Jihočeská univerzita v Českých Budějovicích

Ekonomická fakulta

Katedra aplikované matematiky a informatiky

Vývoj mobilnej aplikácie multimediálny poznámkový blok

Vypracoval: Pavol Helmeci

Vedúci práce: Mgr. Radim Remeš, Ph.D.

České Budějovice 2022

Prehlásenie

Vyhlasujem, že som vypracoval svoju bakalársku prácu nezávisle s použitím iba zdrojov a literatúry uvedených v zozname citovanej literatúry. Vyhlasujem, že v súlade s § 47b zákona č. 111/1998 Zb. v znení neskorších predpisov súhlasím so zverejnením mojej bakalárskej práce, a to - v neskrátenej forme / v úprave vyplývajúcej z vypustenia označených častí archivovaných Ekonomickou fakultou - elektronicky vo verejne prístupnej časti databázy STAG prevádzkovanej Juhočeskou univerzitou v Českých Budějoviciach na svojej webovej stránke, pričom si zachovávam autorské právo na predložené znenie tejto kvalifikačnej práce. Súhlasím tiež s tým, že stanoviská supervízora a odporcov bakalárskej práce, ako aj záznam o priebehu a výsledku obhajoby kvalifikačnej práce by mali byť zverejnené rovnakými elektronickými prostriedkami v súlade s vyššie uvedeným ustanovením zákona č. 111/1998 Zb. Súhlasím aj s porovnaním textu mojej kvalifikačnej práce s databázou kvalifikačných prác, Theses.cz, ktorú prevádzkuje Národný register vysokoškolských kvalifikačných prác a systém odhaľovania plagiátorstva.

_____ _____

Datum Podpis studenta

Poďakovanie

Ďakujem pánovi Mgr. Radimovi Remešovi, Ph.D., vedúcemu mojej bakalárskej práce za rady, pripomienky a návrhy pri jej vedení. Ďakujem Tomášovi Čeloudovi za pomoc pri tvorbe kódu a poskytnuté informácie o vývoji aplikácie.

Obsah práce

1. Úvod 8
2. Ciele a motivácia 10
3. Mobilná aplikácia 11
3.1.1. Aplikácie určené na vzdelávanie 11
3.1.2. Aplikácie pre životný štýl 11
3.1.3. Aplikácie sociálnych médií 12
3.1.4. Aplikácie na zvýšenie produktivity 12
3.1.5. Cloudové služby 12
3.1.6. Aplikácie určené na zábavu 13
3.1.7. Herné aplikácie 13
3.2. História vývoja mobilných aplikácií 14
4. Vývoj mobilných aplikácií 17
4.1. Vytvorenie hlavnej myšlienky a nápadu 17
4.1.1. Prieskum trhu 17
4.2. Vytvorenie návrhu 18
4.3. Výber platformy 18
5. OS Android 20
5.1.1. Výhody vývoja aplikácií v OS Android 20
5.1.2. Nevýhody vývoja aplikácií v OS Android 20
5.2. Architektúra OS Android 21
5.2.1. Linuxové jadro, Knižnice, Android 21
5.2.2. Android Runtime 21
5.2.3. Aplikačný rámec 21
6. iOS 23
6.1. Výhody a nevýhody iOS 23
6.2. Výhody vývoja aplikácií v iOS 23
6.3. Nevýhody vývoja aplikácií v iOS 24
6.4. Architektúra iOS 24

7. Psychológia návrhu aplikácií 27
7.1. Sociálny vplyv a princíp „lajkov“ 27
7.2. Reakcia na zmyslové obrazy a heuristika nedostatku 27
7.3. Podnet, rutina a odmena 28
7.4. Gamifikácia v aplikáciách 28
8. Programovacie jazyky aplikácie NODs 30
8.1. Typescript 30
8.1.1. Výhody programovania v Typescripte 30
8.1.2. TypeScript komponenty 31
8.2. HTML a CSS 31
9. Integrované vývojové prostredie 32
9.1.1. Výhody IDE 33
9.2. Visual Studio Code 34
10. Softvérový rámec 35
10.1. Angular framework 35
11. Vývoj mobilnej aplikácie NODs 37
11.1. Postup programovania mobilnej aplikácie NODs 39
11.1.1. Rozhrania mobilnej aplikácie NODs 40
11.1.2. Služby a komponenty 41
11.1.3. Header a notes component 43
11.2. Firebase note 52
11.3. Testovacie scenáre aplikácie NODs 53
12. Spätná väzba 54
13. Záver 55
I.Summary 56
II. Zoznam použitých zdrojov 57
III. Zoznam obrázkov, tabuliek 61
IV. Zoznam zdrojového kódu 62
V. Zoznam príloh 63
VI. Prílohy 64
VII. Kód 67

1. Úvod

Bakalárska práca pojednáva o problematike vývoja mobilných aplikácií. Mobilnú aplikáciu je možné označiť za softvér vytvorený k spusteniu na smartfónoch, tabletoch a emulátoroch. Miesto pre tvorbu aplikácií poskytujú integrované vývojové prostredia Integrated Development Environment IDE. Umožňujú používateľom efektívne vytváranie, testovanie, ladenie a údržbu aplikácie v priebehu jej životného cyklu. Najfrekventovanejšie používané platformy na distribúciu mobilných aplikácií sú Google Play a Apple App Store.

Teoretická časť popisuje spektrum kategórií mobilných aplikácií a rozoberá spôsoby ich využitia. Zaoberá sa procesmi jednotlivých časti vývoja. Jeho históriou počínajúcou od spustenia prvého smartfónu spoločnosťou IBM v roku 1993. Venuje sa výberom z najpoužívanejších frameworkov a IDE, pojednáva o benefitoch z ich využitia. Pri vývoji mobilnej aplikácie multimediálny poznámkový blok NODs, sa bude používať framework Angular určený na tvorbu škálovateľných aplikácií, ku ktorého hlavným stavebným blokom patria komponenty. Jedná sa o súbor integrovaných knižníc pokrývajúcich veľkú škálu funkcií. V teoretickej časti si priblížime architektúry nepoužívanejších operačných systémov Android a iOS. Práca pojednáva o používateľskej rutine a psychologických princípoch využitých pri návrhu aplikácie multimediálny poznámkový blok s názvom NODs. Zaoberá sa štúdiami používateľskej reakcie na zmyslové obrazy, heuristiku nedostatku, sociálny vplyv, rutinu, podnety a odmeny. Navrhuje funkcie aplikácie, ktoré vplývajú na neurologickú slučku pozitívnym spôsobom a vedú k zdravím prístupom k organizácii aktivít a plnenia používateľom plánovaných úloh.

Praktická časť bakalárskej práce pozostáva z návrhu a vývoju aplikácie NODs multimediálny poznámkový blok, aplikácia je prioritne určená na pridávanie a časovú organizáciu úloh používateľom. Pri tvorbe vo frameworku Angular, budeme používať programovací jazyk Typescript. Jedná sa o nadstavbu jazyka JavaScript rozšírenú o ďalšie atribúty. Využitie u je najmä vo vývoji jednostránkových webových aplikácií, ktoré patria modernému prístupu k vývoju softvéru. Jeho výhodu sú užívateľsky pohodlnejšia manipulácia s aplikáciou a prehľadnejšia orientácia. Pri ukladaní dát v aplikácii NODs, bude používaná back-end služba Firebase, ktorá poskytuje funkcie databázy. Praktická časť sa ďalej venuje popisu funkcií jednotlivých komponent, ktoré sú základnými dielmi frameworku Angular. Úlohe praktickej časti náleží vyhodnotenie a popis užívateľského pohľadu na využitie funkcií aplikácie. Práca sa zaoberá skúmaním prístupu používateľov k plánovaniu a organizácii. Záver práce obsahuje posúdenie hlavných faktorov ovplyvňujúcich prístup užívateľa a návrhoch na budúce

vylepšenia verzie aplikácie, ktoré by mali viesť k lepšiemu používateľskému prístupu k organizácii činností.

2. Ciele a motivácia

Cieľom teoretickej časti bakalárskej práce je priblížiť rozdelenie aplikácií podľa jednotlivých kategórií a histórie ich vývoja. Opis dôležitých krokov pri tvorbe aplikácií a porovnávanie najpoužívanejších operačných systémov, ich architektúr. Popis frameworku a integrovaného vývojového prostredia. Na záver teoretickej časti práca opisuje psychologické faktory implementované pri návrhu aplikácii a približuje fungovanie neurologických slučiek.

V praktickej časti je cieľom popísanie procesu programovania aplikácie multimediálny poznámkový blok s názvom NODs . Priblíženie rozdelenia funkcií jednotlivých komponent, ich spolupráca v chode aplikácie. Záver praktickej časti sa venuje návrhu na budúce vylepšenia mobilnej aplikácie, so zameraním sa na implementovanie funkcií podporujúcich budovanie zdravých používateľských návykov. Zhodnotením použiteľnosti aplikácie na nasadenie v reálnom prostredí.

Motiváciou k vypracovaniu tejto bakalárskej práce bola snaha zlepšiť si znalosti v oblasti programovania vo frameworku Angular. Zameranie sa na vývoj aplikácie, využitím informácií získaných pri štúdiu. Záver práce zahŕňa navrhnutie funkcií pre budúci vývoj aplikácie, implementovaním poznatkov, zameraných na zdravý prístup používateľov a ich motiváciu k plneniu zadaných úloh.

3. Mobilná aplikácia

Mobilnú aplikáciu je možné definovať ako typ aplikačného softvéru, ktorý je navrhnutý na spustenie prostredníctvom mobilných zariadení ako je napríklad smartfón, tablet, poprípade digitálny asistenti. Aplikácie určené pre tieto zariadenia slúžia na poskytovanie rôznorodých služieb používateľom. Poznáme širokú škálu druhov aplikácií, mnohé z nich sa prekrývajú v jednotlivých kategóriách. Napríklad aplikácie, ktoré slúžia predovšetkým na zábavu, alebo ako hry, majú aj vzdelávací účel.

Podľa primárnej funkcie, na ktorú je aplikácia zameraná, ich možno rozdeliť do kategórií:

- Aplikácie určené na vzdelávanie
- Aplikácie na zlepšenie životného štýlu
- Aplikácie sociálnych médií
- Aplikácie na zvýšenie produktivity
- Aplikácie určené na zábavu
- Herné aplikácie

3.1.1. Aplikácie určené na vzdelávanie

Primárnou úlohou vzdelávacej aplikácie je informovať. V tejto triede sú spravodajské aplikácie, aplikácie na učenie cudzieho jazyka, aplikácie určené na vzdelávacie účely.
K najznámejším v tejto skupine patrí Duolingo. Je to služba, ktorá je dostupná pre používateľov od roku 2012. Pri spustení aplikácie si používateľ zvolí cudzí jazyk, o ktorý má záujem, má na výber devätnásť jazykov. Naplánuje lekcie a stanoví ako dlho majú trvať, s akou úrovňou začína. Dôležitý je sociálny aspekt, kde sa užívatelia môžu navzájom podporovať a komunikovať v skupinách. Kľúčovým faktorom pre aplikáciu určenú na vzdelávanie je používateľom podávať správy, informácie, v originálnom zábavnom formáte a to pre veľkú škálu vekových skupín s odlišnými záujmami. (svetandroida, 2017)

3.1.2. Aplikácie pre životný štýl

Tento druh aplikácií pokrýva veľa oblastí , prednostne by tieto aplikácie mali slúžiť na zjednodušenie každodenného života. Patria tu služby ako fitness, zoznamky, donáškový servis, aplikácie zaoberajúce sa mobilitou. Do kategórie takých služieb zaraďujeme firmu Bolt, jedná sa o Estónsku spoločnosť, ktorá ponúka širokú škálu služieb. Firma Bolt bola založená v roku 2013 devätnásť ročným študentom Markusom Villingom, keď dostal nápad spojiť taxi služby v Talline a Rige. V aplikáciách Bolt má užívateľ ponúkané služby prenájmu vozidiel,

mikromobility, zdieľania áut a donášky jedla. Bolt svoje služby ponúka vo viac než 300 mestách v Európe, Amerike, Ázii a Afrike. K najpoužívanejším aplikáciám zameraných na zlepšenie životného štýlu patria Tripadvisor, Uber, Bumble, Wolt, Dámejídlo, IDOS. (Wikipedia, 2022)

3.1.3. Aplikácie sociálnych médií

Sociálne siete, ponúkajú svojim užívateľom predovšetkým príležitosť rýchlej a efektívnej komunikácie, predávania informácií, zdieľaní obsahu a reagovaní na užívateľský obsah. Väčšinou sa jedná o univerzálne služby, ktoré majú rozmanité používateľské základne. Pojem sociálna sieť bol po prvý krát zmienený profesorom J. A. Barnesom v roku 1954 na londýnskej ekonomickej univerzite. Záverom jeho výskumu sociálnych vzťahov medzi živočíchmi v nórsku bolo, že spoločnosť ako takú, je možné definovať ako množinu bodov prepojených linkami. Tieto linky následne vytvárajú kompletnú sieť vzťahov. (socialne-siete8, 2014). Prvá sociálna sieť Six Degrees, bola navrhnutá a vytvorená Andrew Weinreichom v roku 1997. Six Degrees získavala na svojej obľube až do roku 2003, keď bola nahradená konkurenčnou sieťou MySpace, ktorá umožňovala používateľom prehrávanie hudby a zmenu pozadia užívateľského prostredia. Koncom roku 2004 bola táto, populárna sociálna sieť nahradená Facebookom.(wonderopolis, 2022)

3.1.4. Aplikácie na zvýšenie produktivity

V tejto triede nájdeme business aplikácie a aplikácie na zvýšenie produktivity, ktoré nám pomáhajú organizovať jednotlivé úlohy. K týmto aplikáciám náležia rôzne organizéry, aplikácie určené na výpočty. Najznámejšie sú Wolfram, Photomath, Taskade, Miro, MSTeams. Zaraďujeme k nim aj cloudové služby, môžeme ich charakterizovať ako poskytovanie programov servermi dostupnými z internetu, ku ktorým môžu používatelia pristupovať prostredníctvom webového prehliadača. K benefitom cloudových aplikácií patrí jednoduchá operatívnosť z tretích strán, možnosť rýchlej aktualizácie, testovania a nasadenia. To umožňuje podnikom rýchle a efektívne predstavenie produktov na trhu. (petrsmejkal, 2021)

3.1.5. Cloudové služby

Do výhod cloudu zaraďujeme vylepšené možnosti zdieľania dát a ich zabezpečenia, údaje v cloudových službách sú uložené pre oprávnených používateľov a poskytovatelia cloudových služieb si vďaka obrovskému rozsahu môžu dovoliť väčšie investície do zabezpečenia. (techtarget, 2021)

3.1.6. Aplikácie určené na zábavu

Ich účel má prioritnú úlohu zamestnať pozornosť používateľov na čo najdlhší časový úsek, pri rôznych aktivitách kde hľadáme snahu vyplniť náš čas. Táto kategória služieb má najčastejšie obsah vo forme zvuku, textu, videa. K najtrendovejším zábavným mobilným aplikáciám za rok 2020 a 2021 patrí Netflix, Spotify, HBO GO. Netflix je americká streamovacia službu a produkčná spoločnosť založenú v roku 1997 Reedom Hastingsom a Marcom Randolphom. V decembri roku 2021 ponúkal Netflix svoje služby vo forme filmov a televíznych seriálov pre viac ako 221 miliónov používateľov. (Wikipedia, 2022)

Spotify je služba, ktorá poskytuje používateľom množstvo kategórií hudobných žánrov, podcastov, tematickej hudby a mnoho ďalšieho. Okrem základných funkcií streamovacej platformy ponúka funkcie ako zobrazovanie textu piesní, zdieľanie streamovania hudby s užívateľmi a funkciu s názvom *vylepšiť*, ktorá ponúka rozšírenie zoznamov skladieb o odporúčané skladby na základe obľúbených žánrov používateľa . Platforma bola založená v roku 2006 vo švédskom Štokholme. Za mesiac November 2021 mal Netflix 406 miliónov poslucháčov, z toho 180 miliónov platiacich odoberateľov. (Wikipedia, 2021)

3.1.7. Herné aplikácie

Kategória herných aplikácií sa zaraďuje k najpopulárnejším, pri tak obsiahlej kategórii existuje mnoho druhov herných aplikácií závislých od druhu hráča. Britský profesor Richard Allan Bartle, na základe svojho výskumu na univerzite v Essexe rozdeľuje hráčov do štyroch kategórií. Hráči zameraní na dosiahnutie úspechu, je ich približne desať percent z celkového počtu hráčov, majú obľubu v zbieraní odznakov a porovnávaní sa s ostatnými. Ide o typ používateľa, ktorý reaguje na motivačné programy. Hráči prieskumníci chcú poznať nové miesta a tajomstvá, nevadí im opakovanie sa určitej aktivity v hre ak za jej vykonávaním stojí „odomknutie“ neznámeho, do tejto kategórie patrí približne 10 percent hráčov. Hráči zameraní predovšetkým na socializáciu, ktorých je takmer 80 percent. Obľubujú prežívanie hry prostredníctvom interakcie s ostatnými hráčmi. Radi spolupracujú a obľubujú spojenie síl väčšieho počtu hráčov. Hráči v kategórii „killers“, sa vyznačujú podobnými črtami ako hráči zameraní na dosiahnutie úspechu. Odlišujú sa od nich dobrým pocitom zo skutočnosti, že ostatní hráči prehrávajú ich zavinením. Sú súťaživí a motivuje ich víťazstvo, porazenie zvyšných hráčov v hre. (interaction-design, 2020)

K najsťahovanejším kategóriám hier podľa štatistík vyhodnotených z dvadsiatich krajín s najväčším počtom užívateľov podľa analytickej spoločnosti App Annie patria: Puzzle (21%), Casino (19%), Srategické hry (17%), RPG hry (14%). (blog.udonis, 2022)

3.2. História vývoja mobilných aplikácií

Počiatok vývoja mobilných aplikácií je úzko spätý s prvými smartfónmi. Smartfón môžeme definovať ako telekomunikačné zariadenie obsahujúce pridané softvérové aplikácie. (merriam-webster, 2022)

Prvý smartfón bol navrhnutý a spustený spoločnosťou IBM v roku 1993, používateľom ponúkal aplikácie kniha kontaktov, kalendár, hodiny a kalkulačka, poznámkový blok , email a skicár. V roku 1983 mladý Steve Jobs predstavil svoju základnú víziu App Store, miesto kde by si mohli používatelia zabezpečiť potrebný softvér do svojho telefónu. (acodez, 2022)

Smartfónové aplikácie boli v minulosti označované ako funkcie, ktoré sa nachádzali v časti telefónu s názvom mobilná kancelária. V roku 1997 bola po prvý krát spustená hra Snake a to na mobilnom zariadení Nokia 6110. V tejto hre šlo o základnú myšlienku predlžujúceho sa plaziaceho tvora, ktorý sa pohyboval rýchlejšie keď ho používateľ pomocou šípok namieril na bod, ktorý predstavoval jedlo. Had sa postupne predlžoval, ak s ním hráč narazili sám do seba hra skončila. Snake2 bol predstavený v roku 2000 na mobilnom zariadení Nokia 3310, za krátky časový úsek získal veľké množstvo fanúšikov. (theprint, 2022)

V roku 2002 Kanadská spoločnosť BlackBerry Limited pôvodne známa ako Research In Motion (RIM), ktorá sa špecializuje na vývoj softvéru a kyberbezpečnosť pozdvihla hru smartfónov o niečo vyššie. Uvedením na trh telefónu BlackBerry 5810 s inovatívnym, bezkonkurenčným konceptom bezdrôtového e-mailu. Toto mobilné zariadenie disponovalo aplikáciami ako sú jednoduché arkádové hry, editory vyzváňacích tónov, kalendár, poznámkový blok, aplikácia na kreslenie a mnoho ďalších. Tieto aplikácie boli predchodcami tých dnešných. (arkenea, 2022)

V roku 2007 spoločnosť Apple avizovala príchod prvého IPhone, ktorý značne pozmenil celý mobilný priemysel. Prvý telefón od spoločnosti Apple ponúkal nové predinštalované aplikácie ako boli mapy, fotografie, počasie a množstvo iných. Krátko potom prišiel ďalší významný moment, ktorý spôsobil veľkú revolúciu v celom technologickom priemysle. Pred samotným uvedením IPhone 3G na trh Apple prezentoval svoje plány, ktoré zahŕňali predstavenie nástrojov na vývoj softvéru. Bol to významný bod, od ktorého začal IPhone

podporovať vývoj aplikácií od tretích strán použitím vyhľadávacieho nástroja Safari. (arkenea, 2022)

V júli roku 2008 platforma Apple App Store spustila služby obchodu s aplikáciami na smartfóny, kde si používatelia mohli vybrať spomedzi päťsto predstavených aplikácií. V uplynulých troch dňoch od oznámenia o spustení, dosiahla táto distribučná služba desať miliónov stiahnutí. Cena ponúkaných aplikácií sa pohybovala od bezplatného používania po desať dolárov. (arkenea, 2022)

V roku 2012, predstavil Google svoju vlastnú multifunkčnú, digitálnu distribučnú platformu spojením Android Market, Google Music a Google books. Google Play, ktorý slúži ako obchod s aplikáciami a digitálnymi médiami pre certifikované zariadenie, ktoré spúšťa operačný systém Android a jeho deriváty alebo OS Chrome. To umožňuje používateľom využívať aplikácie vyvinuté pomocou súpravy nástrojov na vývoj softvéru pre Android (SDK). (Wikipedia, 2022)

Súbor nástrojov na vývoj softvéru, Software Development Kit SDK obsahuje knižnice, debugger, emulátory , dokumentáciu pre aplikačné programové rozhrania operačného systému Android, zdrojový kód a návody. (technopedia, 2020)

Krátko po uvedení nového IPhone na trh sa firma Apple začal venovať výrobe nielen nových smartfónov, ale aj IPadov, ktorým začali prispôsobovať svoje aplikácie. Používateľom ponúkal nový zážitok cez väčší dotykový displej. Aplikácia na kreslenie DrawSomething si získala za deväť dní od uvedenia na trh viac ako milión používateľov, čo na porovnanie zabralo Facebooku deväť mesiacov. Záujem o mobilné aplikácie stúpal ako aj záujem o kúpu smartfónu. Na počiatku uvedenia distribučných služieb medzi používateľov sa stali najpopulárnejšie hry. Hra Candy Crush Saga a Temple Run 2, Angry Birds predstavovali pre operačný systém iOS šesťdesiat percent zisku. (arkenea, 2022)

Po herných aplikáciách prišli na trh sociálne média Snapchat, Vine a Instagram, ktoré v rokoch 2012 až 2013 získali veľkú popularitu. Koncom roku 2013 bola aplikácia Vine odkúpená Twitterom a získala za krátky čas značný úspech predovšetkým na zariadeniach Windows a Android. Vine sa stal používateľmi najviac sťahovanou aplikáciou na trhu. Fungoval na princípe zdieľania krátkych videí s dvesto miliónmi aktívnych používateľov. Vine bol predchodcom Musical.ly a dnešnej populárnej sociálnej siete TikTok. (businesschief, 2020)

Aplikácie sa stávali čoraz viac obľúbenejšími, ich majitelia začali bohatnúť a nakupovať menšie aplikácie so značným potenciálom. Facebook odkúpil Instagram za miliardu dolárov,

v roku 2022 sa hodnota sociálnej siete Instagram odhaduje na sto miliárd dolárov. Mobilné aplikácie sa začali rozmáhať do mnoho odvetví. Používali sa na širokú škálu činností od aplikácií určené pre inteligentné domácnosti, ktoré zahrňujú kontrolu teploty, svetla, spotrebičov, zabezpečenia domácnosti a mnoho ďalšieho. Po digitálnych asistentov, ktorí nám pomáhajú manipulovať s rôznymi aplikáciami v našich smartfónoch. (arkenea, 2022)

4. Vývoj mobilných aplikácií

Mobilnú aplikáciu môžeme definovať ako softvér, alebo program, ktorý je prioritne navrhnutý na spustenie na smartfónoch a tabletoch. V dnešnej dobe fungujú prevažne na operačných systémoch Android a iOS, ktoré ponúkajú svoje obchody s aplikáciami Google Play od Androidu a AppStore je od iOS. (mindbowser, 2022)

K základným fázam procesu vývoja mobilnej aplikácie patria:

- Vytvorenie hlavnej myšlienky a nápadu
- Prieskum trhu
- Vytvorenie návrhu aplikácie
- Výber platformy
- Fáza vývoja aplikácie
- Fáza testovania a úpravy chýb
- Spustenie Beta verzie aplikácie
- Fáza po spustení (mindbowser.com)

4.1. Vytvorenie hlavnej myšlienky a nápadu

Pred etapou plánovania, je dôležite odpovedať na základné otázky ohľadom konkrétneho nápadu. Čím sa zvyšuje možnosť v budúcnosti predísť chybám a neistotám vo funkčnostiach. Hlavné je vedieť, prečo chceme vyvíjať aplikáciu, aký je náš zámer, prečo je naša aplikácia potrebná. Podstatné je poznať dostupné zdroje, s ktorými chceme pracovať pri vývoji. Návrhári by mali poznať publikum a zákazníkov, pre ktorých vytvárajú aplikáciu. Významný faktor v tomto procese zastupuje analýza a prieskum trhu, znalosť konkurenčných firiem v odvetví. Rozhodujúce je určenie zdrojov z ktorých budeme čerpať či už ide o finančné náklady, alebo informačné zdroje. Dôležitým krokom je vytvorenie marketingového plánu a plánu propagácie. (mindbowser, 2022)

4.1.1. Prieskum trhu

Marketingový prieskum nám pomôže zistiť skutočné potreby zákazníkov a vyhodnotí zoznam ich požiadaviek. Dôležité je absolvovať kroky marketingového výskumného procesu.

1. Definícia požiadaviek, vytvorenie profilu zákazníkov, ktorých chceme osloviť
2. Pripravenie dotazníku, jeho vyhodnocovanie
3. Identifikovanie oblastí záujmu zákazníkov
4. Komunikácia so zákazníkmi o potencionálnom produkte, jeho vlastnostiach
5. Zahájenie zberu požiadaviek zákazníka a ich zakomponovanie do produktu (mindbowser, 2022)

4.2. Vytvorenie návrhu

Zoznam nápadov pre konkrétny softvér je pripravený, je načase mu dať podobu Wireframe. Wireframe sa môže v slovenskom jazyku preložiť ako skica webu, alebo projektová dokumentácia nového webu. Ide o návrh definujúci funkcie a obsah stránok webu alebo aplikácie. K jeho dôležitým funkciám patrí, rozmiestnenie prvkov na stránke, ktorej sa venuje informačný architekt. Jeho úlohou je preniesť do nákresov požiadavky zákazníka a zefektívniť týmto celý proces.(Wikipedia, 2021)

Na vytvorenie wireframe určeného softvéru používame niekoľko krokov. Ako dôležitú časť tohto procesu považujeme zostrojenie user flow diagramu. User flow možno považovať za prehľad, ktorý určuje kde sa zákazník môže v jednotlivom produkte pohybovať, informuje o postupnosti jeho krokov. Tieto kroky má používateľ vykonať na dokončenie danej úlohy. Vytvorenie user flow diagramu pomáha vypracovať logickú cestu , ktorou by sa mal užívateľ vydať pri interagovaní so systémom, diagram zobrazuje vzťahy medzi funkčnosťami systému, potencionálnymi akciami užívateľa a súvisiacimi dôsledkami. Pri vytvorení user flow diagramu je potrebné uskutočniť radu činností, ku ktorým patrí analýza požiadaviek, vytvorenie warframe a dizajnu. (mindbowser, 2022)

4.3. Výber platformy

Pri výbere platformy pre mobilnú aplikáciu máme na výber dve hlavné možnosti. Súčasne vládnu svetu smartfónov operačné systémy Android a iOS. Na obrázku č. 1 „Prehľad operačných systémov mobilných zariadení“ je možné si všimnúť ich výrazne väčší podiel na trhu. Tvorbu mobilných aplikácií zaraďujeme do exponenciálne rozvíjajúceho sa odvetvia, kde je značná konkurencia. Preplnenie trhu v kombinácií s časovou a finančnou náročnosťou značne komplikuje aplikáciám ich vstup na trh. Preto je dôležité hľadať spôsob ako znížiť prípadné straty a zároveň byť efektívnejší. K takým postupom patrí, zvolenie si cesty vývoja pre jednu platformu, nakoľko veľké percento používateľov mobilných telefónov užíva prevažne jeden operačný systém. To umožňuje firmám rýchlejší vstup na trh, overenie myšlienky a cennú spätnú väzbu v prípade, že sa bude aplikácia vyvíjať aj pre ďalšie platformy. (rascasone, 2021)

Obrázok 1 Prehľad operačných systémov mobilných zariadení

	Programovací jazyk	IDE	Podíl na trhu	Distribuce aplikací
Android	Java	Eclipse s Android SDK	81%	App Store
iOS	Objective-C	Xcode	13%	Google paly
Windows Phone	C, C#, C++	Visual Studio	4%	Windows Phone Store
BlackBerry	Java	Eclipse s BlackBerry SDK	1%	BlackBerry World

Zdroj: *Vývoj mobilních aplikací pro platformu Apple iOS* [online]. 2013 [cit. 2022-01-03]. Dostupné z:

https://is.muni.cz/th/n9cb8/Bakalarka.pdf

5. OS Android

Android je operačný systém, založený na jadre Linuxu. Najčastejšie jeho využitie nájdeme v smartfónoch, tabletoch, inteligentných hodinkách a mnohých ďalších zariadeniach. Vývoj operačného systému Android je vedený spoločnosťou Google. Android je voľne dostupný operačný systém, jeho názov pri jednotlivých výrobcov záleží na značke výrobku. U značky Xiaomi má názov MIUI, Samsung má One UI, v prípade že sú v operačnom systéme značné zmeny, ktoré ho odlišujú od pôvodnej verzie je často názov vynechaný. OS Android mal v roku 2016 viac ako osemdesiatpäť percentný podiel na trhu so smartfónmi. Po roku 2015 bol zaradený medzi najpoužívanejšie operačné systémy spolu s iOS od Apple a Windows Phone. Spoločnosť Android Inc. vznikla v roku 2003 v americkom štáte Kalifornia, bola založená Andym Rubinem, Richem Minerem, Nickem Searsem a Chrisem Whitem. V roku 2005 firmu odkúpil Google. V roku 2007 bolo vytvorené spoločnosťami Google, HTC, Intel, LG, Motorola, Nvidia, Qualcomm, Samsung, Texas Instruments. Zoskupenie pod názvom Open Handset Alliance. Výhodou boli nižšie celkové náklady na vývoj aplikácií a služieb, prívetivejšie vývojárske prostredie a open-surce komunita. Prvý komerčný mobilný telefón s operačným systémom Android bol uvedený na trh v Amerike pod názvom T-Mobile G1. (Wikipedia, 2022)

5.1.1. Výhody vývoja aplikácií v OS Android

K pozitívnym faktorom pri vývoji aplikácií pre Android zariadenia patrí vyššia miera flexibility. Android je open-source platforma. Jedná sa o druh platformy, kde sa softvér distribuuje s jeho kódom, čo umožňuje programátorom jeho použitie, modifikáciu a distribúciu s pôvodnými právami. Android patrí k najrozšírenejším operačným systémom s vyšším počtom používateľov ako jeho konkurenti. Jeho výhodou je to, že pri jeho spustení sa nemusí nutne jednať o smartfóny, tento operačný systém sa bežne používa na smart-tv, hodinkách smart-watch, pri chode inteligentných domácností a mnohých ďalších zariadení. (rascasone, 2021)

5.1.2. Nevýhody vývoja aplikácií v OS Android

K nevýhodám patrí zložitá optimalizácia a testovanie aplikácie. Medzi komplikácie vo fáze nasadenia zaraďujeme problémy s aktuálnosťou verzií operačného systému v jednotlivých zariadeniach. Podľa štatistík si používatelia s Androidom aktualizujú systém menej často, to má za dôsledok nutnosť vývoja aplikácií s podporou pre 5 až 6 rokov staré verzie OS. Nevýhoda operačného systému súvisí aj s povahovými črtami a chovaním používateľov, zatiaľ čo

užívatelia s iOS od Apple sú si ochotní za služby zaplatiť. Užívatelia Androidu požadujú lacné, alebo bezplatné služby a zároveň znesú výrazne väčšie množstvo reklamy. Do negatívnych vlastností open-source projektov ako je Android OS, patrí jeho náchylnosť na napadnutie. Zatiaľ čo Apple iOS je viac uzavretejší a bezpečnejší, nakoľko je Android prístupný vývojárom. Používateľom vie ponúknuť viac možností. (rascasone, 2021)

5.2. Architektúra OS Android

Architektúru operačného systému tvorí päť vrstiev. Aplikácie, aplikačný framework, knižnice, Android Runtime a Linux Kernel.

5.2.1. Linuxové jadro, Knižnice, Android

V spodnej časti vrstiev architektúry Androidu sa nachádza upravený Linux 3.6. Vrstva obsahuje všetky základné hardvérové ovládače, ako je kamera, klávesnica, displej, mikrofón a ďalšie. Jadro operačného systému vyniká v obrovskom množstve ovládačov od zariadení a sietí. Vo vrchnej časti linuxového jadra sa nachádza súbor knižníc spolu s open-source modelom webového prehliadača pod názvom WebKit. Android knižnice zaraďujeme do osobitnej kategórie, ide totiž o knižnice založené na jazyku Java, ktoré sú určené predovšetkým pre OS Android. Zahrňujú použitie aplikačného rámca, ku kľúčovým pre vývojárov systému Android patria *android.app*, *android.content*, *android.database*. (tutorialspoint, 2022)

5.2.2. Android Runtime

Táto časť obsahuje dôležitý komponent Dalvik Virtual Machine, ktorý funguje v jazyku Java, je špeciálne navrhnutý a optimalizovaný pre operačný systém Android. Dalvik VM využíva základné funkcie Linuxu, ku ktorým patrí správa pamäte a multi-threading, čo je schopnosť centrálnej procesnej jednotky alebo viac jadrových procesorov poskytovať súbežné spúšťanie vlákien podporované operačným systémom. Vlákno označuje v počítači spustenú inštanciu počítačového programu. (Wikipedia, 2022) Dalvik VM umožňuje aplikáciám Android, fungovať v ich vlastných procesoch s vlastnou inštaláciou virtuálneho stroja Dalvik. Android Runtime je virtuálny stroj od firmy Google, ktorý poskytuje vývojárom sadu základných knižníc, ktoré im umožňujú vyvíjať aplikácie pre Android pomocou štandardného programovacieho jazyka Java. (tutorialspoint, 2022)

5.2.3. Aplikačný rámec

Aplikačný rámec anglicky framework, poskytuje aplikáciám služby vyššej úrovne, ktoré sú zároveň súčasťou jazyka Java. Ponúka niekoľko kľúčových služieb, medzi ktoré patrí správa

aktivít, poskytovanie obsahu, správa zdrojov, správa upozornení, systém zobrazenia. (tutorialspoint, 2022)

6.iOS

IOS je operačný systém určený pre spustenie na smartfónoch od spoločnosti Apple. Prvý krát bol iOS zmienený ako iPhone OS, ktorý bol vyvinutý na základe macOS operačný systém pre počítače od firmy Apple. Telefón s využitím operačného systému iOS bol prvýkrát uvedený na trh v roku 2007, neskôr bol operačný systém prispôsobený ďalším zariadeniam ako multimediálny prehrávač iPod touch a iPad. Spoločnosť Apple je pôvodom z Ameriky so sídlom v Kaliforni, založili ju Ronaldo Wayne, Stephen Wozniak, Steven Paul Jobs. (Wikipedia, 2022).

V roku 1976 predstavila spoločnosť Apple stolný počítač Apple 1, ktorý obsahoval na rozdiel od počítačov tej doby zmontovanú základnú dosku. O rok neskôr bol predstavený Apple 2, modernizovaný s klávesnicou, puzdrom a rozširujúcimi zdrojmi. Po predstavení Apple 2 ,nasledovali domáce počítače s grafickým rozhraním GUI, k najúspešnejším projektom tej doby sa dá zaradiť Macintosh. Jeho predstavenie bolo naplánované v roku 1984. Išlo o doposiaľ najúspešnejší produkt. Osobný počítač obsahoval vstavanú obrazovku, myš, GUI a operačný systém známy ako System 1, čo bola najstaršia verzia dnešného Mac OS. Rokom 1997 Apple predstavil nový iMac a iPod a v roku 2007 prvý iPhone. Najstaršie modly prvého smartfónu obsahovali funkcie ako GPS navigácia, Touch ID, rozpoznanie tváre, možnosti snímania videí a fotografií. V roku 2017 predal Apple viac ako 223 miliónov smartfónov, čím sa jeho zariadenia stali najpredávanejšími produktami roku. Pod vedením Tima Cooka, ktorý prevzal vedenie spoločnosti po Jobsovej smrti spoločnosť expandovala. Vydala novú generáciu iPhonov, iPadov, iMacov a MacBookov spolu s novými produktmi, ako sú Apple Watch a HomePod. O rok neskôr sa tento technologický gigant stal prvou americkou spoločnosťou s hodnotou miliardu dolárov. Súčasne sa jedná o spoločnosť s najvyššou tržnou hodnotou na svete. (thoughtco, 2019)

6.1. Výhody a nevýhody iOS

V operačnom systéme iOS je jeho významná vlastnosť uzavretosť, od toho sa odvíjajú výhody a nevýhody jeho používania. (rascasone, 2021)

6.2. Výhody vývoja aplikácií v iOS

Vývojári nemusia rozmýšľať nad komplikovanou optimalizáciou ako pri Androide, iOS má optimalizáciu pre menej modelov, to robí mobilné aplikácie svižnejšími a rýchlejšími. Uzavretosť ekosystému ponúka ďalšie dôležité výhody, medzi ktoré patrí predovšetkým komunikácia s jednotlivými zariadeniami. Apple vymedzuje programátorom mantinely a udáva

štandardy, to pomáha ich motivácii a lepšej funkčnosti a prepracovanosti aplikácií. Tým umožňuje dosiahnuť lepší užívateľský pôžitok. (rascasone, 2021)

6.3. Nevýhody vývoja aplikácií v iOS

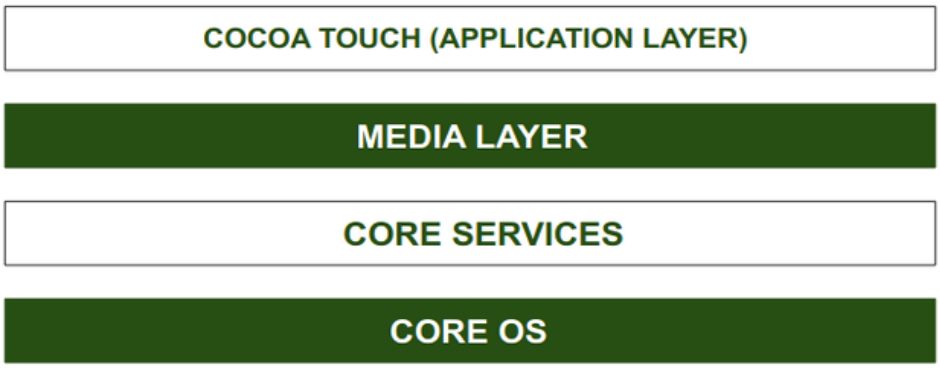
Aplikácia je pri nahrávaní na App Store podrobená komplikovanému schvaľovaciemu procesu, môže byť zamietnutá či už z dôvodu chýb, alebo z dôvodu nedostatočnej relevancie. (rascasone, 2021)

K nevýhodám patrí nedostatok bezplatných verzií aplikácií a používanie zväčša platených. Veľmi časté v aplikácií na App Store je používanie predplatného a demoverzie. Používatelia na vyskúšanie dostanú skúšobnú verziu, v ktorej musia vložiť spôsob platby, aby ste ju získali. Finančné ohodnotenie developera na platforme je komplikovanejšie, nakoľko splnenie kvót spoločnosti Apple na zverejnenie aplikácií je finančne náročné. Naopak v obchode Google Play má vývojár rôznorodejšie možnosti a väčšiu slobodu. (yourtechavatar, 2021)

6.4. Architektúra iOS

IOS je po Androide druhý najpoužívanejší operačný systém, jeho štruktúra je založená na vrstvení. Nižšie vrstvy poskytujú základné služby, na ktoré sa spoliehajú aplikácie a vyššia vrstva poskytuje grafiku a služby súvisiace z rozhraním. Framework je adresár, ktorý obsahuje knižnice, ako sú napríklad hlavičkové súbory, obrázky a pomocné aplikácie na podporu knižnice a ďalšie. Každá vrstva obsahuje viac rámcov, ktoré sú užitočné pre vývojárov. (geeksforgeeks, 2021)

Obrázok 2 Architektúra iOS



Zdroj: *GeeksforGeeks* [online]. 2022 [cit. 2022-01-03]. Dostupné z:

https://www.geeksforgeeks.org/architecture-of-ios-operating-system/

Najnižšia vrstva je CORE OS, sú na nej postavené všetky technológie operačného systému, podporuje ho 64bit procesor čo dovoľuje aplikáciám fungovať rýchlejšie. (geeksforgeeks, 2021)

K technológiám tejto vrstvy patrí :

- Základný rámec Bluetooth
- Rámec externého príslušenstva
- Accelerate Framework
- Rámec bezpečnostných služieb
- Miestny autorizačný rámec (geeksforgeeks, 2021)

Vrstva Core services obsahuje dôležité frameworky, ktoré pomáhajú operačnému systému poskytovať lepšiu funkčnosť. Jedná sa o druhú najnižšiu vrstvu. Nižšie sú uvedené niektoré z nich. Address Book Framework poskytuje prístup k dátam o používateľovi. Cloud Kit Framework poskytuje médium na presun údajov medzi aplikáciou v mobilnom zariadení a službou iCloud. Core Data Framework je technológa, ktorá sa používa na správu dátového modelu. Core Foundation Framework poskytuje funkcie správy údajov a služieb pre aplikácie určené na spustenie v operačnom systéme iOS. Všetky dáta sú v zariadení prístupné pomocou Core Motion Frameworku. Foundation Framework pokrýva funkcie, ktoré sa nachádzajú v rámci Core Foundation. HealthKit Framework spracováva informácie o zdraví používateľa na základe informácií zo získaných dát napríklad pri používaní hodiniek smart watch. HomeKit Framework sa používa na komunikáciu a ovládanie pripojených zariadení v domácnosti. Social

Framework pristupuje k účtom používateľov na sociálnych sieťach a StoreKit Framework podporuje nákup obsahu a služieb z aplikácií pre iOS. Vrstva médií anglicky media layer umožňuje systému grafickú, audio a video technológiu. Do tejto vrstvy patria rámce ako je ULKit graphics, ktorý poskytuje podporu pre navrhovanie grafického rozhrania a animácie obsahu v zariadení. Pri optimalizácii animácií v systéme pomáha rámec Core Animation. Podporu pre prehrávanie zoznamu skladieb a používanie knižnice iTunes umožňuje Media Player Framework. Rámec AV Kit poskytuje rôzne ľahko použiteľné rozhrania na prezentáciu videa, nahrávanie a prehrávanie zvuku a videa.Najvyššia aplikačná vrstva sa nazýva Cocoa Touch, slúži ako rozhranie pre používateľov na prácu s operačným systémom iOS. Podporuje dotykové a pohybové udalosti. Do tejto vrstvy zaraďujeme štyri základné rámce. EvenKit Framework, ktorý zobrazuje štandardné systémové rozhranie využívajúce ovládače zobrazenia na prezeranie a zmenu udalostí. GameKit Framework poskytuje užívateľskú podporu pre zdieľanie údajov súvisiacich s centrom pre hry. MapKit Framework poskytuje mapu rozhrania aplikácie a PushKit Framework je zameraný na podporu pri registrácii. (geeksforgeeks, 2021)

7. Psychológia návrhu aplikácií

Vývojári využívajú psychologické faktory pri tvorbe aplikácií s cieľom formovania správania používateľa a budovania jeho návykov. Aplikácie, ktorým sa darí prilákať a udržať si používateľov a pomáhajú používateľom dosiahnuť ich ciele, sú navrhnuté s ohľadom na princípy psychológie správania. Psychológia môže vplývať na zážitok z aplikácie, od grafického návrhu až po ovládacie prvky navigácie. Existuje nespočetné množstvo zásad, ktoré majú vplyv na počet stiahnutí a používanie aplikácie. (designli, 2018)

7.1. Sociálny vplyv a princíp „lajkov“

Princíp označenia „páči sa mi to“, patrí k najčastejším funkciám pri návrhu aplikácií. Výskum profesora Cialdiniho ukázal, že ľudia s väčšou pravdepodobnosťou vyhovejú požiadavkám tých, ktorých máme radi. Jedná sa o častý fenomén s veľkou účinnosťou, v praxi sa zameriava na konverzačný štýl písania a priateľsky vyzerajúce používateľské prostredie. Správanie nášho okolia motivuje ciele našich konaní. Tendencia konania iných ľudí riadiť naše vlastné konanie sa nazýva sociálny vplyv. Drvivá väčšina spoločnosti je nastavená tak, aby dodržiavala normatívne správanie rovesníkov. Štúdiu sociálneho vplyvu sa venoval psychológ Robert Cialdini, ktorý sa vo svojich prácach zameriaval na psychológiu presvedčovania.
V aplikáciách sa táto spoločenská vlastnosť využíva povzbudením užívateľov na zanechanie ohodnotenia, odporúčania. Umožnenie zdieľania s priateľmi a integrovanie komunity do funkcií aplikácie, za účelom spojenia sa s priateľmi. (designli, 2018)

7.2. Reakcia na zmyslové obrazy a heuristika nedostatku

Národné centrum biotechnológie NeurolImage, predstavilo štúdiu vykonanú v roku 2006, kde bola po prvýkrát dokázaná veľká účinnosť vplyvu zmyslových obrazov na používateľské správanie. Mozog subjektov bol skúmaný pomocou skeneru, zatiaľ čo im boli čítané slová, ktoré boli úzko prepojené s výraznými vôňami. Magnetická rezonancia, functional magnetic resonance imaging fMRI, následne dokázala vplyv vnímania slov na primárnu čuchovú kôru. V aplikáciách sa táto schopnosť ľudského mozgu využíva zapojením vizuálneho zmyslu svojich používateľov prostredníctvom pútavej grafiky, zapojenie zvuku označujúci úlohy a míľniky. Zapojením obrázkov, ktoré používateľom pomôžu vidieť, cítiť, počuť. (designli, 2018)

Jedna z najznámejších štúdií o nedostatku bola publikovaná vo vedeckom časopise Basic and Applied Social Psychlogy Journal. Výskumníci prezentovali subjektom dva druhy reklám, prvá svojím obsahom naznačovala množstvo voľných pracovných miest. Druhá ukazovala, že ich je nedostatok. Subjekty, ktoré videli reklamu s obmedzeným množstvom pozícií považovali danú náborovú spoločnosť za lepšiu, ako tú čo ponúkala množstvo pozícií. Účastníci výskumu verili, že spoločnosť s obmedzeným množstvom pozícií zamestnancom ponúka vyššie mzdy. Ľudí prirodzene priťahujú veci, ktoré sú v obmedzenom množstve. V aplikáciách sa táto črta správania, využíva ponukou časovo obmedzenej ceny. V prípade členstiev v aplikácii sa najčastejšie používajú časové obmedzenia na vyššie úrovne členstiev. (designli, 2018)

7.3. Podnet, rutina a odmena

Návyk človeka sa skladá z neurologickej slučky, podnet spúšťa rutinu prinášajúcu odmenu. Návyky sú ťažko prelomiteľné, pretože opakovaná odmena posilňuje rutinu. Návrhári aplikácií túto slučku využiť na analýzu za účelom vybudovať používateľom zdravé návyky. (designli, 2018)

V procese zmeny návyku, je potrebné identifikovať jednotlivé zložky slučky rutiny. Prísť na jednotlivé činnosti, ktoré u subjektu pomáhajú ako motivácia vo forme odmeny zdravšou alternatívou. V procese zmeny návyku je potreba zahrnúť jednotlivé činnosti. Prvým krokom v tomto procese by malo byť identifikovanie zoznamu odmien, ktoré sú štatisticky považované za motivujúce. K najobohacujúcejším odmenám patrí sociálna afirmácia, je to jedným z dôvodov, prečo majú používatelia tendenciu často kontrolovať upozornenia na sociálnych sieťach. Majú potrebu vedieť čo si o nich ostatní myslia. K často využívaným odmenám v aplikáciách patria začiarknutia v zozname, ocenenia, odznaky, hodnotenia. Pri návrhu aplikácie je potrebné vytvoriť rutinu, do ktorej sa používatelia zapoja, počiatočná rutina však nesmie byť náročná, aby ich neodradzovala. Pri budovaní neurologickej slučky je potreba zapojiť kreativitu, vymyslieť nové spôsoby efektívnej odmeny a bezbolestnej zmeny rutiny. (designli, 2018)

7.4. Gamifikácia v aplikáciách

Princípy gamifikácie sa uplatňujú v aplikáciách za účelom vzbudiť motiváciu v používateľoch. Základným princípom je vizualizácia „cesty“, to ako môže používateľ postupovať za účelom dosiahnutia míľniku a obdržania odmeny. Jednotlivé časti môžu byť zamerané na dĺžku časového úseku stráveným v aplikácii, počtu kliknutí, či splnených

jednotlivých úloh. Dôležitým faktorom je stanovenie cieľa a predstavenie benefitov z jeho dosiahnutia. Je potrebné naznačiť, ako má používateľ postupovať smerom k cieľu. Veda o motivácii dáva najavo, že je omnoho náročnejšie vzdať sa cieľa ak subjekt v jeho plnení pokročil. Pri implementovaní týchto princípov do funkcií aplikácie je podstatné, aby cesta za cieľom začala po jej prvom spustení. Pri snahe o udržaní motivácie sú rozhodujúce kontextové okná, upozornenia a maily sú v správnom použití účinné. Upozornenie používateľa prostredníctvom kontextového okna môže zvýšiť pravdepodobnosť jeho pokračovania v používaní aplikácie až o 71 percent. Odmeňovanie pomocou odznakov, je zamerané na úspechy používateľa dosiahnuté plnením jednotlivých úloh v aplikácii. Systémy prideľovania odznakov slúžia virtuálna reprezentácia stavu používateľa. (strivecloud, 2021)

8. Programovacie jazyky aplikácie NODs

8.1. Typescript

TypeScript je open-source objektovo orientovaný programovací jazyk firmy Microsoft. K najznámejším užívateľom patrí Office 365, Adobe, Bing Maps. Jedná sa o programovací jazyk s nadstavbu JavaScriptu. K jeho funkciám patrí kontrola typov, IntelliSense a refaktorovanie kódu, moduly rozhrania. Jeho syntax je veľmi podobná syntaxi skriptovaciemu jazyku ECMAScript 6. Po zostavení finálnej verzie projektu sa kompiluje do jazyku JavaScript. Na vytváranie šablón používateľského rozhrania sa používa textový značkovací jazyk HTML a kaskádové štýly CSS. Typescript je objektovo orientovaný kompilovaný jazyk a súbor nástrojov. Jedná sa o nad množinu JavaScrpitu, ktorá preberá jeho stavebné bloky. Akýkoľvek súbor JavaScriptu je možné premenovať na formát *.ts*, súbor vo formáte TypeScript. Následne skompilovať s inými súbormi TypeScriptu. (zdrojak, 2013)

Typescript tvoria:

- Statické dátové typy
- Kovariancia a kontravariancia
- Triedy a dedičnosť
- Generické datové typy
- Moduly
- Rozhranie (zdrojak, 2013)

Vďaka programovaniu v IDE môže ponúknuť:

- IntelliSence pre vlastný kód, knižnice JavaScriptu
- Zvýraznenie chyby podčiarknutím kódu
- Refactoring
- Príkazy *Go To Definition* a *Find All References* (zdrojak, 2013)

8.1.1. Výhody programovania v Typescripte

Jeho pozitívnou vlastnosťou je prenosnosť naprieč prehliadačmi, zariadeniami a operačnými systémami. Svoje základné jazykové vlastnosti preberá zo štandardizovanej špecifikácie ECMAScript , ktoré prepája spolu s funkciou generických a typových poznámok .TypeScript transpiler, sa stará o kontrolu chýb v kóde, ktoré generuje pri kompilácii. V prípade, že nájde syntaktickú chybu, zvýrazní ju pred spustením scriptu, aby ju bolo možné včas opraviť.

Jeho ďalšou prednosťou je voliteľný systém statického zadávania a odovzdávania typu prostredníctvom TLS. Ak je premenná deklarovaná bez typu, môže byť jej typ odvodený pomocou protokolu Transport Layer Security TLS na základe načítanej hodnoty. Typescript má podporu pre definície typov, súbor *definition* poskytuje definíciu typov pre externé knižnice JavaScript. To umožňuje ich použitie v programovaní. (tutorialspoint, 2022)

8.1.2. TypeScript komponenty

Jeho zloženie pozostáva z troch hlavných komponent jazyk, kompilátor TypeScript a jazyková služba Typescript. Komponenta jazyk obsahuje syntax, kľúčové slová a typové anotácie. Kompilátor TypeScript pomáha transformovať program do JavaScript kódu, pri kompilácii vykonáva analýzu a kontrolu kódu. Nakoľko webový prehliadač nepodporuje priame spustenie jazyku TypeScript, kód sa prevádza do jazyku JavaScript. Aktuálna verzia kompilátoru podporuje systémový modul *ES6*, nástroj na načítavanie modulov *SystemsJS*, špecifikáciu pre programovanie jazyka AMD. Pomocou balíka *npm* môžeme nainštalovať kompilátor lokálne a globálne, konfigurácia kompilátora je uvedená v súbore *tsconfig.json* v projekte. (javatpoint, 2019)

8.2. HTML a CSS

Textový značkovací jazyk, HyperText Markup Language HTML sa používa na vytváranie obsahovej kostry webových stránok. V minulosti slúžil na grafickú úpravu, dnes sa na ňu používajú kaskádové štýly, ktoré vytvárajú novú nezávislú vrstvu s ktorou je možné samostatne pracovať. *Hypertext* označuje funkciu jazyka prepájať texty pomocou odkazov. *Markup* označuje schopnosť predávania významu blokom textu použitím značiek tagov a vlastností atribútov. (strafelda, 2021)

Značky v HTML:

- Štrukturálne značky
- Sémantické značky
- Štylistické značky (strafelda, 2021)

CSS je skratka pre kaskádové štýly preto, že v nich funguje dedičnosť. Určuje vzhľad, farby, typ písma, veľkosti stránky. Interaguje s prvkami HTML, XHTML alebo XML.

9.Integrované vývojové prostredie

Je skratkou pre integrované vývojové prostredie, poskytuje vývojárom užívateľské rozhranie pre testovanie a písanie kódu. Jedná sa o softvérové platformy, poskytujúce programátorom sadu nástrojov určených na vývoj softvéru. Definuje umiestnenie súborov vizuálnou reprezentáciou, ktorá je pre užívateľov zrozumiteľnejšia, zahŕňa vývojové nástroje, ako sú knižnice kódov, testovacie platformy, kompilátory, editory, debugger a automatizačné nástroje. Majú schopnosti využívania viacerých procesov programovania, je možné v nich programovať využitím väčšieho množstva programovacích jazykov. Počet a typ programovacích jazykov v IDE závisí od jeho druhu.

Najpoužívanejšie IDE:

- Microsoft Visual Studio
- Net Beans
- IntelliJ IDEA Eclipse
- Komodo
- RubyMine
- Xcode
- Enide Studio 2014

Funkcie textového editoru, závisia od IDE prostredia, najčastejšie ponúkajú možnosti zvýraznenia chyby v syntaxe, navigačné nástroje, nástroje pre prispôsobenie rozhrania. Integrované vývojové prostredia sú migračnou formou textových editorov, využívajú technológie plnej funkčnosti, ktorá uľahčuje efektívnu editáciu kódu a následne jeho testovanie s vizuálnou reprezentáciou. Robia proces programovania zrozumiteľnejším a efektívnejším. (cs.education-wiki, 2022)

Používanie IDE je pre používateľov zrozumiteľnejšie a obľúbenejšie v kategóriách:

- Zvýšenej účinnosti
- Spolupráce programátorov
- Projektovom manažmente s využitím programových prostriedkov

IDE slúži predovšetkým na vývoj softvérových aplikácií k jeho hlavným funkciám patria:

- Textový editor
- Ladiaci program

- Kompilátor
- Dokončenie kódu
- Podpora programovacieho jazyka
- Integrácia a použitie zásuvných modulov

S IDE pracujeme pomocou operácií zápisového kódu, slúži na písanie a úpravu programovacieho kódu. Kompilačný kód je prevedený z programového kódu do strojovo spustiteľného kódu, ladiaci kód sa následne testuje. Integrované vývojové prostredie obsahuje sledovanie prostriedkov ako je využitie pamäte a kontrola miesta na pevnom disku. (cs.education-wiki, 2022)

9.1.1. Výhody IDE

- Použitie IDE na vytváranie softvérových aplikácií ovládačov a utilít
- Vývoj softvéru v ľubovoľnom programovacom jazyku, redukuje potrebu detailného pochopenia syntaxu
- Rýchlosť kódovania je dôležitým znakom IDE, je to umožnené vďaka jeho funkciám, ktoré pomáhajú pri organizácii zdrojov, monitorujú chyby a poskytujú skratky pri písaní kódu
- Pri vytváraní aplikácií v IDE sú spravované súbory na vopred určených miestach
- Obsahujú predinštalované knižnice pre konkrétny program a uľahčujú vytváranie databázových aplikácií
- Poskytujú funkcie triedenia, načítavania, spracovávania dát a vyhľadávaní v kóde
- Sú schopné konvertovať kód z viacerých programovacích jazykov na vysokej úrovni vo fáze zostavenia a kompilácie (cs.education-wiki, 2022)

Používanie IDE namiesto editorov je efektívnejšie pri procese :

- Ladenia
- Testovania jednotky
- Refaktoringu a profilovaní kódu
- Integrácii zdrojového kódu, vývojových nástrojov

9.2. Visual Studio Code

Visual Studio Code je integrované vývojové prostredie okrem bežných funkcií každého obsahuje niektoré jedinečné funkcie ku ktorým patrí:

- Podpora naprieč platformami: kód vo Visual Studio Code je multiplatformový funguje na operačnom systéme Windows, Linux, Mac
- Podpora terminálu: v minulosti pri programovaní v IDE, musel používateľ aby začal s konkrétnou akciou, začať od koreňového adresára. Vstavaný terminál poskytuje používateľskú podporu a výrazne uľahčuje prácu pri vývoji
- Multiprojekty
- Podpora Git: IDE umožňuje čerpanie zdrojov z Git Hub, jedná sa o platformu určenú pre softvérových vývojárov (cs.education-wiki, 2022)

Okrem možnosti používania funkcií na zefektívnenie práce k výhodám patria:

- Robustná architektúra
- Freeware
- Nástrojová podpora webových technológií HTML, CSS, JSON

10. Softvérový rámec

Je softvér, ktorý používajú vývojári pri tvorbe aplikácií. Používanie softvérového rámca, frameworku umožňuje používateľom, venovať sa funkčnosti aplikácie na vysokej úrovni. O akúkoľvek nízkoúrovňovú funkcionalitu sa totiž stará samotný framework.

Vývoj softvéru je komplexný proces, vyžaduje si rozdelenie množstva úloh vrátane plánovania, analýzy a návrhu, programovania, testovania, nasadenia a udržiavania. Systémové frameworky umožňujú prevziať kontrolu nad procesom vývoja. (hackr, 2022)

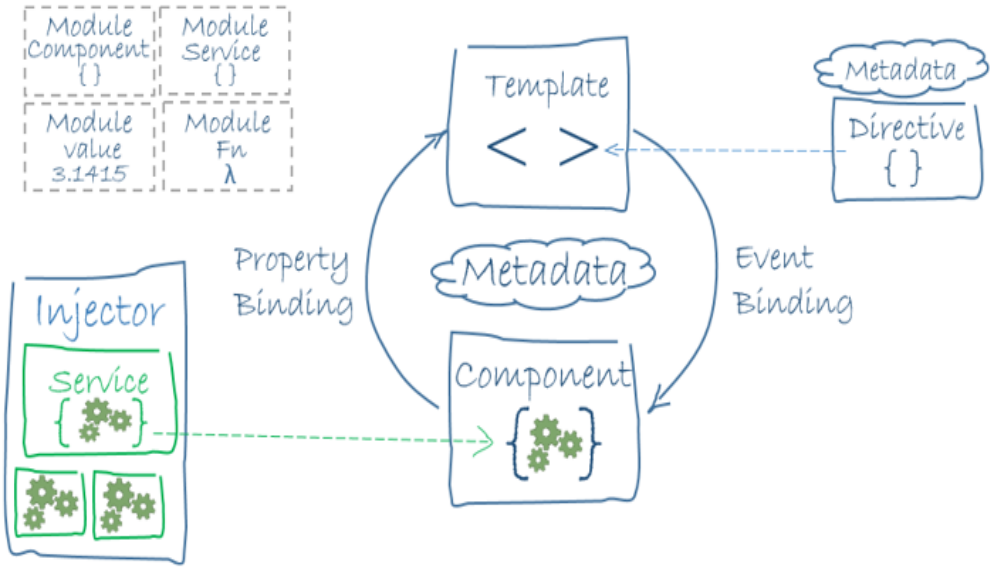
Výhody používania frameworku:

- Bezpečnejší kód
- Možnosť vyhnúť sa duplicitnému, alebo nadbytočnému kódu
- Testovanie a ladenie kódu je oveľa jednoduchšie
- Pri jednotlivých rámcoch dochádza k neustálemu vylepšovaniu činností, vďaka open-source frameworkom
- Pomáha konzistentnému vývoju kódu s menším počtom chýb
- Niekoľko segmentov kódu a funkcií je vopred vytvorených a vopred otestovaných. Vďaka tomu sú aplikácie spoľahlivejšie
- Uľahčuje prácu na sofistikovaných technológiách
- Pomáha pri zavádzaní lepších praktík programovania a vhodného používania návrhových, dizajnových vzorov (hackr, 2022)

10.1. Angular framework

Angular je postavený na základoch objektovo orientovaného programovania. Tvoria ho stavebné bloky komponenty. Komponenta je tvorená triedou, ku ktorej je om viazaná HTML a CSS šablóna pomocou direktív. Týmto spôsobom je front-end aplikácie rozdelený na samostatné celky, aplikácia je zložená podľa princípov OOP. Samotná logika aplikácie funguje používaním služieb, tieto služby sprostredkujú komunikáciu a distribúciu dát medzi aplikáciou vo frameworku a serverovým API. Jednotlivé komponenty získavajú tieto služby pomocou vkladania závislostí. Jedná sa o techniku, ktorá využíva princípy OOP, vkladanie závislostí medzi jednotlivými komponentami, slúži na ich vzájomné používanie a prístup k funkciám. Komponenty a služby sa rozdeľujú do modulov. (itnetwork,2022

Obrázok 3 Architektúra frameworku Angular



Zdroj: *Itnetwork* [online]. [cit. 2022-04-03]. Dostupné z:

https://www.itnetwork.cz/javascript//zaklady/uvod-do--frameworku

Node.js je prostredie umožňujúce spúšťanie programovacieho jazyku JavaScript. Funguje na Chrome V8 engine , čo znamená, že prostredie je rovnaké ako vo webovom prehliadači. Pre vývoj aplikácie v e je potrebná inštalácia Angualr CLI, rozhrania pre príkazový riadok, ktoré umožňuje vytvárať a spravovať softvérové aplikácie.

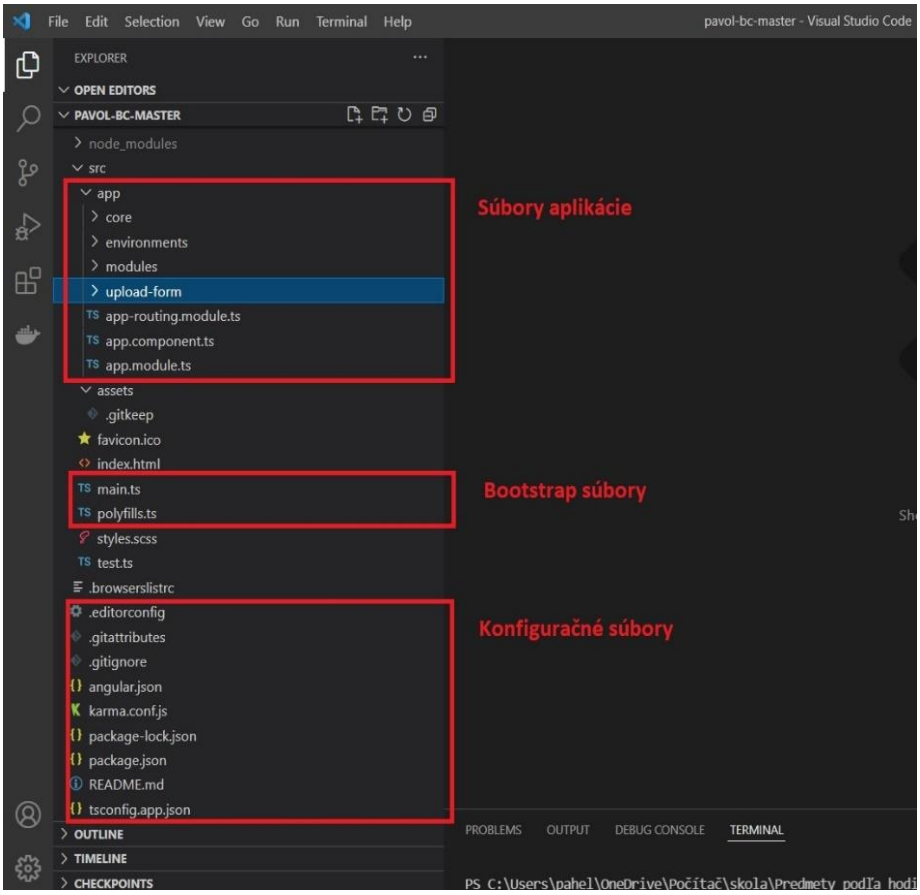
K hlavným stavebným blokom vo frameworku patria:

- Moduly
- Komponenty
- Služby
- Šablóny
- Metadáta
- Dátové väzby
- Smernice
- Vkladanie závislostí

Angular delíme do troch hlavných kategórií:

- Súbory aplikácie
- Boostrap súbory
- Konfiguračné súbory

Obrázok 4 Subory frameworku Angular



Zdroj: autor

Proces Bootstrap aplikácie sa začína v súbore main.ts, kde sa nachádza konfigurácia pre načítanie hlavného modulu. V app.module, v poli dekorátor @NgModule, sa nachádza pole boostrap spolu s pridanou komponentou. V tomto bode načíta app.component ako root komponentu v užívateľskom rozhraní.

11. Vývoj mobilnej aplikácie NODs

Na vývoj aplikácie vo frameworku sa pri tvorbe aplikácie multimediálny poznámkový blok NODs používa Visual Studio Code editor. Visual Studio Code je integrované vývojové

prostredie. Komponenty zohrávajú v Angulare dôležitú úlohu stavebných blokov. Rozdeľujú program do jednotlivých častí, ktoré navzájom spolupracujú. Vytvorenie komponenty môžeme realizovať pomocou terminálu vo vývojovom prostredí, alebo zadaním príkazu *ng generate component* s názvom komponenty.

Každá komponenta obsahuje 4 súbory:

- component.ts
- component.css
- component.spec.ts
- component.html

Component.ts je trieda komponenty, ktorá je písaná v programovacom jazyku TypeScript. Táto trieda implementuje funkciu *ngOnInit* v základnom nastavení. *NgOnInit* je, volaná frameworkom pri spustení aplikácie, ktorý má za úlohu indikovať, ukončenie vytvorenia danej komponenty. *Component.css* je súbor, do ktorého vkladáme kód CSS, ktorý chceme v danej komponente využiť. Na definovanie objektu používame označenie *class*. *Component.html* sa spúšťa v programovacom jazyku HTML. (educba, 2019)

Súbory a štruktúra koreňového adresára:

- app.component.css
- app.component.html
- app.component.spec.ts
- app.component.ts
- app.module.ts
- app-routing.module

Vyššie uvedený zoznam súborov vytvára framework štandardne, po založení projektu. *App.module.ts* súbor frameworku, ktorý obsahuje knižnicu, má v sebe dáta o vytvorených komponentách. Následne sú deklarované v *app.module.ts*. Komponenta aplikácie s predponou app je nadradená a všetky ostatné komponenty sú jej podriadené.Index.html je súbor v jazyku HTML, ktorý je generovaný om, tento súbor obsahuje označenie *app-root*, ktoré odkazuje do súboru *main.ts.* (educba, 2019)

V priečinku *main.ts* je importovaný *AppModule* a *bootstrapModule*, ktorý je v tomto súbore definovaný. Vďaka danému súboru sa načítajú vstupy z *AppModule*, pri prvom spustení aplikácie vo frameworku . *Main.ts* je vstupný bod pre aplikáciu, ktorá obsahuje nadradený

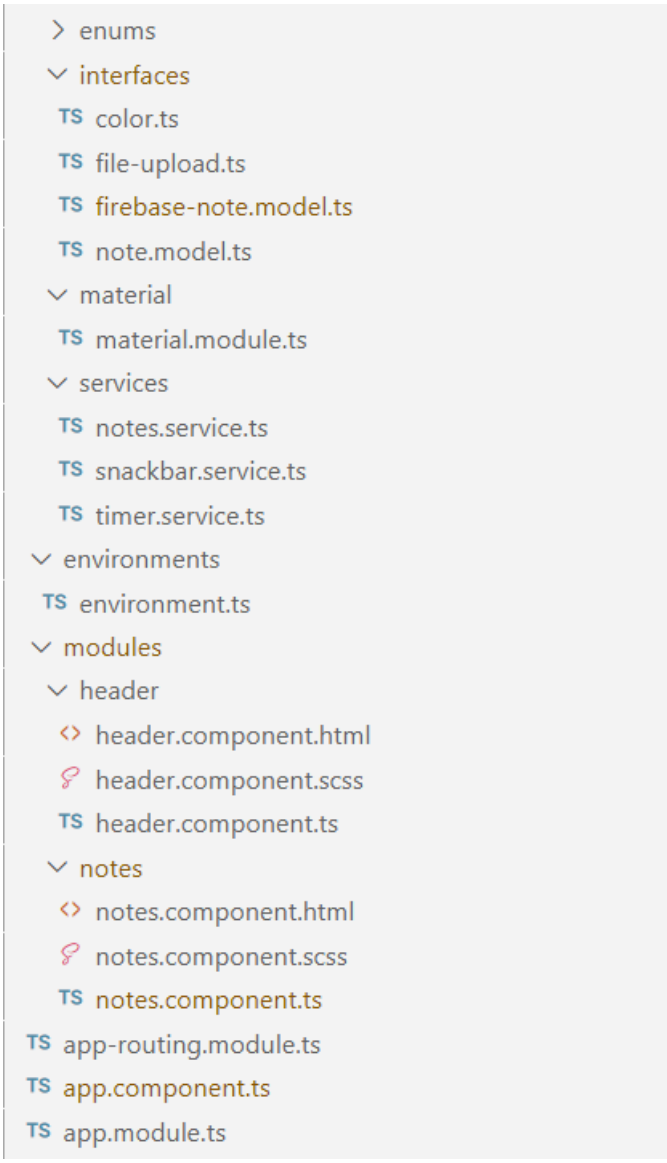
AppModule. Tento modul obsahuje všetky podriadené moduly, ktoré sa ním načítavajú. App.component.ts patrí do skupiny súborov štandardne vygenerovaných aplikáciou . Tento súbor obsahuje definovaný selektor v súbore *index.html*. Každá vytvorená trieda musí obsahovať anotáciu *@Component*. *App.componnet* obsahuje selektor, ktorý funguje ako jedinečný tag pre danú komponentu. *TemplateUrl* obsahuje cestu k súboru html v komponente, *styleUrls* obsahuje cestu k súboru CSS v komponente. *App.component.html* obsahuje predovšetkým kód v programovacom jazyku HTML, to ako bude stránka pôsobiť z užívateľského pohľadu. *App component.ts* má cestu pre svoj vstup do šablóny. Vždy, keď je zavolaný súbor *app.componnet.ts*, stránka je vykreslená dátami k zobrazeniu. (educba, 2019)

11.1. Postup programovania mobilnej aplikácie NODs

Pri založení projektu vo frameworku , používame IDE Visual Studio Code. Ako každý projekt v e, aplikácia NODs sa skladá z jednotlivých komponentov. Komponentu môžeme definovať ako stavebný kameň pre aplikácie . Každá komponenta pozostáva z HTML šablóny, ktorá deklaruje, čo sa vykresľuje na stránke. Selektor CSS definuje, ako sa komponent používa v šablóne. Pri vývoji sa štruktúra mení a funkcie sa presúvajú na základe lepšej funkčnosti a logickosti do iných komponent.

Proces programovania aplikácie NODs začíname vytvorením požadovaných komponent na základe zostrojeného návrhu aplikácie a rozmiestnenia jednotlivých funkcií. K hlavným funkciám mobilnej aplikácie NODs, patrí zadávanie úloh, ich editácia a časový odpočet. Zdané úlohy sa pre prehľadnosť rozdeľujú do jednotlivých kategórií. V prvej verzii tejto aplikácie sú kategórie pevne dané, avšak v nasledujúcej verzii bude možné predávať vlastné kategórie. Ktoré budú obsahovať databázu symbolov na definovanie kategórií. Používateľ ich bude môcť označiť symbolom alebo názvom. Proces pridávania úlohy (viď Príloha A: Diagram činností).

Obrázok 5 Štruktúra aplikácie NODs vo framewroku Angular



Zdroj: autor

11.1.1. Rozhrania mobilnej aplikácie NODs

Mobilná aplikácia obsahuje rozhrania *color.ts*, kde sú definované farby pre jednotlivé kategórie poznámok, *file-upload.ts* je rozhranie používané na nahrávanie používateľom vložených súborov do databáze Firebase Storage. Rozhranie *firebase-note.model.ts*, definuje typy atribútov triedy FirebaseNote, v atribúte *dateOfCreation* vytvárame nový objekt typu dátum. Používame funkciu *getTime()*, ktorá tento atribút konvertuje na milisekundy.
V *note.model.ts*, sa nachádzajú atributy, ktoré sú využívané funkciami v komponnete *notes.component.ts*, kde sú definované pri pridaní úlohy používateľom.

11.1.2. Služby a komponenty

V súbore *material.module.ts*, sa nachádzajú všetky potrebné importy, týkajúce sa navigácie, rozloženia, tlačidiel a mnoho ďalšieho. Služba *note.service.ts,* má za úlohu prácu s Firebase, jej funkcie budú bližšie popísané v nasledujúcich riadkoch. *Timer.service.ts* je služba, ktorá zodpovedá za kontrolu dátumu expirácie u jednotlivých úloh, využíva pri tom funkciu *getDates*, v konštruktore triedy *TimerService* sa nachádza *HttpCllient,* ktorý používa požiadavky *get, post, put* na úpravu a načítanie dát vo Firebase.

Priečinok projektu *environment.ts* obsahuje základný konfiguračný súbor, ktorý poskytuje predvolené prostredie na používanie funkcií noSQL databázy Firebase. Táto noSQL databáza pre cloudové funkcie ponúka konfiguráciu prostredia. Používateľ má na výber konfiguráciu premenných prostredia, použitím rozhrania Firebase CLI a použitie *functions.config.*

Ukážka kódu 1 environment.ts

```
export const environment = {
  firebase: {
    projectId: 'nods-fdb6b',
    appId: '1:108027487592:web:e64e5d20e2a3f8b8cba2dd',
    databaseURL: 'https://nods-fdb6b-default-rtdb.europe-west1.firebasedatabase.app',
    storageBucket: 'nods-fdb6b.appspot.com',
    locationId: 'europe-west',
    apiKey: 'AIzaSyAwY_wFWL5pX8Aa3Qf0qiV5I9KsPPQzEcI',
    authDomain: 'nods-fdb6b.firebaseapp.com',
    messagingSenderId: '108027487592',
    measurementId: 'G-XJQZVGQS46',
  },
    production: false,
    firebaseConfig : {
      apiKey: "AIzaSyDjOT6a4zi4N0pKYz8Wpkv4wy1Ol62FbdA",
      authDomain: "notesoriginal.firebaseapp.com",
      databaseURL: "https://notesoriginal-default-rtdb.europe-west1.firebasedatabase.app",
      projectId: "notesoriginal",
      storageBucket: "notesoriginal.appspot.com",
      messagingSenderId: "640937932701",
      appId: "1:640937932701:web:5481a41e42b7ad470a2174",
      measurementId: "G-S34G9QYSYS"
    }
  };
```

Zdroj: autor

Noteservice slúži ako služba na komunikáciu s databázou Firebase, využíva Firebase Realtime Database. Jedná sa o cloudovú databázu typu NoSQL, ktorá nám umožňuje ukladať a synchronizovať údaje v reálnom čase. Pri používaní Firebase je do projektu aplikácie potrebné nainštalovať závislosti príkazom *npm install firebase @/fire* v príkazovom riadku. Po ich

nainštalovaní je potrebné importovať *Firebase* moduly do projektu. Modulov je viac, podľa ich funkcionality, pre základné fungovanie databázy je potrebné importovať FireModule. Na záver je potrebné do projektu umiestniť vstavanú konfiguráciu prostredia, ktorá nám zaistí prístup k databáze. V službe *NoteService* využívame pri komunikácii s databázou funkcie HTTP, ktoré je možné spustiť prostredníctvom požiadavky. K *HTTP* funkciám Firebase patria funkcie (GET, POST, PUT, DELETE, OPTIONS).

Funkcie triedy NotesService:

- getNotes
- addNote
- deleteNote
- updateNote

Funkcia getNotes vracia objekty FirebaseNote z databázy Firebase. Vo funkcii GET je nastavený ako parameter odkaz sprostredkovaný databázou.

Ukážka kódu 2 Funkcia getNotes

```
getNotes(): Observable<FirebaseNote[]> {
  return this.httpClient.get<FirebaseNote[]>(this.url);
}
```

Zdroj: autor

Funkcia addNote, pridáva úlohy do databázy . Pri vrátení objektu požíva HTTP funkciu POST, v parametri je uvedený odkaz na databázu a objekt úloha. Na úpravu hodnoty zadanej úlohy v parametri používame funkciu JSON.stringify(), pomocou ktorej konvertujeme objekt Typescriptu na reťazec JSON. V tele funkcii addNote sa nachádza cyklus , ktorý kontroluje úlohy zadané používateľom. Ak má vložený obejkt prázdne hodnoty atribútov názvu a obsahu úlohy systém upozorní používateľa, vložením textu „Empty note“ do atribútu úlohy.

Ukážka kódu 3 Funkcia addNote

```
addNote(note: FirebaseNote): Observable<object> {
  if (note.title == '') {
    note.title = 'Empty Note';
  }
```

Zdroj: autor

DeleteNote je volaná v triede *NotesComponent*, jej úlohou je odstránenie vybranej úlohy používateľom. Ak je úloha odstránená pred uplynutím časového odpočtu aplikácia tento krok používateľa vníma ako omyl. Keď je úloha odstránená po uplynutí časového odpočtu, je automaticky zaznamenaná v počítadle nedokončených úloh, ktorého funkciou je vyhodnocovanie štatistík. Funkcia v parametre načítava atribút note.id.

Ukážka kódu 4 Funkcia deleteNote

```
  deleteNote(uid: string) {
    return this.httpClient.delete(this.urlUid + `${uid}.json`).subscribe();
  }
}
```

Zdroj: autor

UpdateNote je funkcia určená na zmenu hodnoty atribútov *notes* vo Firebase, jej využitie je v prípade zmeny stavu úlohy po jej exspirácii. Pri nastavení objektu na novú hodnotu požíva HTTP funkciu PUT, v parametri je uvedený odkaz na databázu a identifikačné číslo objektu pridaného do databázy. Funkcia *JSON.stringify()* konvertuje objekt programovacieho jazyka Typescript alebo jeho hodnotu na JSON reťazec, ktorý je potrebný na zápis do databáze.

Ukážka kódu 5 Funkcia updateNote

```
updateNote(uid: string, note: FirebaseNote): Observable<object> {

  return this.httpClient.put(this.urlUid + `${uid}.json`, JSON.stringify(note));
}
```

Zdroj: autor

11.1.3. Header a notes component

V súbore pod názvom *header* sa nachádza hlavička mobilnej aplikácie NODs. Ktorá obsahuje menu kategórii aplikácie, ktoré budú v budúcom vývoji aplikácie obsahovať funkcie triedenia úloh podľa vytvorených zvolených kategórii. Zahrňuje tlačidlo, ktoré používateľa presmeruje na sociálnu sieť Twitter, kde má možnosť zdieľať svoje skúsenosti s aplikáciou.

Notes component členenie

- notes.component.html
- notes.component.scss
- notes.component.ts

Obsahom komponenty *notes.component.ts*, je kód v programovacom jazyku TypeScript. TypeScript je open-source programovací jazyk vytvorený a spravovaný firmou Microsoft. Ide o nadstavbu nad programovacím jazykom JavaScript, ktorá ho rozširuje o statické typovanie a ďalšie atribúty.

Funkcie triedy NotesComponent

- ngOnInit
- getNotes
- getDate
- expiredNoteCounter
- assignValue
- formatNotes
- onAddNoteClick
- onEditClick
- onDeleteClick
- OnClearNoteClick
- onSelectFile
- startTimer
- onShareClick
- taskPercentageCalculator
- onCompleteClick
- convertBase64ToBlobData
- downloadFile
- onDownloadFileClick

Funkcia *NgOnInit* je volaná, pri procese vytvorenia inštancií smerníc. V kóde aplikácie NODs spustenie tejto funkcie odkazuje na spustenie *getNotes*, *getDate*.

Ukážka kódu 6 Funkcia nhONInit

```
ngOnInit(): void {
  this.getNotes();
  this.getDate();
}
```

Zdroj: autor

GetNotes je privátna funkcia, ktoré využíva službu *notesSerivice* na načítanie úloh z databázy Firebase po spustení aplikácie. Využíva pri tom metódu *next*, ktorá sa používa na odosielanie správ do pozorovateľného objektu, ktoré posiela všetkým komponentom.

Ukážka kódu 7 Funkcia getNotes

```
private getNotes(): void {
  this.notesService.getNotes().subscribe({
    next: (notes: FirebaseNote[]) => {
      this.formatNotes(notes);

      this.expiredNoteCounter();
    },
    error: (error) => {
      console.log(error);
    },
  });
}
```

Zdroj: autor

GetDate je privátna funkcia, ktorá pristupuje k funkcii *getDates* v službe *timerService*.

ExpiredNoteCounter vyhodnocuje používateľom splnená a nesplnené úlohy. Porovnáva čas ich exspirácie, po uplynutí časového úseku nastaveného používateľom je atribút stav úlohy typu boolean nastavený na hodnotu false. V tele funkcie sa používa cyklus *if*, jeho úlohou je identifikácia exspirovanej úlohy, porovnávaním dátumu zadaného používateľom a aktuálneho v milisekundách.

Ukážka kódu 8 Funkcia expiredNoteCounter

```
public expiredNoteCounter(): void {
  const myDate = new Date();

  this.notes.forEach((note) => {
    var date2: number = note.data.date;
    var date3 = Number(myDate.getTime());
    if (date3 > date2) {
      note.isExpired = true;
      note.data.state = false;

      this.notesService.updateNote(note.uid, note.data).subscribe();
    }
  });
}
```

Zdroj: autor

AssignValue v HTML využíva *ngModelChange*, ktorý môžeme definovať ako udalosť špecifickú pre Angular , ktorá sa používa na zachytávanie zmien pri vstupe používateľa. Funkcia využíva *event*, obsahuje informácie o udalosti zadania dátumu uplynutia úlohy používateľom v aplikácii. Následne je hodnota dátum expirácie úlohy priradená atribútu objektu.

Funkcia *formatNotes* je využívaná na zoradenie úloh podľa informácii o dátume jeho pridania. Metóda *Object.keys* vracia pole vlastností daného objektu, pole je iterované v rovnakom poradí ako by bolo vrátené slučkou.

Ukážka kódu 9 Funkcia formatNotes

```
private formatNotes(notes: any): void {
  this.notes = Object.keys(notes).map((key) => ({
    uid: key,
    isExpired: false,
    data: notes[key],
  }));

  this.notes.sort(
    (a, b) =>
      new Date(a.data.date).getDate() - new Date(b.data.date).getDate()
  );

  this.notes.reverse();
  console.log(this.notes);
}
```

Zdroj: autor

OnAddNoteClick využíva funkciu *addNote*, je volaná v službe *notesService* a nastavuje atribúty úlohy za účelom pridávania do databáze Firebase.

Ukážka kódu 10 Funkcia onAddNoteClick

```
onAddNoteClick() {
  this.notesService.addNote(this.note.data).subscribe({
    next: (response: object) => {
      console.log(response);

      this.onClearNoteClick();
      this.getNotes();

      console.log(this.note.data.color);
    },
    error: (error: string) => {
      console.log(error);
    },
  });
}
```

Zdroj: autor

Funkcia *OnEditClick*, používateľovi umožňuje úpravu obsahu zvolenej úlohy po zakliknutí tlačidla *edit* je možné zmeniť hodnoty atribútov objektu. Funkcia volá po spustení funkciu *updateNote*, ktorá mení hodnoty atribútov, následne je volaná funkcia *getNotes*, ktorá načíta obnovenie databázy *Firebase*.

Ukážka kódu 11 Funkcia onEditClick

```
onEditNoteClick() {
  this.notesService.updateNote(this.note.uid, this.note.data).subscribe({
    next: (response: object) => {
      console.log(response);

      this.onClearNoteClick();
      this.getNotes();

      console.log(this.note.data.color);
    },
    error: (error: string) => {
      console.log(error);
    },
  });
}
```

Zdroj: autor

OnDeleteClick umožňuje užívateľovi zvolenú úlohu po zakliknutí odstrániť. Vo funkcii je volaná funkcia *deleteNote* zo služby *notesService* , funkcia volá *taskPercentageCalculator*. Jedná sa o funkciu, ktorá po zavolaní v *onDeleteClick* zistí príčinu odstránenia úlohy. Ak je používateľom zvolená úloha nastavená na hodnotu *true* v atribúte *note.state*, znamená to, že čas na vykonanie úlohy neuplynul a používateľ veľmi pravdepodobne zadal úlohu nesprávne. Preto ju má možnosť odstrániť bez započítania do zoznamu používateľom neurobených úloh. Ak je hodnota atribútu *note.state false*, znamená to, že časový limit na vykonanie úlohy uplynul. Predpokladá sa že používateľ úlohu nestihol vykonať a preto sa táto úloha započíta do zoznamu nesplnených.

Ukážka kódu 12 Funkcia onDeleteClick

```
onDeleteClick(selectedNote: Note) {
  console.log(selectedNote.data);

  if (selectedNote.data.state == false) {
    this.UnCompletedTasks = this.UnCompletedTasks + 1;
  } else {
    console.log('úloha včas odstránená');
  }

  this.taskPercentageCalculator();
  this.notesService.deleteNote(selectedNote.uid);

  this.notes.forEach((note, index) => {
    if (note.uid == selectedNote.uid) {
      this.notes.splice(index, 1);
    }
  });
}
```

Zdroj: autor

OnClearNoteClick je funkcia, použitím ktorej môže používateľ vynulovať hodnoty atribútov pridávaného alebo pridaného objektu.

OnSelectFile v aplikácii využívame na pridávanie jedného alebo viac súborov k úlohe, táto funkcia mení stav atribútu isNotePosted, ktorý umožňuje zobraziť priebeh načítavanie súboru. Vytvára objekt FileReader, ktorý umožňuje aplikáciám asynchrónne čítať obsah súborov, (alebo vyrovnávacích pamätí nespracovaných údajov), uložených v zariadení používateľa. Využíva preddefinovanú funkciu *Push*(), ktorá generuje kľúč zakaždým, keď sa do špecifikovanej referencie Firebase pridá nový potomok . Použitím automaticky generovaných kľúčov pre každý nový prvok v zozname môže niekoľko klientov pridávať potomkov do rovnakého umiestnenia súčasne bez konfliktov pri zápise.

Ukážka kódu 13 Funkcia onSelectFile

```
onSelectFile(event: Event): void {
  this.isNotePosted = false;

  if (!this.note.data.files) {
    this.note.data.files = [];
  }

  const target = event.target as HTMLInputElement;
  const file = (target.files as FileList)[0];
  const reader = new FileReader();
  reader.readAsDataURL(file);
  console.log(file.name);
  reader.onload = () => {
    this.note.data.files.push(new FileUpload(reader.result));
  };
}
```

Zdroj: autor

Funkcia *StartTimer* zobrazuje priebeh načítavania pri odosielaní súboru do databázy využitím prednastaveného časového úseku.

OnShareClick dovoľuje používateľom zdieľanie vloženej úlohy, jej spustení je otvorené okno s názvom emailová schránka s prednastavenou hodnotou predmetu a obsahu emailu určeného na odoslanie.

Úlohou funkcie *taskPercentageCalculator* je výpočet štatistík používateľom splnených a nesplnených úloh, na základe logiky implementovanej vo funkciách, ktoré majú za úlohu identifikovať príčinu odstránenia úlohy používateľom. Pracuje s hodnotami atribútov *CompletedTasks,* jedná sa o počet úspešne dokončených úloh v časovom úseku vopred zadaným. Používateľ musí stlačiť tlačidlo so symbolom *complete* na zvolenej úlohe.

Ukážka kódu 14 Funkcia taskPercentageCalculator

```
taskPercentageCalculator() {
  this.TaskCount =
    this.CompletedTasks / (this.UnCompletedTasks + this.CompletedTasks);
  console.log(this.TaskCount);
  return this.TaskCount;
}
```

Zdroj: autor

Ak používateľ úlohu splní v nastavenom časovom úseku, po zakliknutí tlačidla „splnené“ , je úloha funkciou *onCompleteClick* odstránená a zaznamenaná počítadlom vyhodnocovania štatistík úloh.

Funkcia *convertBase64ToBlobData* zabezpečuje konvertovanie súborov uložených v databáze vo formáte base64 string, do formátu datového typu bližšie nešpecifikovaných binárnych dát v databáze - Binary Large Object BLOB. Nasledovne je vo funkcii *downloadFile* volaná funkcia *convertBase64ToBlobData* s paramaterom *fileBaseUrl*, *filename* a *contentType*.

Formát base64, zodpovedá textovému reťazcu, ktorý obsahuje informácie o obsahu súboru, jeho názov a typ, s ktorými tieto funkcie následne pracujú za účelom konvertovania súboru vhodného na stiahnutie do zariadenia používateľa.

Ukážka kódu 15 Funkcia convertBase64ToBlobData

```
convertBase64ToBlobData(fileBase64Url: string, contentType: string) {
  const sliceSize = 512;
  const byteCharacters = atob(fileBase64Url);
  const byteArrays = [];

  for (let offset = 0; offset < byteCharacters.length; offset += sliceSize) {
    const slice = byteCharacters.slice(offset, offset + sliceSize);

    const byteNumbers = new Array(slice.length);
    for (let i = 0; i < slice.length; i++) {
      byteNumbers[i] = slice.charCodeAt(i);
    }

    const byteArray = new Uint8Array(byteNumbers);
    byteArrays.push(byteArray);
  }

  const blob = new Blob(byteArrays, { type: contentType });
  return blob;
}
```

Zdroj: autor

Ukážka kódu 16 Funkcia downloadFile

```
downloadFile(fileName: string, fileBase64Url: string, contentType: string) {
  const blobData = this.convertBase64ToBlobData(fileBase64Url, contentType);
  const blob = new Blob([blobData], { type: contentType });
  const url = window.URL.createObjectURL(blob);
  const link = document.createElement('a');
  link.href = url;
  link.download = fileName;
  link.click();
}
```

Zdroj: autor

App-routing.module.ts je modul, ktorý ma za úlohu načítanie a nakonfigurovanie smerovača v samostatnom module, ktorý je určený na smerovania, importuje ho *root AppModule*.

Ukážka kódu 17 app.routing.module.ts

```
const routes: Routes = [
  { path: '', redirectTo: '/notes', pathMatch: 'full' },
  { path: 'notes', component: NotesComponent },
];
```

Zdroj: autor

Časť súboru s názvom *routes*, je miesto kde sú konfigurované jednotlivé trasy. Trasy podávajú smerovaču informácie, o tom ktoré ním majú byť vyobrazené. *Angular Route* má vlastnosti *path* a *component. Path* je reťazec zhodný s URL adresou. *Componnet* je komponenta vytvorená, pri navigácií na trasu URL adresy webového prehliadača.

Ukážka kódu 18 app.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: '<app-notes></app-notes>',
})
export class AppComponent {
  title = 'NODs';
}
```

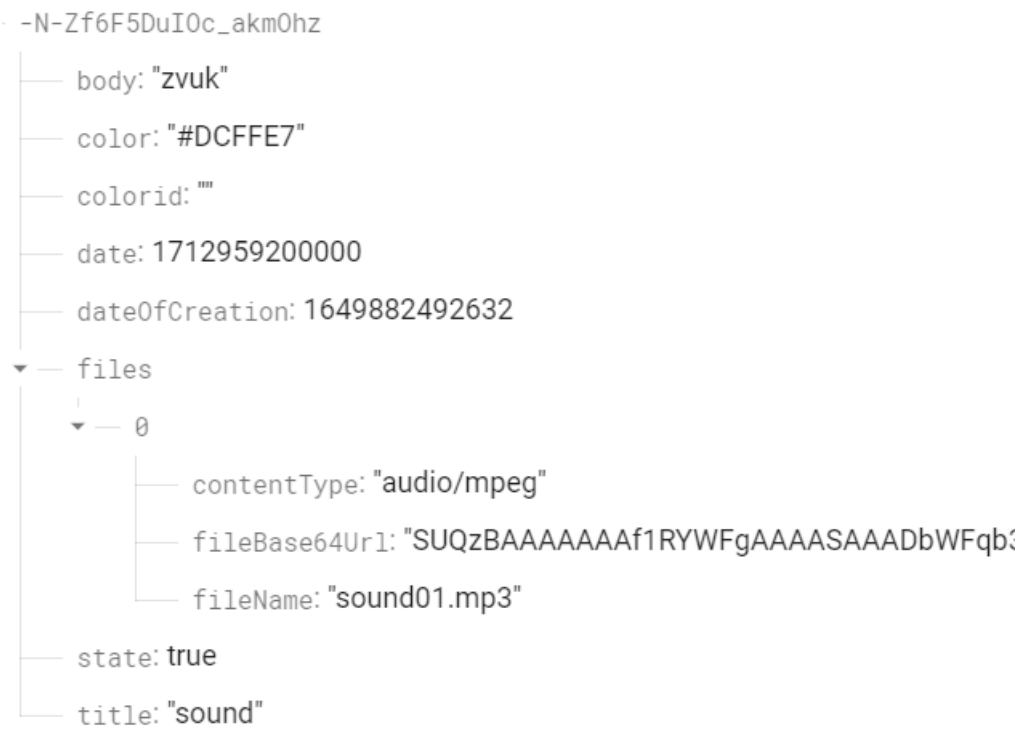
Zdroj: autor

V importe komponenty app-component.ts sa nachádza trieda Componnet, táto trieda je definovaná v *@angular/core*. *Selector* je značka, ktorá sa používa na umiestnenie do *index.html* súboru. *Template* je relatívna, alebo absolútna cesta k súboru šablóny pre Angular komponentu. App.module obsahuje všetky základné importy používané frameworkom.

11.2. Firebase note

Jednotlivé úlohy triedy Note, ktoré sú zadávané používateľom, obsahujú názov úlohy *note.title*, obsah úlohy *note.body*, kategória úlohy na základe farby zvolenej používateľom *note.color* a dátum uplynutia úlohy *note.date*. Atribút *note.state*, určuje stav exspirácie úlohy vloženej používateľom, ak platnosť úlohy vyprší jej stav sa zmení na *false*, ako môžeme vidieť na snímke obrazovky objektu note vytvorenej v databáze *Firebase*. *Note.dateOfCreation* obsahuje dátum, ktorý je zaznamenaný pri vložení jednotlivých úloh používateľom.

Obrázok 6 Úloha pridaná do databázy obsahujúca audio súbor



Zdroj: autor

Dôležité funkcie zahrňujú vloženie, odstránenie úlohy, časový odpočet zadanej úlohy, editáciu úlohy, vyhodnocovanie štatistík splnených a nesplnených úloh v časovom intervale, pridanie zvoleného súboru k úlohe. Pridané súbory môžu byť vo formáte (.jpeg, .png, .doc, .mp3), k jednej úlohe je možné pridať viac súborov. Na ukladanie súborov bude vo vývoji aplikácie používaná *Firebase*. *Firebase* zaraďujeme k službám *Backend-as-a-Service* (Baas).

Poskytuje vývojárom mnoho nástrojov a služieb. Jedná sa o databázový program NoSQL, ktorý ukladá dáta do dokumentov podobných JSON.

Služby *Firebase*:

- Autentifikácia
- Cloudové správy
- Databáza v reálnom čase
- Reportér zlyhaní v reálnom čase
- Monitorovanie výkonu
- Cloudová infraštruktúra na testovanie aplikácií

Firebase podporuje autentifikáciu pomocou hesla, telefónneho čísla, účtov Google, Facebook, Twitter a ďalších známych sociálnych sietí. *Firebase Authentication* (SDK) možno použiť na manuálnu integráciu jednej alebo viacerých metód prihlasovania do aplikácie. Dáta vo *Firebase* sa synchronizujú naprieč všetkými klientmi v reálnom čase a zostávajú dostupné, aj po uvedení aplikácie do stavu offline. Služba Firebase hosting poskytuje rýchly hosting pre aplikáciu, ukladá obsah do vyrovnávacej pamäte. Testovacie laboratórium obsahuje aplikáciu, ktorá je testovaná na virtuálnych a fyzických zariadeniach umiestnených v dátových centrách Google. Funkcia oznámenia poskytuje odosielanie oznámení pomocou platformy Firebase bez ďalšieho kódovania. (educative, 2022)

11.3. Testovacie scenáre aplikácie NODs

Testovanie prebiehalo v dvoch etapách, pri vývoji aplikácie a po dokončení hlavných funkcií. K overeniu jednotlivých funkcionalít mobilnej aplikácie bolo využité manuálne testovanie. Jednotlivé funkcie aplikácie boli odskúšané užívateľom, ktorý následne kontroloval ich výsledky. Testovanie prebiehalo po vývoji aplikácie v lokálnom prostredí, je rozdelené do hlavných činností vykonávaných používateľom podľa typu akcií (viď Príloha B: Testovacie scenáre).

12. Spätná väzba

Mobilnú aplikáciu s názvom NODs, si malo možnosť odskúšať v skúšobnej verzii celkovo šesť používateľov jednalo sa o osoby od 15 do 33 rokov. Prostredie zhodnotili ako prehľadné, v aplikácii sa podľa ohlasov dobre orientuje a používatelia nemali problém pochopiť jej fungovanie. Na základe jednotlivých kategórií je možné uložené poznámky farebne odlíšiť, čím užívateľ získa prehľad o jednotlivých úlohách, táto funkcia bola hodnotená pozitívne. Prácu so súbormi užívatelia zhodnotili ako jednoduchú, rýchlu a efektívnu. Informovanie o štatistikách splnených a nesplnených úloh bolo ohodnotené pozitívnymi ohlasmi. Čo sa týka vylepšení, užívatelia navrhujú filtrovanie poznámok na základe farieb a vytváranie vlastných kategórií. Tieto a ďalšie funkcie budú obsiahnuté v budúcej verzii mobilnej aplikácie.

13. Záver

Hlavným cieľom práce je návrh a vývoj mobilnej aplikácie multimediálny poznámkový blok NODs, ktorej hlavnou úlohou je zaznamenávanie poznámok s časovým odpočtom. Pri návrhu funkcií aplikácie bol použitý softvér na vizualizáciu procesov Microsoft Visio Drawing. Za primárnym účelom jej vývoja stojí záujem o zdokonalenie vedomostí, vylepšenie schopností programovania vo frameworku, štúdium odbornej literatúry a návrh budúcich funkcií využívajúcich psychologické princípy.

Aplikácia NODs obsahuje vyhodnocovanie štatistík na základe pomeru splnených a nesplnených úloh, ktoré sú pridané používateľom. Princípy gamifikácie aplikované v budúcom vývoji budú zahŕňať predstavenie cesty a cieľa, ktorou bude užívateľ postupovať pri včasnom plnení úloh. Aplikácia bude na základe počtu splnených úloh a štatistiky vyhodnotenej z ich pomeru, udeľovať používateľovi jednotlivé odznaky. Odznaky bronzovej striebornej a zlatej farby, na úvod bude užívateľovi predstavený priebeh udeľovania odznakov. Pri úvodnom spustení a splnení prvých úloh dostane používateľ prázdny odznak. Jeho vzhľad sa bude meniť v závislosti pomeru splnených a nesplnených úloh k celkovému počtu pridaných úloh. Aplikácia bude využívať upozornenia kontextového okna, nakoľko úroveň odznaku, počet hviezdičiek (jedna až päť) bude závisieť aj od pravidelnosti jej používania. Odznak bude mať podobu živej animovanej postavy v tvare odznaku, ktorej nálada bude závisieť na počte hviezdičiek získaných používateľom v danej kategórii odznaku. Čím bude aplikácia podporovať používateľskú rutinu. Budúce funkcie aplikácie budú obsahovať prepojenie na fórum, kde si jej používatelia budú môcť navzájom poskytovať rady odporúčania a diskutovať. Používateľ si bude môcť vkladať vlastné kategórie úloha a filtrovať ich.

Aplikácia využíva časový odpočet, ktorý kontroluje dátum zadania úlohy a nastaví stav úlohy na exspirovaný. Hlavným cieľom tejto funkcie nie je pripomenutie úloh používateľom, ale kontrola štatistík z ich plnenia. Cieľ práce návrh a popis vývoja mobilnej aplikácie a následné zhodnotenie jej využiteľnosti a definovanie zámerov budúceho vývoja bol splnený.

I.Summary

This work deals with the idea of multimedia mobile application development. The theoretical part describes the main division of app categories, history of development, most common operation systems and architecture. It deals with psychological factors used in the creation of applications in order to shape the user's behavior and build his habits. The practical part is related to the issue of programming multimedial notepad for smartphones using component-based framework for building scalable applications. The resulting application allows users to record multimedial notes with a time record, share notes on social networks and display basic statistics over stored data. The mobile app uses Angular to develop the application and Firebase Realtime Database for storing data. In the conclusion this work analyzes functionality of an application and recommends implementation of functions for future development.

Keywords: mobile app development, Typescript, Angular, users behaviour , operation systems, framework, multimedial notes, Firebase

II. Zoznam použitých zdrojov

1. 147 Mobile Gaming Statistics for 2022 That Will Blow Your Mind | Udonis. (2022). 2022, z https://www.blog.udonis.co/mobile-marketing/mobile-games/mobile-gaming-statistics
2. Android - Architecture. (2022). 2022, z https://www.tutorialspoint.com/android/android_architecture.htm
3. Avatar, Y. T. (2021). Apple's App Store: Pros And Cons of the Store. 2022, z https://yourtechavatar.com/apples-app-store-pros-and-cons-of-the-store/
4. Bartles Player Types for Gamification. (2020). 2022, z https://www.interaction-design.org/literature/article/bartle-s-player-types-for-gamification
5. Bělovský, V. B. (2013). Vývoj mobilních aplikací pro platformu Apple iOS. Retrieved 2022, from https://is.muni.cz/th/n9cb8/Bakalarka.pdf
6. Best Online Training & Video Courses | eduCBA. (2019, June 26). 2022, z https://www.educba.com
7. A Brief History of Apple Computers. (2019). 2022, z https://www.thoughtco.com/the-history-of-apple-computers-1991454
8. ČÁpka, D. (2022). itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. z https://www.itnetwork.cz
9. Co je to IDE? | Jak to funguje | Potřeba a rozsah Dovednosti a výhody IDE. (2022). 2022, z https://cs.education-wiki.com/2488883-what-is-ide
10. Co je to kód Visual Studio? | Vlastnosti a výhody Rozsah a kariéra. (2022). 2022, z https://cs.education-wiki.com/3958588-what-is-visual-studio-code
11. Dajbych, V. (2013). K čemu je dobrý TypeScript. 2022, z https://zdrojak.cz/clanky/k-cemu-je-dobry-typescript/

12. Edpresso Team. (2022). What is Firebase? 2022, z https://www.educative.io/edpresso/what-is-firebase
13. GeeksforGeeks. (2021). Architecture of IOS Operating System. 2022, z https://www.geeksforgeeks.org/architecture-of-ios-operating-system/
14. História :: Internetové sociálne siete. (2014). z https://socialne-siete8.webnode.sk/historia/
15. K, J. (2019). The History and Evolution of Mobile Apps. 2022, z https://acodez.in/evolution-mobile-apps/#The_History_of_the_Development_of_Mobile_Apps
16. Koďousková, B. (2021). Android vs. iOS: kterou platformu zvolit pro vývoj mobilní aplikace 2022, z https://www.rascasone.com/cs/blog/ios-vs-android-vyvoj-mobilnich-aplikaci
17. Krishnankutty, P. (2020).Nokia's Snake, the mobile game that became an entire generation's obsession. 2022, z https://theprint.in/features/nokias-snake-the-mobile-game-that-became-an-entire-generations-obsession/462873/
18. MacPherson, L. (2018). The Psychology of App Design: How to Help Users Reach Their Goals Using Your App. 2022, z https://designli.co/blog/the-psychology-of-app-design-how-to-help-users-reach-their-goals-using-your-app/
19. The rise and fall of Vine: A brief timeline. (2020). 2022, z https://businesschief.com/technology-and-ai/rise-and-fall-vine-brief-timeline-1
20. Shah, N. (2022). The Evolution of Mobile Apps - 1994 through 2016. 2022, z https://arkenea.com/blog/evolution-of-mobile-apps/
21. Shaikh, M. (2021). A Step-By-Step Guide To Mobile App Development. 2022, z https://www.mindbowser.com/a-step-by-step-guide-to-mobile-app-development/

22. Shore, J. (2021). cloud application. 2022, z https://www.techtarget.com/searchcloudcomputing/definition/cloud-application
23. Singh, V., A., T., L., K., Butt, S., & DigiMantra Labs. (2022). What is Frameworks? [Definition] Types of Frameworks. 2022, z https://hackr.io/blog/what-is-frameworks
24. smartphone. (2022). 2022, z https://www.merriam-webster.com/dictionary/smartphone
25. Smejkal, P. (2021). Co je API a k čemu slouží? 2022, z https://petrsmejkal.cz/clanky/co-je-api-a-k-cemu-slouzi/
26. ŠTráfelda, J. (2021). Co je HTML. 2022, z https://www.strafelda.cz/html
27. Techopedia. (2020). Android SDK. 2022, z https://www.techopedia.com/definition/4220/android-sdk
28. Trlica, D. (2017). Aplikace Duolingo aneb jak trénovat online jazyky. 2022, z https://www.svetandroida.cz/duolingo-online-jazyky/
29. TypeScript - Overview. (2022). 2022, z https://www.tutorialspoint.com/typescript/typescript_overview.htm
30. TypeScript Components - javatpoint. (2019). z https://www.javatpoint.com/typescript-components
31. Vanhaesebroeck, J. (2021). 3 quick & easy ways to improve user retention on your mHealth app. 2022, z https://strivecloud.io/blog/app-gamification/mhealth-gamification-apps/#163f5
32. Who Invented Social Media? (2020). 2022, z https://wonderopolis.org/wonder/who-invented-social-media
33. Wikipedia contributors. (2021). Wireframe. 2022, z https://cs.wikipedia.org/wiki/Wireframe

34. Wikipedia contributors. (2022). Android (operační systém). 2022, z https://cs.wikipedia.org/wiki/Android_(opera%C4%8Dn%C3%AD_syst%C3%A9m)
35. Wikipedia contributors. (2022). Google Play. 2022, z https://cs.wikipedia.org/wiki/Google_Play
36. Wikipedia contributors. (2022). IOS. 2022, z https://cs.wikipedia.org/wiki/IOS
37. Wikipedia contributors. (2022). Multithreading (computer architecture). 2022, z https://en.wikipedia.org/wiki/Multithreading_(computer_architecture)
38. Wikipedia contributors. (2022). Bolt (company). 2022, z https://en.wikipedia.org/wiki/Bolt_(company)
39. Wikipedia contributors. (2022). Netflix. 2022, z https://en.wikipedia.org/wiki/Netflix

III. Zoznam obrázkov, tabuliek

Obrázok 1 Prehľad operačných systémov mobilných zariadení 19
Obrázok 2 Architektúra iOS 25
Obrázok 3 Architektúra frameworku Angular 36
Obrázok 4 Subory frameworku Angular 37
Obrázok 5 Štruktúra aplikácie NODs vo framewroku Angular 40
Obrázok 6 Úloha pridaná do databázy obsahujúca audio súbor 52

IV. Zoznam zdrojového kódu

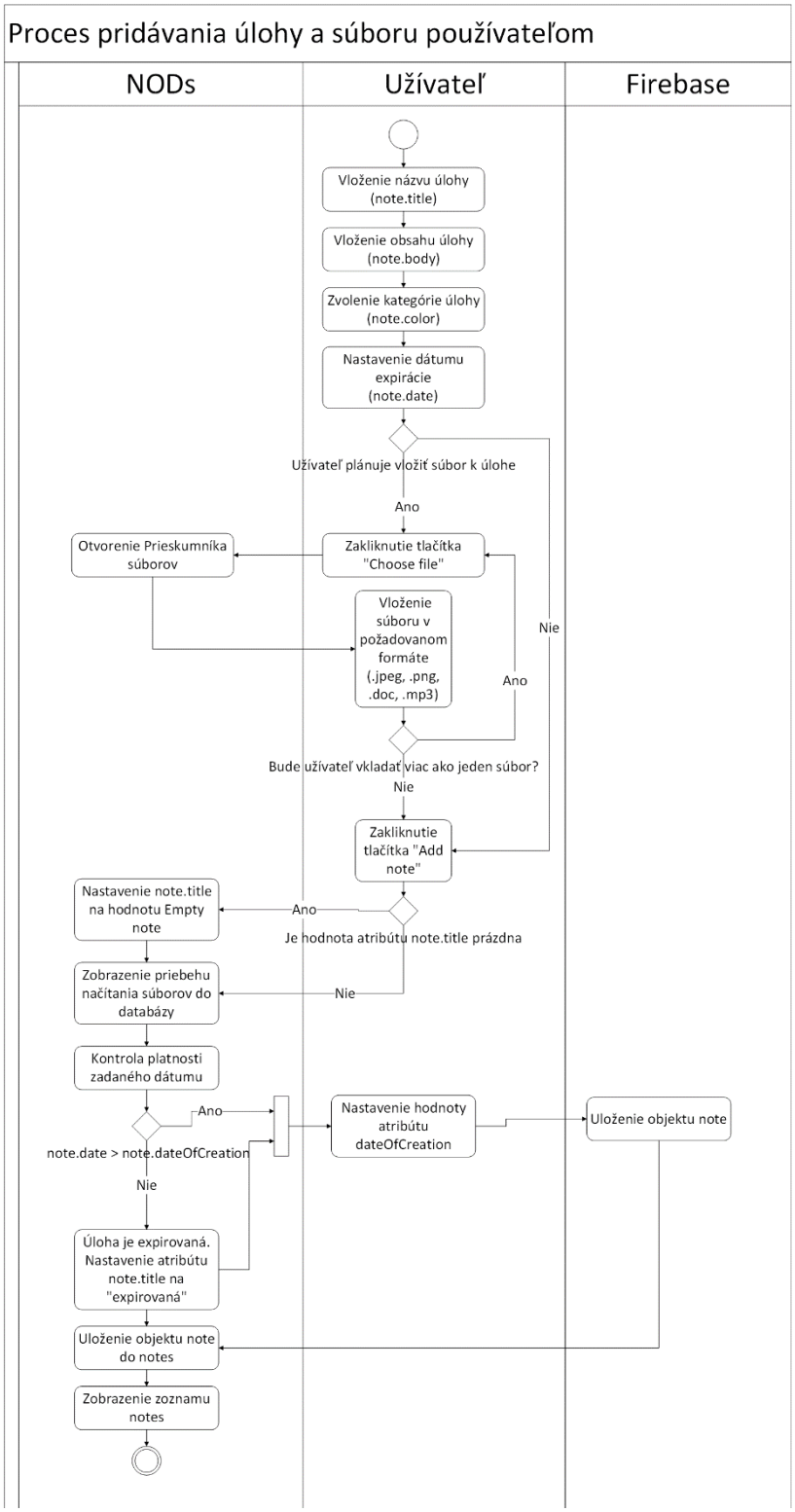
Ukážka kódu 1 environment.ts 41
Ukážka kódu 2 Funkcia getNotes 42
Ukážka kódu 3 Funkcia addNote 42
Ukážka kódu 4 Funkcia deleteNote 43
Ukážka kódu 5 Funkcia updateNote 43
Ukážka kódu 6 Funkcia nhONInit 44
Ukážka kódu 7 Funkcia getNotes 45
Ukážka kódu 8 Funkcia expiredNoteCounter 45
Ukážka kódu 9 Funkcia formatNotes 46
Ukážka kódu 10 Funkcia onAddNoteClick 46
Ukážka kódu 11 Funkcia onEditClick 47
Ukážka kódu 12 Funkcia onDeleteClick 48
Ukážka kódu 13 Funkcia onSelectFile 49
Ukážka kódu 14 Funkcia taskPercentageCalculator 49
Ukážka kódu 15 Funkcia convertBase64ToBlobData 50
Ukážka kódu 16 Funkcia downloadFile 51
Ukážka kódu 17 app.routing.module.ts 51
Ukážka kódu 18 app.component.ts 51

V. Zoznam príloh

Príloha A : Diagram činnosti 65
Príloha B : Testovacie scenáre 66
Príloha C : Zobrazenie poznámky obsahujúcej audio súbor v NODs 67

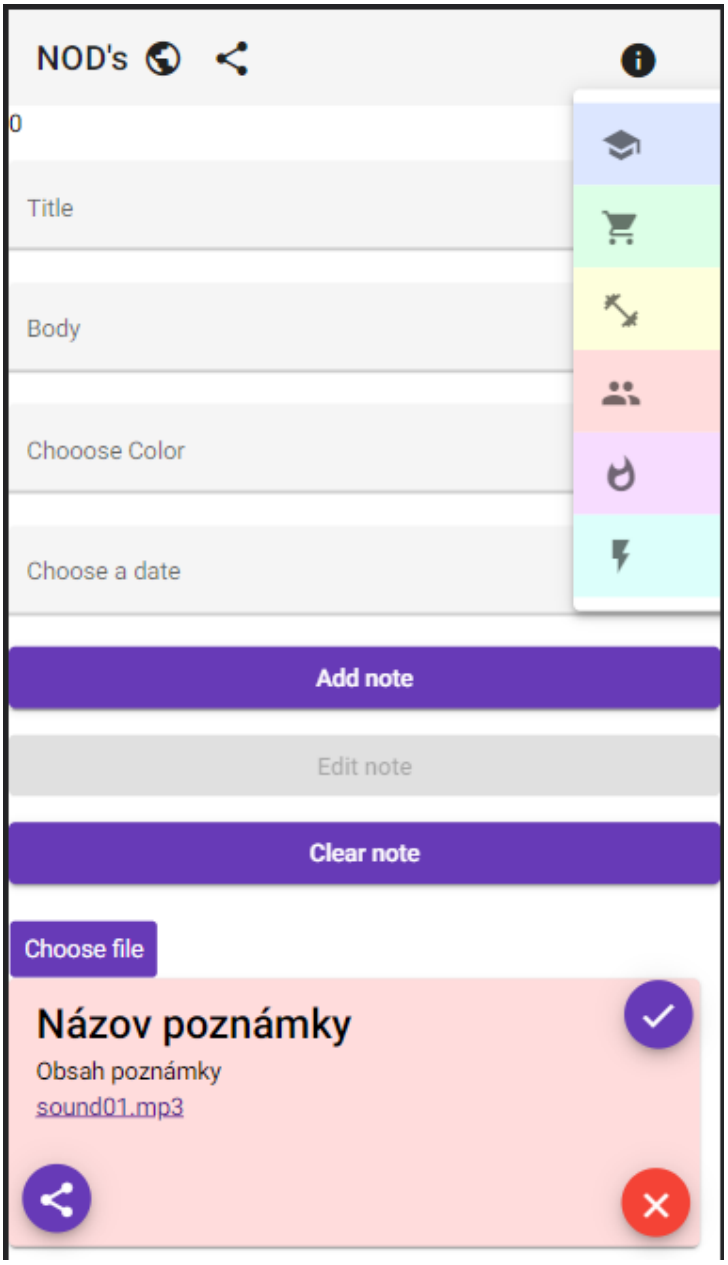
VI. Prílohy

Príloha A : Diagram činnosti



Akcie	Predpoklady	Inštrukcie	Očakávaný výsledok
Pridať úlohu	Užívateľ vyplnil názov, obsah a dátum exspirácie úlohy, kategóriu úlohy a klikol na tlačidlo „Add note“.	Dátum exspirácie pridanej úlohy musí byť minimálne dátum nasledujúceho dňa. Názov a obsah zadanej úlohy nesmie byť prázdny.	Užívateľ pridal úspešne úlohu, úloha sa uloží do databáze a zobrazí v zozname úloh.
Odstrániť exspirovanú úlohu	Užívateľ pridal úlohu a neklikol na tlačidlo so symbolom „splnená“, v časovom intervale od dátumu pridania úlohy do dátumu nastaveného ako exspirácia úlohy. Klikne na tlačidlo so symbolom „odstrániť“.	Dátum dňa odstránenia úlohy je minimálne deň stanovaný na dokončenie úlohy, alebo ide o starší dátum.	Užívateľ odstránil úlohu, odstránená úloha sa započíta do vyhodnocovania splanených a nesplnených úloh ako nesplnená. Úloha sa odstráni z databázy.
Odstrániť neexspirovanú úlohu	Užívateľ pridal úlohu, ktorú chce zo zoznamu odstrániť pred exspiráciou. Klikne na tlačidlo so symbolom „odstrániť“.	Dátum dňa odstránenia úlohy zadanej používateľom je pred dátumom exspirácie.	Užívateľ odstránil úlohu, odstránená úloha sa nezapočíta do vyhodnocovania splanených a nesplnených úloh. Úloha sa odstráni z databáze.
Potvrdiť úspešné ukončenie úlohy	Užívateľ pridal úlohu, ktorú sa mu podarilo splniť v zadanom intervale od dátumu pridania do dňa s dátumom exspirácie.	Dátum dňa ukončenia úlohy zadanej používateľom sa nachádza v intervale od dátumu pridania do dňa pred zadaným dátumom exspirácie.	Úloha sa započíta do vyhodnocovania splanených a nesplnených úloh ako splnená. Úloha sa odstráni z databáze.
Pridať úlohu so súbormi	Užívateľ vyplnil názov, obsah a dátum úlohy, kategóriu úlohy a klikol na tlačidlo „Choose file“. Používateľovi sa otvorí prieskumník súborov kde zvolí súbor a klikne na tlačidlo „otvoriť“. V prípade potreby pridať viac súborov túto akciu zopakuje. Po ukončení pridávania súborov klikne na tlačidlo „Add note“	Dátum pridanej úlohy musí byť minimálne dátum nasledujúceho dňa. Názov a obsah zadanej úlohy nesmie byť prázdny. Formáty pridaných súborov musí byť (.jpeg, .png, .doc, .mp3).	Užívateľ pridal úspešne úlohu, úloha so súbormi sa uloží do databáze a zobrazí v zozname úloh.
Upraviť obsah pridanej úlohy	Používateľ pridal úlohu úspešne, klikol na zvolenú úlohu a upravil údaje úlohy zobrazené vo vstupoch. Po upravení klikol na tlačidlo „Edit“.	Používateľ môže po kliknutí na úlohu upraviť jej názov, obsah a dátum exspirácie a jeho kategória.	Úloha sa úspešne aktualizuje v databáze a zobrazí s novými údajmi v zozname úloh.
Odstrániť obsah úlohy	Užívateľ klikol na úlohu a nasledovne na tlačidlo „Clear note“, po kliknutí sa údaje vynulujú.	Používateľ môže túto možnosť použiť pri rýchlej úprave obsahu.	Vstupy vyplnené používateľom sa vynulujú.

; *Príloha C : Zobrazenie poznámky obsahujúcej audio súbor v NODs*



VII. Kód

Kód je dostupný na nasledujúcom odkaze: https://github.com/pavolh01/pavol-bc-master