|  |  |  |
| --- | --- | --- |
| Functional Design (FD)  Project: Integration | | |
| **Enschede, March 21, 2025 March, 2025** | | |
|  | | |
| **Version 1.0** | **Group: 5** |  |
|  | **Mo van Elk** | 462818 |
|  | **Konstantin Apostolov** | 552608 |
|  | **Jort Bolscher** | 556491 |
|  | **Artur Golovko** | 547206 |
|  | **Tom te Lintelo** | 532911 |

Abbreviations

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| CD | Concept Design |
| FR | Functional Requirement |
| NA | Not Applicable |
| SDR | System Design Requirements |
| TR | Technical Requirement |
|  |  |
|  |  |

Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Functional Requirement | Requirement related to the system functions (‘what?’) |
| Technical Requirement | Requirement related to the system design (‘How?’) |
|  |  |
|  |  |

References

|  |  |  |
| --- | --- | --- |
| **Number** | **Author(s)** | **Title** |
| [1] | J.W. van Dijk | Project Based and Methodological Design: A Practical Approach |
| [2] |  |  |
|  |  |  |

Table of contents Functional Design document

Contents

[1 Introduction 4](#_Toc193278779)

[1.1 Introduction to project 4](#_Toc193278780)

[1.2 Overview of document 4](#_Toc193278781)

[1.3 Objectives 4](#_Toc193278782)

[1.4 Scope and Limitations 4](#_Toc193278783)

[1.5 Document Structure 5](#_Toc193278784)

[2 Analysis of requirements 6](#_Toc193278785)

[2.1 Functional Analysis 6](#_Toc193278786)

[2.2 Parameters 6](#_Toc193278787)

[3 Concept Principles 7](#_Toc193278788)

[3.1 Description of alternative concepts 7](#_Toc193278789)

[3.2 Comparison of concept principles 7](#_Toc193278790)

[3.2.1 Comparison parameter 1: Fall detection 7](#_Toc193278791)

[3.2.2 Comparison parameter 2: Heartbeat detection 7](#_Toc193278792)

[3.2.3 Comparison parameter 3: User Interface 8](#_Toc193278793)

[3.2.4 Comparison parameter 4: Power supply 8](#_Toc193278794)

[3.2.5 Comparison parameter 5: Medication alert 8](#_Toc193278795)

[3.2.6 Comparison parameter 6: emergency alert 8](#_Toc193278796)

[3.2.7 Comparison parameter 7: microcontroller 9](#_Toc193278797)

[3.3 Choice of most promising concept principle 9](#_Toc193278798)

[4 Functional design Elaborations. 10](#_Toc193278799)

[4.1 Overview of functional design 10](#_Toc193278800)

[4.2 Elaboration of functional design 12](#_Toc193278801)

[4.2.1 Electrical 12](#_Toc193278802)

[4.2.2 Electronic 12](#_Toc193278803)

[4.2.3 Software 13](#_Toc193278804)

[4.3 Functional design integration 14](#_Toc193278805)

[5 Conclusion 16](#_Toc193278806)

[Appendix A: Calculation of something difficult 17](#_Toc193278807)

# Introduction

## Introduction to project

This project focuses on developing a smart healthcare device that monitors heart rate, fall detection, provides emergency alerts on falling and medication reminders. The device functions as a wearable watch that ensures patient safety through real-time monitoring and communication with caregivers in the given moment through desktop application and web application. The user gets audbile alerts if it is time to take in their medication and can see their own heartbeat on a display.

## Overview of document

This **Functional Design Document (FD)** provides a detailed breakdown of the system’s design, ensuring that all functional requirements are met. It defines the electronic, software, andnecessary for the development of the system. The document also explains integration strategies, and compliance requirements to ensure the system meets industry standards.

## Objectives

The primary objective of the project are:

* To design a wearable device capable of real-time heart rate monitoring.
* To implement an emergency alert system that can notify caregivers or emergency services.
* To ensure seamless integration with a windows application and web application for monitoring.
* To give notification for medicine intake during the needed times.
* To develop a great and scalable infrastructure for data handling and getting real time data.

## Scope and Limitations

**Scope**

The scope is mostly things to be accessible and where they are going to be accessible.

* Implementation of windows applications and web-based user interfaces.
* Cloud-based storage(e.g. Azure cloud services).
* Integration of **wireless connectivity** for real-time synchronization.
* Implementation of fall detection algorithms (or some way to minimize falls positives) for emergency scenarios.

**Limitations**

The **limitations** of this system include:

* The device is not designed for exposure to water, making it unsuitable for activities such as swimming or showering with the device.
* The device will be dependable on Wi-Fi for real-time data synchronization, which may limit usability in areas with poor connectivity.
* **Limited battery life.**
* Environmental limitations the device cannot function perfectly in highly humid conditions or strong electromagnetic fields, which may interfere with sensor accuracy.
* **No Multi-User Support** on a Device each device is designed to only track a single user switching between multiple users is not supported.
* The project is a proof-of-concept, so there will not be a real watch to wrap around your wrist.

## Document Structure

This document is structured into multiple sections, and we are going to explain for what each section is so it's covering all aspects of the system’s functional design:

* **Section 2: Analysis of Requirements –** Defines the functional and non-functional requirements necessary for the system development.
* **Section 3: Concept Principles –** Explores various design alternatives, comparing and justifying the chosen implementation approach.
* **Section 4: Functional Design Elaboration –** Provides detailed insights into the electrical, mechanical, software, and electronics.
* (*this is extra chapters we should add).*
* **Section 5: Conclusion**
* **Testing and Validation –** Defines the testing methodologies we are going to be using to verify that everything we are going to use works.
* **Project Management –** Outlines the implementation roadmap which part should be done by when, including the milestones, risks and the budget estimations we made previously.
* **Future Considerations and Improvements –** Think of different other solutions for upgrading our system (e.g. waterproofing AI-based health predictions etc.).

# Analysis of requirements

## 2.1 Functional Analysis

A functional analysis of the requirements is where in the system is split according to their functionality. These sub-systems would be the input, processing, output, and sequence control. Input control, which is the sub-system that involves the peripherals of the system that are related to input. Examples would be sensors, data of some sort, user input etc. In the case of this project it would be handling the input from the accelerometer, and heart rate sensor and button for example. Essentially the sub-section is responsible for clean, secure and stable data input. Next would be the processing, this would be the sub-system that would be, as the name implies processing data, sorting, organizing, calculating etc. This part of the system takes in user input and converts/modifies it into system useable data.

## 2.2 Parameters

To come to a functional concept a few key decisions need to be made. When it comes to inputs the system is required to read a user's heartrate and detect when they fall. In terms of output there are multiple parameters that need to be decided. The users should be able to see their heartrate and other statistics on the smartwatch. The user also needs to be alerted when they need to take their medication. Both the users and their healthcare practitioner need to be alerted when a fall or irregularity in heartrate is detected. Lastly, the entire system needs to be consistently and reliably powered. Making the choice of power supply another key parameter.

Our final list of parameters is as follows:

* Fall detection
* Heartbeat detection
* User Interface (watch)
* Medication alert
* Emergency alert practitioner
* Power supply

# Concept Principles

Now that all requirements are know the team made a few concepts the proof-of-concept could be made of.

## Description of alternative concepts

For each parameter a few options are listed.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Option 1 | Option 2 | Option 3 |
| Fall detection | Accelerometer | Gyroscope | Camera with AI recognition |
| Heartbeat detection | ECG sensor | Optical heart rate sensor |  |
| User interface (watch) | OLed display | TFT LCD |  |
| Power supply | USB powered | Solar energy | Rechargeable LiPo battery |
| medication alert | Vibration | Audible ping | Message on phone |
| Emergency alert practitioner | pager | Message on phone (SMS) |  |
| Microcontroller | ESP32 | Arduino | Raspberry |

## Comparison of concept principles

To determine which concept fits the requirements the best a weighted criteria is drawn up.

How good a concept fits the criteria is rated with + and -. The more plusses the better it fits, the reverse for the minus.

### Comparison parameter 1: Fall detection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fall detection | Accuracy | Response time | I2C | cost |
| weight | 50% | 25% | 15% | 10% |
| Accelerometer | + | + | + | + |
| Gyroscope | ++ | + | + | + |
| Camera with AI | ++ | ++ | -- | --- |

The fall detection should not give false positives, which can lead to the emergency getting notified without reason. So, it weighs more. But if the user falls it should respond fast. Furthermore, it needs to work with I2C to communicate with the microcontroller. So, the gyroscope is the best.

### Comparison parameter 2: Heartbeat detection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Heartbeat detection | Accuracy | Power consumption | Cost |  |
| weight | 50% | 25% | 25% |  |
| ECG sensor | ++ | + | - |  |
| Optical heart rate sensor | + | + | + |  |

The detection must be as accurate as possible to receive good data. The ECG sensors are generally better at this, but the downside is that they need to be attached to the skin.

### Comparison parameter 3: User Interface

|  |  |  |  |
| --- | --- | --- | --- |
| User interface (watch) | Readability | User interaction | cost |
| weight | 50% | 40% | 10% |
| OLed | ++ | + | - |
| TFT LCD | + | ++ | + |

The TFT LCD is a touchscreen display, the OLed does not have this. While an OLed has a much better contrast ratio which makes it easier to read.

### Comparison parameter 4: Power supply

|  |  |  |  |
| --- | --- | --- | --- |
| Power supply | Rechargeability | Safety | Size & weight |
| weight | 50% | 20% | 30% |
| USB | - | - | -- |
| Solar | - | + | ++ |
| LiPo | ++ | - | + |

The power supply needs to rechargeable and use it again. The battery can be to big to make the watch traversable. A USB does work but is very unhandy with a constant long cable. The solar panel works good be only operates at daylight. Which leaves the LiPo as the best option

### Comparison parameter 5: Medication alert

|  |  |  |  |
| --- | --- | --- | --- |
| Medication alert | User accessibility |  |  |
| weight | 100% |  |  |
| Vibration watch | + |  |  |
| Audible ping watch | ++ |  |  |
| Mobile message | - |  |  |

For the medication alert there is only one criterion, which is how accessible it is, in other words. Can the user sense that the medication alert was given. This can be done in multiple ways. The mobile messages only works if the patient is using its mobile. The vibration needs to be big to sense it, and if the user is not ‘wearing’ the watch they would miss it. The audible ping has the biggest change of alerting the patient. Only in noise situations and for deaf patients this won’t work. So the best would be a combination of a vibration and audible.

### Comparison parameter 6: emergency alert

|  |  |  |  |
| --- | --- | --- | --- |
| emergency alert practitioner | User accessibility | Connection to emergency services |  |
| weight | 60% | 40% |  |
| Pager | + | - |  |
| Mobile message | + | ++ |  |
|  |  |  |  |

Here the same applies as the medication alert but even more important. The device needs to have contact with the system and to emergency services. Because of this the mobile messages would be the best.

### Comparison parameter 7: microcontroller

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| emergency alert practitioner | Processing power | Memory (RAM) | protocols | connectivity | Operating voltage |
| weight | 15% | 15% | 30% | 30% | 10% |
| ESP32 | + | - | ++ | + | - |
| Arduino uno | + | + | + | - | + |
| Raspberry pi | ++ | ++ | - | + | + |

The microcontroller needs to be able to use I2C and I2S protocols and Wi-Fi for the project. It is also nice to have a fast-processing power, big memory and handy operating voltage, but this is not a big requirement. For this the Raspberry pi would be the best.

## Choice of most promising concept principle

Now all the best options are founded in the weighted comparison the morphological diagram gets updated where all the best options form the comparison is in bolt, but the final selected option is in a red box underlined.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Option 1 | Option 2 | Option 3 |
| Fall detection | **Accelerometer** | Gyroscope | Camera with AI recognition |
| Heartbeat detection | ***ECG sensor*** | Optical heart rate sensor |  |
| User interface (watch) | **OLed display** | TFT LCD |  |
| Power supply | USB powered | Solar energy | **Rechargeable LiPo battery** |
| medication alert | Vibration | **Audible ping** | Message on phone |
| Emergency alert practitioner | pager | **Message on phone (SMS)** |  |
| Microcontroller | ESP32 | Arduino | ***raspberry*** |

The only best options that does not fit for the project is the ECG sensor and the Arduino.

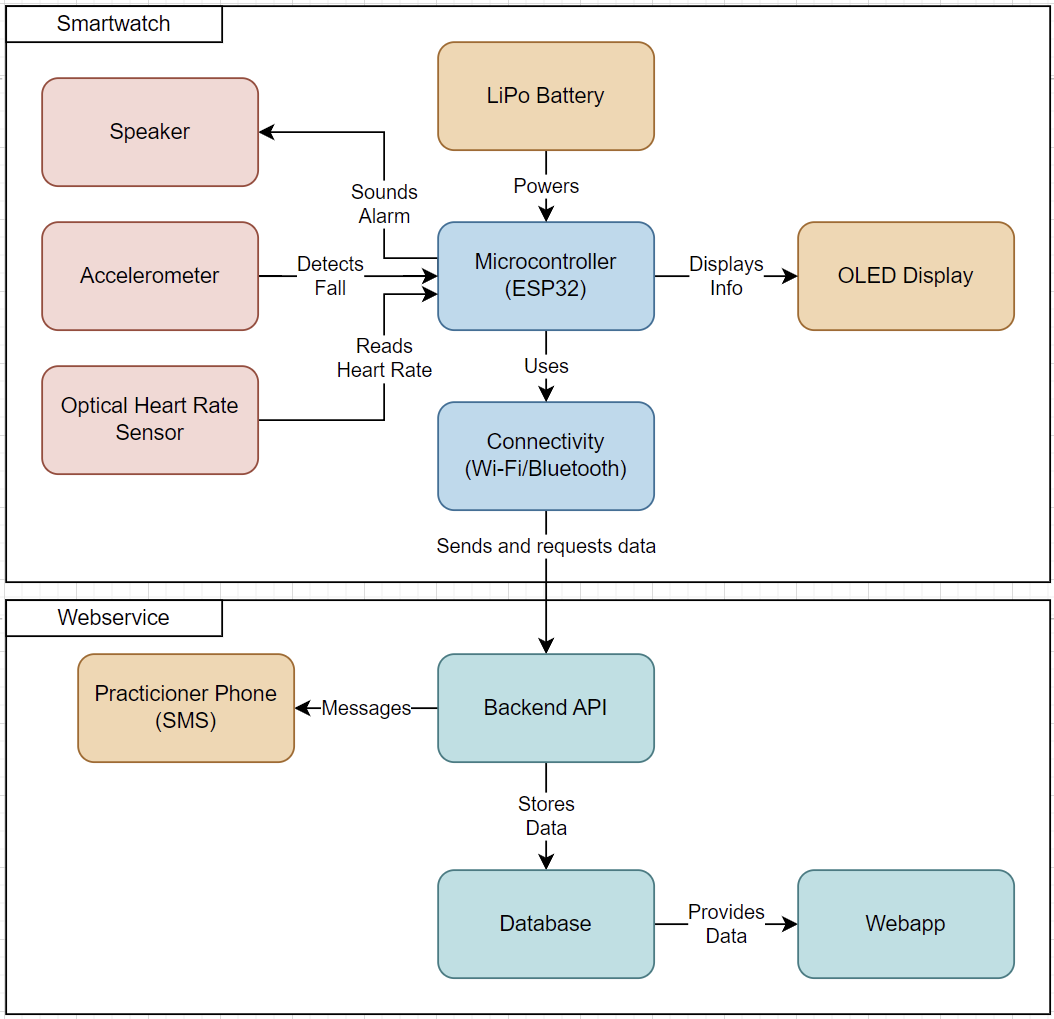
Because the ECG sensor uses electrodes on the skin. This is not handy to wear during everyday life. So, the optical heart rate sensor will be used.

The ESP32 is simply required by the stakeholders. Therefore, while an Raspberry comes as best out of the comparison, an ESP32 will be used.

# Functional design Elaborations.

## Overview of functional design

This diagram describes the system and its parts in a functional manner. It gives an overview of all the previously mentioned chosen parts, how they function and how they interact with each other.

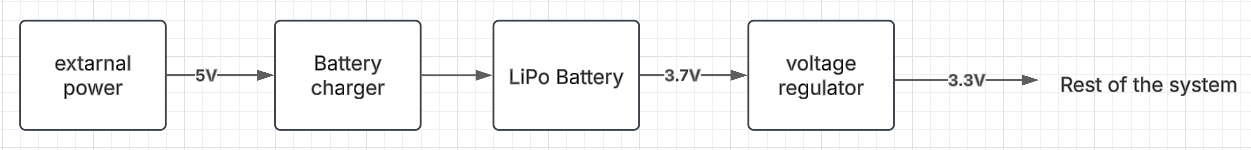
Here the system is divided into two sub-systems: the smartwatch and the webservice. With the Connectivity module and the backend API acting as the bridge connecting the two. The smartwatch has the Microcontroller at its core connecting everything together. The webservice’s backend sends out signals and stores data in a database. It also hosts a webapp powered by the same database.

## Elaboration of functional design

### Electrical

The Electrical design show the flow of the current trough the circuit.

As there is only one power needed for the system the electrical diagram is a linear line from the power input, this gets converted to the operating voltage and goes into the rest of the system. As it all works on 3.3V.



### Electronic

The electronic design shows how all the components are connected to each other.

The code to run on the microcontroller gets delivered through the programmer, which gets its data through a data cable connected to a device and a USB port.

There are 2 sensors that measure data and send it to the microcontroller. This data gets processed and is shown on the display.

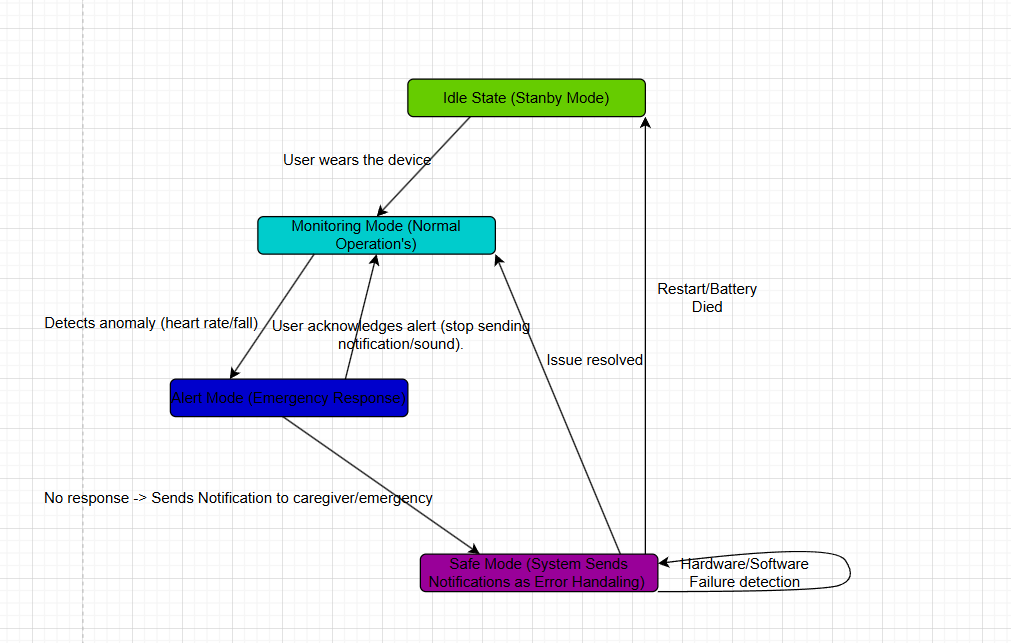
Then there is a memory storage where sound files can be stored, this goes to the microcontroller, and under certain conditions this audio gets send to the sound amplifier and goes into the speaker.

Lastly there are two buttons were the user can select what the screen displays.

A diagram of a computer

AI-generated content may be incorrect.

## Software

* **Software Architecture**:
  + **Modular programming** for scalability.
  + Well-defined **API communication layer** for data exchange.
* **Core Algorithms**:
  + **Heart Rate Monitoring Algorithm** – Processes PPG signals for trend analysis.
  + **Fall Detection Algorithm** – Uses accelerometer data to detect sudden motion changes.
  + **Notification System** – Sends alerts via Bluetooth/Wi-Fi based on predefined triggers.
* **State Transition Diagram**:
  + 
* **Interrupt Handling**:
  + Interrupt-driven **motion detection and heart rate sensing**.
  + Efficient **real-time processing** to minimize latency.
* **Programming Languages & Environment**:
  + **Embedded firmware (C++)** for the smartwatch.
  + **Web-based (JavaScript)** for UI.
  + **Backend (JavaScript or Python)** for data handling.

## Functional design integration

### Introduction

This Functional Design Integration of the smartwatch system ensures that all software components interact with each other seamlessly to deliver an efficient and user-friendly experience for the patient and the caregiver. This chapter outlines how different functional modules such as health monitoring, emergency alerts, and connectivity are integrated into a cohesive system. It also describes the communication flow between the **smartwatch, backend API and user interface.**

Functional integration is crucial in ensuring that features like **heart rate monitoring, fall detection, and medicine notifications** operate reliably across different devices and platforms. This process involves defining **data contracts, designing API interactions, performing integration testing, and database integration (local or cloud)** between components.

### Objectives of Functional Integration

The primary objectives of the **Functional Design Integration** for the smartwatch system include:

* Ensuring smooth data flow between the smartwatch, backend, database and UI.
* Implementing real-time data collection for health functionalities for heart rate monitoring, fall detection and time tracking (Date Time).
* Maintaining system consistency across multiple platform (e.g. Web Service, Window, Mac).
* Enabling secure and reliable Wi-Fi connectivity between the smartwatch and the backend.
* Handling error detection.

### Functional Integration Approach

The integration process has different software modulus and hardware modulus that interact with each other effectively.

**3.1 Smartwatch and Sensor Data Integration**

The microcontroller we are going to use is the LillyGO TTGO t-dispaly V1.1 ESP32 that collects the data from the accelerometer and heart rate monitor. Also for the Time if the ESP32 is connected to the Internet he can take it from there which also the data will be transmitted from the Wi-Fi to the backend.

* Input Data:
* Example Heart rate sensor: {“heart\_rate”: 85, “unit”: “bpm”, “state”: “normal”}
* Accelerometer and Gyroscope: {“Accel”: 9.83, “unit”: “m/s^2”, “Gyro”: 0.05, “unit” “deg/s”}
* Data Processing:
* The smartwatch pre-processes the data and determines abnormalities in the ESP32.
* Send to Backend API/Database
* Send the data through Wi-Fi to the API and the Database.

**3.2 Backend API Calls and Data management**

The backend API receives sensor data from the smartwatch and determines appropriate actions, such as storing health data, triggering emergency alerts, or sending medicine reminders.

* API Endpoints:
* POST /api/health-data -> Receives heart rate and Fall detection data
* POST /api/time -> Receives the Time
* POST /api/notifications -> Sends emergency notifications to caregivers / or time to take the pills needed.
* Data Storage:
* SQLServer database stores all the data.

**3.3 User Device Interaction**

Users interact with the system via:

* Web Interface (you can open it on Windows, Mac, Linux) which will be built on C#.
* Windows Aplication.
* **Integration Mechanism:**
* The Windows Applicaiton and the Web interface are ogng to fetch the data from the backend API every 10 seconds using RESTful API calls.

**3.4. Key Challenges in Functional Integration**

|  |  |
| --- | --- |
| **Challenges** | **Solutions** |
| Data synchronization issues | To keep everything upto real-time we are going to do everything Asyncronsly and using WebSockets or MQTT. |
| Network connectivity problems | Use offline storage & retry mechanism for unreliable connections. |
| Latency in health alerts | Prioritize low-latency API calls. |

**3.5. Conclusion**

Functional Design integration is a complicated task ensuring all the components of the smartwatch system work together. By the previous chapters shown before we showcase of implementing a robust, great connectivity and fast API calls for the notifications.

# Conclusion

For the project a weighted selection is done to ensure the best option is chosen for the key parameters. With this chosen concept the system has few limitations. As it limits the users freedom a bit in how they want to receive messages.

# Appendix A: Calculation of something difficult