1 Introduction

The Internet of Things (IoT) will change humanity in profound and unforeseeable ways[1]. According to a report by Cisco, between 2015 and 2020 the number of connected IoT devices is set to double from 25 billion to 50 billion[1]. This new revolution is driven by lower prices in manufacturing IoT devices as well as new inventions in radio communication and better processing power. This means more data than ever will be collected and automation will be the norm, rather than the exception. Industrial IoT (IIoT) and smart home are prime example for this. New factories are often operated by only a few workers and lights and temperature are increasingly controlled by an AI rather than the human living in it. This leads to higher productivity, better power management and more comfort among other improvements. In most cases, this new technology is used to aid humans, but recent history has shown that IoT brings many diverse challenges in technology, security and social life.

In 2016 a botnet with over 600k infected devices, primarily embedded and IoT devices, overwhelmed several high-profile targets like the DNS provider Dyn with a massive distributed denial-of-service (DDoS) attacks. This caused a temporary outage of their DNS servers making many webpages, among others Twitter, Spotify und Amazon, temporally inaccessible.

The sheer number of IoT devices also makes it important to think about their sustainability. Many IoT devices are cheap to manufacturer and thus only used for a short period of time and are hard to update. This makes basically makes them a throwaway product. Many IoT devices have a stable power connection and are not designed to be energy efficient.

IoT devices are by design often close to humans and are intended to analyze what they say, do and how their body reacts. This leads many people to assume that IoT, especially smart home and activity tracking are incompatible with privacy[2].

In the past many companies overstepped the line what is socially excepted and had to back-roll certain "features"[3]. For many consumers, privacy is the major concern in smart home[3].
These four areas, technological improvements, security, sustainability and privacy, are the four areas shaping the development of IoT and the main drivers for many technological as well as architectural decisions in the development process of the product developed as part of this project. The reminder or this report is structured as follows, ***

1.1 Motivation

In this project we develop a new wireless goal for foosball tables. The backend software to manage user data and a primary architecture was part of a previous project and is shortly discussed in the next chapter. The table and the backend software works as intended, but the physical implementation of the table, shown in figure *** is tailor made for the table we used in the development and not portable across different brands and designs.
SHOW PICTURE OF TABLE!!

In order to make our solution attractive to bars and companies alike the solution neeeds to be easy to implement and manage. Thus, the research question for this project is as follow:

What is the optimal design for a scalable wireless foosball goal?

We will answer the question based on the four aspects, the technology, security, sustainability and privacy.

1.2 Delimitations

Due to time and equipment limitations we will focus on the technology in this report. Also, the findings from our tests are useful as a general reference point when discussing the strength and weaknesses of competing solutions. But as the tests do not follow strict academic rules, these findings are not statistical significant.
We also limit the main scope of this report to how we send and receive data from the goal to its controller. We will show the entire architecture but it is not part of the requirements and often not analyze or explained as throughly as the goal sensors.
Finally, we did not gather input from experts. This is something we would have wished to do, but was not possible due to time constraints.

2 IAFoosball

IAFoosball is the name of the project and startup we developed in the previous semester. It is meant to make foosball more interactive, to show and share statistics, and make it more sociable by using new technologies. The planned services include global and private rankings, table finder, friends, automatic tournaments and more. Because it is part of an university project, we used the latest and greatest technology. The front end is written in Flutter and pReact, the back end is separated in containerized microservices written in Go and all communication is done through gRPC, which uses protobuffs and HTTP/2. Figure 1 shows this architecture.
The foosball table in figure 1 is the development version, so it is a fully featured version. This includes, speakers with Spotify integration, LED lights, automatic ball release and a tablet. The software and hardware running on the table is not new not portable, especially concerning the goals. Figure 2 shows the electronics

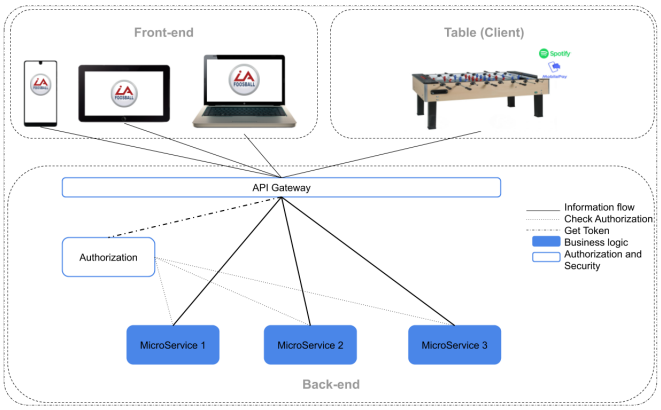

Figure 1: The old IAFoosball architecture

of the old goal setup.

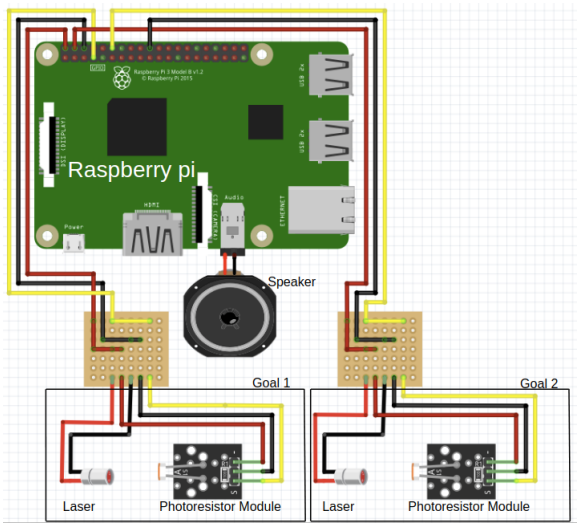

Figure 2: The old IAFoosball goal

The sensors were directly connected to raspberry pi and a simple javascript application registered the goals and send them to a remote server. This required a power outlet and a raspberry pi on each table. It also required cables from each goal to the raspberry pi. This approach did not scale well. Hence, we re-thought the design, making it more varsatile, easier to integrate and manage.

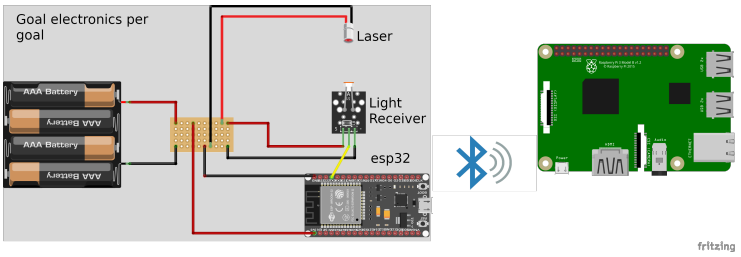

Figure 3: The new IAFoosball goal

Figure 3 shows this new setup. Marked in grey are the components used for each goal, a battery (we use a lithium-ion battery), one laser, one light sensor and one esp32. The esp32 is the component doing the computation and sending data to a raspberry pi via bluetooth low energy (BLE). It is also the main factor for the goal design, which will be discussed next.

3 Requirements

The requirements are split into functional and non-functional requirements, where the former are definitions of what a system is supposed to do and the latter requirments describing how the system is supposed to be. The respective tables are included at the end of the chapter to enhance readability.

The functional requirements are classified with the MoSCoW method into Must, Should, Could, and Would and are listed in section 3[1]. The musts reflect the requirements for our project from this course. Our objective is to create a functional wireless automatic goal registration device driven by a battery and having a low maintenance cost. Power saving mechanisms are thus a top priority and tested in the respective chapter (add reference). The overall architecture is also very important for mangement and fault tolerance. But as these topics are not an essential part of the curriculum they are not explained extensively. Finally, "security in

[1]Class stands for the MoSCoW classifier.

IoT devices is either neglected or treated as an afterthought"[4] by many manufacturers. We split security into two areas, architecture security and encryption, as changing a system is often much harder than upgrading the encryption.
The criteria for the non-functional requirements are shown in section 3. In contrast to many consumer facing technologies usability is not part of the requirements, as the enduser should have as little interaction with the system as possible and physical maintenance is only carried out by professionals[2].
We cover the areas: Reliability, Robustness, Portability, Maintainability and Efficiency, with a focus on this course curriculum.

[2]They would still be nice, but due to time limitations we excluded them.

Functional Requirements

ID	Name	Class	Description
100	Register goals	Must	The system must be able to automatically register goals
101	Transmit goals	Must	The system must be able to automatically transmit goal information to the edge device.
102	OTA updates	Should	The microcontroller should support over the air (OTA) updates
103	Low energy transmission	Must	The microcontroller and its transmission partner must be able to communicate via a low energy transmission protocol
104	Power modes	Must	The microcontroller must be able to support different power mode in which it can operate to save power during unused periods.
105	Wake up from sensor	Must	The microcontroller must be able to wake up from an external sensor input.
106	Wake up from timeout	Must	The microcontroller must be able to wake up after a predefined timeout.
107	Transmission range	Must	The transmission range must be larger than 5 meters without blockage.
108	Pin reads	Must	The microcontroller must support reading from input pins.
109	Centralized configuration	Should	All configurations should be synchronized across the nodes in the system.
110	Node fault tolerance	Should	The state of each node should be synchronized in the system, so that in case of an error, the last operable state can be recovered
111	Adjustable Goals setup	Should	The goals setup need to able to adjust in their width to cover a variety of different goal sizes

Functional Requirements

ID	Name	Class	Description
112	Containers (excl microc.)	Should	All applications, excluding the microcontroller, should run inside containers.
113	Containers	Would	The microcontroller code runs inside containers as well.
114	Single binaries	Could	All applications should be packaged in one binary.
115	Edge storage	Would	The edge part should be able to count goals and syncronize with user devices without the Internet.

Non-Functional Requirements

Area	ID	Name	Description
Reliability	100	Downtime	The downtime due needs to be less once per month.
	101	Recording failures	Less than 5 percent incorrect readings.
Robustness	200	Interference	The system should be able to deal with at least 10 bluetooth enabled devices nearby.
	201	Transmission failures	There should not be more 0.01 percent of transmission failures, including all reasons.
	203	Incorrect data	There should not be more 0.0001 percent of wrong data transmission.
	202	Crashes	In case of a crash, the software must be able to recover automatically.
Portability	300	Supported platforms	The application needs to supported on a wide variety of IoT devices for future changes.
Maintainability	400	Updates	The software needs to be updatable over the air (OTA).
	401	Centralization	As long as a table is connected to the Internet (also indirectly through the raspberry pi), all updates must be available through a central point.
	402	Bug fixes	All bugs need to be addressed latest 6 months after discovery
Efficiency	500	Battery life	The system needs to able to life on a single battery charge for at least two weeks with no more than 20 games per day.
	501	Battery replacement	The battery needs to be able to charge to above 80 percent of its original value after 2 years, with 25 recharges a year.

References

[1] D. Evans, "The internet of things: How the next evolution of the internet is changing everything", *Cisco Internet Business Solutions Group (IBSG)*, vol. 1, pp. 1–11, Jan. 2011.

[2] S. Morrow, *5 reasons privacy and iot are incompatible | iot for all*, `https://www.iotforall.com/five-reasons-privacy-iot-incompatible/`, (Accessed on 05/17/2019), Oct. 2018.

[3] R. MARVIN, *Privacy tops list of consumer smart home concerns - pcmag uk*, `https://uk.pcmag.com/nest-secure/119897/privacy-tops-list-of-consumer-smart-home-concerns`, (Accessed on 05/17/2019), Mar. 2019.

[4] J. Wurm, K. Hoang, O. Arias, A.-R. Sadeghi, and Y. Jin, "Security analysis on consumer and industrial iot devices", in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, IEEE, 2016, pp. 519–524.