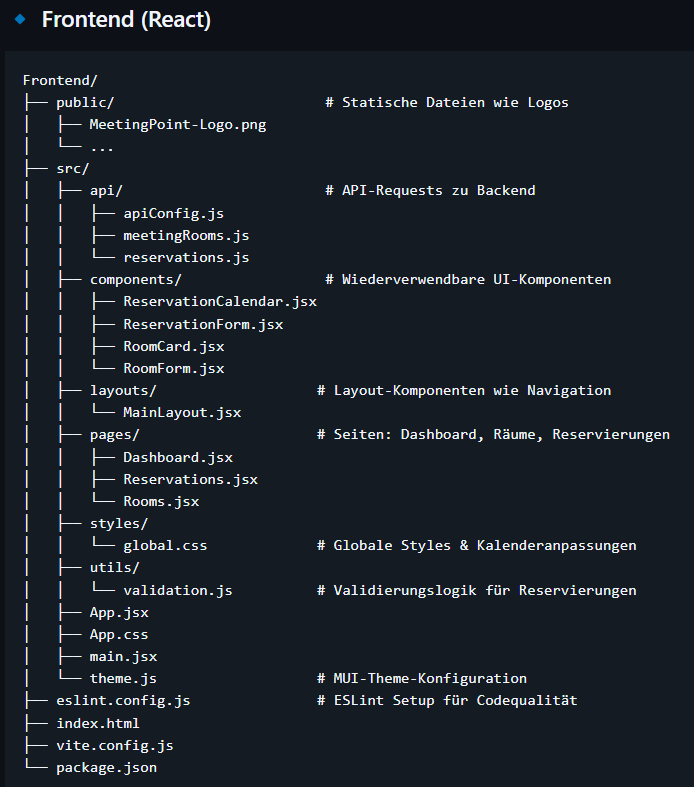
## 4.3. Tunahan

* **Datenbank (MongoDB)**:
  + Design der Datenbank und Modelle für Besprechungsräume und Reservierungen.
  + Sicherstellung, dass nur verfügbare Räume gebucht werden können und keine Überschneidungen entstehen.

## 4.4 Gemeinsame Aufgaben

* **Testing & Qualitätssicherung**:
  + Alle testen gemeinsam die Anwendung. Even testet das Frontend, Yannick das Backend und Tunahan die Datenbank.
  + Durchführung von Unit-Tests und Fehlerbehebung.

## 4.5 Ordner Struktur



Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

# 5. Projektplanung und Zeitmanagement

* Nutzung von Trello für die Zusammenarbeit
* Definition von Meilensteinen (Backend, Datenbank, Frontend, Testing)
* Erstellung eines Gantt-Diagramms zur Visualisierung der Aufgaben

## 5.1 GANTT digramm

A screenshot of a computer

AI-generated content may be incorrect.

## 5.2 Frontend-Entwicklung

* Entwicklung von UI-Komponenten
* Implementierung von Formularen für Räume und Reservierungen
* Integration der API-Schnittstelle

## 5.3 Backend-Entwicklung

* Implementierung von CRUD-Endpunkten
* Validierung der Benutzereingaben (z. B. Verhinderung von Reservierungsüberschneidungen)
* Anbindung an MongoDB

## 5.4 Datenbankintegration

* Design der MongoDB-Modelle
* Implementierung von CRUD-Operationen
* Sicherstellung der Datenintegrität

## 5.5 Testing & Qualitätssicherung

* Unit-Tests für Backend-Endpunkte
* Manuelles Testen des Frontends
* Fehlerbehebung und Optimierung

# 6. Abschluss und Abgabe

Die Anwendung wurde gemäss den definierten Anforderungen entwickelt und erfolgreich getestet. Die finale Abgabe umfasst:

* Den vollständigen Quellcode des Projekts in einem Git-Repository
* Eine ausführliche Dokumentation zur Architektur, den verwendeten Technologien und den wichtigsten Funktionen
* Eine detaillierte Beschreibung der Backend- und Frontend-Funktionalitäten
* Eine Gantt-Diagramm-basierte Projektplanung mit einer Analyse des tatsächlichen Zeitaufwands
* Eine Übersicht über die durchgeführten Tests, gefundene Fehler und deren Behebung