

**STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST**

**Obor: 18. Informatika**

## **Integrace do průmyslu 4.0**

**Jakub Andrýsek**

**Brno 2021**

**STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST**

**INTEGRACE DO PRŮMYSLU 4.0**

**INTEGRATION INTO INDUSTRY 4.0**

**AUTOR** Jakub Andrýsek

**ŠKOLA** Gymnázium Brno, Vídeňská,  
příspěvková organizace

**KRAJ** Jihomoravský

**ŠKOLITEL** Mgr. Jaroslav Páral

**OBOR** 18. Informatika

**Brno 2021**

## Prohlášení

Prohlašuji, že svou práci na téma *Integrace do průmyslu 4.0* jsem vypracoval samostatně pod vedením Mgr. Jaroslava Párala a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Dále prohlašuji, že tištěná i elektronická verze práce SOČ jsou shodné a nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a změně některých zákonů (autorský zákon) v platném změně.

V Brně dne: \_\_\_\_\_

---

Jakub Andrýsek

## **Poděkování**

Děkuji svému školiteli Mgr. Jaroslavovi Páralovi za obětavou pomoc, podnětné připomínky a hlavně nekonečnou trpělivost, kterou mi během práce poskytoval.

Tato práce byla provedena za finanční podpory Jihomoravského kraje.



## Anotace

Cílem práce je navrhnout ucelený systém monitorující chod pletacích strojů ve firmě a přizpůsobit ho co možná nejlépe potřebám firmy.

Můj systém jsem navrhoval na míru pro rodinou firmu na pletení ponožek. Tento systém je schopen v reálném čase zaznamenávat a následně odesílat naměřená data ze strojů na server. Pro uživatele pak systém nabízí moderní webové stránky, kde si může naměřená data přehledně zobrazit a analyzovat.

Systém se skládá ze tří částí, senzorová část, která je připojená k pletacímu stroji a odesílá data. Dále pak server, který veškerá data zpracovává a zobrazuje je uživateli. Poslední částí je podpůrný server, který se stará o aktualizaci a o kontrolu správného chodu senzorů.

## Klíčová slova

IoT, ESP32, web, PHP, Nette, databáze, open-source, průmysl 4.0, automatizace, modernizace

## **Annotation**

Zde přijde anglický překlad anotace.

Gardening is a very common hobby today. However, many people who likes this activity doesn't have enough time for it. Beside work, they have to take care of their families and after this, they don't have any time to take care of plants. My dad is exactly this kind of man. And that inspired me to create PROTOPlant – system for easy and cheap greenhouse automation.

Goal of this thesis is to create universal and available system for greenhouse automation, that will make it easier for these people to take care of their plants.

## **Keywords**

Klíčová slova - jejich překlad do angličtiny.

greenhouse automation, ESP32, PROTOPlant, automation, open-source hardware, open-source software

# Obsah

<b>Úvod</b>	<b>10</b>
<b>1 Konkurence</b>	<b>12</b>
1.1 Hardware . . . . .	12
1.1.1 PLC . . . . .	12
1.1.2 Controllino . . . . .	12
1.1.3 Hardwario . . . . .	13
1.1.4 Arduino . . . . .	13
1.1.5 Srovnání . . . . .	13
1.2 Software . . . . .	14
1.2.1 Node-RED . . . . .	14
1.2.2 Blynk . . . . .	14
1.2.3 Home Assistant . . . . .	14
1.2.4 Porovnání . . . . .	15
<b>2 Integrace do průmyslu 4.0</b>	<b>16</b>
2.1 Popis . . . . .	16
2.2 Řešení . . . . .	17
2.3 Naszaení . . . . .	17
<b>3 Senzory</b>	<b>19</b>
3.1 1. verze - univerzální sensorika . . . . .	19
3.1.1 Řídící deska . . . . .	20
3.2 Uchycení . . . . .	20

3.3	Program . . . . .	20
3.4	2. verze - speciální senzorika . . . . .	22
3.4.1	Řídící deska . . . . .	22
3.5	Uchycení . . . . .	22
3.6	Program . . . . .	23
<b>4</b>	<b>Webový server</b>	<b>24</b>
4.1	Frontend . . . . .	24
4.1.1	Bootstrap . . . . .	24
4.1.2	JavaScript . . . . .	25
4.2	Backend . . . . .	25
4.2.1	PHP . . . . .	25
4.2.2	Nette . . . . .	26
4.2.3	REST API . . . . .	26
4.3	Funkcionalita . . . . .	26
4.3.1	Podrobná statistika . . . . .	26
4.3.2	Aktuální přehledy . . . . .	26
4.3.3	Responzivní grafy . . . . .	27
4.3.4	Porovnávání směn . . . . .	27
4.3.5	Kontrola běhu stroje . . . . .	27
4.3.6	Chytrý výpočetní algoritmus . . . . .	27
4.3.7	Jednoduchý generátor zkušebních dat . . . . .	27
4.4	Webové rozhraní Pletačka IoT . . . . .	27
4.4.1	Úvodní stránka . . . . .	28
4.4.2	Správa senzorů . . . . .	30
4.4.3	Nastavení směn . . . . .	30
4.5	Databáze . . . . .	30
<b>5</b>	<b>Podpůrný server</b>	<b>33</b>
5.1	Kontrola senzorů . . . . .	33
5.2	Automatické aktualizace . . . . .	33

<b>6</b>	<b>Princip fungování Pletačka IoT</b>	<b>34</b>
6.1	Sběr dat . . . . .	34
6.2	Vyhodnocování dat . . . . .	34
6.3	Zobrazování dat . . . . .	35
6.4	Konektivita . . . . .	35
<b>7</b>	<b>Vývoj</b>	<b>36</b>
7.1	Systém Pletačka IoT verze 1.0 . . . . .	36
7.1.1	Senzory . . . . .	36
7.1.2	Web . . . . .	36
7.2	Systém Pletačka IoT verze 2.0 . . . . .	37
7.2.1	Senzory . . . . .	37
7.2.2	Web . . . . .	37
<b>8</b>	<b>Testování</b>	<b>38</b>
8.1	Domácí testování . . . . .	38
8.2	Testování ve firmě . . . . .	38
<b>9</b>	<b>Nasazení</b>	<b>40</b>
9.0.1	Zpětná vazba . . . . .	40
<b>Přílohy</b>		<b>42</b>
<b>A</b>	<b>Přílohy</b>	<b>42</b>
<b>Literatura</b>		<b>44</b>
Seznam obrázků . . . . .		45
Seznam tabulek . . . . .		46

# Úvod

Cílem práce je navrhnout ucelený systém monitorující chod pletacích strojů ve firmě a přizpůsobit ho co možná nejlépe potřebám firmy.

S nápadem vytvořit takovýto systém přišel můj děda, zakladatel firmy na výrobu ponožek. Jeho snem vždy bylo mít takový systém, který by částečně zastal monotónní lidskou práci a nahradil ji efektivní automatizací.

Můj systém jsem tedy navrhoval na míru pro rodinou firmu na pletení ponožek, ve které je okolo 25 pletacích strojů. Tento systém je schopen v reálném čase zaznamenávat a následně odesílat naměřená data ze strojů na server. Pro uživatele pak systém nabízí moderní webové stránky, kde si může naměřená data přehledně zobrazit a analyzovat.

Podle pletacích strojů na kterých tento systém běží jsem projekt pojmenoval Pletačka IoT. Systém se skládá ze tří částí, senzorová část, která je připojená k pletacímu stroji a odesílá data. Dále pak server, který veškerá data zpracovává a zobrazuje je uživateli. Poslední částí je podpůrný server, který se stará o aktualizaci a o kontrolu správného chodu senzorů.

Při vytváření tohoto projektu jsem si dal za cíl

- projekt s otevřeným zdrojovým kódem
- cenová dostupnost
- jednoduché přidání senzorů
- přehledné uživatelské rozhraní

Pro systém jsem si stanovil tyto požadavky

- Počítání upletených ponožek
- Zjišťování poruchovosti strojů
- Porovnání jednotlivých pracovních směn
- Monitorování průběhu výroby

# Kapitola 1

## Konkurence

Tento systém je velice specifický a nedá se srovnávat jako celek. Potenciální konkurenci tohoto systému jsem tedy rozdělil na dva celky.

- Hardware
- Software

### 1.1 Hardware

#### 1.1.1 PLC

PLC neboli programovatelný logický automat je průmyslový počítač k řízení automatizovaných procesů. Automaty zpracovávají data v reálném čase a s co nejkratší odezvou. PLC jsou velmi modulární a dají se skládat různě dohromady, podle potřeby uživatele.

JA  
Note:  
Přidat  
obrázky

#### 1.1.2 Controllino

Firma Controllino[4] se zabývá vývojem zařízení pro průmyslovou automatizaci založenou na platformě Arduino. Zařízení nabízí několik vstupních a výstupních pinů, pomocí kterých si uživatel může připojit své senzory a následně automatizovat některé procesy.

### **1.1.3 Hardwario**

Hardwario[3] je česká firma, která nabízí průmyslové IoT stavebnice. Cílem této firmy je nabídnou průmyslové IoT řešení, které si sami sestavíte podle svých představ. Firma se zaměřuje na nízkoenergetické moduly s vydrží několika let.

### **1.1.4 Arduino**

Arduino [2] je otevřený (open source) projekt který se díky své nízké ceně a jednoduchosti na používání rozšířil po celém světě. Arduino má v nabídce přes deset různých modelů. Desky jsou univerzální a jsou velmi často využívány na kutilské projekty. K Arduinu také existuje velké množství shieldů, které základním modulům dodávají další funkcionalitu. Desky Arduino se programují v jazyce Wiring, vytvořeném přímo pro programování mikrokontrolérů, nebo v jazyce C++.

### **1.1.5 Srovnání**

První tří zmíněné platformy jsou hojně využívány v průmyslu a řídí většinu automatizovaných procesů, jejich nasazení je složité a celé systémy jsou velmi drahé.

Požadavky na platformu

1. Připraveno na montáž na zařízení
2. Průmyslové napětí 5-25 V
3. Open source
4. Barevný displej
5. Bezdrátová konektivita ve výchozím provedení
6. Moderní konektor USB-C

<b>Hardware</b>	1	2	3	4	5	6
PLC	✓	✓	✗	✗	✗	✗
Controllino	✓	✓	✓	✗	✗	✗
Hardwario	✓	✗	✓	✗	✓	✗
Arduino	✗	✗	✓	✗	✗	✗
<b>Moje řešení</b>	✓	✓	✓	✓	✓	✓

Tabulka 1.1: Tabulka srovnání hardwarové konkurence.

## 1.2 Software

### 1.2.1 Node-RED

Node-RED je jednoduché grafické prostředí k programování IoT zařízení. Hlavní výhodou této aplikace je, že celá běží jako webová stránka. Tím umožnuje uživateli rychlou práci bez nutnosti instalovat speciální aplikace. Node-RED programování stojí na principu propojování jednotlivých uzlů. Ve složitějších projektech mohou být ovšem bloky dosti nepřehledné a složité na úpravu.

### 1.2.2 Blynk

Blynk je platforma pro vzdálené ovládání IoT projektů. Základem platformy je jednoduchá mobilní aplikace pro nastavování a vyčítání dat. Aplikace nabízí velké množství widgetů které se připínají na zobrazovací panel. Na osobní projekty do pěti zařízení je aplikace zdarma, jinak je nutné platit měsíční poplatky.

### 1.2.3 Home Assistant

Home Assistant je software pro řízení chytrých domácností. Systém dokáže pracovat s více než 1700 službami. Připojená zařízení se konfigurují pomocí textového souboru. Aplikace také dokáže integrovat mnoho rozšíření,

například ESPHome. To slouží k ovládání mikrokontrolérů ESP které jsou hojně rozšířené v kutilské komunitě. Aplikace také nabízí přehledné widgety k rychlému zobrazení nejdůležitějších dat.

#### **1.2.4 Porovnání**

Node-RED a Home Assistant jsou projekty s otevřeným zdrojovým kódem, utvářené komunitou, díky tomu jsou tyto systémy velmi modulární a rychle se rozvíjejí. Naopak Blynk je uzavřená platforma zaměřená na firmy a vývojáře. Můj systém spojuje užitečné vlastnosti ze všech těchto systémů a nabízí je jako celek v podobě systému Pletačka IoT.

# Kapitola 2

## Integrace do průmyslu 4.0

Pojem Průmysl 4.0 se do České republiky dostal okolo roku 2013 a od té doby se stále více rozšiřuje v průmyslových firmách. Jedna z klíčových částí je IoT (Internet of Things), neboli v internet věcí, který nám zajišťuje vzdálenou kontrolu a řízení strojů pomocí elektroniky, senzorů a různých softwarů. Další vlastností těchto systémů je zaznamenávání a následné ukládání dat do datových úložišť. Moderní IoT řídící systémy se snaží proniknout co nejvíce do hloubky řídících systémů a zpřesnit tak naměřená data, důležitá pro optimalizaci produkce.

### 2.1 Popis

Při návrhu mého systému jsem se snažil řídit těmito zásadami a navrhnout tak co nejmodernější a provozně efektivní systém. Základem bylo zhodnocení stávající situace a navržení možného řešení.

Jednotlivé problémy

- dlouhá doba stání nečinných strojů
- ruční počítání vyprodukovaného zboží
- absence historického přehledu produkce

## 2.2 Řešení

Mým řešením je tedy návrh moderního systému, který by celý tento provoz monitoroval a zobrazoval zaměstnavateli. Dále se také snažím o zhodnocení jednotlivých směn a jejich porovnání. Systém neustále vyvídí a rozšiřuje podle potřeb firmy.

## 2.3 Naszaení

Jak jsem již psal, tento systém je aktuálně nasazen ve firmě ROTEX Vysočina s.r.o[1], která se věnuje výrobou ponožek. Firma pracuje ve dvousměnném provozu a týdně vyprodukuje v průměru 12000 páru ponožek.

Díky mému systému by se ve firmě dala optimalizovat produkce a výkon strojů a tím zefektivnit budoucí výrobu.



Obrázek 2.1: Pletárna ponožek



Obrázek 2.2: Senzor na stroji

# Kapitola 3

## Senzory

Senzory k projektu Pletačka IoT jsou postavené na mikročipu ESP32 a moduly TTGO T-Display. Celý tento systém je navržen tak, že na každém pleťacím stroji je jedna moje elektronika. Každý z těchto senzorů má svoje jedinečné číslo, pod kterým posílá naměřená data na server. Senzor je na pájen z 5 nebo 24 voltů a má spotřebu 120 mA. Návrh senzorů i jejich software mám verzovaný nástrojem Git ve veřejném repozitáři na GitHubu.

GitHub: [Pletacka-board](#)[5]

### 3.1 1. verze - univerzální sensorika

První verzi jsem pojál jako testovací. Bylo tedy potřeba navrhnout univerzální desku a otestovat celý systém.

Při navrhování první verze senzoru jsem se držel těmito body:

- ESP32 s barevným displejem
- vstup ze 4 periferií
- vstupní napětí od 10 do 25V
- teplotní čidlo

- tři barevné diody
- čtyři uživatelská tlačítka

### 3.1.1 Řídící deska

Návrh desky jsem tvořil v aplikaci EAGLE od společnosti Autodesk. Deska má rozměry 75 na 60 mm a v každém rohu má upevňovací díry. Kabely se do desky připojují pomocí 5mm svorkovnice. Na vstupu napájení je měnič napětí který pracuje v rozsahu od 10 do 25 voltů a na výstupu dává 5V.

Řídící procesor celé desky je modul ESP32 TTGO T-Display. Tento čip také zajišťuje WiFi konektivitu s okolím a odesílá naměřená data na server. Pro univerzální detekování vstupů z periferií se využívají optočleny, které předávají signál do mikroprocesoru. K uživatelskému ovládání senzoru jsou zde čtyři programovatelná tlačítka a tři indikační diody. Aktuální naměřená data se zobrazují na displeji a informují obsluhu o zastavení stroje a počtu upletených párů. Senzor je také schopen zaznamenávat data ze čtyř vstupů a teplotu z teplotního senzoru. Viz obrázek 3.1.

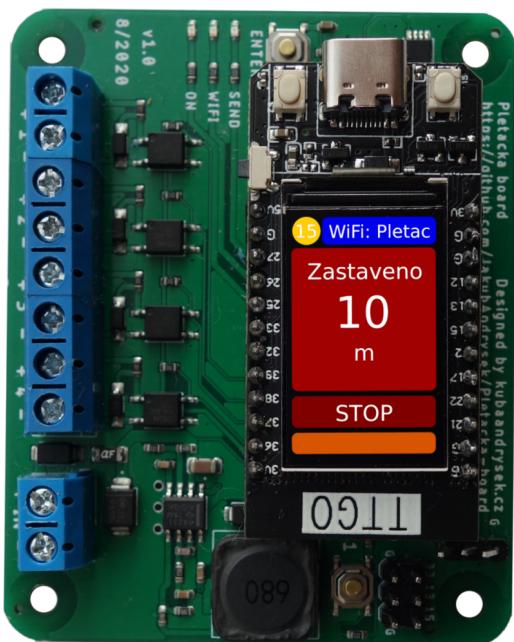
## 3.2 Uchycení

Obal řídící desky je vytisknutý na 3D tiskárně z materiálu PETG. Na přední straně je průhled z plexi skla na barevný displej a okolo něj jsou rozmištěna uživatelská tlačítka. Na boční straně krabičky jsou připravené dvě drážky na protažení stahovacích zip pásků pro uchycení na sloupek stroje. Kabely jsou poté svedeny po konstrukci stroje až k periferiím.

## 3.3 Program

K programování využívám aplikaci Visual Studio Code s rozšířením PlatformIO, které je navržena k programování mikrokontrolérů. Zdrojový kód mám napsaný v jazyce C++. Program se skládá z několika vláken, které se

pravidelně spouštějí a vykonávají. První a zároveň nejdůležitější vlákno je senzorové, zde se periodicky kontroluje stav periferií a při změně se odešle událost na server. Další vlákno zajišťuje pravidelné vykreslování dat na displej a zbylá vlákna se starají o správný chod senzoru. Software také obsahuje ladící mód ve kterém si administrátor může zobrazit stav senzoru v mobilní aplikaci a jednodušeji tak hledat potenciální chybu.



Obrázek 3.1: Senzor - 1. verze

## 3.4 2. verze - speciální senzorika

Po měsíci testování jsem zhodnotil využití jednotlivých součástek a následně jsem vytvořil nový seznam požadavků, přizpůsobený pro lepší chod senzoru. Zařízení je díky tomu mnohem menší, levnější a softwarově rychlejší.

- vstup pouze ze 2 periferií
- vstupní napětí již od 5V
- zredukování rozměrů
- moderní USB-C konektor
- zredukování na dvě tlačítka a dvě indikační diody
- možnost přímého napájení senzoru bez měniče

### 3.4.1 Řídící deska

Návrh druhé desky jsem se rozhodl udělat v open source aplikaci KiCad. Tato aplikace podporuje mnoho rozšíření, která velmi zpříjemní návrh a zjednoduší přípravu podkladů.

V novém návrhu jsem se především zaměřoval na rozměr desky, ten aktuálně činí  $32 \times 76\text{mm}$ , což je o 46 procent menší plocha než u první verze.

Deska si zachovala stejný procesor ESP32 s displejem, ale přišla o dvě tlačítka a jednu indikační diodu. V senzoru se také změnilo zapojení měniče napětí, ten nově dokáže pracovat již od 5V, které následně mění na 3,3V. Na bočních stranách desky vznikla také nová "křidélka" pro zasunutí do vylepšeného krytu.

## 3.5 Uchycení

Druhá verze využívá stejného principu uchycení, jako ta předchozí. Mění se zde však spojení krabičky se senzorovou deskou. V nové verzi jsem desku

navrhl tak, aby se dala jednoduše zasunout do kolejnic které jsou předtištěné v krabičce a následně zafixovat šroubkem ze zadní strany. To umožňuje jednoduchou montáž a rychlé připojení. Tento návrh už má také vyřešené zafixování kabelů ke konstrukci krabičky pomocí 3D tištěných svěrek.

## 3.6 Program

Program druhé verze vychází z minulé, ale přináší s sebou nové funkce a vylepšuje stávající. Novou funkcionalitou je například automatická aktualizace programu přes WiFi, kterou nadále zdokonaluji. Další vylepšení jsem přidal k displeji, který dokáže zobrazit více údajů a automaticky mezi nimi přepínat.



Obrázek 3.2: Senzor - 2. verze

# Kapitola 4

## Webový server

Webový server je nejdůležitější a nejobsáhléjší část celého systému. Webový server mám nasazený na mikropočítači Raspberry Pi 4 Modelu B který má 8 GB operační paměti. Toto zařízení jsem zvolil hlavně kvůli nízké spotřebě elektrické energie a velké komunitě lidí, kteří tento mikropočítač využívají.

Na zařízení běží operační systém Raspberry Pi OS s grafickým rozhraním. Webové stránky běží na HTTP serveru Apache2 a PHP 7.3. Jako databázový systém využívám MariaDB. Server běží lokálně uvnitř firmy v zabezpečené síti, díky čemuž je systém rychlý a nezávislý na internetovém připojení. Celý webový server mám verzovaný také na GitHubu.

GitHub: [Pletacka-website](#)[6]

### 4.1 Frontend

Frontend je vizuální část webové stránky zobrazená uživatelem. Pomocí frontendu se na obrazovku vykresluje veškerý text a jednotlivé prvky stránky.

#### 4.1.1 Bootstrap

Bootstrap je knihovna sloužící k jednoduchému a rychlému vytvoření responsivních webových stránek. Díky této knihovně jsou stránky správně zobrazeny i na mobilních zařízeních. Tento nástroj se vyvíjí od roku 2011 a je

pod otevřenou licencí. Webový server využívá Bootstrap verze čtyři.

### 4.1.2 JavaScript

Na frontendu používám JavaScript společně s technologií AJAX pro aktualizaci částí stránek. AJAX umožňuje překreslovat jen určitou část obsahu stránky bez nutnosti načíst celou stránku znovu. Tím se zásadně zrychluje načítání a interaktivita stránek. Dochází i k značné úspoře přenesených dat. K tomuto efektivnímu překreslování slouží knihovna Naja[7], kterou napsal český vývojář Jiří Pudil. Knihovna také nabízí jednoduchou integraci do PHP frameworku Nette, o kterém budu psát dále.

## 4.2 Backend

Je to nejobsáhlejší část celé této práce. Backend je serverová část webových stránek, neběží tedy u vás na počítači jako frontend, ale na webovém serveru. Celý backend systému Pletačka IoT jsem napsal v programovacím jazyce PHP a ve frameworku Nette[8], který nabízí ucelenou sadu nástrojů k tvorbě webu. Backend se stará o přijímání dotazů ze senzorů a následný zápis do databáze, pohání celý webový server a vytváří databázové výběry. Nejdříve zde popíšu použité technologie a následně rozeberu jednotlivé stránky aplikace.

### 4.2.1 PHP

Webovou aplikaci programuji v PHP ve verzi 7.3. Jako programovací studio jsem zvolil studentskou verzi aplikace PHPStorm, která je velmi mocným nástrojem při tvorbě webu. Testovací verze aplikace mám spuštěnou na svém počítači kde také celý tento systém vyvíjím.

Pro snadnější ladění chyb používám Xdebug, díky kterému si můžu krokovat jednotlivé řádky kódu a rychleji tak nalézt chybu.

Jako systém pro správu balíčků používám nástroj Composer, který se ovládá z terminálu pomocí jednoduchých příkazů. Umožňuje rychlou definici

závislostí a aktualizaci všech modulů pomocí jednoho příkazu.

### 4.2.2 Nette

Nette je webový framework vyvíjený komunitou. Vznikl v České republice a jeho zakladatelem je David Grudl. Nabízí vlastní šablonovací jazyk, na jednoduché a efektivní vykreslování webových stránek. Nette disponuje obsáhlou a velmi dobře zpracovanou dokumentací, ale také velkou komunitou lidí kteří s tímto frameworkm pracují a velmi dobře mu rozumí.

### 4.2.3 REST API

REST API je sada URL identifikátorů sloužících ke komunikaci s webovou stránkou. Webová stránka Pletačka IoT obsahuje základní sadu API. Primárně ji využívají senzory k odesílání naměřených dat a ke zpětnému posílání odpovědí do senzoru. Druhé využití API je k vytváření databázových výběrů, to je voláno nástrojem na automatizaci procesů v nastavený čas.

JPA  
Note:  
Chtěl  
bych se  
ještě  
pobavit  
o defini-  
ci/po-  
pisu  
Web  
API

## 4.3 Funkcionalita

### 4.3.1 Podrobná statistika

Díky tomuto systému má uživatel kompletní přehled o každém zastavení stroje a upletené ponožce. Systém se každé den automaticky zálohují a ukládá data nezáložní disk.

### 4.3.2 Aktuální přehledy

Systém zobrazuje vždy aktuální přehledy naměřených dat pomocí předgenerovaných výběrů.

### **4.3.3 Responzivní grafy**

Číselné přehledy jsou doplněné o jednoduché grafy, které graficky zobrazují naměřená data. Hlavním cílem grafů je co nejlépe uživateli zobrazit porovnání směn porovnání s dalšími pracovními směnami.

### **4.3.4 Porovnávání směn**

### **4.3.5 Kontrola běhu stroje**

### **4.3.6 Chytrý výpočetní algoritmus**

Veškerá naměřená data jsou analyzována mým výpočetním algoritmem. Algoritmus přijímá uložená data z databáze, ze kterých postupným procházením vypočítává pracovní statistiky, které následně ukládá do užších výběrů.

### **4.3.7 Jednoduchý generátor zkušebních dat**

Pro otestování systému jsem připravil jednoduchý generátor dat, který dokáže nasimulovat reálné senzory.

## **4.4 Webové rozhraní Pletačka IoT**

Každá stránka stránka je rozdělena na tři části. Záhlaví, to obsahuje logo a odkazy na nejpoužívanější stránky. Druhou částí jsou samotné webové stránky které budou popsány v dalších odstavcích. Poslední částí je minimalistické zápatí s copyright znakem.

Stránky Pletačky jsem navrhoval tak, aby splňovaly tyto parametry:

- jednoduché rozhraní pro uživatele
- přehledné zobrazení dat
- zobrazovat pouze užitečná dat

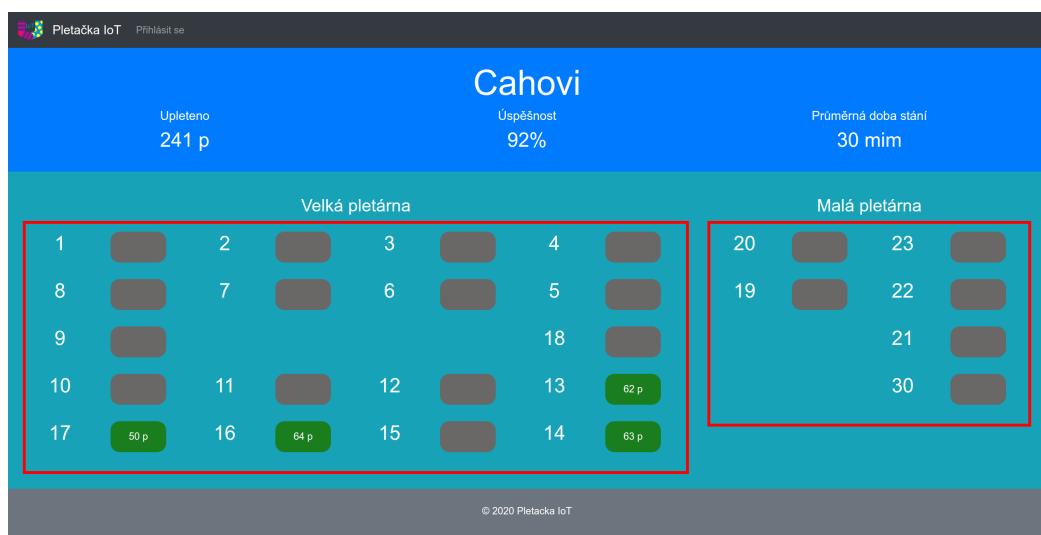
JA  
Note:  
Doplnit  
obrázky  
stránek  
pod  
kapitolu  
nebo  
jeden  
list  
s fot-  
kami

- rychlá editace senzorů
- využití číselných identifikátorů

#### 4.4.1 Úvodní stránka

V horní části úvodní stránky se vypisují tři nejpodstatnější údaje. Jde o celkový počet upletených párů za aktuální směnu. Dále pak úspěšnost vypočítávanou z času zastavení stroje a z celkové času zapnutí stroje. Posledním údajem je průměrná doba stání jednoho stroje.

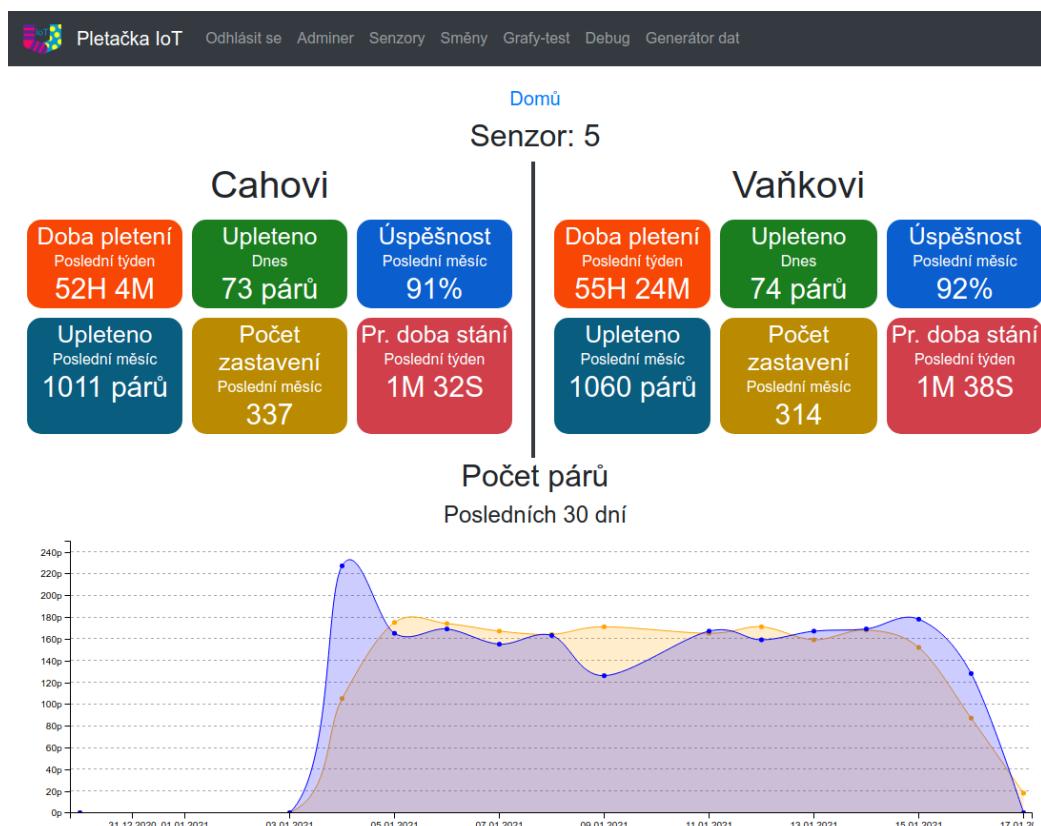
Pod těmito čísly se zobrazuje tabulka s barevnými obdélníky, kde každý představuje jeden stroj. Barva obdélníků udává aktuální stav stroje a text v pozadí tuto informaci doplňuje.



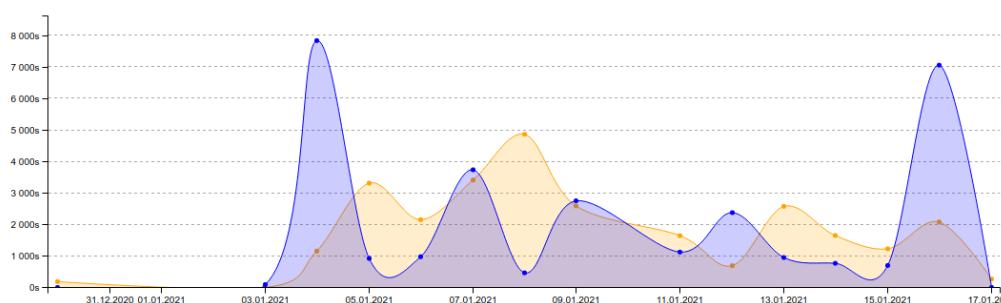
Obrázek 4.1: Úvodní stránka

Po kliknutí na senzor na úvodní stránce, se zobrazí data o právě vybraném stroji. Veškerá data jsou rozdělena do dvou sloupců podle pracovních směn. To umožňuje zaměstnavateli jednoduché porovnávání pracovních směn. V úvodu každého sloupce je obecný přehled naměřených dat za různá období. Pod nimi je přehled v grafech a porovnání nejdůležitější údajů.

JPA  
Note:  
Aktuali-  
zovat  
grafy-  
/obrázky



Doba stání  
Posledních 30 dní



Obrázek 4.2: Přehled ze senzoru

#### 4.4.2 Správa senzorů

Pro vstup do této sekce je nutné uživatelské přihlášení do systému. Stránka pak nabízí přehled senzorů s jednotlivými možnostmi úpravy (viz obrázek 4.3).

Výpis senzorů					
<a href="#">Přidat senzor</a>					
<a href="#">Aktualizovat</a> 12/15/2020 08:52:28					
Číslo	Popis	Datum změny	Zobrazit	Upravit	Smažat
1	Pletacka - 1	2020-11-08 00:14:59	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
2	Pletacka - 2	2020-11-08 00:14:59	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
3	Pletacka - 3	2020-11-08 00:15:03	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
4	Pletacka - 4	2020-11-08 00:15:03	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
5	Pletacka - 5	2020-11-08 00:15:04	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
6	Pletacka - 6	2020-11-08 00:15:04	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
7	Pletacka - 7	2020-11-08 00:15:05	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>
8	Pletacka - 8	2020-11-08 00:15:07	<a href="#">Zobrazit</a>	<a href="#">Upravit</a>	<a href="#">Smažat</a>

Obrázek 4.3: Správa senzorů

#### 4.4.3 Nastavení směn

Jednoduchá stránka na které se nastavuje pořadí směn. Střídání směn probíhá pravidelně po týdnech, proto je nastavení velmi jednoduché (viz obrázek 4.4)

### 4.5 Databáze

Databáze je rozdělená do dvou skupin tabulek.

První skupina tabulek je nastavovací, jedná se o hlavní nastavení webu, nastavení směn a o tabulkou s uživatelem a jejich oprávněním.

Druhá skupina je senzorová. Každý senzor zde má pět tabulek na ukládání svých dat. První senzorová tabulka ukládá čistá nezpracovaná data posílaná

**Nastavení směn**

Rok:

Lichá směna:  ▼

**Přehled nastavení**

Rok	Lichá směna
2019	Cahovi
2020	Vaňkovi
2021	Cahovi
2022	Cahovi
2023	Vaňkovi

Obrázek 4.4: Nastavení směn

přímo ze senzoru. Zbylé čtyři tabulky jsou databázové výběry různých časových úseků, jde o výběr hodinový, denní, měsíční a roční. Tyto tabulky se vytvářejí automaticky pomocí výběrového API. Struktura tabulek je vyobrazena ve schématu 4.5.

JPA  
Note:  
Možná  
bychom  
se mohli  
pobavit  
o úpravě  
"Struk-  
tura  
da-  
tabáze" obrázku.

Databáze				
Nastavení	Směny	Uživatelé	Senzory	Výběr senzoru
id	id	id	číslo	id
název webu	rok	jméno	popis	čas
popis webu	týden	heslo	datum vytvoření	směna
Název směny A	Směna A	oprávnění		čas stání
Název směny B	Směna B			
				Senzory
				čas práce
				celkový čas
				stav počet upletených párů
				čas počet zastavení

Obrázek 4.5: Struktura databáze

# Kapitola 5

## Podpůrný server

Podpůrný server vznikl jako rozšíření pro senzory. Server je naprogramován v Pythonu a běží na Raspberry Pi společně s webovým serverem.

Zdrojový kód na Githubu: [Pletacka-python-server](#)[9]

### 5.1 Kontrola senzorů

Hlavním úkolem tohoto serveru je detekce zapnutých senzorů. Na serveru běží takzvaný Watchdog, jde o periodickou smyčku, který každé čtyři vteřiny čeká na zprávu ze senzoru. Touto zprávou se senzor nahlásí, že je zapnutý. Pokud takováto zpráva nedojde deset vteřin, je senzor prohlášen za vypnutý a v databázi se označí jako neaktivní.

### 5.2 Automatické aktualizace

Bezdrátová aktualizace senzorů je nová funkciálnita, kterou nadále vyvíjím a rozšiřuji. Senzory aktuálně podporují rychlou aktualizaci přes WiFi ze vzdáleného počítače. V počítači stačí vybrat číslo senzoru a nová verze programu se pomocí WiFi připojení nahraje do senzoru.

V nové verzi přibude také hromadná aktualizace senzorů a systém na udržování aktuálních verzí systému ve všech senzorech.

# Kapitola 6

## Princip fungování Pletačka IoT

V předchozích kapitolách byly popsány jednotlivé část systému Pletačka IoT. V této kapitole bude celý systém popsán jako celek.

### 6.1 Sběr dat

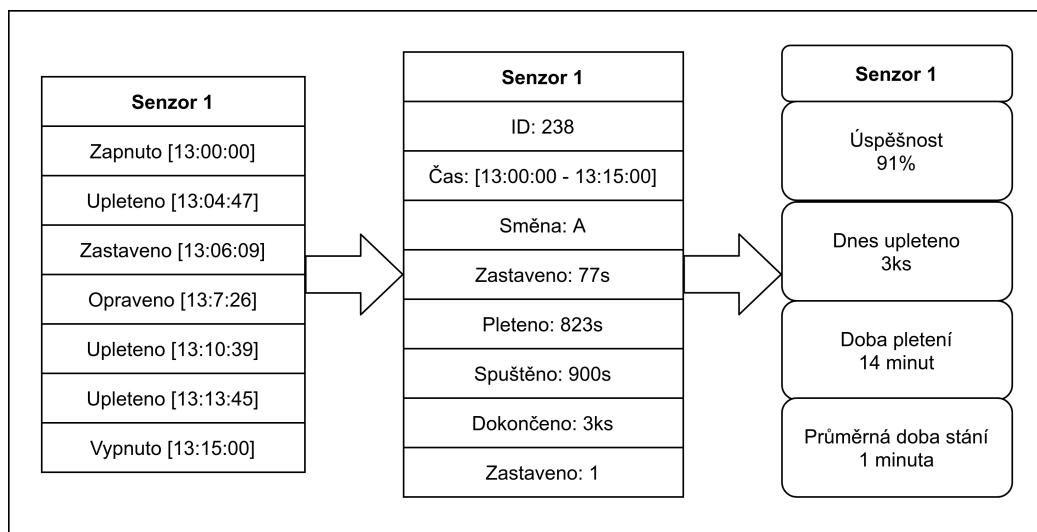
První, a tou nejdůležitější částí, je získávání dat pomocí senzorů. Jakmile senzor zaznamená jakoukoliv změnu, okamžitě tuto zprávu odesílá na server. Odesílání probíhá skrze senzorové API, kde se nejdříve senzor ověří a následně se stav zapíše do databáze k příslušnému senzoru. Po zapsání do databáze se vrátí do senzoru zpráva o provedení zápisu.

### 6.2 Vyhodnocování dat

Dalším krokem je zpracovávání surových dat z databáze. K tomuto účelu běží na serveru výběrové API, které je automaticky spouštěné v nastavený čas. Jde o generování širších výběrů dat, hodinové, denní, měsíční a roční výběry. Tyto výběry se následně ukládají do databáze k danému senzoru. Generování těchto dat probíhá převážně v noci, kdy je server nejméně vytížen.

## 6.3 Zobrazování dat

Posledním krokem je zobrazení dat uživateli. Je to jediná část, se kterou se běžný uživatel dostane do kontaktu. Proto je nutné, aby zobrazení bylo co nejrychlejší a pro uživatele co nejpříjemnější. K rychlému zobrazování se využívají předgenerované výběry, ke kterým se dopočítají dosud nezpracovaná data a celý výsledek se zobrazí uživateli.



Obrázek 6.1: Zpracování dat

## 6.4 Konektivita

Webové stránky se dají jednoduše zobrazit na počítači či notebooku. Stránky jsou responzivní a lze je používat i na mobilních zařízeních. Přístup k webu je pouze z vnitřní sítě firmy, to zajišťuje základní bezpečnost pro systém.

# Kapitola 7

## Vývoj

Na této práci jsem začal pracovat v únoru 2020, kdy jsem si jako úplný nováček četl dokumentaci k jazyku PHP. Původní verzi webového rozhraní jsem začal navrhovat v čistém PHP. Tento způsob byl však velmi zdlouhavý a neefektivní. Po měsíci práce v čistém PHP jsem přešel na framework Nette, který mi práci zjednodušil a posunul mě velmi rychle dál.

### 7.1 Systém Pletačka IoT verze 1.0

Tato verze byla vydána začátkem července, kdy už systém uměl pracovat s virtuálními senzory.

#### 7.1.1 Senzory

Souběžně s programováním webu jsem pracoval na softwaru pro senzory. V této době byly senzory schopné posílat data na server, ale neměli žádný grafický výstup ani nepodporovaly interakci s uživatelem.

#### 7.1.2 Web

Vznikla základní kostra webu a postupně vznikaly první stránky. Data ze senzorů se zatím pouze ukládala do databáze a web s nimi zatím neuměl

pracovat. Začínal se vyvíjet systém na zpracovávání údajů ze senzorů.

## 7.2 Systém Pletačka IoT verze 2.0

Druhá verze přinesla velké rozšíření systému. Tato verze je produkčně nasazena od půlky prosince a do teď běží bez větších problémů.

### 7.2.1 Senzory

Propojení systému s novou verzí senzorů, které nově podporují nahrávání aktualizací přes WiFi, mají přehlednější zobrazování dat na displej a dokážou upozornit na výpadek sítě.

### 7.2.2 Web

Největší proměnou prošlo webové rozhraní. Domovská stránka má přehledné zobrazování stavů senzorů. U senzorů se zobrazují důležitá data a pomocí grafů se dají data jednoduše porovnávat. Přibylo nastavování směn a hromadné přidávání senzorů.

# Kapitola 8

## Testování

Testování systému je jedna z nejdůležitějších částí navrhování jakýchkoliv systémů. Správným otestováním by se měla odladit většina potenciálních chyb.

JPA

Note:

Měli

bychom

probrat

tuto

kapitolu

### 8.1 Domácí testování

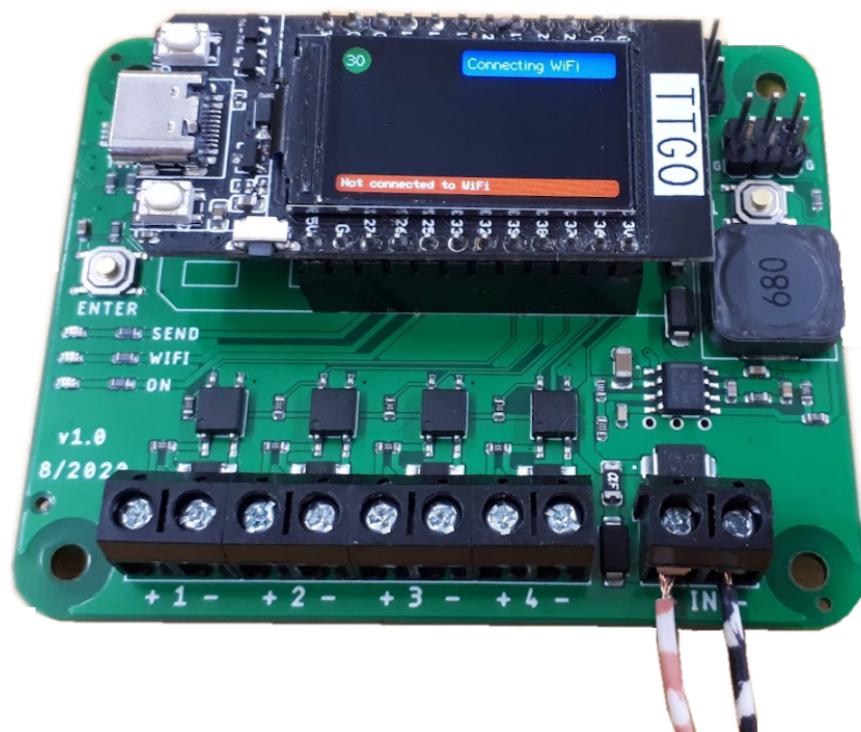
Průběžné testování částí webu probíhalo již při vývoji a kontrolovalo správné fungování nových funkcí.

Později bylo nutné nachystat rozsáhlejší testy a připravit jim testovací databázi s fiktivními daty. Tímto způsobem jsem například kontroloval správnost běhu funkce pro výpočet času zastavení stroje.

### 8.2 Testování ve firmě

V bodě kdy byly odladěny chyby jsem systém nasadil na dva pletací stroje. Nově nasbíraná data byla již reálná a dalo se na nich postavit nové testování. Senzory jsem tedy nechal několik dní sbírat údaje o upletených ponožkách a následně jsem nad nimi spustil generování uživatelsky čitelných dat.

Od půlky prosince probíhá dlouhodobé testování bez zásahu do vygenerovaných dat. Naměřené údaje pravidelně stahuji a kontroluji jejich správnost.



Obrázek 8.1: Testování senzoru

# Kapitola 9

## Nasazení

První nasazení na pletací stroje proběhlo v květnu roku 2020. V první fázi jsem osadil 2 pletací stroje a sbíral z nich data.

Druhé nasazení dalších senzorů proběhlo koncem září, kdy byly doosazeny další dva stroje. Bylo tedy v provozu čtyři senzory a probíhal vývoj nových.

V půlce prosince jsem připravil dalších šest senzorů, které jsem osadil na další stroje.

### 9.0.1 Zpětná vazba

Zde bude zpětná vazba...

  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam nunc magna, sollicitudin id leo eu, viverra congue risus. Aliquam consequat ipsum ut erat placerat consequat nec at diam. Aenean est odio, molestie sit amet nunc in, pretium luctus elit.

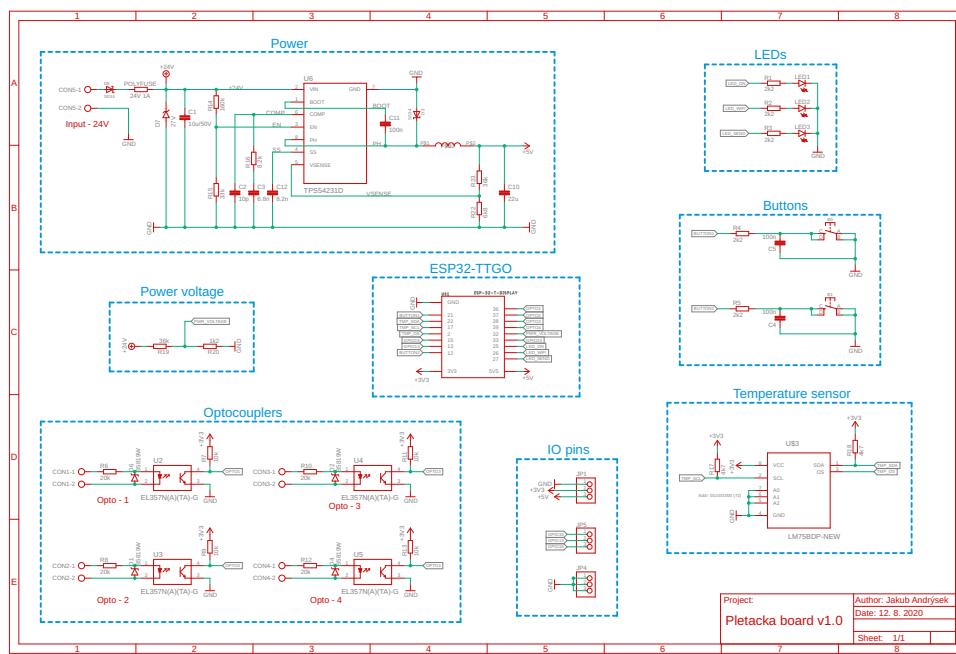
# Závěr

V závěru by mělo být:

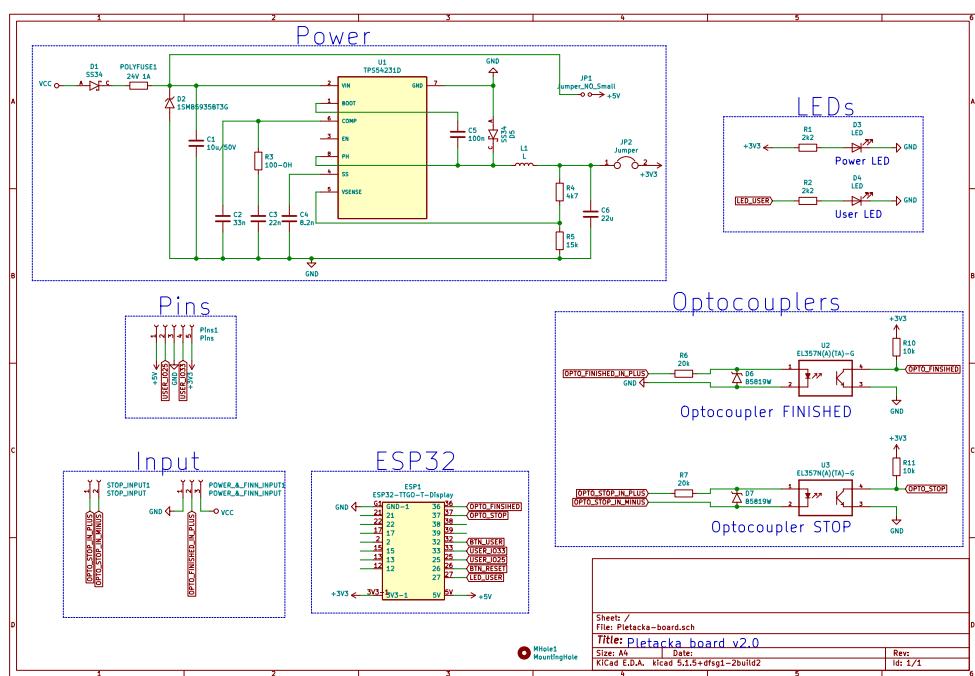
- Rekapitulace cíle práce
- Dosáhnul jsem jej? Ano, nebo ne?
- Zhodnocení průběhu práce
- Co mi práce dala?

# Příloha A

## Přílohy



Obrázek A.1: Schéma senzoru 1. verze



Obrázek A.2: Schéma senzoru 2. verze

# Literatura

1. ROTEX VYSOČINA S.R.O. *Webové stránky firmy ROTEX Vysočina* [online] [cit. 2020-12-08]. Dostupné z: <http://www.rotexvysocina.cz/>.
2. ARDUINO. *Webové stránky Arduino* [online] [cit. 2020-11-15]. Dostupné z: <https://www.arduino.cc>.
3. HARDWARIO. *Webové stránky Hardwario* [online] [cit. 2020-11-15]. Dostupné z: <https://www.hardwario.com>.
4. CONTROLLINO. *Webové stránky Controllino* [online] [cit. 2020-11-15]. Dostupné z: <https://www.controllino.com>.
5. JAKUB ANDRÝSEK. *Návrh a zdrojový kód senzorů* [online] [cit. 2020-12-08]. Dostupné z: <https://github.com/Pletacka-IoT/Pletacka-board>.
6. JAKUB ANDRÝSEK. *Zdrojový kód webového serveru* [online] [cit. 2020-12-08]. Dostupné z: <https://github.com/Pletacka-IoT/Pletacka-website>.
7. JIŘÍ PUDIL. *Webové stránky knihovny Naja* [online] [cit. 2020-12-01]. Dostupné z: <https://naja.js.org>.
8. DAVID GRUDL. *Webové stránky frameworku Nette* [online] [cit. 2020-12-01]. Dostupné z: <https://nette.org>.
9. JAKUB ANDRÝSEK. *Zdrojový kód podpůrného serveru* [online] [cit. 2020-12-08]. Dostupné z: <https://github.com/Pletacka-IoT/Pletacka-python-server>.

# Seznam obrázků

2.1	Pletárna ponožek . . . . .	17
2.2	Senzor na stroji . . . . .	18
3.1	Senzor - 1. verze . . . . .	21
3.2	Senzor - 2. verze . . . . .	23
4.1	Úvodní stránka . . . . .	28
4.2	Přehled ze senzoru . . . . .	29
4.3	Správa senzorů . . . . .	30
4.4	Nastavení směn . . . . .	31
4.5	Struktura databáze . . . . .	32
6.1	Zpracování dat . . . . .	35
8.1	Testování senzoru . . . . .	39
A.1	Schéma senzoru 1. verze . . . . .	42
A.2	Schéma senzoru 2. verze . . . . .	43

# **Seznam tabulek**

1.1	Tabulka srovnání hardwarové konkurence.	14
-----	---	----