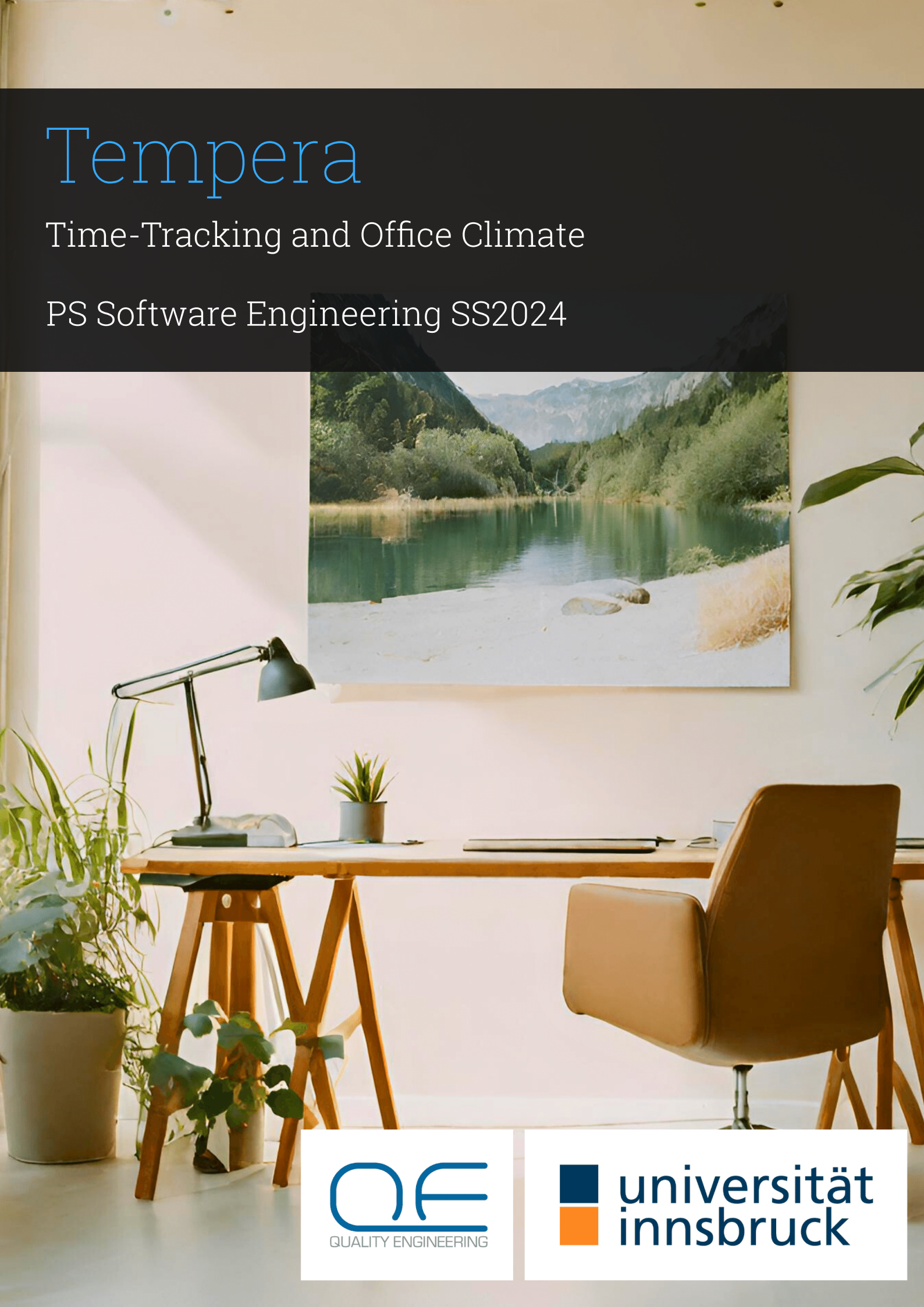
Tempera

Time-Tracking and Office Climate

PS Software Engineering SS2024



QUALITY ENGINEERING

universität innsbruck

Tempera

Time-Tracking and Office Climate

Lehrveranstaltungsleiter:	Dornauer Benedikt Jäger Alexandra Kelter Christopher Mussmann Andrea
Zeitraum:	März, 2024 - Juni, 2024
Fakultät:	Informatik, Quality Engineering

universität innsbruck

Unser Ziel

Klassisches Time-Tracking am Arbeitsplatz kann umständlich sein, da Zeitaufzeichnungen oft manuell mitprotokolliert werden, und sich dabei leicht der ein oder andere Fehler einschleichen kann. Dieses Problem hat auch die Führungsebene der (fiktiven Firma) Guerilla GmbH wahrgenommen. Zum Glück sollte es bald eine neue Lösung geben, die das bisherige Problem löst und darüber hinaus neue Funktionalitäten implementiert, um das Raumklima am Arbeitsplatz (Temperatur, Luftqualität, Lichtintensität,...) zu verbessern. Das Stichwort lautet Tempera , die neue innovative Büroassistenz am Arbeitsplatz.

Abbildung 1: Timemanagement am Arbeitsplatz

Produktidee

In den meisten Fällen sind Mitarbeiter:innen dazu angehalten, ihre Arbeitszeiten, sowie ihre erledigten Aufgaben zu Protokoll zu bringen. Klassisch kann dies mit Hilfe von Listen passieren, wo Arbeitsbeginn, Arbeitsende, Pausen, sowie ein kurzer Tätigkeitsbericht verfasst werden. Hier können jedoch Ungenauigkeiten auftreten, die Nachvollziehbarkeit mangelhaft, bzw. die Aktualität der Daten nicht gegeben sein.

Abhilfe schaffen moderne Arbeitszeitmesser, bekannt auch unter dem Begriff „Stempeluhr“. Sie protokollieren Arbeitsbeginn, Pausen und das Arbeitsende von Arbeitnehmern, in dem diese ein- und ausstempeln. Die Arbeitszeitmesser bieten den Vorteil minutengenau zu dokumentieren, und die Möglichkeit, Arbeitszeiten der Mitarbeiter:innen immer aktuell überblicken zu können. Guerilla GmbH möchte das Konzept der Arbeitszeiterfassung mit *Tempera* auf ein „neues Level heben“.

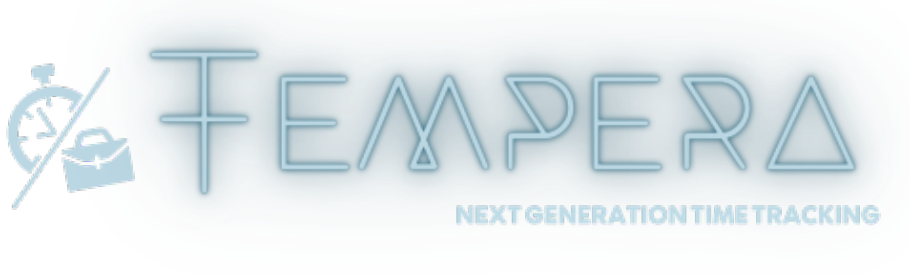


Abbildung 2: Logo von *Tempera*

Bei *Tempera* handelt es sich um ein stromsparendes SmartDevice, das Arbeitnehmer:innen direkt an ihrem Arbeitsplatz abstellen. Anstelle des Stempelns kann über Buttons der:die Arbeitnehmer:in zwischen unterschiedliche Arbeitsmodi wählen. Über eine Schnittstelle werden die Daten gesammelt und über eine Webanwendung zur Einsicht bereitgestellt. Ein Feature ist hier, dass durch Klicken eines Buttons am *Tempera* Gerät auch direkt Projektaufwände protokolliert werden können.

Auf dem Markt gibt es bereits ähnliche Konzepte. Mit einem weiteren Aspekt, neben einer guten Nutzerfreundlichkeit und zahlreichen, aber simplen Funktionalitäten, möchte sich *Tempera* von der Konkurrenz abheben. *Tempera* hat sich nämlich zu Aufgabe gesetzt, maßgeblich das Raumklima am Arbeitsplatz zu verbessern. Ein angenehmes Raumklima spielt eine bedeutende Rolle für das Wohlbefinden der Mitarbeiter:innen und beeinflusst maßgeblich ihre Effizienz bei der Arbeit, indem es eine Atmosphäre schafft, in der sie sich wohl fühlen, sich besser konzentrieren können und somit produktiver sind. Deshalb hat *Tempera* Raumklimasensoren angebracht, die kontinuierlich Raumklimadaten erheben. Diese Daten werden unter anderem genutzt, um den Mitarbeiter:innen Vorschläge zur Verbesserung des Raumklimas im Büro zu machen.

Inhaltsverzeichnis

1

Projektbeschreibung

Bei *Tempera* handelt es sich um ein SmartDevice, bestehend aus einem Arduino, einer Steuereinheit mit vier Tasten, und einzelnen Sensoren. Die *Tempera* -Einheit wird an jedem Arbeitsplatz platziert, dort von der ihr zugewiesenen Person gesteuert und nimmt fortlaufend sowohl Zeit- als auch Raumklimamessungen vor. Mit Hilfe dieser Hardware werden zwei existierende Lösungen kombiniert und erweitert:

1. **Statusanzeige:** Durch Drücken eines der vier Buttons kann in einen von vier Zustands-Modi gewechselt werden. Sind Arbeitende am Arbeitsplatz (gekennzeichnet durch Raumnummer, Vorname und Nachname des Arbeitnehmers) und können sie prinzipiell gestört werden, wählen sie den *Anwesend* - Modus; haben sie ein Meeting, können sie in den *Meeting* - Modus wechseln. Während Phasen intensiven Arbeitens am Arbeitsplatz während dessen sie nicht gestört werden wollen, wählen sie den *Deep Work* - Modus. In der Mittagspause, bzw. am Ende des Arbeitstags wechseln sie in den *Out-of-Office* - Modus. Der jeweilig aktive Modus wird durch eine LED angezeigt.

   Statusänderungen durch Drücken eines Buttons werden vom System gespeichert, ebenso wie die Zeit, die zwischen zwei Statusänderungen vergangen ist. Zusätzlich können User:innen durch wiederholtes Drücken des selben Buttons Timestamps innerhalb eines Status erstellen und so größere Zeitblöcke innerhalb eines Status in kleiner Blöcke unterteilen (z.B. wenn während einer *Deep Work* Session zwischen Projekten gewechselt wird). In regelmäßigen Abständen überträgt *Tempera* diese Daten an einen Accesspoint, der wiederum eine Verbindung zu einem Webserver hat, wo die dazugehörige Webanwendung gehostet ist. Hier stehen Arbeitnehmer:innen nach erfolgreichem Login jeder aufgezeichnete Status sowie die Zeitmarkierungen innerhalb eines Status als Zeitaufzeichnung zu Verfügung. Diese aufgezeichnete Arbeitszeit, die Anwesend, in Meetings oder mit Deep Work verbracht wurde, kann nun den verschiedenen Projekten zugeordnet werden, denen die jeweiligen User:innen zugeteilt sind. Es ist den User:innen auch möglich, sich den Status ihrer Kolleg:innen anzeigen zu lassen und so z.B. unnötige Gänge zum Büro vermeiden, wenn die gewünschte Person im *Out-of-Office* bzw. *Deep Work* - Modus ist.

   Ergänzend können Leiter:innen einer Gruppe (*Group Leader*) die Arbeitsaktivitäten ihrer Mitarbeiter:innen überblicken. Personen wie CEO, CTO, ... (*Manager*) können Zugriffsrechte auf mehrere Gruppen bekommen und bestimmte gruppenbezogenen Daten einsehen (siehe 1.2).
2. **Funktionalitäten einer Raumklimastation:** In einem Bericht der Bundesanstalt für Arbeitsschutz und Arbeitsmedizin [1], werden einige zentralen Faktoren für ein gesundes Raumklima genannt, denen sich *Tempera* annimmt:

   - **Raumtemperatur**: Ein angenehmes Raumklima hängt stark davon ab, wie warm oder kalt es in den Räumlichkeiten ist. Abweichungen sowohl darüber als auch darunter können zu einem Unbehaglichkeitsempfinden führen.

- **Luftfeuchtigkeit**: Hohe Luftfeuchten beeinträchtigen die Fähigkeit des Menschen, durch Schwitzen Wärme abzugeben. Kreislaufbelastung ist häufig die Folge.
- **Luftqualität**: Beeinträchtigungen der Konzentrations- und Leistungsfähigkeit
- **Lichtintensität**: Durch eine schlechte Beleuchtung wird die Sehkraft stark beeinflusst, und führt zu einer kognitiven Anstrengung.

Mit der Hilfe integrierter Sensoren werden die beschriebenen Parameter erhoben und ebenfalls über den Accesspoint an den Webserver geschickt. Mit Hilfe der Daten soll eine Raumklima-Optimierung erzielt werden. Ein Früherkennungssystem informiert betroffene User via E-Mail und in der Webapp rechtzeitig, wenn es zu möglichen Abweichungen kommt. Zum Beispiel warnt es, wenn die Temperatur eine Minimum/Maximum Schwelle erreicht oder eine schlechte Luftqualität vorherrscht.

Das Beispiel in Abbildung 1.1 stellt dar, wie *Tempera* in einem Office-Setting zum Einsatz kommen kann. Im Nachfolgenden wird nun insbesondere auf die *Hardware* eingegangen, sowie die Ansprüche an *Qualitätseigenschaften* des Projekts erörtert.

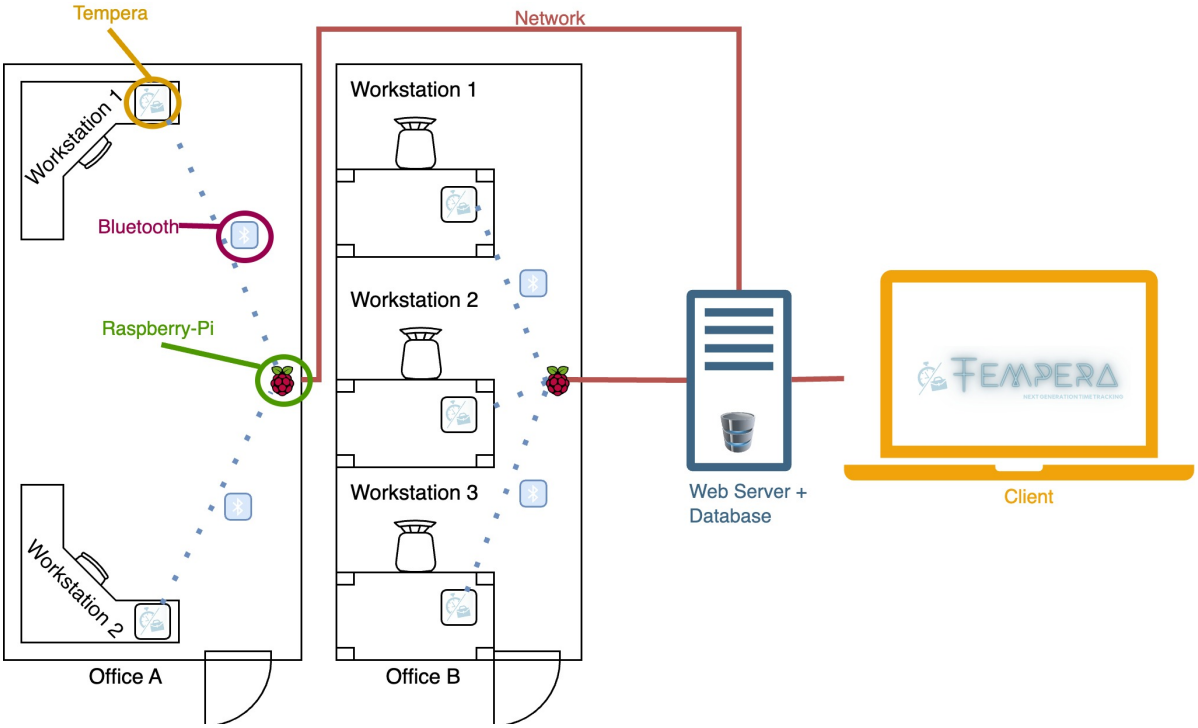


Abbildung 1.1: Sample setup.

1.1. Hardware

Folgende Hardware wird für die Umsetzung des Projekts benötigt und von Ihrem Team von Guerilla GmbH zur Verfügung gestellt, aufgeschlüsselt im Nachfolgenden.

1.1.1. Sensoren und Arduino

Eine einzelne *Tempera* - Messstation besteht aus folgenden Hardwareteilen:

- Ein **Luftsensor** (BME688), der Luftfeuchtigkeit, Temperatur und Luftqualität misst. Es sei hierbei angemerkt, dass die Luftqualität mittels eines VOC (Volatile Organic Compound)-Sensors, welcher die Menge an flüchtigen organischen Verbindungen erkennt, gemessen wird.

- Ein **Lichtsensor** (Fototransistor), der die Lichtintensität misst.
- Vier **physische Buttons**
- Eine **RGB-LED** ist eine LED, die in verschiedenen Farben und Intensitäten leuchten kann. Hinter der Bezeichnung RGB verbergen sich die Farben *rot*, *grün* und *blau*, die individuell kombiniert werden können. Mit dieser RGB-LED sollen der jeweilig aktive Modus, in dem sich die *Tempera* Station befindet, angezeigt werden (Anwesend, Out-of-Office, Meeting, Deep Work). Sowohl die Modi, als auch die dafür gewählten Farben soll fix am Arduino hinterlegt werden.

Die Stationen speichern immer die aktuellsten Werte der Sensoren in einem flüchtigen Speicher (Volatile Memory). Da gewisse Sensoren zum Rauschen (Varianz in den Messungen) neigen, soll darauf geachtet werden, dass die Daten zwischen den Messintervallen akkumuliert werden. Zum Übertragen der Daten an den Accesspoint, umgesetzt mit Hilfe eines *Raspberry Pi*, wird das **BLE** (Bluetooth Low Energy) Protokoll verwendet.

Jede *Tempera* Station hat eine eigene, eindeutige, ID, die bei der Erstkonfiguration des *Tempera* (Arduinos) von den Administratoren festgelegt wird (hardcoded). Diese IDs sollen sowohl in der Länge als auch der Zusammensetzung zur eindeutigen Identifikation einer *Tempera* Station im System geeignet sein. Beachten Sie hier neben der Eindeutigkeit auch, dass diese ID geeignet sein muss, mittels BLE übertragen zu werden (z.B. bereits in den Advertising Paketen der Station). Bedenken Sie bei der Wahl einer geeigneten Methode zur Generierung der IDs auch, dass während des Proseminars Stationen anderer Teams aktiv sind, die ebenfalls IDs aussenden. Die ID-Generierungsstrategie muss in der Webapp implementiert werden (Details siehe 1.3.1).

Bluetooth LE ist ein gängiges Protokoll. Accesspoints sollten BLE-Geräte als *Tempera* Stationen erkennen können, ohne bereits eine Verbindung zu dem BLE-Gerät aufgebaut haben zu müssen. Überlegen Sie sich hier ebenfalls eine geeignete Strategie, *Tempera* Stationen in Ihrem System als solche erkennbar zu machen. Diese Information kann z.B. in den von Ihnen gewählten *Tempera* IDs enthalten sein.

Eine kurze Simulation mit *TinkerCAD* gibt einen Einblick, wie der *Tempera* Prototyp ausschauen könnten. **Dieses Video dient nur zu Demonstrationszwecken und kann *NICHT* als Vorlage für den für das Projekt zu erstellenden Schaltplan verwendet werden.** Der Prototyp unterscheidet sich vom verlangten Endprodukt unter anderem dadurch, dass andere Hardwarekomponenten verwendet werden und einige Funktionalitäten nicht abgebildet sind. Er gibt jedoch einen Eindruck wieder, wie *Tempera* - Einheit umgesetzt werden könnte: `https://youtu.be/rBR98e9qmPM`.

1.1.2. Accesspoint

Der Accesspoint, der vorerst auf einem Raspberry Pi basiert, erfordert von *Tempera* die Nutzung des Betriebssystems RaspberryPi-OS[1] und die Entwicklung der benötigten Accesspoint-Software in Python. Die Accesspoint-Software ist zudem headless.

Auf dem Accesspoint befindet sich eine Datenbank, die die Daten der Sensorstationen zeitlich begrenzt persistent speichert. In einem über eine Konfigurationsdatei namens `conf.yaml` konfigurierbaren Intervall sendet der Accesspoint diese Daten an den Webserver und löscht sie aus der Datenbank, wenn die Übermittlung erfolgreich war.

Das Hinzufügen einer neuen *Tempera* Station zu einem Accesspoint erfolgt über die Weboberfläche des Webservers. Dort erstellen Administratoren neue Sensorstationen und fügen sie genau einem Accesspoint hinzu. Nur dieser Accesspoint darf die Daten der Sensorstation auslesen und an die Webapp übertragen. Stellen Sie sicher, dass ein Accesspoint zu jeder Zeit weiß, mit welchen Sensorstationen er sich verbinden darf.

Um sicherzustellen, dass sich der Accesspoint nur mit einem einzigen Webserver verbindet, wird die IP-Adresse des Webservers ebenfalls in die `conf.yaml` Konfigurationsdatei gespeichert. Accesspoints haben außerdem eine eindeutige ID, die in der Webapp vergeben wird und ebenfalls in der `conf.yaml` Konfigurationsdatei hinterlegt werden muss.

[1] `https://www.raspberrypi.com/software/`

Ein von Ihnen bereitgestelltes Skript, das durch Ausführen von `./configure` aktiviert wird, ermöglicht die Setzung sämtlicher Einstellungen während der Installation der Accesspoints. Zusammengefasst sollen die folgenden Einstellungen durch das Skript und die Konfigurationsdatei einstellbar sein:

1. Das Übertragungsintervall zwischen Accesspoint und Webserver.
2. Die Adresse des Webservers.
3. Eine eindeutige, geeignet generierte ID.

1.2. Webserver und Webapp

Die Webanwendung bietet einerseits Funktionalität an, die gemessenen Klimadaten der einzelnen Sensorstationen durch geeignete Darstellung auszuwerten und bei Grenzwertüberschreitung die betroffenen User:innen über die Webapp selbst, sowie via E-Mail darüber zu informieren. Andererseits ermöglicht sie es, die Zeit, die die Mitarbeiter:innen in einem der vier oben definierten Arbeits-Modi verbringen, für eine detailierte Arbeitszeitaufzeichnung zu verwenden. Dazu können User:innen über geeignete Ansichten aufgezeichnete Zeiträume einzelnen Projekten zuordnen und eine genaue Tätigkeitsbeschreibung verfassen.

Die Basis für die Umsetzung der Webanwendung ist ein Java Spring Boot Skeleton Projekt, das Ihrem Team von Guerilla GmbH zu Verfügung gestellt wird. Die Daten der Anwender, sämtliche Sensordaten und Zeitaufzeichnungen sollen einheitlich und persistent in einer Postgres-Datenbank gespeichert werden. Die Einstellungen für die Datenbankverbindung müssen konfigurierbar sein. Für die finale Projektversion darf kein externer Datenbankserver o.ä. verwendet werden.

Die Webanwendung, die auf einem Webserver gehostet ist, wertet die gesammelten Messwerte der *Tempera* Stationen aus und bereitet sie in geeigneten Ansichten in visueller Darstellung auf (Graphen, Charts, etc).

Alle visuellen Auswertungen in Graphen, Charts etc. sollen übersichtlich gestaltet sein, Möglichkeiten zur Filterung bieten und auf benutzerdefinierte Zeiträume beschränkt werden können. Außerdem sollen Anwender:innen die Möglichkeit haben, sich tabellarische Auflistungen der einzelnen Daten (Sensorwerte und Arbeitszeit) anzeigen zu lassen. Alle Auflistungen müssen ebenso filter- und sortierbar sein.

Beachten Sie, dass die Klimadaten der *Tempera* Stationen als persönliche Daten angesehen werden müssen und nur der Person über die Webapp zugänglich sein dürfen, die der jeweiligen *Tempera* Station zugeordnet ist. Ein umfassenderes Konzept zum Umgang mit diesen Messdaten wird in weiteren Entwicklungsschritten von einem Expertenteam verfasst werden (außerhalb dieses Projekts).

Die detailierte Arbeitszeiterfassung via der *Tempera* Station sowie im Webapp (siehe unten) muss ebenfalls als sensible Daten erachtet werden. Im Detail darf sie auch nur der Person zu Verfügung stehen, die der jeweiligen *Tempera* Station zugeordnet ist. Kumulierte Ansichten der Arbeitszeit sollen jedoch, wie unten im Detail beschrieben, auch bestimmten anderen Personen im System zugänglich sein.

1.2.1. Rollen

Folgende Rollen sollten in der Webanwendung umgesetzt sehen:

Employee

Employee bilden in der Hierarchie die niedrigste Instanz. Jedem Mitarbeitenden ist genau eine *Tempera* Einheit zugewiesen. Loggt sich ein:e Mitarbeiter:in in die Webanwendug ein, kann diese:r auf genau ihre Daten zugreifen: (1) Büroarbeitszeiten und (2) die Klimadaten der Messstation. Bei (1) den Büroarbeitszeiten können den Arbeitszeiten (außer dem Modus „Out-of-Office “) Projekte zugeteilt werden und Beschreibungen, was in diesem Zeitraum erledigt wurde, erfasst werden. User:innen

können außerdem Zeitblöcke in mehrere, kleinere Zeitblöcke aufteilen und jeden der neu entstandenen Zeitblöcke einem Projekt zuordnen.

Jede:r Mitarbeiter:in hat die Möglichkeit, in unterschiedlichen Ansichten und über konfigurierbare Zeitintervalle Überblick über die geleisteten Arbeitsstunden sowohl im Bezug auf die unterschiedlichen Projekte, denen der Mitarbeiter:in zugeteilt ist, als auch den verschiedenen Statusinformationen aus der persönlichen Sensorstation zu erhalten (z.B. wie viele Stunden habe ich für Projekt A gearbeitet? Wie viele Stunden habe ich im Status *Meeting* verbracht? Wie viele Stunden, die ich für Projekt A gearbeitet habe, habe ich im Status *Meeting* verbracht?).

Um die Zuordnung zu erleichtern soll es die Möglichkeit geben, ein Default-Projekt einzustellen, dem standardmäßig alle Arbeitszeitmessungen hinzugefügt werden.

Die Klimadaten der persönlichen Sensorstation können generell nur eingesehen werden.

Ergänzend steht allen Mitarbeiter:innen eine vereinfachte Darstellung zur Verfügung, in denen sie den Status der Kollegen:innen einsehen können: Name, Arbeitsplatz und Aktivität. Diese Funktionalität wird durch eine geeignete Suchfunktion unterstützt.

Mitarbeiter:innen haben die Möglichkeit, den eigenen Status vom Default *Public* auf *Private* (Status sichtbar für alle Personen in der selben Gruppe) oder *Hidden* (eigener Status wird im System nicht mehr angezeigt) zu setzen.

Groupleader

Groupleader verwalten Gruppenzugehörigkeit und fügen ihre Mitarbeiter:innen Projekten hinzu bzw. entfernen sie wieder. User:innen können sowohl mehreren Gruppen als auch mehreren Projekten innerhalb der Gruppen angehören.

Groupleader haben Einblick in die kummulierten Arbeitszeitdaten ihrer Mitarbeiter:innen. Dabei soll kein Rückschluss auf einzelne Mitarbeiter:innen mehr möglich sein, die Zuteilung der Arbeitszeiten zu verschiedenen Statusinformationen muss aber erhalten bleiben.

Ähnlich zur Auswertung der persönlichen Arbeitszeit sollen Groupleader die Arbeitszeitverteilung ihrer Gruppen in den unterschiedlichen Status sowohl gruppenübergreiffend, als auch nach einzelnen Gruppen gefiltert in geeigneten visuellen und tabellarischen Darstellungen ansehen können.

Manager

Manager können Groupleader ernennen. Sie verwalten Projekte (erstellen, editieren, löschen) und vergeben Projekte an Gruppen. Ein Projekt besteht zumindest aus ID, Name und Beschreibung. Sie verwalten (erstellen, editieren, löschen) zudem Gruppen.

Manager haben Einblick in die kummulierten Arbeitszeitdaten ihrer Projekte. Dabei soll kein Rückschluss auf einzelne Mitarbeiter:innen mehr möglich sein, wohl aber auf die Beiträge der einzelnen Gruppen, die am Projekt arbeiten. Außerdem muss die Zuteilung der Arbeitszeiten zu verschiedenen Statusinformationen erhalten bleiben.

Ähnlich zur Auswertung der kummulierten Gruppenarbeitszeit sollen Manager die Arbeitszeitverteilung ihrer Projekte in den unterschiedlichen Status und auf Gruppen verteilt in geeigneten visuellen und tabellarischen Darstellungen ansehen können, sowie passende Filter für entsprechende Teilansichten (sowohl Projekte und Gruppen, als auch Zeiträume) zu Verfügung haben.

Administrator

Administrators können Nutzer:innen hinzufügen, bearbeiten und entfernen. Sie können außerdem die Rechte aller anderen Nutzer:innen verwalten und Rollen zuweisen. Sie verwalten (erstellen, bearbeiten, löschen) zusätzlich auch die Räume.

Administrators sind für die Verwaltung der Accesspoints und der *Tempera* Stationen verantwortlich. Neue Accesspoints und Stationen werden von Administrators in der Webapp angelegt (siehe 1.3.1); Administrators weisen jedem Accesspoint einen Raum, jeder *Tempera* Station einen User zu.

Sie können Accesspoints und Stationen sperren und haben außerdem die Möglichkeit, Accesspoints und *Tempera* Stationen permanent zu entfernen (eine neuerliche Verbindung eines Accesspoints bzw. einer Station mit einer gesperrten ID soll vom Webapp automatisch abgelehnt werden).

Sie konfigurieren darüberhinaus über die Webapp auch die Grenzwerte der Sensorstationen einheitlich für jeden Raum. Diese Grenzwerte werden (nur) in der Webapp hinterlegt und NICHT an den Accesspoint übertragen. Bei jeder Änderung der Grenzwerte müssen Administrators zusätzlich zu den geänderten Grenzwerten die Änderung auch begründen; jede Grenzwertänderung muss außerdem im Auditlog gespeichert werden.

Administratoren verwalten (erstellen, bearbeiten, löschen) außerdem kurze hilfreichen Tipps (Fließtext), was Mitarbeiter:innen bei Grenzwertüberschreitungen tun können, um die jeweiligen Raumklimawerte wieder in akzeptable Bereiche zu bringen.

Administratoren können ihre Statusanzeige nicht vom Default (*Public*) abändern.

1.3. Messdaten und Grenzwerte

Die Messdaten aller Sensoren sowie die Statusdaten mit Zeitintervallen, die temporär in den Accesspoints hinterlegt sind, werden zur permanenten Speicherung auf den zentralen Webserver übertragen. Der Webserver muss eine geeignete REST-Schnittstelle zur Verfügung stellen, über die Sensor- und Statusdaten von den Accesspoints an den Webserver übertragen werden können.

Bei Über- oder Unterschreitungen der von den Administratoren eingestellten Grenzwerten für die Sensoren sollen betroffene User:innen darüber sowohl geeignet über die Webapp als auch per E-Mail verständigt werden. Überlegen Sie sich, wie Sie feststellen können, ob die Warnungen zur Grenzwertüberschreitungen tatsächlich gesehen werden (entweder im Mail, oder in der Webapp) und stellen Sie sicher, dass die Überschreitungswarnungen bis zu diesem Zeitpunkt in der Webapp sichtbar sind.

Beachten Sie auch, dass Grenzwertüberschreitungen nicht unbedingt mit einer von Mitarbeiter:innen gesetzten Handlung sofort behoben sind (z.B. dauert es nach Öffnen eines Fensters, bis sich die Raumluft bessert). Überlegen Sie sich, wie Sie damit umgehen und wann die nächste Warnung ergehen soll. Mitarbeiter:innen sollen auch Möglichkeiten zur Verbesserung der jeweiligen Situation vorgeschlagen werden. Überlegen Sie sich, wie diese am besten verwaltet und kommuniziert werden können.

Auch gute Raumklimawerte sollen kommuniziert werden. Prinzipiell sollte jede Person, die über eine *Tempera* Station verfügt, über die Webapp jederzeit mit einem Blick sehen können, wie gut das momentane Raumklima ist.

1.3.1. Accesspoints und Tempera Stationen

Accesspoints und *Tempera* Stationen müssen von Administrators angelegt werden und in der Webapp bereits existieren, bevor sie Verbindungen aufbauen bzw. Daten senden können. Welche Accesspoints sich mit welchen Sensorstationen verbinden sollen, wird ebenfalls von Administrators im Webapp festgelegt. Zusätzlich müssen sowohl Accesspoints als auch *Tempera* Stationen *enabled* bzw. *disabled* werden können. Bei der Erstkonfiguration eines Accesspoints oder einer *Tempera* Station müssen Administrators das Gerät explizit in den Status *enabled* setzen, um Verbindungsaufbau und Datenübertragung zum Server zu Ermöglichen.

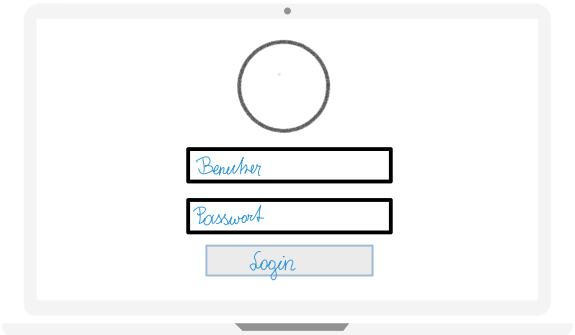
Unter welchen Umständen ein Accesspoint eine Verbindung zum Server aufbauen, bzw. Daten einer bestimmten *Tempera* Station auf den Server übertragen darf, hängt also von verschiedenen Faktoren ab. So soll z.B. jede Verbindung eines Accesspoints vom Webapp abgelehnt werden, dessen ID nicht in der

Datenbank zu finden ist, oder der im Status *disabled* ist. Ebenso dürfen Accesspoints keine Daten von Sensorstationen abfragen, die entweder gar nicht am Server existieren, oder einem anderen Accesspoint zugeordnet sind. Auch dürfen keine Daten einer *disabled Tempera* Station auf den Server übertragen werden.

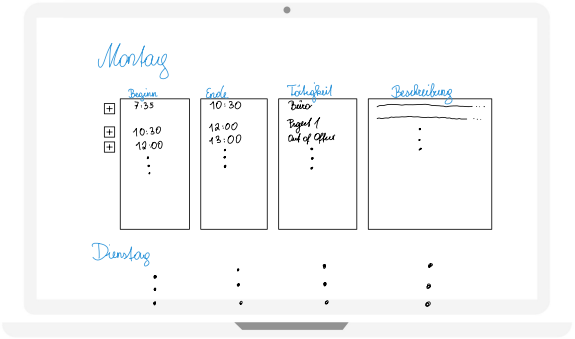
Damit jeder Accesspoint und jede Sensorstation eindeutig einem Objekt am Server zuordenbar ist, müssen eindeutige IDs vergeben werden (siehe auch 1.1.1 und 1.1.2). Die IDs sollen am Server generiert und verwaltet werden und beim Anlegen neuer Accesspoint-Objekte bzw. Sensorstation-Objekte vom System vorbefüllt werden. Überlegen Sie sich sowohl für die Generierung der Sensorstations-IDs, als auch der Accesspoint-IDs geeignete Strategien, um Eindeutigkeit im System zu garantieren und dokumentieren Sie diese geeignet.

1.3.2. Benutzeroberfläche

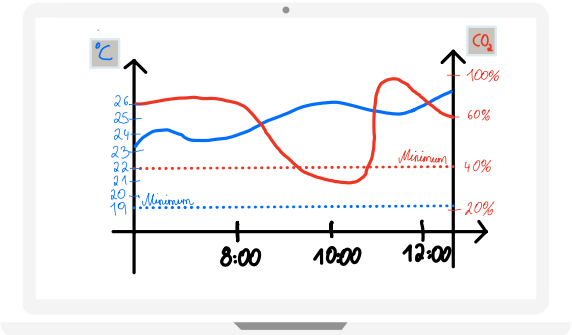
Die Benutzeroberflächengestaltung ist generell Ihnen überlassen. Eine kurze Rapid-Prototyping-Phase des Guerilla-Teams haben zu folgenden Beispielen geführt, welche in Abbildung 1.2 dargestellt sind und können als eine Art Orientierung gesehen werden.



(a) Beispiel einer Login-Seite.



(b)) Beispiel einer Übersichtsseite für zur Einsicht der eigenen Arbeitszeiten. Statusblocks können in kleinere Einheiten aufgeteilt und Projekten zugeordnet werden. Beschreibungen können individuell ergänzt werden.



(c) Dieses Beispiel zeigt die Darstellung von den zeitlichen Verlauf der Temperatur, als auch die Entwicklung der Luftqualität.

Abbildung 1.2: Rapid-Prototyping Beispieldarstellung.

1.4. Qualitätsansprüche an das Projekt

Das Management der Guerilla GmbH hat einige Qualitätsmerkmale für die Prototypentwicklung identifiziert, die insbesondere für das Projekt berücksichtigt werden sollten.

1.4.1. Sicherheit

Damit Sie während der Entwicklung sicher stellen können, dass Sie nur Daten von von Ihnen aufgestellten *Tempera* Stationen erhalten, müssen Accesspoints und *Tempera* Stationen von Administrators im System angelegt werden, bevor Verbindungen mit dem Server zugelassen werden. Außerdem müssen alle Accesspoints und *Tempera* Stationen, die im System angelegt werden, von Administrators aktiviert werden (*enabled*), bevor Datenübertragung an den Server stattfinden darf. Sowohl alle Accesspoints, als auch alle *Tempera* Stationen, die im System registriert sind, können jederzeit von Administrators deaktiviert (*disabled*) werden. Deaktivierte Accesspoints dürfen keine Daten an den Server übertragen; es dürfen auch keine Daten von deaktivierten *Tempera* Stationen auf den Server übertragen werden. (Bereits bestehende Daten einer deaktivierten *Tempera* Station bleiben am Server und werden **nicht** gelöscht).

Außerdem wird vorausgesetzt, dass für jeden User ein Login notwendig ist. Weitere Maßnahmen zur Sicherung des Projektes werden erst in weiteren Entwicklungszyklen umgesetzt (außerhalb des Projekts).

1.4.2. Logging

Das System soll sowohl Log-Files schreiben, als auch ein Audit-Log erstellen.

Audit-Log

Das Audit-Log dient zur chronologischen Dokumentation von User-Handlungen die die Systemsicherheit bzw. das Einhalten von firmeninternen oder externen Richtlinien und Regularien betreffen. Dazu gehören z.B. Handlungen, die die Lösch-Policy betreffen, jedes Login bzw. jeder Login-Versuch, Änderungen bei gespeicherten Usern (insbes. Rollenänderungen), oder Datenbankzugriffe, insbes. auf sensible Daten (z.B. Raumklimadaten).

Administratoren haben Zugang zum Audit-Log des Webservers. Das Audit-Log soll über die Weboberfläche einsehbar und nach verschiedenen geeigneten Ereignistypen durchsuchbar sein. Für jedes Ereignis soll Datum, Uhrzeit, auslösender User, Ereignistyp, das Ereignis selbst (z.B. welches User-Objekt verändert wurde), sowie eine Indikation, ob die Handlung erfolgreich war (z.B. Loginversuch Fehlgeschlagen) gespeichert werden.

Logfiles

Außerdem werden alle wichtigen Ereignisse, die das System insgesamt betreffen, in zwei unterschiedlichen Log-Dateien direkt am Webserver gespeichert. Anhand der Log Dateien soll sowohl das fehlerfreie Zusammenspiel der Systemkomponenten nachvollziehbar sein, als auch Fehlerbehebung anhand von Warnungen, und Errors aus dem laufenden Betrieb ermöglicht werden.

Dabei soll der Log-Dateien alle Informationen loggen, die andere nur die Warnungen und Fehlermeldungen. Zu Ereignissen zählen zum Beispiel das Hinzufügen neuer Verbindungen oder Geräte oder Ausfälle der Sensorstationen, Accesspoints oder des Webservers.

Ebenso soll eine Log-Datei am Accesspoint geschrieben werden. Bitte loggen Sie hier die Kommunikation mit den Sensorstationen, Übertragungen an den Webserver, Datenlöschungen, erfolgreiche Verbindungsversuche und Verbindungsprobleme bzw. -ausfälle sowohl mit Sensorstationen als auch dem Webserver.

Wichtige Informationen, die pro Ereignis gespeichert werden, umfassen mindestens das Datum, die

Uhrzeit, und Informationen zum Ereignis selbst. Wo die Log Dateien gespeichert werden muss außerdem sowohl am Webserver, als auch am Accesspoint konfigurierbar sein.

1.4.3. Ausfallsicherheit

Es kann weder von einer dauerhaft stabilen Stromversorgung ausgegangen, noch kann garantiert werden, dass die Kommunikation zwischen Sensorstation und Accesspoint, bzw. zwischen Accesspoint und Backend störungsfrei verläuft.

Folgende Ausfallsszenarien sollten Sie in der Entwicklung beachten und durch entsprechende Lösungen abdecken:

1. Eingeschränkte Kommunikation zwischen Arduino Sensorstationen und Rasperry PIs.
2. Unerwarteter Neustart eines Rasperry PIs.
3. Temporärer Ausfall der Kommunikationswege zwischen Minirechnern und zentralem Backend
4. Kurzfristiger Ausfall des zentralen Backends.
5. Ausfall der Sensorstation.

Es ist wichtig, dass alle Accesspoints und Sensorstationen automatisch nach einem Stromausfall wieder hochfahren und ihre Dienste selbständig und ohne Zutun von Außen aufnehmen. Es soll außerdem im Besonderen darauf geachtet werden, dass die ACID Bedingungen für Datenbanktransaktionen eingehalten werden.

Ausfälle von Sensorstationen oder Accesspoints müssen in der Webapp und in den Log-Dateien einsehbar sein. Außerdem muss diese Information in die Auswertungen der Messwerte miteinfließen, um das Verfälschen der Auswertungen durch fehlende Datenpunkte zu vermeiden.

1.4.4. Codequalität

SonarQube soll verwendet werden, um regelmäßig die Qualität des Source Codes der Webanwendung zu überprüfen und zu verbessern. Dies sollte bereits zu Beginn des Projekts umgesetzt werden.

2

Weitere Vorgaben

Folgende zusätzliche Anforderungen wurden Ihnen mitgeteilt:

- Webbasierte Multi-User Anwendung, lauffähig und bedienbar auf allen gängigen Plattformen (PC, Tablet, Smartphone) und aktuellen Internet-Browsern.
- Fokus auf *Responsive Design*. Die Strategie Mobile-First könnte hier hilfreich sein.
- Zulässige Technologien (Webserver): Java 17, Spring, OmniFaces, PrimeFaces, Postgres (auf Basis des SWE Skeleton-Projekts[1]). Insbesonders darf für die finale Abgabe kein externer Datenbankserver o.ä. verwendet werden. Grundsätzlich steht es Ihnen frei, ein anderes UI-Framework zu wählen. Sie müssen allerdings in der Lage sein, dieses selbstständig mit Spring und SonarQube zu konfigurieren.
- Verpflichtende Verwendung der Universitäts-GitLab Instanz[2] zur Verwaltung des Source Codes
- Zulässige Technologien (Arduino): C/C++. Es wird empfohlen den Code mit der Programmierumgebung VSCode[3] und PlatformIO[4] zu entwickeln.
- Zulässige Technologien (Accesspoint): Python >= 3.9, bleak[5].
- GATT Services und Charakteristiken müssen so weit wie möglich aus der Blutooth Dokumentation (GATT Specification Supplement)[6] entnommen werden.
- Änderungen am Technologie-Stack dürfen nur in Absprache mit der Proseminarleitung getroffen werden. Dasselbe gilt für den Wechsel von Major Releases eingesetzter Technologien und Frameworks.
- Modulare Softwarearchitektur unter Einhaltung gebräuchlicher Qualitätsmerkmalen und korrekter Anwendung von Architektur- und Entwurfsmustern
- Ausreichende Teststrategie für die Qualtitässicherung:
  - Anforderungen an die Software- und Codequalität Webapp: Testabdeckung von mindestens 65% (JUnit, Überprüfung durch SonarQube). Controller Klassen für UI-Funktionalität sollen aus der Berechnung der Testabdeckung ausgenommen werden, wenn sie keine testenswerte Programmlogik enthalten.
  - Anforderungen an die Software- und Codequalität Accesspoint: Hinreichende Testabdeckung (Unittests).
  - Bereitstellung realitätsnaher Testdaten in ausreichender Qualität und Quantität

[1] https://git.uibk.ac.at/informatik/qe/swe-skeleton
[2] `https://git.uibk.ac.at`
[3] VSCode: `https://code.visualstudio.com/`
[4] PlatformIO: `https://platformio.org/`
[5] Bleak Python Bluetooth Library: `https://pypi.org/project/bleak/`
[6] Bluetooth Spezifikation: `https://www.bluetooth.com/de/specifications/assigned-numbers/`

- Einheitliche Formatierung von Quellcode im gesamten Projekt
- Angemessene Dokumentation des Quellcodes und eingesetzter Installations- und Konfigurationsfiles.
- Angemessene Verzeichnis- und Projektstruktur.
- Keine unerlaubte Nutzung urheberrechtlich geschützter Werke, keine Verletzung von Lizenzbestimmungen. Fremde Inhalte müssen klar als solche inkl. Quellenangabe gekennzeichnet werden.

References

[1] Kersten Bux and Bundesanstalt für Arbeitsschutz und Arbeitsmedizin, eds. *Gesundes Klima und Wohlbefinden am Arbeitsplatz*. 2., durchges. Aufl., Dezember 2012. Dortmund: Bundesanst. für Arbeitsschutz und Arbeitsmedizin, 2012. ISBN: 978-3-88261-691-0.

Zum Verfassen diese Projekts wurden KI basierte Tools als Unterstützung verwendet:

- **LanguageTool**: Rechtschreib- und Grammatikprüfungstool zur ergänzenden Korrektur der Texte