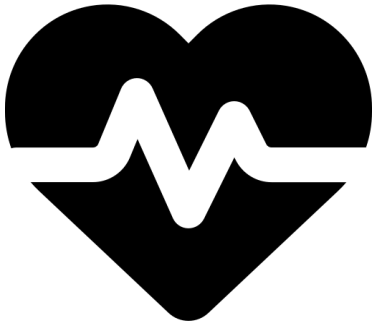


Guard

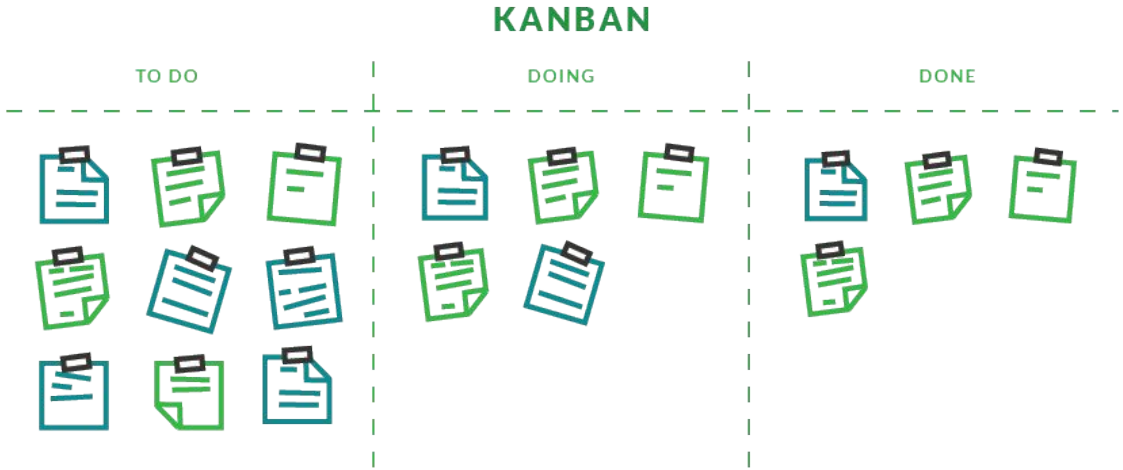
Recap: Ambient Assisted Living

Unser Lösungsvorschlag:

- Device mit EKG-Funktion oder Pulsmonitoring als Hardware
- Software: Erkennung von Herzstillstand *(Asystolie)*
- Evtl. Erkennung anderer lebensbedrohlicher Herzrhythmen
    - *(Kammerflimmern, ventr. Tachykardie, AV-Block Grad III, torsades de pointes, etc.)*
- Bedrohlicher Rhythmus erkannt, was nun?
    - Jede Minute zählt!
    - Daher: **Automatisierte Verständigung der Rettung**

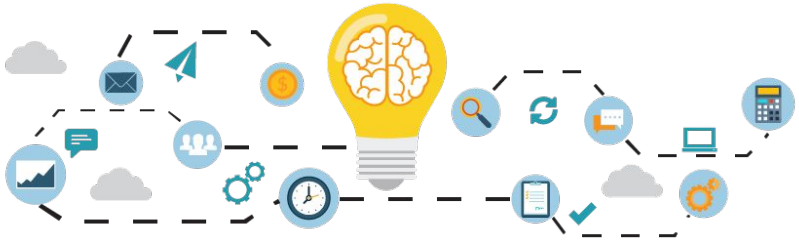
Projektverlauf

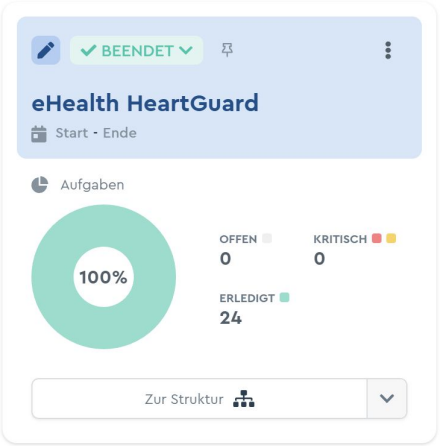
- Projektmanagement mittels Factro (Cloud Kanban, Gantt etc.)
- Wöchentliche Meetings um den Projektfortschritt zu besprechen
- Fertigstellung der besprochenen Arbeitspakete nach vereinbartem Zeitplan
- Abstimmung über WhatsApp/Discord/GoogleDocs/GitHub



Projektverlauf: Prototyping

- Viele Meetings in Präsenz
- Zusammenfügen der einzelnen Teilkomponenten
- Abschluss der letzten Arbeitspakete





Use - Cases

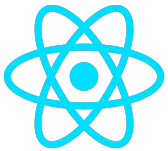
- Überwachung / Früherkennung von Herzproblemen
- Alarmierung von Angehörigen oder Rettungskräften
- Unterstützung von medizinischem Fachpersonal
- Aufzeichnung von EKG - Daten (Forschung und Analyse)
- Dauerhafte Überwachung von Risikopatienten

Backend Architektur

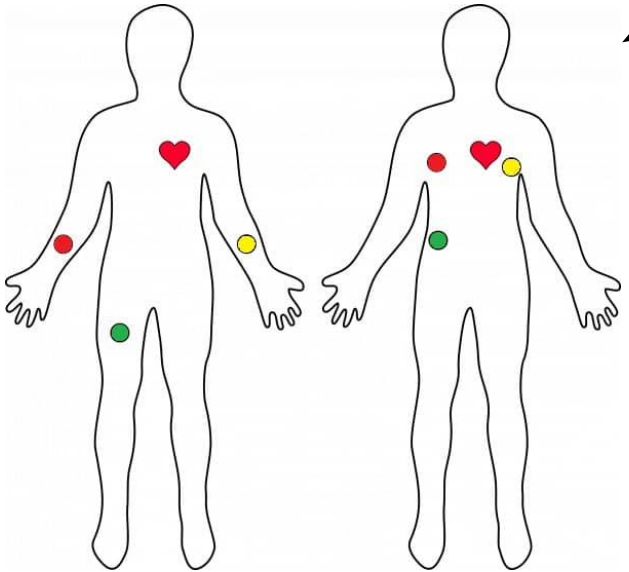
- Spring Boot 3.0.5
- H2 Database Engine
- HAPI FHIR

Frontend Architektur

- React Native 0.71.4
- Expo 48.0.9
- Fontawesome 0.3.0



Prototyp - Aufbau



ECG - Measurement

Message to emergency contact

gatewayapi

Converter

Analyzer

Frontend Application

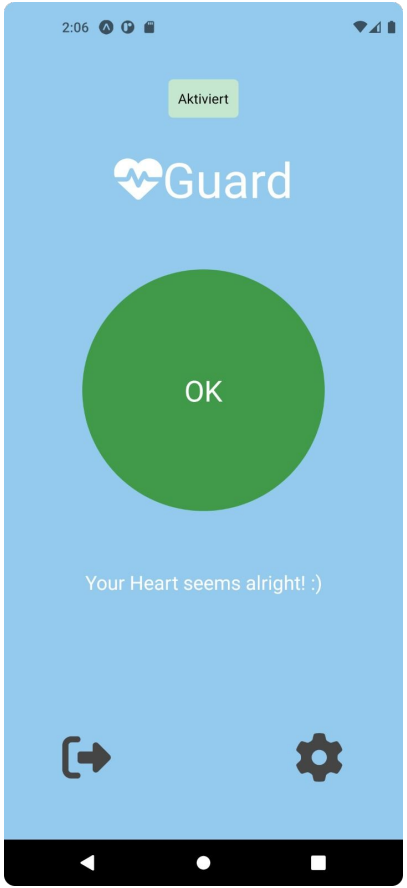
Backend - Erreichte Ziele

- ~ 5200 Zeilen Java-Code
- Eigener Algorithmus zur Erkennung einer Asystolie
- Funktionsfähige Implementierung der externen EKG-Analyse-Library JELY [1] und ihrer Analyseergebnisse
- Vollständige Implementierung einer FHIR-Observation für EKG-Daten, mit FHIR-Extensions zur CustomObservation erweitert um zusätzliche Verwaltungsdaten zu halten
- Custom serialization/deserialization mit FHIR-Parser anstelle von Spring Boots Jackson-Parser
- Grundlegende Implementierung von Spring Security
- Sicheres Passwort-Hashing mit Argon2 [2]
- Eigener Suchalgorithmus um optimale Parameter für Argon2 zu finden
- Vollständig Multi-User fähiges System, beliebige viele Clients pro User, funktionsfähige Verwaltung von Usern und Clients, User können nur auf ihre jeweils eigenen Daten/Clients zugreifen
- Vollständige Persistenz von Usern, Clients, empfangenen EKG-Daten und Analyseergebnissen
- Möglichkeit, alle gespeicherten EKG-Daten und -Analysen eines Users aus einem bestimmten Zeitfenster auszugeben

[1] https://github.com/mad-lab-fau/JELY [2] https://www.password-hashing.net/#argon2

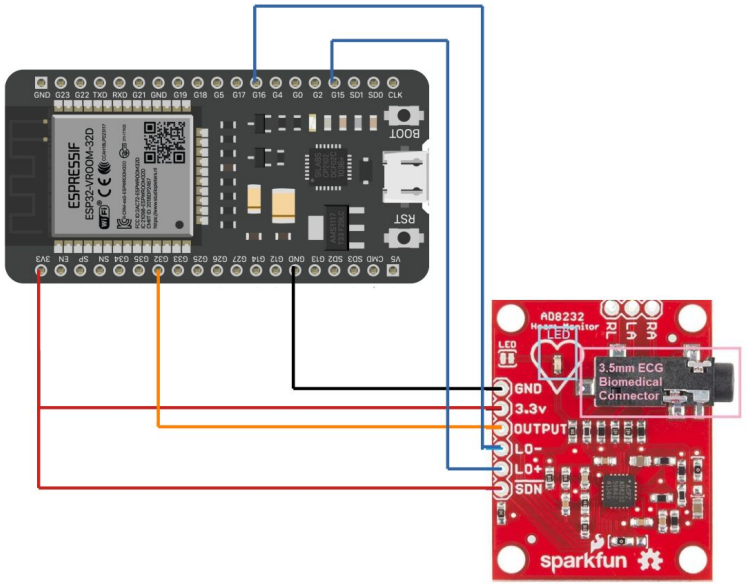
Frontend - Erreichte Ziele

- Registrierung von neuen Usern
- Einloggen mit bestehenden Usern
- EKG Daten von Backend bekommen
- Alarm & Timer starten bei kritischem EKG
- Abbrechen des Timers
- Notfalls-SMS bei ablaufendem Timer
- Bearbeiten eines Users
- Verknüpfung von User und EKG-Gerät



ESP32 - Erreichte Ziele / Funktionalität

- Emuliert EKG-fähiges Device
- Empfängt Daten von AD8232 Modul als Analog Input
- Berechnet Spannungswerte
- Registriert sich beim Backend
- Sendet jede Sekunde Spannungswerte als JSON ans Backend



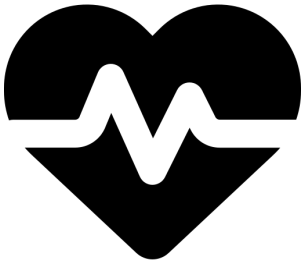
Offen bis zur Produktivversion

Backend:

- Fine-tuning von Spring Security
- TLS/HTTPS
- Kryptographie mit RSA/AES
- Algorithmen zur Erkennung anderer lebensbedrohlicher Rhythmusstörungen
- Wissenschaftliche und klinische Validierung des Programms

Frontend:

- Grafische Anzeige gespeicherter EKGs, genauere Auswertungsergebnisse



Guard - Demo!

Danke für eure Aufmerksamkeit!