



SensorBear

Project Management Report

Matthias Gregorich
Daniel Nebel
Viktor Novak
Damjan Petrovic
Marc Schneeweis

21.10.2025

Inhaltsverzeichnis

1	Product	1
1.1	Definition of Done	1
1.2	Deadlines	1
1.3	Partner	1
1.4	UI-Mockups	2
1.5	Requirements Definition	4
1.6	Collaboration Approach	4
1.7	Resources Stored	4
1.8	Product KPIs	5
1.9	System Architecture	6
1.10	Tech Stack	7
2	Project	8
2.1	Sprints	8
2.2	Kanban-Board	9
2.3	Project KPIs	10
2.4	Stakeholder Analysis	11
2.5	Responsibilities	12
2.6	Project Health Monitors	14
2.7	SWOT Analysis	15
2.8	Earned Value Analysis	17
2.9	Project Environment Analysis	17
2.10	Ceremonies	18
2.11	Version Management Approach	18
3	Attachments	19
3.1	Compliance Guidelines	19
3.2	Meeting Protocols	19
3.3	Time Records	22
3.4	Cooperation Contract	33
3.5	Legal Declaration	34
3.6	Software Requirements Specification	35

1 Product

1.1 Definition of Done

Dieses Projekt ist fertig so bald Bearingpoint die vollständig funktionsfähigen ESP32 Geräte hat, die alle spezifizierten Funktionen gemäß den technischen Anforderungen erfüllen, erfolgreich durch das Backend mit den verschiedenen Frontends (Web, Mobile, Desktop) kommunizieren und alle Meilensteile bereits erreicht und dokumentiert worden sind und das Projekt spätestens am 8. April 2026 abgeschlossen ist.

1.2 Deadlines

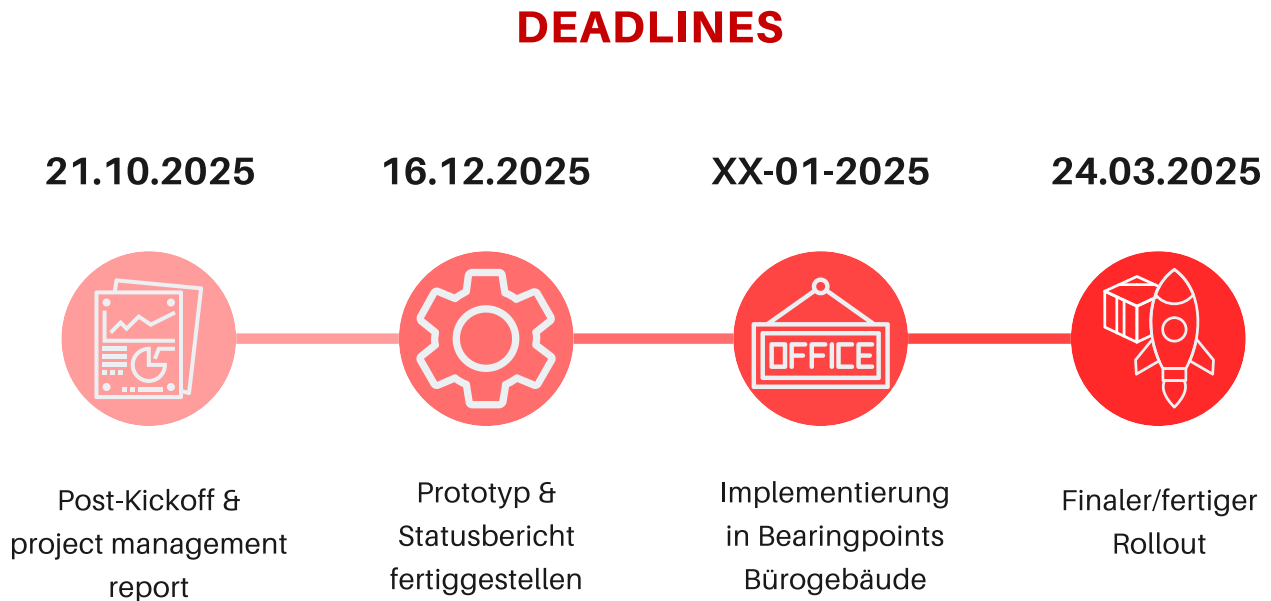


Abb. 1: Deadlines-Übersicht

1.3 Partner

BearingPoint

Abb. 2: Logo von BearingPoint

Der Partner und Auftraggeber dieses Projekts ist **BearingPoint**, eine international tätige Management- und Technologieberatung mit europäischen Wurzeln. Sie unterstützt Unternehmen und den öffentlichen Sektor bei Strategie, Transformation und der Umsetzung digitaler Lösungen. Schwerpunkte sind *Data & Analytics*, *Cloud/Software Engineering*, *Finance & Risk*, *Supply Chain & Operations*, *Customer & Growth* sowie *Regulierung und Compliance*. BearingPoint verbindet Beratung mit eigenen IP-basierten Lösungen und arbeitet mit Technologiepartnern wie SAP, Microsoft und Salesforce zusammen – von der Idee bis zum skalierbaren Betrieb. Betreute Branchen sind u. a. Finanzdienstleistungen, Automotive/Fertigung, Telekommunikation, Energie/Versorger, Transport/Logistik sowie der öffentliche Sektor.

1.4 UI-Mockups

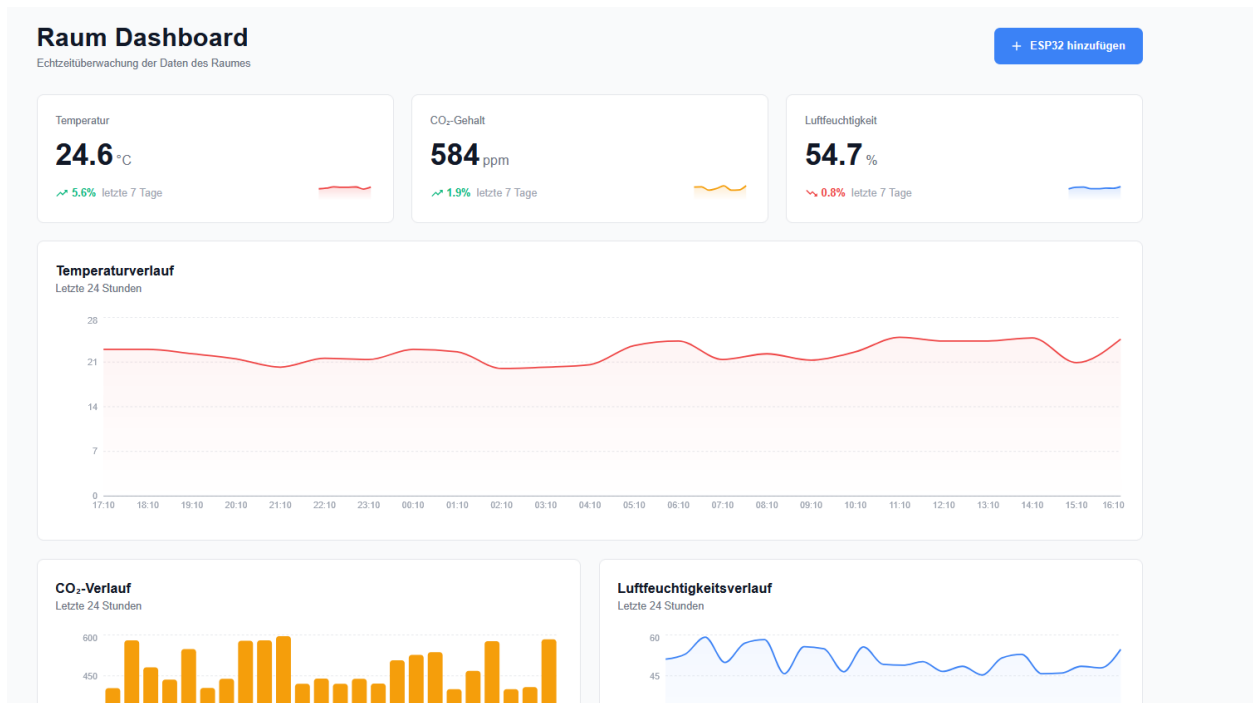


Abb. 3: UI-Mockup für die Web-Version



Abb. 4: UI-Mockup für die IOS-Version

1.5 Requirements Definition

Ausgangslage

- Zu hoher CO₂-Gehalt und zu hohe Temperaturen wirken sich negativ auf den Kreislauf aus. Darauf wird bei BearingPoint geachtet, um das Wohlbefinden der Mitarbeiter zu gewährleisten. Unser Produkt löst dieses Problem durch die Alarmierung bei Werten außerhalb des Normalbereiches, welche auf permanenten Messungen basieren.

Ziele & Aufgaben

- Die Erstellung einer Website, einer nativen Mobile App und einer Desktop Applikation die die Messdaten unseres Sensornetzwerkes so benutzerfreundlich wie möglich, durch Nutzung von nativen Features (z. B.: Widgets, Notifications...), darstellen. Dies ermöglicht den Nutzern die Überwachung der Raumumgebung.
- Das Integrieren neuer Sensoren wird benutzerfreundlich umgesetzt.
- Raumdarstellung samt Raumplanung wird intuitiv gestaltet

Funktionelle Anforderungen

- Datenspeicherung / Datenvisualisierung von vergangenen und Live-Daten
- Das Frontend bietet dem Endnutzer die Möglichkeit die Daten zu exportieren (json / csv)

1.6 Collaboration Approach

Kommunikation: Für die tägliche Kommunikation und den Austausch von Projektinformationen verwenden wir primär Discord (alternativ WhatsApp). Zusätzlich besprechen wir während jeder Projektentwicklungsstunde gemeinsam die aktuellen Aufgaben, den Fortschritt sowie die nächsten Schritte. Mit dem Partnerunternehmen kommunizieren wir über E-Mail sowie Meetings in ihrem Büro.

Aufgaben- und Versionsmanagement: Zur Organisation unserer Aufgaben und User Stories setzen wir ein Kanban-Board in Atlassian Jira ein. Jede Aufgabe wird einem Teammitglied zugewiesen und mit Story Points zur Priorisierung versehen. Die Versionsverwaltung des Source Codes erfolgt über ein GitHub-Repository, in dem neue Features über separate Branches entwickelt und mittels Pull Requests überprüft werden.

Dokumenten- und Ressourcenverwaltung: Alle projektrelevanten Dokumente (z. B. Präsentationen, Berichte, Protokolle, Diagramme usw.) werden in einem gemeinsamen OneDrive-Ordner gespeichert. Dieser ist für alle Teammitglieder zugänglich und strukturiert organisiert, um eine zentrale und übersichtliche Ablage sicherzustellen.

Meetings und Reviews: Alle drei Wochen finden Sprint-Meetings statt, in denen der aktuelle Fortschritt evaluiert und die Planung für den kommenden Sprint vorgenommen wird. Meeting-Protokolle werden ebenfalls im OneDrive abgelegt, um eine lückenlose Dokumentation zu gewährleisten.

Zielsetzung: Ziel dieses Vorgehens ist es, sicherzustellen, dass alle Teammitglieder stets informiert sind, Aufgaben klar verteilt werden und das Projekt effizient sowie in guter Zusammenarbeit umgesetzt wird.

1.7 Resources Stored

Wo und wie Ressourcen gespeichert und für Stakeholder verfügbar sind

Alle projektbezogenen Ressourcen werden in gemeinsam genutzten, versionierten und leicht zugänglichen Bereichen gespeichert, um Transparenz und eine effiziente Zusammenarbeit zwischen allen Teammitgliedern und Betreuern sicherzustellen.

1. Code & Versionsverwaltung

Plattform: GitHub

Inhalt: Quellcode der ESP32-Firmware, der API und des Web-Dashboards

Zugriff: Entwicklerteam mit Schreibrechten; Betreuungslehrkraft mit Leserechten (bei Bedarf)

Struktur: Eigene Branches für neue Features, Dokumentationen in README

2. Dokumentation & Berichte

Plattform: OneDrive (gemeinsamer Projektordner)

Inhalt: Projektdokumentation, Präsentationen, Sitzungsprotokolle, Recherchen und Diagramme

Zugriff: Alle Teammitglieder und der betreuende Lehrer

3. Design & Visualisierungen

Plattform: Figma

Inhalt: UI/Mockups, Systemarchitektur/Diagramme und Dashboard/Entwürfe

Zugriff: Design/ und Entwicklungsteam mit Bearbeitungsrechten

4. Sensordaten & Datenbank

Plattform: Gehostete Datenbank

Inhalt: Erfasste Messdaten (CO₂/Konzentration, Temperatur, Luftfeuchtigkeit, Lautstärke)

Zugriff: Nur Entwickler und Administratoren; im Dashboard sind die Daten lesbar, aber nicht veränderbar

5. Kommunikation & Koordination

Plattform: Discord (alternativ WhatsApp)

Inhalt: Tägliche Kommunikation, Aufgabenabstimmung, kurze Statusmeldungen

Zugriff: Nur Projektteam

Ressourcentyp	Plattform	Zugriffsebene	Zweck
Quellcode	GitHub	Entwickler, Lehrer (Lesen)	Versionskontrolle & Zusammenarbeit
Dokumentation	OneDrive	Team + Lehrer	Berichte, Präsentationen, Protokolle
UI & Architektur	Figma	Team intern	Mockups, Systemdarstellung
Sensordaten	Datenbank	Entwickler	Speicherung & Analyse der Messdaten
Kommunikation	Discord / Teams	Team intern	Koordination und Abstimmung

Tab. 1: Übersicht gespeicherter Ressourcen

Übersichtstabelle

1.8 Product KPIs

- **Dashboard-Ladezeit (LCP – Largest Contentful Paint)**
 - **Definition:** Zeit, bis das größte sichtbare Inhaltselement geladen wurde.
 - **Ziel:** Die Messung der Ladezeit des größten Inhaltselements dient zur Feststellung der Endnutzerverfahrung. Bei zu hohen Messwerten sind Optimierungen vorzunehmen.
- **API-Query-Latency**
 - **Definition:** Misst die End-to-End-Latenz von API-Anfragen.
 - **Ziel:** Die Messung der Latenz von API-Anfragen ermöglicht die gezielte Optimierung des Backends, um den Nutzern ein flüssiges Frontend bieten zu können.

- **Test-Success-Rate**

- **Definition:** Misst, wie viele Code-Tests prozentuell erfolgreich sind.
- **Berechnung:** $(\text{Anzahl der erfolgreichen Tests}) / (\text{Anzahl der durchgeführten Tests}) \times 100$.
- **Ziel:** Die Test-Success-Rate gibt Auskunft über die Codequalität.

1.9 System Architecture

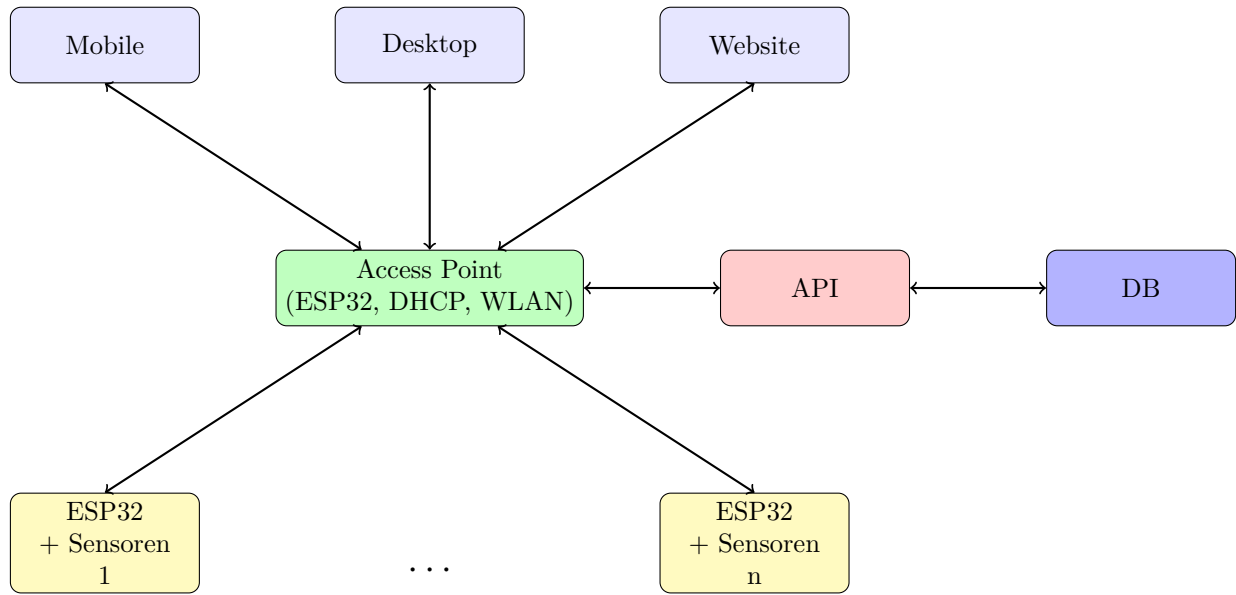


Abb. 5: Übersicht der System Architecture.

Frontend (Mobile, Desktop, Website)

Dient als Benutzerschnittstelle des Systems. Läuft auf mobilen Geräten, Desktop oder im Webbrowser und greift über das lokale Netzwerk auf die API zu. REST-Anfragen liefern Konfiguration und historische Daten, WebSocket-Verbindungen Echtzeitwerte.

Access Point (ESP32, DHCP, WLAN)

Der ESP32-basierte Access Point stellt das lokale, WPA2-gesicherte WLAN bereit und fungiert als DHCP-Server. Er verbindet Sensorgeräte, Frontends und Backend-Komponenten innerhalb des geschlossenen Netzwerks.

API (Application Programming Interface)

Verarbeitet Anfragen der Frontends und Daten der Sensorgeräte und kommuniziert mit der Datenbank. Stellt eine REST-Schnittstelle für Abfragen und eine WebSocket-Schnittstelle für Live-Updates bereit.

Datenbank (DB)

Speichert Messwerte, Raum- und Gerätemetadaten und wird ausschließlich über die API angesprochen.

ESP32 + Sensoren

Jede Messstation besteht aus mindestens einem ESP32-Gerät, der mit bis zu vier Sensoren (z. B. Temperatur, CO₂, Luftfeuchtigkeit, Schalldruck) ausgestattet sein kann. Der ESP32 erfasst die verfügbaren Messwerte und überträgt sie kontinuierlich oder ereignisgesteuert über den Access Point an die API.

1.10 Tech Stack

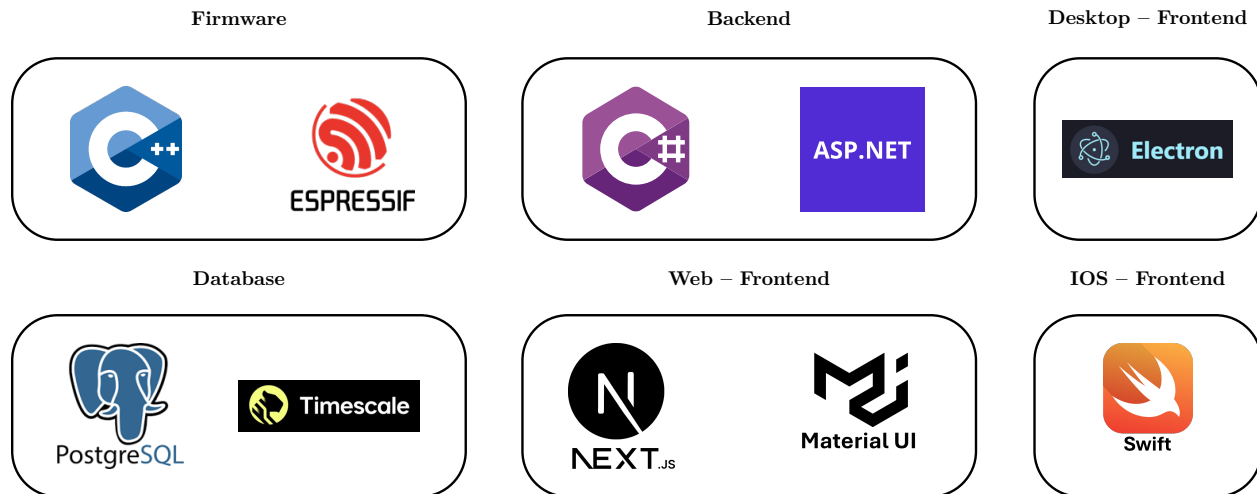


Abb. 6: Tech Stack Übersicht

- **Firmware**

- **C++:** Die gesamte Firmware für unsere ESP32-basierten Messstationen und den Access Point (AP) wird in C++ geschrieben. Da C++ direkt zu plattformspezifischem Assembly compiled wird, ist für Exekution des Codes auf dem ESP32 keine zusätzliche Laufzeitumgebung (z. B. Python-Interpreter) notwendig. Dadurch reduziert sich die Größe der Firmware, was hilft, den Größeneinschränkungen der ESPs aus dem Weg zu gehen. Ein weiterer Grund für die Wahl von C++ als Programmiersprache für die Firmware ist das uneingeschränkte Repertoire von Software-Libraries. Bei Alternativen, wie MicroPython, können nur Software-Libraries genutzt werden, für die Bindings erstellt wurden.
- **Espressif-IDF:** Gewählt wurde Espressif-IDF als Entwicklungs-Tool-Chain, da diese direkt vom Hersteller zur Verfügung gestellt wird und daher auf dem neusten Stand ist. Alternativen, wie die Arduino IDE Tool-Chain, basieren auf einer älteren Version von Espressif-IDF und bieten weniger direkte Kontrolle, da sie eher an Anfänger gerichtet ist.

- **Backend**

- **C#:** C# wurde als Sprache gewählt, da sie memory-safe ist, aber trotzdem eine angemessene Performanz aufweist. Die große Auswahl an von Microsoft erstellten Libraries für häufige Anwendungsfälle ist ein weiterer Aspekt, der in die Entscheidung eingeflossen ist. Sprachen wie Rust oder Go wurden erwägt, bieten jedoch keinen suffizienten Mehrwert im Vergleich zu C#, eine Sprache, die nicht erlernt werden muss.
- **ASP.NET:** ASP.Net wurde als Library zur Implementierung der API gewählt, da diese direkt von Microsoft, dem Maintainer des .NET-Ökosystems, zur Verfügung gestellt wird und daher in der .NET-Welt ein Industrie-Standard ist.

- **Desktop-Frontend**

- **Electron:** Electron ist ein Framework, mit dem sich Desktop-Anwendungen mittels Webtechnologien (HTML, CSS, JavaScript) entwickeln lassen. Es ermöglicht die Einbindung von Webframeworks (React, etc.). Das ermöglicht die Wiederverwendung von Code unseres Web-Frontends. Es kombiniert Chromium (für die Darstellung) und Node.js (für Systemzugriffe) und ermöglicht das Erstellen von Crossplatform Desktop-Applikationen mit Zugriff auf plattformspezifische APIs (z. B. File-System, Benachrichtigungen, etc.).

- **Database**

- **PostgreSQL:** PostgreSQL ist ein open-source und kostenlos für die kommerzielle Nutzung.

PostgreSQL ist zuverlässig für große Datenmengen und bietet durch ein Plugin-System Erweiterbarkeit

- **TimescaleDB:** TimescaleDB ist ein Plugin für PostgreSQL, welches ermöglicht, ausgewählte Tables für Timeseries-Data zu optimieren. Es ermöglicht uns die relationalen Eigenschaften von PostgreSQL zu nutzen, ohne auf Performanz der Timeseries-Data-Queries zu verzichten.

- **Web-Frontend**

- **Next.js:** ist ein React-basiertes Framework für die Entwicklung moderner Webanwendungen. Es erweitert React um wichtige Features wie serverseitiges Rendern (SSR), statische Seitengenerierung (SSG). Diese Features ermöglichen das Erstellen einer performanten Website.
- **Material-UI:** Material UI (MUI) ist eine weit verbreitete React-Komponentenbibliothek, die Googles Material Design umsetzt. Die Bibliothek bietet ein konsistentes und modernes Design out-of-the-box.

- **Mobile-Frontend**

- **SwiftUI:** SwiftUI ist ein modernes UI-Framework von Apple zur Entwicklung von Benutzeroberflächen für iOS, macOS, watchOS und tvOS und basiert auf der Programmiersprache Swift. Die native Ausführung sorgt für eine flüssige Benutzererfahrung und Zugriff auf alle Gerätefunktionen (z. B. Widgets).

2 Project

Wir arbeiten mit einem **agilen Projektmanagement-Ansatz**. Die Planung, Verfolgung und Abstimmung erfolgen in **Jira** auf Basis eines **Kanban-Boards**.

2.1 Sprints

Die Sprint-Dauer beträgt in diesem Projekt 3 Wochen. Am Beginn eines Sprints planen wir die Inhalte, am Ende ziehen wir ein kurzes Review und aktualisieren das Board.

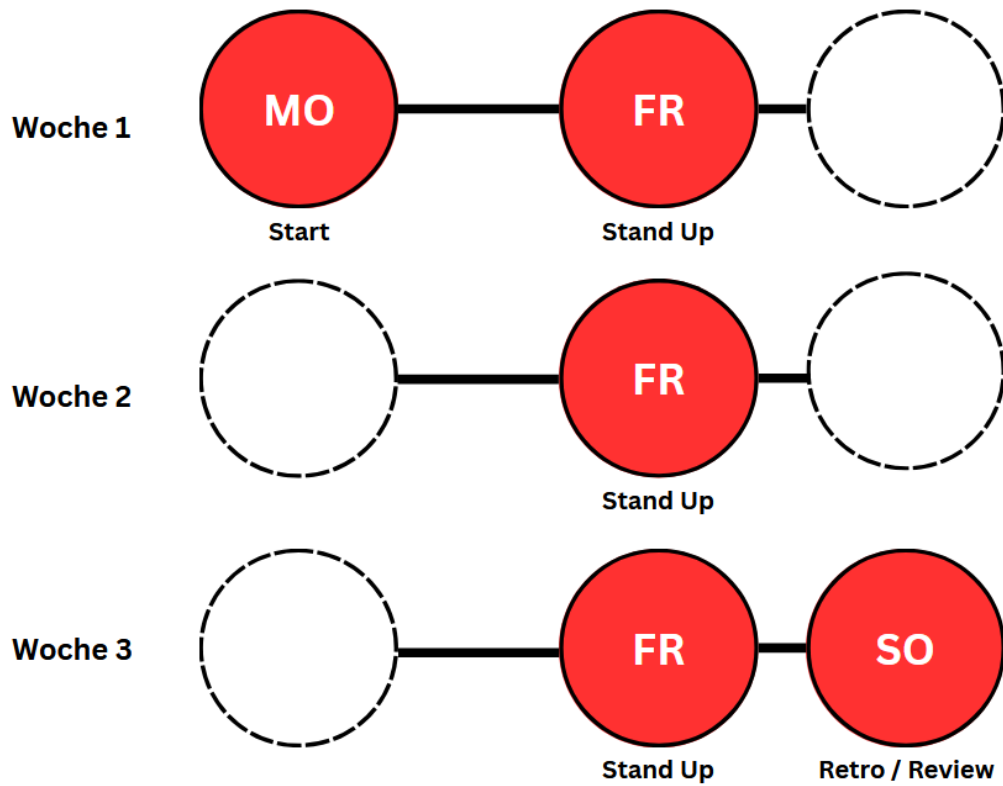


Abb. 7: Ablauf eines 3-wöchigen Sprints

2.2 Kanban-Board

Unser Board besteht aus **vier Spalten**, durch die alle Tickets von links nach rechts wandern:

- **Zu erledigen** — priorisierte, startklare Aufgaben
- **In Arbeit** — aktuell bearbeitete Aufgaben
- **Testing** — Ergebnisse in Prüfung/Abnahme
- **Fertig** — abgeschlossene Aufgaben

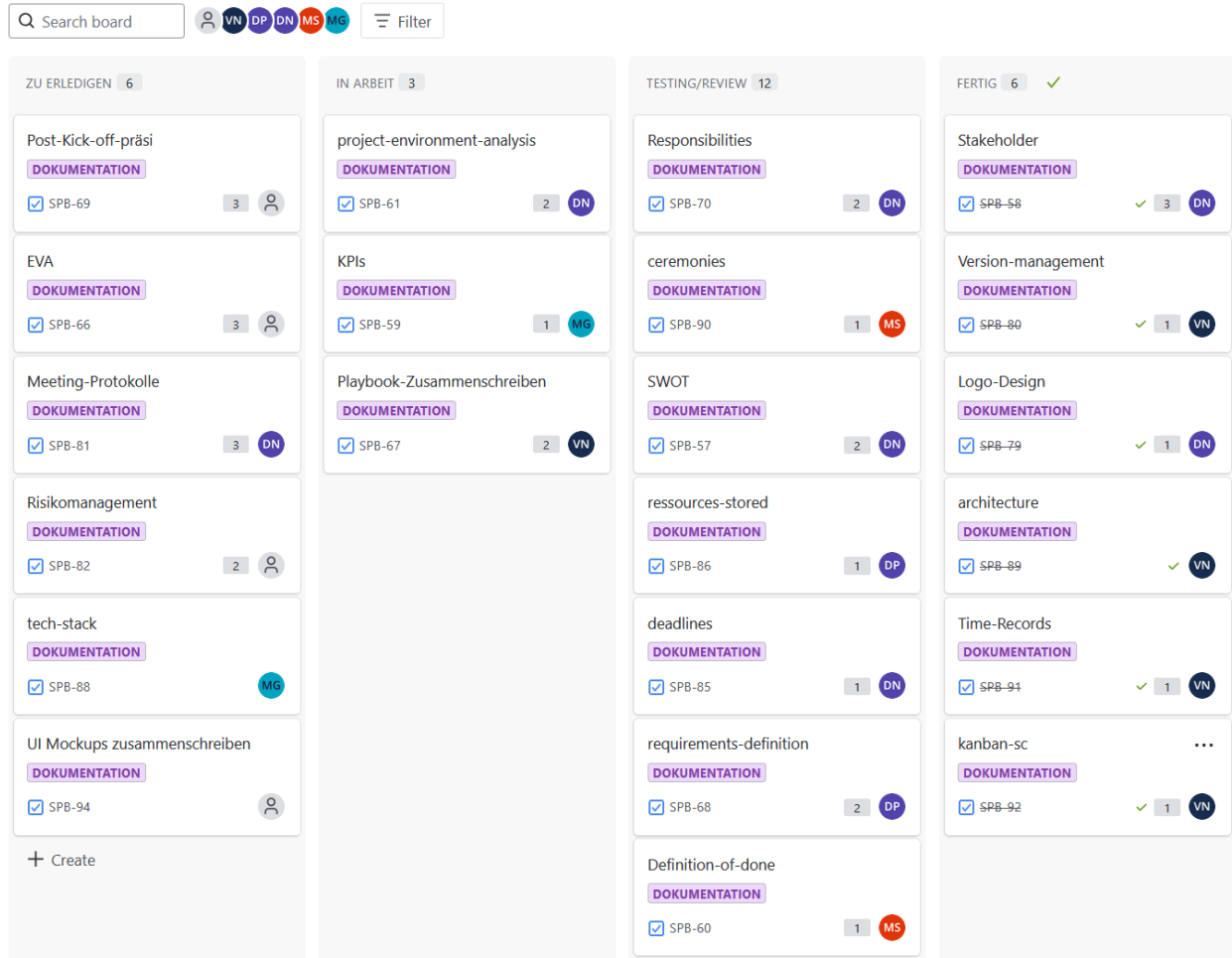


Abb. 8: Kanban-Board vom 20.10.2025

2.3 Project KPIs

Spillover-Rate

Definition: Misst, wie viele Jira-Vorgänge in den nächsten Sprint übernommen werden.

Berechnung: $\frac{\text{Anzahl der unvollständigen Vorgänge eines Sprints}}{\text{Anzahl der gesamten Vorgänge eines Sprints}} \times 100$

Ziel: Die Spillover-Rate hilft, die Selbsteinschätzung der Arbeitskapazität zu beurteilen.

Deadline-Verfehlungen

Definition: Anzahl der verfehlten Deadlines.

Ziel: Die Anzahl der Deadline-Verfehlungen gibt Auskunft über die Effizienz des Projekts und die Selbsteinschätzungsfähigkeit des Projektteams.

Teamzufriedenheit

Definition: Durchschnittliche Zufriedenheit basierend auf einer Team-Umfrage.

Berechnung: Durchschnitt der Bewertungen aller Mitglieder auf einer Skala von 1–10.

Ziel: Die Teamzufriedenheit spiegelt das Projektumfeld wider. Bei niedrigen Werten können nach einer Teamdiskussion Maßnahmen abgeleitet werden, um Stimmung und Arbeitsbedingungen zu verbessern.

2.4 Stakeholder Analysis

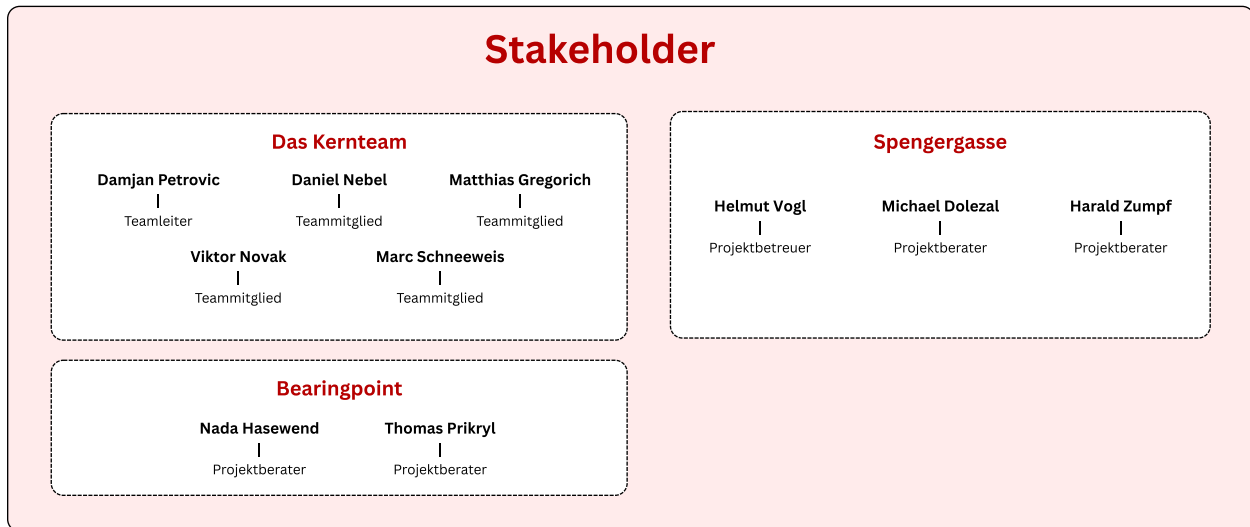


Abb. 9: Stakeholder Analysis — Übersicht

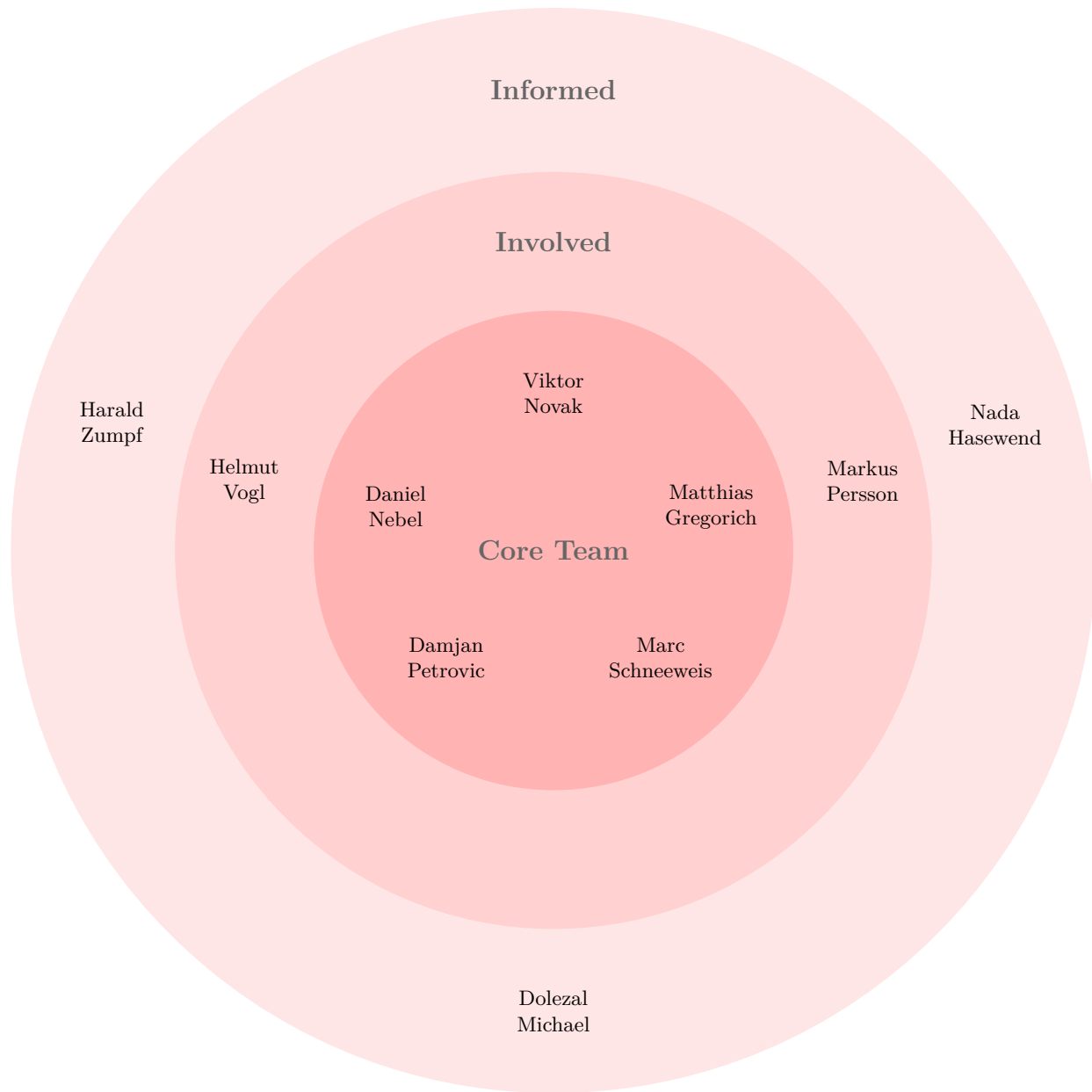


Abb. 10: Stakeholder-Kreise: Core Team, Involved und Informed.

2.5 Responsibilities

Die Zuteilung der Verantwortlichkeiten im *SensorBear*-Projekt orientiert sich an den verschiedenen Projektspekten. So erhält jede Aufgabe die notwendige Aufmerksamkeit und jede Person kann in ihrem gewünschten Fachbereich arbeiten.

- **Matthias Gregorich – Infrastruktur und Datenschnittstellen:** Implementiert die Systeminfrastruktur sowie Schnittstellen zur Kommunikation zwischen Sensoren, Clients und Datenspeicherung. Bewertet und wählt geeignete Technologien für Persistierung und Datenübertragung. Zentral für den Aufbau einer skalierbaren, robusten und effizienten Backend-Struktur.
- **Daniel Nebel – Mobile Applikation:** Verantwortlich für die Entwicklung der mobilen Anwendung des Projekts. Fokus auf benutzerfreundliches Design, hohe Performance und eine intuitive UI/UX für die Darstellung von Messdaten. Ziel ist eine einfache Navigation und ein optimales Nutzererlebnis auf

mobilen Endgeräten.

- **Viktor Novak – Qualitätssicherung und Automatisierung:** Entwickelt Strategien zur Qualitätssicherung und Automatisierung projektspezifischer Anwendungen. Schwerpunkt auf automatisierten Testverfahren zur Verbesserung der Codequalität. Zusätzlich Umsetzung einer Desktop-Applikation mit Electron, die Echtzeit-Push-Benachrichtigungen und erweiterte Datenansichten ermöglicht.
- **Damjan Petrovic – Netzwerk- und Sensorkomponenten:** Verantwortlich für die Konstruktion und Programmierung von ESP32-basierten Sensorknoten in einem Mesh-Netzwerk. Der Fokus liegt auf zuverlässiger Kommunikation, stabiler Datenübertragung und Integration der Sensoren in das Gesamtsystem (Pairing). Fundamentaler Beitrag zur Schaffung der Hardware- und Kommunikationsbasis des Projekts.
- **Marc Schneeweis – Web-Dashboard-Entwicklung:** Konzipiert und entwickelt ein webbasiertes Dashboard zur Visualisierung von Live-Daten, Statistiken und Warnmeldungen. Fokus auf klare Informationsdarstellung, intuitive Benutzeroberfläche und performante Datenanbindung. Trägt wesentlich zu einer transparenten und reaktionsschnellen Datenvisualisierung bei.

2.6 Project Health Monitors

























Attribute	Post Kickoff	Prototyp	finaler Rollout
Teamzusammenarbeit			
Ausgewogenes Team			
Gemeinsames Verständnis			
Vielfalt fördern			
Werte und Kennzahlen			
Geeignete Arbeitsweisen			
Engagement und Unterstützung			
Kontinuierliche Verbesserung			

Abb. 11: Project Health Monitors

2.7 SWOT Analysis

SWOT — Produkt

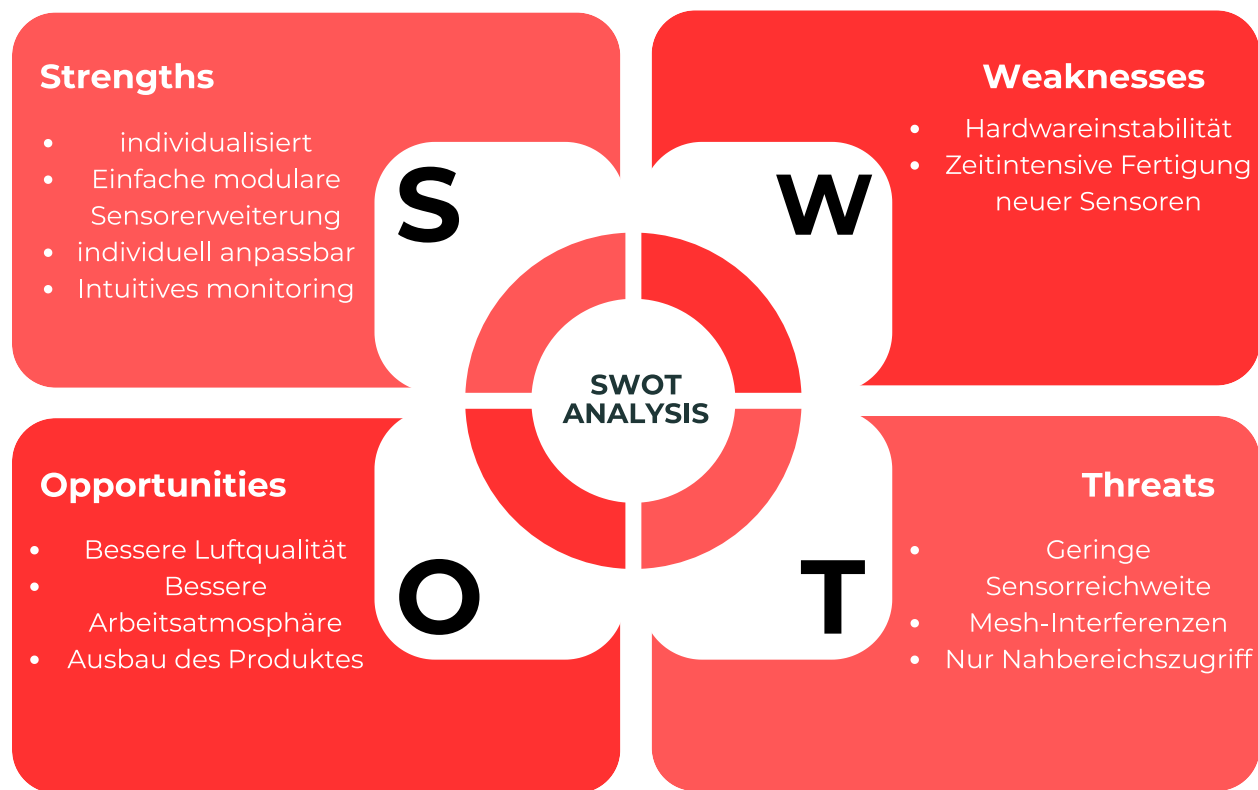


Abb. 12: SWOT-Analyse — Produkt

Strengths

- **Individualisiert:** Das Produkt ist auf die Bedürfnisse sowie auf das Bürogebäude von BearingPoint (Wiedner Gürtel 13/Turm 24, 1100 Wien) zugeschnitten.
- **Einfache modulare Sensorerweiterung:** Für Erweiterungen bestehender oder neuer Räume stehen neue „Plug-and-Play“-Sensoren bereit, die mittels „Pairing Mode“ schnell und unkompliziert konfiguriert werden können.
- **Individuell anpassbar:** Verbundene Benutzerinnen und Benutzer können Grenzwerte, Push-Benachrichtigungen und das Design nach ihren jeweiligen Anforderungen festlegen.
- **Intuitives Monitoring:** Optimierte, benutzerfreundliche Visualisierungen der Sensordaten ermöglichen eine übersichtliche und wirksame Überwachung.

Weaknesses

- **Hardwareinstabilität:** Überhitzung an besonders heißen Tagen, Verschmutzung oder Abdeckung der Sensoren können den Datenfluss beeinträchtigen.
- **Zeitintensive Fertigung neuer Sensoren:** Erweiterungen über die fünf bereitgestellten Messgeräte hinaus erfordern Bestellung zusätzlicher Sensoren und ESP32, das Drucken passender Gehäuse sowie die Konfiguration des zusammengesetzten Messgeräts — ein zeitaufwändiger Prozess.

Opportunities

- **Bessere Luftqualität:** Die CO₂-Messung und entsprechende Hinweise bei Unter- bzw. Überschreitung fördern eine bessere Raumluft.

- **Bessere Arbeitsatmosphäre:** Empfohlene Grenzwerte — angelehnt an *WHO*-Empfehlungen und Metastudien — unterstützen eine optimale, produktive Arbeitsumgebung.
- **Ausbau des Produkts:** BearingPoint kann die bereitgestellten Daten und das System als Grundlage für zukünftige Erweiterungen und neue Funktionen nutzen.

Threats

- **Geringe Sensorreichweite:** Bei großen Büroflächen mit niedriger Messgerätedichte kann das Mesh-Netz instabil werden.
- **Mesh-Interferenzen:** Externe Mesh-Netzwerke, verwinkelte Gebäude oder Stahlbetonwände können zu Störungen im Mesh-Netzwerk führen.
- **Nur Nahbereichszugriff:** Die Bedienung des Produkts ist auf den Nahbereich des aufgespannten Mesh-Netzwerks beschränkt.

SWOT — Team

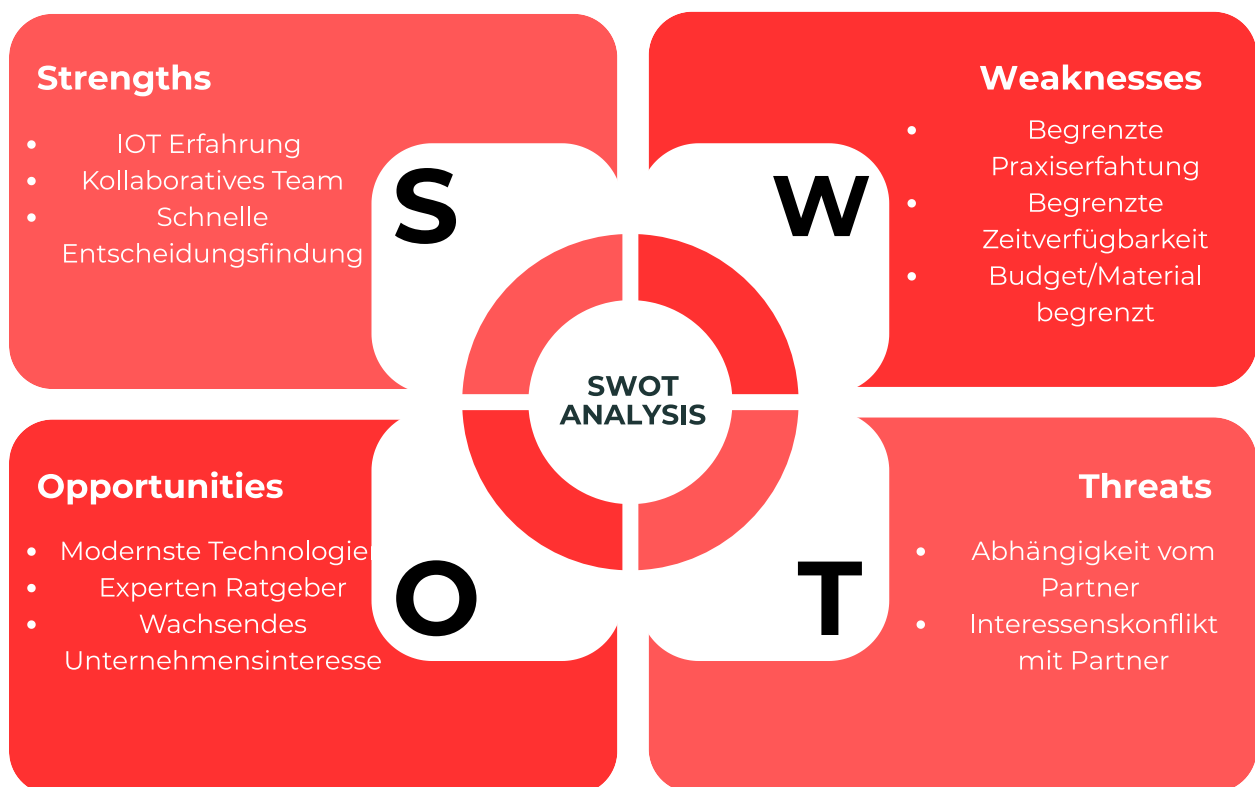


Abb. 13: SWOT-Analyse — Team

Strengths

- **IoT-Erfahrung:** Alle Teammitglieder verfügen aus der HTL-Spengergasse über zwei Jahre Erfahrung in der IoT-Entwicklung.
- **Kollaboratives Team:** Benötigt ein Mitglied eine zweite Meinung oder Starthilfe, stehen andere, in diesem Bereich bewanderte, Teammitglieder bereit.
- **Schnelle Entscheidungsfindung:** Bei offenen Punkten wird ein Meeting einberufen, Vor- und Nachteile werden abgewogen und zeitnah die bestmögliche Entscheidung getroffen.

Weaknesses

- **Begrenzte Praxiserfahrung:** Die Erfahrung an Produkktivsystemen beläuft sich bei jedem Teammitglied auf etwa zwei Monate.
- **Begrenzte Zeitverfügbarkeit:** Durch die laufende Ausbildung an der HTL Spengergasse ist die verfügbare Arbeitszeit eingeschränkt.
- **Begrenzttes Budget/Material:** Begrenzte finanzielle und materielle Ressourcen schränken Investitionen und Skalierung ein.

Opportunities

- **Moderne Technologien:** Mit PostgreSQL, Swift, Next.js etc. arbeiten wir mit modernen Technologien, die eine bestmögliche Benutzererfahrung ermöglichen.
- **Experten-Ratgeber:** Mit Thomas Prikryl (BearingPoint-Entwickler) steht ein erfahrener und kompetenter Ratgeber im Development-Bereich zur Verfügung.
- **Wachsendes Unternehmensinteresse:** Ein erfolgreiches, zufriedenstellendes Produkt öffnet die Tür für zukünftige Zusammenarbeit.

Threats

- **Abhängigkeit vom Partner:** Für die Installation des Mesh-Netzwerks sind wir auf das Bürogebäude von BearingPoint angewiesen; Büroumstellungen oder ein Umzug können zu Konflikten führen.
- **Interessenskonflikte mit dem Partner:** Meinungsverschiedenheiten zu Features oder deren Umsetzung können Risiken für den Projektfortschritt darstellen.

2.8 Earned Value Analysis

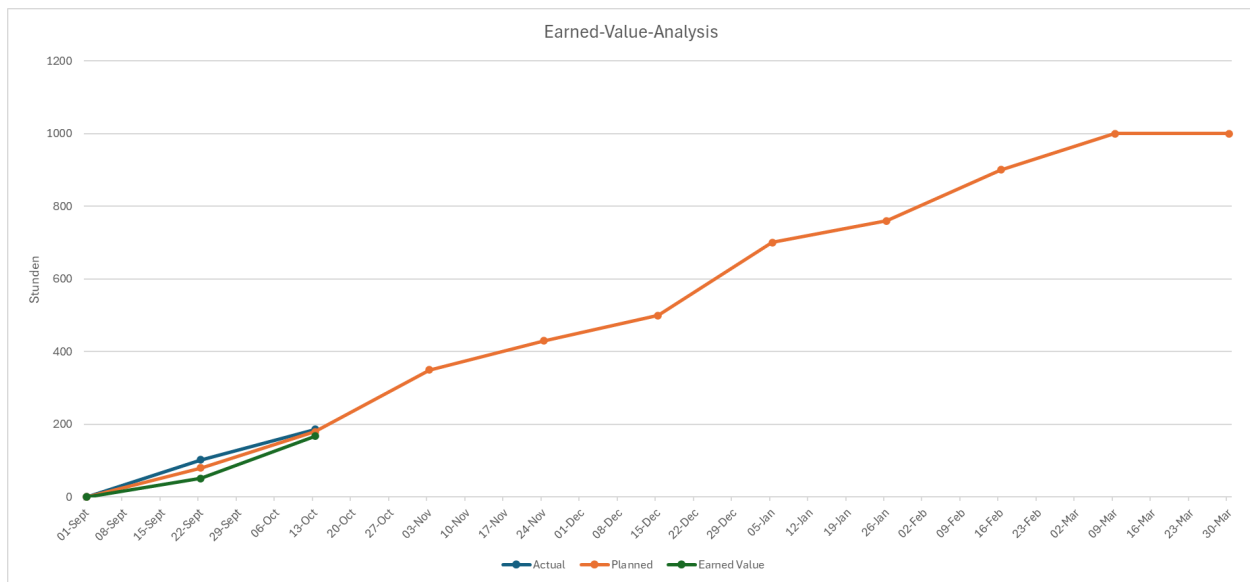


Abb. 14: Darstellung – Earned Value Analysis

2.9 Project Environment Analysis

Extern

- **Partneranforderungen.** BearingPoint definiert spezifische Anforderungen an das Projekt, z. B. den Einsatz bestimmter Sensoren oder technischer Standards. Diese Vorgaben dienen als Grundlage für die technische Umsetzung und werden regelmäßig überprüft, damit die Ergebnisse den Erwartungen des Partners entsprechen.

- **Technologische Abhängigkeiten.** Das Projekt ist auf bestimmte Technologien, Plattformen und Hardwarekomponenten angewiesen, die teilweise von externen Anbietern stammen. Änderungen oder Lieferverzögerungen dieser Abhängigkeiten können den Projektfortschritt beeinflussen und müssen frühzeitig erkannt und eingeplant werden.
- **Beschaffung und (Sonder-)Anfertigung von Sensoren und ESP32-Geräten.** Die Beschaffung und gegebenenfalls Sonderanfertigung von Sensoren sowie ESP32-Geräten kann längere Lieferzeiten mit sich bringen. Diese Aspekte sind frühzeitig in der Projektplanung zu berücksichtigen, um Verzögerungen in der Integrations- und Testphase zu vermeiden.
- **Rechtliche Vorgaben.** Bei der Entwicklung und Nutzung technischer Komponenten sind rechtliche Bestimmungen, insbesondere Urheberrecht und Datenschutz-Grundverordnung (DSGVO), einzuhalten. Dies betrifft sowohl die Nutzung von Softwarebibliotheken als auch den Umgang mit personenbezogenen Daten.

Projektbegleitend

- **Dokumentation & Reporting.** Um Transparenz und Nachvollziehbarkeit zu gewährleisten, wird das Projekt umfassend dokumentiert. Dazu zählen regelmäßige Reports, Protokolle nach Meetings und Fortschrittsberichte, die den Projektverlauf festhalten und Entscheidungen begründen.
- **Feedback.** Regelmäßiges Feedback von BearingPoint und dem schulischen Projektbegleiter dient der Qualitätssicherung und ermöglicht frühzeitige Anpassungen im Entwicklungsprozess.
- **Transparenz im Entwicklungsprozess.** Der Fortschritt wird nachvollziehbar über *Jira*, *GitHub* und regelmäßige Reviews dargestellt.
- **Qualitätsmanagement.** Durch regelmäßige Tests, Code-Reviews und Validierungen wird sichergestellt, dass die Ergebnisse den definierten Anforderungen entsprechen und eine hohe technische sowie funktionale Qualität aufweisen.

Intern

- **Wissenstransfer.** Ein kontinuierlicher Austausch von Fachwissen, Ideen und Erfahrungen zwischen den Teammitgliedern unterstützt den Lernprozess und fördert die Effektivität der Zusammenarbeit. So können Auffälligkeiten schnell geklärt werden.
- **Codequalität.** Der Code wird nach klar definierten Best Practices entwickelt, getestet und dokumentiert. Dies sichert langfristige Wartbarkeit, Stabilität und Nachvollziehbarkeit für zukünftige Projektphasen oder Erweiterungen.
- **Motivation.** Hohe Eigenmotivation und Verantwortungsbewusstsein sind entscheidend für den Projekterfolg. Eigeninitiative, Engagement und Teamgeist helfen, Herausforderungen gemeinsam zu bewältigen.
- **Teamkommunikation.** Strukturierte und regelmäßige Kommunikation über digitale Plattformen wie *Discord* oder *Microsoft Teams* ermöglicht effiziente Abstimmung, schnelle Entscheidungen und fördert ein gemeinsames Verständnis im Team.

2.10 Ceremonies

Wir haben als Team beschlossen, keine zusätzlichen Zeremonien abzuhalten, abseits der zuvor definierten Sprint-Parameter. Sollte jemand ein Meeting verpassen, informiert die betroffene Person das Team so früh wie möglich (z.B. über Discord oder Whatsapp). Verpasste Informationen werden anschließend im Stand Up oder über das Meeting Protokoll nachgeholt. Wenn eine Person krank ist und ihre zugeteilten Aufgaben nicht abschließen kann, werden diese an Teammitglieder verteilt, die das entsprechende Themengebiet gut beherrschen. Wenn Aufgaben keine Dringlichkeit haben dann werden diese auf den nächsten Sprint verschoben.

2.11 Version Management Approach

Der gesamte Code wird mit **Git** auf **GitHub** verwaltet. Die einzelnen Komponenten sind in separaten **Repositories** organisiert.

Branches

- **main**: enthält jederzeit die stabile Release-Version. Auf diesem Branch werden keine direkten Commits ausgeführt, sondern nur Merges von **dev**.
- **dev**: enthält den aktuellen, noch ungetesteten bzw. instabilen Entwicklungsstand und dient als Basis für **feature**.
- **feature**: neue Features werden in diesen Branches pro Jira-Task (z. B. **feature/SPB-123-signin-form**) entwickelt und nach Abschluss der Entwicklung in **dev** gemergt.

3 Attachments

3.1 Compliance Guidelines

Das Projekt ist vollständig **DSGVO-konform** und erfüllt alle Anforderungen an Datenschutz und Datensicherheit. Erfasst werden ausschließlich physikalische Umgebungsdaten (Temperatur, CO₂-Konzentration, Luftfeuchtigkeit, Schalldruckpegel) ohne jeglichen Personenbezug. Die Kommunikation erfolgt ausschließlich innerhalb eines geschlossenen, internetfreien lokalen Netzwerks zwischen Sensorgeräten, Backend und Frontend. Technisch bedingte IP- und MAC-Adressen dienen nur der internen Gerätekommunikation, werden nicht gespeichert. Alle Messwerte werden lokal gespeichert, Benutzerkonten oder personenbezogene Logins existieren nicht. Damit wird höchste Datensouveränität gewährleistet und die Einhaltung der Datenschutz-Grundverordnung sichergestellt.

3.2 Meeting Protocols

Datum	01.09.2025
Teilnehmer intern	Matthias Gregorich, Daniel Nebel, Viktor Novak, Damjan Petrovic, Marc Schneeweis
Teilnehmer extern	Nada Hasewend, Markus Persson, Mladen Stefanovic
Ort	Teams
Ziel	<ul style="list-style-type: none">• Gegenseitiges Kennenlernen• Thema/Idee Pitchen: IOT-Sensor Idee vorstellen und Meinungen einholen
Besprochene Punkte	<ul style="list-style-type: none">• Kommunikationswege festgelegt• IOT-Sensor Projekt fixiert, mit dedizierter Web/Mobile Anzeige
Nächste Schritte	<ul style="list-style-type: none">• Gewünschte Features festlegen auf beiden Seiten• PowerPoint für technische Umsetzung erstellen, für Meeting am 05.09.2025

Tab. 2: Meeting Protokoll vom 01.09.2025

Datum	02.09.2025
Teilnehmer intern	Matthias Gregorich, Daniel Nebel, Viktor Novak, Damjan Petrovic, Marc Schneeweis
Teilnehmer extern	-
Ort	HTL Spengergasse
Ziel	<ul style="list-style-type: none"> • Ziel von Sprint 1 definieren • Tickets/Features definieren • Aufgaben zuteilen
Besprochene Punkte	<ul style="list-style-type: none"> • Setup/Struktur von Jira und Github • Core-Features • Vorstudie/Technologievergleich • Hardwareumsetzung
Nächste Schritte	<ul style="list-style-type: none"> • Features in User Stories festhalten • Sprint 1 Start

Tab. 3: Meeting Protokoll vom 02.09.2025

Datum	05.09.2025
Teilnehmer intern	Matthias Gregorich, Daniel Nebel, Viktor Novak, Damjan Petrovic, Marc Schneeweis
Teilnehmer extern	Nada Hasewend, Markus Persson
Ort	Wiedner Gürtel 13/Turm 24, 1100 Wien/BearingPoint-Büro
Ziel	<ul style="list-style-type: none"> • Gewünschte Features austauschen und finalisieren • Technologien fixieren • Zeitplan absprechen
Besprochene Punkte	<ul style="list-style-type: none"> • Features <ul style="list-style-type: none"> - Raumplanung - Sensor Mesh Integration - Push Nachrichten - Arten der Sensordatenanzeige - Modulare Erweiterung für neue Sensoren/Räume - Widgets • Freiraum/Einstellungsoptionen der Benutzer • Technologische Umsetzung • Kooperationsvertrag
Nächste Schritte	<ul style="list-style-type: none"> • Vorstudien beginnen

Tab. 4: Meeting Protokoll vom 05.09.2025

Datum	25.09.2025
Teilnehmer intern	Matthias Gregorich, Daniel Nebel, Viktor Novak
Teilnehmer extern	-
Ort	HTL Spengergasse
Ziel	<ul style="list-style-type: none"> • Kommunikation unter den Schnittstellen und ESPs klären • Zeitintervall festlegen
Besprochene Punkte	<ul style="list-style-type: none"> • Websocketumsetzung <ul style="list-style-type: none"> - Echtzeitdatenverkehr - Zeitintervalle per Endpoint • Ausfall von ESPs • Datenwiderherstellung • Raumplandarstellung in Backend und Frontend • Hinzufügen neuer ESPs in bestehendes oder neues Mesh ("Pairing-Mode")
Nächste Schritte	<ul style="list-style-type: none"> • UI Mockups • Database Struktur

Tab. 5: Meeting Protokoll vom 25.09.2025

Datum	17.10.2025
Teilnehmer intern	Matthias Gregorich, Daniel Nebel, Viktor Novak, Marc Schneeweis
Teilnehmer extern	-
Ort	Discord
Ziel	<ul style="list-style-type: none"> • Dokumentation aufteilen • Dokumentation Umsetzung
Besprochene Punkte	<ul style="list-style-type: none"> • Project Management Report Aufgaben • Design des Reports • Zuteilung • Deadlines der einzelnen Aufgaben
Nächste Schritte	<ul style="list-style-type: none"> • Dokumentation bearbeiten

Tab. 6: Meeting Protokoll vom 17.10.2025

3.3 Time Records

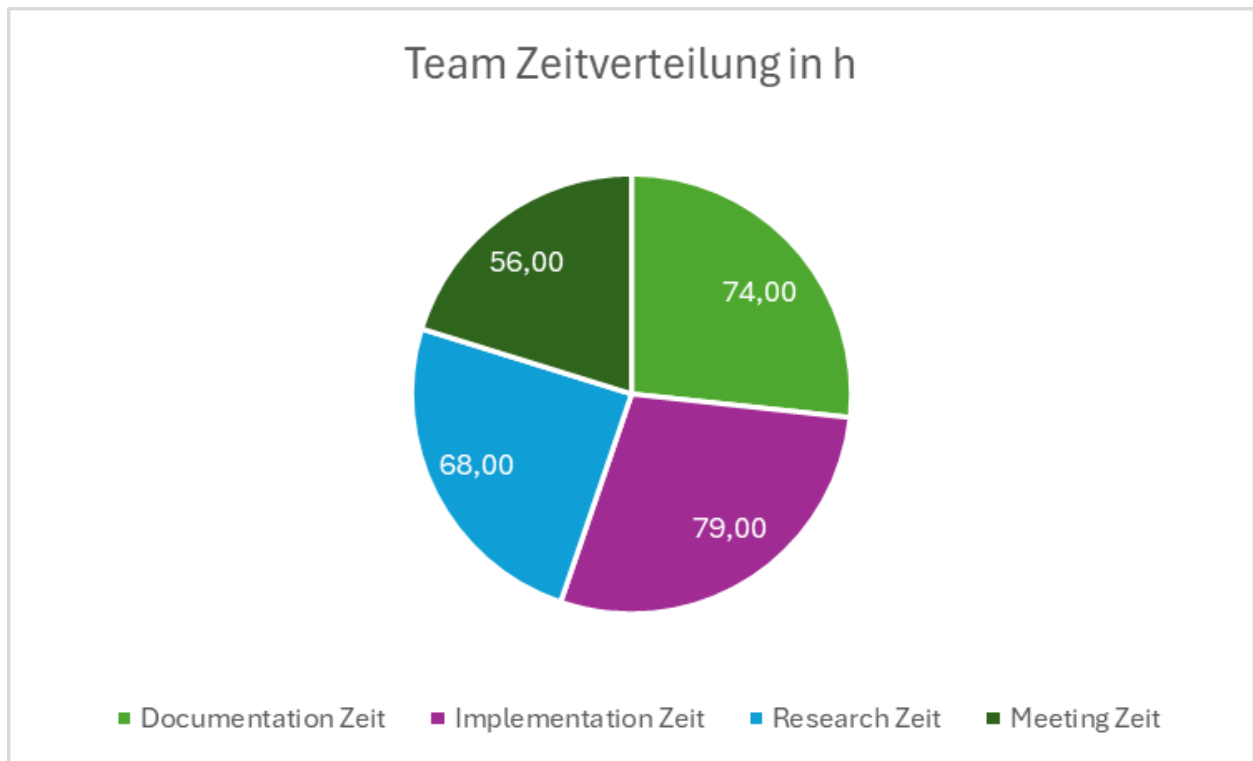


Abb. 15: Team-Zeitdarstellung

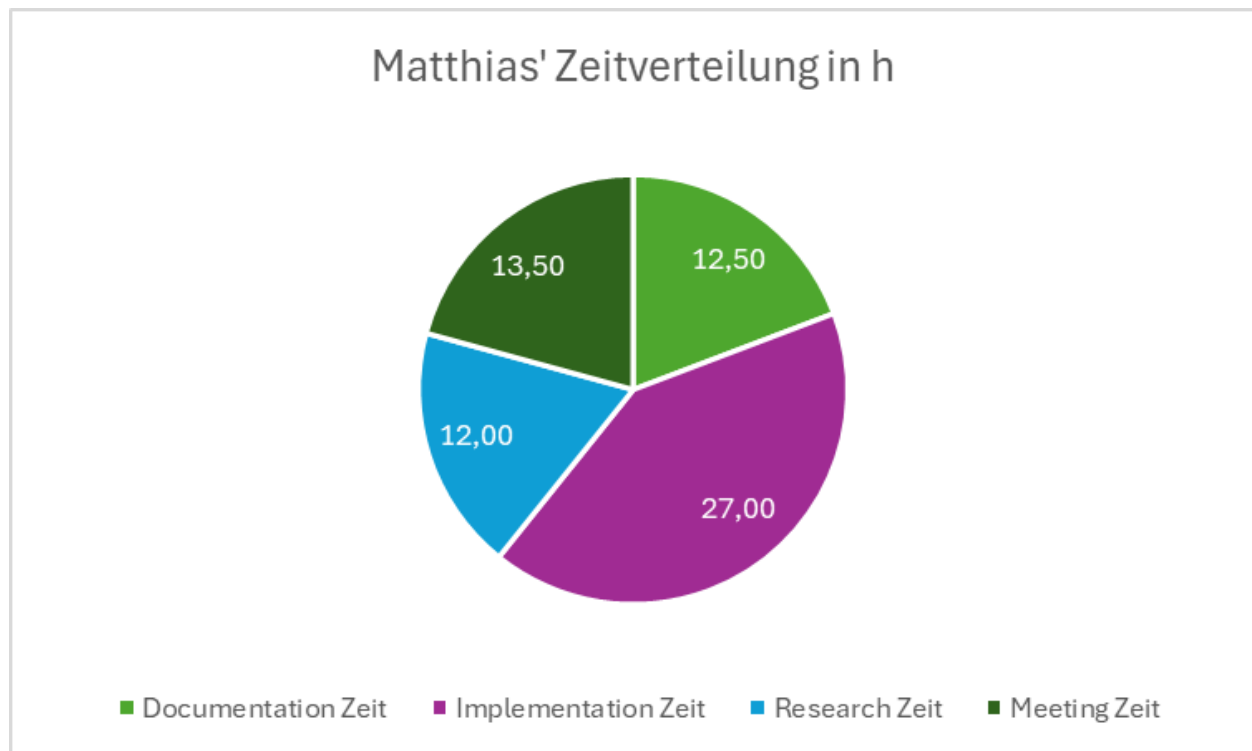


Abb. 16: Zeitverteilung – Matthias Gregorich

Datum	Dauer	Kategorie	Beschreibung
01.09.2025	2:00:00	Meeting	BearingPoint Absprache 1
02.09.2025	2:00:00	Meeting	Unterricht Vorbereitungsphase/Sprint 1 Meeting
03.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
03.09.2025	1:00:00	Projektmanagement	Github aufsetzen
04.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
05.09.2025	3:00:00	Meeting	BearingPoint Absprache 2
10.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
11.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
16.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
17.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
19.09.2025	4:00:00	Implementierung	C# WebSocket Implementation
19.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
20.09.2025	5:00:00	Implementierung	ESP32 WebSocket Client Test mit C# Backend
22.09.2025	3:00:00	Projektmanagement	Jira-Setup und erste User-Stories erstellt
23.09.2025	2:00:00	Implementierung	ESP32 WIFI-AP Firmware
23.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
24.09.2025	3:00:00	Implementierung	ESP32 WIFI-AP DHCP + Testing
25.09.2025	2:00:00	Meeting	Umsetzung von Techstack
26.09.2025	2:30:00	Projektmanagement	Jira Board Besprechung
02.10.2025	2:00:00	Implementierung	TimescaleDB Setup + Testing
03.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
04.10.2025	2:00:00	Implementierung	First Entity definition + First Endpoints Created
07.10.2025	3:00:00	Implementierung	Unterricht Vorstudie, Development
10.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
14.10.2025	2:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	2:30:00	Projektmanagement	Jira Tasks (Techstack, Architecture)
16.10.2025	2:30:00	Projektmanagement	Jira Tasks (KPIs, Version Mangement)
17.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	2:00:00	Meeting	Playbook Besprechung
20.10.2025	3:30:00	Projektmanagement	Jira Tasks (Techstack, Architecture) überarbeitet
Summe	65:00:00		

Tab. 7: Zeitaufzeichnung – Matthias Gregorich

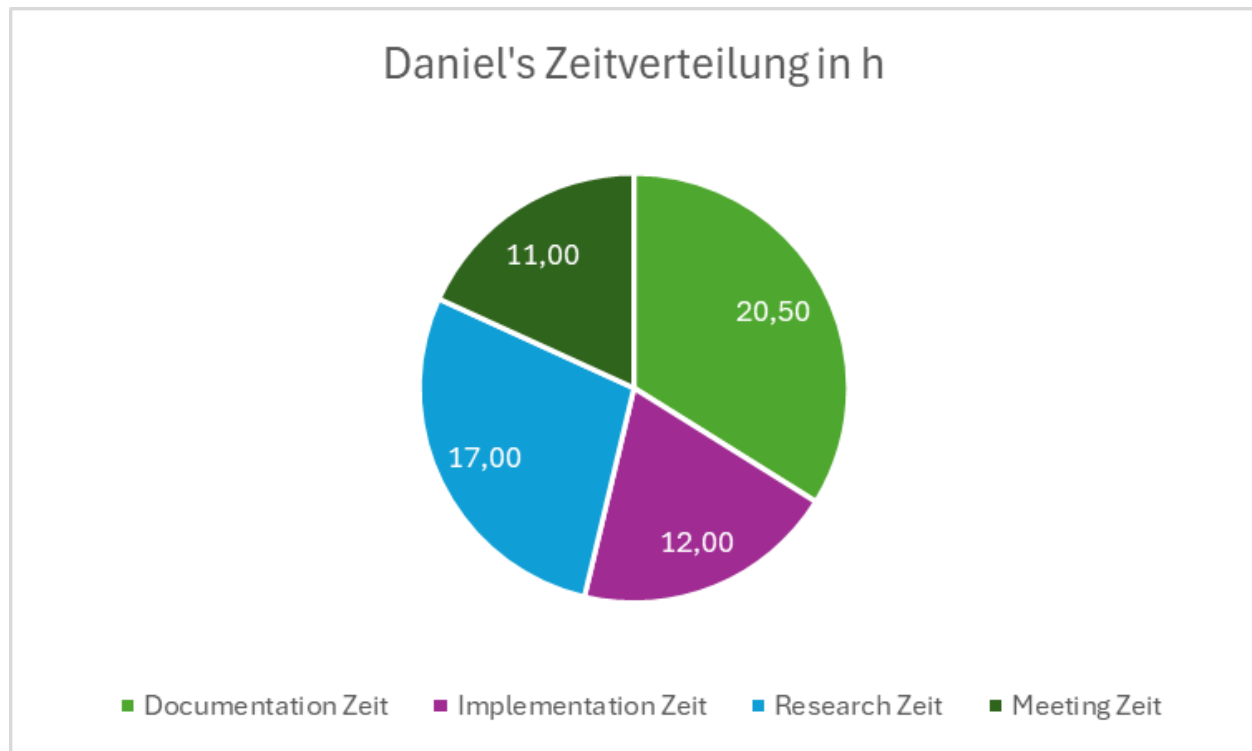


Abb. 17: Zeitverteilung – Daniel Nebel

Datum	Dauer	Kategorie	Beschreibung
01.09.2025	2:00:00	Meeting	BearingPoint Absprache 1
02.09.2025	2:00:00	Meeting	Unterricht Vorbereitungsphase/Sprint 1 Meeting
03.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
03.09.2025	1:00:00	Projektmanagement	Github aufsetzen
04.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
05.09.2025	3:00:00	Meeting	BearingPoint Absprache 2
10.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
10.09.2025	3:00:00	Research	Swift-LiDAR/ARKit
11.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
16.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
17.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
19.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
22.09.2025	3:00:00	Projektmanagement	Jira-Setup und erste User-Stories erstellt
23.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
25.09.2025	2:00:00	Meeting	Umsetzung von Techstack
26.09.2025	2:30:00	Projektmanagement	Jira Board Besprechung
30.09.2025	2:30:00	Research	Playbook (SWOT, Canva Visio, Stakeholder, PE)
03.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
05.10.2025	3:00:00	Implementierung	Figma Mockup
07.10.2025	3:00:00	Implementierung	Unterricht Vorstudie, Development
10.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
14.10.2025	2:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	2:00:00	Meeting	Playbook Besprechung
18.10.2025	3:00:00	Projektmanagement	Jira Tasks (Responsibilities, SWOT, Project Environment Analysis)
19.10.2025	3:00:00	Projektmanagement	Jira Tasks (Stakeholder, Deadlines)
20.10.2025	5:00:00	Projektmanagement	Meetingprotokolle aufbereiten, Projekt-Logo, Playbook schreiben
20.10.2025	3:00:00	Projektmanagement	Post Kickoff Präsentation
Summe	60:30:00		

Tab. 8: Zeitaufzeichnung – Daniel Nebel

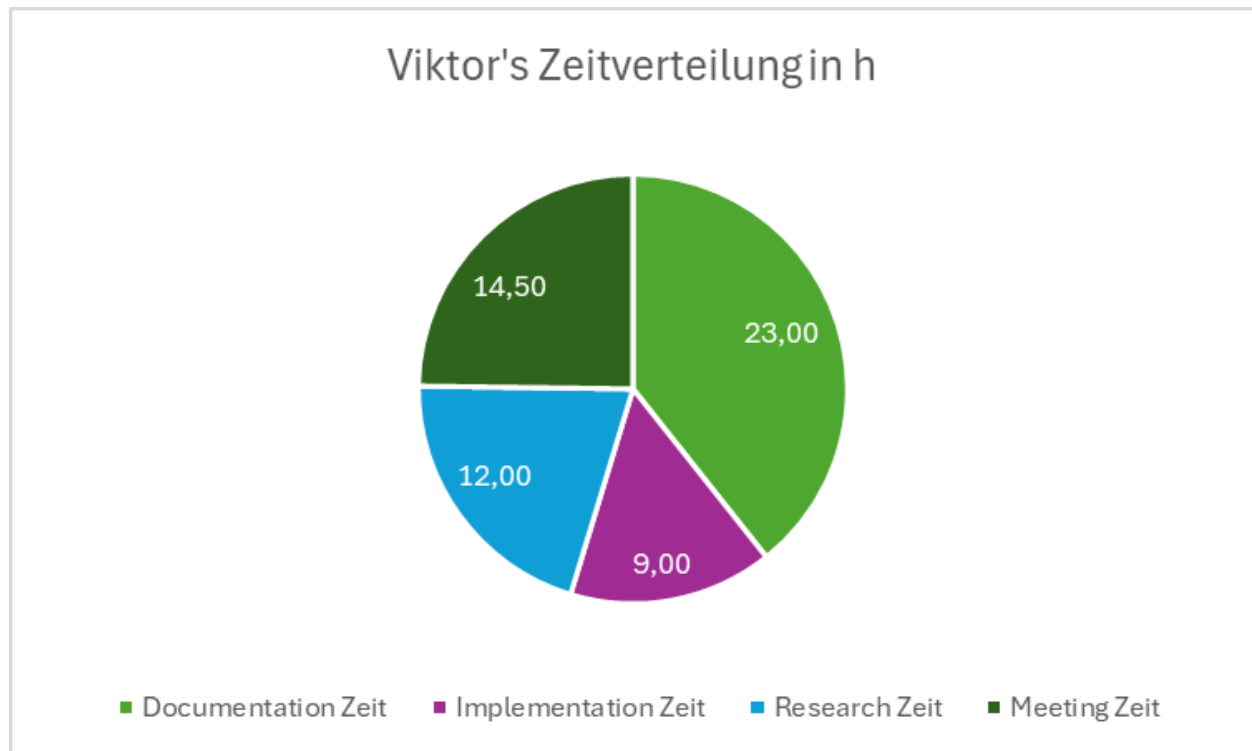


Abb. 18: Zeitverteilung – Viktor Novak

Datum	Dauer	Kategorie	Beschreibung
01.09.2025	2:00:00	Meeting	BearingPoint Absprache 1
02.09.2025	2:00:00	Meeting	Unterricht Vorbereitungsphase/Sprint 1 Meeting
03.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
03.09.2025	1:00:00	Projektmanagement	Github aufsetzen
04.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
05.09.2025	3:00:00	Meeting	BearingPoint Absprache 2
10.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
11.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
16.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
17.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
19.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
23.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
25.09.2025	2:00:00	Meeting	Umsetzung von Techstack
26.09.2025	2:30:00	Projektmanagement	Jira Board Besprechung
27.09.2025	1:00:00	Research	Umsetzung von Techstack
03.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
07.10.2025	3:00:00	Implementierung	Unterricht Vorstudie, Development
10.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
14.10.2025	2:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	2:30:00	Projektmanagement	Jira Tasks (Techstack, Architecture)
16.10.2025	2:30:00	Projektmanagement	Jira Tasks (KPIs, Version Mangement)
17.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	2:00:00	Meeting	Playbook Besprechung
18.10.2025	8:00:00	Projektmanagement	Playbook schreiben
20.10.2025	6:00:00	Projektmanagement	Playbook schreiben
21.10.2025	2:30:00	Projektmanagement	Playbook schreiben
Summe	61:00:00		

Tab. 9: Zeitaufzeichnung – Viktor Novak

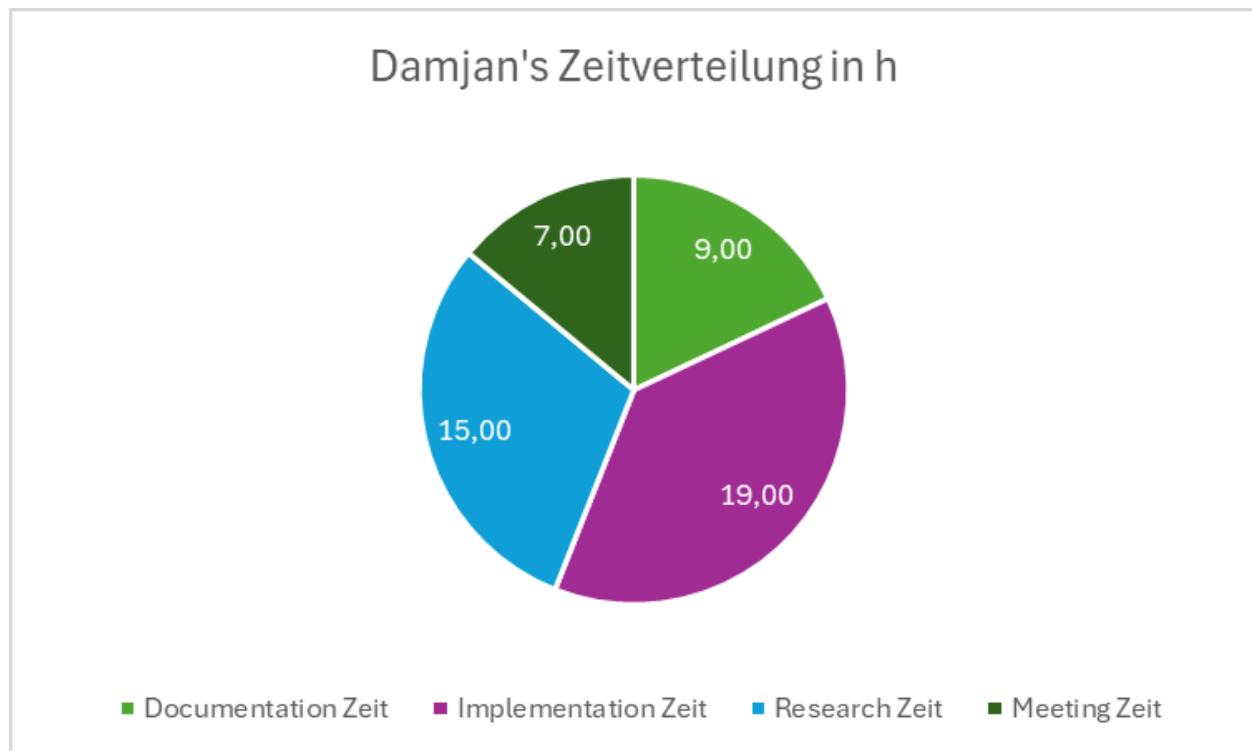


Abb. 19: Zeitverteilung – Damjan Petrovic

Datum	Dauer	Kategorie	Beschreibung
01.09.2025	2:00:00	Meeting	BearingPoint Absprache 1
02.09.2025	2:00:00	Meeting	Unterricht Vorbereitungsphase/Sprint 1 Meeting
03.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
03.09.2025	1:00:00	Projektmanagement	Github aufsetzen
04.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
05.09.2025	3:00:00	Meeting	BearingPoint Absprache 2
10.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
11.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
16.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
17.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
19.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
22.09.2025	3:00:00	Research	Vorstudie Sensoren
23.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
23.09.2025	3:00:00	Implementierung	Planung Sensorgehäuse
27.09.2025	2:00:00	Implementierung	Planung Sensorgehäuse
03.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
03.10.2025	3:00:00	Implementierung	ESP32 Sensor Chip Test
07.10.2025	3:00:00	Implementierung	Unterricht Vorstudie, Development
10.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
14.10.2025	2:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	2:00:00	Implementierung	ESP32 Sensor Chip Test
19.10.2025	3:00:00	Projektmanagement	Jira Tasks (Ressources Stored, Requirements Definition, Collaboration Approach)
20.10.2025	2:00:00	Projektmanagement	Jira Tasks (Ressources Stored, Requirements Definition, Collaboration Approach)
20.10.2025	3:00:00	Projektmanagement	Post Kickoff Präsentation
Summe	50:00:00		

Tab. 10: Zeitaufzeichnung – Damjan Petrovic

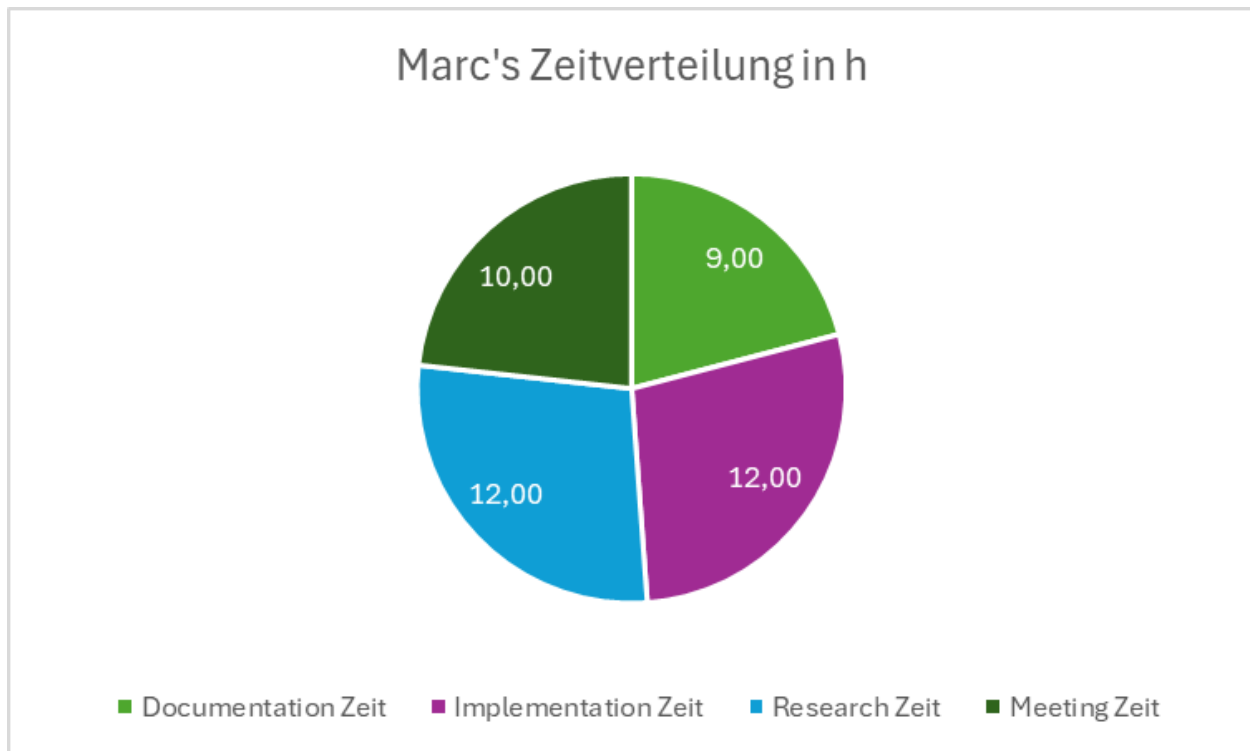


Abb. 20: Zeitverteilung – Marc Schneeweis

Datum	Dauer	Kategorie	Beschreibung
01.09.2025	2:00:00	Meeting	BearingPoint Absprache 1
02.09.2025	2:00:00	Meeting	Unterricht Vorbereitungsphase/Sprint 1 Meeting
03.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
03.09.2025	1:00:00	Projektmanagement	Github aufsetzen
04.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
05.09.2025	3:00:00	Meeting	BearingPoint Absprache 2
10.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
11.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
16.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
17.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
19.09.2025	1:00:00	Research	Unterricht Vorbereitungsphase
23.09.2025	2:00:00	Research	Unterricht Vorbereitungsphase
27.09.2025	1:00:00	Research	Umsetzung von Techstack
03.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
05.10.2025	3:00:00	Implementierung	Figma Mockup
07.10.2025	3:00:00	Implementierung	Unterricht Vorstudie, Development
10.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
14.10.2025	2:00:00	Implementierung	Unterricht Vorstudie, Development
15.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	1:00:00	Implementierung	Unterricht Vorstudie, Development
17.10.2025	2:00:00	Meeting	Playbook Besprechung
19.10.2025	2:00:00	Projektmanagement	Jira Tasks (Project Health, Ceremonies und Definition of Done)
20.10.2025	3:00:00	Projektmanagement	Jira Tasks (Sprintparameter)
20.10.2025	3:00:00	Projektmanagement	Post Kickoff Präsentation
21.10.2025	1:30:00	Projektmanagement	Playbook schreiben
Summe	44:30:00		

Tab. 11: Zeitaufzeichnung – Marc Schneeweis

3.4 Cooperation Contract

KOOPERATIONSVEREINBARUNG

zwischen

1. BearingPoint GmbH
Wiedner Gürtel 13/Turm 24, 1100 Wien

(Name und Adresse des Unternehmens)

(in der Folge „**der Projektpartner, die Projektpartnerin**“)

und

2. Damjan PETROVIC

Daniel NEBEL

Matthias GREGORICH

Marc SCHNEEWEIS

Viktor NOVAK

(Namen der Schüler/Schülerinnen)

(in der Folge „**das Projektteam**“)

PRÄAMBEL

Das Projektteam und der Projektpartner/die Projektpartnerin beabsichtigen gemäß der Prüfungsordnung BMHS, BGBl II Nr. 177/2012 i.d.g.F., die Planung und Durchführung eines Diplomprojektes mit dem Titel:

Sensorbasiertes Monitoring von Raumumgebung

welches die Erstellung von Arbeitsergebnissen, zum Thema des Diplomprojekts, zum Gegenstand hat.

Durch die Zusammenarbeit soll insbesondere den Mitgliedern des Projektteams die Möglichkeit eingeräumt werden, im Rahmen ihrer schulischen Ausbildung bei der Durchführung eines Diplomprojektes an die Verhältnisse im technischen Berufsleben herangeführt zu werden, um dabei die in der Schule erworbenen theoretischen Kenntnisse und Fähigkeiten in der Praxis anzuwenden bzw. zu erweitern. Hingewiesen wird in diesem Zusammenhang auf den unentgeltlichen Charakter dieser Vereinbarung.

§1

Gegenstand

Gegenstand ist die Erstellung von Arbeitsergebnissen zum Thema des Diplomprojekts. Der Projektpartner / die Projektpartnerin wird jedoch darauf hingewiesen, dass es sich um ein Projekt im Zusammenhang mit der schulischen Ausbildung handelt und daher jede Haftung des Projektteams, insbesondere in Hinsicht auf die Unentgeltlichkeit des Vertrages, ausgeschlossen ist.

§2

Laufzeit

Die vorliegende Kooperation tritt am **01.09.2025** in Kraft und wird bis zum schulrechtlich verordneten Termin am **08.04.2026** abgeschlossen.

§ 3

Rechte und Pflichten des Projektteams

Das Projektteam verpflichtet sich, die im Gegenstand genannten Arbeiten sorgfältig und unter möglicher Schonung der Interessen des Projektpartners / der Projektpartnerin durchzuführen.

Das Projektteam verpflichtet sich zur Geheimhaltung aller ihm zur Kenntnis gelangenden Geschäfts- und Betriebsgeheimnisse gegenüber Dritten. Ausnahmen siehe §5 Einsicht und Präsentation.

§4

Rechte und Pflichten des Projektpartners / der Projektpartnerin

Der Projektpartner / die Projektpartnerin verpflichtet sich, das Projektteam in einem zumutbaren Rahmen zu unterstützen und dem Projektteam folgende Hilfsmittel zeitgerecht und für die Dauer des Diplomprojektes kostenfrei zur Verfügung zu stellen:

- Fachliche Betreuung
- Testmöglichkeiten im Büro nach Vereinbarung mit dem Projektpartner / der Projektpartnerin

Sofern der Projektpartner / die Projektpartnerin dem Projektteam urheberrechtlich geschütztes Material oder Daten zur Verfügung stellt, stellt der Projektpartner / die Projektpartnerin sicher, dass dieses Material frei von Rechten Dritter ist. Der Projektpartner / die Projektpartnerin hält das Projektteam diesbezüglich schad- und klaglos.

Sollte das Projektteam im Rahmen dieser Kooperationsvereinbarung ein Werk schaffen, dem Schutz im Sinne des Urheberrechtsgesetzes zukommt, hat der Projektpartner / die Projektpartnerin uneingeschränkte Nutzungs- und Verwertungsrechte. Das Projektteam verpflichtet sich in diesem Zusammenhang keine Ansprüche bezüglich einer Vergütung zu erheben.

§5

Einsicht und Präsentation

Da die Tätigkeit des Projektteams auch Inhalt bzw. Grundlage der an der Schule HTBLuVA Wien V, Spengergasse 20 zu erstellenden Diplomarbeit ist, berechtigt der Projektpartner / die Projektpartnerin die zuständigen Organe des Bundes zur Einsicht und Kontrolle, um die Aufgaben gem. Prüfungsordnung BMHS, SchUG bzw. SchUG BKV zu erfüllen. Das Projektteam ist auch berechtigt, Ergebnisse der Diplomarbeit im Rahmen des Schulunterrichts und bei Schul- und schulbezogenen Veranstaltungen, sowie bei der Präsentation und Diskussion der Diplomarbeit zu verwenden.

§6

Änderungen

Änderungen dieser Vereinbarung bedürfen der Schriftform. Sollte ein Schüler / eine Schülerin, der / die Mitglied des Projektteams ist, während der Laufzeit dieser Vereinbarung aus der HTBLuVA Wien V, Spengergasse 20 ausscheiden (Abmeldung vom Schulbesuch), bleibt die Kooperationsvereinbarung für die verbleibenden Unterzeichner, mit Rücksichtnahme auf eine etwaige Reduktion des Projektumfanges, aufrecht.

Sep 18, 2025

(Ort, Datum)

18-09-2025 | 2:24:37 PM BST

(Ort, Datum)

Damjan Petrovic

Damjan Petrovic (Sep 18, 2025 16:37:45 GMT+2)

(für das Projektteam)

Signiert von:

[Signature]

38845122487A4DF...

(für den/die Projektpartner/in)

Signiert von:



3.5 Legal Declaration

Erklärung

Die unterfertigten Kandidaten/Kandidatinnen haben gemäß den geltenden schulrechtlichen Bestimmungen die Ausarbeitung einer abschließenden Arbeit (Diplomarbeit bzw. Abschlussarbeit) mit folgender Aufgabenstellung gewählt:

Sensorbasiertes Monitoring von Raumumgebungen

Individuelle Aufgabenstellungen im Rahmen des Gesamtprojektes:

- Viktor Novak (5DHIF): **Implementierung von automatisierten Testverfahren und einer Electron-Desktop-App.**
- Daniel Nebel (5DHIF): **Implementierung einer mobilen App zur Anzeige von Echtzeit- und historischen Daten sowie zur Benachrichtigung bei kritischen Messwerten.**
- Damjan Petrovic (5DHIF): **Konstruktion und Programmierung von Netzwerk- und Sensorkomponenten.**
- Marc Schneeweis (5DHIF): **Entwicklung eines benutzerfreundlichen webbasierten Dashboards zur Visualisierung von Live-Daten, Statistiken und Warnmeldungen.**
- Matthias Gregorich (5DHIF): **Planung und Umsetzung einer Infrastruktur und Schnittstellen zur Verarbeitung und Analyse von Daten.**

Die Kandidaten/Kandidatinnen nehmen zur Kenntnis, dass die abschließende Arbeit in eigenständiger Weise und außerhalb des Unterrichtes zu bearbeiten und anzufertigen ist, wobei Ergebnisse des Unterrichtes mit einbezogen werden können, die jedenfalls als solche entsprechend kenntlich zu machen sind.

Die Abgabe der vollständigen abschließenden Arbeit hat in digitaler und in zweifach ausgedruckter Form bis spätestens **08.04.2026** beim zuständigen Betreuer/der zuständigen Betreuerin zu erfolgen.

Die Kandidaten/Kandidatinnen nehmen auch zur Kenntnis, dass ein Abbruch der abschließenden Arbeit nicht möglich ist.

Kandidaten/Kandidatinnen:

**Datum und Unterschrift bzw.
Handysignatur:**

Viktor Novak (5DHIF)

 18.09.2025

Daniel Nebel (5DHIF)

Daniel Nebel 18.09.25

Damjan Petrovic (5DHIF)

Damjan Petrovic 18.09.25

Marc Schneeweis (5DHIF)

Marc Schneeweis 18.09.25

Matthias Gregorich (5DHIF)

Matthias Gregorich 18.09.25

3.6 Software Requirements Specification

Software Requirements Specification

für

SensorBear

Version 1.0

Vorbereitet von Damjan Petrovic

Inhaltsverzeichnis

1. EINLEITUNG	3
1.1 ZWECK	3
1.2 DOKUMENTKONVENTIONEN	3
1.3 ZIELGRUPPE UND LESEREIHENFOLGE	3
1.4 PRODUKTUMFANG	3
1.5 REFERENZEN	3
2. GESAMTE BESCHREIBUNG	3
2.1 PRODUKTPERSPEKTIVE	3
2.2 HAUPTFUNKTIONEN	4
2.3 BENUTZERKLASSEN UND EIGENSCHAFTEN	4
2.4 BETRIEBSUMGEBUNG	4
2.5 DESIGN- UND IMPLEMENTIERUNGS-CONSTRAINTS	4
2.6 BENUTZERDOKUMENTATION	4
2.7 ANNAHMEN UND ABHÄNGIGKEITEN	4
3. EXTERNE SCHNITTSTELLENANFORDERUNGEN	5
3.1 BENUTZERSCHNITTSTELLEN	5
3.2 HARDWARE-SCHNITTSTELLEN	5
3.3 SOFTWARE-SCHNITTSTELLEN	5
3.4 KOMMUNIKATIONSSCHNITTSTELLEN	5
4. SYSTEMFUNKTIONEN	5
4.1 SYSTEM FEATURE: DATENERFASSUNG	5
4.2 SYSTEM FEATURE: DATENVISUALISIERUNG	5
4.3 SYSTEM FEATURE: ADMINISTRATION	5
4.4 SYSTEM FEATURE: BENACHRICHTIGUNGEN	5
5. WEITERE NICHT-FUNKTIONALE ANFORDERUNGEN	6
5.1 LEISTUNGSANFORDERUNGEN	6
5.2 SICHERHEITSANFORDERUNGEN	6
5.3 SICHERHEITSANFORDERUNGEN (SAFETY)	6
5.4 QUALITÄTSATTRIBUTE	6
5.5 GESCHÄFTSREGELN	6
6. WEITERE ANFORDERUNGEN	6
ANHANG A: GLOSSAR	6

1. Einleitung

1.1 Zweck

Dieses Dokument beschreibt die funktionalen und nicht-funktionalen Anforderungen des sensorbasierten Monitoring-Systems. Ziel ist es, eine präzise Grundlage für Implementierung, Validierung und Wartung des Systems bereitzustellen. Es richtet sich an Entwickler, Tester und Projektbeteiligte.

1.2 Dokumentkonventionen

Dieses Dokument folgt der IEEE 830-Struktur für Software Requirements Specifications. Jede Anforderung erhält eine eindeutige Kennung (z. B. REQ-1).

1.3 Zielgruppe und Lesereihenfolge

Entwickler: zur Implementierung und Schnittstellendefinition

Projektleitung / Lehrer: zur Fortschrittskontrolle und Validierung

Tester: zur Erstellung von Testfällen basierend auf funktionalen Anforderungen

Empfohlene Lesereihenfolge: Kapitel 1–2 für Überblick, Kapitel 3–4 für Funktionalität, Kapitel 5 für Qualitätsanforderungen.

1.4 Produktumfang

Das System dient der kontinuierlichen Erfassung und Visualisierung von Umweltdaten (CO₂, Temperatur, Luftfeuchtigkeit, Lautstärke) in Innenräumen. Jeder ESP32-Sensor sendet Messwerte in festgelegten Intervallen über WLAN oder ein Mesh-Netzwerk an eine zentrale API-Datenbank. Ein Dashboard visualisiert Echtzeit- und Verlaufsdaten, zeigt Warnungen bei Grenzwertüberschreitungen und unterstützt Administratoren bei der Sensorverwaltung.

1.5 Referenzen

IEEE 830-1998 – IEEE Recommended Practice for Software Requirements Specifications

ESP32 Documentation (Espressif Systems)

Next.js / Material UI Developer Guide

PostgreSQL / TimescaleDB Documentation

2. Gesamte Beschreibung

2.1 Produktperspektive

Das Monitoring-System ist ein eigenständiges IoT-System mit drei Hauptkomponenten:

- Sensor Nodes (ESP32)
- Backend (API, PostgreSQL/TimescaleDB)
- Frontend (Next.js Dashboard / Swift / Electron)

Datenfluss: Sensor → API → Datenbank → Dashboard

2.2 Hauptfunktionen

Periodische Erfassung von Sensordaten
Speicherung in zentraler Datenbank
Visualisierung und Verlaufsgrafiken
Grenzwertwarnungen
Verwaltung von Sensoren und Räumen
Exportfunktionen (CSV/JSON)

2.3 Benutzerklassen und Eigenschaften

Admin: Konfiguriert Sensoren, Räume, Grenzwerte

User: Beobachtet Echtzeit- und Verlaufsdaten

2.4 Betriebsumgebung

Sensoren: ESP32 Microcontroller (Arduino / C++)

Server: Node.js API (Express), PostgreSQL / TimescaleDB

Frontend: Next.js Webapp (Material UI)

Kommunikation: WLAN / Mesh Netzwerk, HTTPS-Protokoll

2.5 Design- und Implementierungs-Constraints

Kommunikation ausschließlich über WLAN oder Mesh (kein Ethernet)
Energieversorgung über USB
Hardwarebudget auf Prototyp-Niveau begrenzt
Datenübertragung ≤ 10 s Intervalle
Verwendung von Open-Source-Technologien

2.6 Benutzerdokumentation

Online-Benutzerhandbuch (PDF)
Setup-Anleitung für Sensoren
Administrator-Handbuch

2.7 Annahmen und Abhängigkeiten

Stabile WLAN-Verbindung erforderlich
ESP32-Module verfügen über gültige Firmware
Server ist dauerhaft erreichbar
Browser-kompatibel mit Chrome / Safari / Edge

3. Externe Schnittstellenanforderungen

3.1 Benutzerschnittstellen

Dashboard mit responsivem Design, Anzeige von Live-Daten, Diagrammen und Warnmeldungen. Funktionen: Sensorverwaltung, Export, Benachrichtigungen.

3.2 Hardware-Schnittstellen

ESP32 Sensoren mit CO₂-, Temperatur-, Luftfeuchtigkeits- und Lautstärkesensoren. Kommunikation über I²C, UART oder analoge Pins.

3.3 Software-Schnittstellen

API (REST/JSON) ↔ Frontend

API ↔ PostgreSQL / TimescaleDB

3.4 Kommunikationsschnittstellen

HTTPS / WLAN / Mesh Netzwerk

Datenformat: JSON

Verschlüsselung: TLS 1.3

4. Systemfunktionen

4.1 System Feature: Datenerfassung

REQ-1: System misst alle 10 Sekunden*

REQ-2: Sensoren übertragen Messwerte

REQ-3: Mesh heilt sich selbst bei Ausfall von Sensor

4.2 System Feature: Datenvisualisierung

REQ-4: Dashboard zeigt Echtzeit- und Verlaufsdaten

REQ-5: Warnungen bei CO₂ > 1000 ppm* oder Lautstärke > 70 dB*

REQ-6: Benutzer können Zeiträume auswählen

4.3 System Feature: Administration

REQ-7: Admins verwalten Sensoren und Räume

REQ-8: Grenzwerte und Intervalle anpassbar

REQ-9: Datenexport als CSV/JSON

4.4 System Feature: Benachrichtigungen

REQ-10: Push- oder E-Mail-Benachrichtigungen bei Grenzwertüberschreitung

*Diese Standardwerte können vom Benutzer angepasst werden

5. Weitere nicht-funktionale Anforderungen

5.1 Leistungsanforderungen:

Datenübertragung ≤ 10 s, Dashboard ≤ 2 s, max. 50 Sensoren gleichzeitig.

5.2 Sicherheitsanforderungen:

Authentifizierung, HTTPS

5.3 Sicherheitsanforderungen (Safety):

Keine physischen Schäden, sichere Fehlermeldungen.

5.4 Qualitätsattribute:

Zuverlässigkeit ≥ 95 %, Wartbarkeit (GitHub), Benutzerfreundlichkeit, Portabilität.

5.5 Geschäftsregeln:

Nur Admins dürfen Sensoren ändern.

6. Weitere Anforderungen

Wöchentliche Backups auf GitHub und OneDrive, versionierte Dokumentation.

Zukunft: VOC- und Feinstaubsensoren, RFID-Login, Statistikfunktionen.

Anhang A: Glossar

ESP32 – Mikrocontroller mit integrierten WLAN und Bluetooth Fähigkeiten

ppm – Parts per Million (Maßeinheit CO₂)

Mesh-Netzwerk – dezentrale Netzwerktopologie

API – Schnittstelle zwischen Sensoren, Datenbank und Frontend