Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Využití Raspberry Pi pro měření efektivity výroby

Bakalářská práce

Autor: David Drahokoupil
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Jan Štěpán

Hradec Králové Červenec 2019

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 26.7.2019 David Drahokoupil

Poděkování:

Tímto bych chtěl poděkovat vedoucímu bakalářské práce Ing. Janu Štěpánovi za metodické vedení, praktické i teoretické rady v průběhu tvorby této práce. Dále bych chtěl poděkovat firmě Bühler Motor s.r.o. za možnost zpracovávat bakalářskou práci a především IT oddělení za ochotu a poskytnutí odborných informací.

Anotace

Tato práce pojednává o využití jednodeskového počítače Raspberry Pi na kontrolu vyrobených počtů kusů, zjištění poruchovosti a prostojů stroje a operátora ve výrobní firmě. Je popsán celkový návrh jak hardwaru, tak i softwaru. Konečný projekt by měl objasnit, jaká data budou sledována a co vše bude potřebné při pozdějším nasazení komerčního řešení. Toto zařízení bude reálně nasazeno ve výrobní firmě Bühler Motor s.r.o., která se zabývá výrobou komponent převážně do automobilového průmyslu. První část práce se zabývá hardwarem zařízení, tedy samotným Raspberry Pi, plošným spojem, klávesnicí, RFID čtečkou, displejem, tlačítky či diodami. Druhá část práce pojednává o použitých technologiích.

Annotation

Title: Use of Raspberry Pi to measure production efficiency

This work deals with the use of Raspberry Pi to check the number of produced pieces, detect failure rate, machine downtime and operator idle time in the production company. This work describe the overall implementation of hardware and software. The final project should explain to us what data we want to track and what we need to do later on, when we deploy a commercial solution. This equipment will be actually deploy in the production company Bühler Motor, which mainly manufactures components for automotive industry. The first part of the thesis deal with the hardware, so with Raspberry Pi, PCB, keyboard, RFID reader, display, buttons or diodes. The second part of the thesis deal with used technologies.

Obsah

Seznam obrázků

Seznam ukázek kódu

Seznam Zkratek a značek

OEE	Overall Equipment Effectiveness/Celková efektivnost zařízení
FPY	First Pass Yield/Ukazatel kvality výroby
(I)IoT	(Industry) Internet of Things/(Výrobní) Internet věcí
PCB	Printed Circuit Board/Tištěný plošný spoj
OLED	Organic Light Emitting Diodes/Organické elektroluminiscenční diody
RFID	Radio Frequency Identification/Identifikace na rádiové frekvenci
SQL	Strucuted Query Language/Strukturovaný dotazovací jazyk
SoC	System-on-chip/Počítač integrovaný v jediném integrovaném obvodu
USB	Universal Serial Bus/Univerzální sériová sběrnice
NOOBS	New Out Of the Box Software/Software z krabice
ARM	Advanced RISC Machine/Označení architektury počítačů
GPIO	General-purpose input-output /Univerzální vstupní-výstupní pin
DSI	Display Serial Interface/Sériové rozhraní pro displej
PoE	Power over Ethernet/Napájení po síťovém kabelu
SSH	Secured Shell/Zabezpečený komunikační protokol

TCP/IP	Transmission Control Protocol/Internet Protocol Přenosový protokol/protokol síťové vrstvy
ISCSI	Internet Small System Interface/Síťový protokol umožňující připojit úložný prostor
NAS	Network Attached Storage/Síťové uložiště
HAT	Hardware Attached on top/Rozšiřující deska
I2C	Inter-Integrated Circuit/Sériová sběrnice
FID	Firearms identification/RFID čip
VMWARE	Virtual Machine Ware/Virtualizační nástroj
PLA	Polyactic Acid
C/C++	Programming language/Programovací jazyk
PHP	Hypertext Preprocessor/Hypertextový preprocesor
HTML	Hypertext Markup Language/Značkovací jazyk pro tvorbu webu
PIL	Python Imaging Library
IP	Internet Protocol/Internetový protokol
SPI	Serial Peripheral Interface/Sériové periferní rozhraní
URL	Uniform Resource Locator/Jednotná adresa zdroj

1 Úvod

Modernizace výrobního prostředí a zlepšování kvality výrobků je na denním pořádku většiny výrobních firem. S rostoucí kvalitou výrobků je nutné zefektivňovat i samotnou výrobu. Dnešní moderní firmy mají mnoho výrobních zařízení, které usnadňují lidem práci a jsou schopné pracovat 24 hodin denně, 7dní v týdnu. Při využití takového zařízení, je ho nutné přizpůsobit tak, aby pracovalo co nejefektivněji. To znamená snížit prostoje jak samotného zařízení, tak i personálu obsluhující stroj. Dále je také nutné předejít poruchám stroje a zmetkovitosti výrobků. Starší výrobní linky nemají ani žádné počítadlo hotových kusů, takže není přesně známá produkce za časovou jednotku. Nejsou známi ani prostoje strojů ať už z důvodu poruchy nebo nepřítomnosti operátora.

Většina výrobních firem se snaží sbírat co nejvíce výrobních dat, tedy připojit svoji firmu do Internetu věcí (Internet of Things, IoT). IoT je nový trend komunikace a kontroly předmětů mezi sebou nebo s člověkem a to prostřednictvím bezdrátového přenosu dat a internetu. Propojená zařízení umožňují sběr velkého množství dat, která jsou dále zpracovávána. Zpracovaná data se využívají v mnoha oblastech jako je logistika, zdravotnictví, energetika, doprava, meteorologie atd. Technologie IoT se také uplatňuje v oboru chytrých elektroinstalací [1]. K pojmu internet věcí se váže také pojem Industry 4.0, což je oficiální vyjádření 4. průmyslové revoluce, která zahrnuje kompletní digitalizaci, robotizaci a automatizaci lidských činností. Industry 4.0 by mělo zajistit větší rychlost a efektivitu výroby přesnějších, osobitějších, spolehlivějších a levnějších produktů. V průmyslovém oboru se jedná o nahrazení manuální lidské práce robotizací a současné manuální zadávání výrobních dat a postupů bude nahrazeno automatických předáváním informací mezi materiály a jednotlivými výrobními stroji, sklady a tak dále [2].

Rychlý růst IIot (průmyslový internet věcí) je zřejmý. V polovině minulého roku investovalo 91% výrobců do digitalizace výroby. Vzhledem ke slučování hardwaru a softwaru se strojní zařízení stává chytřejším, což umožňuje prediktivní údržbu. Umělá inteligence dokáže detekovat, hlásit a dokonce i opravit výrobní problémy. Velké množství dat, aditivní výroba a internet věcí také pomáhají vyvíjet složitější a vysoce kvalitní produkty. Díky prediktivní údržbě a plynulejší automatizované

výrobě je vysoká úspora energie. Výrobci mohou efektivně měřit konečnou cenu na základě využití celého výkonu stroje [3].

V úvodu práce je zaveden pojem OEE, které slouží pro výpočet efektivnosti výroby. První část práce je věnována veškerému hardware zařízení. Nejprve samotnému Raspberry Pi, které se stará o chod celého zařízení, přijímá data ze senzorů a vstupních zařízení, zpracovává je v aplikaci, která je napsána v jazyce Python a zpracovaná data zasílá na PHP server, který veškerá data ze všech zařízení zasílá do MySQL databáze. Dále jsou hardwarové části popsány vstupní komponenty jako je klávesnice, RFID čtečka a tlačítka. Závěr první části se zabývá výstupními komponenty, tedy displejem a stavovými led diodami.

Druhá část je věnována programové části projektu. Je zde popsáno nastavení Raspberry Pi a samotná aplikace v programovacím jazyce Python, webová aplikace hostovaná na Microsoft Windows serveru psaná v programovacím jazyce PHP. Dále je zde popsána databáze MySQL a výsledné zpracování a zobrazení dat ve Splunku. Závěr práce je věnován zhodnocení navrženého řešení.

2 Cíl práce a metodika zpracování

Cílem této práce je zjištění všech požadavků nutných pro nasazení komerčního řešení. Důraz je kladen především na všechna data, která jsou nutná pro zefektivnění a monitorování výrobního procesu.

Pro potřeby zjišťování dat bude sestrojeno zařízení, které bude přímo na výrobní lince počítat vyrobené kusy pomocí laserové brány. V zařízení budou také zaznamenávána data o obsluze a technikovi stroje. Sledována budou také výrobní data jednotlivých výrobků a chybová data stroje. Tyto údaje budou shromažďovány, vyhodnocovány a následně zobrazovány přímo na lince. Získaná data povedou k přehledu stávajících výrobních procesů a následné optimalizaci těchto procesů a výroby samotné. Bude se také jednat o první krok Industry 4.0 a obecně chytré výroby. Do budoucna se budou moci sledovat i jiná data, která budou například predikovat poruchu zařízení a tím se zamezí zmetkovitost a nepřesnost výroby. Pro tyto potřeby je vhodné nasazení neuronové sítě a strojového učení.

Návrh zařízení bude obsahovat jak kompletní hardware, tak i software. Po celém procesu implementace a po získání dostatečného množství dat budou tyto data použita k nasazení komerčního řešení.

Zařízení bude reálně umístěno ve výrobní firmě Bühler Motor. Jedná se o rodinnou německou firmu s více než 150-ti letou tradicí. Orientuje se především na automobilový průmysl, kde dodává malé převodové motory a vodní čerpadla pro velké firmy.

3 OEE

Pro měření efektivnosti výrobních strojů i samotné výroby je postup nazvaný zkratkou OEE. OEE neboli celková efektivita zařízení je zlatým standardem pro měření produktivity výroby. Jednoduše řečeno – určuje procento výrobního času, který je skutečně produktivní. OEE s hodnotou 100% znamená, že jsou vyráběny pouze dobré díly a to co nejrychleji bez zastavení stroje či linky. Tato hodnota je rozdělena na kvalitu a výkon, kde 100% kvalita jsou pouze dobré díly a 100% výkon je čas bez zastavení. Měření OEE je nejlepší výrobní praxe. Měřením OEE jsou získány důležité informace o tom, jak systematicky zlepšovat výrobní proces. OEE je jedinou nejlepší metrikou pro identifikaci ztrát, pokroku v benchmarkingu a zvyšování produktivity výrobních zařízení. Mezi základní pojmy patří dostupnost, výkon a kvalita.

Dostupnost zohledňuje neplánované a plánované zastávky. Hodnota 100% dostupnosti znamená, že proces je vždy spuštěn během plánovaného výrobního času. Dostupnost bere v úvahu ztrátu dostupnosti, která zahrnuje všechny událost, které zastaví plánovanou produkci na znatelnou dobu. Mezi příklady, které vytvářejí ztrátu dostupnosti, patří neplánované odstávky a plánované odstávky, mezi které patří například doba přechodu. Doba přechodu je zahrnuta v OEE analýze, protože se jedná o čas, který by mohl být použit pro výrobu. I když není možné eliminovat dobu přechodu, může být ve většině případů výrazně snížen. Zbývající čas po odečtení ztráty dostupnosti se nazývá doba běhu.

Výkon zohledňuje pomalé cykly a malé zastávky. Výkonnostní hodnota 100% znamená, že když je proces spuštěn, běží co nejrychleji. Výkon bere v úvahu pokles výkonnosti, který je způsoben, že výrobní proces běží při nižších rychlostech, než je maximální možná rychlost, včetně pomalých cyklů a malých výrobních pauz. Mezi příklady, které způsobují ztrátu výkonu, patří opotřebení stroje, nestandartní materiály a tak dále. Zbývající čas po odečtení poklesu výkonnosti se nazývá čistá doba běhu.

Kvalita bere v úvahu vady, včetně částí, které vyžadují přepracování. Hodnota 100% znamená, že nedochází k chybám a jsou vyráběny pouze dobré kusy. Kvalita odpovídá za vyrobené díly, které nesplňují normy kvality. Mezi příklady, které

vytvářejí ztrátu kvality, patří šrot a části, které je třeba přepracovat. Kvalita OEE je podobná kvalitě FPY v tom, že definuje dobré kusy jako kusy, které úspěšně projdou výrobním procesem, aniž by bylo nutné jakékoli přepracování. Zbývající čas po odečtení ztráty kvality se nazývá plně produktivní čas.

Nejjednodušším způsobem výpočtu OEE je poměr plného produktivního času k plánovanému výrobnímu času. Plně produktivní čas je způsob jak říci, že vyrobené díly jsou vyráběny v ideálním cyklu, tedy bez zastavení. Výpočet je následující:

OEE = (Počet dobrých kusů * Ideální cyklus výroby) / Plánovaná doba výroby

I když se jedná o zcela platný výpočet OEE, neposkytuje informace o třech ztrátových faktorech jako je dostupnost, kvalita a výkon [4].

4 Návrh řešení

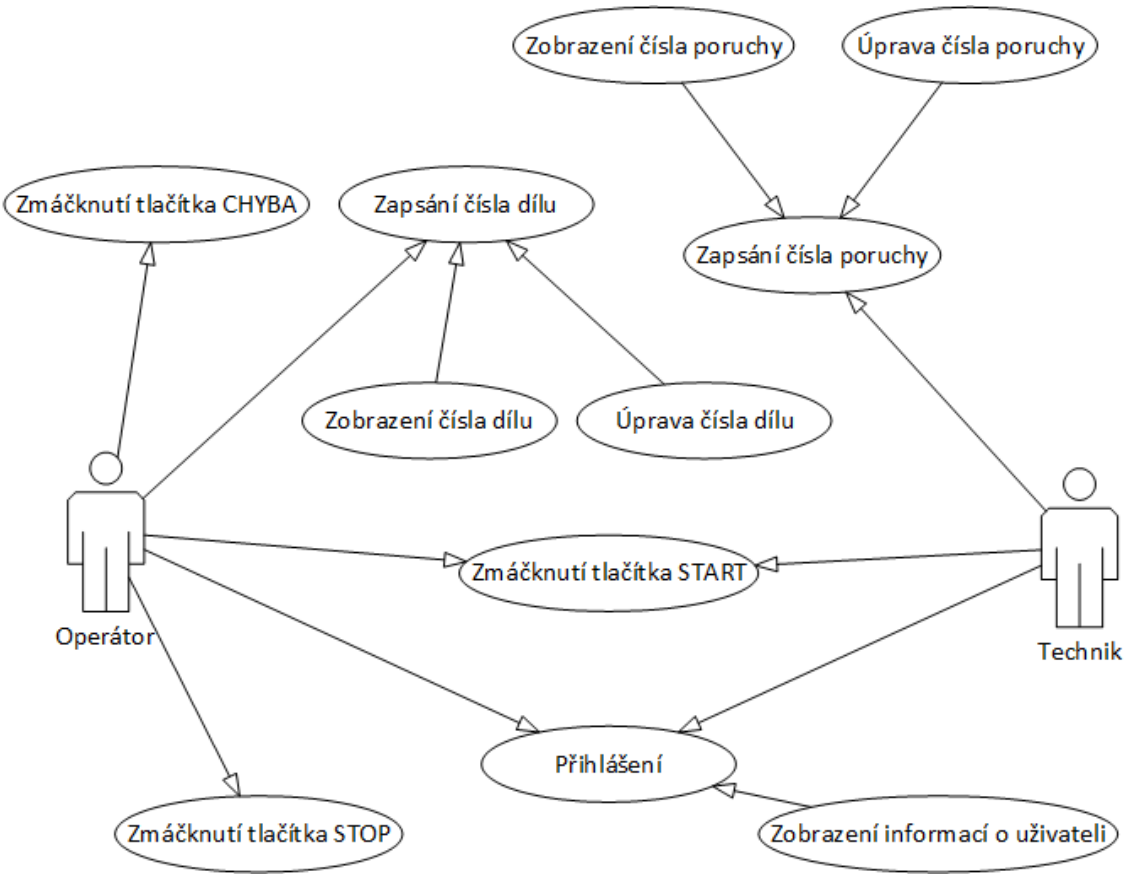
Při návrhu hardwarového řešení byl kladen důraz na odolnost zařízení a 24-hodinové nasazení v prašném a horkém prostředí. Zařízení by mělo mít svůj display, pomocí kterého bude uživatel interagovat se systémem. Na zadávání údajů by mělo zařízení obsahovat klávesnici a stavová barevná tlačítka. Pro kompletní zařízení bude nutné navrhnout krabičku přímo na míru. Pro snadnou údržbu zařízení a výměnu jednotlivých komponent bude zapotřebí sestrojit plošný spoj, který se nasadí na Raspberry Pi. Zde kde se jednotlivé komponenty budou zapojovat pomocí konektorů, což zjednoduší případnou výměnu nefunkčních komponent bez většího výpadku stroje. Laserová brána musí být minimalistická a připevněná přímo ke koncové části výrobní linky. Napájení zařízení bude pomocí klasického napájecího adaptéru, který bude zapojen do 230V elektrické sítě.

Softwarové prostředí by mělo být co nejjednodušší a intuitivní. Ověřování uživatelů bude provedeno pomocí RFID čtečky, kde každý uživatel dostane svůj identifikační čip. Data se budou načítat a ukládat do databáze. Zpracovaná data se budou zobrazovat na monitoru přímo u výrobní linky. Pro základní chod aplikace byl navrhnut následující scénář:

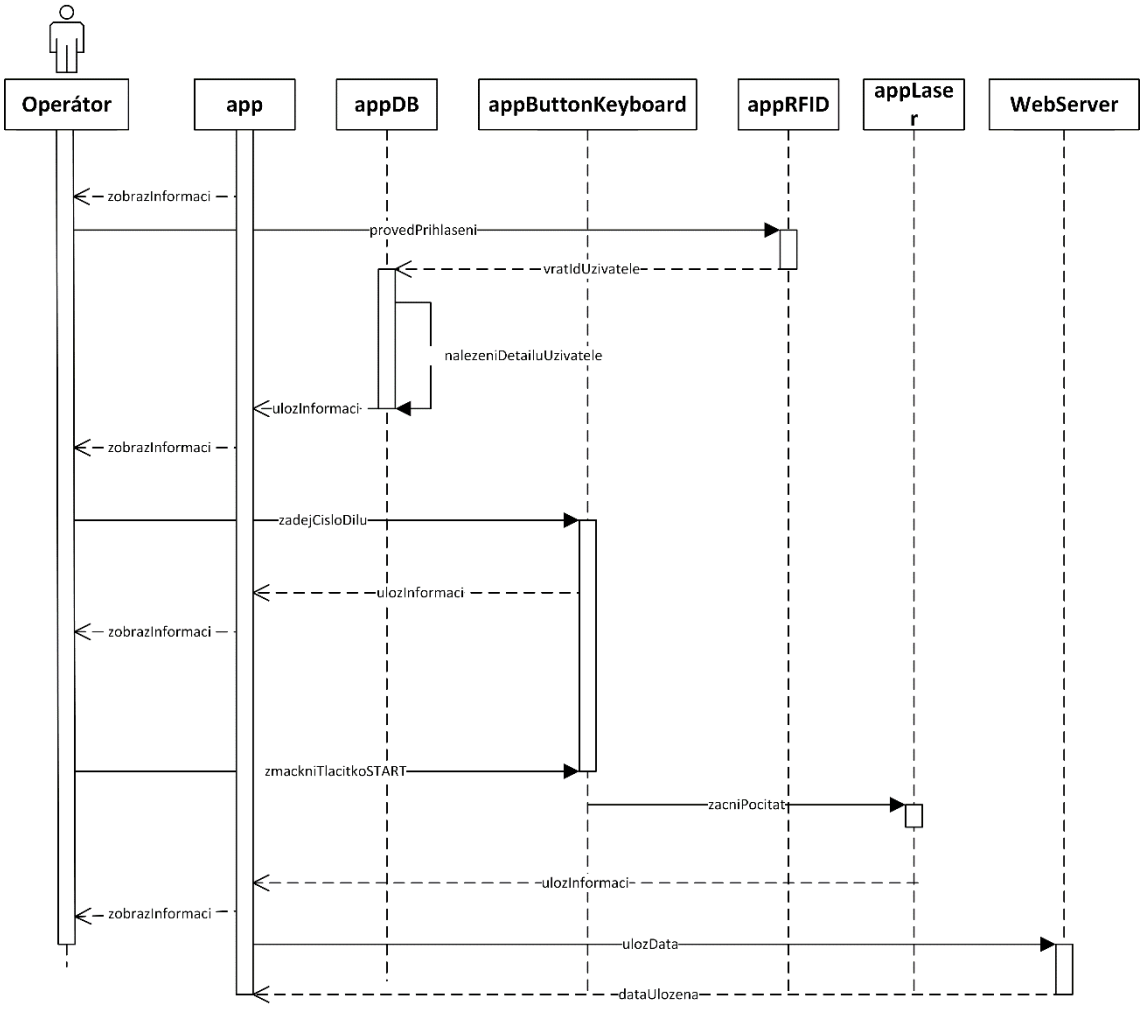
1. Systém zobrazí hlášku k přihlášení operátora
2. Operátor inicializuje přihlášení pomocí tagu
3. Systém zobrazí informace o operátorovi
4. Systém zobrazí hlášku k zadání čísla dílu
5. Operátor zadá číslo dílu
6. Operátor potvrdí číslo dílu
7. Systém zobrazí hlášku k zapnutí aplikace tlačítkem START
8. Operátor stiskne tlačítko START
9. Systém inicializuje počítání kusů
10. Systém zobrazí počet vyrobených dílu
11. Systém zobrazí status programu
12. Stiskne-li operátor tlačítko STOP
    - 12.1. Systém zobrazí status programu
    - 12.2. Systém pozastaví program
13. Stiskne-li operátor tlačítko START
    - 13.1. Systém zobrazí status programu
    - 13.2. Systém inicializuje počítání kusů
    - 13.3. Systém zobrazí počet vyrobených kusů
    - 13.4. Systém čeká na zmáčknutí tlačítka START
14. Stiskne-li operátor tlačítko CHYBA

14.1. Systém zobrazí status programu
14.2. Systém zobrazí hlášku k přihlášení technika
14.3. Technik inicializuje přihlášení pomocí tagu
14.4. Systém zobrazí informace o technikovi
14.5. Systém zobrazí hlášku k zadání čísla chyby
14.6. Technik zadá číslo chyby
14.7. Technik potvrdí číslo chyby
14.8. Systém čeká na zmáčknutí tlačítko START

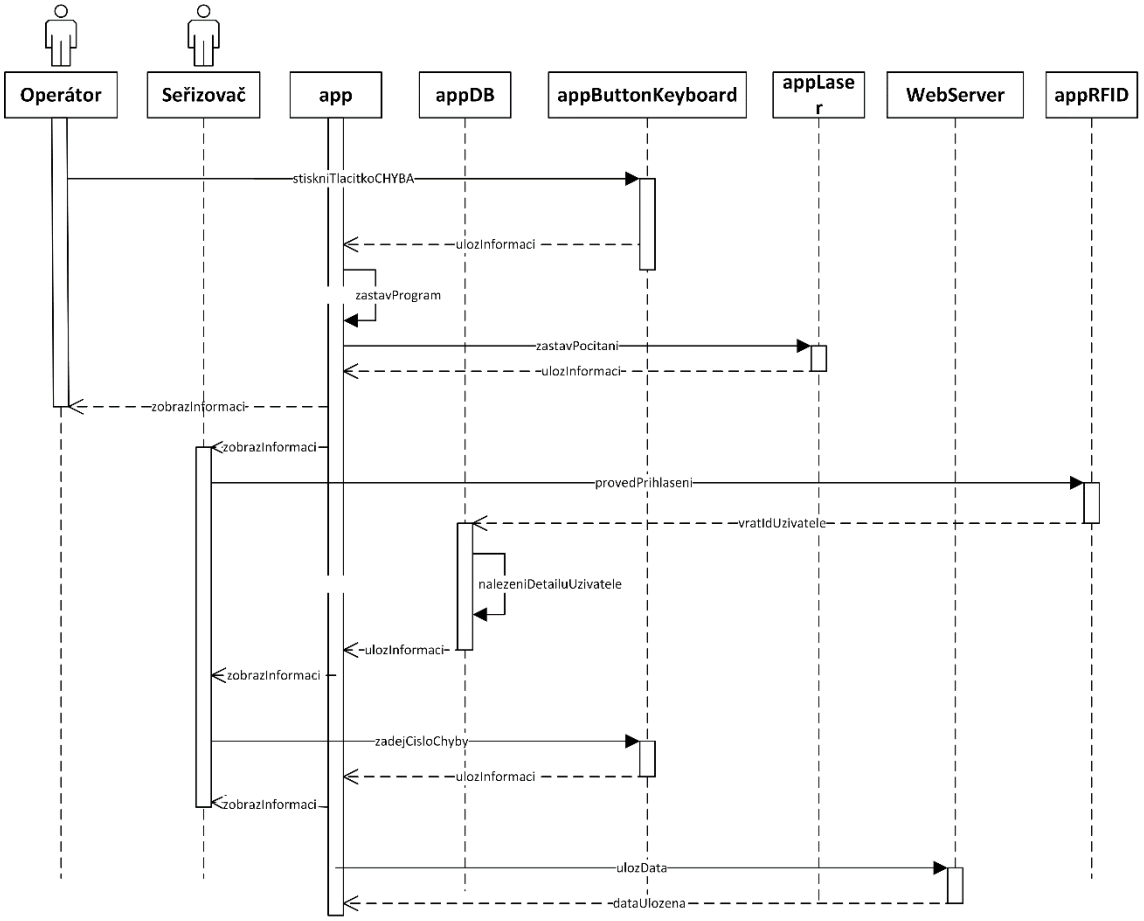
Na obrázku číslo 1 je zobrazen diagram aktérů, aplikace obsahuje dvě role a to operátora zařízení, tedy obsluhu stroje a technika, který bude opravovat vady stroje. Oba aktéři budou interagovat se zařízením pomocí tlačítek, klávesnice a přihlašovat se budou pomocí RFID čipů. Postup v případě chyby je zobrazen na obrázku číslo 3. Zastavení programu je znázorněno na obrázku číslo 4. Obrázek číslo 2 představuje akce při stisku tlačítka START.



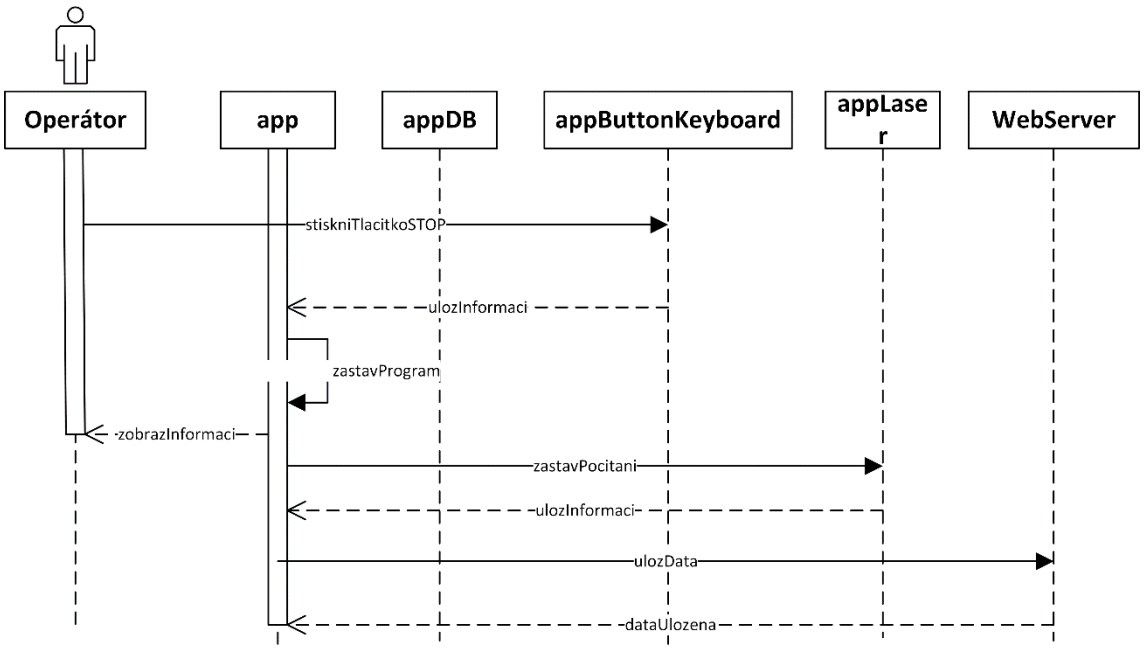
Obrázek 1 – Diagram aktérů [autor]



Obrázek 2 – Diagram startu programu [autor]



Obrázek 3 – Diagram chyby programu [autor]



Obrázek 4 – Diagram zastavení programu [autor]

5 Hardware

Tato část se věnuje hardwarové části práce jako je jednodeskový počítač Raspberry Pi, který je hlavní komponentou práce, OLED displej, sloužící k zobrazování informací uživateli, klávesnice a tlačítka, pomocí kterých je zajištěna interakce uživatele se systémem. V neposlední řadě je použita RFID čtečka pro přihlašování uživatelů, laserová brána pro počítání vyrobených kusů a vlastní krabička zařízení.

5.1 Raspberry Pi

Jako mozek zařízení je použit jednodeskový mikropočítač Raspberry Pi Model 3B+. Jedná se o malý počítač velikosti kreditní karty, který je založený na platformě ARM. Od roku 2012, kdy byl představen první model, byly vydány již 4 řady tohoto velmi oblíbeného počítače. Oblíbený je hlavně pro svoji cenu, univerzálnost a nízkou spotřebu elektrické energie. Za přibližně 1000Kč dostanete zařízení, které je schopno konkurovat běžnému kancelářskému počítači. Oblíbenost tohoto zařízení je ale především u různých kutilů nebo hobby komunity a své využití nalezne i v průmyslu, kde dokáže běžet prakticky 24 hodin, 7dní v týdnu.

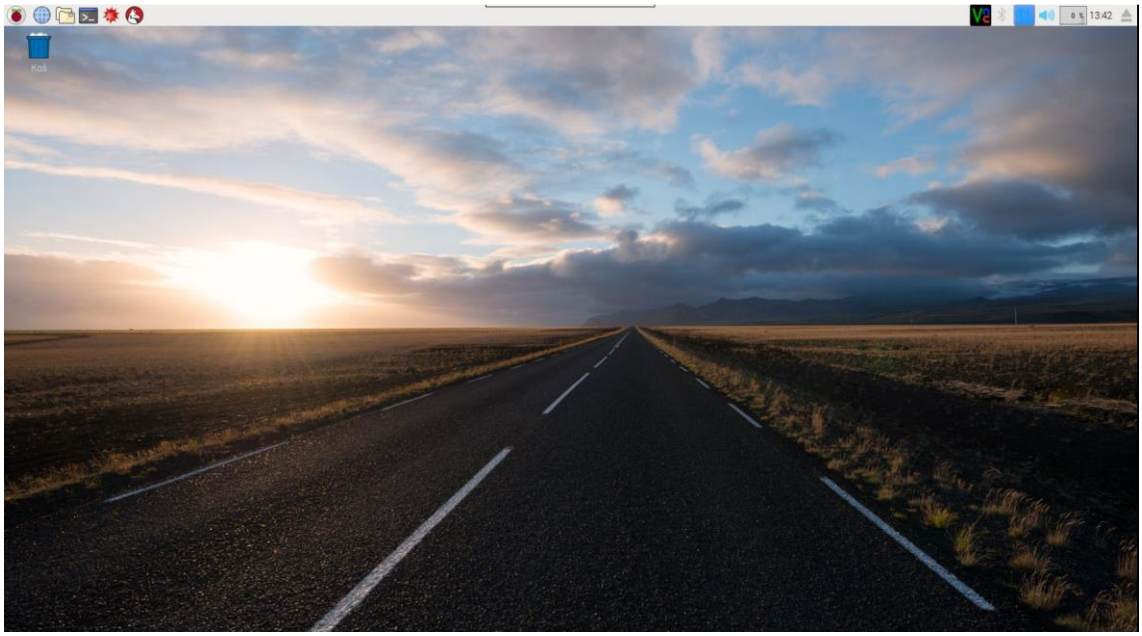
Jádrem systému je procesor typu SoC Broadcom BCM2837B0, který běží na taktu 1.4GHz. Většina systémových komponent včetně grafického a zvukového procesoru je integrována do jedné součástky. Počítači stačí pro bezproblémový chod napětí 5 V a proud 2,5 A. O napájení se stará integrovaný mikro USB port. U modelu 3B+ je také možnost napájení přes PoE. Díky nízké spotřebě energie není potřeba žádný aktivní chladič, procesor produkuje velmi málo odpadního tepla, i během náročnějších výpočetních operací. O rychlý chod aplikací se stará 1GB LPDDR2 SDRAM paměti. Nově u modelu 3B+ lze využít k připojení k internetu mimo Gigabit Ethernet portu také Wifi a to buď 2,4GHz nebo 5GHz IEEE 802,11.b/g/n/ac nebo Bluetooth 4.2. Připojení senzorů a zařízení lze pomocí 40-pinového GPIO headeru. Externí display lze připojit buď pomocí HDMI portu, nebo pomoci DSI display portu, který je přímo na desce zařízení. Pro ukládání dat a zavedení systému slouží slot na micro SD kartu. Pro připojení zvukového výstupního zařízení nechybí ani port na 3,5mm jack [2]. Použitá verze Raspberry Pi je zobrazena na obrázku číslo 5.

Obrázek 5- Raspberry Pi 3 model B+ [6]

5.1.1 Raspbian

Raspbian je bezplatný operační systém založený na Debianu, optimalizovaný pro Raspberry Pi. Raspbian nabízí více než jen čistý operační systém, přichází s více než 35 000 balíčků a knihoven optimalizovaných pro maximální výkon Raspberry Pi. Byl vydaný v roce 2012, nicméně vycházení stále nové verze a systém je aktualizován [7].

Pro projekt byl použit operační systém Raspbian Lite, který se liší od své klasické verze tím, že běží bez uživatelského prostředí, tedy vše je ovládáno přes příkazový řádek. Desktopové prostředí Raspbianu je vyobrazeno na obrázku číslo 6.



Obrázek 6- Prostředí operačního systému Raspbian [autor]

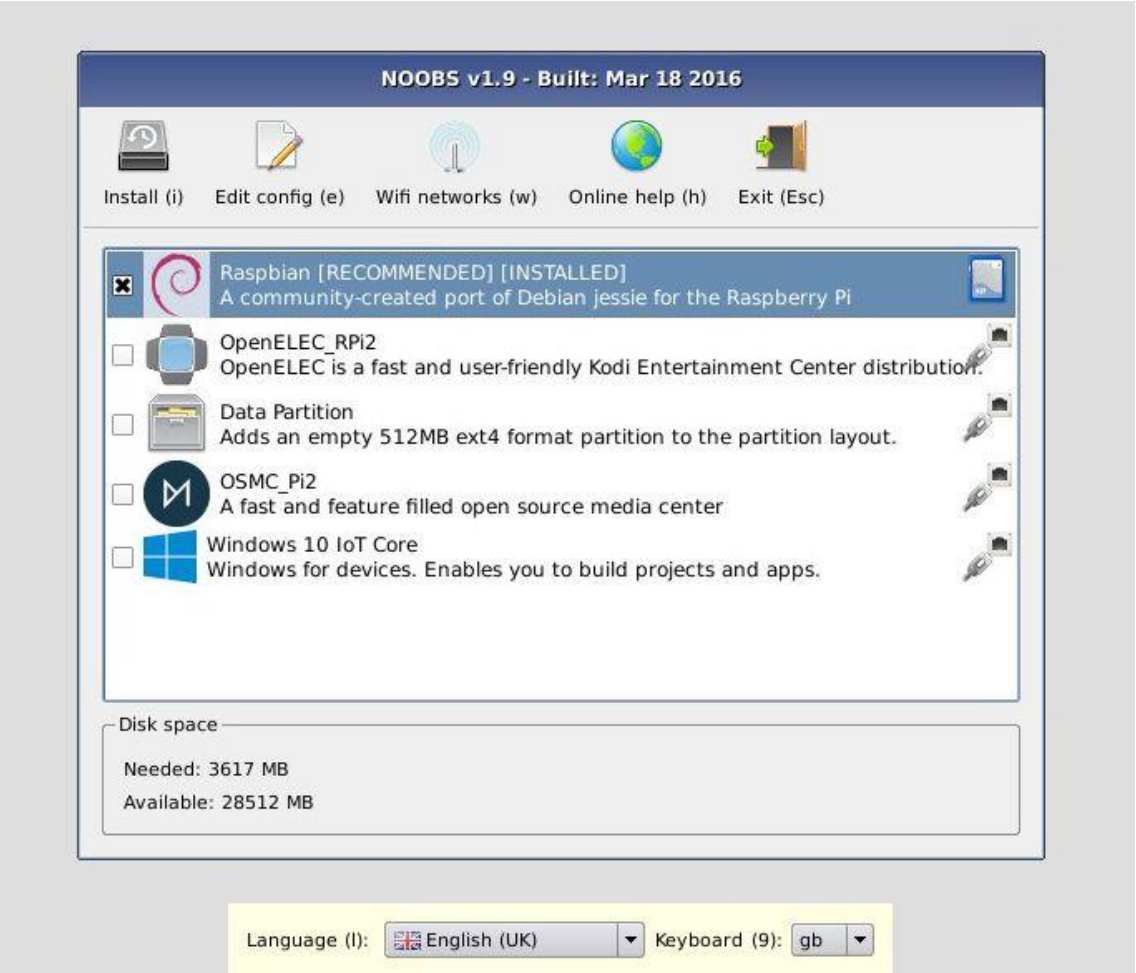
5.1.2 Instalace operačního systému

Počítač Raspberry Pi nemá klasický pevný disk, ale operační systém je uložen na SD kartě. Pro správný chod systému je potřeba SD karta s minimální velikostí 8GB, doporučuje se ale alespoň 16GB. Důležitá je také rychlost karty, je doporučena alespoň karta třídy Class 10.

Na Raspberry Pi jde nainstalovat hned několik operačních systémů. Nejpoužívanějším operačním systémem je Raspbian, která je odladěný přímo pro Raspberry Pi a má již předinstalované knihovny a aplikace. Dále je možné nainstalovat prakticky každou verzi Linuxu, která podporuje ARM architekturu. Mimo klasických operačních systémů lze na Raspberry Pi nainstalovat i populární multimediální centrum Kodi nebo emulátor retro her RetroPie.

Nejjednodušším způsobem nahrání operačního systému na SD kartu je tzv. NOOBS instalace, stačí pouze stáhnout instalační složku o velikosti cca 100MB a rozbalit na naformátovanou SD kartu. Poté jen stačí vložit SD kartu do Raspberry Pi a vybrat si ze seznamu nabídnutých operačních systémů. Po vybrání operačního

systému se vybraný systém stáhne a nainstaluje na Raspberry Pi. Instalátor NOOBS zobrazen na obrázku číslo 7. Druhý způsob je stáhnout přímo obraz operačního systému a pomocí programu jako je například Etcher vytvořit z SD karty bootovací médium [8].



Obrázek 7 – Instalátor NOOBS [9]

5.1.3 SSH

Pro připojení do vzdáleného zařízení byl použit protokol SSH, který usnadňuje práci přes konzoly a umožňuje se k druhému zařízení připojit prakticky odkudkoliv.

SSH je protokol, který umožňuje vzdálené připojení k jinému zařízení. Ovládání obstarává textová konzole. Nutné je připojení ke stejné síti, využívá TCP/IP. SSH vznikl jako náhrada telnetu, který neumožňoval šifrovanou komunikaci a posílal data jako prostý text. SSH oproti telnetu umí data šifrovat a není možno jej odposlouchávat [10].

5.1.4 BerryBoot

BerryBoot je jednoduché zaváděcí menu pro ARM počítače jako je Raspberry Pi. Umožňuje to přidat více linuxových distribucí na jednu SD kartu a pomocí menu je vybírat při startu počítače. Zaváděcí měnu je vyobrazeno na obrázku číslo 8. Dále také umožňuje zálohu a obnovu operačního systému.

V projektu byl BerryBoot využit k tomu, že SD karta je využita pouze k nahrání zavaděče systému. Operační systém není nahraný na SD kartě, ale je v síťovém uložišti iSCSI na Microsoft Windows serveru a přes zavaděč je k němu přistupováno. Výsledkem je, že všechna zařízení využívají jednu sdílenou kopii operačního systému a nedochází k opotřebení SD karty [11].

Obrázek 8: BerryBoot zavaděč [11]

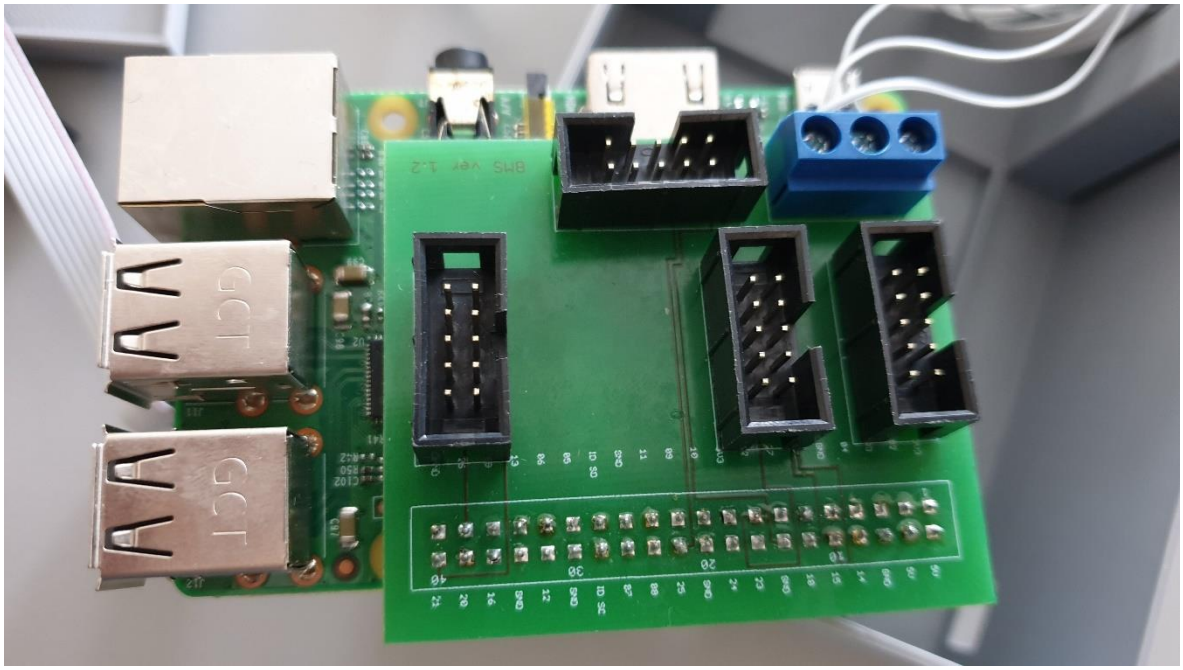
Rozhraní iSCSI je vychází ze dvou rozšířených technologií, a to je SCSI rozhraní, které zajišťuje připojování disků v serverech a technologie TCP/IP, který se stará o chod většiny počítačových sítí. Využití rozhraní je především pro sdílené uložiště, kde více serverů vidí ten samý logický svazek. To se využívá především pro clustering nebo VMWARE [12].

5.2 Plošný spoj

Plošný spoj neboli PCB je tenká deska ze skleněného vlákna, kompozitního epoxidu nebo jiného laminátového materiálu. Vodivé dráhy jsou vyleptány nebo vytištěny na desku, spojující různé komponenty na desce plošných spojů, jako jsou tranzistory,

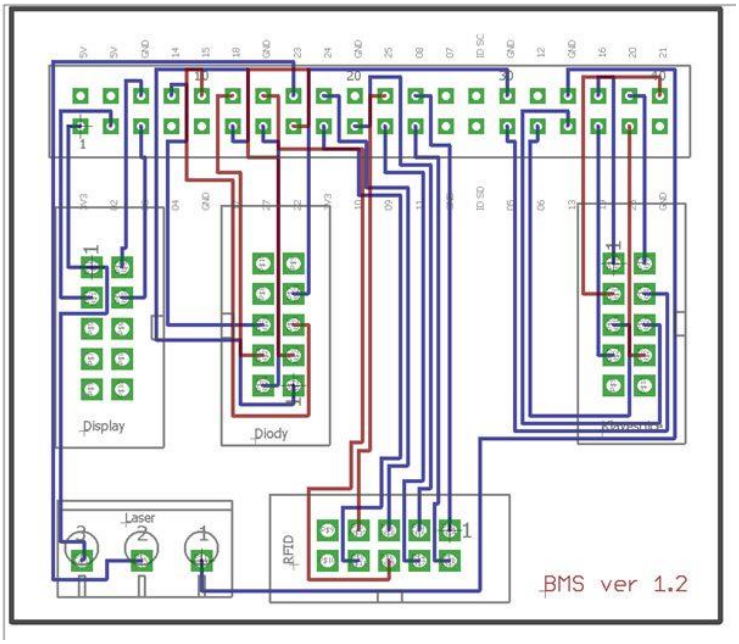
odpory a integrované obvody. PCB se používají jak v stolních, tak v přenosných počítačích. Slouží jako základ mnoha interních počítačových komponent, jako jsou grafické karty, karty řadičů, karty síťového rozhraní a rozšiřující karty. Všechny tyto komponenty se připojují na základní desku, což je také deska s plošnými spoji [13].

Pro připojení všech senzorů k Raspberry Pi byl navrhnut vlastní oboustranný plošný spoj.

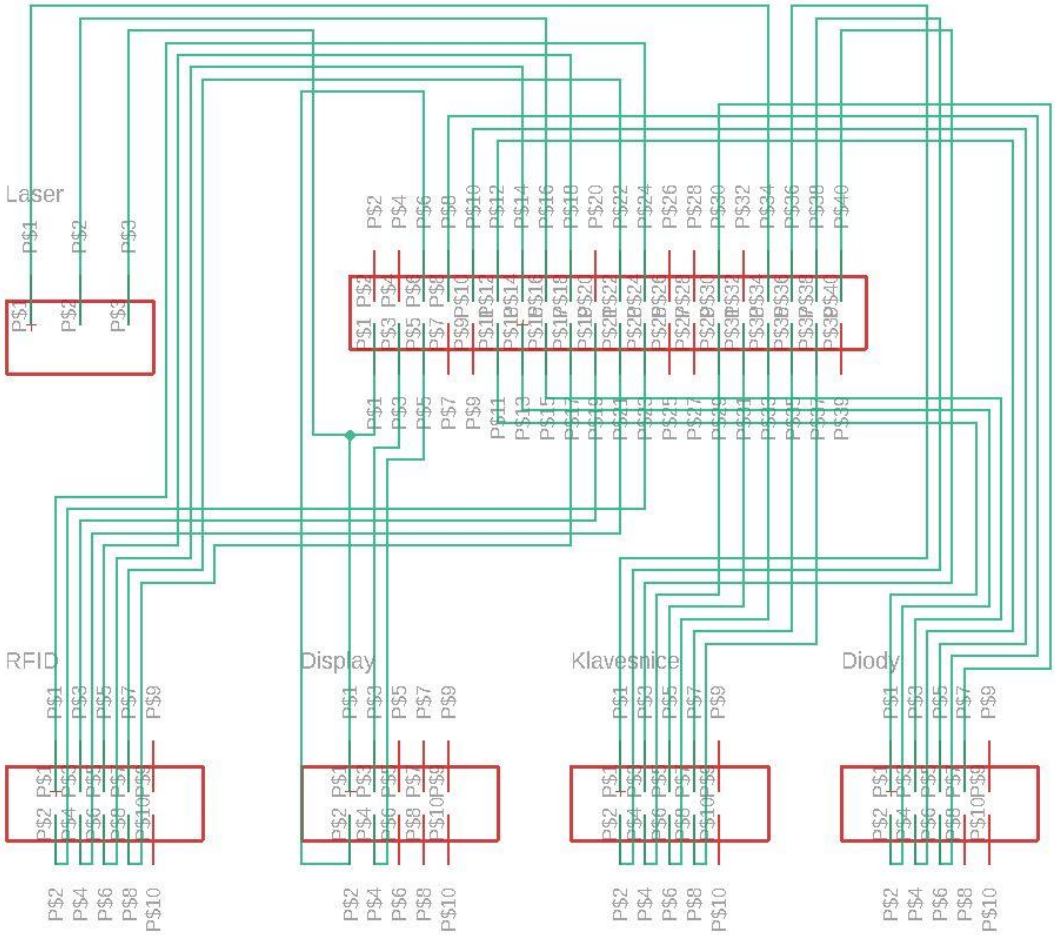


Obrázek 9 - Osazený plošný spoj [autor]

Na jedné straně plošného spoje je GPIO header konektor, který se připojí přímo k Raspberry Pi jako tzv. hat. Na druhé straně plošného spoje se nacházejí konektory na připojení klávesnice, laserového senzoru, displeje, RFID čtečky, tlačítek a diod. Samotné připojení je pak provedeno pomocí samčího konektoru na desce a samičího konektoru na kabelu vedoucího z vstupního nebo výstupního zařízení. Pokud dojde k poškození některého zařízení, lze ho bez problému nahradit bez potřeby pájení. Návrh plošného spoje byl vlastní silou vytvořen v programu Eagle, jehož návrh je vyobrazen na obrázku číslo 10 a obrázku číslo 11. Samotný plošný spoj byl na zakázku vyroben externí firmou a osazení a pájení plošného spoje byl uskutečněn také svépomocí. Výsledný osazený spoj je zobrazen na obrázku číslo 9.



Obrázek 10 - Vlastní návrh plošného spoje [autor]



Obrázek 11 – Vlastní návrh plošného spoje 2 [autor]

5.3 OLED display

Pro účely interakce s operátorem byl v projektu použit 2.42 palcový OLED displej 128O064D s rozhraním I2C a rozlišením 128 x 64 pixelů [14]. Použitý OLED displej je zobrazen na obrázku číslo 12.

OLED je technologie vyzařující ploché světlo umisťováním řad organických tenkých vrstev mezi dva vodiče. Za pomocí elektrického proudu vyzařuje jasné světlo. OLED jsou emisní displeje, které nevyžadují podsvícení, takže jsou tenčí a účinnější než LCD displeje, které vyžadují bílé podsvícení. OLED displeje nejsou jen tenké a efektivní ale poskytují nejlepší kvalitu obrazu a mohou být vyrobeny transparentní, flexibilní, skládací a dokonce i rolovací a roztažitelné [14].

Obrázek 12 - OLED I2C displej [15]

I2C je interní datová sběrnice sloužící pro komunikaci a přenos mezi integrovanými obvody v rámci jednoho zařízení. V současné době tento způsob komunikace využívají především mikro-kontroléry, sériové paměti, inteligentní LCD, audio a video obvody. Obousměrný provoz probíhá pouze po dvou vodičích, což výrazně zjednodušuje výsledné zapojení. Na jednu sběrnici tak může být napojeno více obvodů a to až 1024 čipů s různou adresou na jednu sběrnici [16].

Na displeji budou zobrazovány údaje, jako pokyny k přihlášení, zobrazení přihlášeného uživatele, stav zařízení, počet vyrobených kusů. Na zařízení bude take probíhat interakce s uživatelem, kam bude například zadávat číslo stroje nebo číslo poruchy.

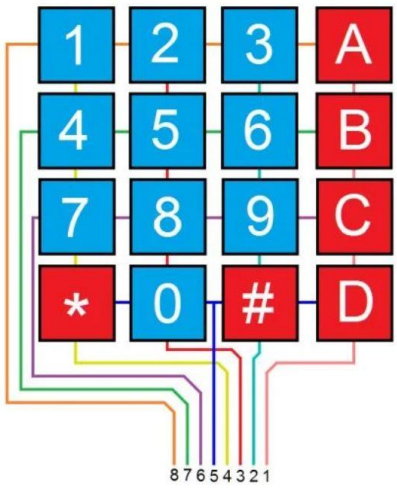
Pro tento display je použita knihovna Adafruit, konkrétně knihovny Adafruit GPIO SPI a Adafruit SSD1306 upraveny přímo pro daný displej. Knihovny jsou dostupné na veřejném uložišti Github, je tedy velice jednoduché je importovat.
O import knihovny se stará několik řádků kódu. Nejdříve je nutné stáhnout repozitář s knihovnou a poté je ve složce se souborem provedena instalace příkazem *install*. Po importu knihovny je možné plně využívat její funkcí. Na obrázku číslo 13 je nastíněn postup instalace knihovny displeje.

```
sudo apt-get install git
git clone https://github.com/adafruit/Adafruit_Python_SSD1306.git
cd Adafruit_Python_SSD1306
sudo python setup.py install
```

Obrázek 13- Ukázka kódu displeje [autor]

5.4 Klávesnice a tlačítka

V programu bude zapotřebí interakce uživatele, která bude řešena maticovou membránovou lepící klávesnicí. Ta obsahuje jak písmena, tak i čísla a speciální znaky. Klávesnice je složena z šestnácti tlačítek, které se zapojují pomocí osmi konektorů přímo k GPIO rozhraní na Raspberry Pi. Schéma zapojení klávesnice je zobrazeno na obrázku číslo 14.



Obrázek 14 - Schéma zapojení klávesnice [autor]

Klávesnice je definovaná v kódu jako pole, kde jsou definovány sloupce a řádky klávesnice. Tento postup je viditelný na ukázce kódu číslo 1. Poté již jednoduchým cyklem hlídáme, které tlačítko bylo stlačeno.

```
MATRIX = [
             [1,2,3,'A'],
             [4,5,6,'B'],
             [7,8,9,'C'],
             ['*',0,'#','D']

          ]

COL= [6,13,19,26] #c1,c2,c3,c4
ROW = [16,20,21,5] #r1,r2,r3,r4

for j in range(4):
    GPIO.setup(COL[j], GPIO.OUT)
    GPIO.output(COL[j],1)

for i in range(4):
    GPIO.setup(ROW[i], GPIO.IN, pull_up_down = GPIO.PUD_UP)

try:
    while(True):
        for j in range(4):
            GPIO.output(COL[j],0)

            for i in range(4):
                if GPIO.input(ROW[i]) ==0:
                    print MATRIX[i][j]
                    while(GPIO.input(ROW[i]) == 0):
                        pass
            GPIO.output(COL[j],1)
except KeyboardInterrupt:
    GPIO.cleanup()
```

Ukázka kódu 1 - Definování klávesnice [autor]

Pro jednodušší ovládání klávesnice je také možné použít MM74C922 čip, který využívá logiku propojení matice šestnácti kláves. Čip enkodéru průběžně kontroluje klávesnici a čeká na stisk klávesy.

Číselné hodnoty na klávesnici jsou využívány k zadávání číselných hodnot, jako jsou čísla poruch nebo čísla vyráběných dílů. Symbol hvězdičky je využit k potvrzování příkazů. Pomocí písmena „D“ lze opravit zadanou číselnou kombinaci.

Mimo klávesnice jsou na zařízení umístěny i tři barevná tlačítka. Jedná se o jednoduchá spínací tlačítka, které slouží k tomu, aby rozpojili dva body obvodu. Ukázka použitého mikrospínače je zobrazena na obrázku číslo 15. Zapojení je jednoduché pomocí dvou kabelů a 10k Ω odporem, který je mezi signálem a uzemněním. V aplikaci pak jen hlídáme zmáčknutí portu Raspberry Pi, do kterého je tlačítko připojeno.

Obrázek 15 – Mikrospínač [15]

Na zařízení najdeme tři barevné tlačítka, každý pro jinou akci. Zelené tlačítko je použito ke startu programu buďto na začátku směny, nebo například po přestávce či odstranění poruchy. Oranžové tlačítko slouží k pozastavení programu při odchodu operátora na oběd či svačinu. Dále potom červené tlačítko slouží k uvedení programu do stavu poruchy, kdy operátor zastaví výrobu a čeká na seřizovače, který odstraní poruchu. V ukázce kódu číslo 2 je zobrazeno jednoduchá akce při zmáčknutí tlačítka.

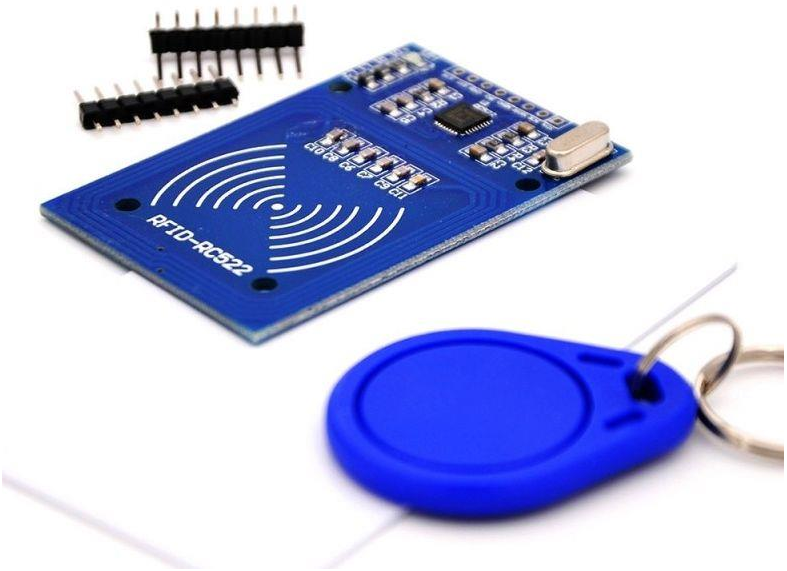
```
1. import RPi.GPIO as GPIO
2. GPIO.setmode(GPIO.BCM)
3. GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
4.
5. while True:
6.     input_state = GPIO.input(18)    #when press button on GPIO
    18
```

Ukázka kódu 2 – Akce při zmáčknutí tlačítka [autor]

5.5 RFID čtečka

RFID funguje tak, že vysílá a přijímá informace přes anténu a mikročip, který je také nazývám integrovaný obvod. Mikročip na čtečce RFID uložen s jakoukoliv informací, co uživatel chce. Existují dva hlavní typy RFID tagů a to bateriově napájené a pasivní. Bateriově napájené RFID tagy obsahují jako zdroj napájení baterii, zatímco pasivní RFID tag pracuje pomocí elektromagnetické energie přenášené z RFID čtečky. Bateriově napájené RFID tagy mohou být také označovány jako aktivní [17].

Pro logování operátorů a seřizovačů je použita RFID čtečka. Přesněji zařízení RC522 [18], které patří mezi nejpoužívanější RFID čtečky a má již zpracovanou knihovnu. Čtečka pracuje na frekvenci 13,56 MHz. Informace pro čtečku jsou zaznamenány na FID zařízení, které se také nazývá tag. Tagy mohou mít různé podoby jako například karty, nálepky, náramky či kulaté přívěsky a také se dělí podle používaných frekvencí. Čtecí zařízení se dá umístit pod krabičku a čtení je možné provádět elegantně přiložením čipu ke krabičce zařízení. Každý čip (tag) má své jedinečné číslo. Na ukládání čísel je používána MySQL databáze, kde je ke každému číslu zadáno jméno, příjmení a pozice. Po přiložení čipu ke čtečce se provede čtení ID a podle něj se získají další informace o uživateli. Použitá čtečka a tagy jsou zobrazeny na obrázku číslo 16.



Obrázek 16 - RFID čtečka a tagy [18]

5.6 Laserová brána

Pro potřebu počítání počtu vyrobených kusů je zapotřebí použít pohybový senzor. Senzorů je velká řada od optických po ultrazvukové. Pro přesné počítání byl vybrán laserový senzor Keyes KY-008 Laser Modul 650NM. Použitý laserový přijímač a vysílač je zobrazen na obrázku číslo 17. Ten se skládá z laserového vysílače a přijímače. Vysílač vydává laserový paprsek a přijímač pomocí diody zkoumá, jestli na něj dopadá laserový paprsek či nikoliv. Z přijímače tedy vycházejí hodnoty buď 0, nebo 1. Laserová brána je ovládaná za pomocí vlákna a struktura této metody je zobrazena v ukázce kódu číslo 3.

```
1. import RPi.GPIO as GPIO
2. import sys
3. import time
4. from time import sleep
5. GPIO.setmode(GPIO.BCM)
6. GPIO.setup(23, GPIO.IN)
7.
8. try:
9.     while True:
10.         if GPIO.input(23)==0:
11.             print "0"
12.             sleep(1)
13.         else:
14.             print "1"
15.             sleep(1)
16. finally:
17.     GPIO.cleanup()
18. GPIO.cleanup()
```

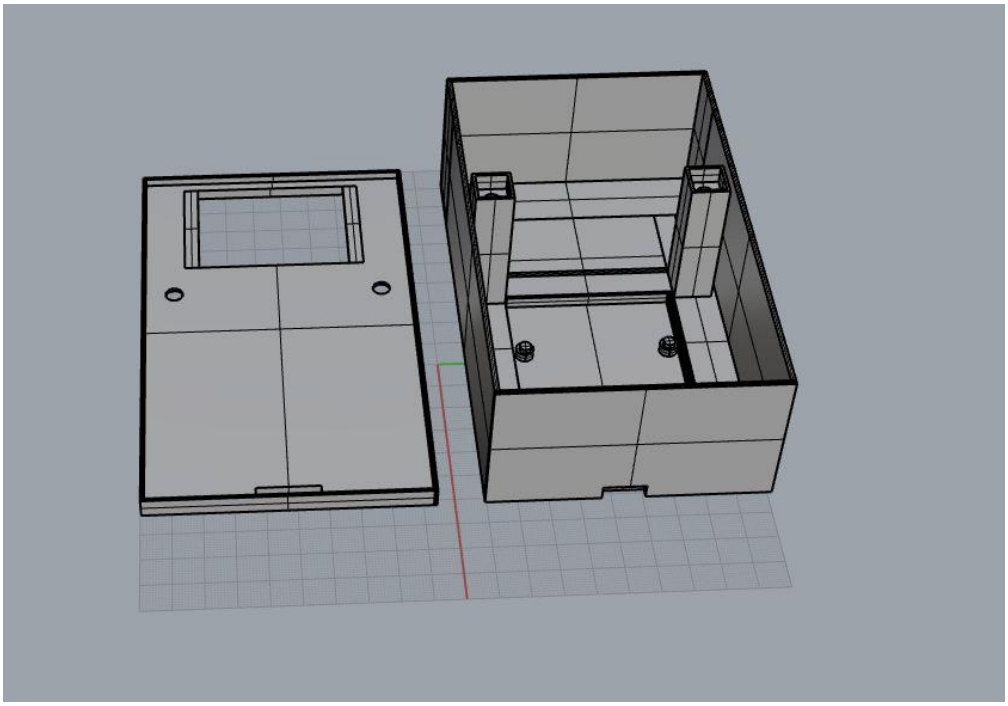
Ukázka kódu 3 – Vlákno třídy appLaser.py [autor]

Obrázek 17 - Laserový přijímač a vysílač [18]

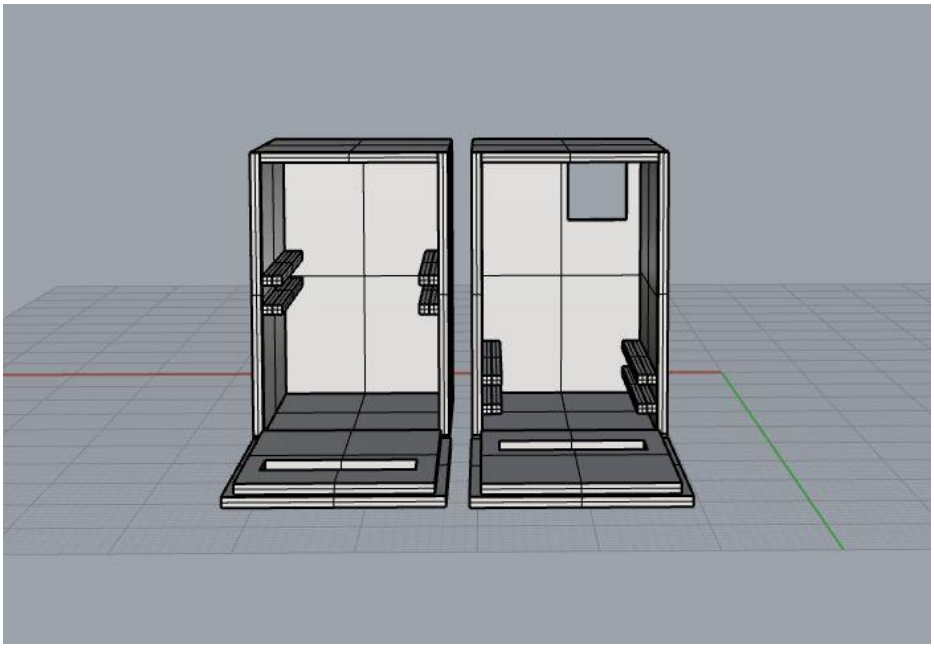
5.7 Krabička zařízení

Jelikož je hardware unikátní, bylo zapotřebí navrhnout i krabičku na celé zařízení. Ta byla vymodelována v grafickém programu Rhino 3D. Jedná se o celistvou krabičku pro Raspberry Pi, HAT s konektory, displejem. Na povrchu krabičky jsou potom umístěny tlačítka, diody a klávesnice. Víko krabičky je připevněno šroubem se závitem. Pro laserovou bránu byly navrhnuty dvě krabičky, jedna pro přijímač a druhá pro vysílač. Krabičky budou připevněny k okraji výrobní linky.

Navrhnuté krabičky byly vytisknuty na 3D tiskárně Raise3D pomocí PLA materiálu. Návrh krabiček je zobrazen na obrázku 18 a 19. Výsledná podoba zařízení je znázorněna na obrázku číslo 20.



Obrázek 19 - Návrh krabičky zařízení [autor]



Obrázek 18 - Návrh krabičky laseru [autor]

Obrázek 20: Finální provedení zařízení [autor]

6 Software

Pro vývoj aplikace byl použit programovací jazyk Python, především z důvodu jeho jednoduchosti a velké databáze již zpracovaných knihoven pro všechny různé senzory a zařízení. Python je také přeinstalován přímo v Raspbianu a to hned ve dvou verzích. Starší verze 2 neboli Legacy a verze 3, která je nejnovějším vydáním jazyka. Pro projekt byla použita verze pythonu 3.7.0.

Python je objektově orientovaný, interpretovaný a interaktivní programovací jazyk. Kombinuje sílu s velmi jasnou syntaxí, která je lehce naučitelná a klade důraz na čitelnost a snižuje náklady na údržbu programu. Obsahuje moduly, třídy, výjimky a dynamické typy. Jsou zde rozhraní a mnoho systémových volání a knihoven, stejně jako různé systémy oken. Nové vestavěné moduly lze snadno psát v jazyce C nebo C++ (nebo v jiných, v závislosti na zvolené implementaci). Python je také použitelný jako rozšiřující jazyk pro aplikace psané v jiných jazycích, které vyžadují snadno použitelné skriptovací nebo automizační rozhraní. Python podporuje různé moduly a balíčky, díky kterým je zajištěna modularita a znovu-použitelnost kódu. Python se dělí na dvě hlavní verze. Python 2, který je původní, a od kterého jsou odvozeny další verze a Python 3, který je používán v současnosti a tudíž je v aktivním vývoji [19].

Pro webovou aplikaci, která zpracovává data, byl použit velmi rozšířený programovací jazyk PHP.

PHP je široce používaný skriptovací jazyk s otevřeným zdrojovým kódem, který je zvláště vhodný pro vývoj webových aplikací a může být vložen do HTML. PHP stránky obsahují HTML s vloženým kódem. Kód PHP je uzavřen ve speciálních znacích pro začátek a konec, který dovoluje přecházet mezi PHP kódem a HTML. Základní syntaxe jazyka je zobrazena v ukázce kódu číslo 4. PHP se odlišuje od ostatních skriptovacích jazyků tím, že kód je prováděn na serveru a následně je odeslán klientovi. Klient obdrží výsledky spuštěného skriptu, aniž by znal zdrojový kód. Hlavní výhodou PHP je to, že je pro nováčka velmi pochopitelný, ale nabízí i mnoho pokročilých funkcí pro zkušeného programátora [20].

```
1.  <!DOCTYPE html>
2. <html>
3. <body>
4.
5. <h1>My first PHP page</h1>
6.
7. <?php
8. echo "Hello World!";
9. ?>
10.
11. </body>
12. </html>
```

Ukázka kódu 4 – Základní syntaxe jazyka PHP [autor]

Získaná a zpracovaná data budou ukládána na databázový server MySQL.

MySQL patří mezi neojoblíbenější open source databáze na světě. Díky osvědčené výkonnosti, spolehlivosti a snadnému používání je MySQL hlavní volbou databáze pro webové aplikace, využívané webovými stránkami jako jsou například Facebook, Twitter, Youtube, Yahoo! a mnoho dalších. MySQL patří pod gigant Oracle, který řídí inovaci, přináší nové funkce pro výkon příští generace webů, cloudů, mobilních zařízení a embedded aplikací [21].

6.1 Aplikace

Hlavní aplikace bude hostována přímo na Raspberry Pi. Aplikace je psána objektově v programovacím jazyku Python. Aplikace je složena z mnoha tříd a jedné spouštěcí třídy.

Spouštěcí třída *app.py* pracuje na principu vlákna, kde se jednotlivé instance třídy spustí a potom vlákno běží na pozadí a čekají na akci nebo zastavení běhu vlákna. V této třídě je importováno mnoho knihoven, které jsou nutné k chodu aplikace. Importovaná knihovna GPIO zajišťuje přístupu a nastavení GPIO konektorů na Raspberry Pi. Další využívanou knihovnou je systémový běh vlákna, který spolu s časovačem zajišťuje běh vlákna na pozadí aplikace. Mezi importovanými knihovny třetích stran byla použita Adafruit databáze knihoven, zajišťující chod především displeje. V neposlední řadě jsou zde importovány

knihovny PIL zajišťující styl písma, obrázků a vykreslení. Nechybí zde ani import vlastních tříd.

Na začátku aplikace jsou definovány proměnné. Mezi dvě první metody patří *startThreading* a *stopThreading*, které obstarávají správný chod vlákna. Další metoda *control* kontroluje zmáčknutí tlačítek a spouštění nad nimi běh vlákna. Nejrozsáhlejší třída *run* má za úkol nastartovat všechny komponenty zařízení, jako je display, RFID čtečku, databázi, klávesnici a tlačítka. Nejprve je vytvořena instance třídy displej, potom je displej spuštěn příkazem *run*, kde je vypsána hláška uživateli k přiložení čipu. Poté je inicializován čip příkazem *run*. Následně je zavolána instance třídy *appDb.py*, která nejprve uloží id čipu a v databázi najde informaci o majiteli čipu a vypíše je na displej. Poté již startuje hlavní část programu, kdy je operátor vyzván k zadání čísla dílu pomocí klávesnice. Následně program pomocí vlákna kontroluje, jestli není stlačeno tlačítko, které vyvolá další akce.

Třída *appButton.py* definuje vlastnosti a fungování tlačítek. V úvodu třídy jsou deklarovány proměnné a definovány diody pomocí GPIO funkce, kde jednoduše deklarujeme, která dioda je zapojena na jakém pinu v Raspberry Pi. Definování diod v programu je vyobrazeno v ukázce kódu číslo 5.

```
1. GPIO.setup(diod_list, GPIO.OUT) #set list of diod
2. GPIO.setup(14, GPIO.OUT) #Green diod
3. GPIO.setup(15, GPIO.OUT) #Yellow diod
4. GPIO.setup(18, GPIO.OUT) #Red diod
```

Ukázka kódu 5 – Definování diod [autor]

Následně jsou použity tři hlavní metody a to metoda *start* definující akci zeleného tlačítka, metoda stop, které představuje oranžové tlačítko a metoda *pause* pro červené tlačítko. Metoda *start* čeká na stisknutí zeleného tlačítka a poté vypíše text „Start, čekám na první díl“ na obrazovku displeje. Dále rozsvítí zelenou diodu a zapne laser. Metoda *stop* hlídá, kdy je stisknuto oranžové tlačítko. Po zmáčknutí vypíše pomocí metody *run* třídy dislej text „Stop, zastaveno“, rozsvítí oranžovou diodu a vypne laser. Poslední ze tří metod je metoda *pause*, která čeká na sktisknutí červeného tlačítka. Poté vypíše na displej hlášku „porucha“, rozsvítí červenou diodu a zastaví laser. Následně vyzve mechanika, aby se přihlásil přiložením čipu, po

přiložení čipu je zavolána třída databáze, kde načtená hodnota čipu je zkontrolována a na displej vypsány informace o seřizovači. Následně aplikace vyzve seřizovače k zadání čísla poruchy. V ukázce kódu číslo 6 je znázorněna akce při zmáčknutí tlačítka START.

```
#START BUTTON
    def Start(self):
        #Check if the button is not already squeezed
        if (Button.GetStatus()!=1):
            print "START"
            global d1
            d1=Display()
            d1.run("Start","Cekam na prvni dil..","")
            global diod_list
            #clear diods
            GPIO.output(diod_list, GPIO.LOW)
            GPIO.output(14,GPIO.HIGH)
            global l1
            #define laser pin
            l1=Laser(23)
            #set status to 1
            Button.SetStatus(1)
            #start laser threading
            l1.startThreading()
            #save datetime.now to variable
            self.varTime=datetime.datetime.now()
            print self.varTime
```

Ukázka kódu 6 – Akce při zmáčknutí tlačítka START [autor]

Třída *appDb,py* obstarává komunikaci s MySQL databází. V třídě je nejprve definováno spojení s databází, kde je nastavena IP adresa serveru, přihlašovací údaje a jméno tabulky databáze. Pro tento účel jsou vytvořeny dvě tabulky v databázi, jedna pro seznam operátorů a druhá pro seznam seřizovačů. Mezi hlavní metody třídy databáze patří metoda *readValue*, která slouží ke čtení hodnot z databáze. V této metodě je nejdříve definován sql příkaz, kde je hledán záznam v databázi shodný s ID čipu přiložený operátorem nebo seřizovačem.

Pro řízení displeje je použita třída *appDisplay.py*, která obsahuje knihovny Adafruit_SSD1306 a SPI. V základní metodě *_init_* jsou základní nastavení displeje. Pro lepší čtení na displeji byl použit externí font Sherif.otf. Následující metoda *run* s parametry text1, text2 a text3 slouží k vypsání textu na displej, přičemž každý parametr je vypsán jinou velikostí písma a na samostatný řádek. Další třídy potom

již jen volají metodu *appDisplay.run(text1,text2,text3)*, kde za parametry dosadí text ve String formátu. Vykreslení textu a metoda *run* je zobrazena v ukázce kódu číslo 7.

```
#method to draw to display once
    def run(self,text,text2,text3):
        self.text=text
        # Draw a black filled box to clear the image.
        self.draw.rectangle((0,0,self.width,self.height), outline=0, fill=0)

        #Draw text on display before looting
        self.draw.text((self.x, self.top),text,  font=self.font, fill=255)
        self.top=self.top+16
        self.draw.text((self.x, self.top),text2,  font=self.font2, fill=255)

        self.top=self.top+18
        self.draw.text((self.x, self.top),text3,  font=self.font2, fill=255)

        self.top=self.top+8

        self.disp.image(self.image)
        self.disp.display()
        time.sleep(0.1)
```

Ukázka kódu 7 – Metoda run třídy appDisplay.py [autor]

Další důležitou třídou je *appKeyboard.py*, která definuje akce po stisknutí specifických kláves na klávesnici. V třídě jsou použity systémové knihovny pro běh vlákna, ovládání GPIO nebo nastavení času. V inicializační metodě třídy *init* jsou vytvořeny Array listy a matice klávesnice. V metodě *controll* jsou for cyklem procházeny listy a kontrolováno, které tlačítko na klávesnici bylo zmáčknuto. V jednotlivých podmínkách se potom kontroluje, jestli je zmáčknuto červené, zelené nebo oranžové tlačítko, a podle toho je určeno, který výstup z klávesnice je ukládán. Pokud je například zmáčknuté oranžové tlačítko „STOP“, tak číselný výstup z klávesnice je uložen jako číslo dílu. V další metodě je definována klávesa „#“, aby sloužila jako „Enter“, tedy potvrzení volby. Jsou zde dvě podmínky, nejprve musí být zmáčknut znak „#“ na klávesnici a poté je kontrolováno, jaké stavové tlačítko je zmáčknuto. Pokud je zmáčknuto tlačítko „STOP“, tak se jedná o potvrzení čísla dílu, a pokud je zmáčknuto tlačítko „CHYBA“, bude potvrzeno číslo poruchy. Další důležitou funkcí, která je řešena pomocí kláves, je mazání špatně zadaného čísla

poruchy nebo čísla dílu. Zde je použita podmínka, když je zmáčknuto písmeno „D“, tak je smazán poslední znak v řetězci znaků daného čísla. Akce pro zmáčknutí tlačítka „D“ je ukázána v ukázce kódu číslo 9.

```
#control if variable is Integer and STATUS = STOP
#save it into array TempItemNumber
if ((type(self.MATRIX[i][j]) is int) and (self.obj.GetStatus()==2)):

    #append comming input of keybord to ItemNumberArry
    self.itemNumberArray.append(int(self.MATRIX[i][j]))
    #update array to string
    itemNumberArrayCleared = [str(a) for a in self.itemNumberArray]

    global TempItemNumber  #define global variable
    #save clear number string with removed "," and "[]"
    TempItemNumber=(''.join(itemNumberArrayCleared))
    print "TempItemNumber:"+TempItemNumber
    #display print
    d1.run("ZASTAVENO","Zadejte cislo dilu:",str(TempItemNumber))
```

Ukázka kódu 9 – Ukázka akce tlačítka „D“ [autor]

Chod laserové brány zajišťuje třída *appLaser.py*. Třída funguje na bázi vlákna, kdy po inicializaci instance třídy běží na pozadí. Hlavní metodou třídy *controll* se kontroluje, kdy se objeví na výstupu zařízení laseru hodnota 0.

```
def controll(self):
        if GPIO.input(23)==0:
            Laser.counter+=1  #++ counter
            Laser.count+=1 #++ count
            #run display
            d1=Display()
            #print to display "START" and variable count
            d1.run("ZAPNUTO",str(Laser.GetCounter()),"")
            print "Tempcount",Laser.GetCount()
             #if obstacle stop wait and dont count
            while GPIO.input(23)==0:
                time.sleep(0.2)
        if (self.run==True):
            global th
            th=threading.Timer(0.2, self.controll)
            th.start()
```

Ukázka kódu 8 – Metoda controll třídy appLaser.py [autor]

Poté je inkrementována proměnná *counter* o jedničku a zobrazí hodnotu proměnné na displeji. Pokud na laseru není hodnota 0, běh vlákna pokračuje a čeká na změnu hodnoty výstupu. Metoda *controll* třídy *appLaser.py* je znázorněna v ukázce kódu číslo 8.

Třída *appRfid.py* zajišťuje přihlašování operátorů a seřizovačů pomocí RFID čipů. Mezi metody této třídy patří *startThreading*, *controll*, *end_read* a *run*. Metoda *controll* nejdříve čeká na přiložení tagu ke čtečce, a to pomocí metod importované knihovny rfid. Poté kontroluje, jestli získaný řetězec znaků získaných z čipu je správný a uloží ho do proměnné. Po úspěšném přečtení id tagu je zavolána metoda *end_read*, která zastaví vlákno rfid a ukončí činnost rfid čtečky. Dále je zde metoda *run*, která slouží ke spuštění čtení rfid tagu pouze jednou, tedy bez použití vlákna.

Pro získání potřebných proměnných ze všech tříd je použita třída *dataGet.py*. Třída importuje knihovny všech aplikací. Po inicializaci proměnných je použita metoda *get_ip_address*, pro získání IP adresy zařízení, které je potom použito v databázi jak identifikátor, aby bylo poznat, ze kterého zařízení data pocházejí. Následně pro každou jednotlivou aplikaci je vytvořena jedna metoda, která získává potřebné proměnné pomocí systémové funkce „self“. Metoda k získání IP adresy zařízení je zobrazena v ukázce kódu číslo 10.

```
1. #method to get IP address of RPi
2.     def get_ip_address(self,ifname):
3.         s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
4.         return socket.inet_ntoa(fcntl.ioctl(
5.             s.fileno(),
6.             0x8915,  # SIOCGIFADDR
7.             struct.pack('256s', ifname[:15])
8.         )[20:24])
```

Ukázka kódu 10 – Metoda k získání IP adresy zařízení [autor]

Po získání všech potřebných dat je nutné data odeslat, to zajišťuje třída *dataSend.py*. Tato třída za pomocí vlákna sbírá data, která jsou každou minutu odeslána na webový server. Pro tuto funkci slouží metoda *controll*, kde je definována adresa serveru a url request. Textový řetězec dat je odeslán na adresu webového serveru, poté jsou všechny dočasné proměnné vynulovány.

Pro potřeby zkoušení a sbírání informací slouží třída *classlog.py*, která zapisuje všechny chybové zprávy do lokálního textového souboru. V pythonu jsou 4 druhy zpráv a to informační, upozornění, a chyby. Každá má specifické id nebo textový řetězec. Informace má hodnotu 0, upozornění hodnotu 1 a chyba hodnotu 2.

Pro správné fungování všech funkcí a senzorů jsou použity knihovny třetích stran, které jsou ve složce Libraries. Jedná se o knihovnu *rfid.py* a *util.py*, které zajišťují funkčnost RFID čtečky,

6.2 Serverová aplikace

Pro serverovou aplikaci byl použit programovací jazyk PHP. Aplikace slouží k získávání hodnot ze všech zařízení z výroby a ukládání dat do MySQL databáze. Nejprve jsou definovány proměnné nutné pro vytvoření připojení do databáze. Potřebné jsou proměnné IP, kde je definován databázový server. Dále host, který obsahuje IP databázového serveru. Pro připojení k databázi jsou také nutné přihlašovací údaje, pro tento účel byl vytvořen speciální servisní účet, jehož údaje jsou uloženy do *uid*, kde je uloženo jméno a *passVal*, obsahující heslo účtu. Jako poslední je proměnná *database* s názvem databáze. Poté je vytvořeno připojení pomocí řetězce ve tvaru:

```
$connection = odbc_connect("Driver={SQL Server};Server=$host;
Database=$database;",$uid, $passVal )
or die("Connection could not established");
```

Pro získané hodnoty ze zařízení jsou použity proměnné:

- *Id* – slouží k identifikaci zařízení
- *Count* – naměřený počet kusů
- *Status* – stav zařízení
- *itemNumber* – číslo vyráběného dílu
- *faultNumber* – číslo poruchy
- *rfidNumberOperator* – id operátora
- *rfidNameOperator* – jméno a příjmení operátora

- *rfidGroupOperator* – skupina operátora
- *rfidNumberMaintenance* – id seřizovače
- *rfidNameMaintenance* – jméno a příjmení seřizovače
- *rfidGroupMaintenance* – skupina seřizovače

Zasílání dat do databáze pomocí SQL dotazu je zobrazeno v ukázce kódu číslo 11.

```
1. if( $connection ) {
2.
3.      echo "Connection established \n";
4.
5.      $query = "INSERT INTO Iot2 (
6.      id,
7.      count,
8.      status,
9.      itemNumber,
10.     faultNumber,
11.     rfidNumberOperator,
12.     rfidNameOperator,
13.     rfidGroupOperator,
14.     rfidNumberMaintenance,
15.     rfidNameMaintenance,
16.     rfidGroupMaintenance)
17.
18.     VALUES (
19.     '".$id."',
20.     '".$count."',
21.     '".$status."',
22.     '".$itemNumber."',
23.     '".$faultNumber."',
24.     '".$rfidNumberOperator."',
25.     '".$rfidNameOperator."',
26.     '".$rfidGroupOperator."',
27.     '".$rfidNumberMaintenance."',
28.     '".$rfidNameMaintenance."',
29.     '".$rfidGroupMaintenance."')";
30.
31.    $stmt = odbc_exec( $connection, $query);
32.    if( $stmt === false ) {
33.         die( print_r( odbc_error(), true));
34.     }
35.    else {
36.    echo "Data byla ulozena \n";
37.        }
38.
39. }else{
40.     echo "Connection could not be established \n";
41.     die( print_r( odbc_error(), true));
42. }
```

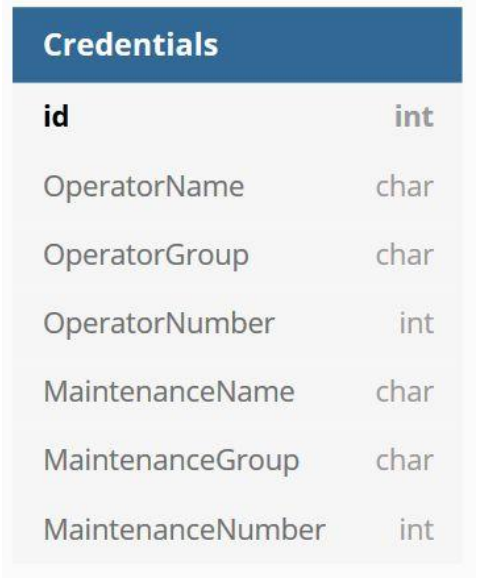
Ukázka kódu 11 – Posílání dat do databáze [autor]

6.3 MySQL databáze

Pro ukládání dat byla použita MySQL databáze s dvěma tabulkami. První pro údaje o operátorech a seřizovačích s názvem *Credentials* a druhá pro ukládání hodnot ze zařízení s názvem *Data*.

6.3.1 Tabulka Credentials

Tabulka *Credentials* slouží k ukládání a načítání hodnot RFID tagu. Diagram tabulky je zobrazen v obrázku číslo 21. Jako privátní klíč této tabulky je použito pole id. Jméno a příjmení je ukládáno do polí *OperatorName* a *MaintenanceName*. Pro zobrazení skupiny jednotlivých uživatelů je použito pole *OperatorGroup* a *MaintenanceGroup*. RFID číslo je uloženo v *OperatorNumber* a *MaintenanceNumber*.

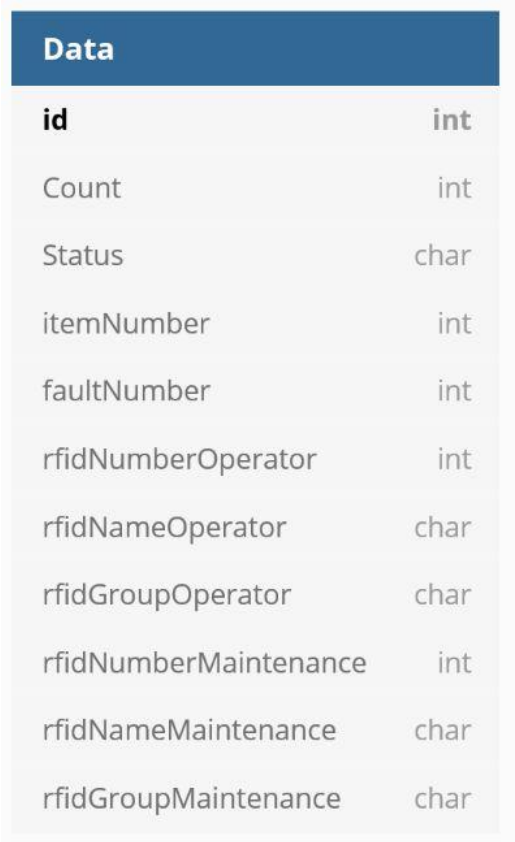


Obrázek 21 – Diagram tabulky Credentials [autor]

6.3.2 Tabulka Data

Tabulka Data pracuje s hodnotami získanými z Raspberry Pi. Tyto data jsou dále zpracovávána a použita k vytváření výrobních přehledů a statistik. Diagram tabulky je vyobrazen na obrázku číslo 22. Stejně jako u tabulky *Credentials* je i v této tabulce použito jako privátní klíč pole id. Počet vyrobených kusů obsahuje pole *Count* a status zařízení zobrazuje pole Status. Číslo stroje a číslo poruchy je ukládáno do pole *itemNumber* a *faultNumber*. Údaje o uživatelích jako jsou RFID čísla, skupiny nebo jména jsou vkládána do polí *rfidNumberOperator*, *rfidNumberMaintenace*,

rfidGroupOperator, *rfidGroupMaintenance*, *rfidNameOperator* a *rfidNameMaintenance*.



Obrázek 22 – Diagram tabulky Data [autor]

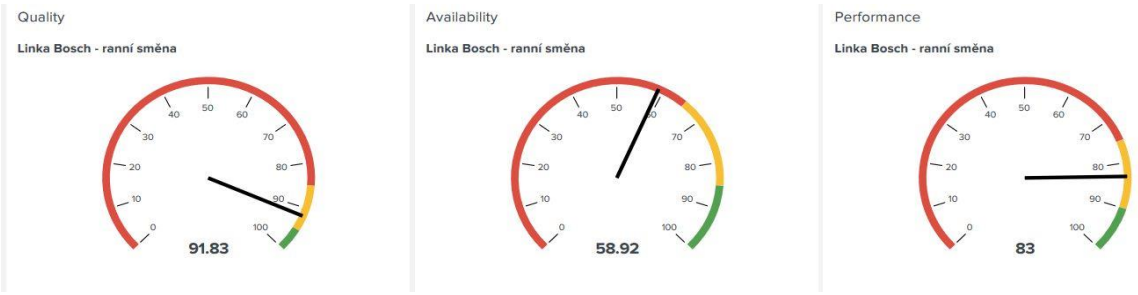
6.4 Splunk

Pro zpracování dat byl použit software Splunk. Tento softwarový produkt, který umožňuje vyhledávat, analyzovat a vizualizovat data získaná z komponent IT infrastruktury nebo firmy. Splunk přijímá data z webových stránek, aplikací, senzorů zařízení a tak dále. Poté, co je definován zdroj dat, indexuje datový tok a analyzuje jej do série jednotlivých událostí, které jdou zobrazit a vyhledat [22].

V projektu je pracováno s daty, které jsou získávány z SQL databáze. Na data se lze poté dotazovat pomocí příkazů podobných SQL syntaxi. Pomocí příkazů byly vytvořeny dashboardy, které zobrazují přehled o datech získaných z výroby. Na lince jsou zobrazeny následující data:

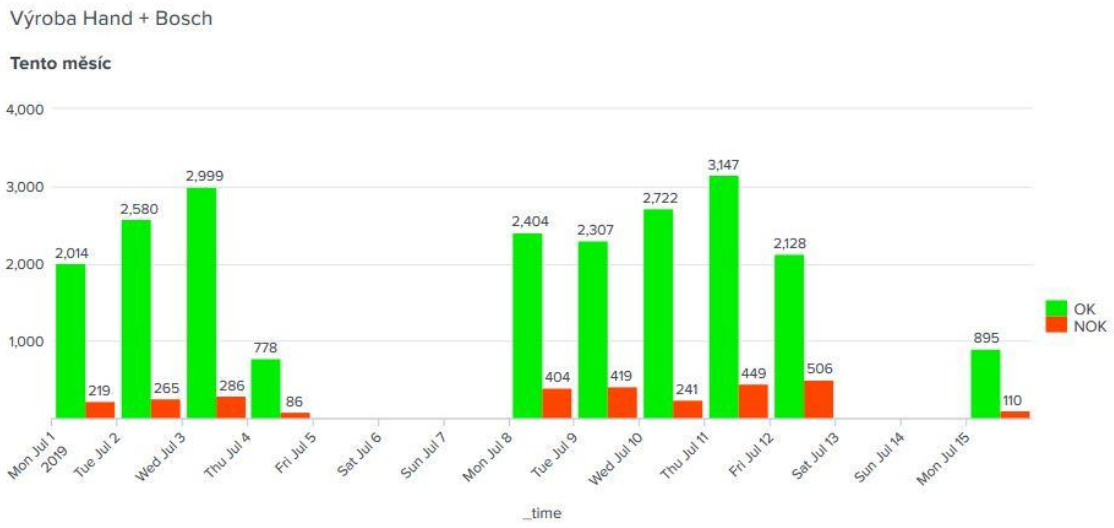
- Ukazatel kvality, který je spočítán vydělením celkového množství dobrých kusů s celkovým množstvím všech výrobků. Jeho hodnota je v procentech a je v rozmezí od 0 do 100.
- Ukazatel dostupnosti, který je spočítán vydělením skutečného času výroby s plánovaným časem výroby. Jeho hodnota je v procentech a je v rozmezí od 0 do 100.
- Ukazatel výkonu, který je spočítán vydělením skutečného vyrobeného množství s teoreticky vyrobeným normovaným množstvím. Jeho hodnota je v procentech a je v rozmezí od 0 do 100.
- Počet vyrobených kusů za směnu se liší podle linek, některé linky mají jen ranní a odpolední směnu a některé i noční. Počet vyrobených kusů je zobrazen jako suma všech výrobků v časovém rozmezí směny.
- Graf OK/NOK kusů za směnu ukazuje, kolik bylo vyrobeno dobrých kusů a kolik kusů s nějakou vadou. Graf je zobrazen formou výseče a jeho hodnota je za jednu směnu.
- Ukazatel OK/NOK kusů za měsíc ukazuje poměry mezi dobrými a špatnými kusy za celkový měsíc na konkrétní lince.

Na obrázku číslo 23 jsou zobrazeny dashboardy ukazatelů kvality, dostupnosti a výkonu. Obrázek 24 vyobrazuje počet vyrobených kusů dané směny a obrázky 25 a 26 zobrazují počet dobrých a špatných kusů v různých souvislostech.

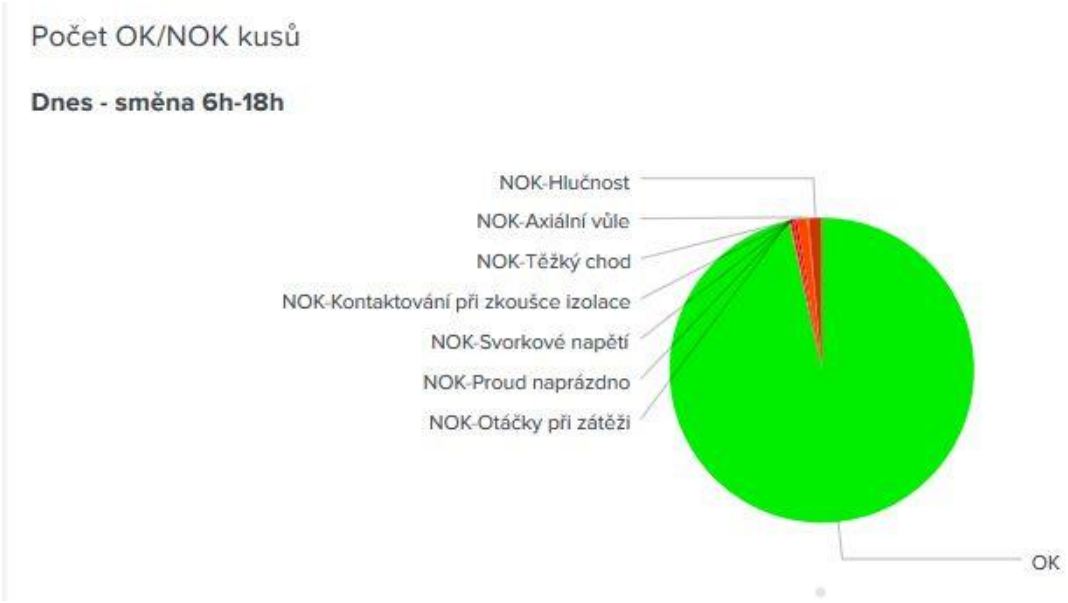


Obrázek 23 – Splunk – dostupnost, kvalita, výkon [autor]

Obrázek 24 – Splunk – počet vyrobených kusů [autor]



Obrázek 25: Splunk-graf OK a NOK kusů [autor]



Obrázek 26: Splunk-typy poruch [autor]

7 Shrnutí výsledků

Tato práce se věnovala návrhu a nasazení zařízení na měření efektivity výroby ve výrobní firmě.

Přínos práce spočívá v zjištění všech potřebných dat nutné ke správnému vyhodnocení efektivity výroby. Získaná data a zkušenosti pomohou ke snadnější implementaci komerčního řešení.

Výsledné zařízení je kompaktní a velmi uživatelsky přívětivé. Ovládání je intuitivní a jednoduché. Získaná data se shromažďují v databázi a jsou vyhodnoceny a zobrazeny na monitoru vedle výrobní linky, kde operátor vidí v reálném čase přehledy z minulé směny a ze současné směny. Zobrazeny jsou i například průměrné vyrobené kusy a operátor tedy vidí, jestli vyrábí v normě či nikoliv.

Zařízení bylo testováno na několika výrobních linkách s kladnými výsledky. Nasazení a nastavení zařízení na výrobní linku je možné ve velmi krátké době. Již od druhého dne je možné sledovat výrobní data. I po delším testování nebylo zjištěno přehřívání ani větší výpadky zařízení způsobené horkým a prašným prostředím výrobní haly. Zařízení tedy splnilo svůj účel a nyní je již připravováno komerční řešení, které bude shromažďovat ještě více údajů a posune firmu o krok blíže k Industry 4.0 a získání statutu chytré továrny.

8 Závěry a doporučení

V úvodní části práce je definován pojem OEE, neboli postup pro měření efektivnosti výrobních strojů i samotné výroby. Dále je popsán základní návrh aplikace pro firmu. Ten doporučuje funkce a vlastnosti budoucího systému. V návrhu je pracováno se scénářem a diagramy aplikace. V hardwarové části práce je popsán jednodeskový počítač Raspberry Pi, který je řídícím centrem celé práce. Po zavedení počítače je popsán použitý operační systém včetně zavedení a instalace systému a možnosti vzdáleného připojení. Použité vstupní komponenty jako je OLED displej, který slouží k interakci systému s uživatelem, klávesnice sloužící pro zadávání výrobních dat a stavové tlačítka, která zajišťují zapnutí, vypnutí či pozastavení aplikace. V neposlední řadě hardwarové části je popsán i návrh vlastního plošného spoje, která je osazen přímo na Raspberry Pi a ke kterému se připojují jednotlivé komponenty pomocí konektorů. RFID čtečka pro přihlašování uživatele je také součástí hardwarové části projektu spolu s laserovou bránou pro počítání množství kusů. Všechny hardwarové komponenty jsou osazeny do krabičky navržené na míru a vytisknuté na 3D tiskárně.

V softwarové části je jak popsán jak chod aplikace běžící na Raspberry Pi tak i serverové aplikace hostované na Microsoft Windows serveru. Ukládání a práce s daty obstarává SQL databáze a celkové vyhodnocení a vizualizace získaných dat bylo provedeno pomocí nástroje Splunk. V závěru práce jsou shrnuty výsledky a využití práce v praxi.

Při nasazování zařízení bylo zjištěno, kolik důležitých dat z výroby nebylo v minulosti sledováno. Tento fakt stěžoval a přidělával práci především kvalitě výroby, která pracovala s nepřesnými daty a tak nemohla zefektivnit výrobu.

Po implementaci zařízení jsou nyní sledována a vyhodnocována důležitá data, která usnadňují práci výrobní kvalitě a umožňují ji zefektivnit jednotlivé výrobní procesy.

Do budoucna je plánované implementovat komerční řešení, které bude ještě přesněji měřit efektivitu výroby a získávat více dat.

V současné době již firma hledá dodavatele komerčního řešení. K sestavení požadavků byla použita data zpracované v této bakalářské práci.

9 Seznam použité literatury

1. **IoT portál.** Iot-portal.cz. [Online] [Citace: 17. 07 2019.] https://www.iot-portal.cz/co-je-iot/.

2. **Antonín Vojáček.** Automatizace.hw.cz. [Online] [Citace: 17. 07 2019.] https://automatizace.hw.cz/mimochodem/co-je-se-skryva-pod-vyrazy-industry-40-prumysl-40.html.

3. **Boost Solutions.** Boost.solutions. [Online] [Citace: 18. 07 2019.] https://boost.solutions/blog/top-9-trends-industry-4-0/.

4. **Vorne Industries Inc.** Oee.com. [Online] [Citace: 17. 07 2019.] https://www.oee.com/oee-factors.html.

5. **Upton Eben, Halfacree Gareth.** *Raspberry Pi: uživatelská příručka.* Brno : Computer Press, 2013. ISBN 978-80-251-4116-8.

6. **Raspberry Pi Foundation.** Raspberrypi.org. [Online] [Citace: 17. 07 2019.] https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus.

7. —. Raspbian.org. [Online] [Citace: 17. 07 2019.] https://www.raspbian.org/.

8. **ECLIPSERA s.r.o.** Navody.arduino-shop.cz. [Online] [Citace: 17. 07 2019.] https://navody.arduino-shop.cz/navody-k-produktum.

9. **raspberrypi.** Github.com. [Online] [Citace: 17. 07 2019.] https://github.com/raspberrypi/noobs.

10. **David Čápka.** Itnetwork.cz. [Online] [Citace: 17. 07 2019.] https://www.itnetwork.cz/software/ssh/ssh-konfigurace-a-pripojeni.

11. **maxnet.** Berryterminal.com. [Online] [Citace: 17. 07 2019.] https://www.berryterminal.com/doku.php/berryboot.

12. **Tomáš Šustek.** Pocitacovasit.php5.cz. [Online] [Citace: 17. 07 2019.] https://www.berryterminal.com/doku.php/berryboot.

13. **Sharpened Productions.** Techterms.com. [Online] [Citace: 17. 07 2019.] https://techterms.com/definition/pcb.

14. **Vishay Ltd.** vishay.com. [Online] [Citace: 7. 21 2019.] https://www.vishay.com/docs/37902/oled128o064dbpp3n00000.pdf.

15. **OLED-Info.** Oled-info.com. [Online] [Citace: 17. 07 2019.] https://www.oled-info.com/oled-introduction.

16. **ECLIPSERA s.r.o.** Arduino-shop.cz. [Online] [Citace: 17. 07 2019.] https://arduino-shop.cz/573-displeje/.
17. **HW server s.r.o.** Vyvoj.hw.cz. [Online] [Citace: 17. 07 2019.] https://vyvoj.hw.cz/navrh-obvodu/.
18. **Camcode Division.** 2.camcode.com. [Online] [Citace: 17. 07 2019.] https://www2.camcode.com/asset-tags/what-are-rfid-tags/.
19. **NXP Semiconductors.** nxp.com. [Online] [Citace: 21. 7 2019.] https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf.
20. **Laskarduino.** Laskarduino.cz. [Online] [Citace: 17. 07 2019.] https://laskarduino.cz/rfid.
21. **Python Software Foundation.** Python.org. [Online] [Citace: 17. 07 2019.] https://www.python.org/doc/.
22. **The PHP Group.** Php.net. [Online] [Citace: 17. 07 2019.] https://www.php.net/manual/en/intro-whatis.php.
23. **Oracle Corporation.** Mysql.com. [Online] [Citace: 17. 07 2019.] https://dev.mysql.com/doc/refman/8.0/en/.
24. **Splunk Inc.** Splunk.com. [Online] [Citace: 17. 07 2019.] https://www.splunk.com/en_us/about-splunk.html.

10 Přílohy

Ukázka hlavní spouštěcí třídy app.py

```
#import libraries

import RPi.GPIO as GPIO
import threading
from threading import Timer
import MySQLdb
import signal
import time
import sys
from pirc522 import RFID
from time import sleep
import Adafruit_GPIO.SPI as SPI
import Adafruit_SSD1306
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import subprocess

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)


#from app.appDisplay import Display
from app.appButton import Button
from app.appKeyboard import Keyboard
from app.appRfid import Rfid
from app.appDb import DbOperator
from app.appDisplay import Display
from app.dataSend import dataSend

class Main():
    th=""
    global send
    send=dataSend()
    global supervisor
    #variable to controll if thread is running only one times in current status,
    #if status is same do thread only once..
    #if status are changed do thread in current status only once again
    supervisor=0

    def __init__(self):
        pass

    #send threading controll
    def startThreading(self):
        global th
```

```
47.         self.run=True
48.         th=threading.Timer(0.2,self.controll)
49.         th.start()
50.
51.     def stopThreading(self):
52.         self.run=False
53.
54.     def controll(self):
55.         global send
56.         global supervisor
57.         #controll status of Button and supervisor value..
58.         #if supervisor is not same as status -
   >thread is run before
59.         if ((Button.GetStatus()==1) and (supervisor!=1)):
60.             send.startOnce()
61.             send.startThreading()
62.             supervisor=1
63.         if ((Button.GetStatus()==2) and (supervisor!=2) and (sup
   ervisor!=0)):
64.             send.stopThreading()
65.             send.startOnce()
66.             supervisor=2
67.         if ((Button.GetStatus()==3) and (supervisor!=3)):
68.             send.stopThreading()
69.             send.startOnce()
70.             supervisor=3
71.         if (self.run==True):
72.             global th
73.             #define threading
74.             th=threading.Timer(0.2, self.controll)
75.             th.start()   #start threading
76.
77.
78.
79.     def run(self):
80.         #AUTORIZATION WITH RFID
81.         print "prilozte chip"
82.         d1=Display()
83.         d1.run("ZASTAVENO","prilozte chip","")
84.         chip=Rfid()  #creating instance of Rfid (app/appRfid)
85.         chip.run()   #run rfid read
86.         d1.run("nacitam...","","")
87.         variable=Rfid.getVariable()
88.         database=DbOperator()
89.         database.readValue(variable)
90.         d2=Display()
91.         d2.run("Prihlaseny uzivatel:",str(DbOperator.GetOperator
   Name()),"")
92.         time.sleep(3)
93.
94.
95.         #ENTERING A PART NUMBER
```

```
96.         print "zadejte cislo dilu:"
97.         d1=Display()
98.         d1.run("ZASTAVENO","Zadejte cislo dilu:","")
99.         k1=Keyboard(Button)
100.              k1.startThreading()
101.
102.
103.              #start BUTTON threading  (pin,'action')
104.              b1=Button(17,'Start')  #define button
105.              b1.startThreading()    #start threading
106.              y1=Button(27,'Stop')
107.              y1.startThreading()
108.              r1=Button(22,'Pause')
109.              r1.startThreading()
110.
111.     if __name__ == "__main__":
112.          m1=Main()
113.          m1.run()
114.          m1.startThreading()
```

Oskenované zadání práce

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2020/2021

Studijní program: Aplikovaná informatika
Forma studia: Kombinovaná
Obor/kombinace: Aplikovaná informatika (ai3-k)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	**David Drahokoupil**
Osobní číslo:	**I1800712**
Adresa:	Prasek 181, Prasek, 50401 Nový Bydžov, Česká republika
Téma práce:	Využití Raspberry Pi pro měření efektivity výroby
Téma práce anglicky:	
Vedoucí práce:	Ing. Jan Štěpán Katedra informačních technologií

Zásady pro vypracování:

Úvod
Definice požadavků na zařízení
Návrh hardware
Implementace hardware
Sběr dat
Závěr

Seznam doporučené literatury:

Exploring Raspberry Pi: Interfacing to the Real World with Embedded Linux, Derek Molloy, 978-1119188681
Raspberry Pi Cookbook: Software and Hardware Problems and Solutions, Simon Monk, 978-1491939109
Modern Programming Languages: A Practical Introduction, Webber, 978-1590282502

Podpis studenta: Datum: 24.7.2019

Podpis vedoucího práce: Datum: 29.7.2019