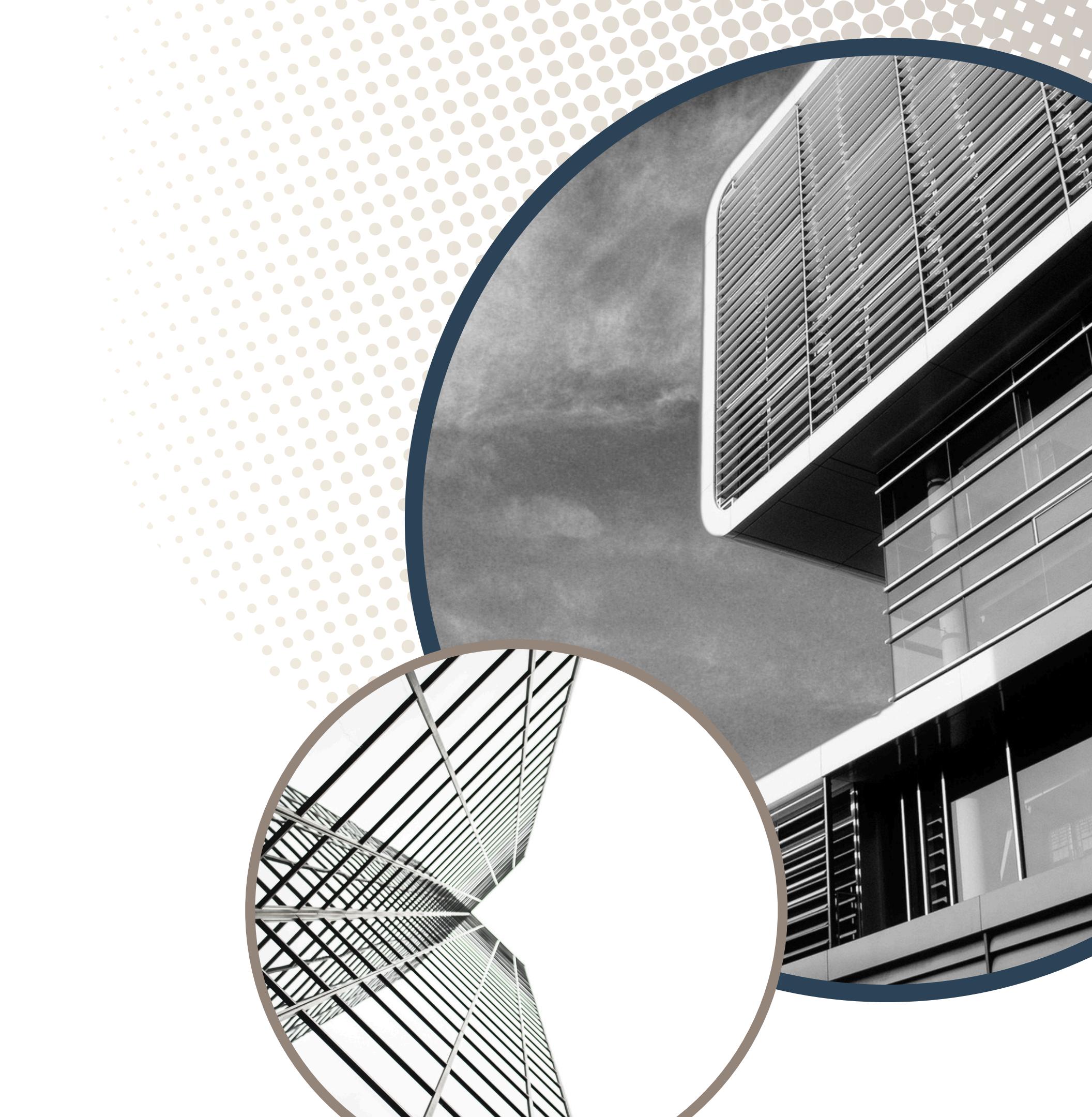


# **TEMPERA**

**Arbeitszeitmesser und  
Raumklimaüberwachung von  
Bürogebäuden**



# Unser Team



**FRONTEND**

Iolanthe



**FRONTEND**

Andreas



**RASPBERRY**

Leonardo



**BACKEND**

Christopher



**BACKEND**

Simon

# Folgen schlechter Klimabedingungen in Büros

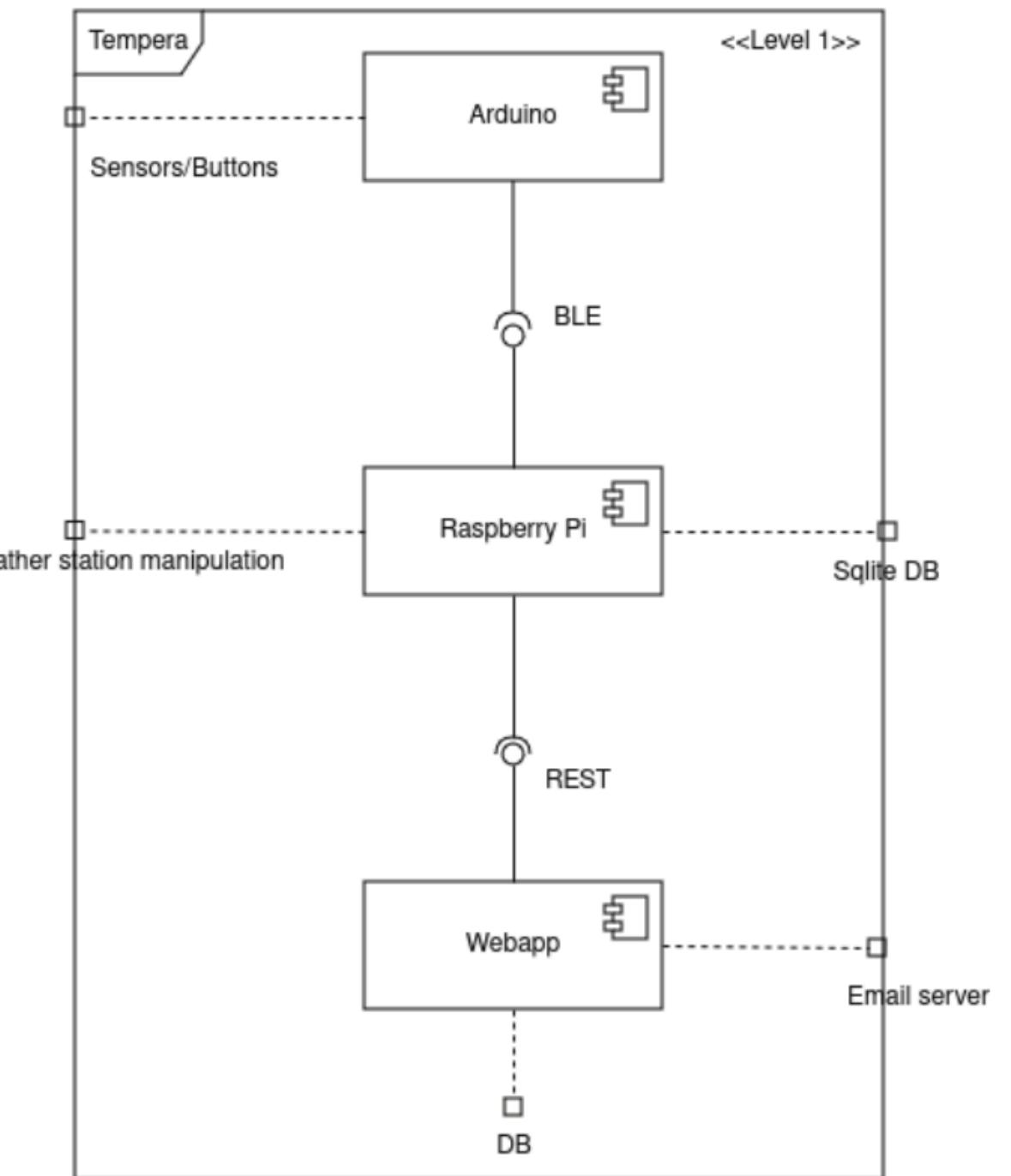
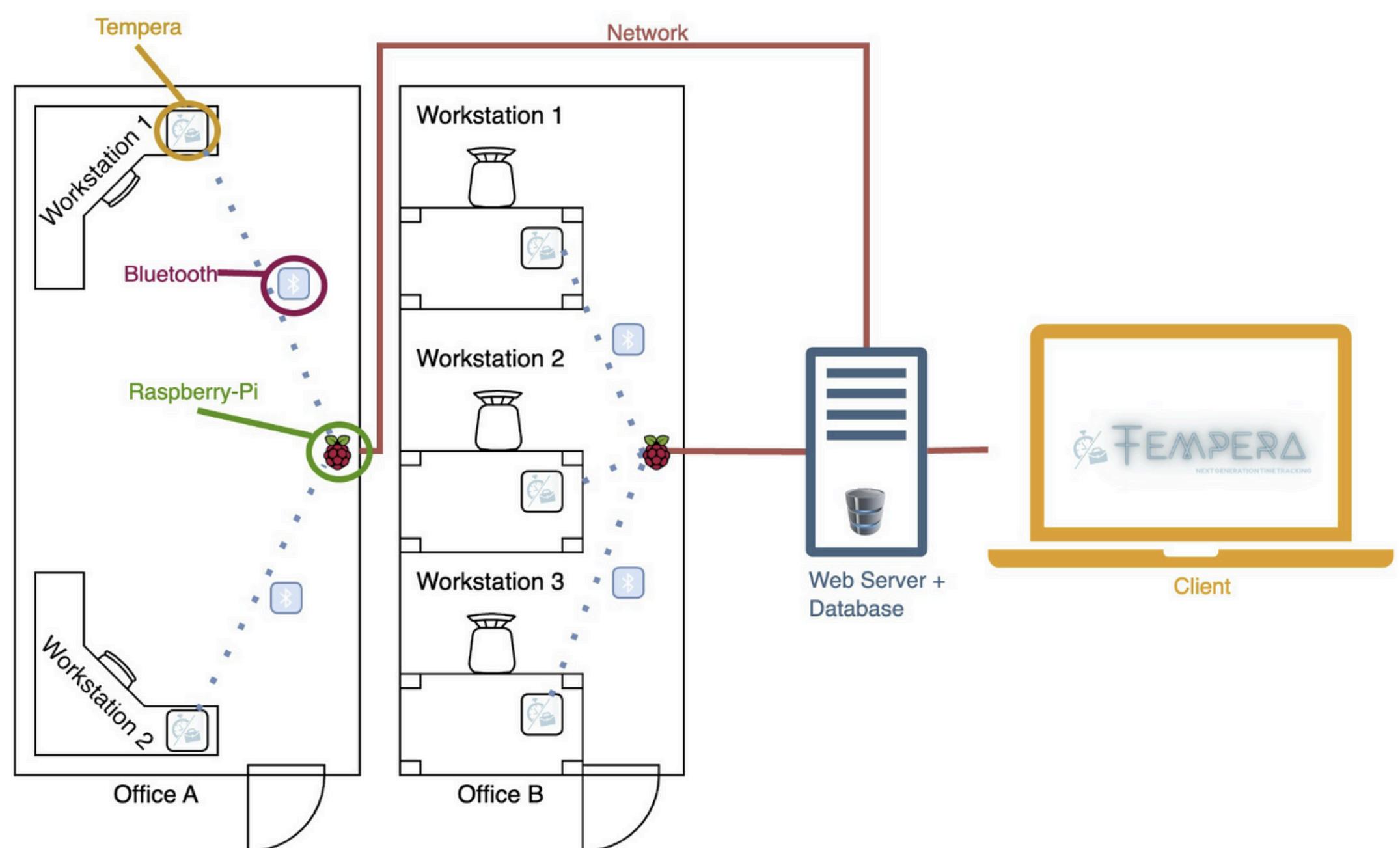
- 01 Schlechte Luftqualität:  
Müdigkeit, Kopfschmerzen und Konzentrationsschwäche
- 02 Hohe Luftfeuchtigkeit:  
Fördert das Wachstum von Schimmelpilzen und Milben->Allergien und Atemwegserkrankungen
- 03 Niedrige Luftfeuchtigkeit:  
Trocknet Schleimhäute aus und erhöht das Risiko von Infektionskrankheiten
- 04 Extreme Temperaturen:  
Können Unbehagen verursachen und folglich die Produktivität senken



# Motivation

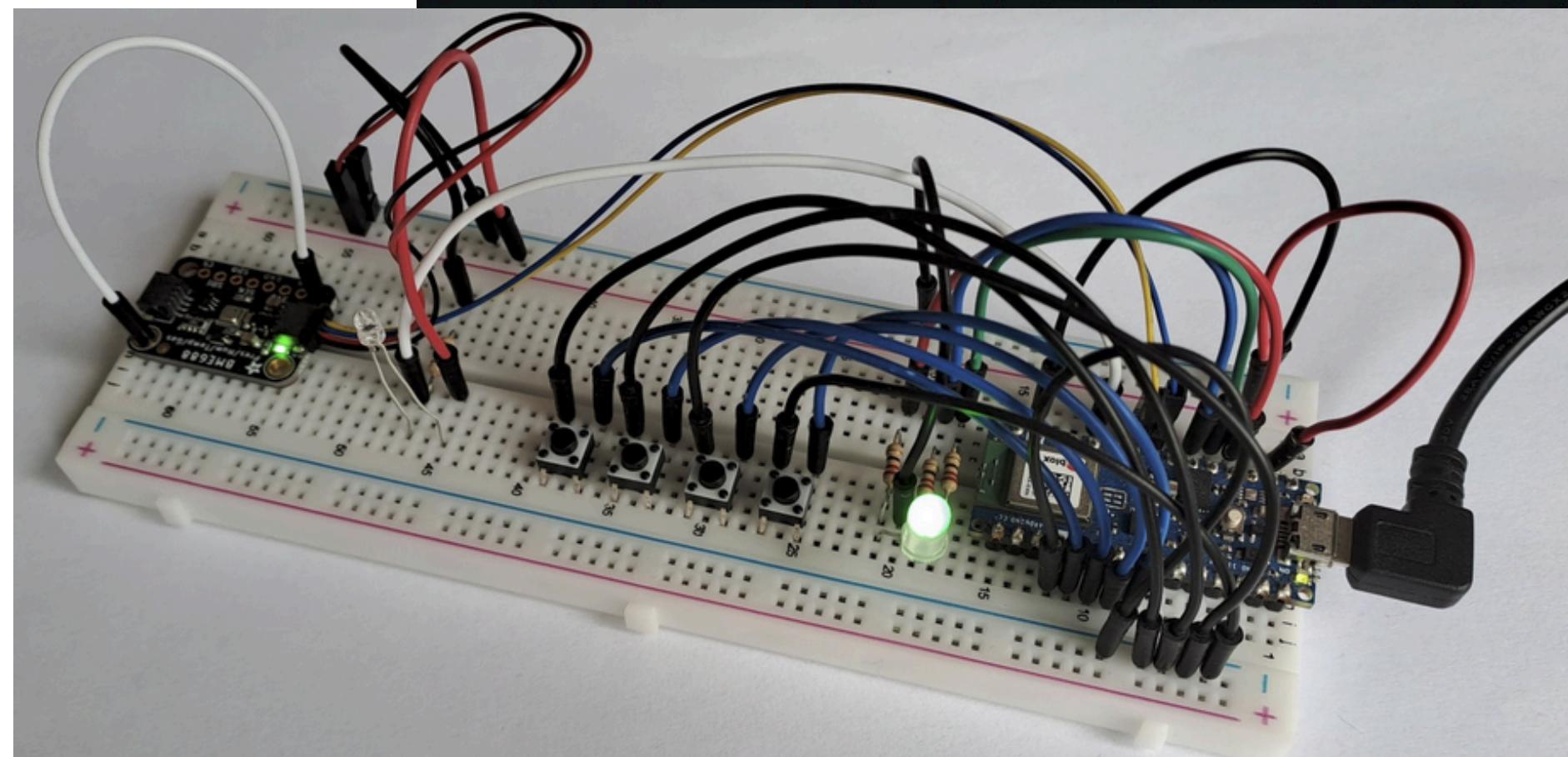
- **Genaues Zeit-Tracking:** Klare und nachvollziehbare Dokumentation der Arbeitszeiten
- **Verbesserung des Raumklimas:** Kontinuierliche Überwachung und Optimierung
- **Erhöhte Produktivität:** Reduzierung von Ablenkungen und Verbesserung von Wohlbefinden von Mitarbeitern
- **Energieeffizienz:** stromsparendes SmartDevice
- **Datensammlung:** Analyse zur Optimierung individueller und allgemeiner Leistung

# Setup



# Hardware

- Kernkomponenten:  
Arduino, Raspberry-Pi,  
RGB-LED
  - Datenübertragung über  
Bluetooth Low Energy  
(BLE) zu einem  
Accesspoint.

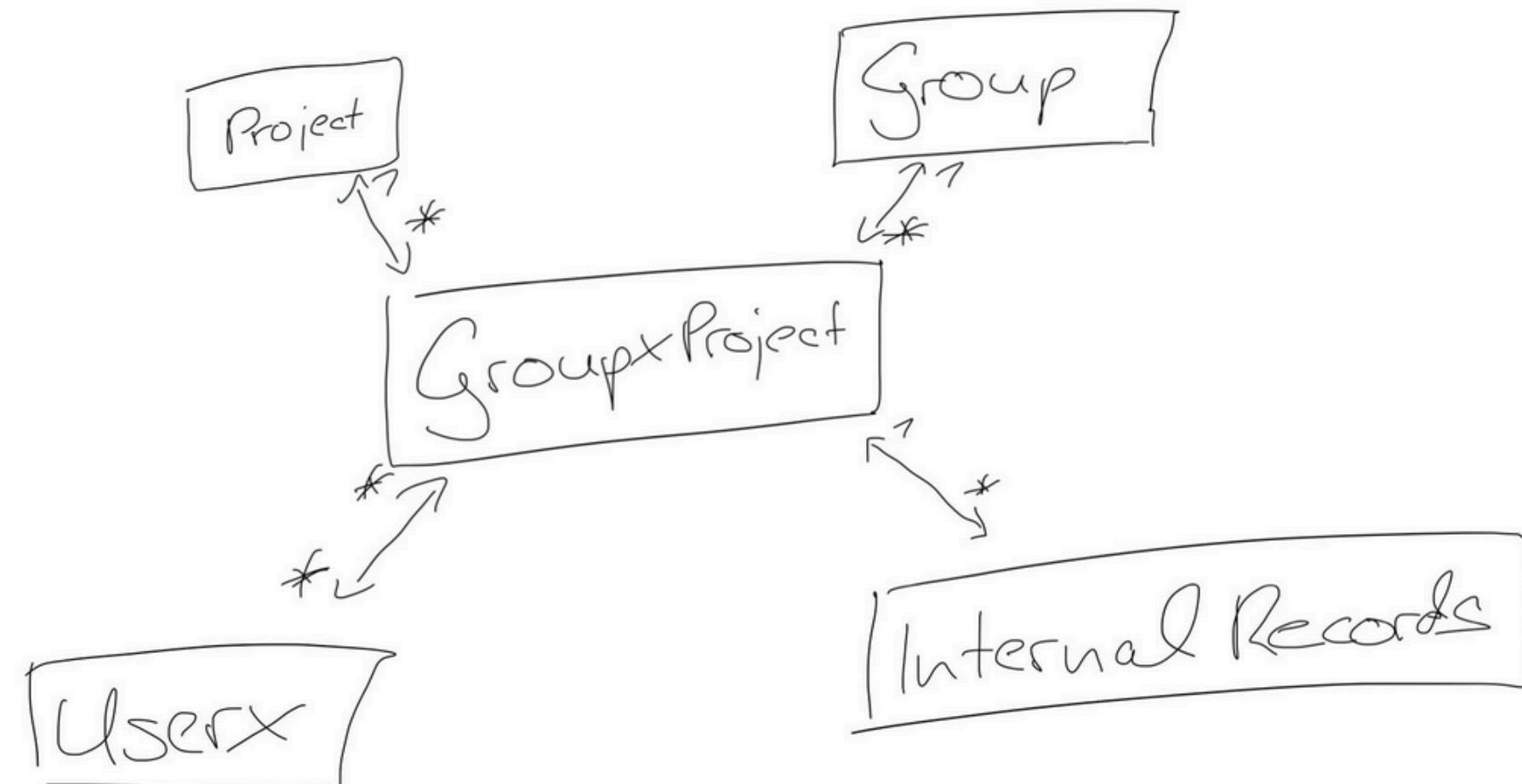


A black and white photograph of a printed circuit board (PCB). The board is densely populated with electronic components, including resistors, capacitors, and integrated circuits. A red wire is visible on the left side, and a yellow wire is on the right side. The board is mounted on a light-colored metal frame.

# Backend

- Spring Boot als Backend-Framework
- PostgreSQL als Datenbanksystem
- Hibernate als ORM (Object-Relational Mapping)
- Einfache Skalierung weil modular
- Transaktionssicherheit

```
@Id  
@ManyToOne(cascade = {CascadeType.PERSIST, CascadeType.MERGE}, fetch = FetchType.EAGER)  
private Groupx group;  
  
@Id  
@ManyToOne(cascade = {CascadeType.PERSIST, CascadeType.MERGE}, fetch = FetchType.EAGER)  
private Project project;  
  
4 usages  
@ManyToMany(cascade = {CascadeType.PERSIST, CascadeType.MERGE})  
private Set<Userx> contributors;  
  
4 usages  
@OneToMany(mappedBy = "groupxProject", cascade = CascadeType.ALL)  
private Set<InternalRecord> internalRecords;  
  
public GroupxProject() {  
    contributors = new HashSet<>();  
    internalRecords = new HashSet<>();  
}  
  
1 usage  
public void addContributor(Userx contributor) {  
    contributors.add(contributor);  
    contributor.getGroupxProjects().add(this);  
}
```



# Frontend

- Angular + PrimeNG + Tailwind + Swagger
- Anwendung auf allen Geräten (PC, Handy, Tablet) möglichst gleiches UI
- RESTful API-Kommunikation über HTTP

```
getAccesspointsByRoomId(roomId: string): Observable<AccessPoint> {
  return this.http.get<AccessPoint>( url: 'http://localhost:8080/api/rooms/accesspoint/' + roomId);
}
```

```
@GetMapping("/accesspoint/{roomId}")
public ResponseEntity<AccessPointDto> getAccessPoints(@PathVariable String roomId) {
```

# **Projektplanung**

# Zeitplan und Meilensteine

- **Bis 15.03.24** - Softwarekonzept:
  - Systemkonzepts, das alle Komponenten und deren Interaktionen beschreibt.
- **Bis 19.04.24** - Grundlagen Kommunikation und Webapp:
  - Kommunikation zwischen den Systemkomponenten.
  - Beginn der Entwicklung der Webanwendung, BLE und REST-APIs.
- **Bis 17.05.24** - Funktionale Systeme:
  - Fertigstellung funktionsfähiger Systemkomponenten
- **Bis 31.05.24** - Lauffähiges Projekt:
  - Integration zu einem vollständig funktionierenden Gesamtsystem.
- **Bis 07.06.24** - Abnahmetests:
  - Durchführung umfassender Tests, Identifikation und Behebung von Fehlern.
- **Bis 21.06.24** - Finale Version:
  - Abschluss aller Entwicklungs- und Testarbeiten, finale Überprüfungen

# Teammitglieder und Verantwortlichkeiten

- Simon:
  - Hauptverantwortlich für die Entwicklung an Arduino und anschließend Backend.
- Leo:
  - Fokus auf Raspberry und Backend-Entwicklung.
- Andi und Io:
  - Entwickeln gemeinsam das Frontend und unterstützen bei Backend-Aufgaben.
- Chris:
  - Arbeitet an Backend, unterstützt die Integration der Komponenten.

# Unsere Highlights I

## **Software-Architektur:**

- JPA & Hibernate (Entitygraphs, lazy loading, dto-Queries)
- Swagger
- Async (Frontend & Raspberry)

## **IoT & Systemzuverlässigkeit:**

- Crontab
- Tenacity

## **Softwarequalität:**

- SonarQube
- Code Reviews
- laufend Unit- & Integrationstests ergänzen

# Unsere Highlights II

## **Benutzungsoberfläche:**

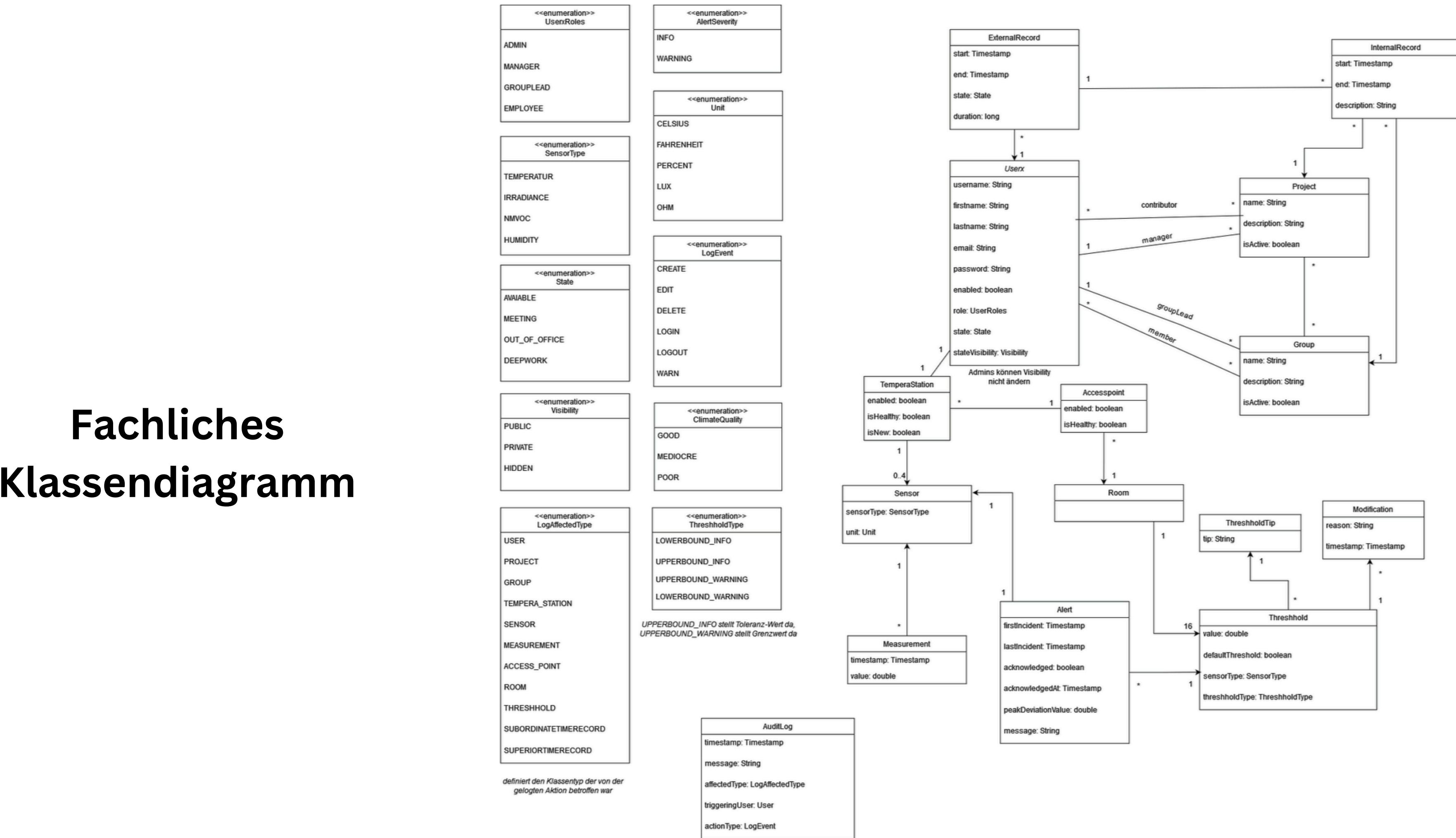
- Dashboard-Like-Approach
- Dark-Mode
- Einheitliches übersichtliches Design

## **Projektmanagement:**

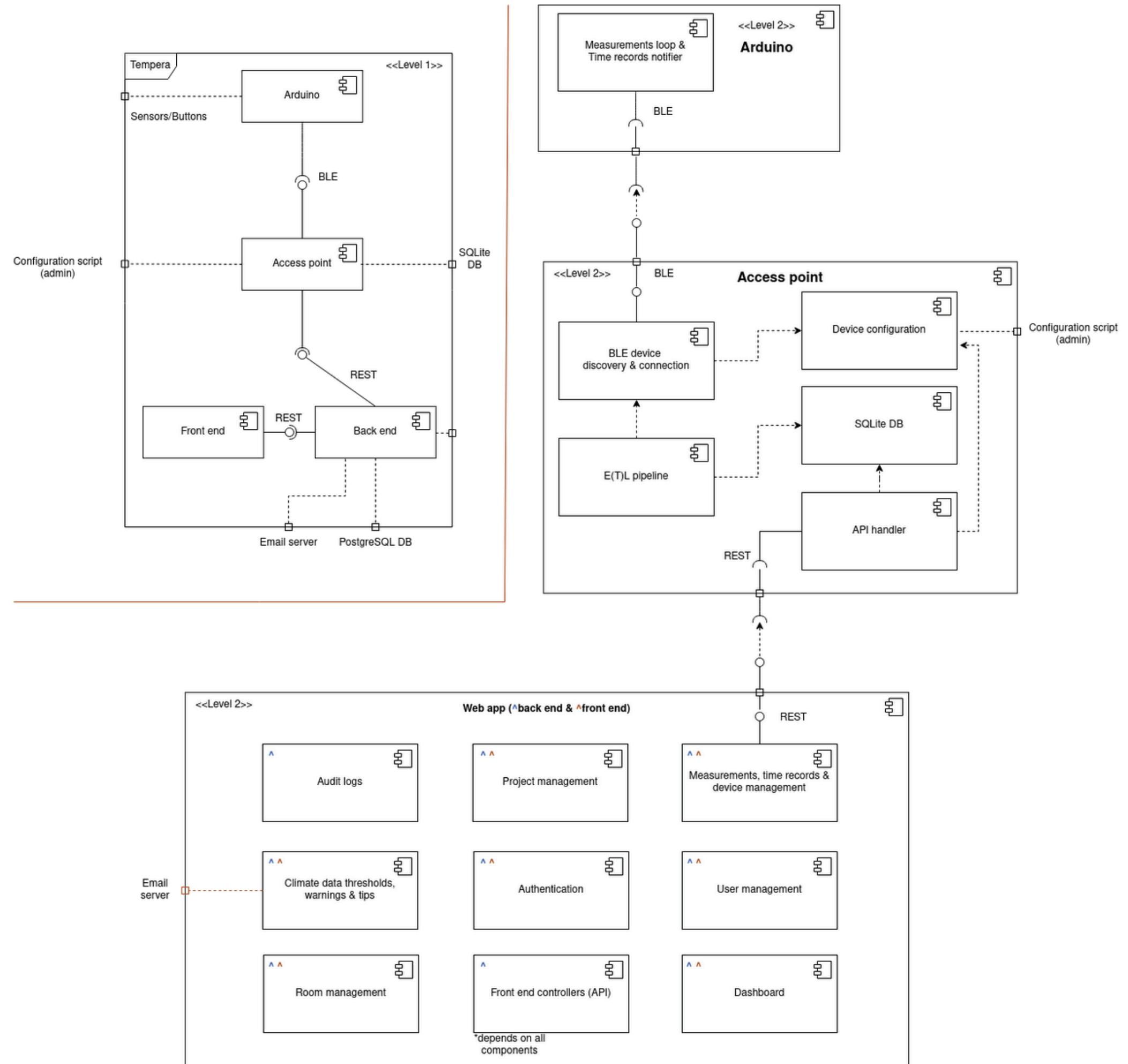
- Issue-Board & Labels in Git
- Unabhängige Entwicklung
- Wöchentliche Meetings
- Spezialisierung

# Artefakte

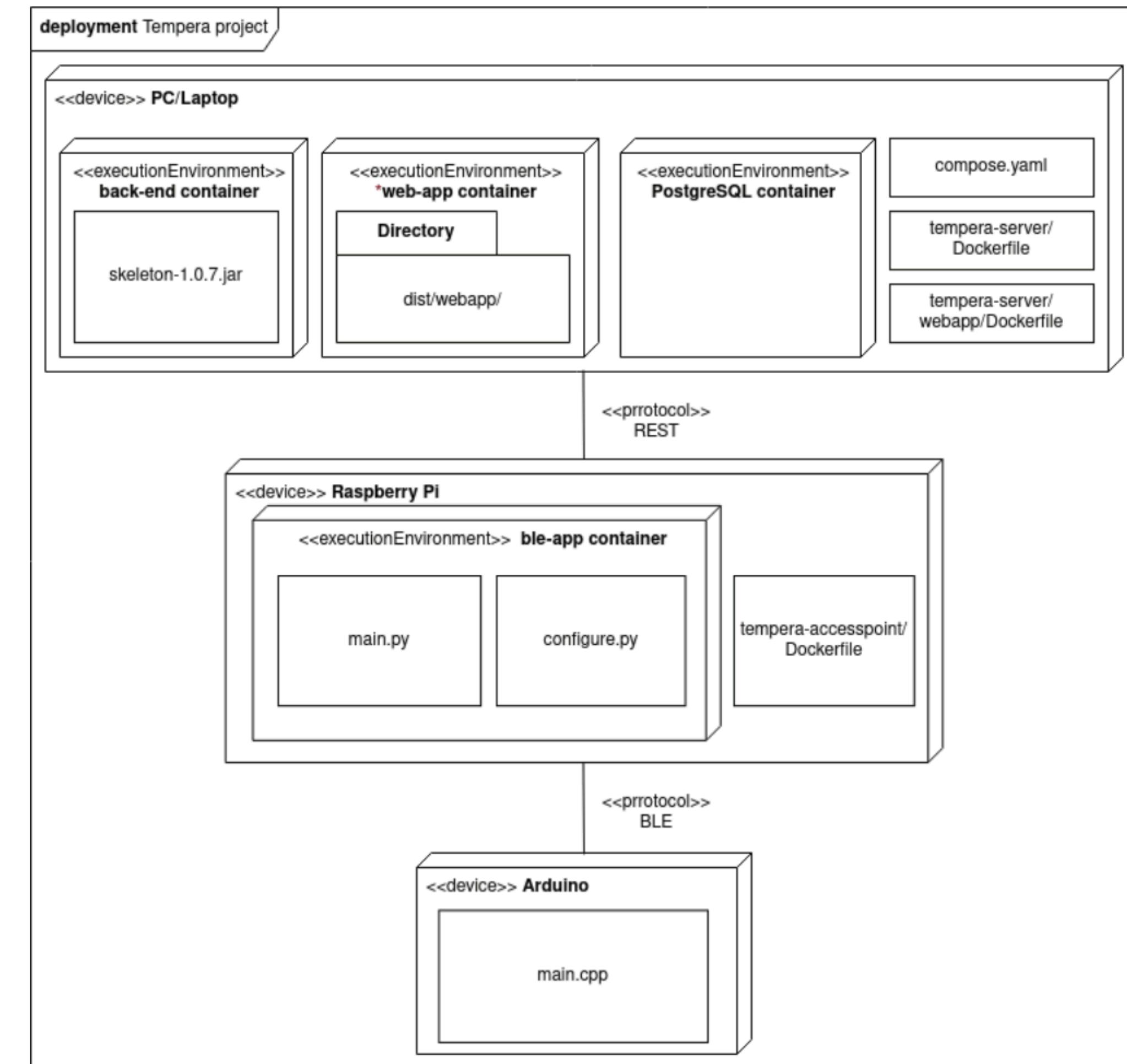
# Fachliches Klassendiagramm



# Komponentendiagramm



# Verteilungsansicht



\* The web-app docker uses 'ng serve' for ease of setup. Angular provides with this command a convenient web server for development, but not safe for production. For production, 'ng serve' should be replaced with a build stage using 'ng build' and serving of the artifact with a securely configured web server deployment stage using e.g., nginx.



*Unsere Philosophie*

**“PERFEKTION IST NICHT  
DANN ERREICHT, WENN  
MAN NICHTS MEHR  
HINZUFÜGEN, SONDERN  
WENN MAN NICHTS  
MEHR WEGLASSEN  
KANN.”**

---

ANTOINE DE SAINT-EXUPÉRY