Spojená škola, Komárňanská 28, Nové Zámky, 940 75

o.z. Stredná priemyselná škola elektrotechnická- S.A.Jedlika

Jedlik Ányos Elektrotechnikai Szakközépiskola

**„Bezpečnostné protokoly bezdrôtových sietí  
– učebná pomôcka“**

Vlastný projekt

Praktická časť odbornej zložky maturitnej skúšky

Nové Zámky riešiteľ:

2018 **Andrej Szalma**

ročník štúdia: **štvrtý**

konzultant:

Ing. Michal Miko

Čestné vyhlásenie

Vyhlasujem, že som túto prácu vypracoval samostatne s pomocou konzultanta   
Ing. Michal Miko a uviedol som všetku použitú literatúru.

..............................................

Poďakovanie

Chcel by som sa poďakovať môjmu konzultantovi práce, Ing. Michalovi Mikovi, za vedenie a cenné pripomienky pri záverečnom spracovaní práce.

Obsah

[Úvod 6](#_Toc2422767)

[1 Cieľ práce 7](#_Toc2422768)

[2 Problematika a prehľad literatúry 8](#_Toc2422769)

[2.1 Android 8](#_Toc2422770)

[2.2 Tvorba aplikácií na Android 8](#_Toc2422771)

[2.2.1 Natívne aplikácie 8](#_Toc2422772)

[2.2.1.1. Začiatočník alebo krátkodobé skúsenosti s vývojom aplikácií 9](#_Toc2422773)

[2.2.1.2. Skúsenosti s vývojom webových stránok 9](#_Toc2422774)

[2.2.2 Webové aplikácie 9](#_Toc2422775)

[2.2.3 Hybridné aplikácie 9](#_Toc2422776)

[2.3 Bezdrôtové technológie a ich bezpečnostné protokoly 10](#_Toc2422777)

[2.3.1 IEEE 802.11 10](#_Toc2422778)

[2.3.2 Šifrovanie 10](#_Toc2422779)

[2.3.3 WEP 11](#_Toc2422780)

[2.3.4 WPA 12](#_Toc2422781)

[2.3.4.1. WPA 12](#_Toc2422782)

[2.3.4.2. WPA2 13](#_Toc2422783)

[3 Návrh a vývoj aplikácie Felix 15](#_Toc2422784)

[3.1 Návrh dizajnu aplikácie 16](#_Toc2422785)

[3.1.1 Návrh palety farieb 16](#_Toc2422786)

[3.1.2 Návrh postavičky Felix 16](#_Toc2422787)

[3.1.3 Návrh rozloženia aplikácie 17](#_Toc2422788)

[3.2 Výskum a štúdium 18](#_Toc2422789)

[3.2.1 Výskum bezdrôtových sietí a ich bezpečnostných protokolov 18](#_Toc2422790)

[3.2.1.1. 1. Kapitola – Základné pojmy 18](#_Toc2422791)

[3.2.1.2. 2. Kapitola – IEEE 802.11 18](#_Toc2422792)

[3.2.1.3. 3. Kapitola – Šifrovanie 18](#_Toc2422793)

[3.2.1.4. 4. Kapitola – Ako funguje WEP? 18](#_Toc2422794)

[3.2.1.5. 5. Kapitola – Ako funguje WPA? 18](#_Toc2422795)

[3.2.2 Štúdium jazyka JavaScript a framework-u React-Native 18](#_Toc2422796)

[3.3 Programovanie Felix-a 19](#_Toc2422797)

[3.3.1 Front-end 19](#_Toc2422798)

[3.3.2 Back-end 19](#_Toc2422799)

Úvod

V dnešnom svete sa každý snaží o odstránenie fyzických spojení, či v odvetví sietí alebo všemožných technológií. Celkom sa nám to aj darí. Aktuálne, každý človek vlastní v priemere 4 zariadenia pripojené k internetu. V roku 2020 je očakávané množstvo 6 zariadení na osobu. Vývoj a modernizácia technológií je veľmi dôležitý aspekt našich životov, ale rovnako dôležité je aj vzdelanie a gramotnosť v oblasti bezpečnosti všemožných technológií.

Pre toto som sa rozhodol, že spracujem tému bezpečnosti bezdrôtových technológií a vytvorím pomôcku vo forme mobilnej aplikácie, ktorá užívateľa uvedie do danej problematiky a bude vzdelávať v danej oblasti.

Nakoľko našou budúcnosťou je generácia mladých ľudí, ktorý si život bez smartfónov a inteligentných zariadení ani nevedia predstaviť a s knihami sa stýkajú čoraz menej, som si uvedomil, že najlepším spôsobom ako zasiahnuť túto cieľovú skupinu je pomocou už vyššie spomenutých smart-zariadení. Z tohto vyplýva aj moje rozhodnutie vytvoriť učebnú pomôcku vo forme mobilnej aplikácie.

Výber názvu aplikácie a štýlu, akým bude informácie interpretovať nebolo vôbec ľahké, ale rozhodol som sa, že chcem vytvoriť aplikáciu, ktorá bude vystupovať ako „kamarát“ užívateľa a nie len ako statický učebný materiál. A preto som vytvoril postavičku Felix, po ktorej som aj nazval aplikáciu. Je to milá, sympatická osoba menom, ktorá je tu pre užívateľa, aby ho naučila základy bezpečnosti bezdrôtových sietí a uviedla do tejto problematiky. Jej cieľom nie je do podrobností vysvetľovať všetky možné bezpečnostné princípy, ale chce užívateľa oboznámiť so základným fungovaním šifrovania a bezpečnostných protokolov.

/\*\* Tak isto aj upozorniť na hrozby ktoré číhajú na každého, kto sa pripojí na sieť Wi-Fi. \*/

1. Cieľ práce

Cieľom mojej práce je predovšetkým zvýšiť úroveň vzdelanosti ľudí o bezpečnosti na bezdrôtových sieťach. Taktiež ako poskytnúť študentom a každému so záujmom o informačné technológie zaujímavú pomôcku na rozšírenie svojich znalostí. Nakoľko by som chcel aby bola táto aplikácia dostupná pre všetkých, som sa rozhodol jej kompletný obsah písať v anglickom jazyku.

Mojim ďalším cieľom je zvýšiť úroveň môjho vzdelania ako v oblasti Sietí tak aj Programovania. Nakoľko s vývojom mobilných aplikácií nemám žiadne skúsenosti, je táto práca pre mňa obrovská výzva, avšak bude mať pre mňa tým väčší prínos.

1. Problematika a prehľad literatúry
   1. Android

Android už dnes vôbec nie je cudzie slovo. Spájame ho hlavne so smartfónmi, či tabletmi. Android tak, ako ho poznáme, je v prvom rade operačný systém. Platforma Android je taktiež balíčkom pre vývojárov softvérových aplikácií. Čo napadne ako prvé vám, keď sa povie slovo „Android“?

Android je zatiaľ najmladší operačný systém, ktorý je používaný na mobilných telefónoch. Ide o systém, ktorý bol vyvinutý na báze Linuxu firmou Android Inc. . Tá bola neskôr odkúpená spoločnosťou Google. Výhodou operačného systému od Googlu je to, že je poskytovaný ako open-source systém. Populárnym sa stal aj vďaka svojmu intuitívnemu prostrediu či množstvu praktických funkcií, ktoré poskytuje. Rozšírenosť androidových telefónov sa rozrastá aj vďaka Google Play - obchodu s aplikáciami ako aj možnosťou inštalácie aplikácií tretích strán. V súčasnosti totiž len Google Play ponúka cez 2 100 000 aplikácií, pričom väčšina z nich je dostupná bezplatne.

* 1. Tvorba aplikácií na Android

Na tvorbu aplikácií na telefóny s operačným systémom Android je v dnešnej dobe veľké množstvo spôsobov a programovacích jazykov. Pri výbere, ktorým smerom sa vydať pri vývoji svojej aplikácii, je dôležité poznať pozitíva a negatíva rôznych spôsobov.

Hlavné dva typy aplikácií na mobilné platformy sú:

* Natívne aplikácie
* Webové aplikácie
* Hybridné aplikácie
  + 1. Natívne aplikácie

Ich názov opisuje aj ich funkčnosť. Sú to aplikácie ktoré sú natívne pre Android/iOS a majú možnosť používať všetky vstavané funkcie týchto platforiem. Je to najrozšírenejší spôsob tvorby mobilných aplikácií, ale taktiež aj najefektívnejší nakoľko aplikácie vie využiť zariadenie naplno bez obmedzení. Donedávna bol tento spôsob vývoju striktne určený len na jednu platformu, no dnes je už možné vyvíjať aj cross-platform natívne aplikácie. Pri rozhodovaní sa aký jazyk si vybrať na vývoj, je dôležité prihliadať na predošlé skúsenosti s programovaním.

* + - 1. Začiatočník alebo krátkodobé skúsenosti s vývojom aplikácií

V prípade, že je človek úplný nováčik v odbore softvérového inžinierstva by bol asi „Android Software Development Kit“ v jazyku „Java“ najvhodnejší nakoľko je to rokmi overený spôsob a je naň nespočetné množstvo materiálov a dokumentácie, nehovoriac o množstve kurzov pre začiatočníkov napr. na stránke [www.udemy.com](http://www.udemy.com)

* + - 1. Skúsenosti s vývojom webových stránok

V prípade, že sa vývojár webových stránok rozhodne pre vývoj natívnej aplikácie je pravdepodobné, že už má určité skúsenosti s jazykmi ako HTML, CSS a JavaScript. Ak je tak, v tom prípade je určite odporúčané vyskúšať framework „React-Native“ v skriptovacom jazyku „JavaScript“, v ktorom sa používa okrem „JavaScriptXML“ v skrat. „JSX“ s podobnou syntaxou ako „HTML“ a na štýlovanie sa používa „CSS“ v trocha pozmenenej podobe. Jeho hlavným pozitívom je však že je cross-platform a teda aplikáciu je možné s minimálnymi zmenami v kóde používať ako na Android tak aj na iOS.

* + 1. Webové aplikácie

Webové aplikácie sú webové stránky, ktoré boli navrhované a vyvíjané primárne na použitie na mobilných telefónoch, t.j. sú responzívne. Tieto aplikácie bežia na všetkých platformách, či na mobilných telefónoch alebo na počítačoch nakoľko musia bežať v prehliadači, čo je aj ich hlavnou nevýhodou. Nie sú také interaktívne a intuitívne ako natívne a sú pomalšie, resp. neplynulejšie nakoľko ich beh závisí hlavne od rýchlosti internetu a nie od výkonu zariadenia. No majú aj svoje výhody a to že tento spôsob je z troch spomenutých asi najjednoduchší nakoľko je dostačujúca znalosť jazykov „HTML, CSS a JavaScript“ k vytvoreniu celej aplikácie, no je možnosť použitia takmer hocijakých technológií a jazykov, takže možnosti sú neobmedzené.

* + 1. Hybridné aplikácie

Tieto aplikácie sú konjunkciou webových a natívnych aplikácií nakoľko sú založené na technológiách používaných pri vývoji webových stránok, no nebežia v prehliadači, ale zväčša v nejakom kontajneri ako je „webview“. Hybridné aplikácie sú cross-platform ako aj webové a vačšinou majú možnosť aj vyžívať vstavané funkcie, senzory zariadenia pomocou vstavaných API. Medzi tieto aplikácie sa z časti radí aj už vyššie spomenutý „React-Native“, ktorý je vyvíjaný pomocou webových technológií, ale však aplikácie v tomto framework-u sa nezobrazujú pomocou „webview“, ale sa kompilujú do natívnych komponentov a z toho dôvodu sa radí viac medzi natívne aplikácie ako hybridné. Medzi nevýhody hybridných aplikácií patrí nižší výkon ako u natívnych aplikácií a sú taktiež menej interaktívne ako webové aplikácie. Avšak majú zďaleka viac výhod, čím sú aj rýchlosť vývoju, cross-platform, nie je potreba webového prehliadača ani prístupu na internet a prístup k funkciám zariadenia pomocou je možný vstavaných API.

* 1. Bezdrôtové technológie a ich bezpečnostné protokoly
     1. IEEE 802.11

IEEE 802.11 alebo 802.11x alebo wi-fi štandard je súbor štandardov Wireless LAN vyvinutých pracovnou skupinou 11 Normalizačného výboru IEEE pre LAN/WAN (IEEE 802).

Rodina 802.11 v súčasnosti obsahuje šesť úprav techník komunikácie vzduchom, ktoré využívajú ten istý protokol. Najpopulárnejšie sú techniky definované pozmeňujúcimi návrhmi b, a g pôvodného štandardu, ktorý pôvodne zahŕňal aj otázku bezpečnosti a neskôr bola prepracovaná v pozmeňovacom návrhu 802.11i. Ostatné štandardy rodiny (c-f, h-j, n) sú servisné vylepšenia, rozšírenia alebo opravy pôvodnej špecifikácie. 802.11b bol prvý všeobecne prijatý štandard. Nasledovali 802.11a a 802.11g.

Štandardy 802.11b a 802.11g používajú frekvenciu 2,4 GHz, ktorá môže byť rušená mikrovlnnými trúbami, bezdrôtovými telefónmi, Bluetooth zariadeniami a inými aplikáciami využívajúcimi túto frekvenciu. Štandard 802.11a už používa rozsah 5 GHz a preto nie je ovplyvňovaný zariadeniami z rozsahu 2,4 GHz.

(Wikipedia, 2018)

* + 1. Šifrovanie

Základným prostriedkom na utajenie správ v sieti je šifrovanie. Šifrovací algoritmus je matematická metóda, ktorou sa konvertujú čitateľné údaje do nečitateľnej podoby pomocou určitého šifrovacieho kľúča. Tento kľúč je potrebný nielen na úspešné zašifrovanie správy ale taktiež ho treba na dešifrovanie. Teda nám nevzniká otázka ako to zašifrujem, ale ako bezpečne predať šifrovací kľúč prijímateľovi. Aj vďaka tomuto dôvodu boli vyvinuté dva druhy šifrovacích algoritmov a to symetrické a asymetrické. Symetrické šifrovanie využíva jeden súkromný kľúč a a asymetrické šifrovanie používa dva kľúče – verejný a súkromný.

Symetrické šifrovanie sa používa na šifrovanie informácií a asymetrické na distribúciu symetrických kľúčov.

Pri Symetrickom šifrovaní obe strany komunikácie zdieľajú rovnaký kľúč. Používajú ho na šifrovanie aj dešifrovanie prenášaných údajov. Šifrovanie týmto spôsobom sa dá nielen použiť na utajovanie správ ale aj ako autentizácia. Distribúcia kľúča je hlavné omedzenie používania súkromného kľúča. Pokiaľ je statický, je ľahko možné, že ho bude poznať viac ľudí než by sme si priali. Preto je dobre ak sa súkromný kľúč často mení.

Pri Asymetrickom šifrovaní sú dva rôzne kľúče, ktoré sú ale vzájomne kompatibilné. Jeden kľúč je verejný a druhý súkromný. Verejným kľúčom sa dáta šifrujú a súkromným dešifrujú. Verejný kľúč je dostupný komukoľvek, s kým komunikujeme, no súkromný by mal ostať prísne utajený u vlastníka. Čo bolo zašifrované verejným kľúčom ide dešifrovať súkromným a to platí aj naopak. Jedným kľúčom sa nedá niečo zašifrovať a dešifrovať. Princíp funguje na matematickej funkcií, ktorej reverzný výpočet sa nedá prakticky previesť. Asymetrické šifrovanie slúži na ochranu prenášaných dát, ale nie k autentifikácie odosielateľa, keďže sa používa verejný kľúč. V porovnaní so symetrickými šifrovacími metódami sú asymetrické výraznejšie pomalšie, pretože metódy a protokoly na nich fungujúce používajú komplikované operácie s veľkými číslami. Preto väčšina systémov šifruje dáta symetrickými metódami a asymetricky sa zašifruje len použité symetrické kľúče. Tu príjemca najprv dešifruje symetrický kľúč a až potom pomocou neho zašifrované dáta. Pri zmene súkromného kľúča sa vždy vygeneruje odpovedajúci verejný kľúč.

(Macuľa, 2010)

* + 1. WEP

WEP je protokol, ktorý riadi prístup k sieti a zabezpečuje prenos informácií. Predstavuje ekvivalent takej bezpečnej komunikácie ako v LAN sieťach. Napovedá tomu aj jeho názov Wired Equivalent Privacy, čo po preklade môže znieť ako “Súkromie ekvivalentné drôtovej komunikácií“, Má teda slúžiť na autentizáciu a ochranu dát šifrovaním s rovnakým tajným kľúčom, čiže sa jedná o symetrické šifrovanie.

Autentizácia pri WEP prebieha buď otvorene – Open systems, alebo pomocou zdieľaného kľúča – Shared-key.

Pri tejto autentizácie pomocou zdielaného kľúča sa používa kľúč o veľkosti 40 bitov, ktorý je rovnaký pre všetkých užívateľov v danej sieti. Pri jeho zadaní sa neoveruje totožnosť užívateľa ale jeho sieťovej karty. Autentizácia prebieha tak, že užívateľ odošle požiadavku o autentizáciu zdieľaným kľúčom, na čo mu prístupový bod odošle údaj, ktorý má užívateľ zašifrovať svojim kľúčom a odoslať nazad. Na základe tejto informácie prístupový bod buď odmietne alebo prijme žiadateľa do bezdrôtovej siete, podľa toho či sa hodnota od užívateľa rovná hodnote, ktorú vygenerovalo AP. Požiadavka je vlastne rámec 802.11, ktorý obsahuje identifikačné údaje a žiadosť o autentizáciu.

Pri otvorenej autentizácii, ako už názov naznačuje nejde o autentizáciu, ktorá by identifikovala žiadateľa o pripojenie. Pri tejto autentizácií sa akýkoľvek užívateľ môže pripojiť k prístupovému bodu. Žiadateľ odošle autentizačný rámec so svojimi identifikačnými údajmi, na základe ktorého ho prístupový bod pridruží k sieti.

WEP využíva symetrické šifrovanie, čo znamená, že šifrovanie a dešifrovanie sa robí rovnakým algoritmom s rovnakým kľúčom. Prenášané dáta medzi prístupovým bodom a užívateľom sú šifrované pomocou 64 alebo 128bitového kľúča. Tento kľúč sa skladá z inicializačného vektora IV, ktorý tvorí 24 bitov kľúča a užívateľského kľúča, ktorý tvorí 40 alebo 104 bitov šifrovacieho kľúča. Slúži na zväčšenie možných kombinácií kľúča ( IV môže nadobudnúť 224 možných hodnôt ). Inicializačný vektor generuje odosielateľ, ktorý ho použije na vytvorenie šifry a odošle ho v záhlaví paketov. Prijímateľ použije inicializačný vektor z prijatých paketov / rámcov na spojenie s WEP kľúčom a dešifruje prijaté údaje. IV sa skladá z 32bitového poľa, kde 2bity zaberá identifikátor kľúča ( KeyID ), ktorý slúži na výber kľúča. 6 bitov IV sa používa ako výplň a vektor zaberá zvyšných 24 bitov.

Na šifrovanie WEP využíva symetrickú prúdovú šifru RC4 ( Ron's Code No. 4 ). Prúdová šifra znamená, že z kľúča určenej dĺžky sa vytvorí šifrovací reťazec tak, aby každý bit textu odpovedal jednému bitu šifry. Vo Wi-Fi sa využíva RC4 o veľkosti 40 bitov. Inicializačný vektor vznikne použitím generátora pseudonáhodných čísiel

(Macuľa, 2010)

* + 1. WPA
       1. WPA

Protokol WEP bol považovaný za nedostatočný zabezpečovací mechanizmus bezdrôtovej siete, preto spoločnosť Wi-Fi Alliance prišla s riešením WPA v roku 2002. WPA bolo ako dočasné riešenie, kým nebude schválená bezpečnostná norma 802.11i, kde predstavuje jeho podmnožinu. Bolo braté do úvahy hlavne aby, keď vyjde 802.11i, bol rozšírený hardvér ktorý by bol schopný toto zabezpečenie vykonávať. 802.11i bolo prijaté ako oficiálny štandard 24. Júna 2004.

WPA už narozdiel od protokolu WEP dokáže autorizovať užívateľov a rieši distribúciu šifrovacích kľúčov. Každé zariadenie obdrží priradený šifrovací kľúč, ktorý je dynamicky 26 generovaný. WPA poskytuje dve autentizačné schémy: PSK mode ( Pre-Shared Key ) a Enterprise mode

PSK mode je v podstate jednoduchší režim používajúci 256bitový prednastavený kľúč, ktorý je určený hlavne pre domáce siete. Kľúč je heslo, ktoré pozostáva z 8 až 63 znakov ASCII kódu alebo 64 hexadecimálnych číslic. Heslo by malo byť čo najzložitejšie – najlepšie by malo obsahovať 22 náhodných znakov, čo by zaručovalo vysokú bezpečnosť pretože komunikácia sa šifruje na jeho základe.

Enterprise mode sa využíva predovšetkým vo väčších sieťach ( firemných ), kde sa vyžaduje autentizácia užívateľov. Tu ale je potrebné mať nastavený autentizačný server – systém RADIUS. Server RADIUS je samostatné zariadenie, alebo ak sa v sieti takéto zariadenie nenachádza, dá sa prístupový bod nakonfigurovať na funkciu RADIUS-u. Autentizačný proces funguje na princípe klient-server.

Pre šifrovanie sa používa algoritmus RC4, z dôvodu spätnej kompatibility. Šifrovací kľúč ma dĺžku 128 bitov, z čoho inicializačný vektor tvori 48 bitov a zvyšných 80 bitov predstavuje tajný kľúč, ktorý sa ale každým paketom mení. Nato slúži protokol TKIP ( Temporal Key Integrity Protocol ). Inicializačný vektor tentoraz ale taktiež uchováva poradové číslo paketu a používa sa na mixovanie kľúčov pre pakety. Najprv sa zmieša 32bitov IV, MAC adresa a dočasný kľúč relácie. Výstup sa zmieša s zvyšnou 16bitovou častou IV a dočasným kľúčom. Jedná sa o hash funkciu, ktorá vytvorí kľúč pre paket ( PPK, Per Packet Key ), ktorý sa následne zašifruje pomocou RC4 rovnako ako pri WEP.

(Macuľa, 2010)

* + - 1. WPA2

Tento protokol bol prijatý v roku 2004. Navrhnutý je pre čo najväčšiu možnú poskytovanú bezpečnosť siete. Označuje sa RSN - “Robust Security Network“. WPA2 úplne nahradzuje protokol WEP, kvôli čomu je spätne nekompatibilná so staršími sieťovými zariadeniami.

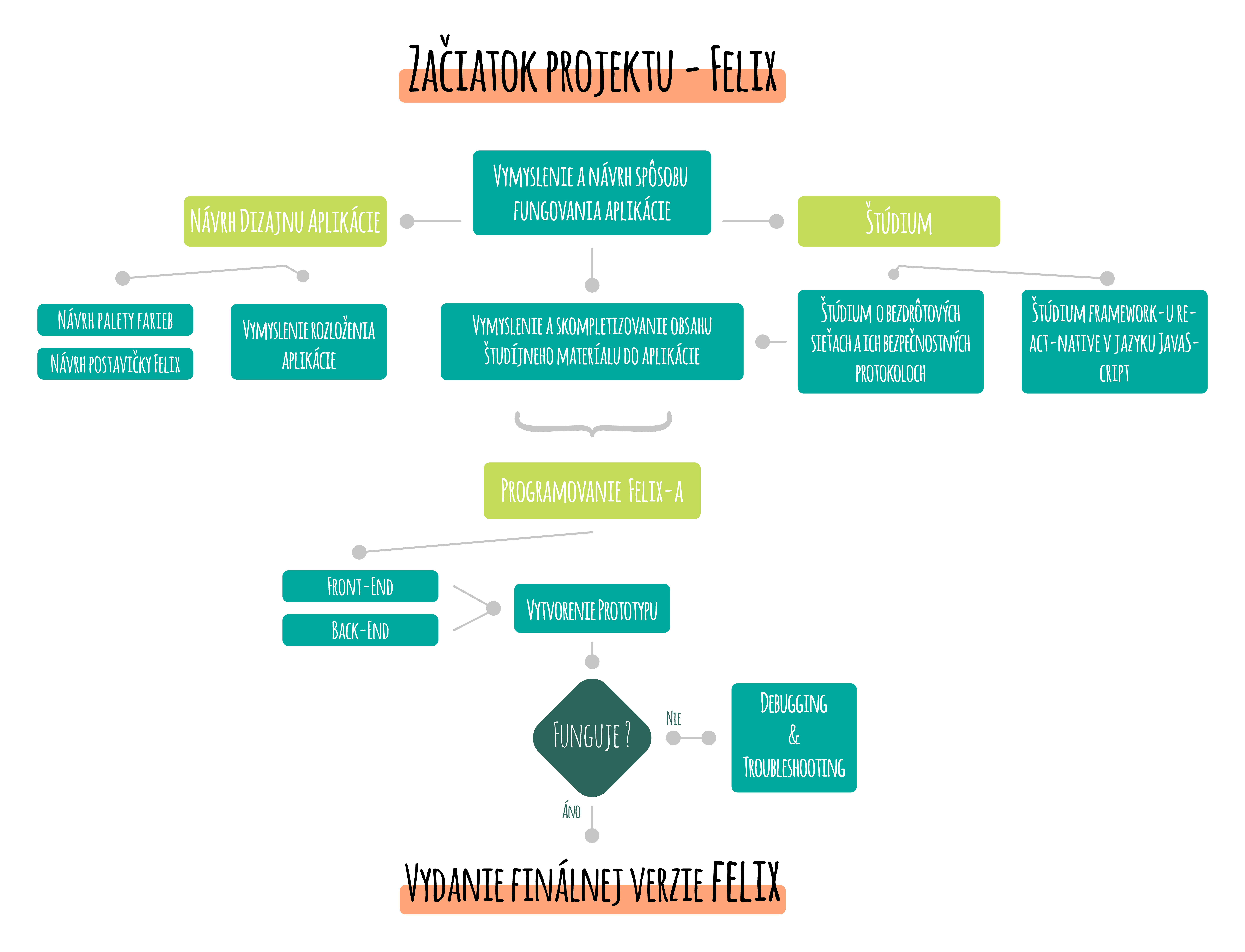
Autentizácia sa poskytuje rovnaká ako pri WPA.

Na šifrovanie sa používa algoritmus AES v podobe protokolu CCMP ( Counter-mode CBC ), ktorý používa dynamicky generovaný 128bitový kľúč. CCMP rozšíri rámec o16 bytov. Pripája počítadlo PN ( Packet Number – v podstate ide o IV ). CCMP obsahuje CBC-MAC ( Cipher Block Chaining-MAC ) označovaný taktiež AES-CCMP. CBC-MAC vytvára 64- bit MIC rámcu, ktorý vznikne zreťazením 128bitových blokov dát. Kľúče na šifrovanie a ochranu dát sú dynamicky menené pre každý rámec, ako pri WPA. Taktiež sa používajú dva typy kľúčov. Pre unicast PTK ( Pairwise Transient Key ) a broadcast GTK ( Group Transient Key )

(Macuľa, 2010)

1. Návrh a vývoj aplikácie Felix

Na začiatok by som chcel pomocou vývojového diagramu vysvetliť môj postup pri práci na tomto projekte. Následne ho slovne opíšem.



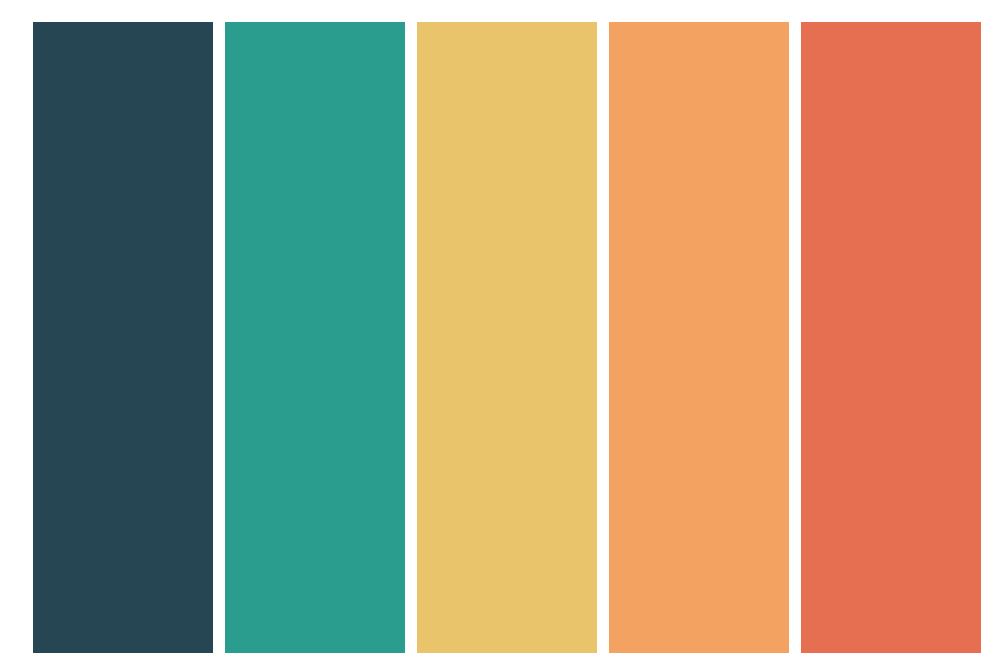
* 1. Návrh dizajnu aplikácie

Pri grafickom návrhu mojej aplikácie som používal software Adobe Illustrator CC a snažil som sa držať najmodernejších spôsobov a štandardov.

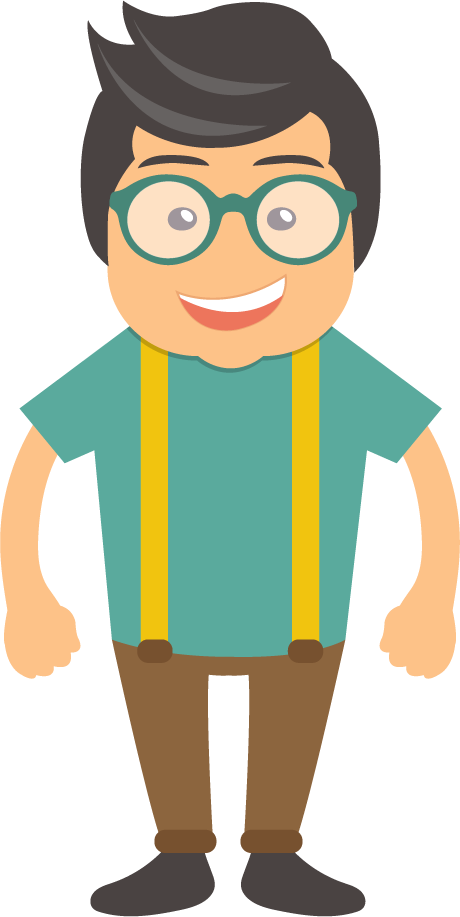
* + 1. Návrh palety farieb

Ako úplne prvý krok som si sa rozhodol zvoliť si paletu farieb, ktoré budem používať pri návrhu dizajnu mojej aplikácie. Vybral som päť farieb a používal som aj ich odtiene. Pri výbere som si pomáhal on-line nástrojom [www.coolors.co](http://www.coolors.co)

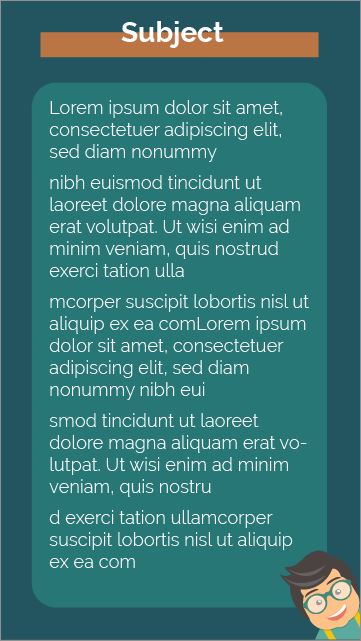
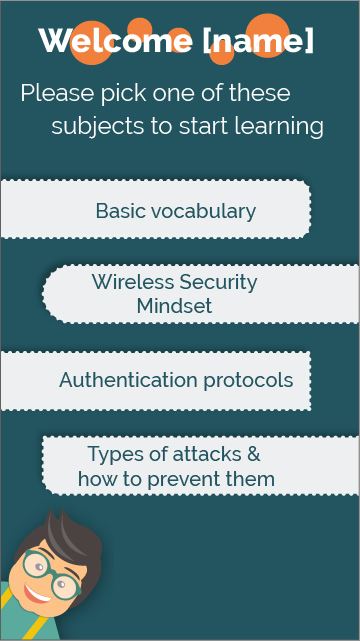
Obrázok 1Navrhnutá paleta farieb



* + 1. Návrh postavičky Felix

Ďalším krokom bol návrh sprievodcu aplikácie – Felix-a. Moja primárna predstava bola aby to bol „hipster“ nakoľko to je v dnešnom svete veľmi obľúbené a hlavne vo vekovej skupine od 16 – 30 rokov, presne tá ktorú som sa mojim projektom snažil zaujať. Nakoľko nemám dostatočné skúsonosti s ilustrovaním a kreslením postavičiek, tak som hľadal inšpiráciu na internete. Po čase sa mi podarilo vytvoriť finálnu podobu Felix-a. 

* + 1. Návrh rozloženia aplikácie

Rozhodol som sa pre veľmi jednoduchý a čitateľný spôsob rozloženia, ktorý je moderný a príjemný pre oko. Navrhol som jednotlivé stránky aplikácie tak aby na nich nebolo príliž veľké množstvo informácií a neodstrašilo to používateľa. Podľa mojich predošlých skúsenstí a pozorovaní som zistil, že ak je používateľom aplikácie mladší človek veku okolo pätnásť rokov, tak by ho veľké množstvo informácií mohlo odradiť od pokračovania používania aplikácie a zároveň vzdelávania sa v danej oblasti.

* 1. Výskum a štúdium

V tejto kapitole sa budem venovať tomu ako a odkiaľ som získaval informácie, ktoré budem nasledovne interpretovať v mojej aplikácii a opíšem moje spôsoby vzdelávania sa o skriptovacom jazyku JavaScript a framework-u React-Native.

* + 1. Výskum bezdrôtových sietí a ich bezpečnostných protokolov

Po rozsiahlom vyhľadávaní na internete, som zistil, že pravdepodobne najlepšími zdrojmi informácií do mojej aplikácie budú knihy, nakoľko je nedostatok web-stránok, ktoré sa zaoberajú teóriou o bezpečnosti bezdrôtových sietí. Na výber som mal nespočetné množstvo kníh, ale zaujala ma hlavne jedna, ktorá bola písaná moderným štýlom, ktorý by mohol zaujať. Je to kniha „A begginers guide - Wireless Network Security“ od autora Tyler Wightson. Po preštudovaní tejto knihy som sa rozhodol, že študijné podklady do mojej aplikácie budem získavať kombináciou mojich znalostí a materíálov z tejto knihy.

Aplikácia Felix je rozdelená na 5 kapitol, ktoré som už z teoretického hľadiska opísal v predošlých kapitolách tejto práce. Rozhodol som sa, že množstvo informácií popísaných v týchto piatich kapitolách je dostatočné, nakoľko je moja práca určená pre laikov v oblasti bezpečnosti bezdrôtových sietí a má užívateľa len uviesť do problematiky.

* + - 1. 1. Kapitola – Základné pojmy

Nakoľko je aplikácia Felix určená pre laikov v tejto problematike, tak som sa rozhodol že prvou kapitolou bude objasnenie základných pojmov v odvetví bezdrótových sietí. Sú to slová a pojmy ako „SSID, Access point, Beacon, ...“.

* + - 1. 2. Kapitola – IEEE 802.11
      2. 3. Kapitola – Šifrovanie
      3. 4. Kapitola – Ako funguje WEP?
      4. 5. Kapitola – Ako funguje WPA?
    1. Štúdium jazyka JavaScript a framework-u React-Native

Okrem vyššie uvedených dôvodov som sa pre tento jazyk rozhodol aj lebo je momentálne najmodernejší a najpoužívanejší skriptovací jazyk. Používa sa na všetkých platformách a je pravidelne vylepšovaný a vyvíjaný. S týmto jazykom som už mal určité predošlé skúsenosti nakoľko sa už tri roky venujem vývoju webových stránok, no nikdy to nebola moja silná stránka.

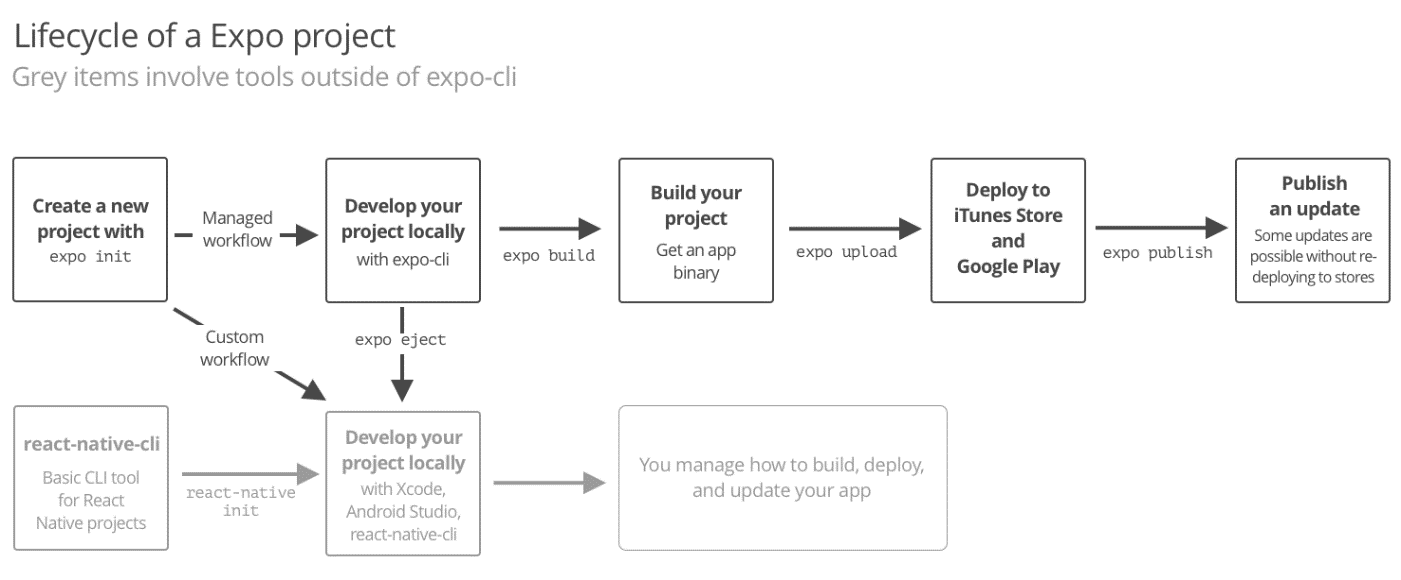
React-Native je framework vyvinutý firmou Facebook určený na vývoj natívnych cross-platform aplikácií na mobilné telefóny. Práca s týmto framework-om je stručne zdokumentovaná v oficiálnej dokumentácii na [www.facebook.github.io/react-native/](http://www.facebook.github.io/react-native/). Táto dokumentácia je vhodná pre ľudí už s predošlými skúsenosťami s jazykom JavaScript. Pre mňa to nebolo vôbec ľahké no bol to jeden z mála overených zdrojov. Tým sa dostávam aj k negatívam, ktoré som zistil až po začatí štúdia a to je nedostatok zdrojov, nakoľko je React-Native nový framework. O to ťažšia bola aj moja práca, keďže na riešenie mojich problémov som musel väčšinou prísť sám, alebo nájsť alternatívne riešenie.

* 1. Programovanie aplikácie Felix

Programovanie mobilných aplikácií v React-Native je naoko zložitý proces, ale po pochopení to nie je vôbec také ťažké. React-native používa natívne komponenty ako bloky z ktorých je aplikácia stavaná, takže predstavuje aj pár nových konceptov, ktoré je potrebné pochopiť pri práci s ním. To sú napr. JSX, componenty, „state“ a „props“, atď.

* + 1. Expo SDK

Pri práci s React-nativom je možné vyvíjať aplikácie alebo pomocou Android SDK (Software Developement Kit) alebo Xcode v prípade IOS, ale však tento spôsob je vhodný pri práci s komplikovanými aplikáciami s rozsiahlym Back-endom.

Ja som sa rozhodol vyvíjať pomocou Expo SDK, čo je vlastne zbierka knižníc pre Android aj IOS napísané natívne na obe platformy a vďaka tomu poskytuje prístup k natívnym komponentom zariadení s týmito OS. Natívne komponenty sú napr. Kamera, kontakty, lokálne úložisko, „Push“ notifikácie, a ostatý hardware zariadenia a komponenty OS. V nasledujúcom obrázku je vidieť tzv. „workflow“ vývoju aplikácie pomocou Expo SDK.

Veľkou výhodou Expo SDK je pre mňa aj možnosť updatovania aplikácie aj po vydaní do obchodov Google Play, alebo iTunes bez toho aby bolo potrebné aplikáciu znova kompilovať a nahrávať do týchto obchodov. Pokiaľ sa jedná o menšie zmeny, alebo o zmeny, ktoré nemenia štruktúru fungovania aplikácie, tak je možné len aplikáciu znova publikovať na Expo servery a užívateľom sa bez hocijakých updateov po najbližšom otvorení aplikácie otvorí najnovšia verzia. Pre mňa je to výhodou z dôvodu, že môžem pridávať učebné materiály a zlepšovať ich a následne takto jednoducho updatenúť aplikáciu.

* + 1. Inicializácia projektu

Za predpokladu že už je Node 8+ nainštalovaný na počítači vývojára, je inicializácia projektu pomocou Expo veľmi jednoduchá. Stačí nainštalovať Expo CLI pomocou jedného príkazu a následne pomocou ďalších pár príkazov inicializovať projekt a spustiť Expo vývojársku konzolu.

npm install -g expo-cli *//Inštalácia Expo CLI*

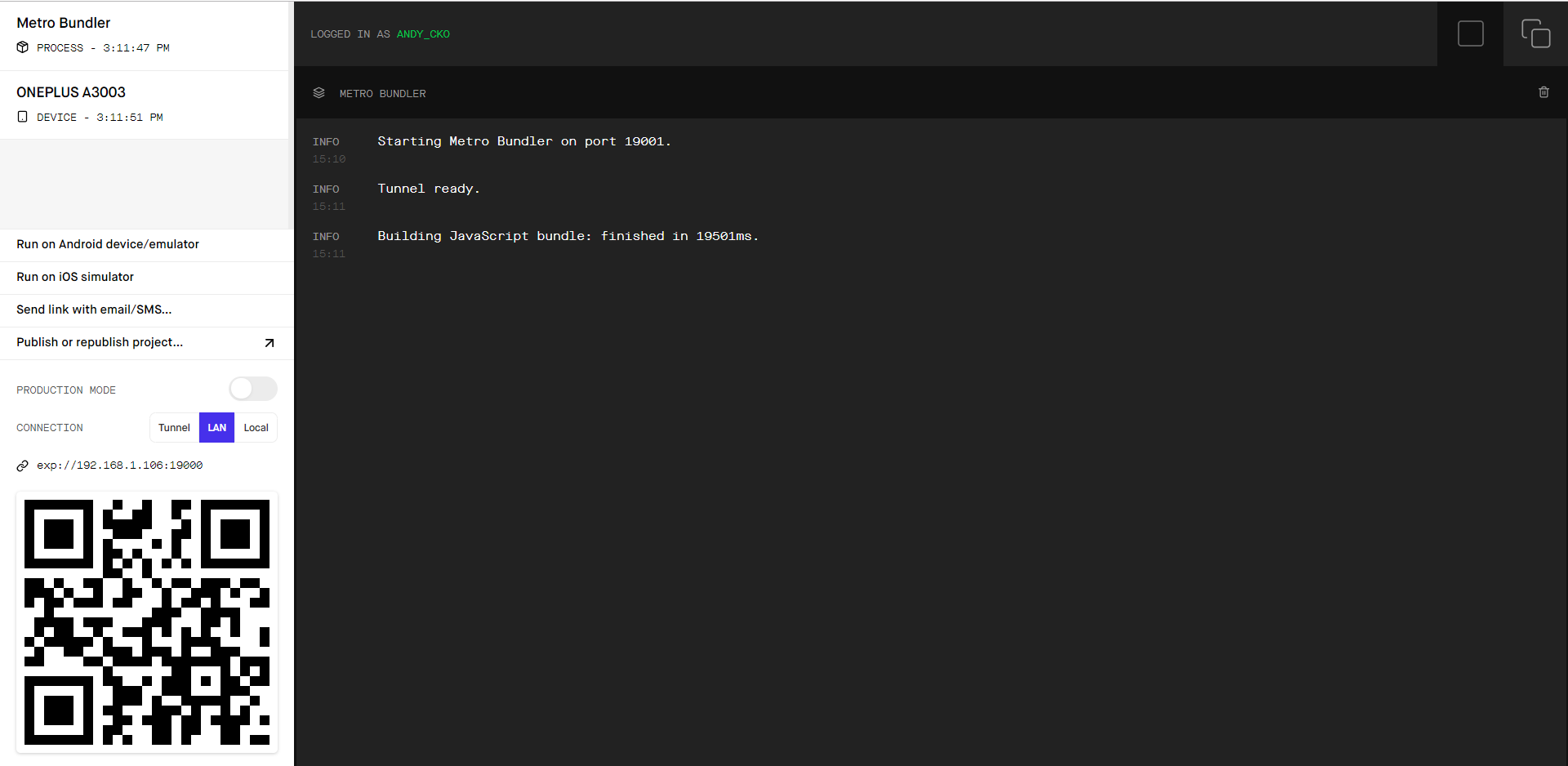
expo init MojProjekt *//Inicializácia projektu*

cd MojProjekt

expo start *//Spustenie Expo vývojárskej konzoly*

Po prebehnutí týchto príkazov sa vytvorí adresár s názvom MojProjekt s celou štruktúrou aplikácie v React-native a je pripravený na vývoj.

Vývojárska konzola, ktorá je spustená pomocou príkazu „expo start“ slúži na vyvíjaní aplikácie priamo na mobilných zariadeniach s funkciou „live reload“, ktorá u načítava najnovší kód do zariadenia hneď po jeho uložení. Po spustení vytvorí server na počítači vývojára a zároveň otvorí jeho webové grafické užívateľské prostredie, kde je možné sledovať stav aplikácie na pripojených zariadeniach a ich konzolové výstupy. Tak isto tam je možnosť nastavenia či chce vývojár pripojiť zariadenia nachádzajúce sa na rovnakej sieti alebo mimo nej. Po vybratí typu siete je vygenerovaný QR kód, ktorý následne naskenuje vývojár pomocou klientskej aplikácie Expo na mobilných zariadeniach a ihneď po naskenovaní sa načíta najnovšia verzia aplikácie na dané zariadenia (Bez hocijakej inštalácie, nakoľko to beží v aplikácii Expo).



* + 1. Štruktúra aplikácie Felix

Aplikácie vyvíjané v React-native sa skladajú z komponentov, ktoré sú organizované určitým smerovačom (Router, Navigator), v ktorom sú vytvorené spojenia medzi rôznymi komponentami a sú zorganizované do jedného alebo viacerých StackNavigator, TabNavigator alebo SwitchNavigator objektov. Záleží na tom, čo vývojár potrebuje vo svojej aplikácii.

Felix je rozdelený na dva StackNavigator objekty, medzi ktorými je následne možné prepínať pomocou SwitchNavigator objektu. Prvý Stack je objekt obsahujúci moje prvé dva Komponenty alebo inak povedané obrazovky, ktoré uvidí užívateľ pred zadaním mena do aplikácie a prihlásenia sa. Po prihlásení vidí užívateľ už len obrazovky druhého Stacku a nemá prístup k tomu prvému. Toto je možné vďaka Switch-u ktorý medzi nimi prepímana.