Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Modul registrácie a prihlasovania webovej aplikácie

Bakalárska práca

2021
Frederik Kohár

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Modul registrácie a prihlasovania webovej aplikácie

Bakalárska práca

Študijný program:	Aplikovaná informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra aplikovanej informatiky
Školiteľ:	RNDr. Marek Nagy, PhD.

Bratislava, 2021
Frederik Kohár

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Frederik Kohár
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Modul registrácie a prihlasovania webovej aplikácie
Registration and login module of web application

Anotácia: Webové aplikácie štandardne využívajú autentifikáciu užívateľov, na základe ktorej sprístupňujú ďalší obsah a funkcionalitu. Bežný prístup je formou mena a hesla. Pre deti, ktoré majú ešte problém s čítaním, je táto forma náročná. Preto treba uprednostniť komunikáciu prostredníctvom obrázkov. Zaujímavým rozšírením overenia je aj hlasová biometria.
Webové aplikácie s veľkým množstvom užívateľov, ako sú napríklad školské portály, vyžadujú ich efektívny manažment. Kľúčovým prvkom je hlavne registrácia, ktorá by mala využiť aspoň jeden spoľahlivý prvok ako napríklad overenie cez email. Prípadne sa spoľahnúť na externé služby prihlasovania, ktorým sa dôveruje.

Cieľ: Vytvoriť modul v základnom Javascipt kóde s prepojením na node.js a socket.io knižnicu. Modul bude obhospodarovať lokálnu databázu všetkých užívateľov portálu. Bude realizovať overovanie klasickou formou (meno+heslo), grafickou formou pre deti a overovanie cez externé služby (univerzitné prihlasovanie, google, facebook,...). Doplnkové overenie bude experimentálne cez hlasovú biometriu. Modul bude poskytovať aj lokálnu registráciu potvrdzovanú emailom.

Vedúci: RNDr. Marek Nagy, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 27.09.2020

Dátum schválenia: 30.09.2020 doc. RNDr. Damas Gruska, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Tu môžete poďakovať školiteľovi, prípadne ďalším osobám, ktoré vám s prácou nejako pomohli, poradili, poskytli dáta a podobne.

Abstrakt

Slovenský abstrakt v rozsahu 100-500 slov, jeden odstavec. Abstrakt stručne sumarizuje výsledky práce. Mal by byť pochopiteľný pre bežného informatika. Nemal by teda využívať skratky, termíny alebo označenie zavedené v práci, okrem tých, ktoré sú všeobecne známe.

Kľúčové slová: jedno, druhé, tretie (prípadne štvrté, piate)

Abstract

Abstract in the English language (translation of the abstract in the Slovak language).

Keywords:

Obsah

Úvod **1**

1 Východiská **3**
1.1 Meno a heslo . . . 3
1.2 Aplikácie tretích strán . . . 4
1.2.1 Saml 2.0 . . . 4
1.2.2 OAuth 2.0 . . . 4
1.2.3 Facebook . . . 4
1.2.4 Zaregistrovanie aplikácie vo Facebooku . . . 6
1.2.5 Google účet . . . 6
1.2.6 Zaregistrovanie aplikácie v Google . . . 8
1.3 QR kód . . . 8
1.3.1 Zloženie QR kódu . . . 8
1.3.2 Informácie v QR kóde . . . 9
1.3.3 Základný prehľad procesu skenovania . . . 10
1.4 Biometrické overovanie . . . 10
1.4.1 Základné typy biometrického overovania . . . 11
1.5 Problém pri registrácii detí . . . 12
1.6 Registrácia pomocou mena, hesla a emailu . . . 12
1.7 Registrácia používateľa vždy osobitnou metódou . . . 12
1.8 Súčasný stav . . . 13

2 Návrh **14**
2.1 Role . . . 14
2.1.1 Prihlasovanie staršieho študenta . . . 14
2.1.2 Prihlasovanie mladšieho študenta . . . 15
2.1.3 Problém pri registrácií iba cez jednu metódu . . . 16
2.2 Použité nástroje a knižnice . . . 16
2.2.1 Javascript . . . 16
2.2.2 Node.js . . . 16
2.2.3 Socket.io . . . 17

2.2.4 bcrypt.js 17
2.3 Databáza 17
2.3.1 Accounts 18
2.3.2 FacebookAccounts, GoogleAccounts, FMFIAccounts 18
2.3.3 VoiceAccounts 18

3 Implementácia **19**
3.1 Registrácia metódou hlasu 19
3.1.1 Vypočítanie energie užívateľa 21
3.1.2 Vykreslenie spectrogramu 21
3.1.3 Prihlásenie metódou hlasu 21
3.1.4 Vzdialenosť polí 22
3.2 Registrácia metódou QR kód 22
3.2.1 Implementácia QR kódu 23
3.2.2 Prihlásenie metódou QR kódu 24
3.3 Registrácia verzus pridávanie metódy 25
3.4 Registrácia meno, heslo a email 25
3.5 Facebook a Google 26

Zoznam obrázkov

1.1 Facebook prihlásenie - používateľ ešte nie je prihlásený v prehliadači . 5
1.2 Facebook prihlásenie - používateľ je prihlásený v prehliadači 5
1.3 Facebook Login Diagram 6
1.4 Facebook App 6
1.5 Google prihlásenie - používateľ ešte nie je prihlásený v prehliadači . . . 7
1.6 Google API 7
1.7 Google App 8
1.8 QR kod 9
1.9 Modul výberu možnosti prihlasovania 13

2.1 Modul registrácie 14
2.2 Modul výberu možnosti prihlasovania 15
2.3 Modul prihlásenia 16
2.4 Relačný model databázy 17

3.1 Vzorkovacia frekvencia 20
3.2 Redukcia poľa 21
3.3 Spectrogram 22
3.4 QR kód 23
3.5 Poškodený QR kód 24

Úvod

V dnešnom modernom svete sa život presúva čoraz viac do online priestorov, čo má za následok vytvorenie množstva internetových stránok, ktoré používajú milióny ľudí. Niektoré z týchto internetových stránok slúžia iba na prehliadanie, preto nie je pre používateľov potrebné ukladať obsah. Pre zložitejšie webové aplikácie sa musí používateľ autentifikovať, aby vedela webová aplikácia určiť, čo môže používateľ v danej aplikácií robiť. Základné pravidlo, ktoré platí pri práci na internete je, že každý používateľ by si mal zvoliť pri každej aplikácií vždy iné heslo. Pri používaní viacerých takýchto zložitejších aplikácií sa môže stať, že pri tom množstve použitých hesiel ich používateľ môže ľahko zabudnúť. Preto sa v poslednej dobe začala dostávať do popredia autentifikácia pomocou tretej strany. Pri používaní internetu môžeme často vidieť, ako pri prihlásení máme okrem mena a hesla na výber aj prihlásiť sa cez Facebook alebo Google. To sú práve aplikácie tretích strán, kde sa zakladá na dôvere v tieto aplikácie.

S nárastom počtu webových aplikácií sa vytvorili aj aplikácie určené deťom. Pre mladšie deti, ktoré majú ešte problém s čítaním a písaním je často náročné zapamätať si heslo a napísať ho na klávesnici. Rovnako majú zakázanú registráciu v aplikáciách tretích strán. Preto je potrebné pomôcť aj takýmto užívateľom webových aplikácií a nájsť spôsob, ako sa jednoducho ale bezpečne autentifikovať do webovej aplikácie.

Dnešné moderné zariadenia nám neprinášajú pomoc pri autentifikácií iba pomocou softvérových riešení, ale taktiež aj vylepšenie hardvérových vlastností zariadení nám môže pomôcť pri riešení autentifikácie používateľov. Dnes máme web kamery ktoré dokážu detailne rozpoznať človeka, mikrofón, ktorý dokáže snímať vlastnosti ľudskej reči alebo skener odtlačku prstov, ktorý dokáže v momente identifikovať človeka.

Cieľom tejto bakalárskej práce je vytvoriť všestranný modul prihlasovania webovej aplikácie s experimentálnym využitím biometrie človeka. V aplikácii sa bude taktiež možné autentifikovať pomocou aplikácie tretej strany. Pre používateľov, ktorí nechcú používať pridané metódy bude stále dostupné klasickú autentifikáciu menom a heslom.

Hlavná časť práce je rozdelená na štyri kapitoly. V prvej kapitole opisujeme základné rozdelenie biometrických možností prihlasovania, fungovanie aplikácií tretích strán a opíšeme si aj klasickú formu mena a hesla. Druhá kapitola opisuje architektúru a popisuje databázový model, kde budú uložený používatelia. V tretej kapitole si popíšeme, ako sme aplikáciu implementovali a na aké problémy sme narazili. Štvrtá

kapitola popisuje používateľskú príručku, kde je napísané, čo všetko je potrebné urobiť, aby sme mohli prihlasovací modul používať v našej aplikácií.

Kapitola 1

Východiská

V dnešnej dobe si každý človek chce uchovať svoje súkromie čo najviac v tajnosti. Z toho dôvodu je zrejmé, že všetci používatelia internetových služieb potrebujú mať nejaké údaje, prostredníctvom ktorých sa v službách na internete autentifikujú a potvrdia, že sú to naozaj oni. Používateľ sa môže overiť troma základnými spôsobmi, a to sú:

- Používateľ pozná kľúč - heslo
- Používateľ ma jedinečnú vlastnosť - hlas, oko, odtlačok prsta
- Používateľ vlastní kľúč - QR kód na kartičke

Následne po autentifikácií nastáva autorizácia. Autorizácia určuje, čo môže autentifikovaný používateľ vidieť a čo všetko môže robiť v našej webovej aplikácií.

1.1 Meno a heslo

Autentifikácia pomocou mena a hesla sa zdá byť najjednoduchší spôsob prístupu do aplikácie. Avšak ľahko sa však môže stať, že používateľ zabudne svoje heslo a tým pádom navždy stratí prístup do aplikácie. V minulosti si užívateľ internetovej služby mohol zvoliť heslo bez požiadaviek na prístup do aplikácie. Kto si nebol vedomý hrozieb uhádnutia hesla iným človekom, ten si zvolil základné jednoduché heslo tvorené malými písmenami, alebo zvolil len čísla. To malo za následok nedostatočné zabezpečenie účtu a účet tak bol ľahko napadnuteľný. Postupom času vývojári internetových aplikácií začali pridávať rôzne požiadavky na heslo, ako napríklad povinnosť zvoliť aspoň jedno veľké písmeno z abecedy, povinnosť zmiešať písmená a čísla, alebo pridať k heslu nejaký iný znak ako je v abecede.

1.2 Aplikácie tretích strán

V tejto časti si opíšeme, ako sa využívajú aplikácie ako Facebook, Google, Twitter vo vlastnej internetovej službe. Využitie aplikácií tretích strán dáva užívateľom veľkú výhodu v tom, že si nemusia pamätať žiadne nové prihlasovacie mená a heslá.

1.2.1 Saml 2.0

SAML(Security Assertion Markup Language) je štandard na výmenu autentifikačných a autorizačných údajov medzi bezpečnostnými doménami. SAML 2.0 je vylepšenie verzie SAML 1.0. SAML 2.0 je protokol, ktorý je založený na XML(Extensible Markup Language), ktorý používa bezpečnostné tokeny, ktoré obsahujú potvrdenia na prenos informácií o používateľovi medzi poskytovateľom identity(Identity Provider) a poskytovateľom služieb (Service Provider)

1.2.2 OAuth 2.0

OAuth 2.0 (RFC 6749) je aplikačný rámec, ktorý umožňuje aplikáciám tretej strany získať obmedzený prístup k službe HTTP a to buď v mene vlastníka prostriedku, alebo povolí aplikácií tretej strany získať prístup vo svojom mene [3]

1.2.3 Facebook

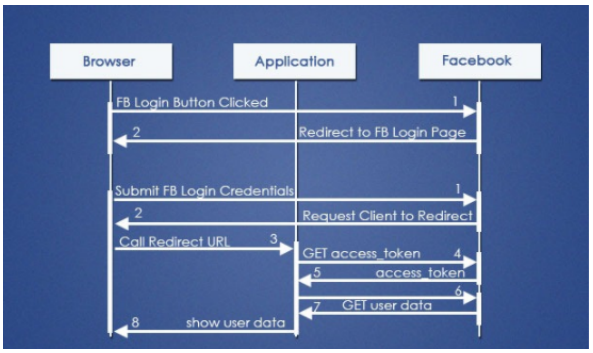
Facebook je sociálna sieť, cez ktorú môžu ľudia komunikovať prostredníctvom chatu, hlasovým hovorom, alebo video hovorom. Používajú ju milióny ľudí na svete, preto je túto metódu vhodné pridať do aplikácií s potrebou autentifikovať sa. Užívateľ, ktorý sa chce prihlásiť do aplikácie, musí mať vytvorený účet na Facebooku. Pri kliknutí na tlačidlo prihlás cez Facebook sa používateľ dostane na prihlasovací modul Facebooku, kde ak je prihlásený tak užívateľa to vráti naspäť do našej aplikácie obr 1.2. Ak tam ešte prihlásený nie je, ľahko vyplní potrebné údaje, a následne ho prihlási do našej aplikácie obr 1.1.

Diagram na obrázku 1.3 znázorňuje, ako aplikácie využívajú autentifikáciu pomocou Facebooku.
1. Pri vstupe na link, alebo uvítaciu webstránku sa zobrazí tlačidlo na prihlásenie službou Facebook. Používateľ klikne na toto tlačítko aby sa prihlásil do webovej aplikácie. Po kliknutí bude presmerovaný na Facebook.
2. Facebook overí ID žiadosti a potom užívateľa presmeruje na prihlasovaciu stránku.
1. Používateľ vloží Facebookové prihlasovacie údaje a potvrdí formulár.
2.Facebook overí prihlasovacie údaje a vytvorí požiadavok na presmerovanie na `redirect_url`. `Redirect_url` je link do našej aplikácie, ktorá sa už postará o zvyšok spracovania.

Obr. 1.1: Facebook prihlásenie - používateľ ešte nie je prihlásený v prehliadači

Obr. 1.2: Facebook prihlásenie - používateľ je prihlásený v prehliadači

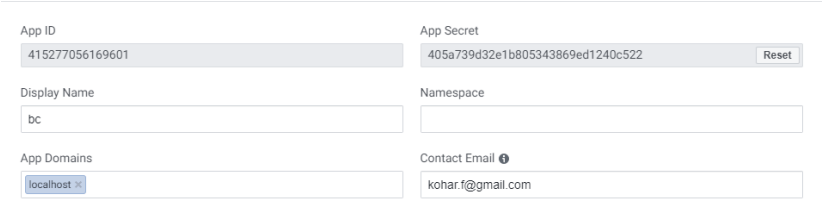
3. Internetový prehliadač otvorí `redirect_url`.
4. Webstránka na adrese `redirect_url` opäť zavolá Facebookovému serveru s požiadavkou na `access_token`.
5. V prípade, že overenie užívateľa na Facebooku je úspešné, server odošle späť `access_token`.
6. Po obdržaní `access_tokenu` aplikácia opäť zavolá Facebook s požiadavkou na informácie o užívateľovi, do požiadavky priloží `access_token`.
7. Facebook po overení `access_tokenu` pošle späť informácie o užívateľovi.
8. Naša aplikácia presmeruje užívateľa na stránku, ktorá uživateľovi zobrazí informácie, ktoré o ňom má.
[4]



Obr. 1.3: Facebook Login Diagram

1.2.4 Zaregistrovanie aplikácie vo Facebooku

Aby sa mohla používať autentifikácia cez Facebook vo vlastnej internetovej službe, je potrebné Facebooku povedať ako sa bude aplikácia volať a na čo sa bude používať. Na webovej stránke `https://developers.facebook.com/` sa vytvorí nová aplikácia, pomocou ktorej sa získava `App ID` a `App Secret` obr.1.4, cez ktoré sa bude internetová služba identifikovať Facebooku.

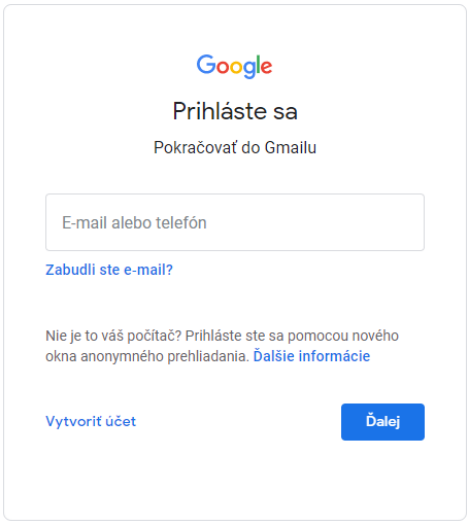


Obr. 1.4: Facebook App

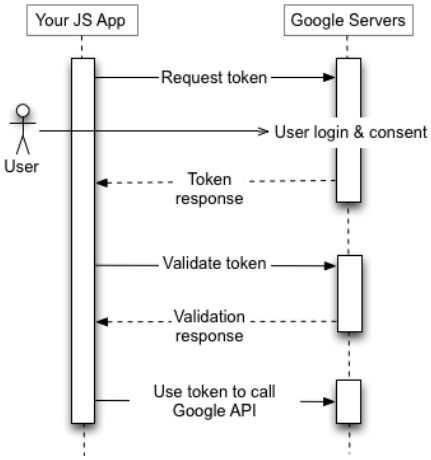
1.2.5 Google účet

Google účet používajú podobne ako Facebook milióny ľudí na svete. Pri registrácii na Google účet dostávame niekoľko dobrých aplikácii, ktoré môžeme využívať. Máme k dispozícií email, kde vieme komunikovať. Podstatná časť tohoto účtu je aj Google Disk, kde si vieme uložiť rôzne súbory, vieme ich zdieľať s inými používateľmi. Disk nám zabezpečí, že aj pri poruche počítača naše súbory ostanú nedotknuté a chránené. Pri vyhľadávaní rôznych informácii na internete sa používa najpopulárnejší Google vyhľadávač, kde na vyhľadávanie informácií nemusíme byť ani zaregistrovaný cez Google účet. Preto je vhodné aj túto metódu prihlasovania pridať do svojej internetovej služby. Google pracuje podobne ako Facebook pri prístupe do našej aplikácie. Pri stlačení tlačidla Prihlás cez Google sa používateľ presmeruje na prihlasovací modul Google, kde ak nie je prihlásený, vyplní svoj email a heslo. Google skontroluje či je všetko správne, a presmeruje ho to do našej aplikácie ako prihláseného používateľa. Ak už je v pre-

hliadači prihlásený, nemusí vyplňovať meno a heslo, ale bude automaticky prihlásený v našej internetovej službe.



Obr. 1.5: Google prihlásenie - používateľ ešte nie je prihlásený v prehliadači

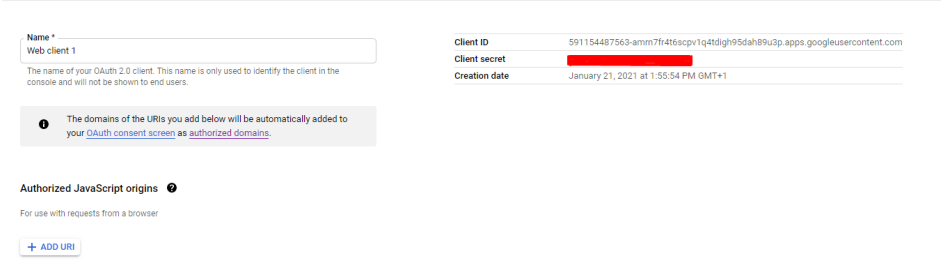


Obr. 1.6: Google API

Na obrázku 1.6 si môžeme lepšie pozrieť, ako funguje celá autentifikácia pomocou Google. Autorizácia začína tým, že aplikácia presmeruje prehliadač na Google, kde sa posielajú parametre, ktoré označujú typ požadovaného prístupu. Na strane Google sa spracováva autentifikácia používateľa, výber relácie a súhlas používateľa. Ako výsledok dostávame prístupový token, ktorý by mal klient pred zahrnutím do žiadosti overiť. Ak nastane vypršanie platnosti tokenu, aplikácia proces zopakuje.

1.2.6 Zaregistrovanie aplikácie v Google

Podobne, ako vo Facebooku, aj v Google je potrebné internetovú službu najskôr zaregistrovať, tentokrát na webovej stránke `https://console.developers.google.com`, kde tiež aplikácia získava `Client ID` a `Client Secret` obr 1.7. Tie budú potrebné na identifikovanie sa aplikácie pre službu Google.



Obr. 1.7: Google App

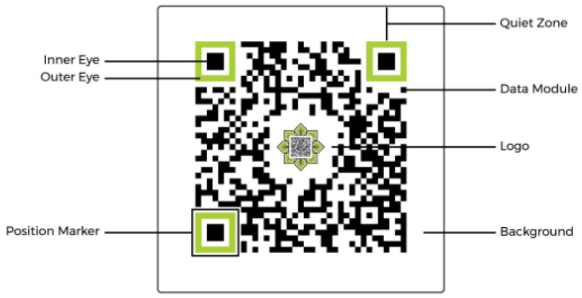
1.3 QR kód

QR kód(quick response code) je typ čiarového kódu, ktorý poznáme z produktov v supermarkete. QR kód sa stará o zakódovanie informácií do usporiadaných štvorčekov. Tieto informácie zo štvorčekov je následne možné odkódovať do informácie. Oproti klasickému čiarovému kódu má QR kód značné výhody, ako napríklad veľkosť dát uložených v kóde. Ďalšia veľká výhoda je, že nie je potrebné vlastniť žiadny špeciálny nástroj, ale stačí použiť akýkoľvek dostupný smartfón. Taktiež prečítanie a spracovanie QR kódu je rýchlejšie ako klasický čiarový kód. Najbežnejšie farby QR kódu sú čierne štvorčeky na bielom pozadí, avšak pri tvorbe vlastného QR kódu sa dajú určiť ľubovolné farby.

1.3.1 Zloženie QR kódu

V tejto časti si ukážeme, aké informácie vieme vyčítať z QR kódu pri pohľade naň.

- Data module(dátový model) - je to štandardná jednotka QR kódu kde sú uložené zakódované informácie. Usporiadanie týchto dátových modulov tvorí väčšinu QR kódu.
- Position marker(značka polohy) - Na každom QR kóde sa nachádzajú tri značky polohy ktoré sú zložené z `inner eye (vnútorné oko)` a `inner eye (vonkajšie oko)`. Tieto značky polohy sú dôležité na to, aby skener vedel rýchlo určiť pozíciu QR kódu a lokalizovať smer skenovania.



Obr. 1.8: QR kod

- Quiet zone(čistá zóna) - Tichá oblasť je prázdna časť QR kódu okolo celého dátového modulu a značiek polôh. Používa sa na pomoc skenerom lepšie určiť miesto kde začína a končí QR kód.

1.3.2 Informácie v QR kóde

V QR kóde máme tiež uložené informácie o QR kóde ako veľkosť, úroveň opravy chýb a dátový typ.[5]

1. Veľkosť - vzrastajúcou popularitou QR kódov sa ich produkcia postupne zvyšovala, preto bolo potrebné zväčšiť ich veľkosť v počte riadkov a stĺpcov. Najmenšie QR kódy aké poznáme majú 21 stĺpcov a 21 riadkov čo umožňuje mať 441 dátových modulov. Táto verzia s 21 riadkami a 21 stĺpcami je označovaná ako verzia 1. Druhá, väčšia verzia je zložená z 25 riadkov a 25 sĺpcov. To znamená že do QR kódu verzie 2 sa vojde 625 dátovych modulov. Postupným pokračovaním sa dostaneme až do verzie 40, ktorá sa skladá z 177 riadkov a 177 stĺpcov. Tu sa vojde až 31329 dátových modulov.

2. Úrovne opravy chýb - Pri používaní QR kódu vo vytlačenej forme sa môže stať, že časť kódu zašpiníme alebo inak poškodíme, čo by malo znefunčniť celý QR kód. Avšak QR kód obsahuje informáciu o úrovni opravy chýb. Preto bude pri vhodnom nastavení použiť aj poškodený kód, aj keď z neho nejakú časť úplne odstránime. Princíp je taký, že z hlavného QR kódu sa vyberú jedinečné body na jednoznačné určenie kódu. Tieto jedinečné body budú uložené do pôvodného QR kódu ako záloha. Poznáme 4 úrovne korekcie chýb v QR kóde, ktoré určujú aké množstvo záložných údajov sa uloží v QR kóde. To znamená, že čím vačšia úroveň korekcie, zvýši sa aj počet riadkov a stĺpcov potrebných na uloženie jedinečných bodov. Tieto úrovne sú nasledovné:

   (a) L - dovoľuje poškodenie do 7%

   (b) M - dovoľuje poškodenie do 15%

(c) Q - dovoľuje poškodenie do 25%

(d) H - dovoľuje poškodenie do 30%

3. Dátový typ - QR kódy môžu obsahovať až 7 089 číselných znakov alebo 2 953 alfanumerických znakov. Môžu tiež ukladať bajty a kanji, ale tie sa používajú menej často. Tieto čísla predpokladajú najnižšiu úroveň korekcie chýb. To znamená, že použitie QR kódu zahŕňa čokoľvek, čo na komunikáciu používa čísla, písmená, interpunkciu a symboly.

1.3.3 Základný prehľad procesu skenovania

1. Skener QR kódov najprv rozpozná 3 značky polohy v QR kóde. Pri použiť vhodnej čistej zóny sa skener rýchlo zorientuje a je informovaný o tom kde sú okraje QR kódu.

2. Skener začína vpravo dole, kde sa nachádza indikátor režimu, ktorý označuje, akým dátovým typom je určený zvyšok kódovaných údajov. Indikátor režimu je zložený zo štyroch dátových modulov.

3. Po zistení potrebných údajov z indikátoru režimu sa skener posunie vyššie na indikátor počtu znakov, ktorý je zložený z ôsmich dátových modulov. Tieto dátové moduly označujú, počet znakov kódovaných údajov.

4. Následne po získaní identifikátorov počtu znakov a režimu sa skener posúva ďalej cez dátové moduly až kým nenarazí na koncový identifikátor.

5. Po prečítaní koncového identifikátoru skener pokračuje na dátové moduly kde je uložená úroveň opravy chýb.

1.4 Biometrické overovanie

Biometrické overenie sa používa v oblasti zabezpečenia, kde si používateľ nemusí pamätať žiadne prihlasovacie meno, heslo alebo email. Namiesto toho sa používajú biometrické vlastnosti používateľa, pomocou ktorých sa overuje, či má osoba prístup ku konkrétnemu zariadeniu. Každý človek má v tele nejaké jedinečné fyzikálne a biologické vlastnosti, ktoré sa ľahko dajú porovnať s predtým nahratými vlastnosťami v databáze. Prihlasovanie pomocou biometrie sa nepoužíva iba v počítačoch, ale často slúži aj ako vstup cez dvere alebo brány. Aj pre moderné smartfóny platí, že výrobcovia sa snažia pridať do svojich zariadení biometrické overenia na vstup to telefónu. Najprv to bol snímač odtlačku prstu na zadnej strane telefónu, postupom času sa to prepracovalo na snímač odtlačku prstu vložený priamo v displeji mobilného telefónu. Taktiež sa začalo v najnovších telefónoch používať overenie pomocou rozpoznávania tváre.

1.4.1 Základné typy biometrického overovania

V tejto časti si popíšeme základné typy biometrických overovaní, ktoré sa v súčastnosti používajú najviac.

Skenery odtlačkov prstov

Princíp skenovania odtlačkov prstov je založený na princípe, ktorý sa používal už dávno, keď sa pomocou atramentu a papiera porovnávali odtlačky prstov. Každý jednotlivec má iný odtlačok prsta, čo dáva výhodu jednoznačne určiť každého jednotlivca. Môže to mať však aj nevýhody v podobe zranenia na prste, kedy sa môže časť kože spáliť alebo úplne odstrániť. Novšie verzie skenerov odtlačkov prstov sa vedia dostať až pod kožu, kde snímajú aj teplotu pod kožou, čo pomáha pri bezpečnosti a spoľahlivosti tohoto systému. Je to lepšie preto, pretože pri obyčajnom porovnávaní odtlačku prsta môže niekto umelo vytlačiť vzor človeka, ktorý len priloží k snímaču.

Rozpoznávanie tváre

Táto technológia funguje na princípe porovnávania s pôvodnou schválenou tvárou v databáze. Technológia vykonáva desiatky meraní z ktorých vytvára odtlačky tváre používateľa, ktorý sa snaží získať prístup. Ak sa dostatočný počet odtlačkov tváre zhoduje so schválenou tvárou, vtedy je udelený prístup. Biometrické overenie pomocou rozpoznávania tváre je však zložitý problém, pretože porovnávanie nemusí byť správne pri pohľade s iných uhlov, alebo môže mať problém s rozpoznávaním dvoch podobne vyzerajúcich ľudí.

Identifikácia hlasu

Táto technológia sa používa na overenie osoby pomocou rozpoznávania jej hlasových vzorov. Identifikácia hlasu funguje veľmi dobre hlavne preto, že každý človek sa svete má jedinečné fyzické, fonetické aj morfologické vlastnosti. Preto je bezpečnosť tejto technológie veľmi vysoká, keďže je veľmi odolná voči podvodom. Veľká výhoda tejto technológie je dostupnosť na zariadeniach. Mikrofóny sú prakticky v každom mobilnom telefóne, nemusia mať fotoaparát alebo iné snímače. Taktiež mikrofón vlastní veľká časť notebookov. Ak sa náhodou stane, že používateľ má bežný stolný počítač bez mikrofónu, mikrofón sa dá získať ľahko po napojení slúchadiel, kde aj tie cenovo najdostupnejšie mikrofón obsahujú.[2]

Očné skenery

Prvý typ očných skenerov je skener sietnice, ktorý funguje tak, že smerom do oka premieta jasné svetlo, pomocou ktorého vytvára viditeľné vzory krvných ciev, ktoré

skener číta a porovnáva s informáciami v databáze. Druhý typ očného skenera je skener na rozpoznávanie dúhovky. Tento skener funguje podobne ako skener sietnice, avšak tu neporovnáva krvné cievy, ale hľadá jedinečné vzory v farebnom kruhu okolo očnej zrenice. Nevýhodou tejto technológie je, že pre človeka ktorý musí mať okuliare alebo nosí kontaktné šošovky je to nepoužiteľné.

1.5 Problém pri registrácii detí

Deti, ktoré ešte nevedia dobre čítať a písať, budú mať problém sa dostať do akejkoľvek internetovej služby. Môžu mať problém nezapamätať si heslo alebo prihlasovacie meno. Preto je vhodné myslieť aj na nich, a umožniť im jednoduchý, ale bezpečný spôsob ako tieto služby využívať. Ako dobrá možnosť, ktorá pomôže deťom v prihlasovaní by mohla byť autentifikácia pomocou ich hlasu, alebo QR kódom uloženým na kartičke.

1.6 Registrácia pomocou mena, hesla a emailu

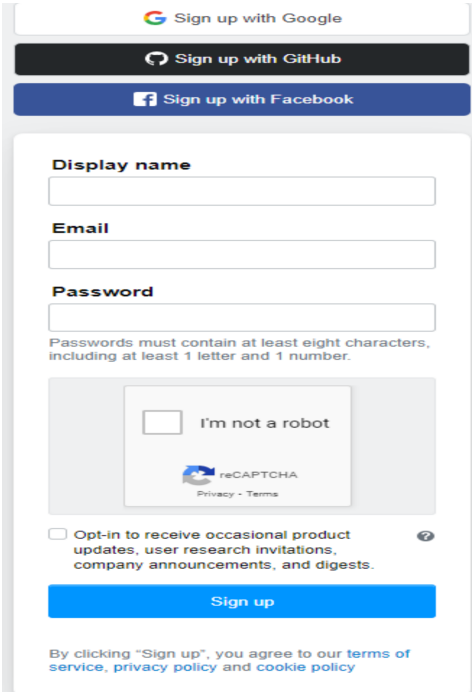
Ak je súčasťou metódy spomenutej vyššie(meno a heslo) aj povinnosť zadania emailu, tak zabudnuté heslo už nebude žiadny problém, pretože používateľ si môže ľahko poslať nové heslo na email a podľa inštrukcií si môže znova nastaviť prístupové údaje také, aké chce. Niekedy sa môže stať, že dôsledkom nepozornosti užívateľ vyplní zlú emailovú adresu, čo bude pre užívateľa ten istý problém ako keby žiadnu emailovú adresu nezadával. Preto je vhodné vymyslieť spôsob aby sa to nestalo, napríklad povinnosť potvrdiť verifikačný link v emaile. Ak náhodou mail nepríde, užívateľ bude mať informáciu o tom, že jeho prihlasovacie údaje nie sú v poriadku.

1.7 Registrácia používateľa vždy osobitnou metódou

Ak sa používateľ zaregistruje pomocou viacerých metód, bude to mať za následok viac účtov v našej internetovej službe. To niekedy nechceme, pretože to môže mať za následok, že na jednom účte si uloží svoju prácu, ale pri prihlásení cez iný účet neuvidí svoju prácu. Preto je dobré vymyslieť spôsob ako tomu zabrániť. Za vhodné riešenie by sa mohlo považovať zakázanie registrácie viacerými metódami. To však nie je možné, pretože každá metóda má potrebné iné údaje. Kvôli tomu internetová služba nevie zistiť, ktorý zákazník sa chce registrovať.

1.8 Súčasný stav

Na stránke `stackoverflow.com` sa dá registrovať pomocou Facebooku, Google a Github účtu a pomocou mena a hesla. Pri zaregistrovaní cez meno, heslo a email je nutné potvrdiť aktiváciu. To znamená, že je potrebné sa prihlásiť na email, a kliknúť na aktivačný kód, ktorý následne používateľa presmeruje na stránku a až vtedy je možné ju využívať ako prihlásený člen. Pri opätovnom pokuse o registráciu rovnakého emailu používateľa systém upozorní, že už registrovaný bol a vypýta si emailovú adresu, kde zašle návod na obnovenie účtu. Ak užívateľ zadá iný email ako pri prvotnej registrácii, zmení sa mu aj v jeho účte na `stackoverflow.com`. Pri použití Facebook účtu používateľa presmeruje na Facebookovskú stránku, kde vyplní email a heslo. Následne to vráti už prihláseného užívateľa. Ak užívateľ teraz použije prihlasovanie Google účtom, stránka ho presmeruje na stránku Google kde sa prihási a to ho vráti ako prihláseného užívateľa. Ak systém zistí, že email na Facebooku a na Google sú zhodné, tak sa použije účet skôr registrovaného účtu. Zobrazený modul registrácie je na obrázku 1.9



Obr. 1.9: Modul výberu možnosti prihlasovania

Kapitola 2

Návrh

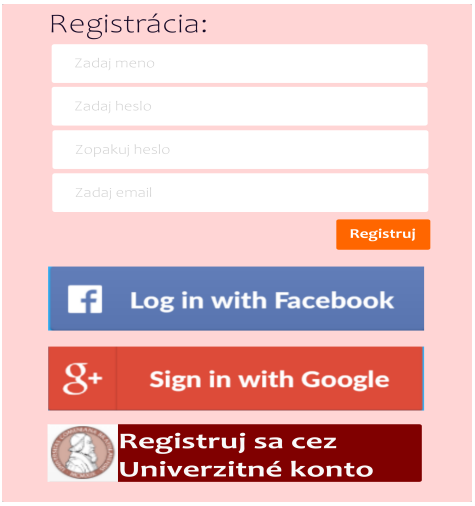
V časti návrh si opíšeme ako bude prihlasovací modul vyzerať, a detailnejšie si opíšeme navrhnutný databázový model.

2.1 Role

starší študent - študent nad 16 rokov
mladší študent - študent pod 16 rokov
zodpovedná osoba - rodič, alebo zástupca mladšieho študenta

2.1.1 Prihlasovanie staršieho študenta

Starší študent sa registruje pomocou jednej zo štyroch hlavných metód. Čiže cez Facebook, Google, Univerzitné konto a registrácia menom, heslom, emailom. Modul je zobrazený na obrázku 2.1.



Obr. 2.1: Modul registrácie

Používateľ si vyberie jednu z nich, ak to bude Facebook, Google, Univerzitné konto,

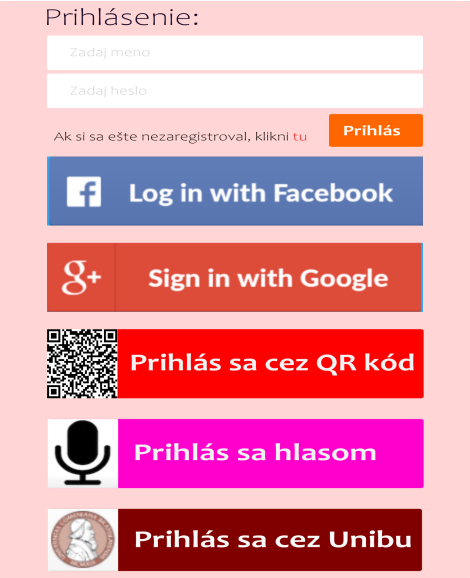
tak ho aplikácia presmeruje na prihlásenie sa do konkrétnej metódy, odkiaľ dostaneme potrebné informácie na zaregistrovanie sa do portálu. Ak si zvolí registráciu emailom, musí zadať používateľské meno, heslo a email. Po registrácii bude musieť ísť na svoj email, kde potvrdí aktivačný link. Po vstupe do portálu sa používateľovi zobrazí modul na výber prihlasovacích možností, kde si môže zvoliť, cez ktoré metódy sa chce prihlasovať v budúcnosti. Modul je zobrazený na obrázku 2.2.

Obr. 2.2: Modul výberu možnosti prihlasovania

Postup bude rovnaký ako pri registrácii, avšak tu už používateľ nebude môcť vytvoriť dva rôzne účty, ale všetky metódy prihlasovania sa uložia k jeho `id` v čítanke. Vždy je potrebné, aby bola zvolená aspoň jedna možnosť prihlasovania. V týchto možnostiach pridáme aj prihlasovanie pomocou rozpoznania hlasu a prihlasovanie pomocou QR kódu. Pri budúcom prístupe na stránku sa pre prihlásenie zobrazí nasledovný modul prihlásenia- obrázok 2.3

2.1.2 Prihlasovanie mladšieho študenta

Pri registrácii mladšieho študenta chceme, aby mala k jeho účtu prístup aj jeho zodpovedná osoba. To zabezpečíme tak, že rodič sa zaregistruje cez svoju obľúbenú metódu. Následne po vstupe do portálu a v module pre možnosti prihlásenia zvolí jednu z metód - prihlasovanie cez QR kód alebo prihlasovanie pomocou hlasu. Týmto zabezpečíme, že mladšiemu študentovi bude stačiť pri prihlásení ukázať jeho vygenerovaný QR kód na kameru, alebo povedať heslo do mikrofónu. Ak sa stratí QR kód, stačí, keď sa zodpovedná osoba prihlási do portálu, a tento kód si môže kedykoľvek znova stiahnuť.



Obr. 2.3: Modul prihlásenia

2.1.3 Problém pri registrácií iba cez jednu metódu

V súčasnom svete chceme, aby sme mali všetky informácie hneď, rovnako požadujeme okamžitý prístup ku všetkým naším potrebám. Ak sa používateľ zaregistruje cez jednu metódu, očakáva, že prístup do našej internetovej služby bude vždy dostupný. Čo však môže nastať je to, že používateľ ktorý sa registruje napríklad len cez Facebook, nebude mať prístup k našej aplikácii, pretože v tej dobe môže mať Facebook náhly výpadok. Preto bude vždy lepšie si zvoliť viac metód prihlasovania.

2.2 Použité nástroje a knižnice

Aby sme vytvorili funkčnú aplikáciu, musíme použiť vhodné technológie, ktoré budú tiež ľahko upraviteľné v budúcich úpravách aplikácie.

2.2.1 Javascript

Javascript je jazyk prehliadačov. S nárastom tvorby a využívania webových stránok sa z jazyka Javascript stal jeden z najpoužívanejších jazykov. Javascript má niekoľko veľkých výhod. Jednou z veľkých výhod je, že pri deklarácií premenných nemusíme určiť, či pôjde o integer, string alebo iné typy. [1].

2.2.2 Node.js

None.js je Open-source prostredie Javascriptu, ktoré vykonáva kód mimo webového prehliadača. Slúži na spúšťanie skriptov na strane servera.

2.2.3 Socket.io

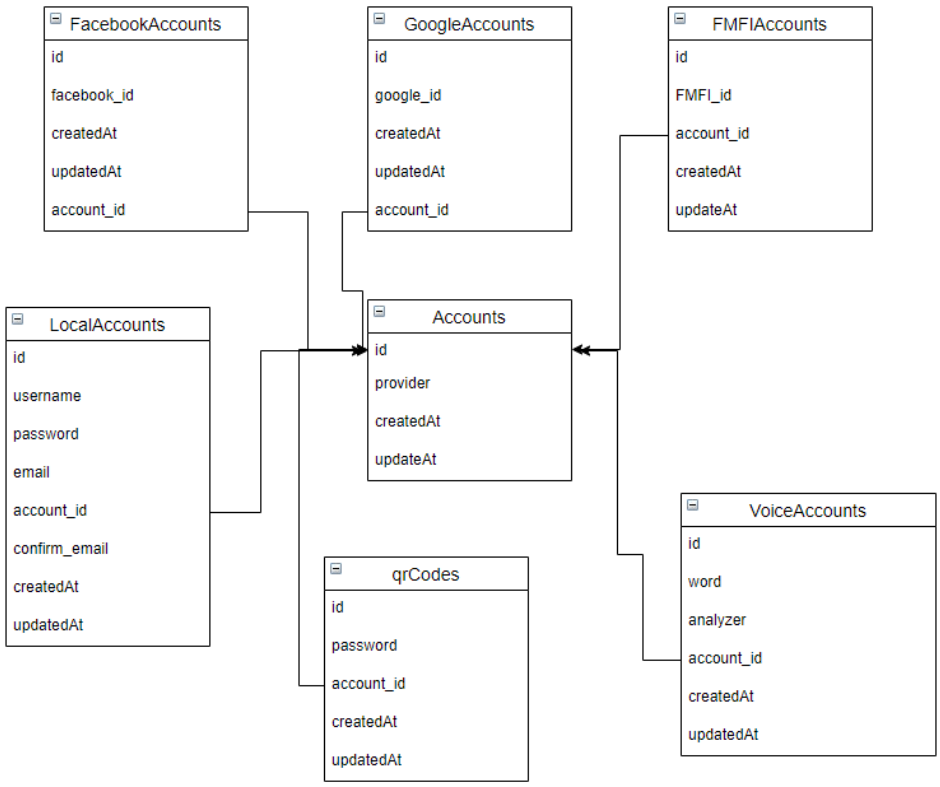
Socket.IO je knižnica, ktorá umožňuje v reálnom čase obojsmernú komunikáciu založenú na udalostiach medzi prehliadačom a serverom.

2.2.4 bcrypt.js

Bcrypt je knižnica, ktorá pomáha so zašifrovaním hesla pri vkladaní do databázy a taktiež tieto zašifrované heslá dokáže dešifrovať.

2.3 Databáza

Aby bolo možné vytvoriť prihlasovací modul, tak si musíme vytvoriť databázu a v nej uchovávať používateľov. Relačný model databázy je navrhnutý tak, aby uchovával iba tie najpotrebnejšie veci pre autentifikáciu. Relačný model databázy si môžeme pozrieť na obrázku 2.4.



Obr. 2.4: Relačný model databázy

2.3.1 Accounts

Tabuľka Accounts obsahuje iba primárný kľúč id typu integer a provider typu varchar. V stĺpci provider budú uložené údaje o tom, ktorou metódou sa používateľ zaregistroval do našej webovej aplikácie. Stĺpce createdAt a updatedAt sa vytvorili použitím Sequelize.js a určujú, kedy používateľ vytvoril účet, alebo ho aktualizoval. Tieto stĺpce budú typu datetime.

2.3.2 FacebookAccounts, GoogleAccounts, FMFIAccounts

Tabuľky FacebookAccounts, GoogleAccounts, FMFIAccounts sú rovnako štruktúrované. Obsahujú primárny kľúč id, ktorý je typu integer, facebook_id, google_id, FMFI_id sú stĺpce vytvorené pre používateľove id z aplikácie tretej strany. Pre tento sĺpec sme nastavili typ varchar, pretože google_id obsahuje aj nečíselné znaky. Cudzí kľúč account_id sa odkazuje na primárny kľúč tabuľky Accounts

2.3.3 VoiceAccounts

Keďže každý používateľ si môže nahrať do databázy viac slov, ale pre každé slovo si môže uložiť iba jeden hlasový odtlačok, musíme zabezpečiť unikátne spojenie stĺpcov account_id a word. Do stĺpca analyzer sa budú ukladať polia ako varchar.

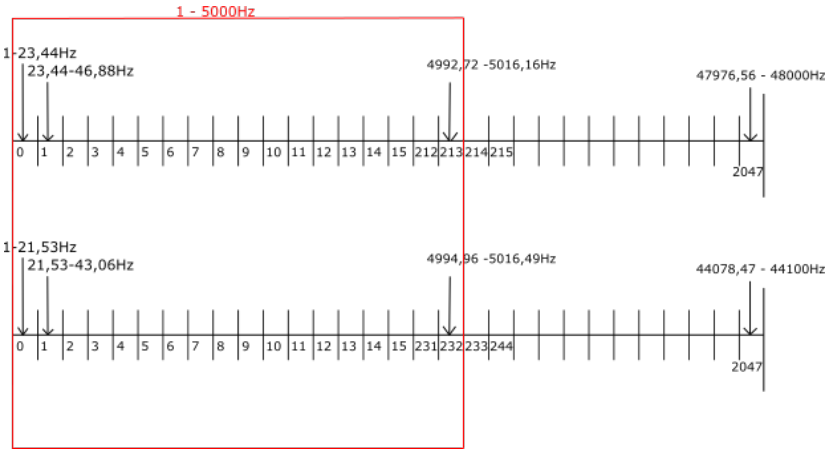
Kapitola 3

Implementácia

3.1 Registrácia metódou hlasu

Na registráciu pomocou hlasovej biometrie musí byť používateľ prihlásený v našej webovej aplikácii, takže sa musí najprv registrovať niektorou z iných metód, ktoré aplikácia poskytuje. Týmto získame prístup k používateľovmu id, čo znamená, že používateľ nemusí zadávať žiadne používateľské meno. Od používateľa potrebujeme iba to, aby si vybral nejaké slovo, ktoré ide prečítať a ku ktorému sa priradia jeho jedinečné vlastnosti reči. Po uložení vybratého slova webová aplikácia spustí požiadavku na webový prehliadač s tým, že potrebuje prístup k mikrofónu. Následne sa používateľovi zobrazí výzva na povolenie využitia mikrofónu našou webovou aplikáciou, ak má používateľ pripojených viac mikrofónov, môže si vybrať, ktorý mikrofón sa má použiť. Toto povolenie je potrebné iba vtedy, ak užívateľ prichádza do webovej aplikácie prvý krát. Pri ďalšom navštívení stránky si už webový prehliadač pamätá, že táto aplikácia má prístup povolený. K prístupu k mikrofónu sme použili getUserMedia[7], kde sme nastavili, že chceme informácie iba z mikrofónu, pretože web kamera pre nás nie je teraz potrebná. Prvý problém, ktorý nám táto metóda prináša je, že rôzne typy zariadení môžu mať inú vzorkovaciu frekvenciu(sample rate). Vzorkovacia frekvencia vyjadruje rozsah frekvencií, ktoré dokáže mikrofón zachytiť. Notebooky majú štandardnú vzorkovaciu frekvenciu 44100Hz, pre tablety je to 48000Hz. Tieto štandardné vzorkovacie frekvencie sa využijú vo WebAudioApi [8]. V súčasnej dobe nie je možné tieto štandardné vzorkovacie frekvencie modifikovať. To znamená, že vždy keď zavoláme funkciu, ktorá nám vráti pole prvkov, tak dostaneme pole dĺžky 48000 s jednotlivými hodnotami pre každú frekvenciu. Aby sme nemuseli pracovať s takými veľkými dátami, využijeme schopnosť analyzéru, a tou je nastavenie fftSize. Pre fftSize môžeme nastaviť iba hodnoty, ktoré sú mocniny dvojky. Nastavíme fftSize na 2048. Urobíme výpočet, kde vzorkovaciu frekvenciu podelíme veľkosťou fftSize. Teda 48000 / 2048 = 23,44. V tomto momente už nebudeme mať pole dĺžky 48000, ale pole bude mať po-

dobu [1-23.44Hz, 23.44-46.88Hz,...]. Keďže ľudský hlas sa pohybuje vo frekvenciách od 1 - 5000Hz, musíme nájsť prvky, ktoré patria pod túto hranicu. Vydelíme teda 5000 / 23.44, čo je približne 213 prvkov. Teda k tomu, aby sme vedeli rozpoznať hlasové vlastnosti používateľa, musíme vybrať práve prvých 213 prvkov. Pri notebookoch a vzorkovacej frekvencii 44100 máme prvých 5000Hz uložených v 5000/(44100/2048), čo je približne 232 prvkov. Rôzna vzorkovacia frekvencia by nerobila problém v prípade, že by používateľ pracoval vždy na rovnakom zariadení. Avšak ak sa zaregistruje cez notebook a bude sa pokúšať prihlásiť cez tablet, nebude to fungovať. Preto dôležité zredukovať tieto polia na spoločný počet prvkov. Tento počet nastavíme na 200. Takže, ak sa používateľ zaregistruje cez notebook, tak vieme že pole, kde sú frekvencie od 1 - 5000hz je na prvých 213 prvkoch. Toto pole zredukujeme tak, že z poľa dĺžky 213 prvkov odpočítame veľkosť poľa na ktorú chceme dané pole zredukovať, teda 200. Zistili sme, že pole je väčšie o 13 prvkov, čo znamená, že z poľa veľkosti 213 potrebujeme odstrániť 13 prvkov. Aby sme zachovali čo najpresnejšie informácie v poli, nebudeme tieto prvky odstraňovať zo začiatku alebo z konca. To zabezpečíme tak, že nájdeme ktoré pole má väčší počet prvkov. Od tohto počtu prvkov odpočítame veľkosť menšieho poľa. Takto získaný rozdiel nám udáva, koľko prvkov musíme vymazať z väčšieho poľa. Na zistenie toho, ktoré prvky máme vymazať vypočítame index týchto prvkov. Na výpočet použijeme vzorec, kde dlhšie pole vydelíme rozdielom týchto dvoch polí. Index si označíme ako `i` a teda z výpočtu sme zistili, že musíme odstrániť každý `i-ty` prvok z väčšieho poľa. Pre ukážku riešenia daného problému si môžeme pozrieť obr. Vidíme, že máme pole značené od 0 po 29, čo znamená, že veľkosť nášho poľa je 30. Určime si, že potrebujeme nové pole dĺžky 20 tak, aby sme ho poškodili čo najmenej. Vypočítame si teda, že musíme odstrániť 10 prvkov. To znamená, že keď z 30 prvkov potrebujeme odstrániť 10 prvkov, tak musíme odstrániť každý tretí prvok. To vypočítame ako 30 / 10. Takto sme zabezpečili čo najmenšie poškodenie poľa.



Obr. 3.1: Vzorkovacia frekvencia

Obr. 3.2: Redukcia poľa

3.1.1 Vypočítanie energie užívateľa

Aby sme si do databázy zbytočne neukladali informácie, kedy používateľ nehovorí, vypočítame si energiu používateľa v danom momente. To vypočítame tak, že si spočítame každú hodnotu frekvencie a vydelíme počtom frekvencií. Tu sme testovaním zistili, že keď používateľ začne hovoriť, tak sa táto hodnota pohybuje od 15 a vyššie. Preto sme nastavili spodnú hranicu na 15, ktorú keď prekročí, tak sa odštartuje ukladanie do poľa, ktoré uložíme do databázy. Tu si nastavíme semafor, že používateľ začal hovoriť. Na zistenie, kedy používateľ prestal hovoriť použijeme práve tento semafor, ktorý vypneme, keď používateľovi klesne priemerná hranica frekvencie pod 15. To sa niekedy môže stať aj v polovici slova, napríklad pri krátkom nádychu alebo znížení hlasitosti. Tento problém sme zabezpečili tak, že pri klesnutí priemernej hranice pod 15 nastavíme počítadlo, ktoré začne počítať čas. Keď táto hranica prekročí jednu sekundu, môžme si byť istý, že používateľ prestal hovoriť. Pre poslanie informácií na server sme použili metódu fetch, ktorá na pozadí používateľa presmeruje na `/auth/voice/registration`, kde metódou `POST` pošleme dáta z nášho vytvoreného poľa pre frekvencie a slovo, ktoré používateľ zadal.

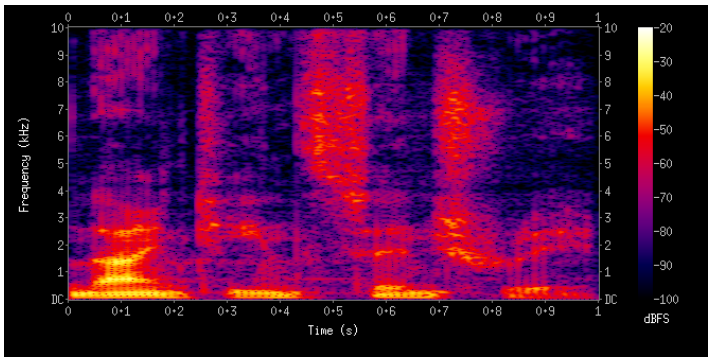
Takto prijaté dáta na serveri spracujeme, skontrolujeme či sa tam toto slovo ešte pre daného užívateľa nenachádza, a následne uložíme do databázy.

3.1.2 Vykreslenie spectrogramu

Od toho času, ako používateľ zapne svoj mikrofón v prehliadači, budeme používateľovi taktiež zobrazovať jeho hlas v grafickej forme. Na vykreslenie hlasu použijeme spectrogram, ktorý podľa výšky čísla jednotlivých frekvencií zafarbí daný pixel na teplejšie, alebo chladnejšie farby. Obrázok spectrogramu si môžeme pozrieť na obrázku 3.3. Na tomto obrázku vidíme, že čas plynie horizontálne a jednotlivé čísla z frekvencií plynú vertikálne.

3.1.3 Prihlásenie metódou hlasu

Pri prihlásení metódou hlasu je postup takmer rovnaký ako pri registrácií metódou hlasu. Tu však používateľ musí zadať svoje id. Po zadaní id sa pošle požiadavka na server, ktorú aplikácia spracuje, vyhľadá v databáze používateľa so zadaným id, a vráti odpoveď v podobe náhodne vybraného slova, ktoré si používateľ zadal pri registrácii.



Obr. 3.3: Spectrogram

Taktiež si z databázy získame hlasové vlastnosti užívateľa pre dané slovo. Následne sa spustí mikrofón, hlasové informácie sa zanalyzujú a do nového poľa sa uložia dané frekvencie. Teraz máme dve polia, jedno z databázy, druhé od používateľa. Na porovnanie podobnosti týchto dvoch polí použijeme knižnicu `euclidean-distance`, ktorá nám vypočíta ako ďaleko sú od seba tieto polia vzdialené. Následne určíme hranicu, pri akej vzdialenosti používateľa prihlási, alebo nie.

3.1.4 Vzdialenosť polí

Na to, aby sme mohli použiť knižnicu na výpočet vzdialenosti dvoch polí musíme zabezpečiť, že počet prvkov v oboch poliach bude rovnaký. To zabezpečíme tak, že nájdeme ktoré pole má väčší počet prvkov. Od tohto počtu prvkov odpočítame veľkosť menšieho poľa. Takto získaný rozdiel nám udáva, koľko prvkov musíme vymazať z väčšieho poľa. Na zistenie toho, ktoré prvky máme vymazať vypočítame index týchto prvkov. Na výpočet použijeme vzorec, kde dlhšie pole vydelíme rozdielom týchto dvoch polí. Index si označíme ako `i` a teda z výpočtu sme zistili, že musíme odstrániť každý `i-ty` prvok z väčšieho poľa.

3.2 Registrácia metódou QR kód

Na registráciu pomocou QR kódu sa musíme, podobne ako pri registrácii pomocou hlasu, prihlásiť a registrovať niektorou z inej metódy. Takto registrovaný a prihlásený užívateľ nám poskytne svoje id, ktoré je uložené v databáze pre našu webovú aplikáciu. Takto získané id spojíme pomocou dvojbodky s náhodne vygenerovaným textovým reťazcom. Tento náhodný textový reťazec ukladáme do databázy ako heslo pre príslušné ID. Na vykreslenie textového reťazca do QR kódu používame knižnicu `qrcode` na serverovej strane. Na webový prehliadač pošleme už takto vygenerovaný obrázok. Tento vygenerovaný obrázok si môže používateľ stiahnuť a vytlačiť, alebo odfotiť mobilným telefónom. Používateľovi, ktorý si QR kód vytlačí odporúčame nastaviť veľkosť ob-

rázku na minimálne 1,5 centimetra na výšku a 1,5 centimetra na šírku. Na ukážku sa používateľovi s id 93 vygeneruje náhodný reťazec v podobe c983e721d4feb, teda celý náš kód bude preložený v textovom formáte vyzerať nasledovne 93:8615aa627a6a4. QR kód vygenerovaný z tohoto reťazca je vyobrazený na obrázku 3.4.

Obr. 3.4: QR kód

3.2.1 Implementácia QR kódu

Pre autentifikáciu pomocou QR kódu sme sa pokúsili docieliť to, aby mal QR kód čo najmenej riadkov a stĺpov. Hlavný dôvod je ten, že čím je menej riadkov a stĺpcov, tým môžeme QR kód vytlačiť v menších rozmeroch. Aby bol QR kód bezpečný, tak musíme zvoliť vhodnú dĺžku náhodného reťazca, ktorý bude použitý ako heslo. Tento náhodný reťazec je zložený z malých písmen anglickej abecedy a z čísel od nula až po deväť. Pri nastavení veľkosti hesla na 13 sa spolu s dvojbodkou a ID používateľa vygeneruje QR kód verzie 2, čo prestavuje kód zložený z 25 riadkov a 25 stĺpcov. Nastavením dĺžky hesla na 13 docielime, že možných kombinácií všetkých hesiel je 36^{13} , čo po prepočte vyjde na 170 581 728 179 578 208 256 všetkých možností. Keďže sa predpokladá, že niektorí používatelia si QR kód iba vytlačia a nezabezpečia proti poškodeniu, tak sme sa rozhodli QR kódu nastaviť najvyššiu úroveň opravy chýb, čo predstavuje písmeno H. Pri tomto nastavení sa veľkosť QR kódu zvýši na verziu 3, čo predstavuje 29 riadkov * 29 stĺpcov. Táto veľkosť za zvýši kvôli tomu, že QR kód musí uchovať viac dát ako zálohu ako je spomínané v sekcii Východiská. Postupným poškodzovaním QR kódu sme sa dostali k tomu, že naša webová aplikácia prečíta a rozanalyzuje aj takýto kód na obr. 3.5.

Obr. 3.5: Poškodený QR kód

3.2.2 Prihlásenie metódou QR kódu

Pre prihlásenie pomocou QR kódu znova použijeme rozhranie getUserMedia(), kde tentokrát nastavíme, že požadujeme iba zapnutie web kamery. Rovnako ako pri práci s mikrofónom, aj tu sa sa prvej návšteve webovej aplikácie bude požadovať potvrdenie na prístup k web kamere. Takto spustené video vložíme do canvas elementu, ktorý slúži na vykreslenie. Týmto zabezpečíme, že používateľ vidí spätnú väzbu v reálnom čase a tým docielime to, že uvidí kde presne sa nachádza QR kód a prípadne ak časť kódu je mimo záberu kamery tak vie, že ho musí presunúť. Na analýzu QR kódu použijeme knižnicu JSQR.js na strane webového prehliadača. Po nastavení QR kódu pred kameru naša webová aplikácia analyzuje QR kód, po zanalyzovaní QR kódu sa video zatvorí a webová aplikácia kód preloží na reťazec a tento reťazec rozdelíme podľa dvojbodky na id a heslo. Toto id a heslo pošleme na server cez "/auth/qrcode", kde server skontroluje, či sa toto id nachádza v databáze a či má k nemu správne heslo. Ak áno, používateľ môže pokračovať v práci ako prihlásený, ak nie, vypíše sa chybová hláška. Aby sme takýto QR kód dokázali prečítať a zanalyzovať, je potrebné aby kamera vedela zachytiť všetky potrebné detaily. Pri vytlačení QR kódu vo veľkosti 1,5 centimatra dĺžka 1,5 centimetra výšky je potrebné tento QR kód umiestniť pred kameru približne vo vzdialenosti 5 centimetrov. Čím bude náš vytlačený QR kód vytlačený vo väčších rozmeroch, tým ho môžeme vzdialiť od kamery ďalej. Pri rozmeroch QR kódu 5 centimetrov do dĺžky a 5 centimetrov do výšky našej aplikácií postačí, aby používateľ nastavil kód približne 50 centrimetrov od obrazovky. Ak sa používateľ rozhodne používať mobilný telefón namiesto vytlačenej formy, snímač QR kódu nemusí vždy správne fungovať pretože pri nastavení slabého jasu na mobilnom telefóne je takmer nemožné rozdeliť QR kód na farby čiernu a bielu, čo znamená že webová aplikácia nevie určiť dáta z QR kódu. Pri nastavení príliš vysokého jasu na mobilnom telefóne nastáva problém, že jas je

niekoľkonásobne vyšší ako okolité svetlo, čo znamená, že display mobilného telefónu vyžaruje príliš veľa svetla a my môžeme vidieť, že naša kamera sníma tento QR kód veľmi rozmazane. Aj pre tento problém bolo vhodné použiť vykreslenie výstupu z kamery do obrazovky, pretože používateľ vidí, ako reaguje kamera na jeho QR kód a podľa potreby si môže zvýšiť alebo znížiť jas.

3.3 Registrácia verzus pridávanie metódy

V ďalších bodoch sa budeme venovať ostatným metódam, ktoré sú dostupné v našej webovej službe. Kým pre hlas a QR kód platilo, že si ich mohol pridať iba prihlásený užívateľ, pre ostatné metódy platí, že sa môže registrovať buď nový používateľ, alebo si ich chce pridať už existujúci používateľ. Na to, aby si mohol pridať tieto metódy musí byť prihlásený. To, či je momentálne prihlásený, zistíme, ak req.session.passport alebo req.session.passport.user nie je undefined.

3.4 Registrácia meno, heslo a email

Pri registrácií metódou menom, heslom a emailom využijeme podobné vlastnosti ako má väčšina terajších aplikácií. Týmito vlastnosťami sú, že používateľ musí zadať užívateľské meno, email, heslo a potvrdenie hesla. Na to, aby užívateľ mohol vkladať tieto údaje sme vytvorili formulár v HTML. Jednotlivé kolónky sú <label> elementy. Týmto elementom <label> vieme nastaviť, aký formát vstupu od používateľa do nich očakávame. Pre meno použijeme klasický formát `text`, pre email nastavíme formát na `email`, čo pomáha s kontrolou emailu. Ak teraz používateľ nezadá zaváňač a odošle formulár, webový prehliadač mu automaticky vypíše, aký formát emailu očakáva. Pre heslo sme zvolili formát "password", čo je potrebné kvôli bezpečnosti. Takto zvolený formát zabezpečí, že sa do kolónky nevypíše napísaný znak, ale je na miesto neho napísaná hviezdička. Takto nikto neuvidí, aké heslo je napísané a neuvidí to ani sám používateľ. Preto sa môže ľahko stať, že urobí preklep a následne sa nebude vedieť prihlásiť, preto sme pridali ďalšiu kolónku na potvrdenie hesla, kde ak sa heslá nerovnajú, vypíše sa chybová hláška. Minimálnu dĺžku mena a hesla sme nastavili na 6 znakov. V prípade, že používateľ vyplnil všetko správne sa na serveri toto heslo zašifruje pomocou knižnice bcrypt. Po zašifrovaní hesla musíme zistiť, či je používateľ prihlásený.

Ak áno, z momentálneho prihlásenia si vypýtame jeho hlavné id, ktoré používa v aplikácií. Následne uložíme do tabuľky LocalAccounts jeho meno, zašifrované heslo, získané hlavného id pre aplikáciu a hodnotu potvrdený email nastavíme na false. Po uložení do databázy sa užívateľovi pošle verifikačný link na potvrdenie, že sa chce registrovať. Po kliknutí na link sa užívateľovi hodnota potvrdeného emailu nastaví na

true.

Ak používateľ nie je momentálne prihlásený, preto predpokladáme, že si vytvára nový účet. Postup je rovnaký ako v predchádzajúcej časti, avšak tu najskôr vytvoríme nový účet pre našu aplikáciu v tabuľke Accounts, a následne sa id z tejto tabuľky nastaví ako `account_id` v tabuľke LocalAccounts.

```
this.passport.use("facebook" + (this.isRegistration ? "-reg" : ""),
    new strategy({
          clientID: "156146464654651651651651651",
          clientSecret: "abc1651651651ca165165c1a1651ca16516ca16516",
          callbackURL: "moja-stranka.sk/auth/facebook/callback",
          passReqToCallback: true,
        }
```

Listing 3.1: Inicializácia metódy Facebooku

3.5 Facebook a Google

Keďže registrácia a prihlasovanie pomocou Facebooku a Google funguje na rovnakom princípe, zhrnieme to do tejto sekcie. Predpokladáme, že vývojár už má zaregistrovanú svoju aplikáciu vo Facebooku aj v Googli. Keďže webová aplikácia využíva knižnicu socket.io, potrebujeme zabezpečiť, aby sa webová stránka, kde má užívateľ otvorenú našu webovú aplikáciu nikdy neobnovila. Avšak pri pridávaní týchto metód nastal problém, že keď používateľ klikne na prihlásenie pomocou Facebooku alebo Google, aplikácia otvorí novú záložku kde sa má používateľ prihlásiť. Preto sme využili schopnosť socketu a tou je pridávanie záložiek do miestností.

```
ioServer.on("connection", (socket) => {
  for (let i = 0; i < sessions.length; i++) {
    if (sessions[i].sessionId === socket.request.session.id) {
      sessions[i].sockets.push(socket.id);
    } else if (i === sessions.length) {
      sessions.push(makeNewObj(socket));
    }
  }
  if (sessions.length === 0) sessions.push(makeNewObj(socket));
  socket.join(socket.request.session.id);
  socket.on("disconnect", () => {
    for (let i = 0; i < sessions.length; i++) {
      if (sessions[i].sockets.includes(socket.id)) {
        delete sessions[i].sockets[sessions[i].sockets.indexOf(socket.
           id)];
      }
```

```
16      }});
```

Listing 3.2: Funkcia na vytvorenie socketu

Takto sme si zabezpečili, že každá nová záložka je v miestnosti spolu s našou webovou aplikáciou, takže teraz tieto záložky dokážu komunikovať o tom, či sa prihlásenie na Facebooku alebo Googli podarilo.

Aby sme mohli použiť knižnicu Passport.js, potrebujeme mu nainicializovať potrebné údaje.

ClientID a clientSecret sú údaje, ktoré sme získali zaregistrovaním svojej aplikácie v službe. CallBackUrl je url link, na ktorý nás Facebook presmeruje po úspešnom alebo neúspešnom prihlásení. Na to, aby sme mohli používať údaje z Facebooku v do funkcie, v ktorej riešime prihlásenie a registráciu, potrebujeme passReqToCallback nastaviť na true. Do tejto funkcie posielame callbackovú funkciu `cb`, cez ktorú sa nastavením na cb(null, false); neprihlásime. Naopak, ak používateľa nájdeme v databáze a všetko prebehne v poriadku, používateľa prihlásime príkazom cb(null, res1);

V nasledujúcom kroku musíme nastaviť, čo sa stane, keď aplikácia dostane get požiadavku na `/auth/facebook/callback`. V prípade, že používateľ sa vo Facebooku úspešne prihlásil, presmeruje ho na `/auth/success`. Ak nie, presmerujeme ho na `/auth/fail`.

```
1 this.express.get("auth/facebook/callback",
2     this.passport.authenticate("facebook"),
3         { failureRedirect: "/auth/fail",
4         successRedirect: "/auth/success" }
5       )
6     );
```

Listing 3.3: Presmerovanie po prihlásení na Facebooku

`/auth/success` a `/auth/fail` sú jednoduché stránky, ktoré slúžia práve pre spomínaný problém s presmerovaním stránky. Po načítaní stránky `/auth/fail` sa toto okno iba zatvorí.

```
1 window.addEventListener('load', (e) => {
2         window.close();
3     })
```

Listing 3.4: /auth/fail

Ak používateľa presmeruje na stránku `/auth/success`, znamená to, že sa mu podarilo prihlásiť vo Facebooku. Po tom, ako sa dostane na `/auth/success`, odošle sa cez socket správa na našu webovú aplikáciu príkazom `socket.emit('loggedIn');` a toto okno sa automaticky zatvorí.

```
1 window.addEventListener('load', (e) => {
2         var socket =  io();
```

```
3         socket.emit('loggedIn');
4         socket.on('closeWindowConfirm', () => {
5              window.close();
6          });
7     })
```

Listing 3.5: /auth/success

Zdroje a použitá literatúra

[1] CROCKFORD, Douglas. Javascript:The Good Parts. YAHOO! PRESS, 2008. ISBN: 978-0-596-51774-8.

[2] Voice biometrics: The voice print will become online banking's greatest ally, Bank Services, Banco Bilbao Vizcaya Argentaria,21 august 2020. Citované dňa: 7. február 2021. Dostupné na: https://www.bbva.com/en/voice-biometrics-the-voice-print-will-become-online-bankings-greatest-ally/

[3] D Hardt, Ed. The Oauth 2.0 Authorization Framework. Microsoft 2012. ISSN:2070-1721

[4] Joseph, Kulandai, Java Facebook Login with OAuth Authentication, Javapapers, 16. október 2014. Citované dňa: 7.február 2021. Dostupné na: https://javapapers.com/java/java-facebook-login-with-oauth-authentication/

[5] How to Make a QR Code | Creating QR Codes, Scott, SqroutQR, 3. september 2020: Citované dňa: 21.január 2021. Dostupné na: https://www.sproutqr.com/blog/how-to-make-a-qr-code

[6] SAML2 SSO Integration,14. máj 2021. Dostupné na: https://www.ibm.com/docs/en/essm/10.1.3?topic=configuration-saml2-sso-integration

[7] MediaDevices.getUserMedia(),14.máj 2021. Dostupné na: https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia

[8] Web Audio API ,14.máj 2021. Dostupné na: `https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API`