Technická dokumentácia

REZERVAČNÝ SYSTÉM PRE FITNESS CENTRUM

DENIS ČAPKOVIČ, RICHARD DOMINIK, NICOLAS ORSÁG, ŠIMON BABÁL

Obsah

1. Katalóg požiadaviek

1.1 Úvod

1.1.1 Účel dokumentu

Tento dokument slúži na opísanie požiadaviek ku projektu Rezervačný systém pre fitness centrum. Obsahuje požiadavky zadávateľa, školiteľa a je záväzný pre zadávateľa a realizátorov projektu. Použije sa na vyhodnotenie správnosti implementácie.

1.1.2 Rozsah využitia systému

Účelom aplikácie je riešiť rezerváciu klientov fitness centra, pričom aplikácia má ponúkať možnosť ukladania rozličných informácii o klientoch, strojoch a procedúrach vo fitness centre. Aplikáciu bude určená len pre zamestnancov fitness centra, aby za jej pomoci mohli ponúkať služby fitness centra efektívnejšie.

1.1.3 Slovník pojmov

- okno času = 45 minút
- aktivita = stroj alebo procedúra na ktorú sa da objednať

1.1.4 Odkazy a referencie

Dokumentácia k React

https://reactjs.org/docs/getting-started.html

Dokumentácia k Laravel

https://laravel.com/docs/6.x

1.1.5 Prehľad nasledujúcich kapitol

V ďalších kapitolách sa čitateľ môže dozvedieť o tom, pre koho je aplikácia určená, konkrétnych atomických požiadavkách, ktoré bude spĺňať finálna verzia aplikácie.

1.2 Všeobecný popis

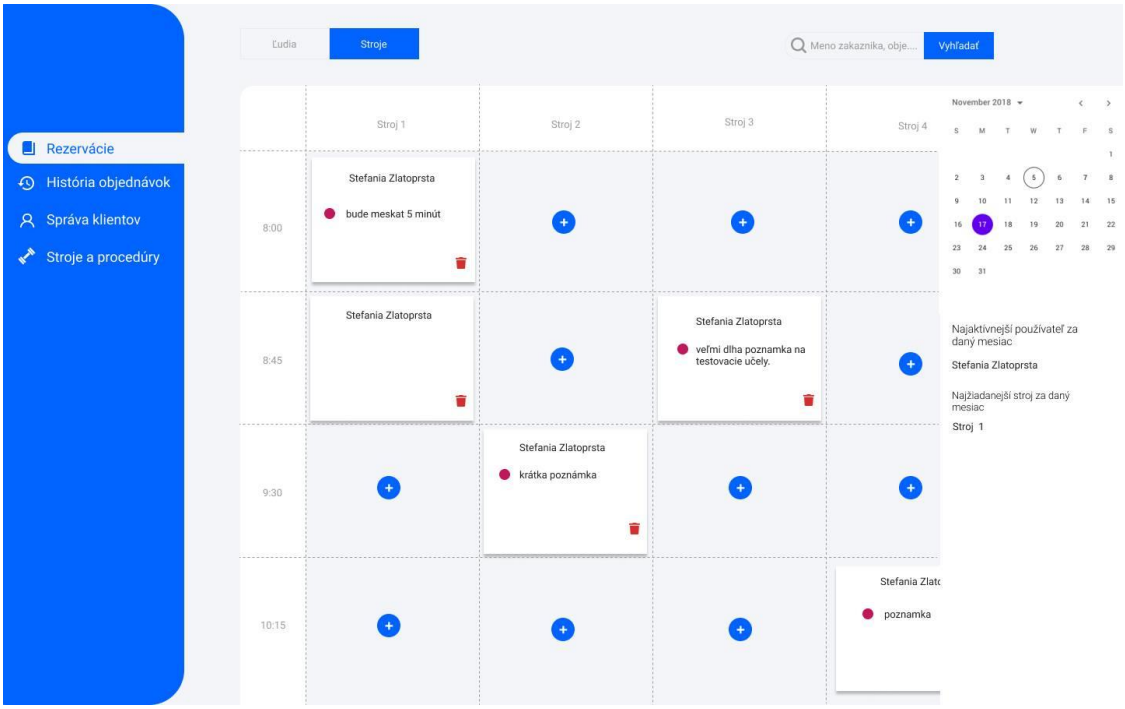
1.2.1 Perspektíva systému

Aplikácia bude slúžiť hlavne na objednávku klientov fitness centra na rôzne stroje a procedúry. Okrem tejto funkcionality bude ukladať informácie o klientoch a o procedúrach. Používateľ aplikácie bude mať možnosť tieto informácie aj meniť. Zároveň bude aplikácia ponúkať veľmi základné štatistiky o návštevnosti. Aplikácia bude slúžiť len pre jedného užívateľa, čiže administrátora, ale zdrojový kód bude písaný takým spôsobom, aby bolo možné dotvoriť samoobslužné objednávanie pre klientov, ak niekedy v budúcnosti vznikne takáto potreba.

1.2.2 Funkcie systému

Aplikácia sa bude pripájať k databáze, z ktorej bude získavať informácie o klientoch, strojoch, objednávkach a bude túto databázu aktualizovať. Pri aktualizáciách bude upravovať informácie o

klientoch, ktorých bude môcť pridávať a odstraňovať a rovnako bude môcť pracovať aj s informáciami o strojoch a objednávkach. Prístup k jednotlivým funkciám je znázornený na nasledovnom obrázku.



Obrázok 1: Ukážka hlavnej obrazovky informačného dochádzkového systému

1.2.3 Charakteristika používateľa

V aplikácii bude len jeden typ používateľa. Tento používateľ teda bude môcť využívať všetky funkcie aplikácie. Aplikácia je určená pre zamestnanca fitness centra, ktorý eviduje objednávky zákazníkov a vyťaženosť fitness centra.

1.2.4 Všeobecné obmedzenia

Aplikáciu bude možné spúšťať len cez webový prehliadač. Podmienky pre minimálne verzie prehliadačov sú nasledovné:

- Google Chrome 49+
- Mozilla Firefox 46+
- Microsoft Edge 13+
- Apple Safari 8+
- Opera 27+

1.2.5 Predpoklady a závislosti

Systém bude závislý na databáze, z ktorej bude získavať údaje o klientoch, strojoch, procedúrach a objednávkach.

1.3 Špecifické požiadavky

1.3.1 Funkčné požiadavky

R01 Možnosť vyhľadávať v klientoch

Aplikácia umožňuje rýchle fulltextové vyhľadávanie klientov v databáze podľa mena, priezviska alebo telefónneho čísla. Klienta je možné vyhľadať pomocou ktoréhokoľvek údaju a je možné vyhľadávať aj pomocou viacerých údajov naraz. Klienta je možné vyhľadávať pri vytváraní rezervácii, alebo pri administrácii klientov.

R02 Pridávanie klientov

Aplikácia umožňuje pridávanie klientov, ktorí sa budú môcť následne dať nastaviť ako objednávatelia aktivít a vyhľadávať v systéme.

R03 Odstraňovanie klientov

Aplikácia umožňuje odstránenie klienta z databázy. Po odstránení klienta daný klient už nie je ďalej evidovaní prostredníctvom aplikácie a nie je možné ho v aplikácii vyhľadať.

R04 Príznak o aktívnosti klienta

Aplikácia umožňuje nastaviť klientovi príznak, či je aktívny alebo neaktívny. Pri vyhľadávaní klientov sa najskôr zobrazujú klienti, ktorý majú príznak, že sú aktívny.

R05 Príznak o podpísaní GDPR

Aplikácia umožňuje nastaviť klientovi príznak, či podpísal alebo nepodpísal GDPR.

R06 Príznak o rekreačnej karte

Aplikácia umožňuje nastaviť klientovi príznak, či využíva rekreačnú kartu.

R07 Editovanie klientov

Aplikácia umožňuje zmenu údajov o klientoch, čiže meno a priezvisko, telefónne číslo, príznak o podpísaní GDPR a vlastnení športovej kartičky a úpravu poznámky o klientovi.

R08 Možnosť rezervácie aktivít pre klienta v danom čase

Aplikácia umožňuje vytvoriť rezerváciu pre daného klienta na dané časové okno a na danú aktivitu. Každej rezervácii prislúcha práve jedno okno času.

R09 Pridávanie poznámok do objednávky aktivity

Každá objednávka má klienta, aktivitu a ešte je k nej možné pridať poznámku.

R10 Rýchly prístup k dnešnému dňu v kalendári

Aplikácia umožňuje rýchly prístup k dnešnému dňu na kalendári, aj keď sa administrátor nachádza ľubovoľnej časti kalendára.

R11 Editovanie objednávok aktivít

Aplikácia umožňuje zmeniť klienta, pridať poznámku alebo zmeniť aktivitu na dané okno.

R12 Odstraňovanie objednávok aktivít

Aplikácia umožňuje odstránenie objednávky na danú aktivitu.

R13 Prezeranie histórie objednávok pre klientov

Aplikácia umožňuje prezeranie histórie objednávok klientov.

R14 Prezeranie histórie objednávok pre aktivity

Aplikácia umožňuje prezeranie histórie objednávok pre aktivity.

R15 Pridanie stroja/procedúry

Aplikácia umožňuje pridanie ďalšieho stroja/procedúry do databázy.

R16 Odstránenie stroja/procedúry

Aplikácia umožňuje odstránenie stroja/procedúry z databázy. Po zmazaní stroja/procedúry z databázy sa informácie o objednávkach na tento stroj/procedúru ponechajú v databáze, ale vo webovej aplikácii už nebude možný prístup k týmto informáciám.

R17 Prihlásenie do aplikácie

Aplikácia umožňuje prihlásenie používateľa. Bez prihlásenia nie je možný prístup ku žiadnym informáciám o klientoch. Používateľ sa prihlasuje pomocou svojho používateľského mena a hesla.

R18 Logovanie

Aplikácia loguje udalosti, počas ktorých používateľ mení obsah tabuliek v databáze.

R19 Prezeranie obsadenosti

Aplikácia umožňuje prezeranie obsadenosti časových okien na jeden týždeň. Túto funkcionalitu poskytuje aplikácia pre klientov a nie je potrebné sa kvôli nej prihlasovať.

1.3.2 Požiadavky nevzťahujúce sa na funkcionalitu

Systém bude dodaný ako webová aplikácia, ktorá sa bude spúšťať cez jeden z podporovaných webových prehliadačov, ktoré sú vyššie uvedené. Bude ju možne spustiť na rôznych platformách od mobilných telefónov po stolové počítače, čiže aplikácia bude plne responzívna.

1.3.3 Požiadavky rozhrania

Aplikácia bude interagovať s používateľom cez webový prehliadač.

2. Návrh

2.1 Úvod

2.1.1 Účel dokumentu

Tento dokument slúži ako návrh rezervačného a objednávacieho systému pre fitness centrum a je určený pre vývojárov systému. Dokument dôkladne popisuje funkcie a metódy informačného systému a podáva návrh na implementáciu.

2.2 Podrobná špecifikácia vonkajších interfejsov

Aplikácia bude bežať na serveri a komunikovať s PostgreSQL databázovým serverom, kde bude uložený celý obsah aplikácie.

2.3 Používané technológie

2.3.1 PHP/PostgreSQL

V PostgreSQL databáze bude uložený celý obsah aplikácie, čo sú všetky data aplikácie. PHP bude komunikovať s PostgreSQL a pracovať s dátami v databáze.

2.3.2 Laravel

Laravel je open source PHP framework pre webové aplikácie. Aplikácia bude postavená na tomto frameworku. Bude slúžit na vytvorenie backendu celej aplikácie.

2.3.3 Javascript

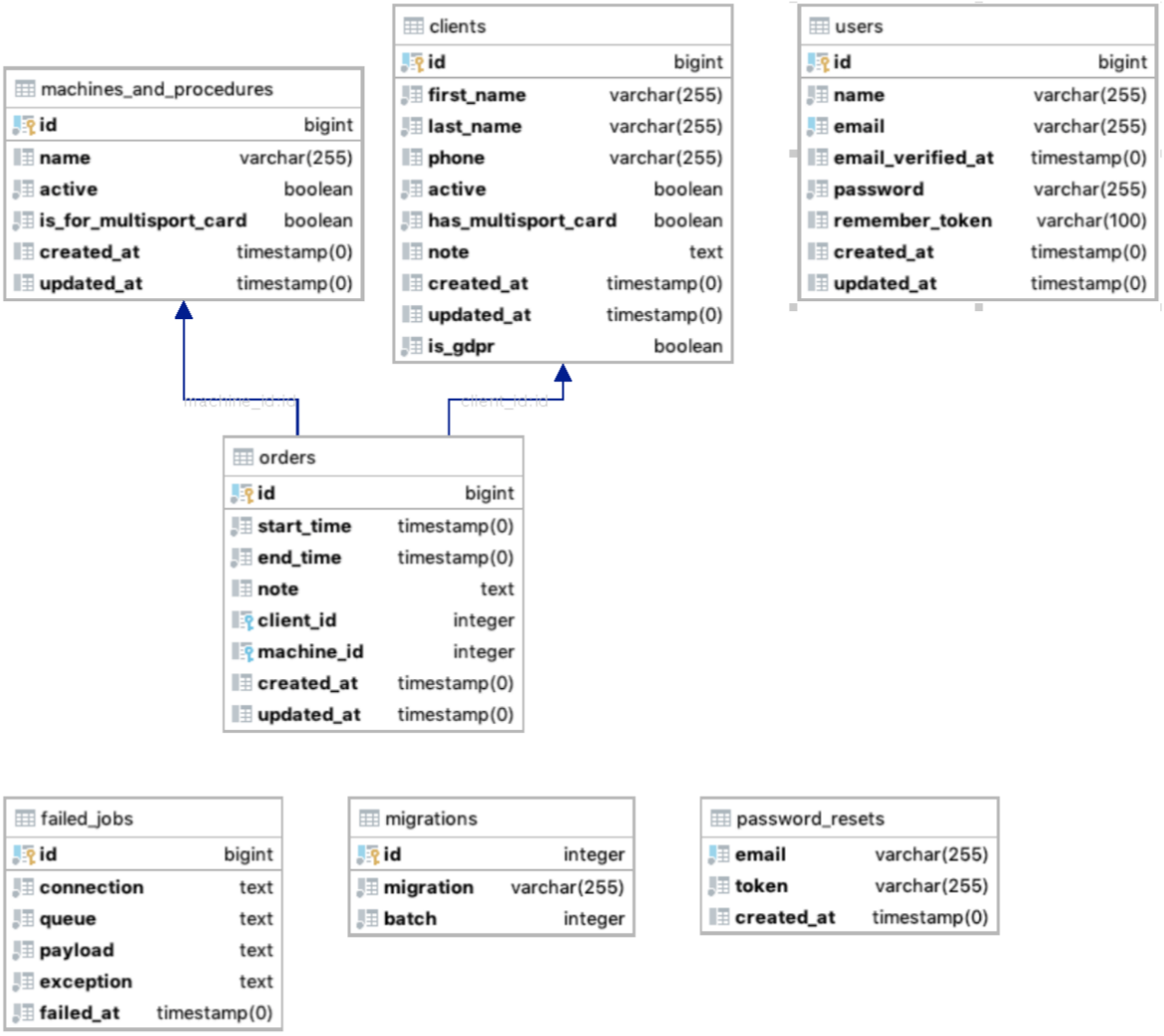
Táto technológia umožnuje dynamicky aktualizovať obsah aplikácie na strane klienta.

2.3.4 React

Knižnica javascriptu ktorá nám bude slúžiť na vytvorenie frontendu celej aplikácie

2.4 Podrobný dátový model perzistentých údajov atď.

2.4.1 Databázový model



Obrázok 2: Dátový model dochádzkového systému

Machines_and_procedures:

Táto tabuľka obsahuje dáta o aktivitách pre klientov. Obsahuje ich názov a binárne hodnoty, či sú aktívne a či je možné na ne použiť multisport kartičku.

Clients:

Táto tabuľka obsahuje dáta o klientoch. V tabuľke sa nachádzajú meno, priezvisko, telefónne číslo, poznámka a binárne hodnoty, či klient podpísal GDPR, či je aktívny a či má multisport kartičku.

Orders:

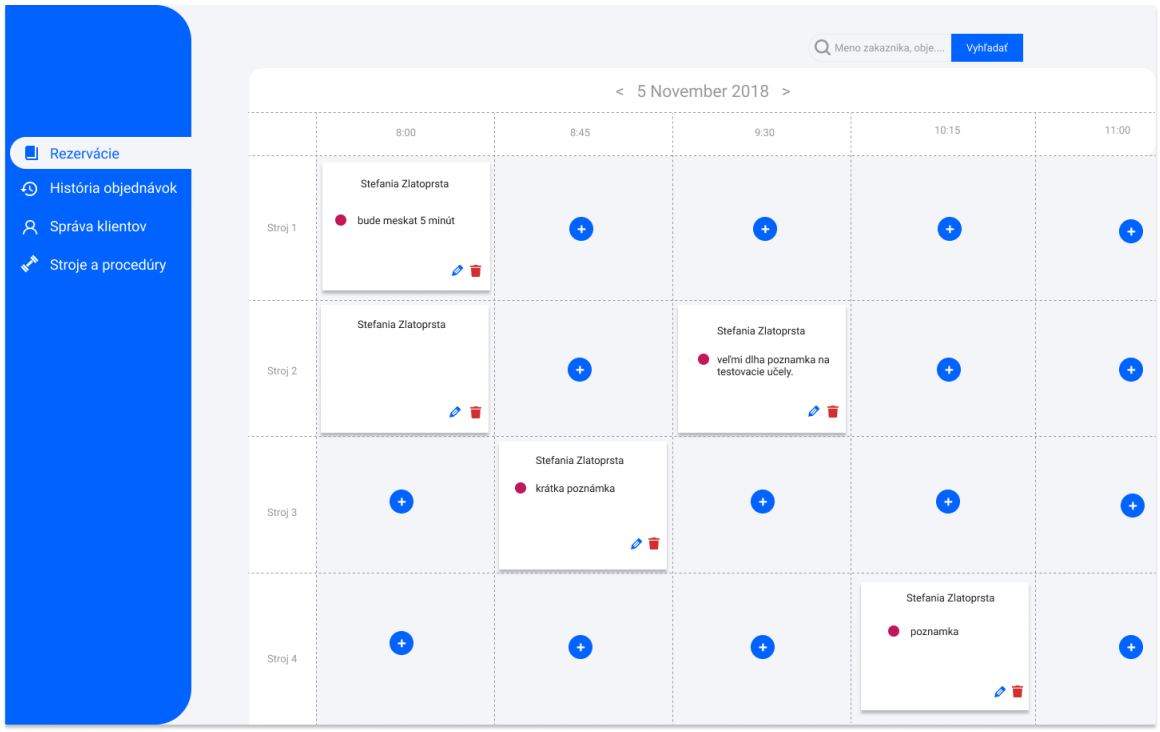
Táto tabuľka obsahuje dáta o jednotlivých rezerváciach na dané časové okná, čiže začiatok a koniec, poznámka a identifikátory klientov a aktivít.

Users:

Táto tabuľka obsahuje dáta o používateľoch webovej aplikácie, čiže ich mail, meno, kedy bol email verifikovaný, heslo a kedy bol účet pre používateľa vytvorený a kedy bol naposledy upravovaný.

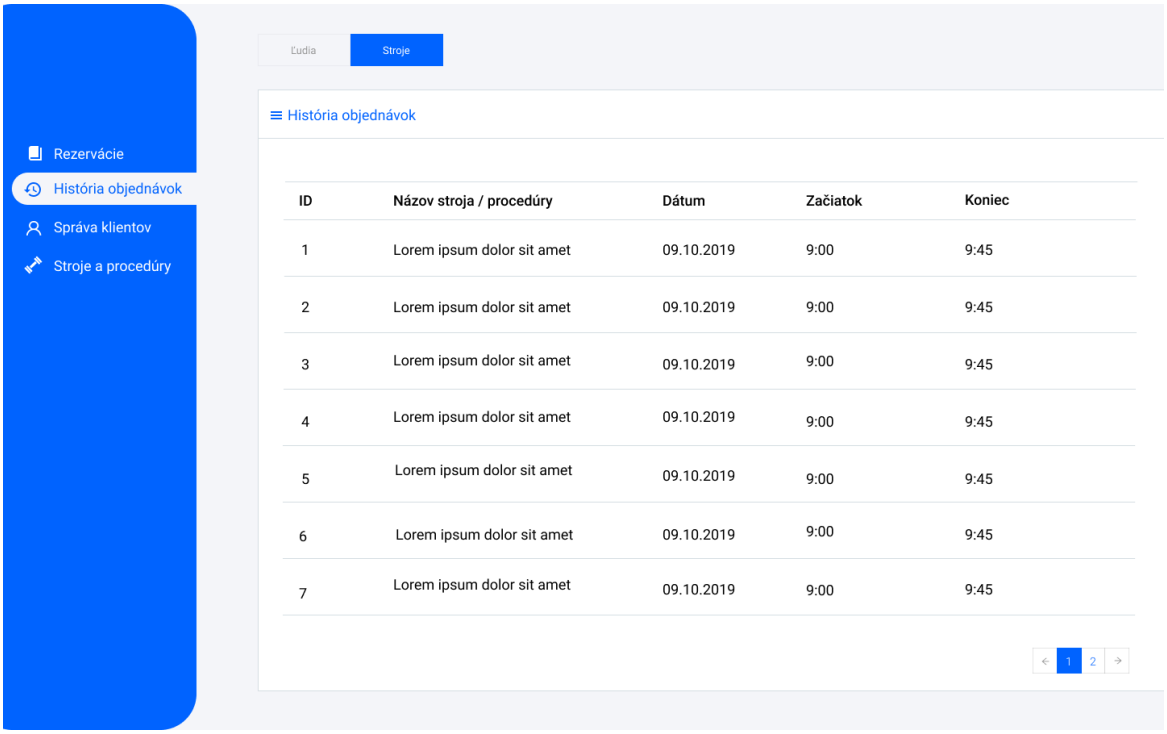
2.5.Návrh používateľského rozhrania

2.5.1 Home screen :



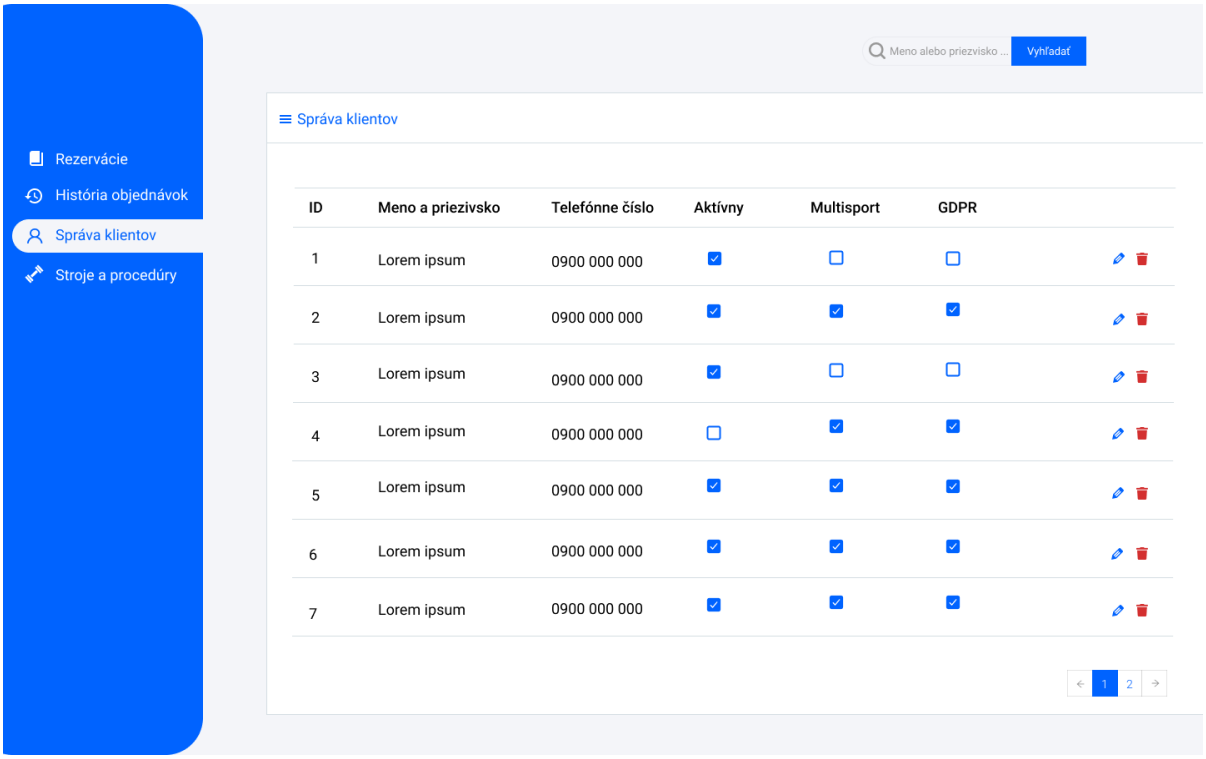
Obrázok 3: Návrh objednávacej obrazovky dochádzkového systému

2.5.2 História objednávok:



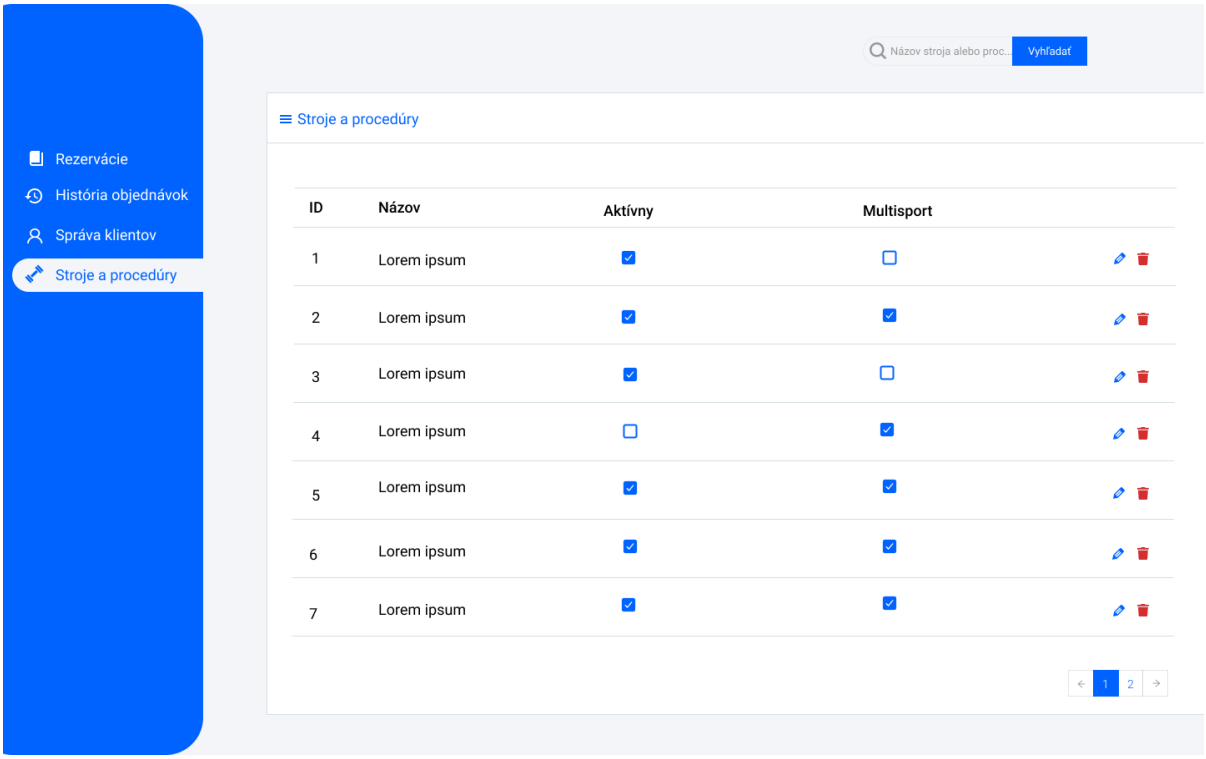
Obrázok 4: Návrh obrazovky histórie objednávok dochádzkového systému

2.5.3 Správa klientov:



Obrázok 5: Návrh obrazovky správy klientov

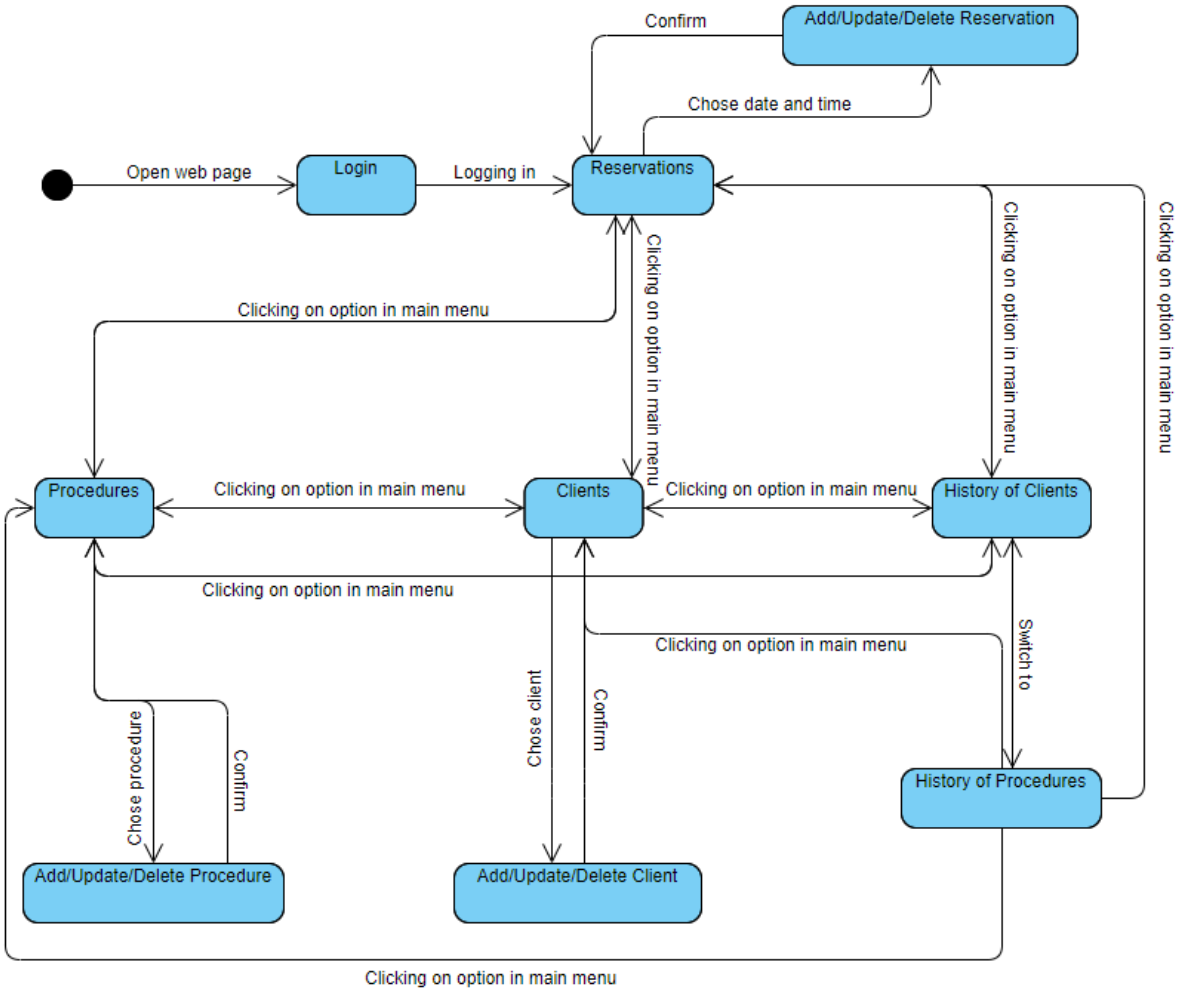
2.5.4 Stroje a procedúry:



Obrázok 6: Návrh obrazovky správy strojov a procedúr

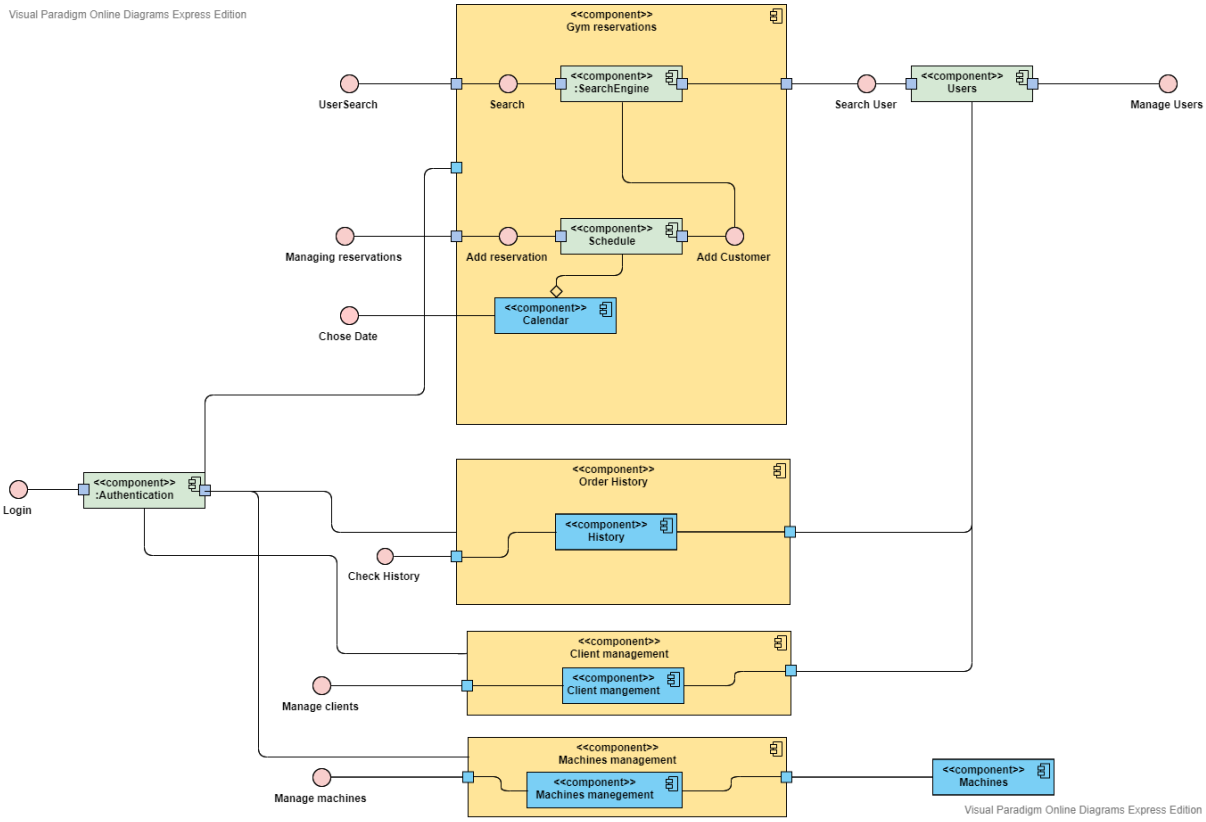
2.6.Návrh implementácie

2.6.1 UML – State diagram používateľského rozhrania



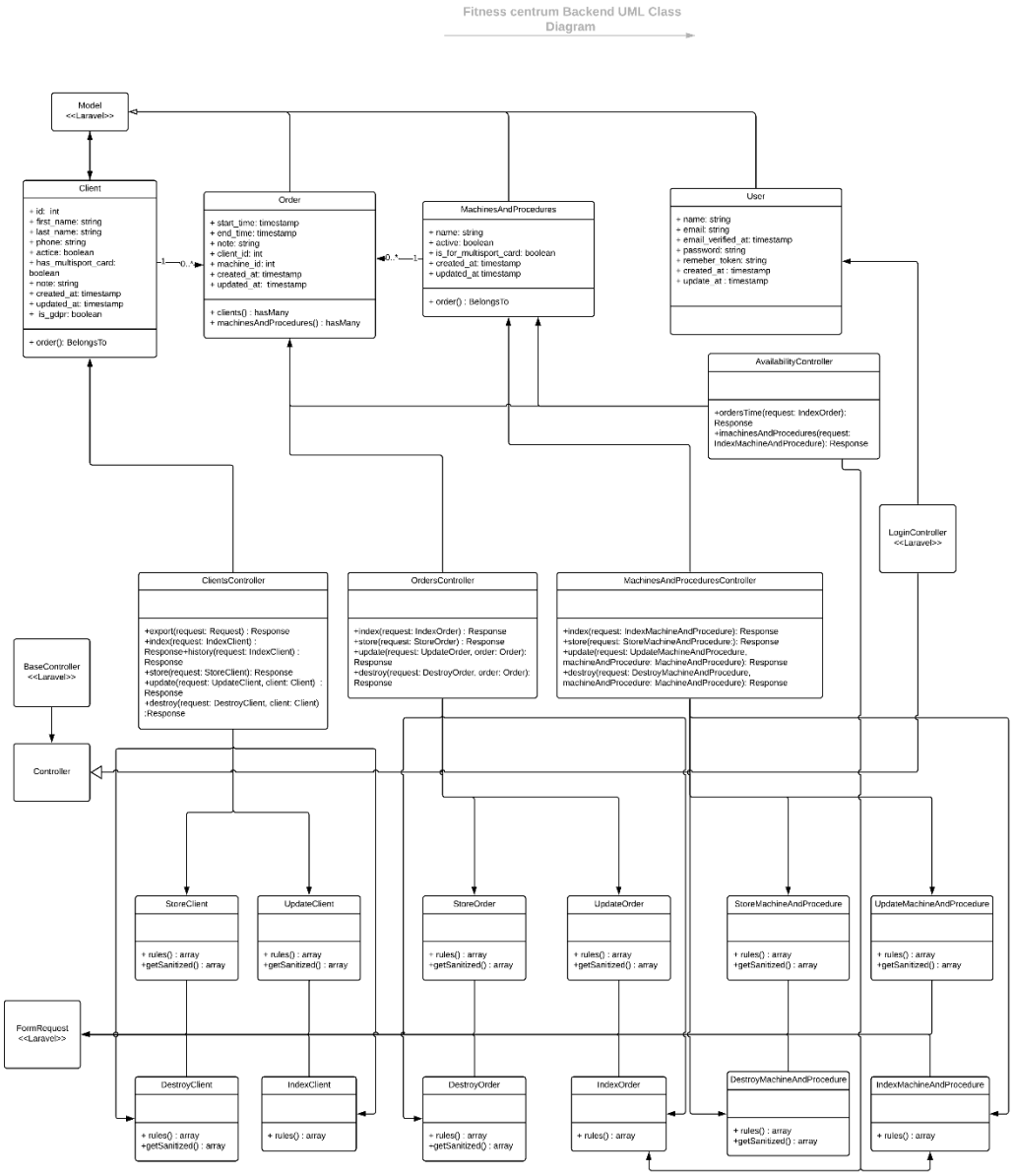
Obrázok 7: Stavový diagram pre používateľské rozhranie

2.6.2 Diagram komponentov



Obrázok 8: Diagram komponentov dochádzkového systému

2.6.3 Triedny diagram – backend



Obrázok 9: Triedny diagram dochádzkového systému pre backend

Models:

Modely slúžia na reprezentáciu jednotlivých entít, čiže klientov, objednávky, aktivity a používateľov. Jednotlivé modely obsahujú totožné atribúty, aké sú obsiahnuté pri tabuľkách v databáze.

Controllers:

Kontrolery slúžia na administráciu dát. Parametrom každého kontrolera býva request. Metóda index sa používa na vrátenie všetkých dát danej entity, metóda store ukladá jednu položku, metóda update položky

aktualizuje a metóda destroy zmaže položku z databázy. Metóda export vyexportuje dáta z databázy do súboru vo formáte excel na robenie štatistík a analyzovanie jednotlivých rezervácií. Všetky metódy kontrolerov okrem export vracajú ako výstup response vo formáte JSON.

Requests:

Requesty slúžia ako parameter pre kontrolery a používajú http metódy GET, POST a DELETE. Index requesty využívajú metódu GET a ich parametrami sú informácie pre stránkovanie, čiže číslo strany, koľko položiek sa má zobraziť na jednu stranu a podľa čoho majú prísť dáta usporiadané. Store a update requesty využívajú metódy POST a ako parametre sú atribúty jednotlivých entít. Destroy requesty využívajú metódu DELETE, pričom ako parameter obsahujú ID položky, ktorá sa má vymazať z databázy. Na obrázkoch nižšie sú ukázané parametre jednotlivých requestov.

```
'orderBy' => ['in:id', 'nullable'],
'orderDirection' => ['in:asc,desc', 'nullable'],
'page' => ['integer', 'nullable'],
'perPage' => ['integer', 'nullable'],
```

Obrázok 10: Parametre requestu pre zoznam aktivít

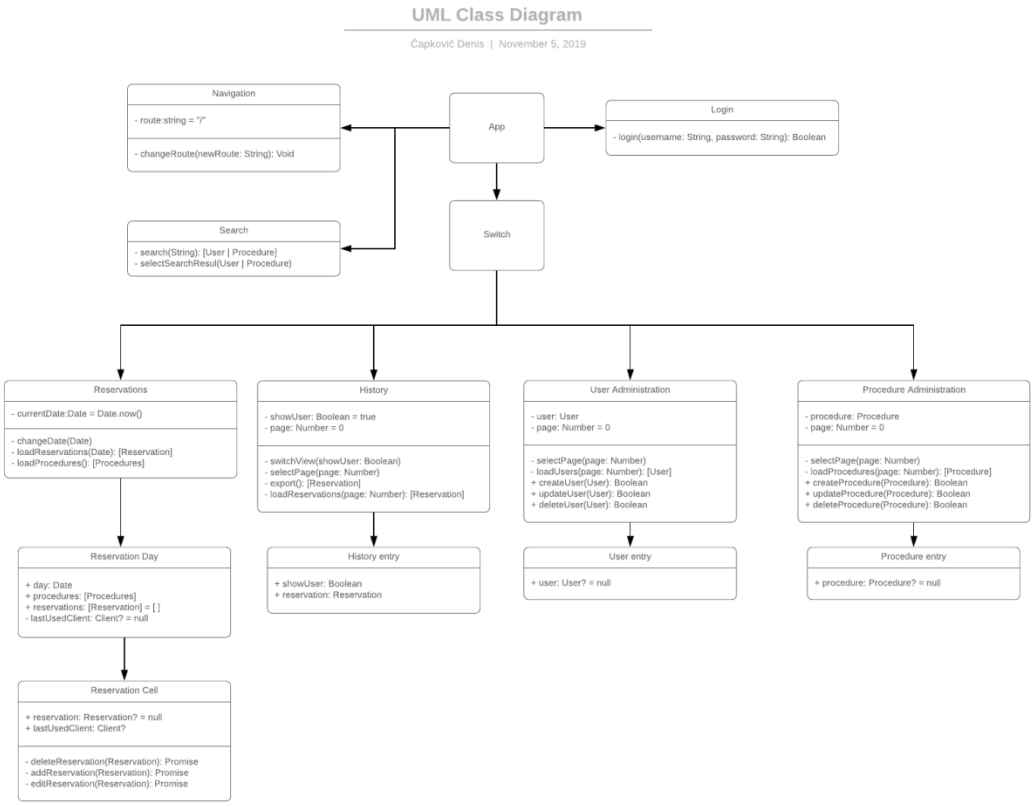
```
'name' => ['required', 'string'],
'active' => ['required', 'boolean'],
'is_for_multisport_card' => ['required', 'boolean'],
```

Obrázok 11: Parametre requestu pre uloženie activity

```
'name' => ['sometimes', 'string'],
'active' => ['sometimes', 'boolean'],
'is_for_multisport_card' => ['sometimes', 'boolean'],
```

Obrázok 12: Parametre requestu pre aktualizovanie aktivity

2.6.4 Triedny diagram – frontend



Obrázok 13: Triedny diagram dochádzkového systému pre frontend

2.6.5 Komunikácia medzi frontend a backend

Frontend a backend budú spolu komunikovať cez http requesty. Tieto requesty budú používať metódy GET, POST a DELETE. Metóda GET sa bude používať pre zisťovanie informácii z databázy, pomocou metódy POST budeme vedieť tieto informácie upravovať, alebo pridávať ďalšie a pomocou metódy DELETE sa budú môcť tieto informácie mazať.

Domain	Method	URI	Name	Action	Middleware
	GET|HEAD	api/v1/clients		App\Http\Controllers\Api\v1\ClientsController@index	api,jwt.auth
	POST	api/v1/clients		App\Http\Controllers\Api\v1\ClientsController@store	api,jwt.auth
	GET|HEAD	api/v1/clients/history		App\Http\Controllers\Api\v1\ClientsController@history	api,jwt.auth
	GET|HEAD	api/v1/clients/history/export		App\Http\Controllers\Api\v1\ClientsController@export	api,jwt.auth
	POST	api/v1/clients/{clientId}	update	App\Http\Controllers\Api\v1\ClientsController@update	api,jwt.auth
	DELETE	api/v1/clients/{clientId}	destroy	App\Http\Controllers\Api\v1\ClientsController@destroy	api,jwt.auth
	GET|HEAD	api/v1/machines-and-procedures		App\Http\Controllers\Api\v1\MachinesAndProceduresController@index	api,jwt.auth
	POST	api/v1/machines-and-procedures		App\Http\Controllers\Api\v1\MachinesAndProceduresController@store	api,jwt.auth
	GET|HEAD	api/v1/machines-and-procedures-public		App\Http\Controllers\Api\v1\AvailabilityController@machinesAndProcedures	api
	DELETE	api/v1/machines-and-procedures/{machinesAndProcedureId}	destroy	App\Http\Controllers\Api\v1\MachinesAndProceduresController@destroy	api,jwt.auth
	POST	api/v1/machines-and-procedures/{machinesAndProcedureId}	update	App\Http\Controllers\Api\v1\MachinesAndProceduresController@update	api,jwt.auth
	POST	api/v1/orders		App\Http\Controllers\Api\v1\OrdersController@store	api,jwt.auth
	GET|HEAD	api/v1/orders		App\Http\Controllers\Api\v1\OrdersController@index	api,jwt.auth
	GET|HEAD	api/v1/orders-time-public		App\Http\Controllers\Api\v1\AvailabilityController@ordersTime	api
	POST	api/v1/orders/{orderId}	update	App\Http\Controllers\Api\v1\OrdersController@update	api,jwt.auth
	DELETE	api/v1/orders/{orderId}	destroy	App\Http\Controllers\Api\v1\OrdersController@destroy	api,jwt.auth
	POST	api/v1/user/login		App\Http\Controllers\Api\v1\UserController@login	api
	POST	api/v1/user/logout		App\Http\Controllers\Api\v1\UserController@logout	api
	POST	api/v1/user/refresh		App\Http\Controllers\Api\v1\UserController@refresh	api
	POST	api/v1/user/register		App\Http\Controllers\Api\v1\UserController@register	api

Obrázok 14: Zoznam ciest, cez ktoré komunikuje frontend s backendom

Jednotlivé parametre requestov sú bližšie popísané pri triednom diagrame backendu.

2.7 Návod na inštaláciu

2.7.1 Prerekvizity - backend

Fail2Ban

Pre ochranu pred hackerskými útokmi je dobré mať nainštalované Fail2Ban. Táto aplikácia na základe logov určuje, z ktorej IP adresy pravdepodobne prebiehajú nejaké útoky. Nainštalovať ju vieme nasledovným príkazom:

sudo apt-get install -y fail2ban

Webový server Nginx

Ďalej budeme potrebovať nejaký webový server, napr. Nginx, ktorý nainštalujeme nasledovnými príkazmi:

sudo apt-get install -y software-properties-common

sudo add-apt-repository ppa:nginx/stable

sudo apt-get update

sudo apt-get install -y nginx

sudo service nginx restart

Git, Curl, Wget

Ďalej budeme potrebovaŤ nástroje Git, Curl, Wget, ktoré nainštalujeme nasledovným príkazom:

sudo apt-get install -y git curl wget

Repozitáre pre PHP a následná inštalácia

Budeme potrebovať pridať repozitáre pre PHP 7:

sudo apt-get install -y software-properties-common

sudo apt-get install -y python-software-properties

sudo add-apt-repository ppa:ondrej/php

sudo apt-get update

Následne treba PHP 7 nainštalovať:

sudo apt-get install -y php7.2 php7.2-fpm php-mysql php7.2-mysql php-mbstring php-gettext libapache2-mod-php7.2 php-doctrine-dbal php7.2-pgsql php-xml redis-server

sudo systemctl restart php7.2-fpm

Composer

Teraz potrebujeme nainštalovať Composer, ktorý nám neskôr pomôže doinštalovať rôzne závislosti pre našu webovú aplikáciu:

curl -sS https://getcomposer.org/installer | sudo php -- --install-dir=/usr/local/bin --filename=composer

Databáza PostgreSQL

Ďalej nainštalujeme databázu PostreSQL:

sudo apt-get install -y postgresql postgresql-contrib

2.7.2 Kroky k inštalácii – backend

Treba ísť do priečinka, v ktorom budeme mať aplikáciu(ak má server Nginx pôvodnú konfiguráciu, tak to bude priečinom /var/www/), a do príkazového riadka zadať príkaz na naklonovanie repozitára, ktorý obsahuje aplikáciu:

git clone git@github.com:TIS2019-FMFI/fitness.git

Následne treba nakonfigurovať server Nginx v konfiguračnom súbore:

sudo nano /etc/nginx/sites-available/default

V tomto súbore medzi „server {„ a koncovú zátvorku budeme vkladať nasledovné reťazce:

Pre zakázanie prístupu k súborom, ktoré začínajú bodkou vložíme:

location ~ /\.

{

deny all;

}

Pridanie PHP interpretera, pričom 7.0 nahradíme verziou PHP, ktorú sme pred tým nainštalovali:

location ~ \.php$ {

include snippits/fastcgi-php.conf;

fastcgi_pass unis:/run/php/php.7.0-fpm.sock;

}

Následne v súbore nájdeme riadok:

root /var/www/html;

a nahradíme ho nasledujúcim riadkom:

root /var/www/fitness/src/backend/public;

Teraz po upravení súboru treba reštartovať server týmto príkazom:

sudo service nginx restart

Teraz nastavíme povolenia pre súbory, ktoré budú používané:

sudo chown -R www-data:www-data /var/www/ fitness/src/backend /public

sudo chmod 755 /var/www

sudo chmod -R 755 /var/www/ fitness/src/backend /bootstrap/cache

sudo chmod -R 755 /var/www/ fitness/src/backend /storage

Teraz ostáva už len nainštalovať závislosti pomocou Composer. Prejdeme do priečinku /var/www/ fitness/src/backend a zadáme príkaz:

composer install

2.7.3 Prerekvizity – frontend

Node, NPM

Budeme potrebovať nainštalovať Node server a NPM:

curl -sL https://deb.nodesource.com/setup | sudo bash -

sudo apt-get install -y npm

#Gulp, Grunt and Yarn

ln -s /usr/bin/nodejs /usr/bin/node

sudo npm install --global yarn gulp-cli grunt-cli

2.7.4 Kroky k inštalácií

Na nainštalovanie treba ísť do priečinku /var/www/fitness/src/frontend za zadať tam príkazy na doinštalovanie závislostí a vytvorenie produkčného buildu:

npm install

npm run production //Your command to compile frontend javascript

2.7.5 Nastavenie databázy

Najskôr treba vytvoriť globálneho používateľa postgres:

sudo su - postgres

psql

Následne treba vytvoriť databázu pre webovú aplikáciu:

CREATE DATABASE appname_db;

Následne vytvoriť používateľa databázy:

CREATE USER appname_user WITH PASSWORD 'password_of_your_choosing';

Teraz je potrebné pridať vytvorenému používateľovi potrebné práva. Následne môžeme odísť z prostredia konfigurovania databázy:

GRANT ALL PRIVILEGES ON DATABASE appname_db TO appname_user;

\q

exit

2.7.6 Nastavenie prostredia

Backend

Prostredie sa nastavuje v subore .end, pričom v súbore .env.example je príklad, ako má tento súbor vyzerať. Súbor .env vytvoríme teda zo súboru .env.example:

cd /var/www/fitness/src/backend

cp .env.example .env

nano .env

Zmeníme riadky:

APP_ENV=local

APP_DEBUG=true

na

APP_ENV=production

APP_DEBUG=false

Následne je treba v súbore .end ešte nastaviť meno databázy, meno používateľa a heslo, ktoré sme vytvorili pri nastavovaní databázy.

Teraz treba spustiť niekoľko nasledovných príkazov na dokončenie nastavenia prostredia a pripojenia databázy:

php artisan key:generate

php artisan migrate:install

php artisan migrate

php artisan jwt:secret

Frontend

Treba ísť do priečinku /var/www/fitness/src/frontend/src a v súbore App.tsx treba nájsť riadok:

export const url = 'http://localhost';

a http://localhost treba nahradiť URL, na ktorej sa nachádza backend.

2.7.7 Šifrovanie SSL

Pre šifrovanie SSL treba stiahnuť Certbot:

sudo apt-get install -y software-properties-common

sudo add-apt-repository ppa:certbot/certbot

sudo apt-get update

sudo apt-get install -y python-certbot-nginx

Upravenie existujúceho virtualhostu:

sudo nano /etc/nginx/sites-available/default

zmena riadka:

server_name _;

na

server_name example.com www.example.com;

a zmena riadkov:

listen 80 default_server;

listen [::]:80 default_server;

na

listen 80 default_server;

listen [::]:80;

Kopírovanie súboru, ktorý sme upravili do sites-enabled:

sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/example.com

sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/

Inštalácia SSL certifikátu poskytovaného LetsEncrypt:

sudo certbot --nginx

3. Testovacie scenáre

3.1. Prihlásenie

Akcia: Používateľ pri vstupe na stránku zadá svoje prihlasovacie meno a heslo.

Reakcia: Skontroluje sa, či daný používateľ existuje. Ak existuje, overí sa, či bolo zadané správne heslo. Ak bolo zadané správne heslo, zobrazí sa stránka objednávania klientov. Pri nesprávnom hesle sa zobrazí oznámenie o tom, že heslo a meno používateľa nie sú správne.

3.2. Vyhľadanie klienta - vytváranie objednávky

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ na stránke vytvárania objednávok zadá informáciu o klientovi(meno/priezvisko/telefón) a klikne na klienta, s ktorým chce pracovať.

Reakcia: Klient sa uloží do premennej(frontend) a je s ním možné ďalej pracovať.

3.3. Pridanie klienta

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ na stránke administrácie klientov vyplní informácie o klientovi zvolí pridať klienta.

Reakcia: Klient sa uloží do databázy a pri kroku 2 a 3 ho je možné vyhľadať.

3.4. Odstránenie klienta

Kroky ktoré sa vykonajú predtým: 1, 3

Akcia: Používateľ zvolí odstrániť klienta.

Reakcia: Klient sa odstráni z databázy a pri kroku 2 a 3 ho nie je možné vyhľadať.

3.5. Príznaky o aktívnosti klienta, GDPR a rekreačnej karty

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ zmení stav príznakov o aktívnosti klienta alebo GDPR alebo rekreačnej karty buď pri pridaní klienta alebo pri aktualizovaní údajov o klientovi.

Reakcia: Klient má aktualizované informácie a je to možné overiť cez krok 3.

3.6. Editovanie klientov

Kroky ktoré sa vykonajú predtým: 1, 3

Akcia: Používateľ zmení údaje o klientovi a zvolí aktualizovať údaje o klientovi.

Reakcia: Klient má aktualizované informácie a je to možné overiť cez krok 3.

3.7. Vytvorenie rezervácie

Kroky ktoré sa vykonajú predtým: 1, 2

Akcia: Používateľ vyberie časové okno a vyplní dodatočné údaje k rezervácií. Následne potvrdí vytvorenie rezervácie.

Reakcia: Klient má vytvorenú rezerváciu. Táto rezervácia sa zobrazuje po prihlásení v danom časovom okne.

3.8. Prístup k dnešnému dňu v kalendári

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ prejde na vzdialený dátum niekedy v budúcnosti. Následne klikne na presunutie sa k dnešnému dňu.

Reakcia: Klient je presmerovaný v kalendári na dnešný deň.

3.9. Editovanie objednávky

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ vyberie časové okno, na ktoré je vytvorená rezervácia a upraví informácie o rezervácií. Následne zmenu potvrdí.

Reakcia: Rezervácia je zmenená. Táto zmena sa zobrazuje po prihlásení v danom časovom okne.

3.10. Odstránenie objednávky

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ vyberie časové okno, na ktoré je vytvorená rezervácia a odstráni ju.

Reakcia: Rezervácia je odstránená. Táto zmena sa zobrazuje po prihlásení v danom časovom okne.

3.11. Prezeranie histórie objednávok pre klienta

Kroky ktoré sa vykonajú predtým: 13

Akcia: Používateľ vyhľadá klienta.

Reakcia: Používateľovi sa zobrazí história objednávok pre klienta, ktorého vybral.

3.12. Pridanie aktivity

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ na stránke administrácie aktivít vyplní informácie o aktivite zvolí pridať aktivitu.

Reakcia: Aktivita sa uloží do databázy a je možné na ňu vytvárať objednávky.

3.13. Editácia aktivity

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ na stránke administrácie aktivít zmení údaje o aktivite a klikne na tlačidlo uložiť zmeny.

Reakcia: Informácie o aktivite sa v databáze zmenia a je možné ďalej pracovať so zmenenou aktivitou.

3.14. Odstránenie aktivity

Kroky ktoré sa vykonajú predtým: 1

Akcia: Používateľ na stránke administrácie aktivít vyhľadá aktivitu a zvolí možnosť odstrániť ju.

Reakcia: Aktivita sa odstráni z databázy a už nie je možné na ňu vytvárať objednávky.

3.15. Prezeranie obsadenosti

Akcia: Klient prejde na stránku prezerania obsadenosti aktivít.

Reakcia: Klientovi sa zobrazí kalendár s obsadenosťou časových okien.

3.16. Export dát

Kroky ktoré sa vykonajú predtým: 1

Akcia: Klient klikne na tlačidlo/odkaz exportu dát.

Reakcia: Klientovi sa začne sťahovať súbor vo formáte excel s dátami z databázy.