SmartBrew

Tímový projekt

Tím 6

Alexandra Frniaková
Marek Krchňavý
Michaela Nemcová
Lukáš Novota
Peter Stríž
Marek Vajda

Kontakt:
tim6_tp2021@googlegroups.com

Riadenie projektu

Úvod

Tento dokument obsahuje informácie k riadeniu projektu. Poskytuje základné členenie na **Členovia, Zápisnice, Šprinty, Metodiky**.

Členovia

- Kompetencie jednotlivých členov a podiel práce na projekte v jednotlivých šprintoch.

Zápisnice

- Zápisnice z jednotlivých týždňov, spolu s ich templatom.

Šprinty

- Vyhodnotenia jednotlivých šprintov spolu s ich templatom.

Metodiky

- Rôzne metodiky a usmernenia pre efektívnu prácu.

*Last updated on **22. 11. 2021** by **Michaela Nemcova***

Členovia

Člen	Kompetencie
Alexandra Frniaková	- front-end - zápisnice - organizácia a správa odovzdávaných dokumentov
Marek Krchňavý	- databáza (návrh a implementácia) - back-end (dopyty na databázu)
Michaela Nemcová	- front-end - správa dokumentácie - zápisnice
Lukáš Novota	- mock server pre testovanie - back-end (testy)
Peter Stríž	- scrum master - moduly (príprava a implementácia) - návrhy častí systému - automatizácia vývoja
Marek Vajda	- back-end (API pre moduly a pre front-end) - testy - automatizácia vývoja

Uvedené kompetencie nie sú fixné, členovia sa môžu a aj sa podieľajú tiež na iných častiach projektu.

Hodnotenie členov

Šprint / Člen	Saška	Marek K.	Miška	Lukáš	Peťo	Marek V.
01 - Zlatý Bažant	17 %	16 %	17 %	17 %	17 %	16 %

Šprint / Člen	Saška	Marek K.	Miška	Lukáš	Peťo	Marek V.
02 - Pilsner Urquell	17 %	17 %	16 %	16 %	17 %	17 %
03 - Corgoň	16 %	17 %	16 %	17 %	17 %	17 %
04 - Krušovice	---	---	---	---	---	---

*Last updated on **22. 11. 2021** by **Michaela Nemcova***

Zápisnice

Zápisnica template

> **TODO**
>
> Pri vypĺňaní template je potreba prepísať hore v hlavičke atribút **title** do formátu: **DD.MM.YYYY** *(napr. 04.10.2021)*. A sidebar*position na najnižšie číslo zo zápisníc* (napr. predošlá má číslo 3, tak moja bude mať číslo 4)_.

Zápisnica - DD.MM.YYYY

Zhrnutie taskov

Zhrnutie, kto je ako na tom so svojimi taskami, ako progressujú stories a podobne. Stačí v odrážkach.

Prezentácia vypracovaných taskov

Pokiaľ niekto mal task ako analýzu, alebo je to niečo, čo potrebuje celý tím odsúhlasiť.

Identifikácia nových problémov

S čím sme mali pri vypracovávaní taskov problém a aké nové issues vznikli.

Vytvorenie nových stories/taskov

Aké nové tasky vznikli a komu boli priradené.

Stories

Novovzniknuté stories.

Tasky

Novovzniknuté tasky.

Diskusia

Iné témy, ktoré sa prebrali na stretnutí.

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 27.9.2021

Zhrnutie taskov

- prvé skupinové stretnutie týkajúce sa práce na projekte

Vytvorenie nových stories/taskov

- prebehlo bližšie oboznamovanie sa s témou projektu
- základné spoločné stretnutia budú prebiehať každý týždeň v pondelok
- na komunikáciu v tíme bude využivaný Discord
- prostredie na riedenie scrumu - Jira

Dlhodobé úlohy

- kontrola scrumu - Peter Stríž
- dokumentácia - Michaela Nemcová

Stories

- tím si má do dalšieho stretnutia naštudovať scrum (vhodnú granulitu taskov,...)

Tasky

- príprava Jira - Peter Stríž
- príprava Discord - Marek Vajda

Diskusia

- na stretnutí boli opísané technológie a postupy potrebné pri realizácii projektu

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 4.10.2021

Zhrnutie taskov

- Jira: je vytvorené prostredie na pridelenie úloh jednotlivým členov tímu podľa princípu scrum
- Discord: pripravený na komunikáciu členov tímu
- Github repozitár: založený na ukladanie a prístup k spoločným kódom
- Docusaurus: založený na dokumentáciu

Identifikácia nových problémov

- boli určené hlavné epiky:
  - Ako sládek (osoba, ktorá varí pivo) chcem vedieť komunikovať so senzormi a efektormi (vytvorenie endpointov, modulov,...)
  - Chcem, aby sa pivo varilo automaticky: príprava receptu = nastav teplotu a udržuj ju, reaguj na eventy - zahrievaj na určitú teplotu. Je potrebné určiť architektúru systému, databázy (teda spôsob ukladania dát)
  - Chcem vidieť stav zatiadení na monitore - potrebné vizualizovať
  - My ako tím chceme splniť všetky podmienky na úspešné zvládnutie predmetu (skompletizovanie webovej stránky, dokumentácie, dodržanie správnej formy odovzdaného diela)

Postupy a riešenia:

- ovládanie a vizualizácia dát na monitore - webová aplikácia zobrazujúca sa v browseri
- frontend: JavaScript, React (interaktivita, pridanie údajov (receptu), tie sa ukladajú, dajú sa pozmeniť; tabuľky s hodnotami,...)
- backend: Node.js, možnosť využiť postupy z existujúcej práce (Flask), Sequelize
- databáza: Postgresql

Vytvorenie nových stories/taskov

- naštudovať si komunikáciu pomocou mqtt

Stories

- Webová stránka tímu - Marek Vajda
- Sprevádzkovať Docker na Raspberry Pi (kontajnery ...) - Marek Vajda
- Prieskum procesov pivovaru - Alexandra Frniaková
- Priemyselná vizualizácia scadu (prieskum) - Michaela Nemcová
- Analýza nástrojov (HW) - Peter Stríž

Tasky

- Update zápisníc - Alexandra Frniaková
- Vytvorenie templates pre dokumentáciu - Michaela Nemcová
- Nástroje na tvorbu dokumentácie - Peter Stríž
- Export z Jira - Peter Stríž

Diskusia

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 11.10.2021

Zhrnutie taskov

Všetkým členom sa podarilo vypracovať tasky z minulého týždňa.

Identifikácia nových problémov a riešení

Asynchrónnosť systému

Problém

Kroky receptu (jednotlivé moduly) nepôjdu úplne synchrónne a lineárne.

> Príklad: modul č. 1 dostane príkaz "zohrej na XY °C a potom teplotu udržuj XY minút" okrem toho môžem chcieť miešať, tak pošlem do 2. modulu príkaz "miešaj na XY otáčok po dobu XY minút" keď 1. modul dokončí, tak backend môže synchronizovať moduly (2. modul nastaví na 0rpm)

TL;DR - moduly musia vedieť bežať asynchrónne

Návrh riešenia

- bude centrálny backend, a moduly budú periodicky oznamovať backendu svoj stav (dáta pôjdu do DB) a **zároveň** sa môžu spýtať, čo majú robiť
- každý modul bude mať nastavený pull-rate --> koľko času musí prejsť medzi dopytmi
    - tento pull-rate musí byť variabilný a musí sa dať v priebehu varenia meniť
- podľa pull-rate si bude backend checkovať, či je modul živý alebo mŕtvy a môže to rovno backend vyriešiť
- backend musí byť robustný, aby vedel spravovať každý modul
- modul bude vedieť nahlásiť, v akom stave je (IN PROGRESS, DONE, FAILED) a sám si podľa pull-rate bude pýtať od backendu, čo má vykonať

Architektúra HW

Návrh riešenia

- i. ESP má oba motory
- ii. ESP má senzory z prvej nádoby
- iii. ESP má senzory z druhej nádoby
- ESPčka sú pripojené cez WiFi na RPI
- moduly si pýtajú veci od backendu, nie priamo z DB
- služby pre moduly napr.
  - zapísať hodnoty zo senzorov
  - vypýtať, čo má modul robiť
  - stav vykonávania

Rozloženie logiky !

Problém

Chceme všetku logiku na backende (moduly budú "blbé"), alebo nejakú logiku delegovať na moduly?

- ak vypadne internet, treba zabezpečiť, aby modul sám zastal

Návrh riešenia

> ⚠ **TBD**
>
> Tím musí vypracovať analýzu, ako by varenie prebiehalo pri oboch konfiguráciách.

Návrh front-endu

Front-end musí odrážať granularitu receptu a zároveň byť user-friendly aj pre ľudí so základnými počítačovými skillmi.

Front-end by mal vedieť načítať údaje pre časti pivovaru z nejakého konfiguračného súboru.

Granularita receptu

Problém

Akú granularitu zvoliť pre recept, aby podporoval ukladanie do DB, zobrazenie na FE a asynchrónne vykonávanie inštrukcií.

Návrh riešenia

- recept bude rozdelený na **KROKY**
- KROK - súbor inštrukcií, ktoré môžu byť vykonávané asynchrónne, ale musia byť všetky ukončené, aby sa recept mohol posunúť na ďalší krok
- každý krok sa bude skladať z jednotlivých **INŠTRUKCIÍ**
- inštrukcia - najmenšia jednotka granularity receptu, obsahuje:
  - ID zariadenia, na ktoré je mierená
  - konkrétnu inštrukciu, ktorú má vykonať (napr. naštartuj motor, nastav teplotu)
  - parametre inštrukcie (napr. trvanie, teplota)

Priebeh receptu:

- backend si do QUEUE nastaví inštrukcie z prvého kroku
- keď backendu príde dopyt z modulu, tak pokiaľ je v queue nejaká inštrukcia pre daný modul, backend mu pošle danú inštrukciu (modul si ju spracuje sám)
- po odovzdaní všetkých inštrukcií modulom bude backend čakať, kým mu moduly neoznámia, že splnili inštrukciu
- pokiaľ všetky moduly majú inštrukcie splnené, vtedy backend prejde na ďalší krok a proces a zopakuje
- toto celé sa opakuje, kým v recepte existujú nesplnené kroky

Poznámky

- backend si musí (napr. v databáze) držať pre každé zariadenie (efektor) súbor inštrukcií, ktoré vie dané zariadenie vykonať
  - asi bude potrebné ukladať aj to, aké parametre inštrukcia berie

Tasky do najbližšieho stretnutia

- **všetci** - deploy prvej funkčnej iterácie na RPI (aby už to malo aj nejakú jednoduchú statickú HTML stránku)
- **všetci** - kuknúť Git repozitár predošlého tímu
  - budúci týždeň diskusia, čo odtiaľ prevziať
  - vytvoriť si template z motora
  - v repozitári je problém s HW ako je navrhnutý mikropočítač
- **Saška** - dokončiť analýzu varenia piva
- **Saška** - prepísať recept na primitívne a aj pokročilejšie kroky a nastavenia

- **Miška** - návrh frontendu (inšpirovať sa Lego Mindstorms)
- **Peťo** - pohľadať ovládateľný chladiaci efektor (napr. čo sa používa pri domácich pivovaroch)
- **Lukáš a Marek K.** - návrh databázy
- **Lukáš** - analýza TimeScale
- **všetci** - vytvoriť analýzu jednotlivých operácií v oboch prípadoch rozloženia logiky, či by náhodou nenastala nejaká veľmi komplikovaná situácia

Doplnenie informácií

Recept

- bude sa zohrievať celá nádoba (hlavne prvá), v druhej nádobe sa bude chladiť (celá nádoba)
- v prvej nádobe sa bude variť slad
- potom sa zapne výveva, vytvorí podtlak a nasaje obsah prvej do druhej nádoby
- v druhej nádobe sa bude chmeliť, chmel bude v sitku, to sa potom vyberie a pivo bude zrieť v druhej nádobe
- nebude sa merať hladina

Senzory a efektory

Senzory

- teplomer (viacero)
- merač prúdu (pre vývevu, či už všetko precucla)
- tlakový senzor
- menič na motor - vie nastaviť:
    - ako dlho má ísť
    - ako dlho sa má rozbiehať a spomalovať
    - na koľko otáčok má ísť

Efektory

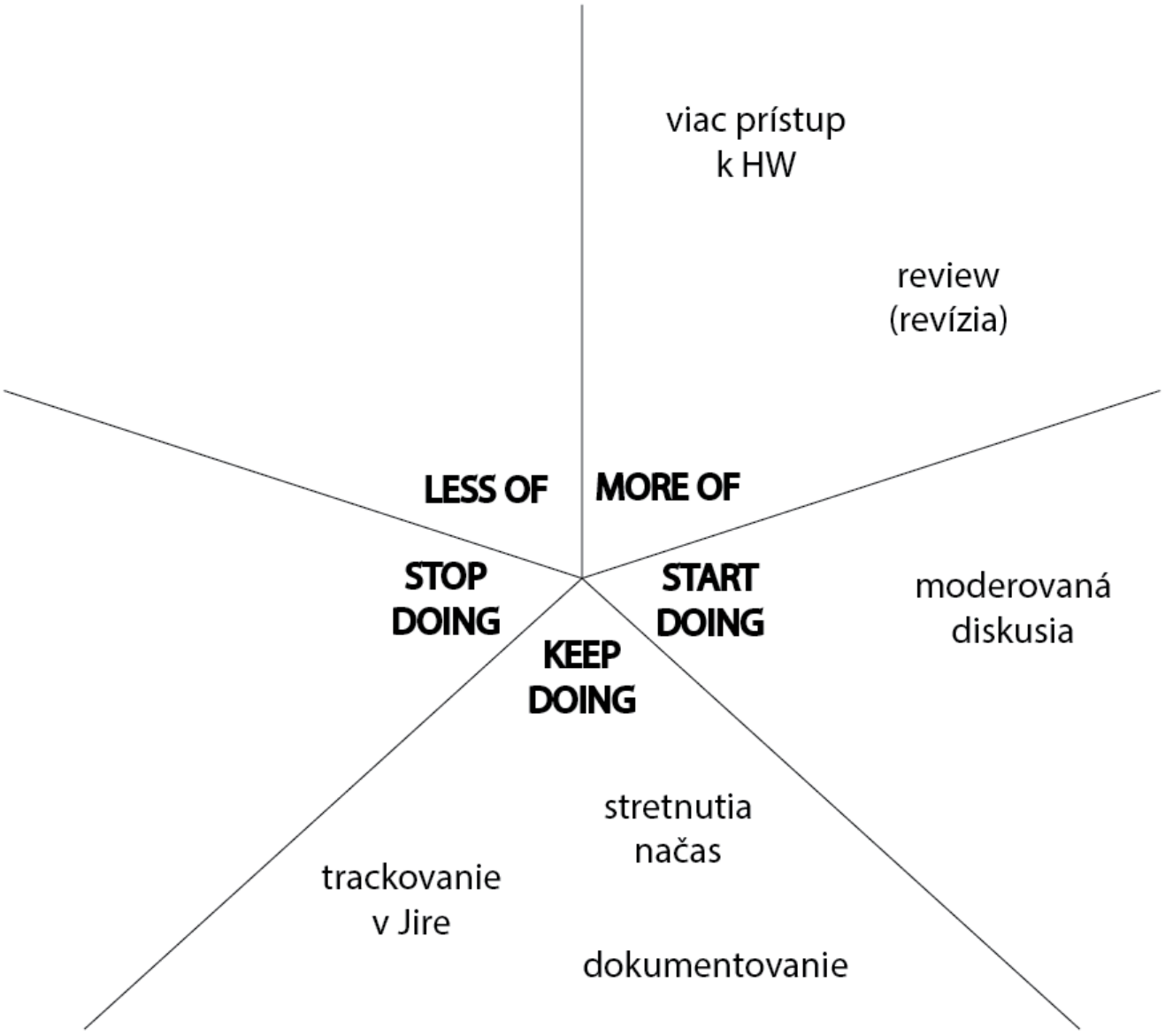
- motor
- servomotor
- ventil na prvej nádobe
- výveva

- ohrevné pásy
- kryostat - zariadenie na nastavenie teploty **(ešte nie je)**
- *nemáme ovládateľné chladiace efektory*

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 18.10.2021

Retrospektíva



Zhrnutie taskov

Všetkým členom sa podarilo vypracovať tasky z minulého týždňa.

Prezentácia vypracovaných taskov

Marek K. prezentoval svoj návrh relačno-entitného modelu DB. Po diskusii sme sa dohodli, že návrh upraví kvôli identifikovaným problémom.

Miška prezentovala návrh front-endu vo Figme. Návrh spravili aj Saška a Peťo. Po diskusii sme sa dohodli, že použijeme Peťov návrh.

Identifikácia nových problémov

- Peťo potrebuje viac HW - kábliky a breadboard

Komunikácia medzi BE a modulmi

- nebudeme využívať MQTT, ale WebSocket
- komunikácia bude ale stále vedená jednosmerne (modul si bude periodicky pýtať inštrukcie od BE)
- paralelizované budú len moduly medzi sebou, nie inštrukcie v rámci modulu (tie budú serializované)

Potreba nových inštrukcií a krokov

Inštrukcia DELAY

Zabezpečuje čakanie modulu na ďalšiu inštrukciu. Modul si odpočet času bude zabezpečovať sám.

Krok ALERT

Krok upozorní používateľa, že je potreba vykonať niečo manuálne (napr. nasypať kvasnice, vybrať chmeľ, ...).

Front-end (aj back-end) by čakali na potvrdenie od používateľa, že danú vec vykonal.

Logovanie

Priebeh varenia

- log - čas ; či to je chyba ; správa
- logovať sa budú všetky údaje dokopy (v jednom stĺpci)

Back-end

- logy z BE stačí ukladať do súborov

Nedostupnosť HW pre testovanie

Potrebujeme testovací server, ktorý by simuloval moduly.

Doplňujúce informácie

- FE si bude každú sekundu pýtať najnovšie dáta, ktoré si BE cachuje v objekte
- handlovanie správnosti hodnôt sa bude vykonávať na module (keby napr. boli vracané zlé teploty)

Vytvorenie nových stories/taskov

Stories

- hotový relačno-entitný model (spolu s validáciou a deploy skriptami)
- chceme vedieť na FE zobraziť teplotu - **definition of done**
  - musí byť možné vytvoriť recept, v ktorom bude inštrukcia "zvýš teplotu"
- virtuálny testovací modul

Tasky

- definovať formáty JSONov pre komunikácie:
  - BE - modul
  - BE - FE
- **Lukáš** - vytvoriť testovací simulátor modulov
- **Miška, Saška** - definovať JSONy, ktoré bude FE posielať na BE a vice versa
- **Peťo** - vytvoriť stránku pre nový šprint v dokumentácii
- **Peťo** - práca na fyzickom zariadení
- **Marek K.** - upraviť návrh DB podľa diskusie

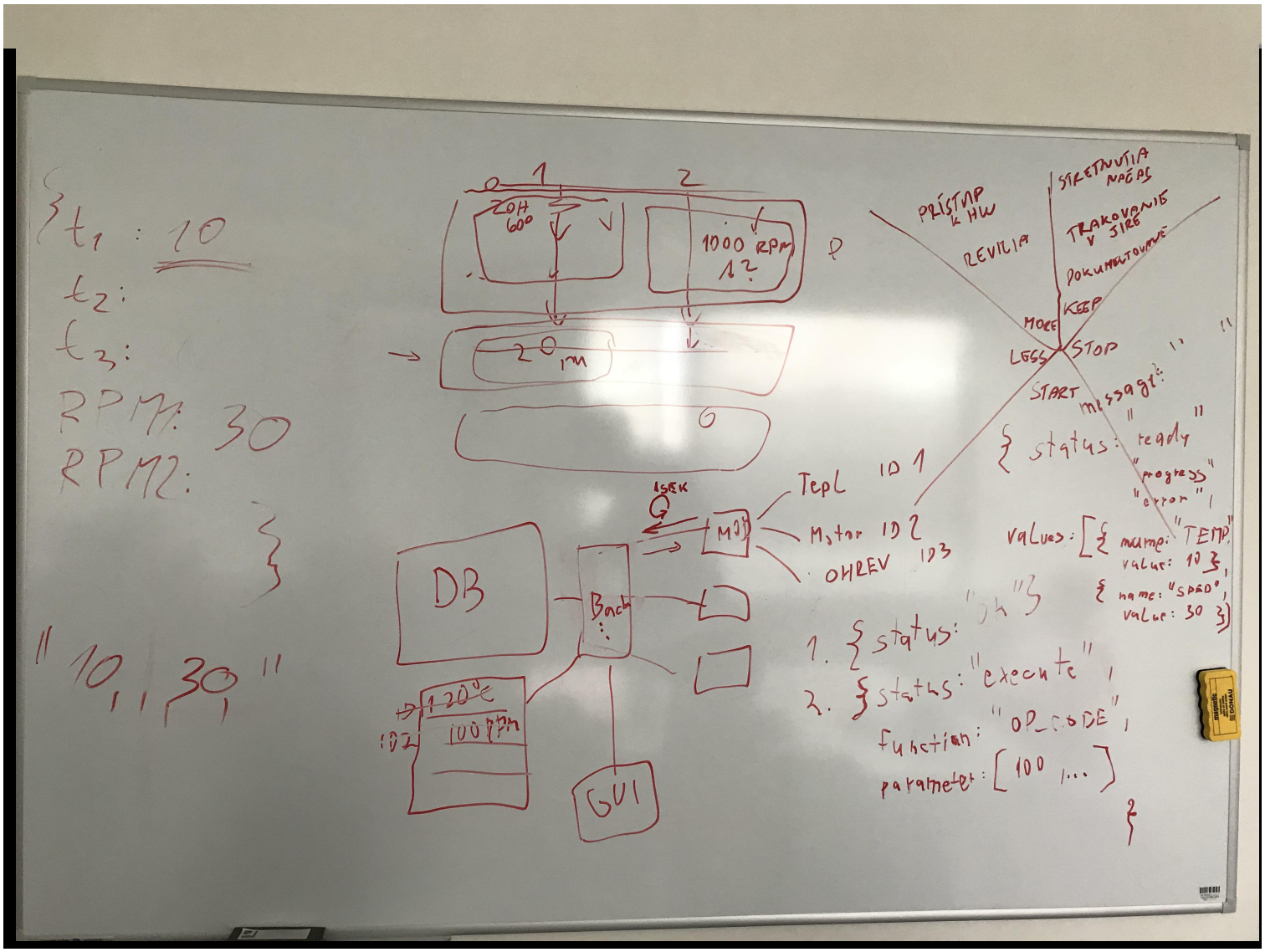
- **Marek V.** - vytvoriť API end-pointy pre komunikáciu s FE

Diskusia

Dohodli sme sa, že je potreba viesť štruktúrovanejšiu a moderovanú diskusiu. Marek V. navrhol metodiku pre vedenie diskusie.

Nápady

Nasledujúce nápady spolu aj s nejakou úplne základnou štruktúrov údajov sme si aj nakreslili:



*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Zápisnica - 25.10.2021

Zhrnutie taskov

Pokračuje priebežná práca na taskoch. Bol určený spôsob ohodnocovania jednotlivých taskov. Jednotkou hodnotenia bude vytvorenie jedného komponentu (frontend/backend), od neho sa bude odvíjať hodnotenie ostatných taskov.

Prezentácia vypracovaných taskov

Bol odprezentovaný upravený návrh frontendu, ktorý pozostával z hlavnej obrazovky: pred pridaním receptu, počas varenia receptu; obrazovka výberu receptu, spolu s neho znením + postupom krokov ktoré je potrebné vykonať; obrazovka pridávania receptov.

Prebieha práca na mock serveri.

Boli priravené prvé komponenty, na ktoré bude nadvezovať tvorba frontendu.

Pokračuje sa na návrhu databázy, spolu s prípravou komunikácie medzi jednotlivými vrstvami aplikácie.

Identifikácia nových problémov

Kroky sa nebudú posielať a spracovávať paralelne – všetko pôjde sériovo, pričom oneskorenia budú minimálne. Frontend si to ale bude vykresľovať ako skupinu (podľa template, do ktorého bude autor receptu len dopísavať hodnoty). Takýchto tamplatov bude viacero, každý krok bude mať vlastný názov, ktorý bude spolu s krokmi zobrazený na frontende.

Z čoho sa skladá informácia pri zasielaní receptu: Step (napr. heat up), parameter (napr. 70 (stupňov celzia)), poradie, id

Recepty bude možné upravovať, niektoré recepty ale budú označené príznakom, že tie sa upravovať dať nedajú.

Postup pri výbere receptu: frontend si vypýta list receptov - názvy aj celé recepty. Po kliknutí na vybraný recept sa zobrazý na druhej strane obrazovky. Po potvrdení receptu sa používateľovi zobrazia podmienky, ktoré musí splniť pred začiatkom varenia. Po ich odkliknutí začína proces varenia.

Vytvorenie nových stories/taskov

Tasky

Práca na fronende - tvorba komponentov a prepojení s backendom. Natiahnutie údajov z backendu a ich zobrazenie - pri danom story pointe sa jedná hlavne o teplotu. - tento task treba rozdeliť na menšie subtasky a rozdeliť medzi ne bodové hodnotenie.

Diskusia

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 02.11.2021

Zhrnutie taskov

Splnené definition of done - produkt je v tejto etape funkčný a otestovaný.

Prezentácia vypracovaných taskov

Ukončenie šprintu a vytvorenie retrospektívy - viď Šprint 2.

Vytvorenie nových stories/taskov

Stories

- Ako používateľ chcem vytvoriť nový recept.
- Ako používateľ chcem spustiť varenie podľa receptu.
- Ako používateľ chcem otestovať novú funkcionalitu na mock serveri.

Tasky

- **Peťo** - prementovať stories na štýl "Ako používateľ chcem ..."
- **Marek K.** - doplniť do dokumentácie example JSON pre vytváranie receptu
- **Marek V.** - nový endpoint pre BE - vytvorenie názvu bloku
- **Lukáš** - rozšírenie funkcionality mock serveru
- **Saška a Miška** - front-end

Diskusia

Doplnenie info k inštrukciám

- pumpa ide ako **delay** inštrukcia
- motor je instantná inštrukcia
- kontrolujeme teplotu

Plány do budúcna

- na FE posielať, koľkokrát bol recept použitý, aby sme mohli ukázať "obľúbené" recepty

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 08.11.2021

Išlo o klasický stand-up.

Zhrnutie taskov

Všetci členovia pokročili na svojich častiach.

- Peťo - demo verzia funkcionality na module
- Lukáš - robustnejší mock server - znovupripojenie po strate pripojenia
- Marek K. - vytvorenie receptu a načítanie receptu pred začiatkom varenia
- Miška a Saška - práca na obrazovke pre výber / pridanie receptu
- Marek V. - práca na back-ende - ošetrovanie API

Vytvorenie nových taskov

Na stretnutí nevznikli nové tasky.

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Zápisnica - 15.11.2021

Zhrnutie taskov

Šprint sa nepodarilo úspešne dokončiť. Otvorené tasky sa presúvajú do ďalšieho šprintu.

Prezentácia vypracovaných taskov

- funkcionalitu na BE sa podarilo dokončiť (Marek V. a Marek K.)
- mock server funguje pre definované use cases (Lukáš)
- FE nie je dokončený (Miška a Saška)
- Peťo vytvoril usmernenie pre formát JSONov, ktoré sa posielajú medzi časťami systému

Vytvorenie nových stories/taskov

Stories

Pokračujeme v stories z predošlého šprintu.

Tasky

- Marek V. - vytvoriť možnosť pre spúšťanie testov pri pushovaní na GitHub
- Peťo - tento týždeň dokončiť a napojiť modul na pivovar
- Marek K.
  - dorobiť handlovanie chýb pri vkladaní do databázy + nastavovanie statusov
  - prerobiť atribúty v DB na CamelCase
- Saška a Miška
  - pokračovať v taskoch na FE
  - menu bar - Home, Výber receptu, História (možno aj nejaký status pivovaru)
  - shutdown tlačidlo a správa na BE
- Lukáš

- systémový test pre BE + debug
- funkcionalita pre korektné vypnutie RPi na BE (endpoint, skript)

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zápisnica - 22.11.2021

Na stretnutí sa najprv preberalo odovzdanie dokumentácie (Míľnik 1), potom členovia prezentovali stav taskov.

Zhrnutie taskov

- Marek K.
  - čo spravil:
    - hotfix dockeru pre BE, aby sa dal vybuildovať
    - premenovanie atribútov v DB na camel case
    - handlovanie promisov na BE (nedokončené)
  - čo spraví do konca šprintu:
    - logovanie (aspoň prvotná verzia)
    - dokončenie aktívnych taskov v Jire
- Saška
  - čo spravila:
    - rozdelenie formuláru na zadávanie ingrediencií na kategórie + CSS update
    - update vizuálu stránky pre výber receptu
  - čo spraví do konca šprintu:
    - hlavné menu pre navigáciu na front-ende
    - výber a úprava receptu ako single-page app. (sidebar zostane rovnaký, len sa v ňom budú prekreslovať údaje)
- Peťo
  - čo spravil:
    - spájkovanie modulu
    - objednanie konvertera pre modul
    - dokončené ESP s handlovaním teplomeru a motoru
    - testovacia aplikácia pre posielanie inštrukcií na moduly
- Marek V.
  - čo spravil:
    - testovanie
      - Jest na BE
      - ešte to potreba doladiť

    - čo spraví do konca šprintu:
        - web stránka (+ zistenie, čo všetko treba na web stránke)
- Miška
    - čo spravila:
        - dokončovanie dokumentácie
        - prvotný návrh vizuálu pre hlavné menu stránky
    - čo spraví do konca šprintu:
        - funkčné skladanie inštrukcií pre recept
- Lukáš
    - čo spravil:
        - shutdown RPi z BE - nie hotové
    - čo spraví do konca šprintu:
        - testovanie procesu varenia

Identifikácia nových problémov

- robiť viac **reviews** na hotové tasky
    - pokiaľ člen nemá dosť práce, môže sa venovať tomuto
- iniciatívnejšie sa zapájať do robenia taskov, pokiaľ sám nemám tasky (člen si sám môže niečo nájsť, nemusí čakať na stretnutie)

Vytvorenie nových stories/taskov

Na stretnutí neboli vytvorené nové tasky.

Diskusia

- bolo by dobré, keby sme do konca zimného semestra (resp. do začiatku letného semestra) mali už fungujúci prototyp, aby sme sa v ďalšom semestri mohli venovať TP Cupu
- dokumentácia pre Miľník 1 má obsahovať ako cieľ:
    - viem si vyklikať (vytvoriť recept)
    - viem spustiť varenie
    - viem si zobraziť priebeh varenia
    - postup varenia sa loguje do databázy

Šprinty

Názov šprintu

> **TODO**
>
> Pri vypĺňaní template je potreba prepísať hore v hlavičke atribút **title** do formátu: **XY - [Názov šprintu]**, kde XY je číslo šprintu *(napr. 01 - Zlatý Bažant)*.

Kedy?

Od: (dátum)

Do: (dátum)

Cieľ šprintu

Zhrnutie, čo je cieľom a malo by byť výsledkom šprintu.

Obsah šprintu

Stručne o stories, ktoré máme naplánované na šprint.

Story XY

Ja ako XY chcem XY, lebo ... (a podobné veci)

Výsledky šprintu

Čo sa nám v šprinte podarilo a čo sme nestihli.

Splnené úlohy

- *Úloha 1*
- *Úloha 2*

- ...

Nesplnené úlohy

- *Úloha 1*
- *Úloha 2*
- ...

Poznámky do budúcna

Niečo, čo si máme zapamätať alebo nejaké odkazy dobudúcna, čo sme sa naučili na tomto šprinte.

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Zlatý bažant

Kedy?

Od: 3.10.2021

Do: 18.10.2021

Cieľ šprintu

Úlohou prvého šprintu bol prvotný setup všetkých nástrojov akými sú Git, Jira či nástroj na tvorbu dokumentácie Docusaurus.

Ďalej bolo potrebné vytvoriť repozitáre pre jednotlivé časti projektu a v rámci nich vytvoriť jednoduché demo.

Obsah šprintu

Krátky opis stories, na ktorých sme pracovali počas šprintu. Treba dodať že nie všetci členovia figurujú ako zodpovední za nejakú story, avšak na práci sa podielal každý. Každá story totiž obsahovala viacero úloh, ktoré sme už do tohto zápisu neuviedli, lebo ich bolo veľmi veľa (stories + úloh bolo cez 40).

Technical prototype on RPI

Hlavný story. Prototyp pivovaru, ktorý sa automaticky bude nasadzovať na RPI a pri startupe sa naštartuje automaticky.

Frontend analysis

Základný dizajn frontendu spolu s technológiami. Viac na Frontend.

Analysis on how to brew a beer

Ako sa vôbec varí pivo, keďže väčšina členov tímu nevie aké sú tam kroky, a následne sa z tejto analýzy dá ľahšie zistiť, že čo treba spraviť. Viac na Brewery.

Basic tools

Základné nastroje ktorými sú: git, jira, dokumentácia.

Architecture overview

High level pohľad na časti systému. Viac na Architektúra.

Analyze existing SCADA software

Analýza iných softvérov, ktoré sú určné na podobné použite. Viac na Scada.

Team webpage

Základná tímová stránka.

Analysis of tools to use on HW

Aké rôzne komponenty sa dajú použiť na HW. Viac na Modbus.

Výsledky šprintu

Nástroj Jira nám ponúka nasledujúce vizualizácie hlavných stories a taskov.

Splnené úlohy

Key	Summary	Issue type	Epic	Status	Assignee
SB-3	Tool for creating documentation	Task	SPLNENIE PODMIE...	DONE	PS
SB-8	Analysis on how to brew a beer	Story	ZABEZPEČENIE AUT...	DONE	AF
SB-10	Basic tools	Story	SPLNENIE PODMIE...	DONE	PS
SB-12	Analyze existing SCADA software	Story	ZABEZPEČENIE AUT...	DONE	MN
SB-18	Architecture overview	Story	KOMUNIKÁCIA SO ...	DONE	MV
SB-24	Update documentation with new Record	Task	SPLNENIE PODMIE...	DONE	AF
SB-25	Create export from Jira	Task	SPLNENIE PODMIE...	DONE	PS
SB-28	Create templates for documentation	Task	SPLNENIE PODMIE...	DONE	MN
SB-37	Update Records from 11.10.2021	Task	SPLNENIE PODMIE...	DONE	MN
SB-38	Frontend analysis	Story	VIZUALIZOVANIE S...	DONE	MN
SB-41	Create temporary static HTML on frontend	Task	VIZUALIZOVANIE S...	DONE	MN

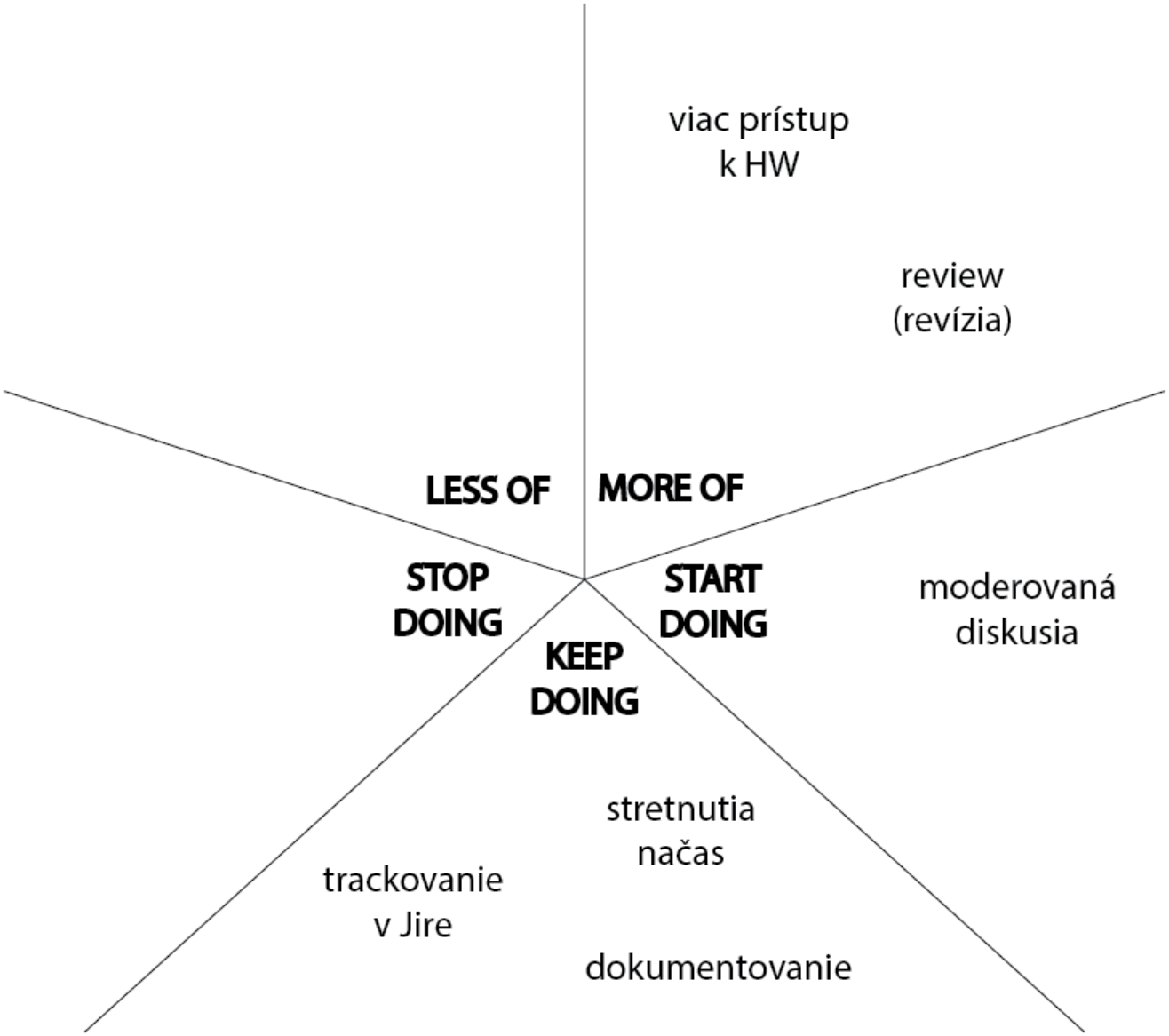
Nesplnené úlohy

Key	Summary	Issue type	Epic	Status	Assignee
SB-5	Technical prototype on RP	Story	KOMUNIKÁCIA SO ...	IN PROGRESS	MV
SB-13	Team webpage	Story	SPLNENIE PODMIE...	IN PROGRESS	MV
SB-29	Analysis of tools to use on HW	Story	ZABEZPEČENIE AUT...	IN PROGRESS	PS

*poznámka: *Technical prototype on RPI* sa podarilo takmer celé a hlavná funkcionalita bola, avšak nepodarilo sa nám stihnúť jednu úlohu, ktorú bolo treba odkomunikovať s ostatnými členmi tímu až po ukončení šprintu.

Poznámky do budúcna

Počas retrospektívy vznikol nasledujúci starfish diagram.



*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Pilsner Urquell

Kedy?

Od: 18.10.2021

Do: 2.11.2021

Cieľ šprintu

Hlavnou úlohou tohto šprintu bolo vedieť si zobraziť nejaké údaje zo senzorov.

Znamená to, že sme museli spraviť návrh grafického rozhrania. Návrh štruktúry API a komunikácie medzi časťami systému.

Obsah šprintu

Hlavným story bolo zobrazenie údajov na grafickom rozhraní. Hlavne sa to týkalo zobrazenia akutálnej teploty, ktorú nameral teplomer.

Show basic data on frontend from sensors

Hlavné story, pri ktorom bolo treba vytvoriť základné grafické rozhranie. Ďalej bolo treba umožniť získavanie najaktuálnejších dát z backendu. Takisto sme museli upraviť moduly, aby posielali správy o aktuálnom stave senzorov. Všetko toto zahrňovalo aj návrh komunikácie medzi jednotlivými komponentami spolu s jej implementáciou.

Create mock server, that acts like module for easier testing

Vytvorenie testovacieho modulu, ktorý si vie každý člen tímu spustiť u seba lokálne. Dôležité to je preto, lebo nie každý má prístup k HW a aby sme si mohli otestovať jednotlivé časti, musíme nejako simulovať modul.

Keep documentation updated

Story do ktorého sme dávali veci ako pridanie novej zápisnice, či prihlásenie sa do TP Cup.

[Zlatý bažant] Technical prototype on RP

Z predchádzajúceho šprintu sme tento story dokončili ihneď po stretnutí, keď sme si zadefinovali niečo, čo nám chýbalo pri poslednej pod-úlohe.

[Zlatý bažant] Analysis of tools to use on HW

Takisto z predchádzajúceho šprintu. Pričom bolo vykonaná analýza ohľadom zariadení na reguláciu teploty.

[Zlatý bažant] As a Team, we want to have a team webpage

Práca na tímovom webe bola minimálna. Jediné čo pribudlo, je odkaz na našu dokumentáciu priamo z webu v sekcií dokumenty. Farebné štýly boli upravené tak, aby dodržovali našu identitu.

Výsledky šprintu

Nástroj Jira nám ponúka nasledujúce vizualizácie hlavných stories a taskov. Podarilo sa nám spraviť 116.5/133 (87%) story pointov. Výsledný burndown graf nebudeme ukazovať, pretože nástroj Jira neponúka zobrazenie story pointov v sub-úlohách a výsledkom bolo to, že sme mali story s 90 bodmy, ktorý sme dokončili tesne pred koncom šprintu, čiže graf vyzeral akoby sme nič nerobili a zrazu väčšina zmizla.

Splnené úlohy

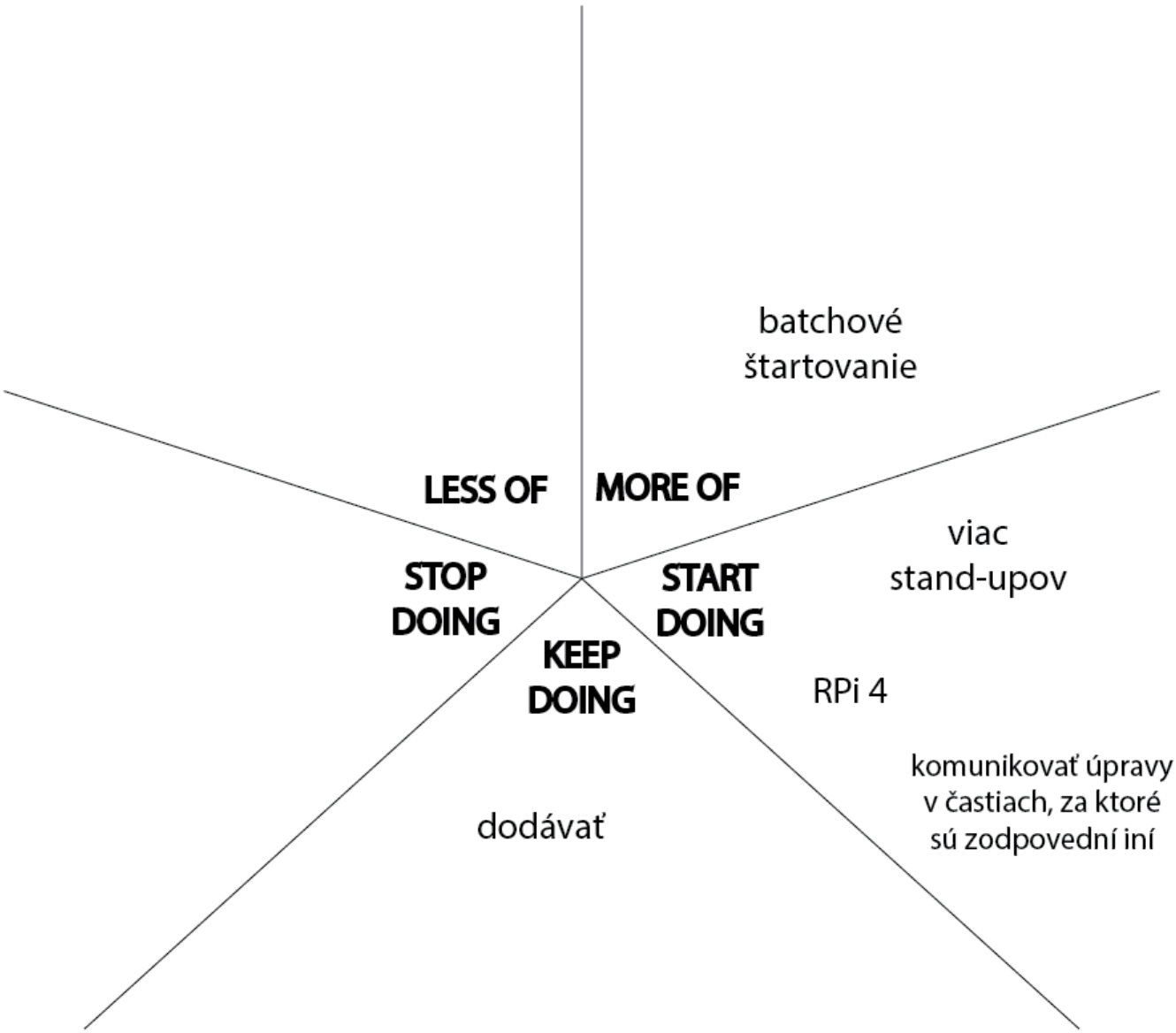
Key	Summary	Issue type	Epic	Status	Assignee
SB-42	Show basic data on frontend from sensors	Story	VIZUALIZOVANIE S...	DONE	PS
SB-44	Create mock server, that acts like module for easier testing	Story	KOMUNIKÁCIA SO ...	DONE	LN
SB-45	Keep documentation updated	Story	SPLNENIE PODMIE...	DONE	PS
SB-5	[From prev sprint] Technical prototype on RP	Story	KOMUNIKÁCIA SO ...	DONE	MV
SB-29	Analysis of tools to use on HW	Story	ZABEZPEČENIE AUT...	DONE	PS
SB-58	documentation - meeting guide	Task	SPLNENIE PODMIE...	DONE	MV
SB-69	TP-Cup registration	Task	SPLNENIE PODMIE...	DONE	AF

Nesplnené úlohy

Key	Summary	Issue type	Epic	Status	Assignee
SB-13	As a Team, we want to have team webpage	Story	SPLNENIE PODMIE...	IN PROGRESS	MV

Poznámky do budúcna

Počas retrospektívy vznikol nasledujúci starfish diagram.



*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Corgoň

Kedy?

Od: 2.11.2021

Do: 15.11.2021

Cieľ šprintu

Hlavným cieľom tohto šprintu bolo vedieť vytvoriť nový recept a následne tento recept použiť na varenie.

Obsah šprintu

Bolo treba premyslieť grafické rozhranie a urobiť ho viac prívetivím pre používateľa (na konci stránky sú aj obrázky). Ďalej bolo treba zjednodišiť testovanie jednotlivých častí systému - testovací server a spúšťací script.

Hlavnou častou bolo vytváranie receptov a následné použitie týchto receptov na spústenie varenia. Čiže hlavná logika a proces akým sa presúvajú inštrukcie zo stavu `vykonávania` do `ukončeného` stavu.

As a Developer, I want mock server to simulate instructions for brewing

Podpora vykonávania funkcií na testovacom servery.

As a Developer, I want easier startup of every app

Jednoduchý script ktorý spustí celý systém (viac info Spustenie pivovaru).

As a Student, I want to keep documentation updated

Klasická úloha vrámci ktorej sa vypisujú zápisnice či šprinty a podobné veci dokumentačného charakteru.

As a User, I want to create a new recipe

Vytváranie nového receptu. Zahrňuje návrh GUI, zmeny v databáze a aj úpravu backendu.

As a User, I want to use a simple recipe to test if brewing works

Aplikovanie receptu na varenie. Zahrňuje úpravu GUI, takisto aj logiku procesu varenia na backende a spracovanie inštrukcií na moduloch.

[Zlatý bažant] As a Team, we want to have a team webpage

Práca na tímovom webe bola menšia. Pribudlo zopár textov, ktoré sme použili vrámci našej prihlášky do TP Cup.

Výsledky šprintu

Nástroj Jira nám ponúka nasledujúce vizualizácie hlavných stories a taskov.

Podarilo sa spraviť hlavne backendovú časť celého šprintu, avšak na frontende vzniklo veľa problémov a preto sa tam nestihlo všetko dokončiť. Hlavná kostra tam už je, ale nieje to plne funkčné.

Vzhľadom na korona situáciu, sme nemohli veľa pracovať na úlohách a preto sa nestihli ani spraviť všetky zápisnice načas. Takisto sme si dali veľmi veľa úloh, ktoré boli veľmi náročné a aj preto sa nestihli dokončiť úplne.

Splnené úlohy

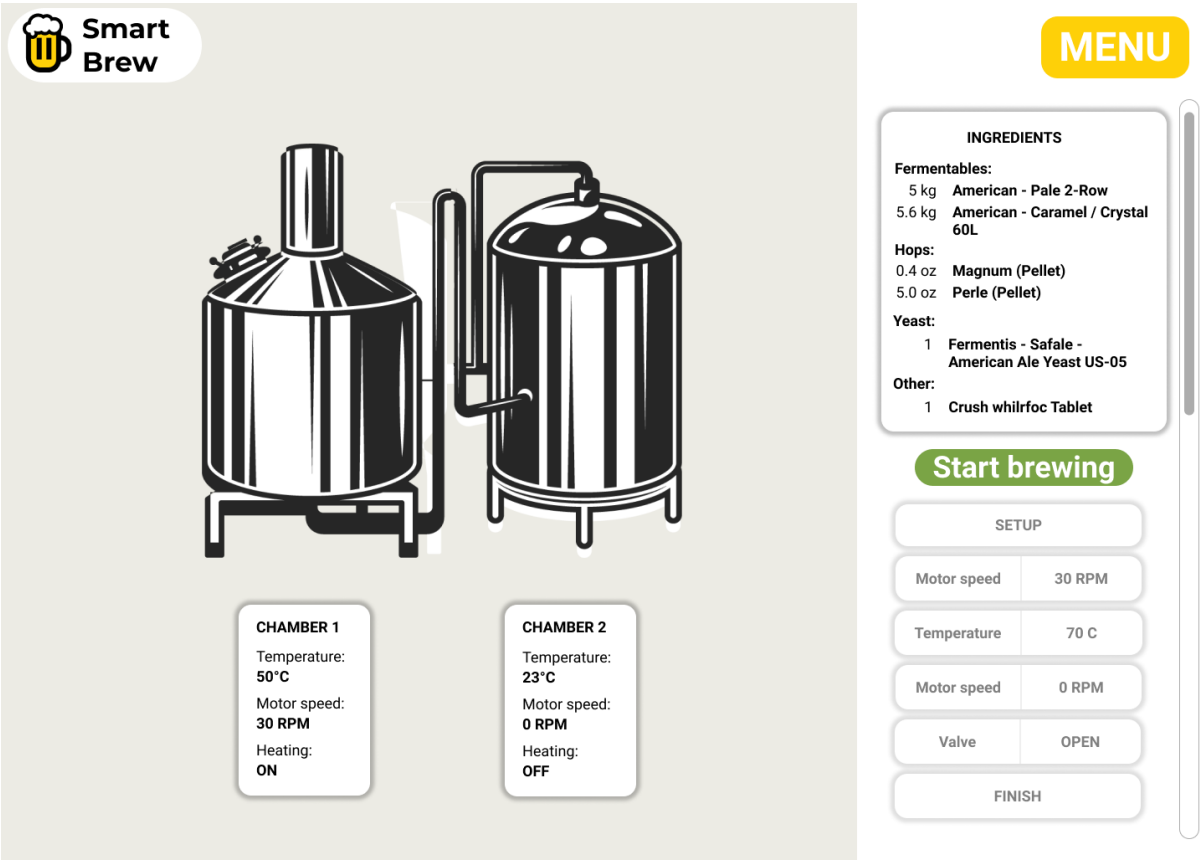
SB-75	As a Developer, I want mock server to simulate instructions for brewing	Story	KOMUNIKÁCIA SO ...	DONE	LN
SB-97	As a Developer, I want easier startup of every app	Story	SPLNENIE PODMIE...	DONE	PS

Nesplnené úlohy

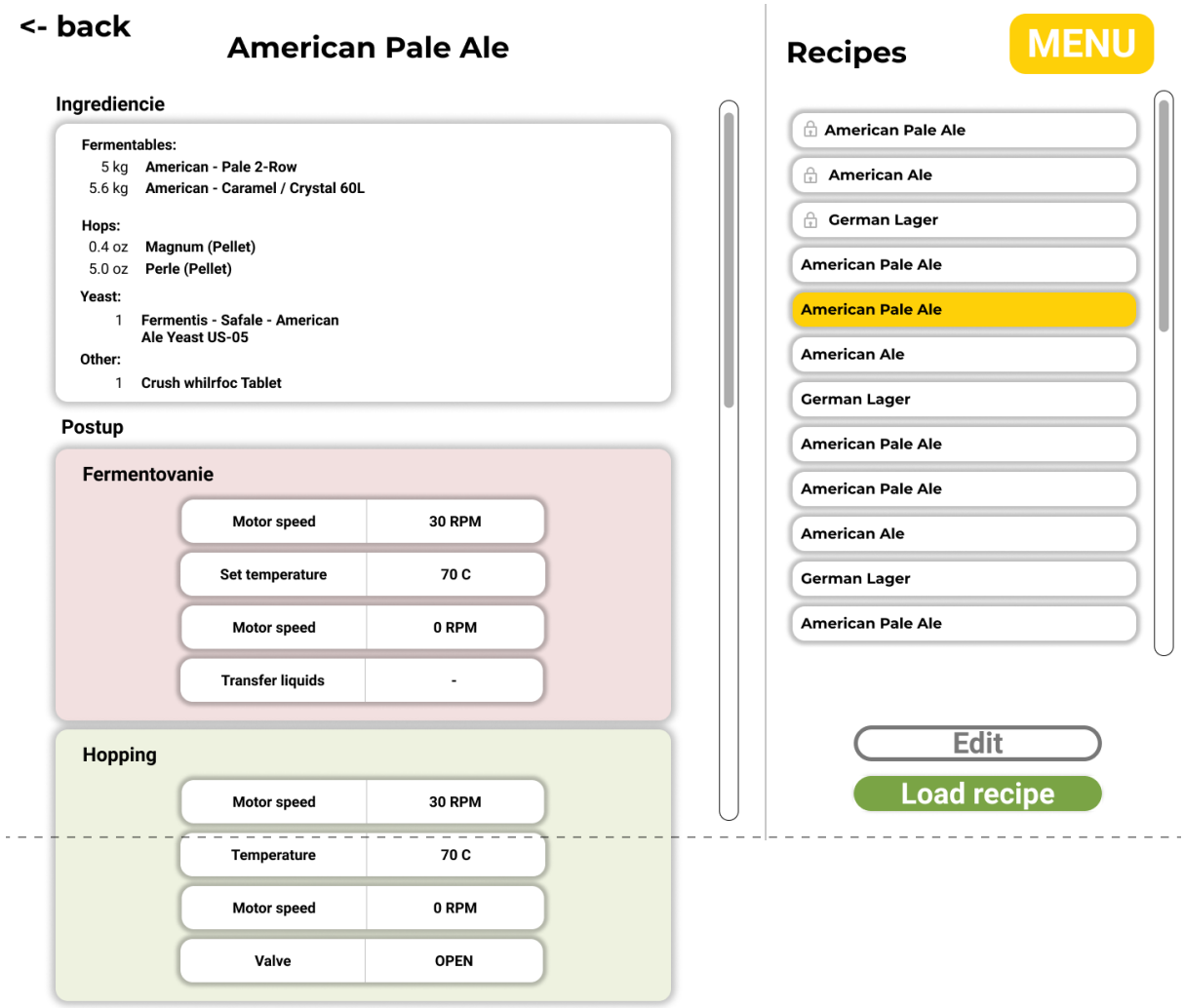
SB-78	As a Student, I want to keep documentation updated	Story	SPLNENIE PODMIE...	IN PROGRESS	PS
SB-82	As a User, I want to create a new recipe	Story	ZABEZPEČENIE AUT...	IN PROGRESS	MN
SB-83	As a User, I want to use a simple recipe to test if brewing works	Story	KOMUNIKÁCIA SO ...	IN PROGRESS	PS
SB-13	As a Team, we want to have a team webpage	Story	SPLNENIE PODMIE...	IN PROGRESS	MV

Návrh GUI

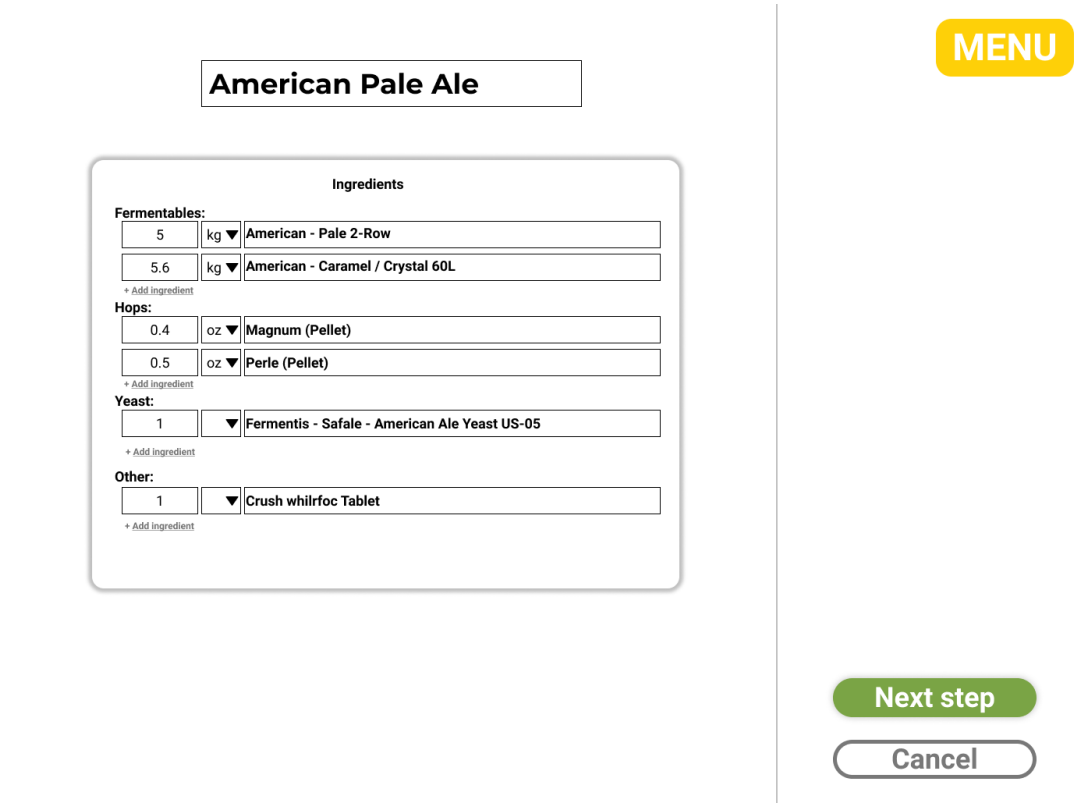
Prehľadová obrazovka



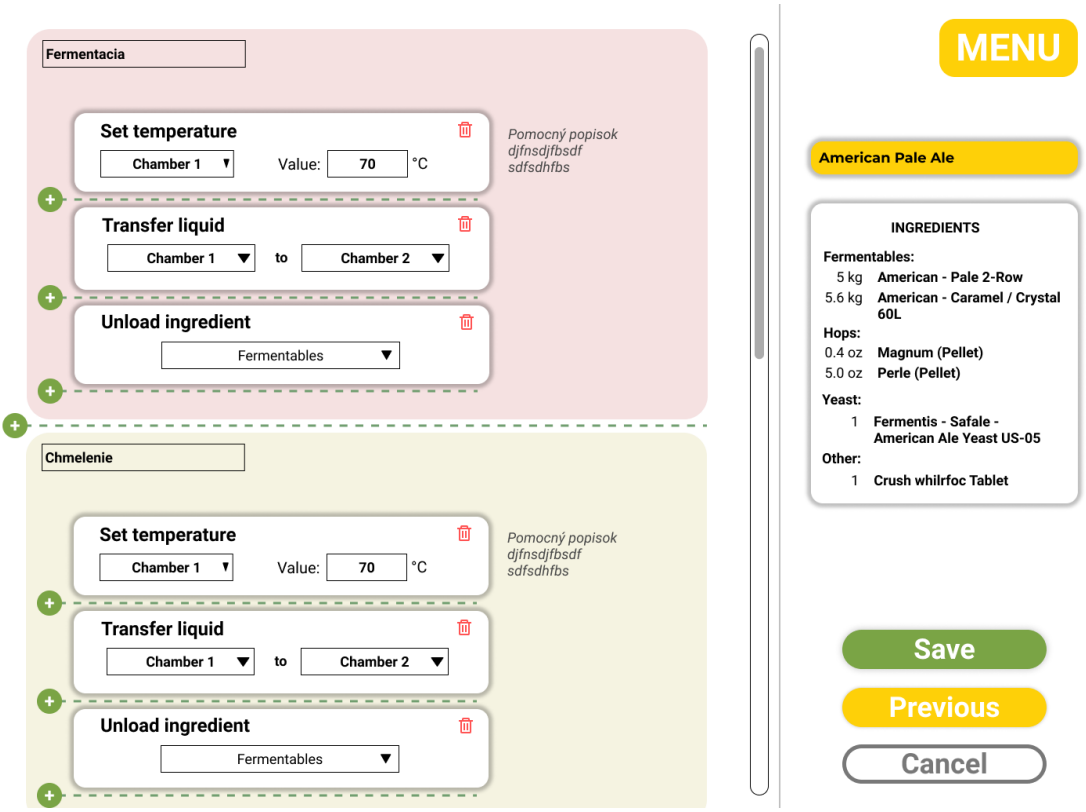
Výber receptu



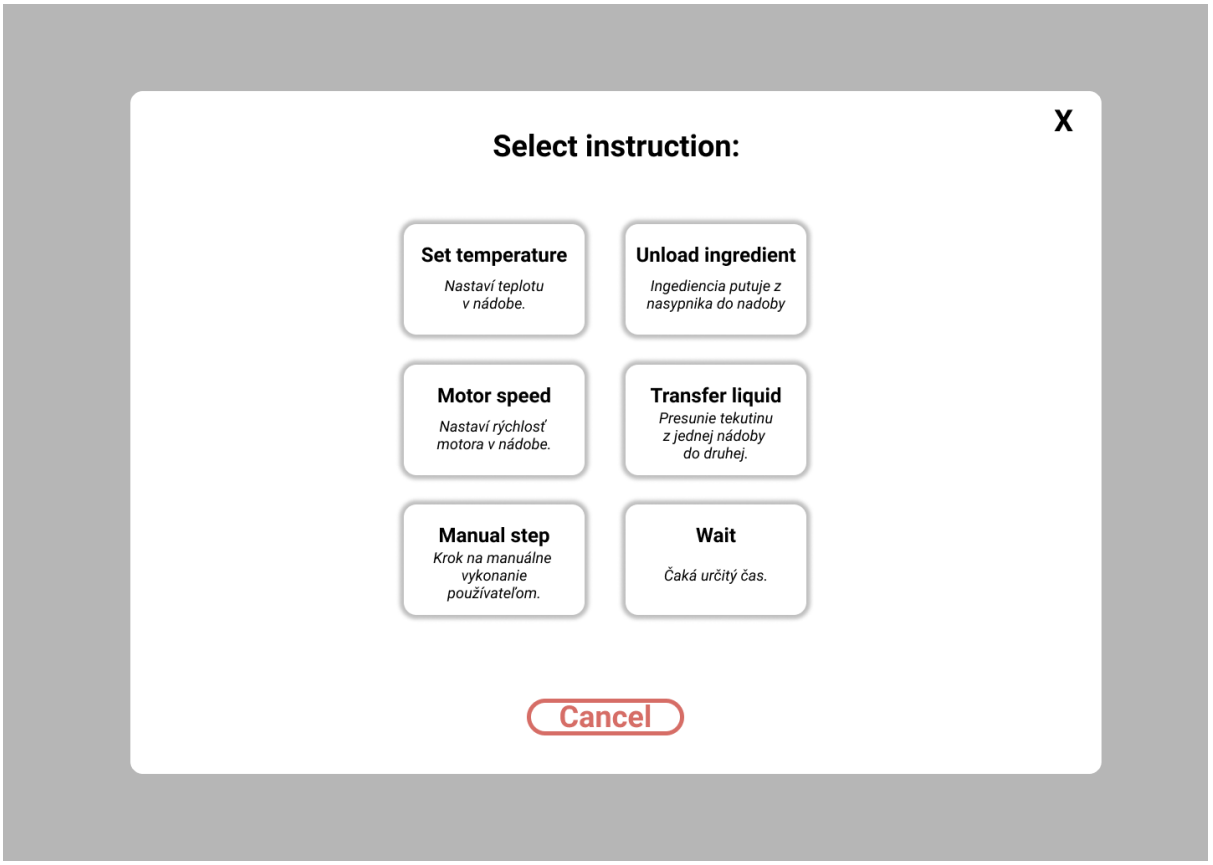
Vytváranie receptu (časť 1.)



Vytváranie receptu (časť 2.)



Vytváranie receptu (časť 2.) - detail



Krušovice

Kedy?

Od: 15.11.2021

Do: 29.11.2021

Cieľ šprintu

Dokončiť úlohy z Corgoň, ktoré sa týkali prototypu tvorby a spúšťania receptov.

Obsah šprintu

Stručne o stories, ktoré máme naplánované na šprint.

Story XY

Ja ako XY chcem XY, lebo ... (a podobné veci)

Výsledky šprintu

Čo sa nám v šprinte podarilo a čo sme nestihli.

Splnené úlohy

- *Úloha 1*
- *Úloha 2*
- ...

Nesplnené úlohy

- *Úloha 1*

- *Úloha 2*
- *...*

Poznámky do budúcna

Niečo, čo si máme zapamätať alebo nejaké odkazy dobudúcna, čo sme sa naučili na tomto šprinte.

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Metodiky

Komunikácia tímu

Kanály

Na komunikáciu sú určené viaceré kanály:

- Facebook Messenger
- Discord
- Google Meet

Messenger

Tím cez Messenger komunikuje vzniknuté problémy a situácie, ktoré treba **rýchlo vyriešiť**. Tiež si cez daný kanál dohaduje stretnutia.

Tento kanál je primárny na oznamovanie dôležitých informácií pre celý tím.

Discord

Tím má na Discorde vlastný server, cez ktorý uskutočňuje **stand-up hovory**. V textových kanáloch členovia posielajú dokumenty a správy, ktoré potrebujú byť uchované na dlhší čas (Discord vhodnejší než Messanger, keďže Messanger má neprehľadnú históriu).

Google Meet

Tím cez Google Meet uskutočňuje stretnutia s vedúcim tímu, ktoré sa vzhľadom na pandemickú situáciu uskutočňujú online. Pre lepšie sústredenie členov a celkovú atmosféru hovoru majú všetci zúčastnení zapnuté web kamery.

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Písanie dokumentácie pre riadenie projektu

Čo je potrebné zapisovať

Po každom stretnutí tímu s vedúcim vypracuje vybraný člen tímu **zápisnicu zo stretnutia**.

Po každom ukončení šprintu vypracuje scrum master **vyhodnotenie šprintu**.

Forma

Zápisnice/vyhodnotenia sú vypracované podľa predpripravených šablón:

- Šablóna zápisnice
- Šablóna šprintu

Usmernenia

Zápisnice

Je na zodpovednosti správcu dokumentácie, aby boli v dokumentácii aktuálne zápisnice a aby zápisnice boli zverejňované najneskôr do ďalšieho stretnutia (do týždňa).

Zápisnice by mali dodržiavať predpísaný formát, čo najviac sa to dá. Formát nie je striktný a nemusí sa dodržať, pokiaľ sa na stretnutí niektorá časť šablóny nepreberala.

Šprinty

Je na zodpovednosti scrum mastera, aby boli v dokumentácii aktuálne vyhodnotenia šprintov a aby vyhodnotenia šprintov boli vypracované najneskôr do týždňa od uzavretia šprintu.

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Metodika pre návrh front-endu

Za návrh front-endu sú zodpovední členovia implementujúci front-end (Alexandra Frniaková a Michaela Nemcová). K návrhu sa môžu pridať aj iní členovia tímu.

Nástroje

Pre návrh front-endu sa využíva nástroj Figma.

Usmernenia

Pokiaľ sa iný člen tímu ako zodpovední chce vyjadriť k návrhu, resp. niečo zmeniť, musí do toho zapojiť zodpovedných členov tímu.

Rozhodnutia ohľadom návrhu nikdy nerobí len jeden člen tímu, vždy to musí odsúhlasiť aspoň jeden, ale najlepšie aspoň dvaja iní členovia.

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

GitHub

Ako verzovací systém budeme používať GitHub po názvom organizácie smart-brew.

V rámci tejto GitHub organizácie budeme pracovať v nasledovných repozitároch:

- documentation - pre dokumentáciu
- web - pre tímovú stránku
- frontend - GUI pre používateľov
- backend - aplikačný server, ktorý je spojený s databázou a rieši požiadavky z frontendu cez REST API
- websocket-module - ESP32 modul (viac info Moduly)
- module-mock-server - testovací server, ktorý slúži ako náhrada modulu pre lokálny development (viac info Testovací server)
- startup - spustenie systému (viac info Spustenie systému)

Commit

Každý commit bude dodržovať nasledujúci formát

Formát commitov

```
typ(jira): popis
```

- `typ` je jedným z: `feature`, `fix`, `chore`
- `jira` je označenie Jira issue (napr. `SB-1`) - dôležité dodržať názov aby sa dalo linkovať automaticky medzi Jirou
- `popis` je krátky popis úkonu

Ukážka commitu

```
+ feature(SB-2): Login page
+ fix(SB-34): Missing files from upload page
```

Tvorba vetiev

Každá nová fičúra sa bude tvoriť na osobitnej vetve, pričom sa neskôr bude mergovať pomocou Pull request do main vetvy.

Formát vetiev

```
typ(jira)/popis
```

- `typ` je jedným z: `feature`, `bug`, `hotfix`
- `jira` je označenie Jira issue (napr. `SB-1`)
- `popis` je krátky popis úkonu - všetko **malé písmená**, pričom slová sú **oddelené pomlčkou**

Ukážka vetvy

```
+ Správne
+ feature(SB-2)/login-page
+ bug(SB-34)/missing-files-from-upload-page


- Nesprávne
- feature(SB-2)/Login page                      # nesprávny formát popisu
- bug(34)/missing-files-from-upload-page        # nesprávne označenie Jira issue
- bug(SB-34):missing-files-from-upload-page     # ':' namiesto '/'
```

Pull request

Názov pre Pull request musí taktiež dodržať istý formát ako pre názvy commitov.

Formát pull request

```
typ(jira)/popis
```

1. Ak sa v Github actions nachádza script, ktorý vykonáva automatický test, musí najprv **úspešne** prejsť, aby sa mohli zmeny merge-núť
2. Ďalej je potrebné spraviť code review a zapísať tento fakt do Jiry

3. Až následne vykonať merge
4. Uzavrieť issue v Jire

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Jira

Pre prácu na projekte a trackovanie úloh využívame Jiru.

S Jirou pracujeme metodológiou Scrumu.

Úlohy

- `Epic`
  - Epic zahŕňa stories a tasky pre časť funkcionality projektu (moduly, back-end, front-end, organizácia).
  - Spravidla sa na ňom podieľajú viacerí členovia tímu.
- `Story`
  - Story je stavané z pohľadu "Ako používateľ chcem ...", napr. "*Ako používateľ chcem vytvoriť nový recept.*"
  - Väčšinou sa na ňom podieľajú viacerí členovia tímu.
- `Task`
  - Task ponúka najnižšiu granularitu pre story. Ide už o konkrétne implementačné úlohy.
  - Pracuje na ňom vždy jeden člen tímu.

Hodnotenie

Tím bodovo hodnotí zložitosť jednotlivých **taskov**. Tieto hodnotenia sa potom spočítajú a výsledok je použitý ako body pre dané story, ktorému tasky patria.

Prechody

`TO DO` --> `IN PROGRESS` --> `WAITING FOR REVIEW` --> `DONE`

Pokiaľ člen tímu začne vykonávať pridelený task, presunie ho do `IN PROGRESS`.

Po dokončení tasku ho presunie do `WAITING FOR REVIEW` s tým, že mu ho musí iný (aspoň jeden) člen tímu odsúhlasiť.

Definition of DONE

Pre každý šprint je nastavený **Definition of DONE**. Ide o konkrétne story alebo viacero stories, ktoré keď sa podarí tímu splniť, šprint môže byť považovaný za úspešne splnený.

Zodpovednosti

S Jirou pracuje najmä scrum master:

- vytvára epics, stories a tasks
- dohliada na plnenie taskov a ich aktualizáciu v Jire
- uzatvára šprint po ukončení

Tasky môžu vytvárať aj iní členovia tímu po dohode so zvyškom tímu a vyhodnotení, že daný task je naozaj potrebný.

*Last updated on **19. 11. 2021** by **Michaela Nemcova***

Metodika stretnutí

Tu je zapísaná základná organizácia stretnutí tímu, ktorej sa budeme snažiť držať.

Kedy?

Pondelok o 14:00 - konzultácie s vedúcim

Štvrtok o 20:30 (alebo inokedy po dohode) - krátky stand-up

Časti stretnutia a práce v tíme

Štruktúra ako by malo vyžerať tímové stretnutie.

Časť 1 - prezentácie

Každý člen v krátkosti oboznámi svojich kolegov na čom pracuje, aké urobil rozhodnutia a prečo ich urobil. Zároveň môže poukázať na miesto kde sa zasekol alebo potrebuje dorobiť niečo od iného člena tímu. Prezentujeme ideálne na projektore alebo zdieľame obrazovku.

Časť 2 - voľná konzultácia

Tu sa riešia problémy, vymýšľajú sa nové veci atď.

Časť 3 - problémy na úvahu

Táto časť prebieha neustále. Vždy keď nejaký člen chce aby sa všetci zamysleli nad spôsobom riešenia dákeho problému, povie to buď na stretnutí alebo to napíše na discord do chanelu `problémy`. Snažíme sa zapisovať všetko aj na discord.

Pravidlá

Pravidlo 1 - V prípade, že chceme zasiahnuť do rozbehnutej diskusie, hlásime sa aby sme sa neprekrikovali počas stretnutí. Diskutujúca dvojica sa vždy posnaží dať hlásiacemu sa členovi priestor čo najskôr.

Pravidlo 2 - Vždy sa snažiť predstaviť svoje riešenie problému ako návrh, ktorý sa môže upraviť v prípade, že to spadá pod špecializáciu, ktorú člen tímu vykonáva. Teda ak mám nastarosti databázy mám automaticky väčšie práva vykonávať rozhodnutia ohľadom databáz ak som zvážil, že sa nejedná o naozaj kritickú časť projektu. Snažíme sa byť čo najviac samostatní a prezentovať našu prácu kolegom, ktorých pripomienky musí zobrať prezentujúci člen tímu do úvahy. Preto je informačné stretnutie aspoň davakrát do týždňa veľmi dôležité.

Iné usmernenia

Vzhľadom na pandemickú situáciu a dištančné vzdelávanie sa od 8.11. do odvolania uskutočňujú stretnutia online pomocou kanálov Google Meet / Discord (viď Metodika komunikácie).

*Last updated on **19. 11. 2021** by **Michaela Nemcová***

Motivačný dokument

Motivačný dokument

Tím 6

Alexandra Frniaková
Marek Krchňavý
Michaela Nemcová
Lukáš Novota
Peter Stríž
Marek Vajda

Kontakt:
tim6_tp2021@googlegroups.com

Náš tím

Sme skupina študentov so skúsenosťami aj mimo školy, či už z práce pre rôzne IT firmy (Innovatrics, MicroStep-MIS, TatraMed Software), alebo z rôznych kurzov. Na tímovom projekte je našou prioritou vytvoriť produkt, ktorý bude kvalitný a dobre použiteľný v praxi, a pritom sa niečo nové naučiť. Vítame výzvy a nie je problém, ktorý by sme spoločnými silami nevedeli prekonať.

Tímu nechýba organizovanosť, schopnosť komunikovať a prezentovať svoje nápady a znalosť zostavovania zrozumiteľnej dokumentácie. K povinnostiam pristupujeme svedomito a zakladáme si na dobrej tímovej spolupráci.

V tímovom “repertoári” máme široké spektrum technológií, ktoré členovia tímu ovládajú. Hlavné zameranie tímu sú hlavne webové aplikácie a umelá inteligencia. Konkrétne pre webové aplikácie sú to technológie:

- JavaScript / TypeScript, PHP
- React, Vue.js, Angular, Laravel
- Node.js (Express), Django
- HTML5 a CSS3

Všetci členovia majú tiež skúsenosti s relačnými databázami (PostgreSQL a MySQL) a s technológiou Docker.

Z oblasti umelej inteligencie majú viacerí členovia skúsenosti zo svojich bakalárskych prác, konkrétne s oblasťami NLP (Alexandra Frniaková, Lukáš Novota, Marek Krchňavý), spracovanie obrazu (Michaela Nemcová a Peter Stríž) v jazyku Python - knižnice scikit-learn, TensorFlow a PyTorch. Dvaja kolegovia, Peter Stríž a Marek Vajda, majú skúsenosti aj s platformou Raspberry Pi či Arduino a majú absolvovaných viacero predmetov orientovaných na siete a takisto aj predmet Mikropočítače.

Našou prioritou pri výbere boli témy ponúkajúce možnosť práce s technológiami, s ktorými by sme si radi rozšírili znalosti. Okrem toho by sme sa radi zlepšili v tímovej spolupráci a komunikácií. Chceli by sme, aby výsledkom našej práce bol produkt, na ktorý môžeme byť hrdí.

Motivácia

Téma č. 3

DataHub pre rôzne typy zariadení, ich spracovanie / analýzu / vizualizáciu

Táto téma nás všetkých zaujala najmä svojou komplexnosťou, rozsahom a uplatnením v praxi. Vďaka toľkým oblastiam a technológiám by sme dokázali využiť skill-set nášho tímu naplno a každý by si vedel nájsť to svoje. Aj preto sme sa rozhodli pre túto tému ako pre naše číslo jeden.

Keďže projekt má slúžiť existujúcej komunite ľudí, vidíme v ňom aj možnosť získať skúsenosti pri navrhovaní dobrého používateľského zážitku a používateľského rozhrania, a taktiež ako výhodu považujeme možnosť získavania pravidelného feedbacku. Toto by pre nás bolo aj hlavnou motiváciou k svedomitej a kvalitnej práci, ktorej výsledkom by bola kvalitná a intuitívna aplikácia.

Naši členovia tímu majú značné skúsenosti so spracovaním veľkého množstva dát a ich vizualizáciou vďaka ich bakalárskym prácam. Vďaka týmto skúsenostiam si uvedomujeme, že aby mali koncoví používatelia z aplikácie úžitok, musia byť dáta vhodne a prehľadne prezentované, čiže sa budeme snažiť o čo najlepší používateľský zážitok.

Naším cieľom v tejto téme je vytvoriť nie len dobre fungujúcu, ale aj prehľadnú a intuitívnu aplikáciu. Motiváciou k tomu je možnosť komunikácie s koncovými používateľmi a veľký rozsah obsiahnutých oblastí a technológií, čo vnímame ako výbornú príležitosť naučiť sa niečo nové a rozvíjať sa v odbore ďalej.

Téma č. 8

Educational Content Engineering Hub - Databáza otázok, odpovedí, úloh a riešení [ECEH-DU]

Táto téma nás zaujala najmä tým, že ide o nástroj, ktorý by sme všetci uvítali počas štúdia, a tiež by sme tým mohli pomôcť mnohým terajším a budúcim študentom. Pri našom štúdiu nám veľmi pomáhal podobný nástroj - Stack Overflow. Avšak na tejto platforme nie je dosť zreteľne rozlíšené, kto je expert a kto nie. Preto by bola takáto špecifická platforma, ako ECEH-DU, ktorá sa viaže len na nejakú konkrétnu oblasť (v tomto prípade na fakultu), oveľa väčším prínosom pre študentov.

Na téme nás zaujalo aj to, že ide už o existujúci projekt, pre ktorý by sme robili nadstavbu. Teda by sme mohli získať ďalšie skúsenosti pri práci s cudzím kódom, čo sa mimoriadne hodí v pracovnej sfére.

Naším plusom v súvislosti s touto témou je znalosť rozsiahleho spektra technológií pre vývoj webových aplikácií a skúsenosti aj mimo školy. Členovia ovládajú viacero frameworkov pre front-end aj back-end, a tiež aj relačné databázové technológie a nástroje Docker a Git. Aj keď nie všetci poznáme požadovanú technológiu PHP, sme otvorení novým veciam a radi sa ju naučíme.

Vďaka tomu, že my sami by sme radi takýto nástroj používali a chceli by sme, aby bol kvalitný, nás to bude hnať k čo najlepším výsledkom a dokážeme vytvoriť produkt, ktorý bude pomáhať mnohým študentom a nebudú si musieť prechádzať takými problémami, ako sme si prechádzali my.

Téma č. 14

IoT platforma na priemyselnú automatizáciu - malý pivovar

Prácu na tomto projekte vidíme ako veľkú príležitosť využiť naše existujúce vedomosti, a ďalej ich rozširovať. Veľmi nás zaujala práca na rôznych úrovniach, od komunikácie so senzormi, ukladania údajov, až po zobrazovanie stavov na front-ende.

Za dôležité plus taktiež považujeme možnosť práce so zariadením, ktorého dáta využívame, a jeho nastavovanie. Lepšie si tak uvedomujeme dôležitosť týchto dát a ich formátu. Na projekte nás tiež zaujalo, že nejde o riešenie pre nejaký špecifický problém, ale musí byť dostatočne všeobecné na to, aby sa dalo využiť aj v iných sférach.

Veríme, že vedomosti, ktoré sme nadobudli počas štúdia a ostatných aktivít, tvoria pevné základy, na ktorých pri práci na tomto projekte môžeme stavať. Vďaka rozsiahlosti témy si každý z tímu dokáže nájsť to, čo by ho bavilo a s čím už má skúsenosti.

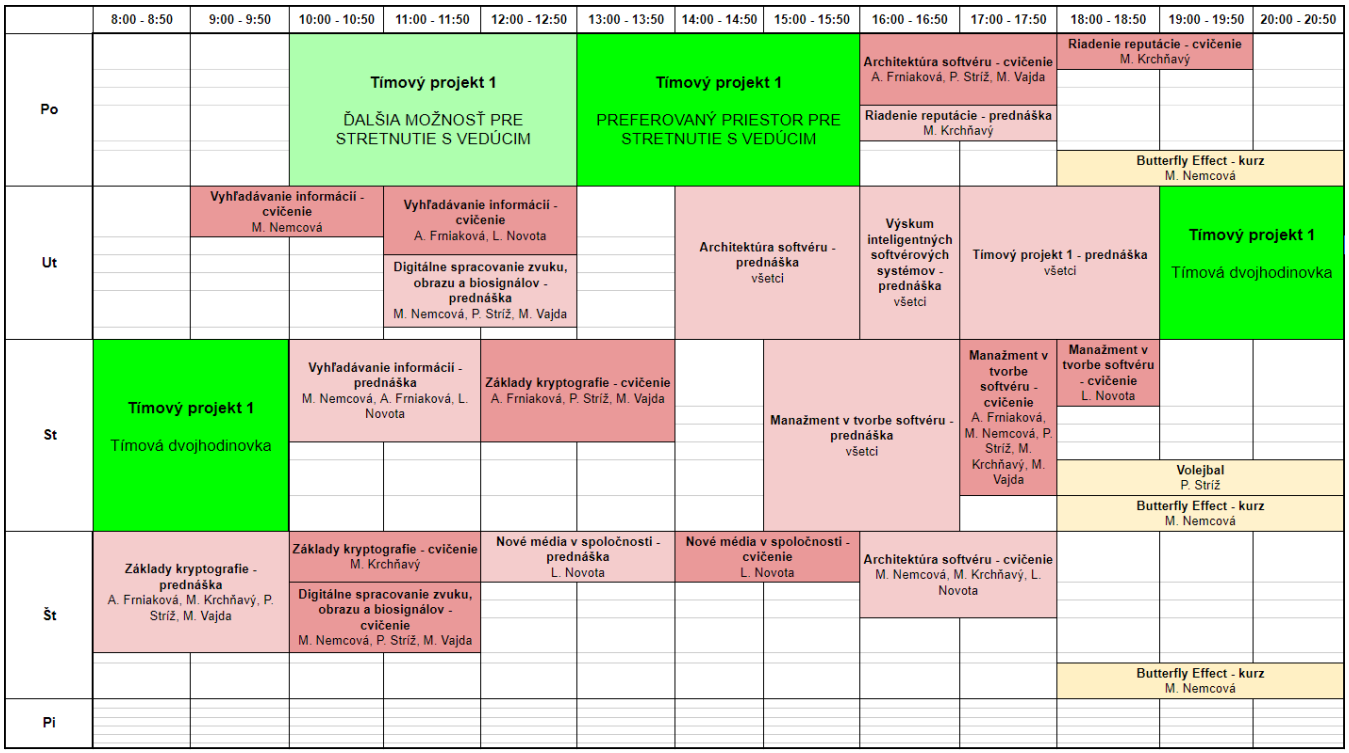
Naším cieľom je poskytnúť kvalitné riešenie, ktoré bude bezpečne spĺňať všetky požiadavky kladené obsluhou zariadenia. Keďže ide o automatizáciu, je pravdepodobné, že s podobnými projektami sa budeme stretávať stále častejšie v kariérnom prostredí. Tento projekt by nám umožnil nadobudnúť potrebné skúsenosti a naučiť sa rôzne nové veci, čo nás motivuje ku kvalitnej práci.

Príloha A

Témy sú zoradené od najvyššej priority po najnižšiu.

Poradie	Číslo témy	Názov témy
1	3	DataHub pre rôzne typy zariadení, ich spracovanie / analýzu / vizualizáciu
2	8	Educational Content Engineering Hub - Databáza otázok, odpovedí, úloh a riešení [ECEH-DU]
3	14	IoT platforma na priemyselnú automatizáciu - malý pivovar
4	12	Spektrometrické rozpoznávanie túh do pera
5	15	Ion Mobility Spectrometry for Rapid HEMP Potency Testing
6	19	Automatizácia procesov KYC (Know your client) a AML (Anti-money laundering)
7	13	Navigácia v smartfóne pomocou rozšírenej reality
8	9	Monitorovanie a správa systému pre výrobný areál [LOMON]
9	11	(Q)SAR analýza fototoxických látok
10	5	Vytvorenie inteligentného model-based agenta (umelá inteligencia na báze Knowledge grafov) pre tvorbu komplexných dátových štruktúr a vzťahov pre aplikovaný výskum v klinickej onkológii
11	4	Adverse Media Screening
12	16	FIIT WIX

Príloha B



Ružovou sú označené prednášky a cvičenia. Žltou sú vyznačené osobné povinnosti. Zelenou sú označené voľné časy na stretnutie.

Preferované časy na stretnutia

Stretnutie s vedúcim:

- Pondelok 13:00 - 16:00 (preferované)
- inokedy v Pondelok v rozmedzí od 9:00 do 16:00 (ďalšia možnosť)

Tímové stretnutia:

- Utorok 19:00 - 21:00 (2 hodiny)
- Streda 8:00 - 10:00 (2 hodiny)

Inžinierske dielo

Úvod

Tento dokument obsahuje informácie k produktu. Poskytuje základné delenie na časti **Architektúra, Analýzy a Návody.**

Architektúra

- **Api**: Spôsoby komunikácie (requesty a odpovede) medzi komponentami jednotlivých vrstiev.
- **Návrh databázy**: Štruktúra databázy vyjadrená diagramom.
- **Front-end**: Návrh používateľského prostredia a jeho opis
- **Moduly**: Opis fungovania modulu
- **Podporované údaje**: Zadefinované interfaces. Údaje z akých zariadení sa využívajú.
- **Podporované funkcie**: Zoznam podporovaných funkcií systémom a štruktúry inštrukcii.
- **Testovací server**: Opis funkcionality.

Analýzy

- **Brewery, modbus, SCADA**: analýza nástrojov využitých pri funkčnom pivovare daného typu

Návody

- **Migrácie, Raspberry Pi, Spustenie pivovaru, Tímová VM, Tímový web + dokumentácia**: Návody k správnemu využívaniu daných nástrojov.

Ciele

Cieľom projektu je vytvoriť ucelené riešenie na automatizovanie procesov v priemysle. Projekt je vykonávaný v spolupráci so Strojníckou fakultou STU, ktorá z veľkej časti zastrešuje materiálne požiadavky a fyzické prevedenie pivovaru. Pivovar využíva dve prepojené nádoby, pričom v každej prebieha iná fáza varenia. Naša práca bude prebiehať nad údajmi z pivovaru, konkrétne nad rôznymi efektormi (motory, násypníky, ventily,...) a senzormi (teplomer, tlakomer,...). Centrálnym komponentom systému je Raspberry Pi, ktoré bude poskytovať zázemie pre sieť senzorov a efektorov. Automatizovaná výroba piva pozostáva z pridania nového receptu (respektíve výberu už existujúceho receptu), procesu výroby piva, priebežného zobrazovania priebehu varenia a vytvorenia zápisu o ukončenom procese.

Komunikácia bude prebiehať medzi niekoľkými vrstvami. Moduly periodicky odovzdávajú svoje údaje back-endu, ktorý ich spracuje (niektoré si ukladá v dočasnej pamäti a niektoré ukladá do databázy). Taktiež si

periodicky pýtajú úlohy, ktoré majú vykonávať. Rovnakú úlohu vykonáva aj front-end. Pri procese varenia si od back-endu periodicky pýta stav varenia, ktorý zobrazuje používateľovi. Pri tvorení nového receptu zase posiela údaje na back-end, kde sa zapisujú do databázy.

Komunikácia medzi front-endom a back-endom sa bude riešiť pomocou REST API. Na komunikáciu sa bude využívať formát JSON. Štruktúra JSON-u sa bude meniť podľa vrstvy komunikácie a obsahu správy.

Kroky (inštrukcie) pre jednotlivé moduly sa budú posielať a spracovávať sériovo. Teda inštrukcie na moduly sa budú zasielať v osobitných správach. Frontend si ale tieto kroky bude triediť a spájať do blokov podľa určitých etáp výroby pre lepší user experience.

Všetky potrebné údaje o receptoch a jednotlivých vareniach sa budú ukladať do databázy. To umožní znovupoužitie už vytvoreného receptu a vygenerovanie protokolu z varenia.

Ciele na tento semester

Pripraviť funkčný prototyp tak, aby vykonával základné úlohy. V užívateľskom prostredí si používateľ bude vedieť vytvoriť nový recept. Taktiež bude vedieť začať proces varenia, a na obrazovke zobraziť jeho priebeh. Backend spracuje dané údaje, ktoré získa z modulov, a pošle ich na frontend - používateľ bude môcť kontrolovať stav varenia. Stav varenia sa tiež bude logovať do databázy (čo sa neskôr môže použiť na vytváranie protokolu o varení). Bude funkčná aspoň určitá podmnožina modulov (nakoľko nie sú niektoré zariadenia dostupné, nedá sa s nimi zatiaľ pracovať). Front-end a back-end bude vedieť spracovávať všetky procesy, funkcionalitu zatiaľ nefunčných modulov budeme simulovať.

*Last updated on **22. 11. 2021** by **Alexandra Frniakova***

Návody

Migrácie

V projekte využívame ORM Prisma. Pre vytvorenie migrácie treba upraviť súbor `schema.prisma` podľa potreby.

Potom vytvoriť novú migráciu, ktorá sa automaticky aplikuje na databázu na adrese `localhost:5432`.

Ak nevykonáš zmeny tak sa nevytvorí nová migrácia iba sa aplikujú čakajúce migrácie.

Vytvorenie novej migrácie alebo aplikovanie migrácií

```
yarn migrate
```

Pri použití príkazu `yarn migrate` sa databáza naseeduje. Ak však už bola raz naseedovaná pri opakovaní príkazu sa u neseeduje. Preto ak sa napríklad doplní do súboru `seed.js` ďalší záznam, je potrebné seedovanie spraviť samostatne.

Seedovanie databázy

```
yarn seed
```

Záznamy, ktoré už v databáze sú sa opakovane nepridávajú.

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Raspberry pi

Pripojenie do rpi

> ⚠ **VHODNÝ TERMINÁL A REŠTART**
>
> Odporúčam použiť `git bash` alebo `wsl` terminál na prístup do rpi. Taktiež by malo byť spojenie odolné voči reštartom. To znamená, že reštart virtuálneho stroja alebo rpi by nemal nič znefunkčniť a po čase cca `1 min` by malo byť pripojenie znova dostupné.

1. krok

Ak nemáš vygenerované ssh kľúče treba ich vygenerovať bez hesla (treba len stále štláčať enter).

Vygenerovanie ssh kľúčov

```
ssh-keygen
```

2. krok

> ⚠ **PODMIENKA POUŽITIA PRÍKAZU**
>
> Ak budeme mať heslo do virtuálneho stroja `heslo="neznáme"` môžes použiť príkaz na skopírovanie tvojho verejného kľuča do `authorized_keys`.

Ak je heslo neznáme pošli nejakému členovi napr. (Marek Vajda alebo Peter Stríž) svoj verejný kľúč aby ti ho nahrali na virtuálnom stroji do `authorized_keys`. V prípade, že skôr spomenuté možnosti nie sú dostupné, je potrebné sa prihlásiť pomocou privátneho kľúča do virtuálneho stroja, ktorý zadáš do prikazu `ssh -i [cesta ku privátnemu kľúču]> user@ip`. Následne si môžeš nahrať svoj kľúč samostatne.

Skopírovanie verejného kľúča do virtuálneho stroja

```
ssh-copy-id ubuntu@team06-21.studenti.fiit.stuba.sk
```

3. krok

Pripojenie príkazom ssh, ktorý používa tzv. `jumphost` na pripojenie sa do reverzného terminálu rpi. Port je `2222` alebo `2224`.

pripojenie do rpi cez ssh jumphost

```
ssh -J ubuntu@team06-21.studenti.fiit.stuba.sk:22 -p 2222 pi@localhost
```

> **ALTERNATÍVN+E PRIPOJENIE**
>
> Alternatívne je možné sa pripojiť najprv cez ssh do virtuálneho stroja. A následne do reverzného prikazového riadka na porte `2222` alebo `2224`.
>
> **alternatívne príkazy**
>
> ```
> ssh -p 22 ubuntu@team06-21.studenti.fiit.stuba.sk
> ssh -p 2222 pi@localhost
> ```

Pripojenie pomocou vnc

Na pripojenie pomocou vnc si treba stiahnuť vnc viewer. A následne sa prihlásiť do účtu. Rpi sa následne zobrazí v účte tímu.

prihlasovací email

```
marek.vajda009@gmail.com
```

password

```
tp062122
```

Spustenie browsera do fullscreenu na Rpi

Rpi je nastavené tak aby sa po spustení otvoril webový frontend v prehliadaci. Momentáne sa v grafickom rozhraní spustí len webový prehliadač `chromium` bez ostatného grafického rozhrania. Toto nastavenie sme docielili v súbore `/home/pi/.config/lxsession/LXDE-pi/autostart`. Ak chceme aby sa otvorilo po štarte aj štandardné grafické rozhranie musíme nasledujúci príkaz zadať do súboru `/etc/xdg/lxsession/LXDE-pi/autostart`. Alebo ak nefunguje treba použiť `sudo nano /etc/xdg/lxsession/LXDE-pi/autostart` súbor.

príkaz na spustenie prehliadaca vo fullscreen

```
@chromium-browser localhost --start-fullscreen --kiosk
```

Ako funguje pripojenie

Pripojenie funguje pomocou reverzného ssh spojenia. Rpi v intervale `1 min` spúšťa pomocou `sudo crontab -e` skript, ktorý kontroluje vytvorenie reverzného ssh spojenia na virtuálnom stroji.

sudo crontab -e príkaz

```
*/1 * * * * sudo . /create_ssh_tunnel.sh > tunnel.log 2>&1
```

Skript na vytvorenie reverzného konzolového spojenia sa pripája na virtuálny stroj `team06-21.studenti.fiit.stuba.sk`. Port `2224` na virtuálnom je následne tunelovaný ako port `22` na raspberry pi.

create_ssh_tunnel

```
#!/bin/bash
createTunnel() {
  /usr/bin/ssh -N -f -R 2222:localhost:22 ubuntu@team06-21.studenti.fiit.stuba.sk
  if [[ $? -eq 0 ]]; then
    echo Tunnel to jumpbox created successfully
  else
    echo An error occurred creating a tunnel to jumpbox. RC was $?
  fi
}
/bin/pidof ssh
if [[ $? -ne 0 ]]; then
  echo Creating new tunnel connection
```

```
  createTunnel
fi
```

Zoznam príkazov na nastavenie rpi komunikácie

príkazy

```
# add ssh file to filesystem root on boot partition of rpi to enable ssh
# add wpa_supplicant file to rpi root partition boot
# this for automatic wifi connection
# example of wpa_supplicant file(file is deleted after first reboot):
country=SK # Your 2-digit country code
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
network={
    ssid="marek-nb"
    psk="76O6y;69"
    key_mgmt=WPA-PSK
}
#update and upgrade of new rpi
sudo apt-get update && sudo apt-get upgrade

#docker
curl -sSL https://get.docker.com | sh

#pip
sudo apt-get install libffi-dev libssl-dev
sudo apt install python3-dev
sudo apt-get install -y python3 python3-pip

#docker-compose
sudo pip3 install docker-compose

# docker on startup
sudo systemctl enable docker

# make script for reverse shell
# change 2222 to wanted port if needed
# this command should be run from rpi for adding to known hosts
# /usr/bin/ssh -N -f -R 2224:localhost:22 ubuntu@team06-21.studenti.fiit.stuba.sk
# after that you need to connect to rpi with ubuntu
# ssh -p 2222 pi@localhost
sudo nano create_ssh_tunnel
```

```
#script
#!/bin/bash
createTunnel() {
  /usr/bin/ssh -N -f -R 2224:localhost:22 ubuntu@team06-21.studenti.fiit.stuba.sk
  if [[ $? -eq 0 ]]; then
    echo Tunnel to jumpbox created successfully
  else
    echo An error occurred creating a tunnel to jumpbox. RC was $?
  fi
}
/bin/pidof ssh
if [[ $? -ne 0 ]]; then
  echo Creating new tunnel connection
  createTunnel
fi

chmod +x create_ssh_tunnel

# make new sudo cron schedule for script
crontab -e
*/1 * * * * ./create_ssh_tunnel > tunnel.log 2>&1

ssh-keygen
# ssh to ubuntu ssh copy id
ssh ubuntu@team06-21.studenti.fiit.stuba.sk
vim .ssh/authorized_keys # add id_rsa.pub generated by rpi + increment identificator at
the pi@raspberrypiv[version-number]
#unlock port for listening firevall
sudo netstat -tulpn # check listening ports
sudo ufw allow 2222

# ssh keygen for ssh deploy keys one for frontend and one for backend
ssh-keygen -t rsa -f ~/.ssh/frontend -C "frontend"
ssh-keygen -t rsa -f ~/.ssh/backend -C "backend"
cat  ~/.ssh/frontend.pub

cat  ~/.ssh/backend.pub

# add deploy keys to repos
# In the server's SSH configuration file (usually ~/.ssh/config), add an alias entry for
each repository.

cat <<EOT >> ~/.ssh/config
Host github-frontend
HostName github.com
User git
```

```
IdentityFile ~/.ssh/frontend

Host github-backend
HostName github.com
User git
IdentityFile ~/.ssh/backend
EOT

git clone github-frontend:smart-brew/frontend.git
git clone github-backend:smart-brew/backend.git

# create startup file
chmod +x startup
#!/bin/bash
cd ~/frontend
sudo docker-compose up
cd ~/backend
sudo docker-compose up

#autostart of chromium
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart
#add
@chromium-browser localhost --start-fullscreen --kiosk
```

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Spustenie pivovaru

Link na repozitár, ktorý obsahuje automatický script: startup

Spustenie testovacej verzie

V tejto verzii je nahradený modul, testovacím serverom, ktorý simuluje funkcionalitu modulov.

Postup

0. Stihanuť si všetky repozitáre

- Testovací modul (link na repo: module-mock-server)
- Frontend (link na repo: frontend)
- Backend (link na repo: backend)

```
git clone https://github.com/smart-brew/module-mock-server.git
git clone https://github.com/smart-brew/frontend.git
git clone https://github.com/smart-brew/backend.git
```

1. Spustiť testovací modul

```
cd module-mock-server
docker compose build
docker compose up
```

2. Spustiť backend (spolu aj s databázou)

```
cd backend
docker compose build
docker compose up
```

> **POZNÁMKA**
>
> Ak nebola doposial spustená databáza a sú problémy s nesprávnymi dátami, treba vykonať manuálne migráciu spolu so seedom údajov - viac info: Migrácie

3. Spustiť GUI

```
cd frontend
docker compose build
docker compose up
```

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Tímová VM

SSH

```
ssh -i path/to/key ubuntu@team06-21.studenti.fiit.stuba.sk
```

Údaje od SSH pripojenia

Adresa

```
team06-21.studenti.fiit.stuba.sk
```

Username

```
ubuntu
```

Key (ukážka)

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpA************************
*****************************
**********************8+1y9A==
-----END RSA PRIVATE KEY-----
```

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Tímový web + dokumentácia

Tímový web aj spolu s dokumentáciou sa automaticky deploy-ne na http://team06-21.studenti.fiit.stuba.sk, pričom dokumentácia je na http://team06-21.studenti.fiit.stuba.sk/docs/.

Setup deploymentu

Deployment sa robí automaticky z najnovšieho commitu na `master` vetve za pomoci GitHub Actions

Po commite sa vykoná krátky script, ktorý zbuilduje stránku a nahrá súbory pomocou `rsync` z GitHubovej virtuálky na náš stroj `team06-21.studenti.fiit.stuba.sk` do priečinka `~/web` a `~/web/docs`.

Server

Ako server na hostovanie webu sa používa http-server, keďže všetky súbory sú statické, stačí ich len vedieť poskytnúť používateľovi.

Automatické spustenie servera:

1. Mať nainštalovaný `http-server` na VM
2. Vo VM spustiť príkaz:

```
crontab -e
```

3. Na koniec súboru pridať:

```
@reboot cd ~/web && sudo http-server -p 80
```

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Analýzy

Brewery

Základná charakteristika

Pozostáva z dvoch nádob, ktoré sú navzájom prepojené. Na rozdiel od zariadenia s troma nádobami mu chýba špeciálna nádoba pre ohrev vody. Tím sa zníži nielen potreba priestoru na skladovanie pivovaru, ale aj čas k vareniu piva.

Základné postupy pri varení piva

1. Rozdrvený jačmenný slad sa zmieša s teplou (nie vriacou) vodou. Nechá sa namáčať. Možné pridať iné prísady, ako napríklad pomletú ryžu. Dôkladným zamiešaním vzniká hustá kaša, tzv. vystierka.
2. Zmes sa za stáleho miešania zahrieva pri teplote do 100 stupňov niekoľko hodín. Existujú rôzne spôsoby udržiavania teploty, ktoré sú popísané v ďalšej časti. Pri tomto procese sa škrob mení na cukry.
3. Následne je potrebné scediť tuhú časť zmesi od tekutej. Výsledkom je roztok sladkej chuti nazývaný slatina.
4. Slatina sa ďalej varí s chmeľom, ktorý do zmesi pridá horkú príchuť. Chmeľ sa necháva len vylúhovať, po tomto procese sa odstráni. Výsledkom je mladina, ktorá sa vírením zbaví kalov.
5. Po schladení sa pridávajú kvasnice. Dochádza k zmene cukrov na alkohol.

Zahrievanie

Existujú dva hlavné spôsoby ako zabezpečiť zohrievanie pri varení piva v dvoch nádobách. Sú nimi RIMS a HERMS.

Spoločné znaky:

- na redistribuovanie sa využíva pumpa
- poskytujú zdroj tepla na zohrievanie
- neustále prihrievajú zmes mimo kade a navracajú ju tak, aby sa udržala konzistentná teplota počas daného procesu

HERMS Využíva ďalšiu nádobu, v ktorej sa zahrieva voda. Cez vodu vo vnútri tejto nádoby vedie cievka, cez ktorú je tlačená zmes z kade. Tento proces prebieha v cykloch kedy sa zmes zohreje a následne navráti do

kade.

RIMS Zdroj tepla prichádza do priameho kontaktu so zmesou. Tento proces taktiež prebieha cyklicky. Ďalším spôsobom je pôsobenie tepla priamo na spodok kade, v tom prípade je ale potrebné obsah dostatočne premiešavať aby nedošlo ku pripáleniu.

Možné kroky automatizácie pri procese varenia

- miešanie v kadi - určenie rýchlosti miešania (počet otáčok za minútu) a potrebného času (minúty)
- pohyb medzi kaďami. Nastavenie cirkulácie na istý objem (napr. cm3 za minútu). Otvorenie a uzatvorenie kohútikov (onen/close).
- udržanie správnej teploty - teplomer (v stupňoch celzia). Podľa toho ako je zabezpečené zahrievanie je potrebné: zvýšiť teplotu / znížiť teplotu
- pridanie chmeľu / kvasiniek. Ak automaticky tak na timer (v minútach/hodinách) podľa etapy procesu, možná kontrola či je vhodná teplota už dosiahnutá alebo podobne.

Recept:

- Vodu na varenie piva predohriať na 68 stupňov.
- Pomlieť jačmeň a namáčať ho v tejto teplote po 30 minúť.
- Prefiltrovať pevnú zložku zmesi.
- Nechať prevrieť po 70 minút. Pridávať chmeľ v intervaloch. Pridať Whirlfloc tabletu 15 min do konca.
- Ochladiť na teplotu 18 stupňov, premiestniť do suda a pridať kyslík spolu s kvasnicami.
- Uzatvoriť nádobu a držať 2 týždne.

Postupy pri vykonávaní receptu, ako je zakreslené v diagrame

Vstup: voda v ohrevnom kotli, namletý jačmeň v násypníku, kvasnice v násypníku, chmeľ pripravený vo vrecúšku, iné prísady pripravené (whirlgloc tablety, pomletá ryža, cukor,...).

- Ohrievač sa má zohriať vodu na 68 stupňov. Teplomer bude každých n sekúnd hlásiť priebežnú teplotu vody. Keď je teplota dosiahnutá, ohrievač túto teplotu musí udržiavať – t.j. ak klesne pod nejakú hranicu tak ju dohreje.

- Násypník dostane príkaz pridať jačmeň do vody. Nastaviť časovač na 30 minút, počas toho miešať zmes pri určitom počte otáčok za minútu. Po vypršaní času sa ukončí miešanie. Otvorí sa ventil na prvej nádobe, zapne sa výleva – podľa nejakého nastavenia bude pumpovať tekutinu do druhej nádoby. Zatvorí sa ventil na druhej nádobe.
- Druhá nádoba sa zohreje na požadovanú teplotu 98 stupňov (ohrievač, kontroluje teplomer). Časovač sa nastaví na 70 minút po dosiahnutí teploty varu. N minút do konca časovača sa z násypníka pridá do nádoby chmeľ na vylúhovanie. N minút do konca procesu sa pridá whirlfloc tableta. Časovač ohlási koniec 70 minút.
- Odstránenie kalov?
- Tekutina sa má zachladiť na teplotu 18 stupňov // vykonáva kryostat??. Pridajú sa kvasnice z násypníka, nechá da odležať pričom sa teplota udržiava.

Výstup: pivo pripravené na točenie, zvyšky pomletého jačmeňa v prvej nádobe, odstránený chmeľ

Zdroje

- https://www.homebrewhappyhour.com/rims-herms-systems-explained-compared/
- https://siov.sk/wp-content/uploads/2020/04/Kvasn%C3%A9-technol%C3%B3gie-v%C3%BDroba-piva.pdf
- https://buchvald.sk/postup-pri-vyrobe-piva/
- https://beerandbrewing.com/beer-recipes/?cbb_web_beer_recipes%5Bquery%5D=easy // inšpirácia k receptu

*Last updated on **18. 11. 2021** by **Michaela Nemcová***

Modbus

Modbus je otvorený protokol pre vzájomnú komunikáciu rôznych zariadení (RTU, PLC, dotykové displeje, I/O rozhrania apod), ktorý umožňuje prenášať dáta po rôznych sieťach a zberniciach. Komunikácia funguje na princípe predávánia datových správ mezi klientom a serverom (**master** a **slave**).

Ako modbus funguje?

Zvyčajne býva jeden master a viacero slave zariadní (pri TCP môže byť viac mastrov). Inicializovať komunikáciu môže **iba master** a slave mu vždy len posiela odpoveď.

Protokol modbus určuje že každé zariadenie má nejakú štruktúru paketu. Býva tam:

- id ciela
- funkcia, ktorú chcem vykonať
- (optional) data
- CRC checksum

Prenosové médiá

- Ethernet cez TCP/IP
- Sériový prenos (RS-232C, RS-422, RS-485, optické vlákno, radiový prenos)
- MODBUS PLUS vysokorychlostná sieť

Preferovaný je sériový prenos cez RS-485, za režimu linky 19200 baudov, 8 dátovych bitov a párna parita.

Funkcie a úložisko

Typ	Operácia	Veľkosť	Povolené adresy
Coil	Read-Write	1 bit	0x0000 – 0xFFFF
Discrete input	Read	1 bit	0x0000 – 0xFFFF

Typ	Operácia	Veľkosť	Povolené adresy
Input register	Read	16 bits	0x0000 – 0xFFFF
Holding register	Read-Write	16 bits	0x0000 – 0xFFFF

Zdroje

- https://modbus.org
- https://en.wikipedia.org/wiki/Modbus
- https://www.youtube.com/watch?v=JBGaInI-TG4
- https://www.npmjs.com/package/modbus-serial
- Meniče H-300 https://docs.google.com/document/d/1RQNIFq4vbOvbZFfxEuz4CkOA-SDlksPzpciTiBFUaeA/edit?usp=sharing
- https://www.youtube.com/watch?v=i46jdhvRej4

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

SCADA

SCADA je skratka pre **S**upervisory **C**ontrol **A**nd **D**ata **A**cquisition.

Základná charakteristika

Ide o koncept kontrolného systému, kde je potreba monitorovať, kontrolovať, získavať a analyzovať real-time dáta z priemyselných zariadení a procesov.

> Any application that gets data about a system in order to control that system is a SCADA application.

Kľúčové komponenty

SCADA systém sa najčastejšie skladá z komponentov:

- Supervisory computer
- Remote terminal units
- Programmable logic controllers
- Communication infrastructure
- Human-machine interface

Supervisory computer

- **jadro SCADA systému** - zbiera dáta z procesu a posiela príkazy do pripojených zariadení
- zahŕňa počítač a softvér komunikujúci s pripojenými zariadeniami + HMI softvér bežiaci na pracovných staniciach operátorov
- pri veľkých systémoch môže ísť o viacero serverov, viacero HMI softvérov na klientských počítačoch (proste massive)

Remote terminal/telemetry units (RTUs)

- prepája fyzické zariadenia (senzory, efektory) na hlavný kontrolný systém (SCADA systém)
- prenáša telemetrické dáta na hlavný systém
- pomocou správ z hlavného systému kontroluje pripojené fyzické zariadenia

Programmable logic controllers (PLCs)

- počítač používaný na automatizáciu elektromechanických procesov
- môže mať veľa inputov a outputov
- má naprogramovanú logiku, ktorá spracováva dáta z pripojených senzorov a riadi pripojené efektory

Communication infrastructure

- prepája hlavný kontrolný systém s RTU-čkami a PLC-čkami
- využíva komunikačné protokoly podľa industry štandardov

Human-machine interface (HMI)

- využívané pre vizualizáciu a kontrolu procesu
- má nejakú obrazovku (napr. monitor), cez ktorú môžu používatelia interagovať priamo so strojmi alebo kontrolovať zariadenia
- prezentuje dáta získané zo zariadení (PLC / RTU)
- obrazovka zobrazuje grafickú reprezentáciu vybavenia / procesov, ktoré je treba monitorovať
  - mimic diagram
- grafická reprezentácia môže byť vytvorená cez špeciálny softvér
  - napr. SIMATIC WinCC

Využitie

- elektrárne
- vodárne, čističky vôd
- inteligentné budovy, továrne
- ...

Zdroje

- https://en.wikipedia.org/wiki/SCADA
- https://steemit.com/steemstem/@ibk-gabriel/scada-systems-principle-and-applications

- https://www.copadata.com/en/industries/food-beverage/food-beverage-insights/creative-brewing/
- https://www.automationworld.com/products/control/blog/13307242/rtu-or-plc-which-is-right-for-you

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Architektúra

API

Rôzne komponenty spolu komunikujú rôznym spôsobom, preto je dôležité mať jedno miesto, kde sa dá prehľadne pozrieť, ako fungujú naše API.

Module <--> Backend

Modul komunikuje smerom na backend cez WebSocket a posiela periodicky nasledujúce správy:

Posielanie údajov z modulu na backend

Posielanie údajov z modulu na backend

```
{
  "moduleId": "<názov modulu>",
  "status": "<ok | progress | error>",
  <podporované údaje>
}
```

- **<podporované údaje>** - sú rovnaké údaje ako v Podporované údaje

Posielanie inštrukcií z backendu na modul

Posielanie inštrukcií z backendu na modul

```
{
  "moduleId": "<názov modulu>",
  "type": "instruction",
  "category": "<ketegória>", // "MOTOR", "PUMP", ...
  "device": "<názov zariadenia>", // "MOTOR_1", "PUMP_2", ...
  "instruction": "<inštrukcia>", // "SET_TEMPERATURE", "SET_MOTOR_SPEED", ...
  "parameter": "<parameter>"
}
```

- hodnoty budu podľa definície v Podporované funkcie

Backend <--> Frontend

Pri každom dopyte a jeho odpovedi, sa budú nachádzať aj `created_at` a `updated_at`, avšak pre prehľadnosť, tu niesú napísané.

Podporované funkcie

FE môže získať informácie o podporovaých funkciách v rámci systému.

```
GET /api/function
```


```
[
  {
    "id": 1,
    "instruction": "SET_TEMPERATURE",
    "name": "Set temperature",
    "category": "TEMPERATURE",
    "units": "°C",
    "inputType": "float",
    "description": "Sets temerature for selected chamber",
    "devices": [
      {
        "id": 1,
        "name": "Chamber 1",
        "device": "TEMP_1"
      },
      {
        "id": 2,
        "name": "Chamber 2",
        "device": "TEMP_2"
      }
    ]
  },
  {
    "id": 2,
    "instruction": "SET_MOTOR_SPEED",
    "name": "Set motor speed",
    "category": "MOTOR",
    "units": "RMP",
    "inputType": "float",
    "description": "Sets rpms for selected motor",
    "devices": [
```

```
      {
        "id": 3,
        "name": "Motor 1",
        "device": "MOTOR_1"
      },
      {
        "id": 4,
        "name": "Motor 2",
        "device": "MOTOR_2"
      }
    ]
  },
  {
    "id": 3,
    "instruction": "TRANSFER_LIQUIDS",
    "name": "Transfer liquids",
    "category": "PUMP",
    "units": null,
    "inputType": null,
    "description": "Transfers liquids from first chamber to second",
    "devices": [
      {
        "id": 5,
        "name": "Pump 1",
        "device": "PUMP_1"
      }
    ]
  },
  {
    "id": 4,
    "instruction": "UNLOAD",
    "name": "Unload ingredient",
    "category": "UNLOADER",
    "units": null,
    "inputType": null,
    "description": "Unloads selected ingredient into chamber",
    "devices": [
      {
        "id": 6,
        "name": "Fermentables",
        "device": "FERMENTABLE"
      },
      {
        "id": 7,
        "name": "Yeast",
        "device": "YEAST"
```

```
      },
      {
        "id": 8,
        "name": "Hops",
        "device": "HOPS"
      },
      {
        "id": 9,
        "name": "Other",
        "device": "OTHER"
      }
    ]
  },
  {
    "id": 5,
    "instruction": "WAIT",
    "name": "Wait",
    "category": "SYSTEM",
    "units": "Minutes",
    "inputType": "float",
    "description": "System will wait for given amount of minues",
    "devices": []
  },
  {
    "id": 6,
    "instruction": "MANUAL",
    "name": "Manual step",
    "category": "SYSTEM",
    "units": null,
    "inputType": "string",
    "description": "System will wait for manual inervention",
    "devices": []
  }
]
```


Výber všetkých receptov

Pri zapnutí si používateľ bude môcť vybrať, či chce použiť už vytvorený recept, alebo či chce vytvoriť nový recept.

Pokiaľ si vyberie použitie vytvoreného receptu, front-end pošle GET request na back-end URL:

```
GET /api/recipe
```

Ako odpoveď obdrží základné informácie (id, názov, čas vytvorenia) pre **všetky** recepty v JSON formáte:

```
[
  {
    "id": 3,
    "name": "Smoky Grove Lichtenhainer",
    "description": "Light, gently tart, and smoked—lichtenhainer is an unusual beer, yet surprisingly good for all seasons and one you'll want to brew and enjoy often.",
    "locked": false
  },
  {
    "id": 4,
    "name": "Burke-Gilman The Hopsplainer",
    "description": "Courtesy of the brewing team at Burke-Gilman in Seattle, here is a homebrew-scale recipe for the double hazy IPA that won GABF gold in 2020.",
    "locked": true
  },
  ...
]
```

Výber jedného receptu

FE ponúkne používateľovi výber z prijatých receptov podľa názvu. Pri vybratí receptu sa odošle GET request na backend:

```
GET /api/recipe/{recipe-id}
```

Odpoveďou bude JSON so všetkými dátami receptu:

```
{
  "id": 3,
  "name": "Smoky Grove Lichtenhainer",
  "description": "Light, gently tart, and smoked—lichtenhainer is an unusual beer, yet surprisingly good for all seasons and one you'll want to brew and enjoy often.",
  "locked": false,
  "Ingredients": [
    {
      "id": 5,
      "recipeId": 3,
      "name": "American - Pale 2-Row",
```

```
      "amount": 5.6,
      "type": "Fermentable",
      "units": "Kg"
    },
    {
      "id": 6,
      "recipeId": 3,
      "name": "Fermentis - Safale - American Ale Yeast US-05",
      "amount": 1,
      "type": "Yeast",
      "units": ""
    },
    {
      "id": 7,
      "recipeId": 3,
      "name": "Magnum (Pellet)",
      "amount": 1,
      "type": "Hops",
      "units": "oz"
    },
    {
      "id": 8,
      "recipeId": 3,
      "name": "Crush whilrfoc Tablet",
      "amount": 1,
      "type": "Other",
      "units": ""
    }
  ],
  "Instructions": [
    {
      "id": 11,
      "recipeId": 3,
      "templateId": 2,
      "instruction": "SET_MOTOR_SPEED",
      "param": 30,
      "category": "MOTOR",
      "device": "MOTOR_1",
      "blockId": 1,
      "block": "Fermentation",
      "ordering": 1
    },
    {
      "id": 12,
      "recipeId": 3,
      "templateId": 1,
```

```
      "instruction": "SET_TEMPERATURE",
      "param": 85,
      "category": "TEMPERATURE",
      "device": "TEMP_1",
      "blockId": 1,
      "block": "Fermentation",
      "ordering": 2
    },
    {
      "id": 13,
      "recipeId": 3,
      "templateId": 2,
      "instruction": "SET_MOTOR_SPEED",
      "param": 0,
      "category": "MOTOR",
      "device": "MOTOR_1",
      "blockId": 2,
      "block": "Yeasting",
      "ordering": 3
    },
    {
      "id": 14,
      "recipeId": 3,
      "templateId": 1,
      "instruction": "SET_TEMPERATURE",
      "param": 23,
      "category": "TEMPERATURE",
      "device": "TEMP_1",
      "blockId": 2,
      "block": "Yeasting",
      "ordering": 4
    }
  ]
}
```


Výber receptu na varenie

Pokiaľ je používateľ s vybraným receptom spokojný, klikne na tlačidlo **"Vybrať recept"**. Recept sa následne načíta na hlavnú obrazovku a tiež sa odošle POST request na BE s vybraným receptom (aby BE vedel, že čo sa bude robiť):

```
POST /api/recipe/{recipe-id}/load
```

Pridanie nového receptu

Pokiaľ používateľ chce vytvoriť nový recept, pomocou obrazovky pre pridanie receptu vykliká všetky požadované kroky a klikne na tlačidlo **"Pridať recept"**. Na BE sa odošle PUT request:

```
PUT /api/recipe
```


```
{
  "name": "Perfect Northeast IPA (NEIPA)",
  "description": "I have no idea what I'am doing",
  "locked": false,
  "Ingredients": [
    {
      "name": "American - Pale 2-Row",
      "amount": 5.6,
      "type": "Fermentable",
      "units": "Kg"
    },
    {
      "name": "Fermentis - Safale - American Ale Yeast US-05",
      "amount": 1,
      "type": "Yeast",
      "units": ""
    },
    {
      "name": "Magnum (Pellet)",
      "amount": 1,
      "type": "Hops",
      "units": "oz"
    },
    {
      "name": "Crush whilrfoc Tablet",
      "amount": 1,
      "type": "Other",
      "units": ""
    }
  ],
  "Instructions": [
    {
      "templateId": 4,
      "param": null,
      "deviceId": 6,
      "blockId": 1,
```

```
      "ordering": 4
    },
    {
      "templateId": 1,
      "param": "60",
      "deviceId": 1,
      "blockId": 1,
      "ordering": 3
    },
    {
      "templateId": 5,
      "param": "5",
      "deviceId": null,
      "blockId": 2,
      "ordering": 2
    },
    {
      "templateId": 2,
      "param": "100",
      "deviceId": 3,
      "blockId": 2,
      "ordering": 1
    }
  ]
}
```

Ako odpoveď na FE príde JSON s vygenerovaným ID pridaného receptu:

```
200 OK
{
  "id" : xxx
}
```

Po vytvorení receptu sa vrátime na výber receptov, kde bude už nový recept zobrazený.

Začiatok varenia

Po vybratí receptu sa recept načíta na hlavnú obrazovku. Používateľ dostane príležitosť skontrolovať, či je recept správny. Pokiaľ je s receptom spokojný, spustí varenie. Pri spustení varenia sa na BE pošle PUT request:

```
PUT /api/brew/0/start
```

```
{
  "recipeId": 123 // id vybraného receptu
}
```

FE čaká na odpoveď z BE, či sa všetko úspešne spustilo:

```
200 OK
{
    "brewId" : xxx
}
```

FE si uloží ID varenia pre ďalšie dopyty.

Pokiaľ na BE nastane porucha alebo chyba pri spúšťaní, na FE pošle odpoveď:

```
500 SERVER ERROR
{
  "error" : "Temp error message."
}
```

FE túto chybu následne ohlási používateľovi.

Periodické dopyty na back-end

Pokiaľ sa varenie spustí úspešne, FE prejde do módu, kde sa periodicky dopytuje BE na stav receptu. Každú 1 sekundu na BE odošle GET request na URL:

```
GET /api/data
```

Pokiaľ všetko prebieha v poriadku

V ideálnom prípade BE odpovie formou:

```
200 OK
{
```

```
  "data": {
    <podporované údaje>
  },
  "instruction" : {
    "currentInstruction": 23,
    "currentValue": 36.5,
    "status": "IN_PROGRESS",
  },
  "brewStatus": "IN_PROGRESS"
}
```

- **<podporované údaje>** - sú rovnaké údaje ako v Podporované údaje

FE túto odpoveď spracuje a obnoví obrazovku.

Pokiaľ nastane chyba

Pokiaľ došlo k chybe niekde v pipeline, BE odpovie formou:

```
500 SERVER ERROR
{
  "error": "Temp error message.",
  "module": {
    "moduleId": 234,
    "device": "MOTOR_1",
    "category": "MOTOR",
    "error": "Cannot communicate with device"
  },
  "instruction" : {
    "currentInstruction": 23,
    "currentValue": 36.5,
    "status": "FAIL"
  },
  "brewStatus": "STOPPED"
}
```

Úspešné ukončenie varenia

Pokiaľ BE úspešne ukončil varenie, pri najbližšom GET dopyte (`GET /api/data`) BE pridá položku `status: "FINISHED"`:

```
200 OK
{
  ...
  ...
  "brewStatus": "FINISHED"
}
```

FE o tejto skutočnosti upovedomí používateľa a ukončí mód periodických dopytov.

Potvrdenie manuálnej inštrukcie

Keď nastane situácia, že treba vykonať manuálnu inštrukciu, systém bude čakať až kým používateľ nepotvrdí jej vykonanie.

```
POST /api/brew/{brewId}/instruction/{instructionId}/done
```

```
200 OK
```

Až po nasledujúcom dopyte v Periodické dopyty na back-end sa prejde na ďalšiu inštrukciu, keď nám BE pošle, že sme na nasledujúcej inštrukcii.

Úprava parametrov počas varenia

Pokiaľ nastane zmena parametrov nejakých krokov, ktoré ešte neboli vykonané, FE odošle POST request na BE:

```
POST /api/brew/{brewId}/instruction/{instructionId}
```

```
{
  // celá inštrukcia s upravenými paremtrami, viď Štruktúra JSONu inštrukcie"
}
```

Pokiaľ úprava prebehne v poriadku, BE odpovie správou:

```
200 OK
```

Pokiaľ úpravu nebolo možné vykonať, BE odpovie správou:

```
400 BAD REQUEST
{
    "error" : "Temp error message."
}
```

Note: je potrebné, aby FE po tomto POST requeste spustil timer na periodické dopyty odznova, aby sme sa vyhli nepríjemnostiam s asynchronicitou BE.

Pauza počas varenia

Na front-ende by mala byť možnosť pauznúť varenie používateľom. Používateľ klikne na tlačidlo **"Pauza"**. FE sa opýta, či si je používateľ istý.

Po potvrdení je na BE odoslaný POST request:

```
POST /api/brew/{brewId}/pause
```

BE by mal zastaviť časovače na všetkých aktívnych procesoch a odpovedať formou:

```
200 OK
```

Pokiaľ pri zrušení nastane chyba, BE odpovie formou:

```
500 SERVER ERROR
{
    "error" : "Temp error message."
}
```

Pokračovanie varenia

Ak chceme pokračovať vo varení, pošleme:

```
POST /api/brew/{brewId}/resume
```

Zrušenie varenia

Na front-ende by mala byť možnosť prerušiť varenie používateľom. Používateľ klikne na tlačidlo **"Zrušiť varenie"**. FE sa opýta, či si je používateľ istý.

Po potvrdení je na BE odoslaný POST request:

```
POST /api/brew/{brewId}/abort
```

BE by mal zrušiť všetky procesy, správne vypnúť všetky zariadenia a odpovedať formou:

```
200 OK
```

Pokiaľ pri zrušení nastane chyba, BE odpovie formou:

```
500 SERVER ERROR
{
    "error" : "Temp error message."
}
```

Now it's time to panic.

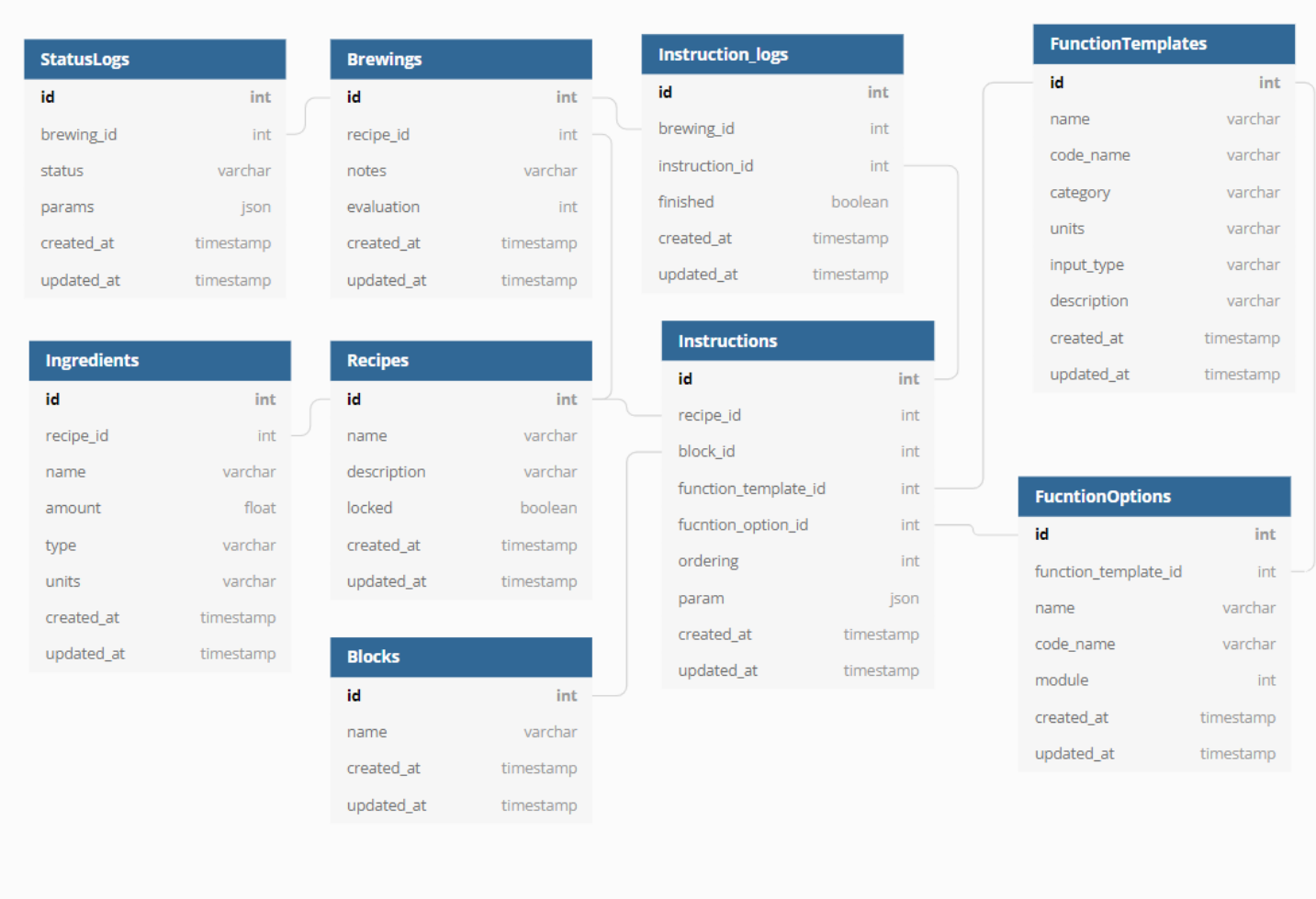
*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Databáza

Technológia PostgreSQL

Návrh databázy

Návrh databázy je nasledovný:



Front-end

Link na repozitár - Frontend

Technológie

Pre používateľské rozhranie budeme používať **React Native framework**.

Pre IDE je treba doinštalovať tieto rozšírenia - **eslint, prettier, editorconfig** (odporúčaný je VSCode).

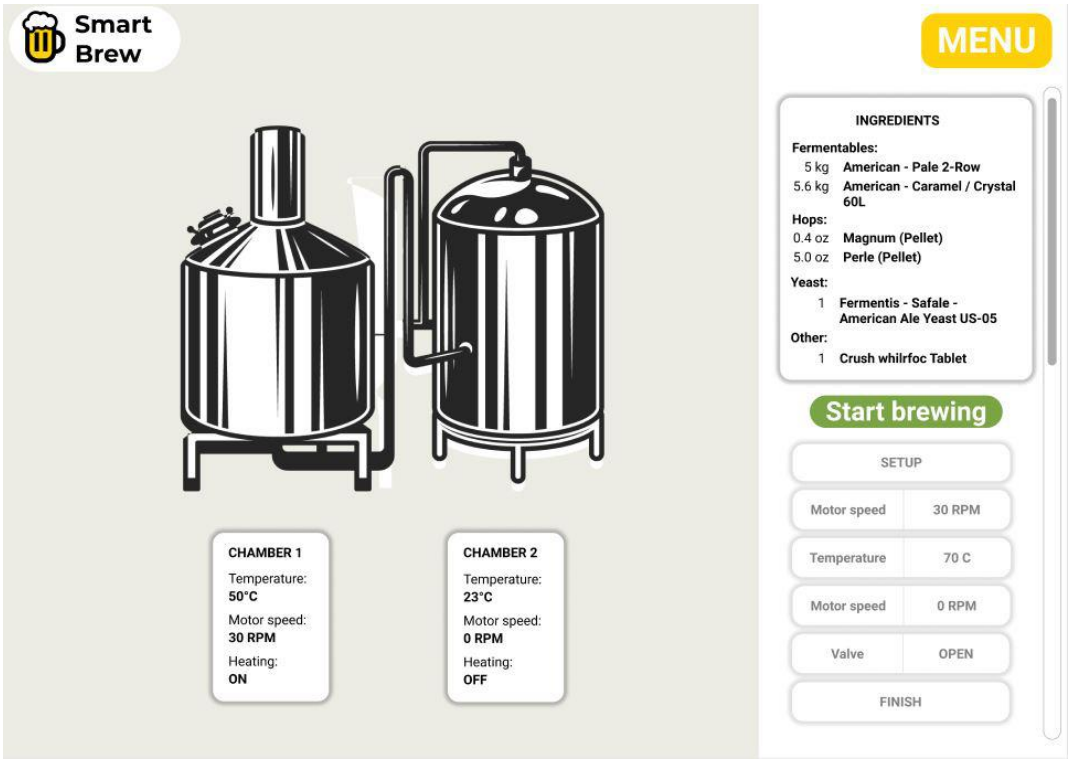
Komunikácia s back-endom

Pre komunikáciu využijeme REST API cally a správy vo formáte JSON. Viac info na API.

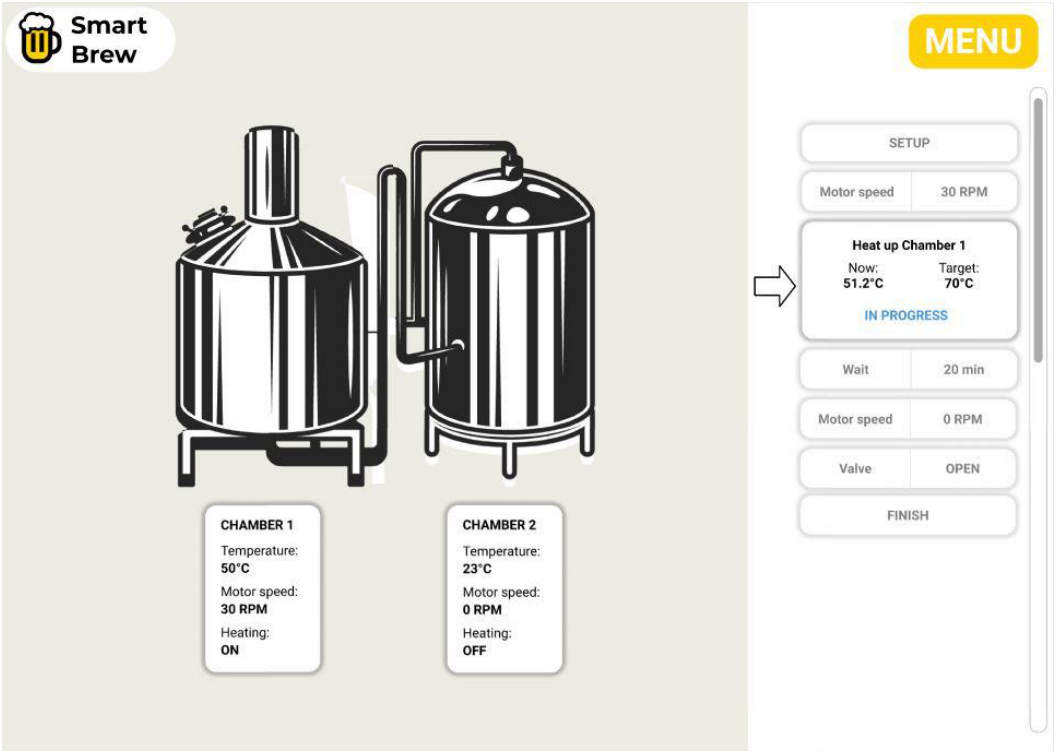
Opis prostredia

Návrh užívateľského prostredia prešiel viacerými kolami úprav. Dole je opísaná verzia aktuálna po dokončení 3. šprintu, spolu s návrhmi úprav do ďalších šprintov.

Hlavná obrazovka



Obrázok zobrazuje stav, kedy je navolený recept, ale nie je potvrdené začatie procesu. Na ľavej strane obrazovky sa nachádza zariadenie, spolu s informáciami o procesoch vykonávajúcich sa v jednotlivých nádobách. Napravo sú zobrazené potrebné ingrediencie, spolu s inštrukciami, ktoré sa budú vykonávať. (Používateľ si môže skontrolovať, či má všetky dostupné ingrediencie, a v prípade potreby vybrať iný recept).

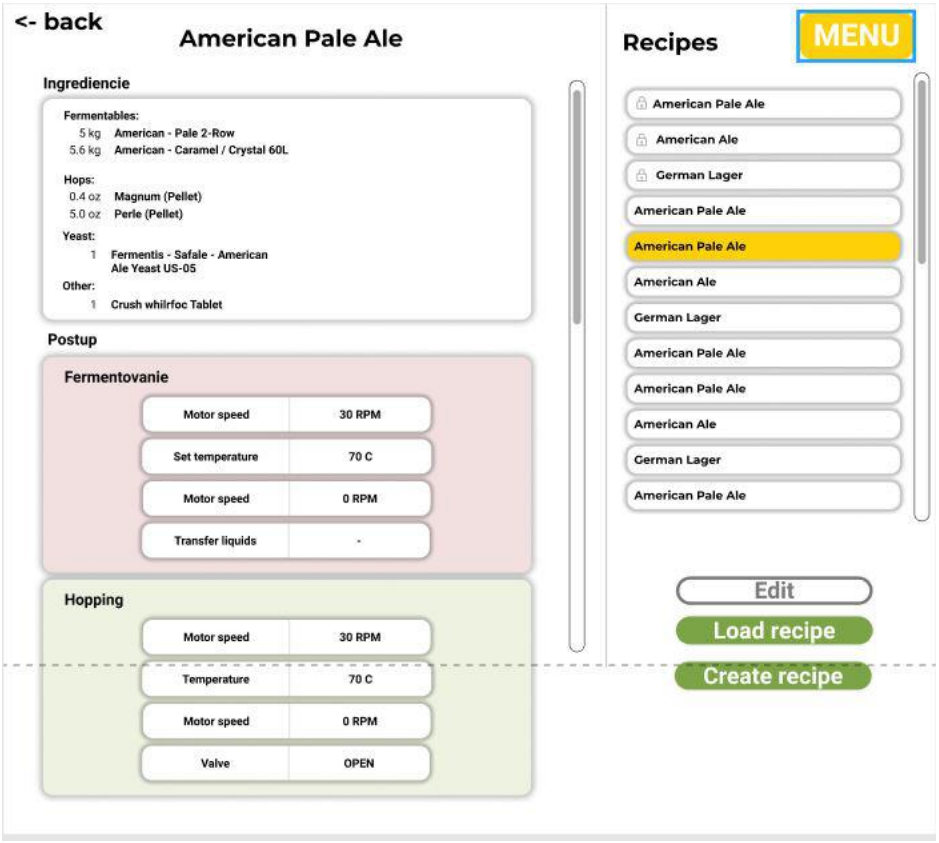


Po začatí varenia sa používateľovi zobrazujú bližšie informácie o prebiehajúcich (i ukončených a nasledujúcich) procesoch.

Odsúhlasené úpravy

- Informácie k zariadeniu budú zobrazované inak. K detailnejšiemu nákresu zariadenia budú priradené informácie, ktoré sa priamo týkajú danej časti zariadenia.
- zobrazované budú 4 typy tlačidiel (buď v menu, alebo každé bude permanentne zobrazené na hornej časti obrazovky). Budú odkazovať na stránky: hlavná stránka, výber receptu, história varenia. Ďalej bude možné tlačidlom korektne vypnúť zariadenie (t.j. zastaviť celé Raspberry Pi)
- Po stlačení tlačidla **Start brewing** sa zobrazí vyskakovacie okno s informáciami, ktoré ingrediencie treba dať do ktorého násypníka. // toto je už pripravené, len to treba presunúť na danú stránku

Výber receptu

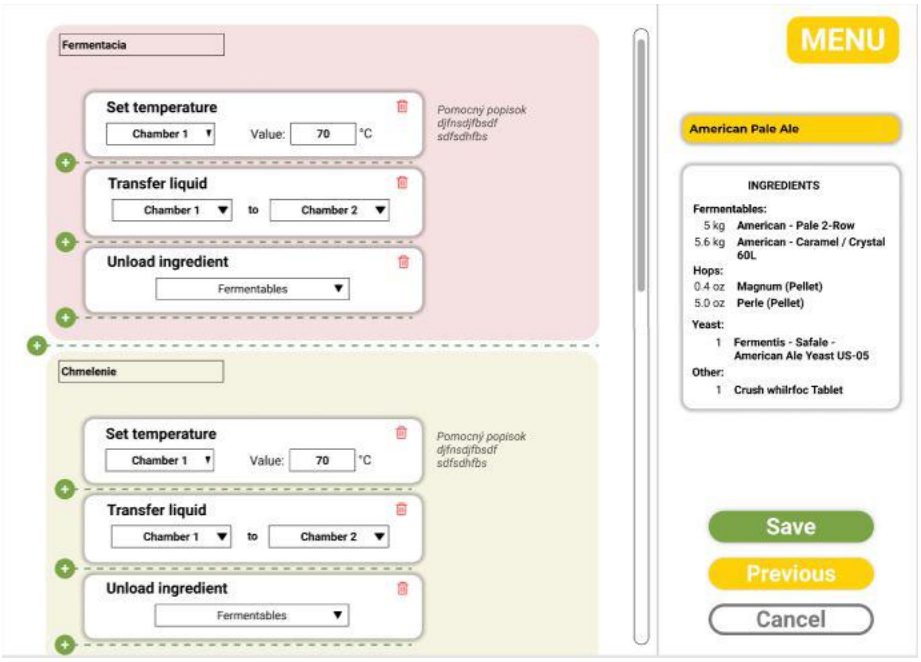


Pokiaľ si používateľ chce zvoliť recept na varenie, kliknutím na možnosť **Výber receptu** v menu prejde na obrazovku výberu receptu. Na pravej strane obrazovky sa zobrazí zoznam všetkých dostupných receptov. Po kliknutí na recept sa zobrazia informácie k danému receptu. Používateľ tento recept môže použiť tak, aký je, a načítať ho na hlavnú obrazovku, alebo ho upraviť. Tiež si môže vytvoriť úplne nový recept.

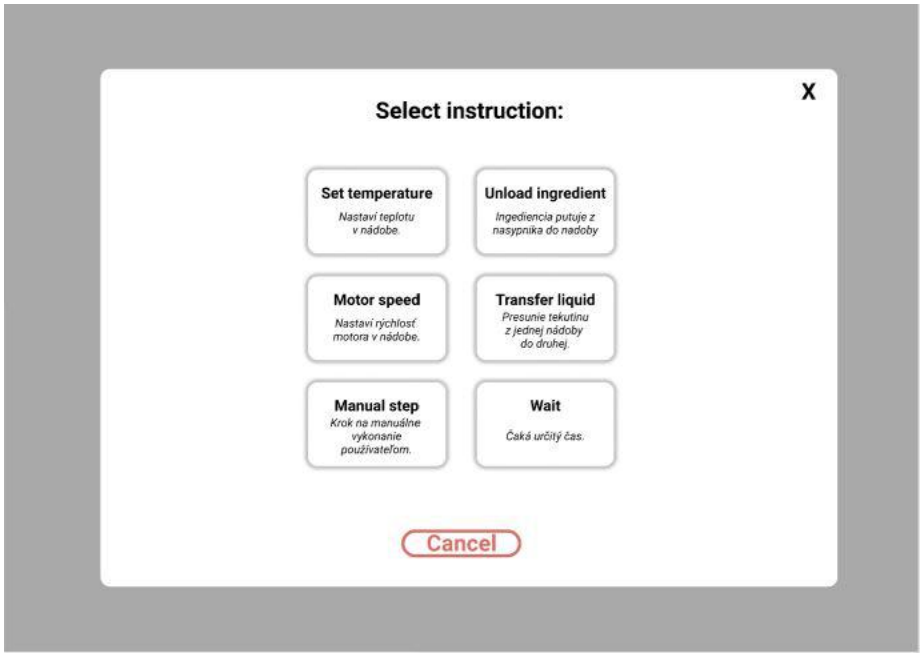
Pridanie ingredencií

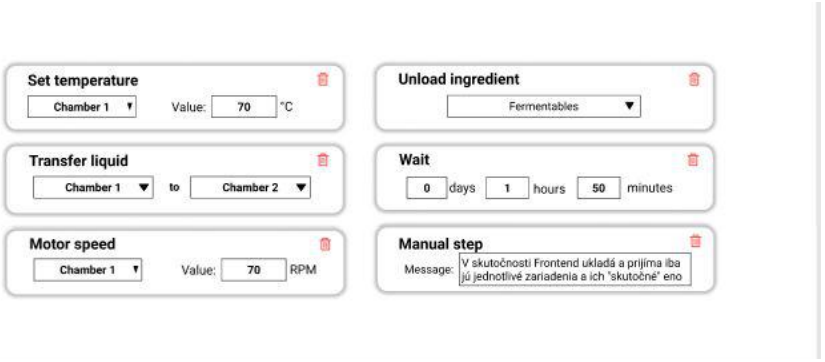
Pokiaľ si používateľ vyberie možnosť upraviť/vytvoriť recept, dostane sa na obrazovku pre vytvorenie zoznamu potrebných ingrediencií. Formulár na pridanie ingredencií obsahuje delenie na jednotlivé kategórie ingrediencií, a možnosť pridať a odobrať (v návrhu chýba) ingredienciu. Vždy je potrebné nastaviť množstvo, jednotku a popis ingrediencie (jej názov). Po stlačení tlačidla **Next step** používateľ prejde na úpravu/zostavovanie inštrukcií v recepte. Pri **Cancel** sa najprv zobrazí upozornenie, že sa dané údaje stratia, pokiaľ bude používateľ pokračovať. Pri pokračovaní sa vracia na stránku s výberom receptu.

Pridanie inštrukcií



Na obrazovke úpravy/zostavovania inštrukcií v recepte sa používateľovi vpravo zobrazuje zoznam ingrediencií, ktoré si v predošlom kroku navolil. Inštrukcie sú zobrazené naľavo. Delené sú do osobitných blokov označených vlastným názvom. Je možné pridávať a odoberať bloky, a rovnako aj inštrukcie v rámci blokov. Každá inštrukcia obsahuje pomocný popisok, ktorý jednoduchým spôsobom objasňuje používateľovi, čo daná inštrukcia vykonáva. Po kliknutí na zelené plusko sa zobrazí obrazovka výberu inštukcie. Spolu s ukážkou identifikovaných typov inštrukcií je zobrazená nižšie.





Výber inštrukcie a rovnako aj typy inštrukcií pre zostavovanie receptu sú už implementované, ale samotné zostavovanie receptu ešte nie.

História varenia

Túto obrazovku je potrebné zostaviť.

Do budúcna

Je potrebné

- označovať jednotlivé násypníky (Akú úlohu môžu zastávať - jačmeň sa bude pridávať iba jedným z nich, a podobne.)
- definovať, odkiaľ sa bude načítavať stav zariadenia (backend/konfiguračný file). Napríklad ktoré násypníky sú dostupné a môžu sa využiť pri varení jednotlivých receptov

- šipočka ukazujúca či teplota v jednotlivých nádobách stúpa alebo klesá

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Moduly

Link na repozitár - Module

Technológie

Používame ESP32 Dev board. Kód je písaný v **C++**.

Odpurúčaný setup pre ~~develomplemt~~ development je v README v rámci git repozitára.

Komunikácia s back-endom

Pre komunikáciu využijeme websocket a správy budú vyzerať ako v Podporované údaje.

Periodický update

```
{
  "moduleId": 123,
  "status": <status modulu>, // ERROR, OK
  <podporované údaje>
}
```

Ako v skratke funguje modul?

Setup

Najprv si treba zadefinovať aké zariadenia máme kde pripojené a dať ich do poľa všetkých zariadení (v budúcnosti by sme chceli aby tento krok nebolo treba manuálne robiť pred nasadením modulu, ale dala by sa modulu poslať konfigurácia).

Kód

Modul sa pripojí na WiFi podľa konfigurácie a následne sa pripojí cez WebSocket na backendový server. Modul potom periodicky posiela svoj aktuálny stav, a aj stav všetkých pripojených zariadení.

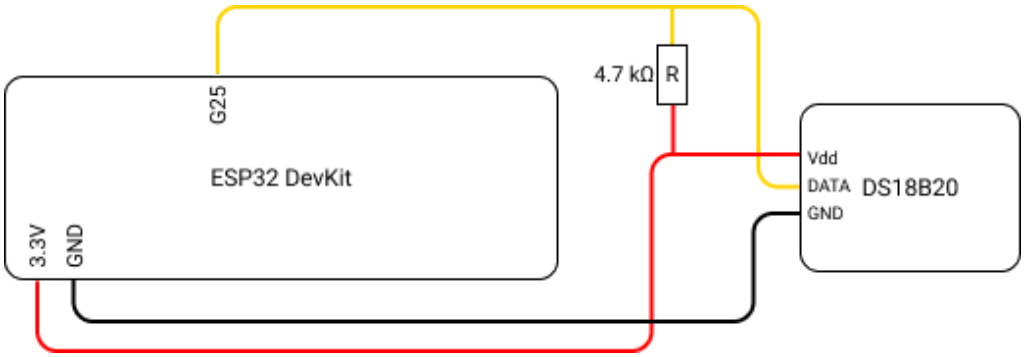
Na modul je možné poslať z backendu aj inštrukciu, ktorú má modul vykonať. Formát je zadefinovaný v Podporované funkcie. Modul následne vo svojich periodických updatoch posiela aj informáciu o aktuálnom stave danej funkcie

Diagram zapojenia

Zapojenie meniča H300

**Originálny autor diagramu H300: http://team18-19.studenti.fiit.stuba.sk/*

Zapojenie teplomera DS18B20



Notes

- pridať viac diagramov zapojenia modulu a rôznych zariadení
- automatická kontrola WiFi spojenia (ak bolo prerušené)
- dynamická konfigurácia pripojených zariadení

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Podporované údaje

Tieto `interface` predstavujú kategórie, ktoré sú využívané aj v Podporované funkcie.

Zoznam podporovaných údajov systémom:

```
interface Temperature {
  temp: number;
  regulation_enabled: boolean;
  state: string;
  device: string; // "TEMP_1" or "TEMP_2"
}

interface Motor {
  speed: number;
  rpm: number;
  state: string;
  device: string; // "MOTOR_1" or "MOTOR_2"
}

interface Unloader {
  unloaded: boolean;
  state: string;
  device: string; // "FERMENTABLE", "YEAST", "HOPS", "OTHER"
}

interface Pump {
  enabled: boolean;
  state: string;
  device: string; // "PUMP_1"
}
```

Príklad

Údaje sa budú posielať tak, že najprv sa zadefinuje do akej kategórie patrí zariadenie, potom jeho bližšie údaje. "DEVICE" bude jedinečný názov naprieč všetkými zariadeniami z danej kategórie.

Príklad posielaných údajov

```
{
  "TEMPERATURE": [
    {
      "temp": 50,
      "regulation_enabled": true,
      "state": "IN_PROGRESS",
      "device": "TEMP_1"
    },
    {
      "temp": 21.5,
      "regulation_enabled": false,
      "state": "WAITING",
      "device": "TEMP_2"
    }
  ],
  "MOTOR": [
    {
      "speed": 30,
      "rpm": 25,
      "state": "WAITING",
      "device": "MOTOR_1"
    },
    {
      "speed": 0,
      "rpm": 0,
      "state": "WAITING",
      "device": "MOTOR_2"
    }
  ],
  "UNLOADER": [
    {
      "unloaded": true,
      "state": "WAITING",
      "device": "FERMENTABLE"
    },
    {
      "unloaded": false,
      "state": "WAITING",
      "device": "YEAST"
    },
    {
      "unloaded": false,
      "state": "WAITING",
      "device": "HOPS"
    },
```

```
    {
      "unloaded": false,
      "state": "WAITING",
      "device": "OTHER"
    }
  ],
  "PUMP": [
    {
      "enabled": false,
      "state": "WAITING",
      "device": "PUMP_1"
    }
  ]
}
```

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Podporované funkcie

Zoznam podporovaných funkcií systémom:

Name	Input type	Units	Category	Code name	Type	Choices	Code choices
Temperature	field	°C	TEMPERATURE	SET_TEMPERATURE	float	Chamber 1, Chamber 2	TEMP_1, TEMP_2
Motor	field	RMP	MOTOR	SET_MOTOR_SPEED	float	Motor 1, Motor 2	MOTOR_1, MOTOR_2
Transfer liquids	executable	-	PUMP	TRANSFER_LIQUIDS	-	Pump 1	PUMP_1
Unload	executable	-	UNLOADER	UNLOAD	-	Fermentables, Yeast, Hops, Other	FERMENTABLE, YEAST, HOPS, OTHER
Wait	field	Minutes	SYSTEM	WAIT	float	-	-
Manual step	field	-	SYSTEM	MANUAL	string	-	-

- **Name** - pekný názov na Frontende
- **Code name** - ako sa bude v skutočnosti volať daná funkcia
- **Category** - je kategória, do ktorej patrí daná funkcia (využívané v Podporované údaje)
- **Input type** - či bude mať možnosť meniť paramaetre alebo sa daná funkcie iba vykonáva
- **Type** - validácia na Frontende, v skutočnosti všade posielam string
- **Choices** - aké možnosti budú ponúkané - ktorý konkrétny prvok sa má spustiť
- **Code choices** - predstavujú jednotlivé zariadenia a ich "skutočné" meno - toto meno sa bude zhodovať naprieč systémom s menom ktoré bude ukazovať stav jednotlivých senzorov

V skutočnosti Frontend ukladá a prijíma iba: **Code name**, **Code choices**, **Category** a hodnoty **parametra**. Ostatné veci si už sám vyhľadá a doplní podľa špecifikácie.

Štruktúra posielanej inštrukcie Backend --> Modul

```
{
  "moduleId": <id modulu>,
  "instruction": <code name pre inštrukciu>,
  "param": <hodnota parametru>,
  "category": <kategória inštrukcie/zariadenia>,
  "device": <zariadenie>,
}
```

Ukážka

```
{
  "moduleId": 1,
  "instruction": "SET_MOTOR_SPEED",
  "param": 30,
  "category": "MOTOR",
  "device": "MOTOR_1"
}
```

Štruktúra posielanej inštrukcie Backend --> Frontend

```
{
  "id": <id inštrukcie>,
  "recipeId": <id receptu>,
  "templateId": <id template pre inštrukciu>,
  "instruction": <code name pre inštrukciu>, // aj toto sa bude mapovať na krajší názov
  "param": <hodnota parametru>,
  "category": <kategória inštrukcie/zariadenia>,
  "device": <zariadenie>, // na FE sa bude mapovať na krajší názov - napr. "TEMP_1" ->
"Nádoba 1",
  "blockId": <id bloku>, // ak by boli dva názvy rovnaké, ale chceme to mať ako dva
rôzne bloky
  "block": <názov bloku>,
  "ordering": <poradové číslo>,
}
```

Ukážka


```
{
  "id": 234,
  "recipeId": 123,
  "templateId": 1,
  "instruction": "SET_MOTOR_SPEED",
  "param": 30,
  "category": "MOTOR",
  "device": "MOTOR_1",
  "blockId": 1,
  "block": "Fermentation",
  "ordering": 2
}
```


Štruktúra template Backend --> Frontend

FE si vie získať ako vyzerajú všetky inštrukcie, ktoré sú podporované systémom.

```
{
  "id": <id template>,
  "instruction": <code name pre inštrukciu>,
  "name": <pekny nazov instrukcie>,
  "category": <kategoria>,
  "units": <jednotka>,
  "inputType": <typ inputu>,
  "description": <popis>,
  "devices": [
    {
      "id": <id zariadenia>,
      "name": <pekny nazov zariadenia>,
      "device": <zariadenie>
    },
    ...
    ...
  ]
}
```

Ukážka

```
{
  "id": 1,
  "instruction": "SET_TEMPERATURE",
  "name": "Set temperature",
  "category": "TEMPERATURE",
  "units": "°C",
  "inputType": "float",
  "description": "Sets temerature for selected chamber",
  "devices": [
    {
      "id": 1,
      "name": "Chamber 1",
      "device": "TEMP_1"
    },
    {
      "id": 2,
      "name": "Chamber 2",
      "device": "TEMP_2"
    }
  ]
}
```

Štruktúra posielanej inštrukcie Frontend --> Backend

```
{
  "templateId": <id vzoru pre danú inštrukciu>,
  "blockId": <id bloku>,
  "param": <hodnota parametru>,
  "device": <zariadenie>,
  "ordering": <poradové číslo>,
}
```

Ukážka

```
{
  "templateId": 123,
  "blockId": 23,
  "param": 85,
```

```
  "device": "TEMP_1",
  "ordering": 1
}
```

TODO

- domysliet vytiahnutie chmeľu
- ukončenie

*Last updated on **18. 11. 2021** by **Michaela Nemcova***

Testovací server

Testovací server slúži na simuláciu správania modulov. Od vývoja modulov závisí hlavne vývoj backendu a aj frontendu, preto sme sa rozhodli vytvoriť testovací server, ktorý simuluje správanie reálneho modulu, čím napomáha k rýchlejšiemu vývoju práve backendu a frontendu.

Server posiela a prijíma údaje cez WebSocket. Posielanie údajov a prijímanie inštrukcií prebieha rovnako ako je zadefinované v API.

Server po pripojení na backend posiela periodicky namodelované dáta. Server taktiež prijíma inštrukcie z backendu, na ktoré odpovedá zmenenými dátami podľa jednotlivých inštrukcií. Zoznam podporovaných inštrukcií s konkrétnymi parametrami sa nachádza v Podporované funkcie.

V prípade zmien na backende so spusteným testovacím serverom, nie je potrebné tento testovací server spúšťať znova, automaticky sa pripojí späť na backend.

Príklad komunikácie

Server posiela údaje typu:

```
{
  "moduleId": "test-module-1",
  "state": "ok",
  "TEMPERATURE": [
    {
      "TEMP": 50,
      "REGULATION_ENABLED": true,
      "STATE": "IN_PROGRESS",
      "DEVICE": "TEMP_1"
    },
    {
      "TEMP": 21.5,
      "REGULATION_ENABLED": false,
      "STATE": "WAITING",
      "DEVICE": "TEMP_2"
    }
  ],
  "MOTOR": [
    {
      "SPEED": 30,
```

```
      "RPM": 25,
      "STATE": "WAITING",
      "DEVICE": "MOTOR_1"
    },
    {
      "SPEED": 0,
      "RPM": 0,
      "STATE": "WAITING",
      "DEVICE": "MOTOR_2"
    }
  ],
  "UNLOADER": [
    {
      "UNLOADED": true,
      "STATE": "WAITING",
      "DEVICE": "FERMENTABLE"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "YEAST"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "HOPS"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "OTHER"
    }
  ],
  "PUMP": [
    {
      "ENABLED": false,
      "STATE": "WAITING",
      "DEVICE": "PUMP_1"
    }
  ]
}
```


Príklad inštrukcie, ktorú server prijme z backendu:

```
{
  "moduleId": "test-module-1",
  "type": "instruction",
  "category": "TEMPERATURE",
  "device": "TEMP_2",
  "instruction": "SET_TEMPERATURE",
  "parameter": "100"
}
```

Na základe prijatej inštrukcie prebehne buď okamžitá alebo postupná zmena dát. Po prijatí inštrukcie vyššie môžu odosielané dáta vyzerať nasledovne:

```
{
  "moduleId": "test-module-1",
  "state": "ok",
  "TEMPERATURE": [
    {
      "TEMP": 50,
      "REGULATION_ENABLED": true,
      "STATE": "IN_PROGRESS",
      "DEVICE": "TEMP_1"
    },
    {
      "TEMP": 90,
      "REGULATION_ENABLED": true,
      "STATE": "IN_PROGRESS",
      "DEVICE": "TEMP_2"
    }
  ],
  "MOTOR": [
    {
      "SPEED": 30,
      "RPM": 25,
      "STATE": "WAITING",
      "DEVICE": "MOTOR_1"
    },
    {
      "SPEED": 0,
      "RPM": 0,
      "STATE": "WAITING",
      "DEVICE": "MOTOR_2"
    }
  ],
```

```
  "UNLOADER": [
    {
      "UNLOADED": true,
      "STATE": "WAITING",
      "DEVICE": "FERMENTABLE"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "YEAST"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "HOPS"
    },
    {
      "UNLOADED": false,
      "STATE": "WAITING",
      "DEVICE": "OTHER"
    }
  ],
  "PUMP": [
    {
      "ENABLED": false,
      "STATE": "WAITING",
      "DEVICE": "PUMP_1"
    }
  ]
}
```


*Last updated on **18. 11. 2021** by **Michaela Nemcova***