FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Mobilní aplikace pro evidenci pronájmu reklamních ploch
Student:	Bc. Anna Škorupová
Vedoucí:	Ing. Jiří Hunka
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Cílem práce je návrh a implementace mobilní android aplikace, která umožní komunikaci se serverem společnosti, zaměřené na provoz reklamních médií. Aplikace bude sloužit pro evidenci pronájmu reklamních ploch a také pro dokumentaci instalace a technického servisu reklam a reklamních nosičů. Cílem práce je i návrh uživatelského rozhraní.

Postupujte v těchto krocích:
1. Analyzujte potřeby zákazníků v kontextu návrhu mobilní aplikace.
2. Definujte funkční a nefunkční požadavky a klíčové vlastnosti navrhované mobilní aplikace.
3. Seznamte se s dokumentací REST API serveru firmy.
4. Na základě analýzy proveďte řádný softwarový návrh, včetně papírového modelu.
5. Zvolte vhodné technologie a architekturu, aby byla aplikace napsaná s použitím moderních postupů.
6. Implementujte prototyp aplikace.
7. Řešení podrobte vhodným testům.
8. Na základě testů navrhněte úpravy a realizujte výslednou aplikaci.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 25. února 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práca

Mobilná aplikácia na evidenciu prenájmu reklamných plôch

Bc. Anna Škorupová

Katedra softwarového inženýrství
Vedúci práce: Ing. Jiří Hunka

27. júna 2021

Poďakovanie

Chcela by som poďakovať svojmu vedúcemu práce, ako aj rodine a priateľom, ktorý ma podporovali pri písaní tejto diplomovej práce.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov, a skutočnosť, že České vysoké učení technické v Praze má právo na uzavrenie licenčnej zmluvy o použití tejto práce ako školského diela podľa § 60 odst. 1 autorského zákona.

V Prahe 27. júna 2021 . . . . . .

Odkaz na túto prácu

Škorupová, Anna. *Mobilná aplikácia na evidenciu prenájmu reklamných plôch.* Diplomová práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Táto diplomová práca sa zaoberá analýzou, návrhom a implementáciou Android mobilnej aplikácie, ktorá komunikuje so serverom spoločnosti zameranej na prevádzkovanie reklamných médií. Aplikácia slúži na evidenciu prenájmu reklamných plôch a zároveň na dokumentáciu inštalácie a technického servisu reklám a reklamných nosičov. Na základe analýzy potreby zákazníkov som zvolila vhodnú architektúru, technológie a vytvorila návrh používateľského rozhrania, na ktorom som realizovala testovanie použiteľnosti. Následne som aplikáciu implementovala a podrobila automatizovaným testom.

Klíčová slova mobilná aplikácia, Android, Kotlin, reklamné plochy, reklamné nosiče, reklamné kampane

Abstract

This diploma thesis deals with the analysis, design and implementation of an Android mobile application that communicates with the server of a company focused on the operation of advertising media. The application is used to record the lease of advertising space and to document the installation and technical service of advertisements and advertising holders. Based on the analysis of customer needs, I chose the appropriate architecture, technology and created a user interface design, on which I performed usability testing. Subsequently, I implemented the application and subjected it to automated tests.

Keywords mobile application, Android, Kotlin, advertising space, advertising holders, advertising campaigns

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

História technologického vývoja prešla niekoľkými zlomovými momentami. Rovnako ako nám v 18. storočí para priniesla priemyselnú revolúciu, tak aj v súčasnosti majú inovácie v informačných technológiách kľúčový dopad na priemysel. „Používanie technológie sa vyvinulo z automatizácie štruktúrovaných procesov na systémy, ktoré sú skutočne revolučné v zavádzaní zmien do základných obchodných postupov [8].“

V období neustálych a rýchlych zmien dokážu byť úspešné len tie spoločnosti, ktoré naplno využívajú technologické inovácie. Ako už v roku 1988 predpovedal časopis Fortune „Už to nebude len pomoc zo strany počítačov. Spoločnosti nimi budú žiť, formovať stratégiu a štruktúru tak, aby vyhovovali novým informačným technológiám. [9]“. Na jednej strane je zrejmé, že z digitálnej revolúcie vyťažili podniky naprieč všetkým odvetviam, avšak množstvo spoločností, inak predurčených k neúspechu, bolo k tomuto prijatiu inovácií prinútených. Ustavične sa meniace požiadavky a preferencie používateľov si to nepochybne vyžadujú.

Vo svete biznisu je nesmierne dôležitá spoľahlivosť a presnosť dát v požadovanom čase. A práve pomocou kvalitných informačných systémov (IS) vieme dosiahnuť túto presnosť a rýchlu dostupnosť informácií. IS spoločnostiam pomáhajú nie len s ich každodennými procesmi, ale aj s vytváraním rozhodnutí a plánov. Príkladom sú obchodné komunikačné systémy, riadenie obchodných operácií, rozhodovanie spoločnosti, či vedenie záznamov o spoločnosti [10]. Kvalita IS teda do veľkej miery predurčuje úspech spoločností.

S mobilitou ako kľúčovým očakávaním používateľov vzrástla dôležitosť mobilných IS. Mobilné technológie sa dokázali veľmi rýchlo prispôsobiť potrebám ľudí. Zo zariadení určených iba na telefonovanie sa postupným pridávaním jednoduchých aplikácií, ako napríklad kalendár a hry, stali multifunkčné zariadenia, hosťujúce širokú škálu aplikácií [11].

V priebehu posledných niekoľkých rokov sledujeme výrazný pokles cien smartfónov súčasne s nárastom ich funkcionality. Aj z tohto dôvodu to už

dávno nie sú unikátne nástroje, dostupné len určitej skupine ľudí. Mobilné telefóny používajú miliardy ľudí po celkom svete, vrátane rozvojových krajín. Mobilné IS zmenili spôsob práce, umožňujúc prístup k informáciám kedykoľvek a odkiaľkoľvek.

V tejto diplomovej práci sa zaoberám mobilným IS pre mediálnu spoločnosť, zameranú na prevádzku a prenájmom reklamných plôch. Objednávky, ktoré táto spoločnosť prijíma, sú reprezentované reklamnými kampaňami, ktoré sú definované časovým obdobím a obsahom. Reklamná kampaň pozostáva z reklamných plôch (napr. nákupné centrum), pričom každá reklamná plocha je tvorená viacerými reklamnými nosičmi (lepiaca plocha, regály, nákupné vozíky, atď.). Denná agenda tímov technických pracovníkov tejto spoločnosti predstavuje servis a inštalácie reklám na reklamné nosiče podľa stanoveného časového harmonogramu.

Zadávajúca spoločnosť v poslednom období prešla nárastom reklamných kampaní, ako aj reklamných plôch a reklamných nosičov. To si vyžiadalo aj nárast počtu technických pracovníkov a zlepšenie harmonogramu inštalácií a servisu reklamných nosičov. Technická správa reklamných nosičov sa stala logisticky náročnejšia, v dôsledku čoho vznikla potreba vytvoriť IS na evidenciu prenájmu reklamných plôch a zároveň na dokumentáciu inštalácie a technického servisu reklám a reklamných nosičov.

Tento IS má podporiť rýchlosť a efektivitu technických pracovníkov, ako aj kontrolu a vyhodnotenie práce týchto zamestnancov. Pre manažérov má slúžiť na prehľad stavu reklamných kampaní a evidenciu prenájmu reklamných plôch. Má byť realizovaný jednoduchou formou, vhodnou pre technických pracovníkov, z čoho vznikla požiadavka na IS vo forme mobilnej aplikácie.

Cieľom tejto diplomovej práce je návrh a implementácia diskutovanej mobilnej aplikácie, ktorá má umožniť komunikáciu so súkromným serverom zadávajúcej spoločnosti, a to so zameraním na návrh používateľského rozhrania. V prvej časti tejto práce vykonám analýzu potrieb zákazníka, definujem funkčné a nefunkčné požiadavky na aplikáciu a zoznámim sa s REST API servera zadávajúcej spoločnosti.

Kapitola 1

Analýza

Na úvod si priblížime rôzne typy aplikácií. Uvedieme si vlastnosti mobilných aplikácií a popíšeme rozdiely medzi jednotlivými typmi mobilných aplikácií.

V ďalšej časti prebehne analýza požiadaviek na aplikáciu. Zadefinujeme si rôzne skupiny, ako aj rôzne metódy zberu požiadaviek, a nakoniec popíšeme výslednú formuláciu funkčných a nefunkčných požiadaviek. Na základe týchto požiadaviek budú identifikované prípady použitia.

Následne sa vykoná analýza aplikačnej platformy. V rámci tejto časti sa porovnajú rôzne mobilné platformy a na základe toho sa vyberie platforma najvhodnejšia pre túto aplikáciu, spolu s minimálnou podporovanou verziou. Ďalej sa tiež porovnajú najpoužívanejšie programovacie jazyky pre túto platformu a vyberie sa najvhodnejší.

Na záver tejto kapitoly sa zoznámime s API súkromného servera zadávajúcej spoločnosti.

1.1 Mobilné aplikácie

Dnešné mobilné telefóny, zariadenia s neuveriteľne malým a výkonným výpočtovým hardvérom, nám umožňujú voľne vytvárať, ukladať, spracovávať a pristupovať k informáciám takmer nezávisle od našej polohy [12, 13]. S rýchlym vývojom mobilných komunikačných technológii a faktom, že chytré telefóny sa stali súčasťou nášho každodenného života, sa vývoj mobilných aplikácií stal veľkou a rýchlo rastúcou súčasťou softvérového priemyslu. Množstvo vyvinutých a používaných mobilných aplikácií sa každým rokom zvyšuje, avšak je potrebné si uvedomiť, že vývoj mobilných aplikácií sa líši od vývoja tradičných desktopových aplikácií. Pre vývoj kvalitnej mobilnej aplikácie je preto dôležité pochopiť kľúčové vlastnosti, ktoré charakterizujú mobilné aplikácie. [14]

V tejto kapitole si najprv definujeme rôzne typy aplikácií, od štandardných webových aplikácií po natívne aplikácie. Následne rozoberieme charakteristické vlastnosti mobilných aplikácií, a aj tie, ktorými sa odlišujú od desk-

topových aplikácií. Na záver si uvedieme rozdiely medzi jednotlivými typmi mobilných aplikácií.

1.1.1 Typy aplikácií

K úspešnému vývoju aplikácie je dôležité vedieť rozlíšiť rôzne typy aplikácií. V tejto kapitole si uvedieme jeden zo spôsobov, akým je možné aplikácie klasifikovať.

Štandardné webové aplikácie

Pod týmto pojmom rozumieme tradičné aplikácie, ku ktorým vieme pristupovať prostredníctvom prehliadačov primárne z desktopových počítačov, avšak vieme k ním pristúpiť aj pomocou mobilných zariadení. Sú vyvíjané pomocou webových technológií a pristupujeme k ním unikátnou URL. Pomocou tejto adresy sa dopytujeme na vzdialený server, ktorý nám odpovedá prostredníctvom prenosového protokolu. [5]

Responzívne webové aplikácie

Responzívne webové aplikácie sú webové aplikácie, ktoré využívajú responzívny webový dizajn. To znamená, že aplikácia síce má jediný zdroj obsahu, avšak disponuje viacerými variantami štýlu. Server sa na základe vlastností zariadenia (napríklad rozmery), ktoré obsluhuje, rozhodne, ktorý štýl využije na vykreslenie obsahu webovej aplikácie. [5]

Mobilné webové aplikácie

Sú to webové aplikácie optimalizované pre mobilné zariadenia, ku ktorým pristupujeme prostredníctvom natívneho prehliadača. Rovnako ako štandardné webové aplikácie, sú vyvíjané pomocou webových technológií a reprezentované unikátnou URL. Znamená to, že nie je potrebná inštalácia do zariadenia, avšak výkon aplikácie závisí od výkonnosti prehliadača. Tieto aplikácie nevyžadujú pamäť zariadenia, pretože aplikačné dáta sú uložené na vzdialenom serveri. [15]

Mobilné webové aplikácie sú efektívne z hľadiska vývoja ako aj údržby, a taktiež sú plne prenositeľné. Tým poskytujú jednotnosť naprieč platformami. Nevýhodou týchto aplikácií je, že nemajú plný prístup k nízkoúrovňovým funkciám zariadenia a tiež ťažšie zvládanie veľkej grafickej záťaže. [15]

Natívne aplikácie

Pod týmto pojmom rozumieme aplikácie, ktoré je potrebné inštalovať do nášho zariadenia, pričom všetky dáta potrebné pre funkčnosť aplikácie sú

uložené na danom zariadení. To znamená, že nie je potrebná komunikácia so serverom, a teda ani žiadne sieťové pripojenie. [14]

Natívne aplikácie priamo využívajú služby poskytované mobilnou platformou, na ktorej sú postavené, a to prostredníctvom API, čiže aplikačného programového prostredia. Toto prostredie ponúka funkcie týkajúce sa grafiky, bezpečnosti, lokácie, či komunikácie a posielania správ [16]. Vďaka tomu môžu mať tieto aplikácie lepší výkon, grafiku a tiež lepší užívateľský zážitok. Natívne aplikácie teda môžu plne využívať funkcie daného operačného systému, avšak za cenu toho, že jazyk, ako aj vývojové prostredie využívané pri vývoji aplikácie, sú špecifické pre danú platformu. To vedie k takzvanej fragmentácii, čo znamená, že aplikácia napísaná v jazyku špecifickom danej platformy nie je prenositeľná na inú platformu [17]. Aplikáciu je teda potrebné udržiavať pre viacero platforiem, čo má za následok viac nákladov, viac času stráveného testovaním a údržbou a hlavne horšiu prenositeľnosť. Natívne aplikácie sú distribuované prostredníctvom špecializovaných obchodov s aplikáciami, ako napríklad Google Play Store pre operačný systém Android. [15]

Hybridné aplikácie

Hybridné aplikácie sú webové aplikácie zaobalené do natívnej aplikácie. Rovnako ako webové aplikácie sú vyvíjané pomocou webových technológií a na plné fungovanie potrebujú prístup na internet. To znamená, že môžu byť distribuované pre rôzne mobilné platformy, vrátane platforiem Android a iOS. Tieto aplikácie je ale potrebné inštalovať ich do zariadenia, vďaka čomu vyzerajú ako natívne aplikácie, pristupujúc k rovnakým funkciám mobilného zariadenia. Rovnako ako natívne aplikácie môžu byť nainštalované prostredníctvom špecializovaných obchodov s aplikáciami. [5, 15]

Progresívne webové aplikácie

Sú to webové aplikácie, ktoré sa vyznačujú charakteristikami ako podpora pri slabom alebo žiadnom sieťovom pripojení, bezpečnosť , možnosti spracovania na pozadí, podpora pre push notifikácie. Progresívne webové aplikácie sú takisto obsluhované vzdialeným serverom prostredníctvom protokolu HTTPS, čo znamená, že oproti hybridným a natívnym aplikáciám nenastáva oneskorenie aktualizácií. [15]

K týmto aplikáciám môžeme pristupovať rovnako ako k webovým aplikáciám, čiže prostredníctvom webového prehliadač. Následne máme možnosť inštalácie tejto aplikácie do nášho zariadenia, čo znamená využitie celej obrazovky. Progresívne webové aplikácie po inštalácii podporujú režim bez sieťového pripojenia, a to napríklad prostredníctvom využívania pamäte cache a push notifikácií. [15]

1.1.2 Vlastnosti mobilných aplikácií

V tejto kapitole čerpám z [14]. Mobilné aplikácie sa od tradičných desktopových aplikácií líšia vo veľkej miere ako vlastnosťami, tak aj vývojovým procesom. Základom pre vývoj kvalitnej mobilnej aplikácie je preto dobrá znalosť dôležitých vlastností, ktoré charakterizujú mobilné aplikácie, ako aj tie, ktorými sa líšia od desktopových aplikácií.

Medzi základné charakteristické črty kvalitnej mobilnej aplikácie patria spoľahlivosť, efektívnosť, funkcionalita, flexibilita, prenositeľnosť, udržateľnosť, použiteľnosť, ako aj dobre zadefinované používateľské požiadavky [18, 19, 20, 21, 22]. Mobilné telefóny sú tiež osobnejšie, pretože oproti desktopovým počítačom sú určené pre jediného používateľa a poznajú množstvo jeho osobných údajov.

Pri vývoji mobilných aplikácií sa musíme vysporiadať s mnohými problémami, ktorými sa vyznačuje aj klasické softvérové inžinierstvo. Takéto problémy zahŕňajú napríklad spoľahlivosť, výkonnosť, bezpečnosť, pamäťové obmedzenie alebo integráciu s iným zariadením [23]. Avšak vývoj mobilných aplikácií prináša aj problémy, ktoré sú charakteristické len pre tento typ vývojového procesu, ako napríklad integrácia so senzormi daného zariadenia, spotreba energie, integrácia s inými aplikáciami, problémy spojené s charakteristikami či už natívnych alebo hybridných aplikácií, krátkotrvajúce relácie, alebo zložitosť testovania [24, 25].

Čo sa týka stanovenia priorít obsahu, mobilné aplikácie sa od desktopových aplikácií líšia v napríklad gravitáciou rozloženia (vertikálne namiesto horizontálneho), textami, grafikou, indikátorom procesného stavu, globálnou, ako aj kontextovou navigáciou, pätičkou, hypertextom, lokalizovaným, ako aj personalizovaným vyhľadávaním, alebo tiež využívaním funkcionalít zariadenia.

V tejto kapitole si ďalej uvedieme základné vlastnosti, ktoré odlišujú mobilné aplikácie od desktopových aplikácií, a to v kategóriách hardware a software. Tieto poznatky boli získané na základe online prieskumu [14] vykonanom v komunite mobilného vývoja a výskumu.

1.1.2.1 Hardware

Mobilné telefóny majú oproti desktopovým počítačom nižšiu výdrž batérie, slabšiu výkonnosť CPU, a v neposlednom rade aj menšiu pamäť. Aj keď je pravdou, že mobilné telefóny vďaka kratším kódom potrebným pre beh mobilných aplikácií nevyžadujú veľa diskového priestoru, tak v porovnaní s desktopovými počítačmi sú stále menej výkonné. Z tohto dôvodu je pri vývoji mobilných aplikácií veľmi dôležité dbať na dobrú architektúru a efektívnosť kódu.

Pre mobilné aplikácie je aj čas spustenia dôležitejším faktorom ako pre desktopové aplikácie, pretože používatelia mobilných telefónov sú zvyknutí na krátke, efektívne a časté relácie s rýchlou odozvou, zatiaľ čo desktopové

aplikácie sú používané na dlhšie relácie, zamerané skôr na skúmanie informácií do hĺbky.

Ďalšia dôležitá charakteristika, ktorá odlišuje mobilné aplikácie od desktopových, je využívanie vstupných zariadení mobilných telefónov. Väčšina dnešných smartfónov obsahuje napríklad akcelerometer merajúci zrýchlenie pohybu zariadenia, dotykové obrazovky, ktoré poskytujú používateľské vstupy vo forme dotyku, potiahnutia prstom po obrazovke, či rôznych iných gest, ďalej mikrofón, viacero kamier, reálnu alebo virtuálnu klávesnicu, či globálny pozičný systém [24]. Platí zásada, že ak chceme vytvoriť používateľsky príjemnú mobilnú aplikáciu, mala by od používateľa vyžadovať čo najmenej vstupov prostredníctvom klávesnice, a radšej danému používateľovi poskytnúť možnosti, ktoré má na výber.

Kvôli malým rozmerom mobilných telefónov sú aj veľkosť a tvar obrazovky, rovnako ako štýl mobilného zariadenia a rozloženie prvkov dôležitými faktormi. A keďže tieto vlastnosti sa u mobilných telefónov veľmi líšia, je počas vývoja mobilnej aplikácie vhodné zamerať sa len na určitý počet funkcií, ktoré sa majú správať rovnako, nezávisle na rozmeroch mobilného telefónu.

Fyzické parametre prostredia, v akom sú používané mobilné zariadenia, ako napríklad pohyby, vibrácie, hluk, osvetlenie, či teplota a vlhkosť, sa na rozdiel od desktopových počítačov môžu veľmi líšiť. Pri návrhu mobilných aplikácií treba brať do úvahy tieto premenlivé parametre prostredia, ako aj fakt, že používatelia venujú mobilným aplikáciám len malú pozornosť. Riešením je zamerať sa na rýchle používanie aplikácie a používateľom ponúkať malý počet možností.

Čo sa týka fragmentácie zariadenia, u mobilných aplikácií je na rozdiel od desktopových aplikácií dôležité, aby boli nie len kompatibilné s rôznymi verziami mobilných zariadení a operačných systémov, ale tiež aby boli naprogramované pre rôzne mobilné platformy.

1.1.2.2 Software

Vlastnosti mobilných aplikácií týkajúce sa softvéru rozdelíme na vlastnosti súvisiace s interakciou aplikácie a vlastnosti súvisiace s procesom vývoja mobilnej aplikácie.

Interakcia s aplikáciou

Pri návrhu mobilnej aplikácie je veľmi dôležité dbať na užívateľský zážitok, označovaný tiež ako UX. Ak chceme dosiahnuť čo najlepší užívateľský zážitok, musíme mobilnú aplikáciu navrhnúť takým spôsobom, aby sa používateľ počas interakcie s aplikáciou cítil príjemne a tiež by mal mať pocit, že je schopný dosiahnuť akúkoľvek úlohu. Na dosiahnutie dobrého užívateľského zážitku je odporúčané zamerať sa na účel mobilnej aplikácie, a tiež dbať na odozvu a rýchly čas spustenia aplikácie.

Ďalším dôležitým aspektom, na ktorý treba klásť dôraz pri návrhu mobilnej aplikácie, a ktorý súvisí s užívateľským zážitkom, je užívateľské rozhranie. Na to, aby sme dosiahli aplikáciu príjemnú na používanie, je potrebné, aby bola konzistentná. To znamená, že by mala byť pocitovo v súlade s mobilnou platformou, na ktorej aplikácia beží, a teda mala by zdieľať štýl základných elementov s inými aplikáciami [24]. K tomu je potrebné dodržiavať usmernenia používateľského rozhrania hosťujúcej platformy, ako aj programátorské štandardy danej platformy. Čo sa týka štandardov používateľského rozhrania, mnohé z nich sú už priamo implementované v SDK, ktoré je súčasťou platformy [24]. Pri návrhu je tiež dôležité myslieť na to, že aplikácia má používateľovi ponúkať jasne vymedzené užívateľské cesty namiesto viacerých možností, ako aj možnosť kroku späť a ukončenia akcie v akomkoľvek stave aplikácie.

Po každom dokončení akcie by mobilná aplikácia mala dať používateľovi spätnú väzbu. Pri návrhu mobilnej aplikácie je preto potrebné zabezpečiť, aby v žiadnom okamihu práce s aplikáciou nastala situácia, kedy by aplikácia nereagovala a používateľ by tak nebol oboznámený o stave vykonanej akcie.

Kvalitné mobilné aplikácie sa tiež vyznačujú efektívnym chybovým hlásením. To znamená, že používateľ by mal dané chybové hlásenie rýchlo zaregistrovať a porozumieť mu, a na základe získanej informácie by mal byť schopný dopátrať sa k problému a vyriešiť ho. Mobilná aplikácia by tiež mala byť schopná používateľa na vzniknutý problém upozorniť aj v momente, keď aktuálne nepoužíva danú aplikáciu, a to prostredníctvom push notifikácií.

Ďalšou vlastnosťou mobilných aplikácií súvisiacou so softvérom je integrácia s inými aplikáciami. Väčšina zabudovaných zariadení disponuje len softvérom, ktorý bol do tohto zariadenia nainštalovaný v priebehu výroby. Smartfóny však obsahujú množstvo aplikácií z rôznych zdrojov, ktoré spolu môžu komunikovať [24]. Je preto dôležité, aby táto integrácia s inými aplikáciami a dátami bola pred nasadením aplikácie dôkladne otestovaná a verifikovaná.

Vývojový proces

Mobilná aplikácia by mala mať jasne definované zameranie. Na rozdiel od desktopových aplikácií, ktoré majú široký záber s množstvom poskytovaných funkcií, by mobilná aplikácia mala byť zameraná na najdôležitejšie funkcie, potrebné na splnenie stanoveného účelu. Zároveň platí, že mobilná aplikácie by na splnenie úlohy mala od používateľa vyžadovať čo najmenej klikov alebo iných gest a mala by odpovedať čo najrýchlejšie.

Počas vývoja mobilnej aplikácie je dôležité držať sa usmernení ohľadom štýlu k mobilnému zariadeniu, na ktorý aplikáciu vyvíjame, a tiež dodržiavať správanie daného zariadenia. Na rozdiel od desktopových aplikácií, ktoré na dosiahnutie úlohy používateľovi ponúkajú viacero možností, by mobilná aplikácia mala používateľovi poskytnúť len jednu cestu, a z tohto dôvodu je

vhodné nasledovať overené zdroje usmernení.

Pri vývoji mobilných aplikácií často nastávajú situácie, kedy zákazník na začiatku nezadá všetky požiadavky na aplikáciu, alebo sa rozhodne už zadefinované požiadavky zmeniť v priebehu vývoja aplikácie. Zákazníci od vývojového tímu očakávajú flexibilitu a schopnosť okamžite riešiť zmeny v požiadavkách, a to zvyčajne v stanovenom termíne. Preto je vhodné, aby bol dizajn mobilnej aplikácie jednoduchý a zákazníkom prinášal vysokú hodnotu, a v neposlednom rade aby spolupráca vo vývojovom tíme bola efektívna.

V priebehu operácií, ktoré trvajú dlhšiu dobu, ako napríklad vstupno-výstupné operácie, prístup do databázy, či sieťové pripojenie, je dôležité zabezpečiť, aby mobilná aplikácia ostala responzívna. Na dosiahnutie tejto responzivity je potrebné zohľadniť oblasť užívateľského rozhrania, prístup do databázy, ako aj funkcionalitu.

Keďže mobilné zariadenia sú na rozdiel od desktopovým počítačov určené jednému používateľovi, mobilné aplikácie by mali byť personalizované. To znamená, že by mali napĺňať individuálne potreby používateľa a správať sa podľa toho, ako si to používateľ vyžaduje. Mobilná aplikácia by teda mala poskytovať individuálny obsah, a takisto by mala mať kontrolu nad dátami, ktoré sa zdieľajú, ukladajú, či používaj pri ďalšie aktivite.

Ďalšia charakteristika, ktorá odlišuje mobilné aplikácie od desktopových, je využívanie senzorov mobilných telefónov na lokalizáciu používateľa. Väčšina dnešných smartfónov disponuje senzormi monitorujúcimi pohyb zariadenia a gestá, majú tiež globálny poziční systém, viaceré kamery, či sieťové protokoly. Mobilné aplikácie by preto mali byť schopné poskytovať informácie založené na polohe, čo zvyšuje ich hodnotu a poskytuje lepší užívateľský zážitok.

Pre kvalitné mobilné aplikácie je na rozdiel od tradičných desktopových aplikácií dôležitá vlastnosť dosiahnuteľnosti kdekoľvek a kedykoľvek. Mobilné zariadenia sa stali našou súčasťou, a spoliehame sa, že nezávisle od našej polohy budú aplikácie na našom zariadení neustále dostupné, plne funkčné a aktualizované.

Čo sa týka aplikačnej bezpečnosti, mobilné aplikácie umožňujú inštalácie rôznych aplikácií, na rozdiel od zabudovaných zariadení, ktoré nie sú otvorené takýmto priamym spôsobom cudziemu softvéru. Tieto nové aplikácie, ktoré máme možnosť do nášho mobilného zariadenia nainštalovať, však predstavujú bezpečnostnú hrozbu, pretože dané aplikácie môžu byť škodlivé a po inštalácií môžu nie len ovplyvniť fungovanie nášho zariadenia, ale aj ukradnúť dáta uložené lokálne na našom zariadení. [24]

Oproti tradičnému softvéru v mobilných aplikáciách väčšina chýb, spojených s aplikačnou bezpečnosťou, nastáva pri správe relácií. Na ochranu pred najčastejšími bezpečnostnými hrozbami je odporúčané dodržiavať nasledujúce zabezpečenia.

- Šifrovanie dát prechádzajúcich cez sieť

- Využitie SSL vrstvy na šifrovanie dát a požadovanie certifikátu zariadenia, čím znemožníme prístup útočníkov k informáciám na mobilnom zariadení
- Doba platnosti aplikačných relácií by mala byť obmedzená, pretože prípadné škodlivé požiadavky na server môžu byť aktívne po celú dobu aplikačnej relácie
- Platnosť požiadaviek na server by mala byť takisto časovo obmedzená, pretože na rozdiel od desktopových aplikácií je tu väčšia hrozba zachytávania a pozmenenia dát do dobu platnosti požiadavky na server
- Opakované požiadavky na server je potrebné zamedziť, aby sme útočníkom zabránili opakovaný prístup napríklad ku geografickej polohe alebo osobným informáciám, pretože štandardne je potrebné tento prístup aplikáciám povoliť jednorazovo a o nasledovných prístupoch už nie sme informovaní

1.1.3 Rozdiely medzi typmi mobilných aplikácií

V dnešnej dobe majú vývojári mobilných aplikácií možnosť výberu z rôznych vývojárskych, ako aj distribučných stratégií [15]. Dodržiavajúc klasifikáciu aplikácií uvedenú v predchádzajúcej kapitole 1.1.1, pri návrhu mobilnej aplikácie máme na výber z nasledujúcich možností.

- Mobilné webové aplikácie
- Natívne aplikácie
- Hybridné aplikácie

Uvedené typy mobilných aplikácií sa vyznačujú určitými charakteristikami a každá z nich má množstvo dobých, ako aj zlých stránok, a teda neexistuje jedno riešenie, ktoré by bolo vhodné pre všetky aplikácie. Pri výbere vhodného riešenia je preto dôležité porozumieť týmto charakteristikám, a na základe toho vedieť rozlišovať medzi jednotlivými typmi aplikácií. V tejto kapitole si preto uvedieme základné rozdiely medzi uvedenými typmi mobilných aplikácií, formulované do kritérií, na základe ktorých by sme sa mali pri výbere typu aplikácie rozhodovať, a to po technickej ako aj netechnickej stránke. V celej nasledujúcej sekcii budem čerpať z [5].

1.1.3.1 Technické kritériá

- Podpora platforiem a verzií

  Je dôležité uvedomiť si, aké mobilné platformy a verzie chceme podporovať, aký rozsah zariadení plánujeme pokryť, ako aj zohľadniť naše

vývojárske schopnosti a znalosti technológií súvisiacich s vývojom mobilných aplikácií. Mobilná webová aplikácia a tiež hybridná sú vhodným riešením v prípade, že chceme podporovať viacero platforiem. Natívna aplikácia naopak nie je v tomto prípade vhodná, keďže aplikácia tohto typu by musela byť vyvíjaná osobitne pre každú platformu.

- Vymoženosti zariadenia

  Tiež by sme si mali premyslieť, aké vymoženosti zariadenia chceme využívať. Ak potrebujeme priamy prístup napríklad k súborovému systému, kamere, čítačke čiarového kódu alebo funkcii Bluetooth, mali by sme zvoliť natívnu alebo hybridnú verziu aplikácie. Mobilná webová aplikácia by v tomto prípade nebola vhodným riešením, keďže aplikácie tohto typu nie sú schopné plného prístupu k daným schopnostiam zariadenia.

- Užívateľský zážitok

  Ak nám záleží na bohatom užívateľskom zážitku, natívna aplikácia by mohla byť správnou voľbou. Natívne aplikácie totiž oproti mobilným webovým aplikáciám poskytujú responzívnejšie rozhranie a kvalitnejšiu interakciu, disponujúc schopnosťou využívania gest špecifických pre dané zariadenie, ako aj krátkym časom spustenia. Strednou cestou sú hybridné aplikácie, ktoré majú prístup k natívnym API prostredníctvom HTML kódu, avšak hybridnou aplikáciou nedokážeme používateľom poskytnúť takú úroveň užívateľského zážitku, akú vieme dosiahnuť natívnou aplikáciou.

- Výkonnosť

  Natívna aplikácia je vhodná aj prípade, ak chceme realizovať graficky náročné používateľské rozhranie, alebo také, ktoré bude vyžadovať nadmerné spracovanie údajov. Mobilné webové ani hybridné aplikácie by nemuseli byť dostatočne výkonné, pretože oproti natívnym aplikáciám pracujú nad vrstvami, ktoré ich oberajú o výpočtové zdroje. Nech sa už rozhodneme pre akékoľvek riešenie, táto výkonnosť by mala byť v procese vývoja aplikácie otestovaná.

- Aktualizácie

  Pri výbere typu aplikácie je tiež dôležité zohľadniť potrebu budúcej aktualizácie našej aplikácie, pretože v tomto smere je priateľskejším riešením voľba mobilnej webovej aplikácie. Ak by sme si totiž zvolili natívnu aplikáciu, museli by sme podporovať niekoľko verzií súčasne, čo by viedlo k zvýšeniu zložitosti vývoja aplikácie. Čo sa týka hybridných aplikácií, tak s implementáciou v lokálnej časti aplikácie sa dostaneme do rovnakých problémov ako pri natívnych aplikáciách.

1.1.3.2 Netechnické kritériá

- Schvaľovací proces

  Ak si zvolíme riešenie natívnou alebo hybridnou aplikáciou, musíme rátať so schvaľovacím procesom. Ten však môže narušiť agilný vývoj aplikácie, keďže tento spôsob vývoja vyžaduje rýchle iterácie, ktoré v sebe zahŕňajú aj spätnú väzbu používateľov.

- Distribúcia

  Najlepší spôsob, ako zaistiť jednoduché nájdenie aplikácie používateľmi, je využitie obchodov s aplikáciami (takzvané app stores). Z tohto dôvodu je distribúcia jednoduchšia u natívnych a hybridných aplikácií. Ani tento spôsob distribúcie však nie je ideálny, keďže počet aplikácií v spomínaných obchodoch stále rastie.

- Monetizácia

  Ak sa rozhodneme pre natívnu alebo hybridnú aplikáciu využívajúcu obchody s aplikáciami, prinesie nám to výhodu aj v súvislosti s monetizáciou. Jednoduchá a presne definovaná platobná brána nám totiž môže zaistiť dobrý výmenný kurz. Má to však aj nevýhodu, keďže majiteľ platformy si ponecháva značnú časť platby.

1.1.3.3 Zhrnutie

Nasledujúca tabuľka 1.1 sumarizuje všetky zmienené technické a netechnické kritéria vhodné pri výbere typu mobilnej aplikácie, spolu s hodnotením jednotlivých typov mobilných aplikácií v zmysle týchto kritérií.

1.2 Analýza požiadaviek

V dobe, keď sú ľudia zvyknutí používať informačné technológie v takmer všetkých oblastiach života, sú nároky na dobrú použiteľnosť aplikácií nastavené vysoko. Preto ak chceme, aby bola naša aplikácia užitočná a napĺňala náročné očakávania našich budúcich používateľov, je veľmi dôležité urobiť dôkladnú analýzu požiadaviek. V tejto časti práce sa preto venujem prieskumu s potenciálnymi používateľmi, prostredníctvom ktorého sa pokúsim zistiť, aké sú ich požiadavky na funkcie poskytované aplikáciou. Výstupom tohto prieskumu budú takzvané hrubé používateľské požiadavky. Najprv si však uvedieme, aké skupiny požiadaviek rozlišujeme.

1.2.1 Skupiny požiadaviek

Funkčné požiadavky

Kritériá	**Natívne aplikácie**	**Hybridné aplikácie**	**Webové aplikácie**
Úsilie potrebné na podporu platforiem a verzií	Vysoké	Stredné	Nízke
Prístup k vymoženostiam zariadenia	Plný	Plný	Čiastočný
Užívateľský zážitok	Výborný	Výborný	Dobrý
Výkonnosť	Veľmi vysoká	Veľmi vysoká	Vysoká
Aktualizácie pre používateľov	Potrebné	Potrebné	Nepotrebné
Schvaľovací proces	Povinný	Niekedy	Nepovinný
Jednoduchosť distribúcie	Stredná	Stredná	Vysoká
Monetizácia v obchodoch s aplikáciami	Dostupná	Dostupná	Nedostupná

Tabuľka 1.1: Zhrnutie kritérií výberu typu mobilnej aplikácie. Prevzaté z [5].

Funkčné požiadavky popisujú, aké funkcie by mal budúci systém poskytovať, alebo aké sú podmienky jeho fungovania. Funkčné požiadavky sú definované zadávateľom. [26]

Nefunkčné požiadavky

Ďalším druhom sú nefunkčné požiadavky, ktoré majú za cieľ oboznámiť návrhárov systému s aplikačnou doménou a tiež s podmienkami, ktoré s touto doménou súvisia. Príkladom nefunkčných požiadaviek sú spoľahlivosť, bezpečnosť, alebo výkonnosť. [26]

Systémové charakteristiky

Treťou skupinou sú systémové charakteristiky, ktoré sa zvyčajne vytvárajú opačným spôsobom. To znamená, že popisujú charakteristiky systému, ktoré neboli definované zadávateľom, a teda nie sú žiadúce. [26]

Požiadavky na obmedzenia

Posledným druhom sú požiadavky na obmedzenia, ktoré sa používajú pre stanovené limity k alternatívam návrhu systému. Tieto požiadavky na obmedzenia musia byť platné nezávisle na spôsobe riešenia problému. [26]

Nezávisle od toho, akej skupiny požiadavky patria, platí, že všetky požiadavky by mali byť atomické, teda dostatočne konkrétne, a tiež verifikovateľné. Ďalej by mali byť úplné a konzistentné, aby nedošlo k rozporu medzi požiadavkami. [26]

1.2.2 Metódy zberu požiadaviek

V tejto časti sa pozrieme na najpoužívanejšie metódy zberu požiadaviek, a následne určíme, ktoré z týchto metód sú v našom prípade najvhodnejšie.

Rozhovor

Často používanou metódou je rozhovor [27], ktorý vedieme so zadávateľom alebo koncovými používateľmi. Tento intuitívny spôsob zberu požiadaviek, ktorý má pôvod v sociálnych vedách, však nie je taký jednoduchý ako by sa mohlo zdať. Poznáme tri typy rozhovoru, a to štruktúrovaný, pološtruktúrovaný a neštruktúrovaný. Neštruktúrovaný rozhovor je najprínosnejší, a práve pri tomto spôsobe zberu požiadaviek je dôležité, aby bol vedený skúseným analytikom, schopným viesť rozhovor správnou cestou. Pri štruktúrovanom rozhovore sa môže stať, že sa nevenujeme určitým témam, ktoré sa môžu zdať nepodstatné, avšak môže to viesť k vynechaniu požiadaviek. Veľmi prínosným spôsobom je aj pološtruktúrovaný rozhovor, pri ktorom máme vopred pripravenú len základnú sadu otázok. [26]

Brainstorming

Rozhovor, najmä jeho pološtruktúrovaný variant, sa častokrát dopĺňa metódou zvanou brainstorming [27]. Ak máme viacero zadávateľov, otázky sú položené všetkým naraz, a táto spoločná diskusia môže byť pre zber požiadaviek veľmi obohacujúca. Počas brainstormingu platí zásada, že žiaden príspevok diskusie nezahadzujeme. Všetky odpovede, nápady a diskusia participantov sú zaznamenané. [26]

Laddering

Počas brainstormingu môže byť použitý laddering [27]. Táto technika pomáha viesť diskusiu a prináša hierarchickú štruktúru odpovedí, ktoré sa postupne upresňujú, čím získavame ďalšie informácie. [26]

Dotazníky

Pravdepodobne najdôležitejšou neosobnou metódou zberu požiadaviek sú dotazníky. Táto metóda sa využíva najmä v predbežnej fáze zberu [27] a zvykne byť nasledovaná nejakou osobnou metódou. Slabou stránkou dotazníkov je, že neodhaľujú nové informácie o aplikačnej doméne. Je preto dôležité, aby dotazník vytváral skúsený analytik, s dostatočnou znalosťou danej domény. [26]

Analýza úloh

Ďalšou metódou je analýza úloh, ktorá spočíva v dekompozícii úloh [27, 28]. To znamená, že sa najprv navrhnú úlohy najvyššej úrovne a z nich sa následne odvodia podúlohy. Týmto spôsobom získame ako používateľské, tak aj systémové požiadavky. Každá takáto požiadavka je následne zdokumentovaná formou viacerých používateľských, prípadne systémových scenárov.[26]

Pozorovanie

Pozorovanie nám umožňuje pozorovať budúcich používateľov v ich vlastnom prostredí, ktorým je problémová doména. Výsledky pozorovania sa líšia podľa toho, či používateľ vie o tom, že je sledovaný. Ak o tom totiž vie, má tendenciu meniť svoje správanie. Pozorovanie je však najhodnotnejšie, ak sa používateľ správa prirodzene. To sa dá docieliť napríklad použitím skrytých kamier, avšak s ohľadom na súkromie. [26]

Prototypovanie

Posledná metóda zberu požiadaviek, ktorú si spomenieme, je prototypovanie. Táto metóda pomáha vytvoriť detailné požiadavky, avšak až v neskoršej fáze zberu, keď už máme základné požiadavky identifikované. [26]

1.2.3 Formulácia požiadaviek

Pri zbere požiadaviek tejto aplikácie som sa rozhodla aplikovať metódu pološtruktúrovaného rozhovoru, ktorý bol uskutočnený čiastočne aj prostredníctvom emailovej komunikácie.

1.2.3.1 Funkčné požiadavky

- FP1. Správa účtu používateľa
  - Popis:
    - Prihlásenie pomocou mena a hesla
    - Odhlásenie
  - Priorita: Vysoká
  - Typ: Funkčná požiadavka
  - Zložitosť: Nízka
- FP2. Evidencia reklamných kampaní
  - Popis:
    - Zoznam všetkých reklamných kampaní
    - Zoznam reklamných plôch prislúchajúcich vybranej reklamnej kampani
    - Možnosť označenia zvolenej reklamnej kampane ako neukončená alebo ukončená
  - Priorita: Vysoká
  - Typ: Funkčná požiadavka
  - Zložitosť: Stredná
- FP3. Evidencia reklamných plôch
  - Popis:
    - Zoznam všetkých reklamných plôch
    - Možnosť filtrácie zoznamu reklamných plôch minimálne podľa nasledujúcich parametrov
      - Názov
      - Geografická poloha
      - Stav reklamnej plochy
    - Možnosť vyhľadávania v zozname reklamných plôch na základe vstupného textu
    - Detail reklamnej plochy
    - Zoznam reklamných nosičov prislúchajúcich vybranej reklamnej ploche
    - Pridanie poznámky k zvolenej reklamnej ploche
  - Priorita: Vysoká
  - Typ: Funkčná požiadavka

  - Zložitosť: Vysoká

- FP4. Evidencia reklamných nosičov
  - Popis:
    * Zoznam reklamných nosičov prislúchajúcich vybranej reklamnej ploche
    * Možnosť vyhľadávania v zozname reklamných nosičov na základe vstupného textu
    * Galéria fotografií konkrétneho reklamného nosiča
    * Pridanie fotografie k zvolenému reklamnému nosiču, a to prostredníctvom aplikácie Fotoaparát daného zariadenia
    * Vymazanie fotografie vybraného reklamného nosiča z galérie fotografií
  - Priorita: Vysoká
  - Typ: Funkčná požiadavka
  - Zložitosť: Vysoká

- FP5. Možnosť zapnutia automatickej synchronizácie iba cez WiFi
  - Popis: Aplikácia bude automaticky synchronizovať dáta iba v prípade, keď je zariadenie pripojené prostredníctvom WiFi
  - Priorita: Nízka
  - Typ: Funkčná požiadavka
  - Zložitosť: Nízka

1.2.3.2 Nefunkčné požiadavky

- NP1. Cieľová platforma
  - Natívna Android aplikácia
  - Dodržiavanie štandardov operačného systému Android
  - Podpora pre operačný systém Android verzie 4.4 a vyššie

- NP2. Režim s internetovým pripojením
  - Na využívanie všetkých funkcií aplikácie je potrebné internetové pripojenie
  - Možnosť nastavenia automatickej synchronizácie iba v prípade, keď je zariadenie pripojené prostredníctvom WiFi

- NP3. Režim bez internetového pripojenia

  - Aplikácia musí byť schopná vykonávať funkcie, ktoré nevyžadujú komunikáciu so serverom, aj v režime bez internetového pripojenia
  - Aplikácia má zaistiť konzistenciu dát pri zmene režimov s a bez internetového pripojenia

- NP4. Prístup k aplikácii Fotoaparát
  - Aplikácia má požadovať prístup k aplikácii Fotoaparát daného zariadenia

- NP5. Orientácia
  - Aplikácia má byť primárne orientovaná na výšku
  - Orientácia na šírku nemusí byť aplikáciou podporovaná

- NP6. Jazyk
  - Aplikácia má byť iba v českom jazyku

- NP7. Bezpečnosť
  - Aplikácia bude využívať autentizáciu používateľov

- NP8. Rozšíriteľnosť
  - Aplikácia má byť implementovaná takým spôsobom, aby umožnila jednoduché pridanie funkcionality, a to bez zásahu do už implementovanej funkcionality

1.3 Prípady použitia

Prípad použitia opisujú interakciu používateľa so systémom v jednotlivých krokoch, a to za účelom dosiahnutia konkrétneho cieľa. Je to metodológia aplikovaná v rannom štádiu vývoja softvéru, slúžiaca na identifikáciu a organizáciu funkčných požiadaviek. Táto technika pomáha zvládať komplexné projekty dekompozíciou na používateľské funkcie. Zároveň je to však prostriedok vhodný na komunikáciu so zákazníkmi, a to vďaka svojej jednoduchosti. [29]

Táto metodológia bola prvýkrát formulovaná v roku 1986 švédskym počítačovým vedcom Ivar Hjalmar Jacobsonom [30]. Ten sa tiež významne podieľal na vývoji jazyka UML, pričom prípady použitia znázorňujeme práve pomocou UML diagramu. Diagram prípadov použitia obsahuje tieto hlavné komponenty [29].

- Aktéri

  Aktér, komponent reprezentujúci osobu alebo externý systém, je nejakým spôsobom prepojený s modelovaným systémom.

- Prípady použitia

  Prípad použitia opisuje interakciu aktéra s modelovaným systémom, a to za účelom dosiahnutia konkrétneho cieľa. Prípady použitia sú zvyčajne vyvolané aktérmi.

- Vzťahy

  Diagram prípadov použitia tiež znázorňuje vzťahy, ktoré môžu byť nasledujúcich typov [31].

  - Vzťahy medzi aktérmi
    Medzi aktérmi môže existovať vzťah generalizácia, ktorý označuje zdedenie roly jedného aktéra druhým aktérom.
  - Vzťahy medzi aktérmi a prípadmi použitia
    Tento vzťah musí byť prítomný v každom diagrame prípadov použitia. Platí, že jeden aktér musí byť asociovaný s aspoň jedným alebo viacerými prípadmi použitia, a zároveň viacero aktérov môže byť asociovaných s jedným prípadom použitia.
  - Vzťahy medzi prípadmi použitia
    * Zahrnutie
      Vzťah zahrnutie môžeme použiť medzi dvoma prípadmi použitia, a označujeme ho ako «Include». Používame ho v prípade, že funkcionalita jedného prípadu použitia je súčasťou druhého prípadu použitia. Tento vzťah je užitočný na znovupoužitie určitej funkcionality, ktorú zdieľajú viaceré prípady použitia v danom diagrame.
    * Rozšírenie
      Tento vzťah dvoch prípadov použitia, označovaný ako «Extend» , rozširuje základný prípad použitia, a to pridaním funkcionality, ktorá je reprezentovaná druhým prípadom použitia.
    * Generalizácia
      Vzťah generalizácie používame, ak chceme označiť zdedenie funkcionality jedným prípadom použitia jeho potomkom.

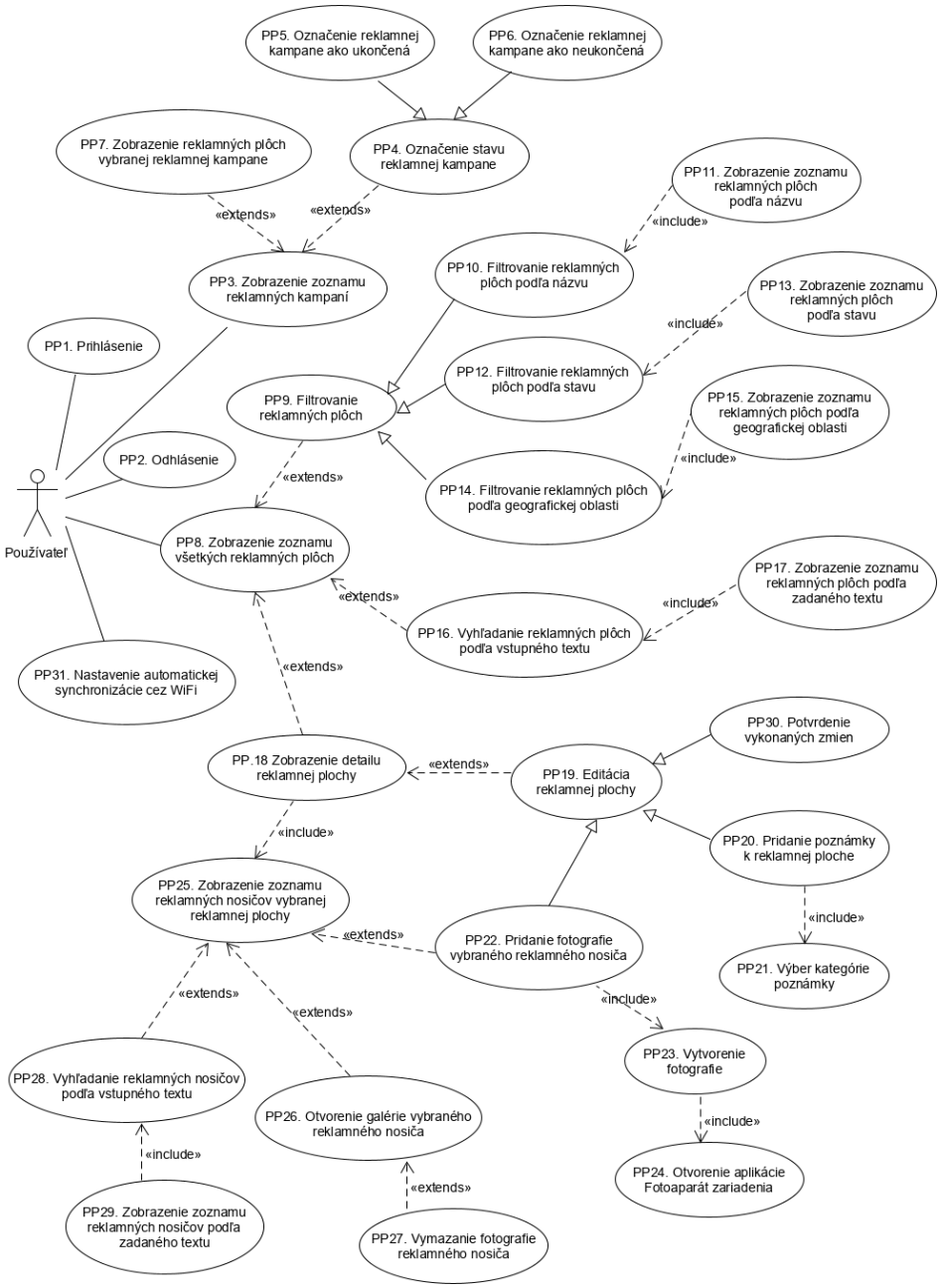
1.3.1 Identifikácia prípadov použitia

Diagram prípadov použitia 1.1 znázorňuje všetky prípady použitia identifikované v našom systéme.

Ďalej si tiež uvedieme popis týchto prípadov použitia. U niektorých komplikovanejších si uvedieme aj jednotlivé kroky, popisujúce interakciu medzi aktérom a aplikáciou, potrebné na dosiahnutie daného cieľa.

- PP1. Prihlásenie

  Aplikácia umožní prihlásenie používateľa pomocou mena a hesla.



Obr. 1.1: Diagram prípadov použitia.

- PP2. Odhlásenie

  Aplikácia umožní odhlásenie prihláseného používateľa.

- PP3. Zobrazenie zoznamu reklamných kampaní

Aplikácia umožní zobraziť zoznam všetkých reklamných kampaní spolu s informáciou o ich ID a období, ako aj s označením stavu každej reklamnej kampane.

- PP4. Označenie stavu reklamnej kampane

  Po zobrazení zoznamu všetkých reklamných kampaní má používateľ možnosť označiť stav akejkoľvek reklamnej kampane.

- PP5. Označenie reklamnej kampane ako ukončená

  Používateľ má možnosť vybranú reklamnú kampaň označiť ako ukončenú.

- PP6. Označenie reklamnej kampane ako neukončená

  Používateľ má možnosť vybranú reklamnú kampaň označiť ako neukončenú.

- PP7. Zobrazenie reklamných plôch vybranej reklamnej kampane

  Po zobrazení zoznamu všetkých reklamných kampaní má používateľ možnosť zobraziť všetky reklamné plochy akejkoľvek vybranej kampane.

- PP8. Zobrazenie zoznamu všetkých reklamných plôch

  Aplikácia umožní zobraziť zoznam všetkých reklamných plôch spolu s informáciou o ich kóde, adrese a celkového počtu fotografií, ako aj s označením stavu každej reklamnej plochy.

- PP9. Filtrovanie reklamných plôch

  Po zobrazení zoznamu reklamných plôch má používateľ možnosť filtrovania tohto zoznamu podľa zvolených parametrov.

  Základná cesta:

  1. Prípad použitia začína, keď sa používateľ rozhodne vyhľadať reklamné plochy podľa určitých parametrov.
     a) Používateľ sa rozhodne vyhľadať reklamné plochy na základe názvu.
     b) Používateľ sa rozhodne vyhľadať reklamné plochy na základe geografickej oblasti.
     c) Používateľ sa rozhodne vyhľadať reklamné plochy na základe stavu.
  2. Aplikácia ponúkne parametre, podľa ktorých je možné filtrovať reklamné plochy a tiež zoznam hodnôt, ktoré môžu dané parametre nadobúdať.
     a) Používateľ si zvolí hodnotu názvu reklamnej plochy.
     b) Používateľ si zvolí hodnotu geografickej oblasti reklamnej plochy.

c) Používateľ si zvolí hodnotu stavu reklamnej plochy.

3. Aplikácia vyhľadá a zobrazí reklamné plochy, ktoré spĺňajú požadované kritériá.

- PP10. Filtrovanie reklamných plôch podľa názvu

  Používateľ má možnosť filtrovať zoznam reklamných plôch na základe zvoleného názvu zo zoznamu názvov, ktoré mu ponúkne aplikácia.

- PP11. Zobrazenie zoznamu reklamných plôch podľa názvu

  Aplikácia umožní zobrazenie zoznamu reklamných plôch podľa zvoleného názvu.

- PP12. Filtrovanie reklamných plôch podľa stavu

  Používateľ má možnosť filtrovať zoznam reklamných plôch na základe zvoleného stavu zo zoznamu stavov, ktoré mu ponúkne aplikácia.

- PP13. Zobrazenie zoznamu reklamných plôch podľa stavu

  Aplikácia umožní zobrazenie zoznamu reklamných plôch podľa zvoleného stavu.

- PP14. Filtrovanie reklamných plôch podľa geografickej oblasti

  Používateľ má možnosť filtrovať zoznam reklamných plôch na základe zvolenej geografickej oblasti zo zoznamu oblastí, ktoré mu ponúkne aplikácia.

- PP15. Zobrazenie zoznamu reklamných plôch podľa geografickej oblasti

  Aplikácia umožní zobrazenie zoznamu reklamných plôch podľa zvolenej geografickej oblasti.

- PP16. Vyhľadanie reklamných plôch podľa zadaného textu

  Po zobrazení zoznamu reklamných plôch má používateľ možnosť vyhľadania reklamných plôch podľa zadaného textu.

  Základná cesta:

  1. Prípad použitia začína, keď sa používateľ rozhodne vyhľadať reklamné plochy na základe zadaného textu.
  2. Aplikácia ponúkne zadanie vstupného textu.
  3. Používateľ zadá text.
  4. Aplikácia vyhľadá a zobrazí reklamné plochy, u ktorých nájde zhodu so zadaným textom.

- PP17. Zobrazenie zoznamu reklamných plôch podľa zadaného textu

  Aplikácia umožní zobrazenie zoznamu reklamných plôch podľa zadaného textu.

- PP18. Zobrazenie detailu reklamnej plochy

  Aplikácia umožní zobrazenie detailu vybranej reklamnej plochy.

- PP19. Editácia reklamnej plochy

  Po zobrazení detailu vybranej pracovnej plochy má používateľ možnosť vytvárania zmien tejto plochy.

- PP20. Pridanie poznámky k reklamnej ploche

  Používateľ má možnosť editácie vybranej reklamnej plochy pridaním poznámky k tejto ploche.

  Základná cesta:

  1. Prípad použitia začína, keď sa používateľ rozhodne pridať poznámku k zvolenej reklamnej ploche.
  2. Aplikácia v rámci detailu reklamnej plochy ponúkne pridanie poznámky v rôznych kategóriách, ako napríklad:
     - Inštalácia povolená
     - Predajňa trvalo uzavretá
     - Predajňa dočasne uzavretá
     - Málo produktov
     - Iné
  3. Používateľ si zvolí typ poznámky.
  4. Aplikácia poskytne možnosť pridania textu k zvolenej poznámke.
     a) Používateľ zadá text poznámky.
     b) Používateľ nechce špecifikovať poznámku a preskočí tento krok.
  5. Používateľ potvrdí pridanie poznámky odoslaním zmien v detaile reklamnej plochy.

- PP21. Výber kategórie poznámky

  Výber kategórie poznámky reklamnej plochy je pre používateľa nutným krokom k pridaniu poznámky.

- PP22. Pridanie fotografie vybraného reklamného nosiča

  Používateľ má možnosť editácie reklamnej plochy pridaním fotografie vybraného reklamného nosiča. Tento krok je možné vykonať po zobrazení zoznamu reklamných nosičov určitej reklamnej plochy.

  Základná cesta:

1. Prípad použitia začína, keď sa používateľ rozhodne pridať fotografiu k zvolenému reklamnému nosiču.
2. Aplikácia otvorí aplikáciu Fotoaparát daného zariadenia.
3. Používateľ v aplikácií Fotoaparát vytvorí fotografiu.
4. Aplikácia ukončí aplikáciu Fotoaparát.
5. Používateľ potvrdí pridanie fotografie nosiča odoslaním zmien v detaile reklamnej plochy.

- PP23. Vytvorenie fotografie

  Používateľ má možnosť vytvorenia fotografie reklamného nosiča prostredníctvom aplikácie Fotoaparát daného zariadenia. Tento krok je potrebný pre pridanie fotografie vybraného reklamného nosiča.

- PP24. Otvorenie aplikácie Fotoaparát zariadenia

  Aplikácia umožní otvorenie aplikácie Fotoaparát daného zariadenia. Je to zároveň nutným krokom pre vytvorenie fotografie reklamného nosiča.

- PP25. Zobrazenie zoznamu reklamných nosičov vybranej reklamnej plochy

  Súčasťou zobrazenia detailu reklamnej plochy je zobrazenie zoznamu reklamných nosičov vybranej reklamnej plochy.

- PP26. Otvorenie galérie vybraného reklamného nosiča

  Po zobrazení zoznamu reklamných nosičov zvolenej reklamnej plochy má používateľ možnosť otvoriť galériu fotografií zvoleného reklamného nosiča.

- PP27. Vymazanie fotografie reklamného nosiča

  Po otvorení galérie fotografií reklamného nosiča má používateľ možnosť vymazať zvolenú fotografiu.

- PP28. Vyhľadanie reklamných nosičov podľa zadaného textu

  Po zobrazení zoznamu reklamných nosičov zvolenej reklamnej plochy má používateľ možnosť vyhľadať reklamné nosiče podľa zadaného textu. Základná cesta tohto používateľského prípadu prebieha podobne ako PP16. Vyhľadanie reklamných plôch podľa zadaného textu.

- PP29. Zobrazenie zoznamu reklamných nosičov podľa zadaného textu

  Aplikácia umožní zobrazenie zoznamu reklamných nosičov, pri ktorých sa vyskytuje zhoda s používateľom zadaným textom.

- PP30. Potvrdenie vykonaných zmien

  Používateľ má možnosť potvrdiť vykonané zmeny vybranej reklamnej plochy.

- PP31. Nastavenie automatickej synchronizácie cez WiFi

  Používateľ má možnosť zapnúť automatickú synchronizáciu dát iba v prípade, keď bude zariadenie pripojené prostredníctvom WiFi.

1.3.2 Pokrytie funkčných požiadaviek

Tabuľka 1.2 zobrazuje mapovanie prípadov použitia na funkčné požiadavky aplikácie.

1.4 Aplikačná platforma

Cieľom tejto kapitoly je výber aplikačnej platformy a programovacieho jazyka. Na úvod si zanalyzujeme trhový podiel mobilných platforiem a porovnáme tie najúspešnejšie, ktorými sú Android a iOS. Na základe toho následne vyberieme mobilnú platformu vhodnú pre našu aplikáciu, ako aj minimálnu verziu, ktorú budeme podporovať. Ďalej si porovnáme Javu a Kotlin, najpoužívanejšie programovacie jazyky pre vývoj Android aplikácií, a na základe toho vyberieme jazyk vhodný pre vývoj našej aplikácie.

1.4.1 Trhový podiel mobilných platforiem

Nasledujúci graf 1.2 zobrazuje celosvetový trhový podiel mobilných operačných systémov za rok 2020. Oranžový stĺpec reprezentuje platformu Android a šedý platformu iOS.

Najúspešnejšie technologické platformy sú Android, poskytovaný spoločnosťou Google, so 73.06% podielom na trhu a iOS od spoločnosti Apple s 26.28% podielom. Zvyšné operačné systémy sú zanedbateľné, tvoriac dohromady menej ako 1% podielu na trhu. Medzi tieto firmy patrí napríklad aj Microsoft, ktorý už uznal, že plán predaja smartfónov na ich platforme skončil neúspechom [32]. Amy Hood, finančná riaditeľka spoločnosti Microsoft, informovala: „V tomto štvrťroku sme nemali žiadny významný príjem z predaja mobilných telefónov“ [32]. Takisto aj Canonical, spoločnosť poskytujúca operačný systém Ubuntu Linux, sa vzdala vývoja svojho mobilného operačného systému [33]. Čo sa týka BlackBerry, pretrváva už iba ako značka [34], a súčasné mobilné telefóny pod touto značkou fungujú na platforme Android [35].

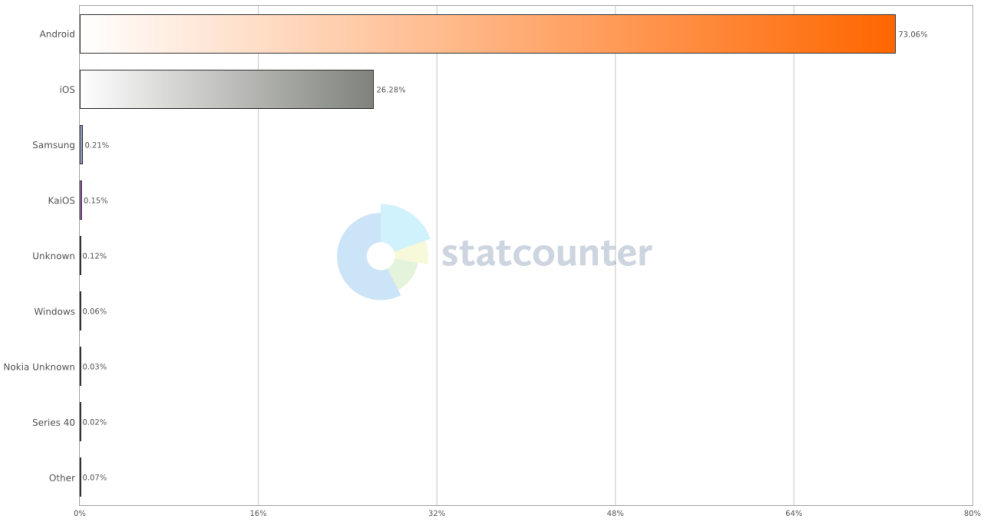
Graf 1.3 zobrazuje vývoj celosvetového trhového podielu mobilných operačných systémov od mája 2012 do mája 2021, meraný v mesačných intervaloch. Vývoj platformy Android je znázornený oranžovou farbou a šedá farba reprezentuje operačný systém iOS.

	FP1	**FP2**	**FP3**	**FP4**	**FP5**
PP1	✓				
PP2	✓				
PP3		✓			
PP4		✓			
PP5		✓			
PP6		✓			
PP7		✓	✓		
PP8			✓		
PP9			✓		
PP10			✓		
PP11			✓		
PP12			✓		
PP13			✓		
PP14			✓		
PP15			✓		
PP16			✓		
PP17			✓		
PP18			✓	✓	
PP19			✓	✓	
PP20			✓		
PP21			✓		
PP22			✓	✓	
PP23				✓	
PP24				✓	
PP25			✓	✓	
PP26				✓	
PP27				✓	
PP28				✓	
PP29				✓	
PP30			✓	✓	
PP31					✓

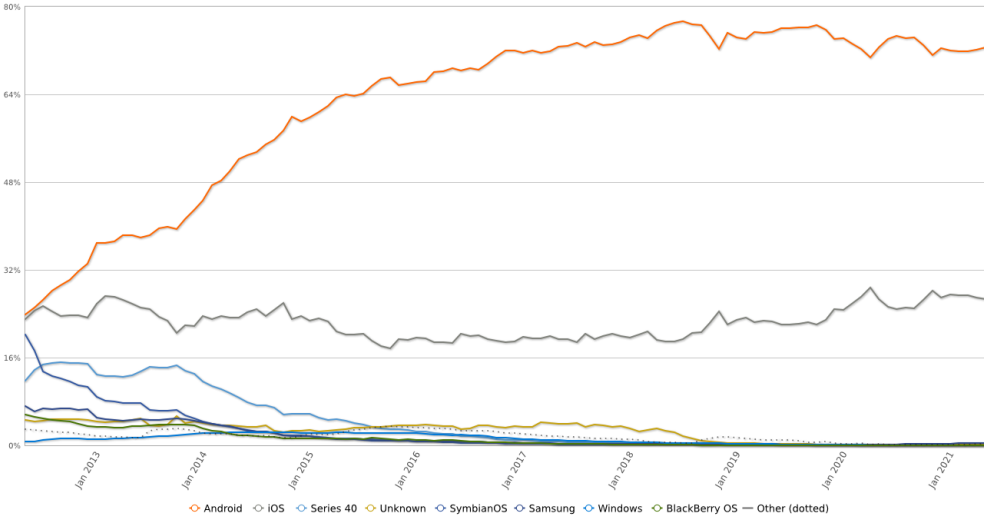
Tabuľka 1.2: Pokrytie funkčných požiadaviek.

1.4.2 Porovnanie platforiem Android a iOS

Ak si teda vyberieme naozaj kvalitný smartfón, môžeme si byť istí, že bude postavený na platforme Android alebo iOS. Obe tieto operačné systémy napredujú veľmi rýchlo, poskytujúc stále viac a viac pokrokovejšiu funkcionalitu. Sú vyspelé, a je len málo funkcií, ktoré sú výnimočné iba pre jeden OS [36].



Obr. 1.2: Celosvetový trhový podiel mobilných operačných systémov za rok 2020. Prevzaté z [1].



Obr. 1.3: Celosvetový trhový podiel mobilných operačných systémov od mája 2012 do mája 2021. Prevzaté z [2].

Obe však majú svoje silné aj slabé stránky, ktoré si porovnáme v nasledujúcich kategóriách.

- Jednoduchosť použitia

  Keď Steve Jobs 9.1.2007 v San Franciscu predstavil svetu prvý iPhone, bol to v tom čase skvost v jednoduchosti použitia. Dnes už ale nevyniká tak ako kedysi. Android dozrel, a jeho rozhranie je veľmi podobné

rozhraniu iOS. V niečom je lepšie, v inom zas horšie. Veľkou výhodou Androidu oproti iOS čo sa týka rozhrania je flexibilita. Android totiž poskytuje väčšiu kontrolu nad systémom a aplikáciami, čím sa mu podarilo udržať vernosť používateľov k tejto platforme [37]. [38]

- Cena a dizajn

  Zatiaľ čo iPhone má konzistentný dizajn naprieč verziami, vzhľad Android smartfónov sa veľmi líši. Apple má totiž pod kontrolou celý výrobný proces, čo im umožňuje produkovať len smartfóny s prvotriednym dizajnom. To však ale znamená, že iPhone nižšej cenovej kategórie nenájdeme. Na druhej strane, cenové kategórie Android mobilných telefónov sa vďaka ich variabilite líšia, pohybujúc sa medzi 100 € až 1 000 €. To nám dáva možnosť zaplatiť len za to, čo od mobilného zariadenia naozaj požadujeme. [38]

- Uzavretý a otvorený systém

  Android je oproti iOS open source a zároveň otvorený alternatívnym aplikáciám. Na druhej strane, platforma iOS je uzavretý systém, čo platí aj smerom von, a teda Android smartfóny nedisponujú aplikáciami od spoločnosti Apple. [38]

- Hlasový asistent

  Prvý hlasový asistent s názvom Siri predstavila spoločnosť Apple. Ostal však na relatívne základnej úrovni a nechal sa predbehnúť Google Asistentom. Ten dokáže používateľom zjednodušiť život efektívnym využitím aplikácií ako Google Mapy či Google Kalendár. Google Asistent je ale dostupný aj na platforme iOS. [38]

- Aktualizácie

  Čo sa týka vydania aktualizácií, iOS je na tom výrazne lepšie, a to vďaka tomu, že Apple má plnú kontrolu nad svojimi mobilnými zariadeniami. Ak teda vlastníme podporovaný iPhone, môžeme sa spoľahnúť, že je plne aktualizovaný. Tiež je ale pravdou, že množstvo týchto aktualizácií malo za následok WiFi problém [39].

  Android, naopak, necháva aktualizácie na zodpovednosti výrobcov smartfónov. Používateľom Androidu sa teda môže stať, že najnovšie bezpečnostné aktualizácie sa k nim ani nedostanú [40]. S takto zastaralou bezpečnosťou fungujú takmer tri štvrtiny Android zariadení [40]. [38]

- Bezpečnosť

  Android nemá na rozdiel od iOS až také striktné pravidlá na vpustenie aplikácií do svojej platformy prostredníctvom obchodu s aplikáciami Google Play. Spoločnosť Apple teda z tohto hľadiska zabezpečuje vyššiu bezpečnosť, avšak ani používatelia iPhone nie sú stopercentne chránený

pre vírusom [41]. Android je na tom s bezpečnosťou tiež celkom dobre. Google oznámil, že v Google Play je len 0.16% škodlivého softvéru [42]. [38]

- Periférie

  Android ma výhodu jednoduchého zapojenia iných zariadení, poskytujúc štandardný USB port. IPhone má takzvaný Lightning konektor, ktorý vďaka svojej jedinečnosti komplikuje pripojenie iných zariadení, a navyše je drahší. [38]

1.4.3 Výber aplikačnej platformy

Táto aplikácia bude postavená na platforme Android, a to z dôvodu, že to bola jedna z nefunkčných požiadaviek zadávajúcej spoločnosti. Táto požiadavka vyplynula z faktu, že technický pracovníci, ktorí sú zároveň hlavnými budúcimi používateľmi aplikácie, disponujú mobilnými telefónmi nižšej cenovej kategórie. Takmer vo všetkých prípadoch sú to práve smartfóny s operačným systémom Android.

1.4.4 Minimálna podporovaná verzia

Nasledujúca tabuľka 1.3 zobrazuje názvy jednotlivých verzií operačného systému Android, spolu s API levelom a dátumom prvého vydania.

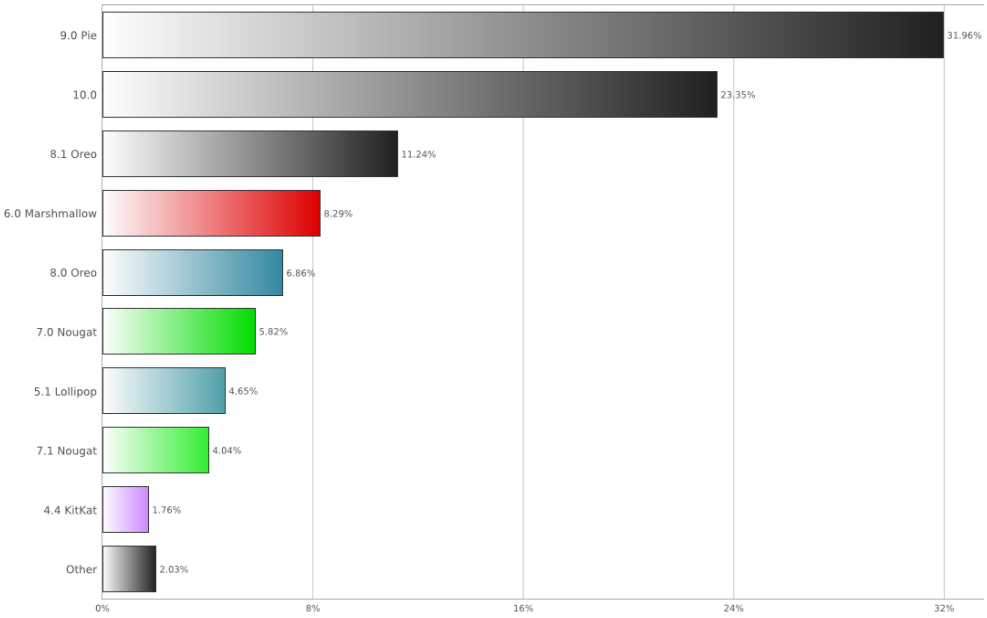
Názov	**Verzia**	**API level**	**Dátum prvého vydania**
(bez názvu)	1.0	1	23. 9. 2008
	1.1	2	9. 2. 2009
Cupcake	1.5	3	27. 4. 2009
Donut	1.6	4	15. 9. 2009
Eclair	2.0	5	26. 10. 2009
	2.0.1	6	
	2.1	7	
Froyo	2.2.x	8	20. 5. 2010
Gingerbread	2.3 - 2.3.2	9	6. 12. 2010
	2.3.3 - 2.3.7	10	

Honeycomb	3.0	11	22. 2. 2011
	3.1	12	
	3.2.x	13	
Ice Cream Sandwich	4.0.1 - 4.0.2	14	18. 10. 2011
	4.0.3 - 4.0.4	15	
Jelly Bean	4.1.x	16	9. 7. 2012
	4.2.x	17	
	4.3.x	18	
KitKat	4.4 - 4.4.4	19	31. 10. 2013
Lollipop	5.0	21	12. 11. 2014
	5.1	22	
Marshmallow	6.0	23	5. 10. 2015
Nougat	7.0	24	22. 8. 2016
	7.1	25	4. 10. 2016
Oreo	8.0	26	21. 8. 2017
	8.1	27	5. 12. 2017
Pie	9	28	6. 8. 2018
Android10	10	29	3. 9. 2019
Android11	11	30	8. 9. 2020
Android12	12	31	18. 5. 2021

Tabuľka 1.3: Verzie operačného systému Android. Prevzaté z [6] [7].

Pri návrhu aplikácie je správne využívať vlastnosti najnovších verzií, avšak tiež je potrebné podporovať staršie verzie. Odporúča sa podporovať aspoň 90% aktívnych zariadení [43]. Graf 1.4 zobrazuje celosvetový trhový podiel Android verzií mobilných zariadení a tabletov za rok 2020.

Z grafu môžeme vidieť, že v uplynulom roku tvorili Android verzie nižšie ako 4.4 KitKat celkovo len 2.03% celosvetového trhového podielu. To znamená,



Obr. 1.4: Celosvetový trhový podiel Android verzií mobilných zariadení a tabletov za rok 2020. Prevzaté z [3].

že podpora od verzie KitKat by pokryla 97.97% aktívnych zariadení, čo je viac ako odporúčaný podiel. Zadávajúca spoločnosť sa preto rozhodla nižšie verzie zanedbať a podporovať verzie od 4.4 vyššie, ako je uvedené v nefunkčných požiadavkách na aplikáciu.

1.4.5 Porovnanie programovacích jazykov Java a Kotlin

Java, jeden z najobľúbenejších programovacích jazykov [44], bola prvýkrát vydaná už v roku 1995 spoločnosťou Sun Microsystems. Jedným z najväčších benefitov tohto jazyka je jeho nezávislosť na platforme, a to pomocou spôsobu kompilácie. Kód napísaný v jazyku Java sa totiž kompiluje do formátu zvaného Java bajtkód, a následne sa spúšťa prostredníctvom Java virtuálneho stroja (JVM) [45]. Existujú stovky JVM programovacích jazykov [46], pričom jedným z nich je Kotlin. A práve tento jazyk vystriedal Javu, čo sa týka pozície prioritného jazyka pre vývoj Android aplikácií. Spoločnosť Google to oznámila v roku 2019 na konferencii Google I/O v San Franciscu [47].

Kotlin je, rovnako ako Java, staticky typovaný a objektovo orientovaný programátorský jazyk. Je plne interoperabilný s Javou, umožňujúc miešanie týchto dvoch jazykov v rámci jedného kódu. Vývoj jazyka Kotlin začal v roku 2010, a to skupinou JetBrains programátorov, žijúcich v Petrohrade [48]. Jazyk bol pomenovaný podľa ostrova Kotlin, ktorý sa nachádza blízko Petrohradu [49]. Vo februári 2016 bola prvýkrát verejne vydaná verzia jazyka Kotlin, a

následne v máji 2017 bol už súčasťou vývojového prostredia Android Studio [50]. Dnes tento jazyk používa viac ako 60% Android vývojárov [51].

Správny výber programovacieho jazyka pre našu Android aplikáciu vyžaduje porozumenie odlišností diskutovaných jazykov. V nasledujúcej časti si ich preto porovnáme, a to v 15 kategóriách. Čerpať budem z [48] [52] [53].

- Kontrolované výnimky

  Kontrolovaná výnimka musí byť buď odchytená, alebo uvedená v deklarácii danej funkcie. Jazyk Java na to využíva *try/catch* blok. Kotlin však kontrolované výnimky nemá, a teda odpadá nutnosť odchytávania, prípadne deklarácií.

- Korutiny

  Android aplikácie využívajú jedno hlavné vlákno. V prípade, že začne výpočet náročný na CPU alebo dlhotrvajúci vstupno-výstupný prenos, hlavné vlákno sa zablokuje, čo spôsobí zaseknutie používateľského rozhrania.

  V Jave môžeme v takomto prípade vytvoriť ďalšie vlákna, avšak Kotlin nám poskytuje lepšie riešenie, a to pomocou takzvaných korutín. Tie nám poskytnú nie len asynchrónne riešenie, ale aj možnosti ako súbežnosť a aktéri.

- Dátové triedy

  Čo sa týka implementácie tried, ktoré majú za úlohu len reprezentáciu dát bez ďalšej funkcionality, Kotlin jednoznačne skóruje.

  V Jave musíme v rámci takýchto tried vytvárať nie len členy reprezentujúce samotné dáta, ale aj množstvo funkcií ako konštruktor, funkcie na vygenerovanie hash kódu či reťazca daného objektu, prípadne na porovnanie obsahu dvoch objektov, ako aj funkcie slúžiace na získanie alebo nastavenie hodnoty, a to pre každého člena.

  Na druhej strane v Kotline stačí, ak do deklarácie triedy pridáme kľúčové slovo *data*, a o všetko ostatné vrátane spomínaných funkcií sa postará kompilátor.

- Rozširujúce funkcie

  Ak by sme v Jave potrebovali rozšíriť funkcionalitu triedy, museli by sme vytvoriť novú triedy, ktorá by dedila z tej pôvodnej. Kotlin však na rozdiel od Javy poskytuje takzvané rozširujúce funkcie, prostredníctvom ktorých vieme existujúcim triedam jednoducho dodefinovať novú funkcionalitu, a to bez nutnosti dedenia.

- Funkcionálne programovanie

Java síce od verzie 8 poskytuje podporu pre funkcionálne programovanie, avšak pri vývoji Android aplikácie je dostupná iba časť tejto funkcionality.

Kotlin, na druhej strane, poskytuje prostriedky ako napríklad funkcie vyššieho rádu či lambda výrazy. Funkcie vyššieho rádu sú také funkcie, ktorých návratová hodnota je funkcia, alebo funkciu prijímajú ako parameter. V Kotline sú navyše všetky funkcie takzvané „prvotriedne". To znamená, že môžu byť uložené v premenných a rôznych dátových štruktúrach.

Ďalším prostriedkom pre funkcionálne programovanie v Kotline sú lambda výrazy. Tie prijímajú parametre a majú návratové hodnoty, čím sú podobné funkciám, avšak na rozdiel od funkcií môžu byť implementované v rámci tela funkcie a taktiež nemusia mať názov. Využitím lambda výrazov tiež šetríme pamäť, ktorá by sa alokovala v prípade funkcie.

- Implicitné rozširujúce konverzie

  Java podporuje implicitné rozširujúce konverzie. To znamená, že menšie dátové typy, ktoré sú navzájom kompatibilné, sú v prípade potreby automaticky konvertované na väčšie. Kotlin však túto implicitnú konverziu nepodporuje.

- Inline funkcie

  Inline funkcia je funkcia, ktorá je pri zavolaní vložená do riadku. Kompilátor totiž nahradí funkčné volanie samotnou implementáciou funkcie. Využitím inline funkcií sa tak vyhneme réžii, ktorá je spojená s volaním klasických funkcií. Kotlin na rozdiel od Javy podporuje inline funkcie. V Jave dokážeme takéto volanie docieliť iba pomocou funkcií typu final.

- Null Safety

  Java nám umožňuje priradiť akejkoľvek premennej hodnotu *null*. Ak sa však následne pokúsime pristúpiť k tomuto objektu, vyvoláme tým výnimku neplatného ukazovateľa, takzvanú *null pointer exception*.

  Kotlin preto poskytuje Null Safety, čo znamená, že tento typ výnimky nikdy nenastane, ak nie je explicitne zavolaná. V tomto jazyku sú implicitne všetky premenné takzvané „non-nullable". To znamená, že ak sa pokúsime takejto premennej priradiť hodnotu *null*, nastane chyba už počas kompilácie. Aj v Kotline je ale možné priradiť premennej hodnotu *null*, a to označením premennej ako „nullable", čo docielime pridaním znaku *?* za dátový typ pri deklarácii premennej.

- Primitívne typy

  Java má osem takzvaných primitívnych typov. Sú to *boolean, byte, char, short, int, long, float* a *double*. Na rozdiel od Javy sú v Kotline tieto dátové typy objektami.

- Chytré pretypovanie

  Zatiaľ čo Jave je potrebné explicitné pretypovanie, Kotlin má o čosi chytrejší kompilátor, umožňujúc automatické pretypovanie. Toto pretypovanie nastáva v prípade použitia operátora *is*, pomocou ktorého je možné overiť typ premennej. V takom prípade, keďže typ premennej je už overený, by bolo explicitné pretypovanie nadbytočné.

- Statické členy

  V Jave máme možnosť vytvárať takzvané statické členy, a to pomocou kľúčového slova *static*. Statické členy nepatria konkrétnej inštancii, ale triede ako takej. To znamená, že nie je potrebné vytvárať inštanciu triedy, v ktorej sa statický člen nachádza, ale naopak, vieme k nemu pristupovať priamo. Kotlin vytváranie statických členov nepodporuje.

- Podpora pre konštruktory

  Kotlin môže mať okrem primárneho aj viacero sekundárnych konštruktorov. Sekundárny konštruktor je deklarovaný kľúčovým slovom *constructor*, a mal by odkazovať na primárny konštruktor. V jave to síce môžeme dosiahnuť preťažením konštruktora, avšak sekundárny konštruktor v Kotline nadväzuje na primárny a teda je potrebné uviesť len doplňujúcu implementáciu.

- Ternárny operátor

  Ternárny operátor pozostáva z jednej podmienky a dvoch hodnôt. Návratovou hodnotou je jedna z týchto hodnôt, v závislosti od toho, či je podmienka splnená. Java ternárny operátor podporuje, Kotlin nie.

- Zástupný znak

  Java poskytuje možnosť použitia zástupného znaku, ktorým vieme nahradiť typ parametra, premennej či návratovej hodnoty neznámym typom, a to pomocou znaku *?*. Kotlin toto použitie zástupného znaku neumožňuje.

- Odvodzovanie typu

  V Kotline je oproti Jave veľkou výhodou odvodzovanie typu. Typ premennej teda môžeme uviesť explicitne, avšak nie je to povinné.

- Lenivé načítanie

  Kotlin, na rozdiel od Javy, poskytuje takzvané lenivé načítanie, ktoré umožňuje odložiť inicializáciu objektu až do chvíle, kedy ho potrebujeme využiť. Pri štarte programu sa tak vieme vyhnúť načítavaniu nepotrebných objektov, čím dokážeme zrýchliť načítanie aplikácie.

Nasledujúca tabuľka 1.4 sumarizuje všetky diskutované odlišnosti jazykov Java a Kotlin.

Vlastnosť	**Java**	**Kotlin**
Kontrolované výnimky	Poskytuje	Neposkytuje
Korutiny	Neposkytuje	Poskytuje
Dátové triedy	Potrebné písať množstvo štandardného, opakujúceho sa kódu	Stačí pridať kľúčové slovo *data* do deklarácie triedy
Rozširujúce funkcie	Neposkytuje	Poskytuje
Funkcionálne programovanie	Od verzie Java 8, avšak len obmedzene	Poskytuje množstvo prostriedkov, ako napríklad funkcie vyššieho rádu či lambda výrazy
Implicitné rozširujúce konverzie	Poskytuje	Neposkytuje
Inline funkcie	Neposkytuje	Poskytuje
Null Safety	Neposkytuje	Poskytuje
Primitívne typy	Premenné primitívnych typov nie sú objekty	Premenné primitívnych typov sú objekty
Chytré pretypovanie	Neposkytuje	Poskytuje
Statické členy	Poskytuje	Neposkytuje
Podpora pre konštruktory	Neposkytuje sekundárne konštruktory, viacero konštruktorov len pomocou preťaženia	Poskytuje viacero sekundárnych konštruktorov
Ternárny operátor	Poskytuje	Neposkytuje
Zástupný znak	Poskytuje	Neposkytuje

Odvodzovanie typu	Neposkytuje	Poskytuje
Lenivé načítanie	Neposkytuje	Poskytuje

Tabuľka 1.4: Porovnaie programovacích jazykov Java a Kotlin.

1.4.6 Výber programovacieho jazyka

Na základe predošlého porovnania som sa pre túto aplikáciu rozhodla použiť programovací jazyk Kotlin. Toto rozhodnutie vyplynulo najmä z nasledujúcich dôvodov.

Kotlin je moderný jazyk s množstvom vylepšení oproti Jave, ako napríklad chytré pretypovanie, korutiny, prostriedky pre funkcionálne programovanie či rozširujúce funkcie. V porovnaní s Javou je tiež omnoho stručnejší, čo je zabezpečené dátovými triedami, nekontrolovaním výnimiek, chytrým pretypovaním či sekundárnymi konštruktormi. Stručnosť kódu je pri programovaní veľmi dôležitá, zabezpečujúc prehľadnosť kódu, ako aj zvyšujúc efektivitu programátora. Ďalšou významnou vlastnosťou Kotlinu je Null Safety, ktorá nám umožňuje písať bezpečnejší kód. Pri rozhodovaní tiež zavážil fakt, že Kotlin je spoločnosťou Google odporúčaným jazykom pre vývoj Android aplikácií.

1.5 Analýza API

REST (Representational State Transfer) je architektonický štýl, ktorý slúži na jednotný a jednoduchý prístup k zdrojom inej aplikácie alebo služby. Charakterizujú ho nasledovné princípy. [54]

- Jednotné rozhranie

  Prístup k jednému zdroju je definovaný jednotne, a to pomocou operácií CRUD, čiže vytvorenie, čítanie, aktualizovanie a vymazanie.

- Oddelenie klienta a servera

  Aplikácie klient a server by mali byť od seba úplne nezávislé. Klient by mal poznať iba identifikátor zdroja a server by mal s klientom interagovať iba zaslaním požadovaných dát.

- Bezstavovosť

Bezstavovosť znamená, že každá požiadavka musí obsahovať všetky informácie potrebné na jej vykonanie.

- Cache

  Využitím cache pamäte medzi klientom a serverom dokážeme zvýšiť efektivitu na strane klienta. Požiadavky by preto mali byť označené podľa toho, či je možné ich uložiť do pamäte cache.

- Vrstvená architektúra systému

  Klient a server nemusia komunikovať priamo, môžu sa medzi nimi vyskytovať sprostredkovatelia. Je totiž umožnené vrstvenie služieb, čím môžeme dosiahnuť väčšiu variabilitu.

- Kód na požiadanie

  Odpoveď zo servera môže obsahovať kód, ktorý je možné z klienta spustiť na požiadanie. Tento princíp je voliteľný.

Aplikačné rozhranie, ktoré spĺňa tieto architektonické obmedzenia, nazývame REST API.

Existujúci súkromný server zadávajúcej spoločnosti mal implementované rozhranie REST API, avšak pri analýze tohto API bolo identifikovaných niekoľko nedostatkov. Hlavným nedostatkom bolo posielanie veľkého množstva dát, v prípadoch kedy to nie je potrebné, ako napríklad hneď po prihlásení do aplikácie. Z tohto dôvodu vznikla diskusia so zadávajúcou spoločnosťou, na základe ktorej začala zadávajúca spoločnosť meniť API. Posledná dohodnutá verzia navrhnutých úprav REST API je nasledovná.

- POST /login

  Táto metóda slúži pre prihlásenie existujúceho používateľa. Požiadavka má povinné telo, ktoré pozostáva z používateľského meno a hešovaného hesla.

- POST /logout

  Metóda na odhlásenie prihláseného používateľa. Rovnako ako pri prihlásení, má požiadavka informáciu o používateľskom mene a tiež hešované heslo.

- POST /campaigns/[0-9]+/settings

  Táto metóda slúži na označenie stavu zvolenej reklamnej kampane.

- POST /uploadAreaNote

  Pomocou tejto metódy dokážeme pridať poznámku k zvolenej reklamnej ploche.

- POST /uploadRackImage

  Metóna na pridanie fotografie zvoleného reklamného nosiča. Požiadavka obsahuje obrázok zahešovaný pomocou base64.

- GET /campaigns

  Táto metóda slúži na získanie všetkých reklamných kampaní.

- GET /campaigns/id

  Metóda na získanie kamnpane so zadaným id.

Zadávajúca spoločnosť však doposiaľ nedokončila implementáciu tohto návrhu. Z tohto dôvodu bude v rámci tejto diplomovej práce aplikácia testovaná iba na mockovanom REST API, ktoré zodpovedá uvedenému návrhu.

Kapitola 2

Návrh

V tejto kapitole sa budeme venovať návrhu aplikácie. V prvej časti sa zameriame na návrh používateľského rozhrania. Zostavíme definíciu produktu, obchodné požiadavky a zoznam úloh aplikácie. Vytvoríme lo-fi prototyp a popíšeme proces testovania použiteľnosti tohto prototypu. Na záver popíšeme výber architektúry a technológií vhodných pre túto aplikáciu.

2.1 Návrh používateľského rozhrania

Používateľské rozhranie, označované skratkou UI, je rozhranie, prostredníctvom ktorého používateľ komunikuje s počítačom. Môže mať tieto základné formy [55].

- Rozhranie príkazového riadku

  Používateľ so systémom interaguje prostredníctvom sady príkazov.

- Grafické používateľské rozhranie

  Používateľ interaguje s vizuálnymi prvkami softvéru.

- Používateľské rozhranie riadené hlasom

  Používateľ interaguje s počítačom pomocou hlasu. Príkladom sú hlasový asistenti ako Siri alebo Google Asistent.

- Používateľské rozhranie riadené gestami

  Používateľ interaguje v 3D priestore pomocou pohybu. Príkladom je virtuálna alebo obohatená realita.

Komunikácia prostredníctvom hlasu ani gest nie v našej aplikácii potrebná, preto sa budeme ďalej zaoberať len grafickým rozhraním.

Návrh používateľského rozhrania je cyklický proces pozostávajúci z 3 krokov, a to návrhu, implementácie a vyhodnotenia systému z hľadiska použitia

človekom. Zameranie na človeka znamená, že rozhranie a interakcie sa snažíme navrhnúť tak, aby uspokojili potreby a očakávania používateľov. Používateľské rozhranie by malo byť organizované, efektívne, odolné voči chybám a v neposlednom rade tiež pohodlné na používanie. Hlavné princípy dobrého používateľského rozhrania sú nasledovné. [56]

- Orientácia na používateľa
- Zobrazovanie iba užitočných informácií
- Spätná väzba systému
- Predvídateľnosť správania používateľa
- Tolerancia chýb používateľa
- Odolnosť voči chybám
- Jednoduchosť použitia systému

V tejto kapitole si ďalej popíšeme základné kroky návrhu používateľského rozhrania. Prvým krokom je vytvorenie definície produktu. Následne je možné vykonať posúdenie potrieb. Tento krok je vhodný v prípade, ak si zadávateľ nie je istý tým čo chce, alebo ak cieľová skupina používateľov nie je známa [56]. V našom prípade posúdenie potrieb nie je nutné, a preto sa ním ďalej nebudem zaoberať. Po definícii produktu teda nasleduje identifikácia obchodných požiadaviek. Ďalej je potrebné vytvorenie prípadov použitia, ktoré boli diskutované v kapitole 1.3. Ďalším krokom je identifikácia úloh aplikácie. Po dokončení týchto krokov sa vytvorí prototyp aplikácie. Ten sa vyhodnotí zvolenou technikou, a na základe tohto vyhodnotenia sa vykonajú úpravy v návrhu používateľského rozhrania.

2.1.1 Definícia produktu

Pred tým ako sa pustíme do samotného riešenia problému, je dôležitým krokom definovanie produktu. Definícia produktu popisuje rozdiel medzi súčasným a požadovaným stavom produktu. Je dôležitým počiatočným krokom, pretože prostredníctvom tejto definície lepšie porozumieme produktu. Zároveň nám teda pomáha vyjasniť, aký problém budeme riešiť. Definícia produktu by v sebe mala obsiahnuť názov a popis produktu, čo má vykonávať a pre koho je určený. Dobrá definícia produktu sa vyznačuje tým, že je zameraná na používateľa. Zároveň by mala byť dostatočne všeobecná, avšak iba do takej miery, aby nás počas návrhu sprevádzala správnym smerom. [57, 56]

Definícia produktu je nasledovná.

Felco je mobilná aplikácia pre spoločnosť zameranú na prevádzkovanie reklamných médií, určená na evidenciu prenájmu reklamných plôch a zároveň na dokumentáciu inštalácie a technického servisu reklám a reklamných nosičov.

2.1.2 Obchodné požiadavky

Obchodné požiadavky definujú obchodné potreby zadávateľa, a tým aj kritéria jeho úspechu. Tieto požiadavky nedefinujú čo má byť produkt schopný vykonávať, ani ako bude implementovaný. Pred vytváraním obchodným požiadaviek je potrebné rozumieť produktu, a preto tento krok nasleduje až po dokončení definície produktu. Je dôležité, aby obchodné požiadavky boli stopercentne naplnené. [58]

Ďalej je uvedená výsledná formulácia obchodných požiadaviek.

- Prehľad o stave reklamných kampaní
- Prehľad reklamných plôch a reklamných nosičov
- Jednoduchšia a rýchlejšia dokumentácia inštalácie a technického servisu reklám a reklamných nosičov
- Efektivita práce technických pracovníkov
- Kontrola práce technických pracovníkov

2.1.3 Zoznam úloh

Po dokončení predchádzajúcej analýzy, spočívajúcej v definovaní produktu a stanovení obchodných požiadaviek, a zároveň po vytvorení používateľských prípadov, sa môžeme presunúť na ďalší krok návrhu UI, ktorým je vytvorenie zoznamu úloh aplikácie. Tento zoznam určuje všetky činnosti, ktoré aplikácia poskytuje, a to z hľadiska interakcie s používateľom. Úlohy je vhodné zoskupiť do kategórií podľa obrazoviek aplikácie. Ďalej v rámci kategórie je tiež vhodné ich rozdeliť na zobrazovacie úlohy a úlohy reprezentujúce používateľské akcie.

- Prihlásenie
  - Zobrazenia
    - Obrazovka prihlásenia
    - Zobrazenie správy o neúspešnom prihlásení
  - Používateľské akcie
    - Zadanie prihlasovacích údajov
    - Prihlásenie
- Reklamné kampane
  - Zobrazenia
    - Zoznam reklamných kampaní
    - Zobrazenie ID, názvu a obdobia každej reklamnej kampane

        * Označenie každej reklamnej kampane ako ukončená alebo neukončená
    - Používateľské akcie
        * Označenie reklamnej kampane ako ukončená alebo neukončená
        * Výber reklamnej kampane

- Reklamné plochy
    - Zobrazenia
        * Zoznam vyfiltrovaných reklamných plôch
        * Zobrazenie aplikovaných filtrov
        * Zobrazenie ponúkaných možností filtrovania na základe názvu, stavu alebo geografickej oblasti reklamných plôch
        * Zobrazenie názvu, kódu, adresy a celkového počtu fotiek každej reklamnej plochy
        * Označenie stavu každej reklamnej plochy
    - Používateľské akcie
        * Výber filtrov reklamných plôch
        * Aplikovanie filtrov
        * Zrušenie výberu filtrov
        * Zadanie textu na vyhľadávanie reklamných plôch
        * Vyhľadanie reklamných plôch podľa zadaného textu
        * Výber reklamnej plochy
        * Návrat na prehľad kampaní

- Detail reklamnej plochy
    - Zobrazenia
        * Zobrazenie názvu, kódu, adresy a celkového počtu fotiek reklamnej plochy
        * Označenie stavu reklamnej plochy
        * Zoznam reklamných nosičov prislúchajúcich danej reklamnej ploche
            · Zobrazenie názvu a počtu fotiek reklamného nosiča
            · Zobrazenie aplikácie Fotoaparát daného zariadenia
            · Zobrazenie galérie fotiek reklamného nosiča
        * Zobrazenie možnosti pridania poznámky v rôznych kategóriách
    - Používateľské akcie
        * Zadanie textu na vyhľadávanie reklamných nosičov
        * Vyhľadanie reklamných nosičov podľa zadaného textu

        * Otvorenie aplikácie Fotoaparát zvoleného reklamnému nosiča
            · Vytvorenie fotografie
            · Návrat z aplikácie Fotoaparát
        * Otvorenie galérie reklamného nosiča
            · Vymazanie fotografie z galérie reklamného nosiča
            · Návrat z galérie nosiča
        * Výber kategórie poznámky
            · Zadanie textu poznámky
        * Aplikovanie zmien reklamnej plochy
        * Návrat na zoznam reklamných plôch

- Účet
    - Zobrazenia
        * Obrazovka účtu
        * Zobrazenie základných informácií o aplikácii
        * Zobrazenie informácií o synchronizácii aplikácie
    - Používateľské akcie
        * Zapnutie/vypnutie automatickej synchronizácie cez WiFi
        * Odhlásenie

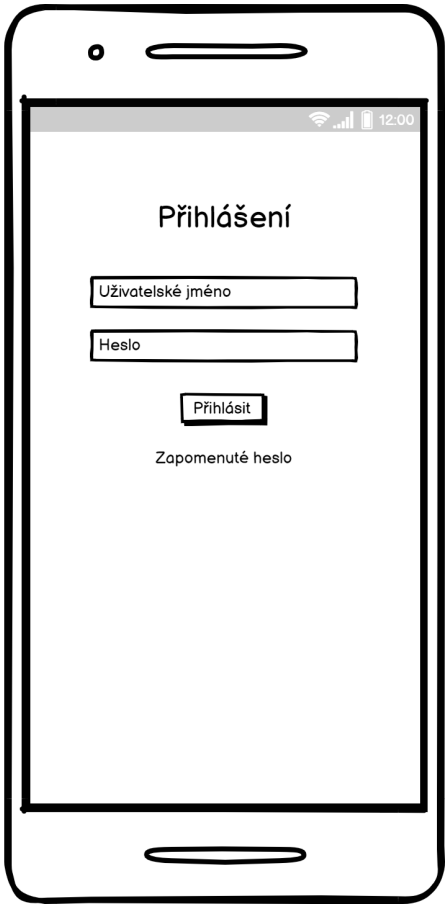
2.1.4 Lo-fi prototyp

Lo-fi prototyp je vizuálna pomôcka, reprezentujúca kostru aplikácie. Tento prototyp vytvárame s cieľom dosiahnutia takého rozloženia prvkov, ktorý by čo najlepšie spĺňal daný účel. Názov Lo-fi je skratka z anglického low-fidelity, čiže nízka vernosť, pričom vernosť chápeme ako úroveň detailov prototypu. Lo-fi prototyp teda nemá farby ani grafické prvky, zameriavajúc sa iba na rozloženie prvkov, funkcionalitu a správanie aplikácie. Je to jednoduché, lacné a efektívne riešenie tvorby prototypu, umožňujúce rýchle zavádzanie zmien. Tento typ prototypu je veľmi vhodným nástrojom pre komunikáciu nápadov v projektovom tíme. [56]

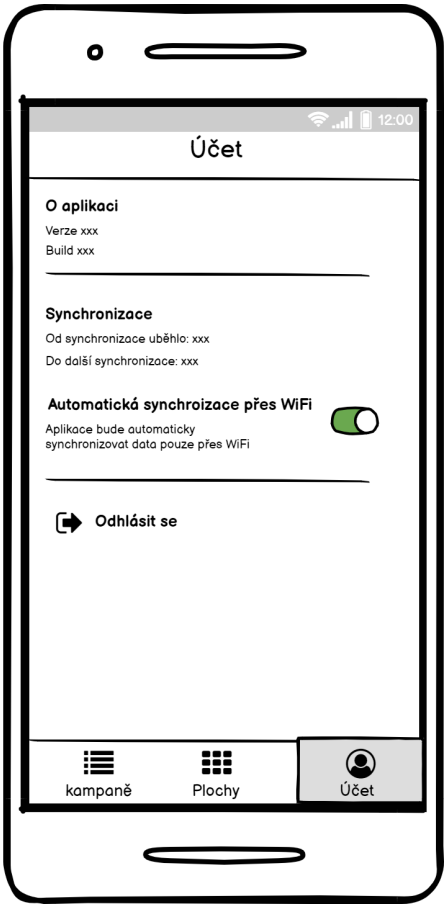
Vieme ho vytvoriť formou papierového prototypu iba pomocou pera a papiera, alebo môžeme použiť softvérový nástroj. V tejto práci som sa rozhodla zvoliť online nástroj Balsamiq.

Obrazovka prihlásenia a účtu

Obrazovka prihlásenia, znázornená na obrázku 2.1, zobrazuje možnosť prihlásenia pomocou používateľského mena a hesla. Po neúspešnom prihlásení bude používateľ informovaný o nesprávnom zadaní údajov, a to prostredníctvom krátkej toastovej správy. V prípade zabudnutého hesla má používateľ možnosť



Obr. 2.1: Obrazovka prihlásenia.



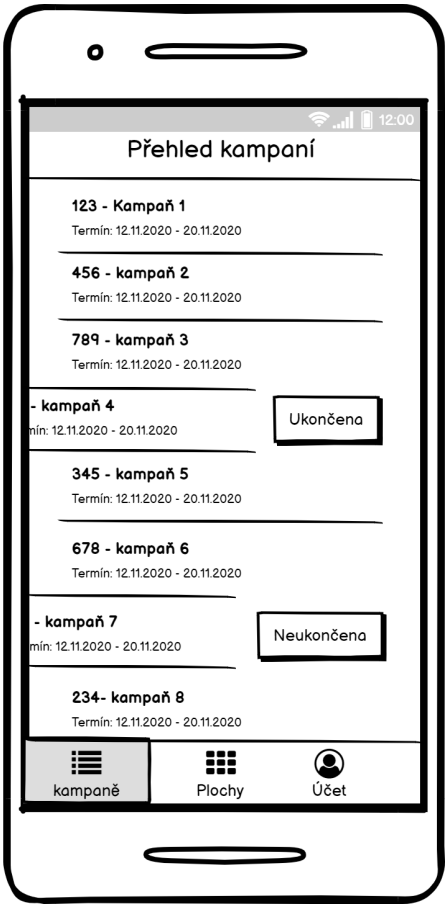
Obr. 2.2: Obrazovka účtu.

kliknúť na tlačidlo „Zapomenuté heslo“, na základe čoho sa otvorí dialóg, požadujúci používateľské meno.

Hlavné menu aplikácie sa vyskytuje v spodnej časti obrazovky a ostáva nemenné po celú dobu prihlásenia používateľa. Skladá sa z troch položiek, pričom poslednou je obrazovka účtu, znázornená na obrázku 2.2. Okrem možnosti odhlásiť sa poskytuje základné informácie o aplikácii. Ďalej informuje o stave synchronizácie aplikácie a poskytuje možnosť zapnutia automatickej synchronizácie dát iba prostredníctvom WiFi.

Obrazovka reklamných kampaní

Obrazovka 2.3 zobrazuje prehľad všetkých reklamných kampaní. Je prvou položkou hlavého menu aplikácie, pretože reklamná kampaň predstavuje základnú zložku biznis štruktúry aplikácie. Zoznam reklamných kampaní má byť krátky, preto nie sú potrebné možnosti filtrovania ani vyhľadávania. Re-

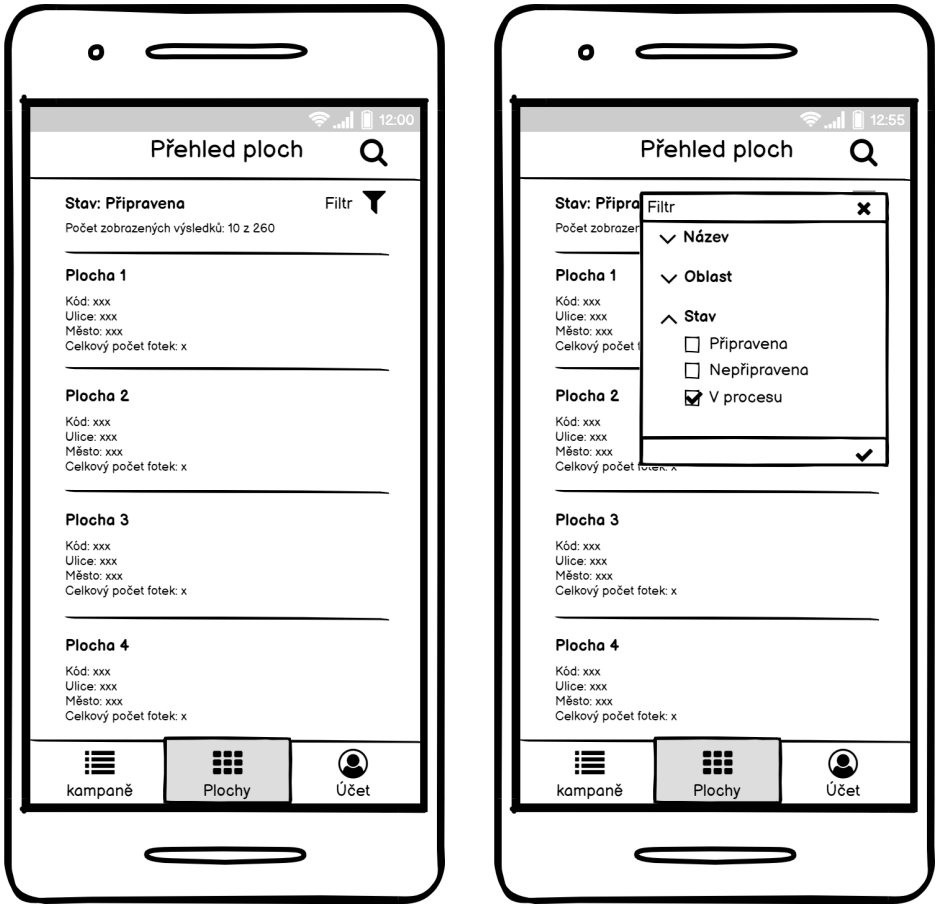


Obr. 2.3: Obrazovka reklamných kampaní.

klamné kampane majú priradené identifikačné číslo, názov a tiež obdobie trvania kampane. Každá kampaň je tiež označená ako Ukončená alebo Neukončená. Používateľ má možnosť tento stav zmeniť, a to potiahnutím položky smerom vľavo, na základe čoho sa objaví tlačidlo umožňujúce zmenu stavu kampane. Po kliknutí na zvolenú kampaň sa zobrazí zoznam reklamných plôch prislúchajúcich danej kampani. To sa už ale týka obrazovky, ktorú si opíšeme ďalej.

Obrazovka reklamných plôch

Pod druhou položkou menu sa skrýva obrazovka reklamných plôch, znázornená na obrázku 2.4. Každá reklamná plocha je identifikovaná názvom a kódom. Poskytuje tiež informácie o adrese a celkovom počte fotiek danej plochy. Zoznam reklamných plôch má rádovo stovky položiek, a preto má používateľ možnosť vyhľadávania a filtrovania v tomto zozname. Obe tieto

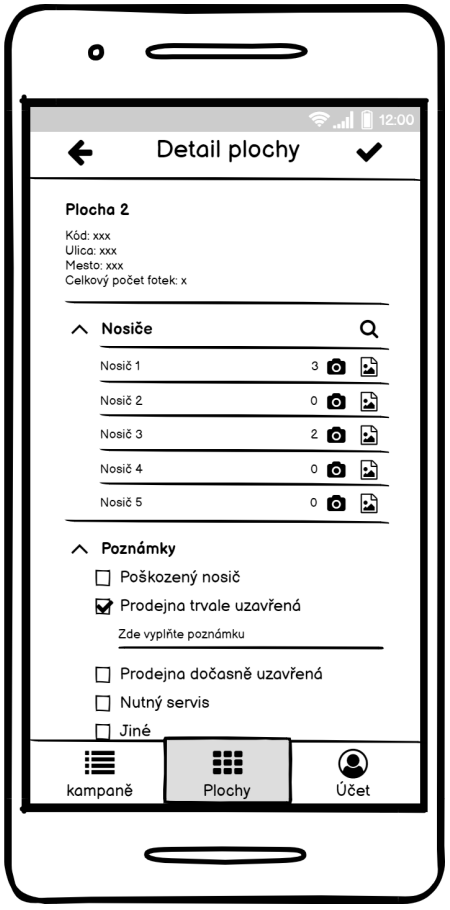


(a) Prehľad reklamných plôch. (b) Filtrovanie reklamných plôch.

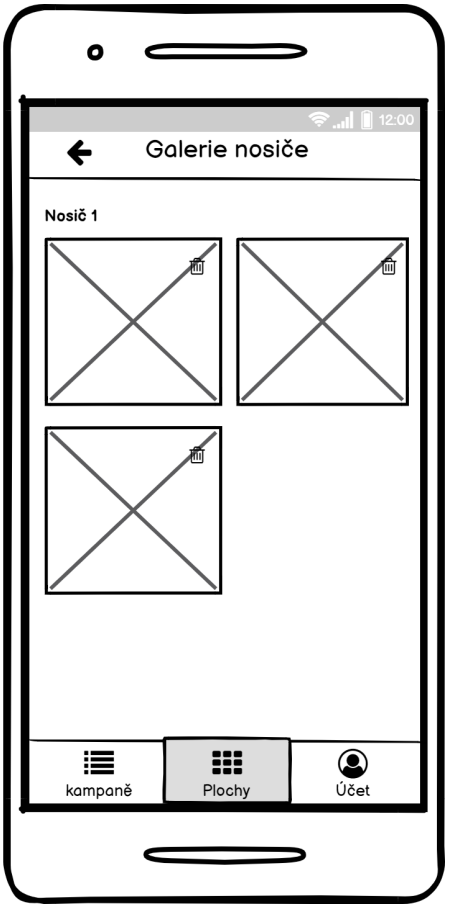
Obr. 2.4: Obrazovka reklamných plôch.

možnosti sú dostupné v pravom hornom rohu obrazovky. Ak sa na obrazovku reklamných plôch dostaneme výberom reklamnej kampane, zoznam reklamných plôch bude obsahovať len plochy prislúchajúce danej kampani. V takomto prípade má používateľ možnosť návratu na obrazovku reklamných kampaní. Po kliknutí na položku zo zoznamu reklamných plôch sa zobrazí detail zvolenej plochy, ktorý si popíšeme v ďalšej časti.

Čo sa týka možností filtrovania zoznamu reklamných plôch, používateľ má možnosť výberu z troch kategórií, ktorými sú názov, oblasť a stav plochy. Filtrovanie prebieha prostredníctvom dialógu, znázorneného na obrázku 2.4b. Kategórie sú zložiteľné a poskytujú možnosť výberu viacerých položiek, ako aj žiadnej položky. V pravom dolnom rohu tohto dialógu sa nachádza tlačidlo na potvrdenie zvolených filtrov. Na obrazovke reklamných plôch, a to nad samotným zoznamom plôch, nájdeme informáciu o všetkých aplikovaných filtroch.



Obr. 2.5: Obrazovka detailu reklamnej plochy.



Obr. 2.6: Obrazovka galérie reklamného nosiča.

Obrazovka detailu reklamnej plochy a galérie reklamného nosiča

Obrazovka detailu zvolenej reklamnej plochy 2.5 poskytuje okrem základných informácií aj zoznam reklamných nosičov prislúchajúcich danej ploche a možnosť pridania poznámky, pričom oba tieto celky sú z dôvodu prehľadnosti zložiteľné.

Čo sa týka zoznamu reklamných nosičov, z dôvodu možného vyššieho počtu položiek má používateľ možnosť vyhľadávania v tomto zozname podľa zadaného textu. Každá položka je identifikovaná názvom, po ktorom nasleduje informácia o počte vytvorených fotografií. Ďalej má používateľ možnosť vytvorenia fotografie a otvorenia galérie zvoleného reklamného nosiča. Vytvorenie fotografie sa uskutoční prostredníctvom aplikácie Fotoaparát daného zariadenia, z ktorej sa vieme vrátiť do našej aplikácie, a to späť na obrazovku detailu reklamnej plochy. Obrazovka galérie reklamného nosiča 2.6 zobrazuje

všetky fotografie prislúchajúce tomuto nosiču. Používateľ má možnosť vymazania zvolenej fotografie kliknutím na ikonu v pravom hornom rohu fotografie. Obrazovka galérie nosiča tiež umožňuje návrat na obrazovku detailu plochy.

Používateľ má možnosť pridať poznámku k danej reklamnej ploche, a to výberom z ponúkaných kategórií poznámok. Tento výber je navrhnutý formou zaškrtávacieho políčka, čo umožňuje výber viacerých kategórií a tiež žiadnej. Po výbere kategórie sa zobrazí textové pole, do ktorého je možné napísať detail poznámky, avšak tento krok nie je povinný.

V prípade, že sme detail reklamnej plochy akokoľvek zmenili, či už pripadaním poznámky, vytvorením alebo vymazaním fotografie, máme možnosť tieto zmeny aplikovať, a to potvrdzujúcim tlačidlom v pravom hornom roku obrazovky. Taktiež máme možnosť návratu na obrazovku reklamných plôch.

2.1.4.1 Testovanie prototypu

V tejto kapitole čerpám z [56]. Testovanie lo-fi prototypu aplikácie je veľmi dôležité nie len z hľadiska použiteľnosti, ale aj z dôvodu, že v tejto fáze návrhu je zavádzanie zmien rýchle, efektívne a lacné. Prototyp aplikácie môžeme testovať kvalitatívnym alebo kvantitatívnym spôsobom.

Kvantitatívne testovanie je čisto vedecké. Je založené na štatistických metódach, vyžadujúcich veľké množstvo dát, a teda výstupom takéhoto testovanie sú čísla. Pomocou tohto testovania môžeme nájsť odpovede na otázky typu „Ako dlho trvá používateľovi, kým nájde správnu položku menu?“.

Na druhej strane, využitím kvalitatívneho testovania môžeme nájsť konkrétne problémy súvisiace s návrhom používateľského rozhrania aplikácie. Kvalitatívne testovania vyžaduje iba malý počet testujúcich osôb. Využitím tohto spôsobu testovania môžeme nájsť odpovede na všeobecnejšie otázky, ako napríklad „Ako je možné vylepšiť určitý proces.“. Kvalitatívne testovanie môžeme vykonávať ako s používateľmi, tak aj bez nich.

Kvalitatívne testovanie bez používateľov je vykonávané expertom v danej oblasti a výstupom je expertný posudok. Príkladom takéhoto testovania je heuristická evaluácia alebo kognitívna prechádzka. Kognitívna prechádzka spočíva vo vžití sa do role používateľa expertom, ktorý vykonáva testovanie. Tento expert dostane pred testovaním podrobný popis prototypu a následne sa snaží prechádzať systémom z perspektívy používateľa. Výsledok tohto testovania je však neurčitý. Druhým spôsobom je heuristická evaluácia, ktorá spočíva v ohodnotení heuristických pravidiel testujúcim expertom. Poznáme viacero heuristických pravidiel, avšak najvýznamnejšie sú tie od Jakoba Nielsena z roku 1994. Je to nasledujúcich desať heuristík [59].

1. Viditeľnosť stavu systému

   Používateľ by mal dostávať rozumnú spätnú väzbu, a to tak, aby bol vždy informovaný o tom, čo sa práve v systéme deje.

2. Zhoda medzi systémom a skutočným svetom

   Systém by mal odrážať reálny svet. Texty a pojmy použité v systéme by mali byť používateľom známe, informácie by mali byť v rozumnom a prirodzenom usporiadaní.

3. Kontrola a sloboda používateľa

   Používateľ by mal mať v každej situácii istotu, že proces môže kedykoľvek ukončiť. Dodá mu to pocit slobody a nedostane sa do situácie, ktorá by mohla vyvolať pocit frustrácie.

4. Konzistencia a štandardy

   Dodržiavanie konvencií platforiem a odvetvia je dôležité preto, aby sa používateľ nemusel zamýšľať nad tým, či rôzne slová alebo akcie znamenajú to isté.

5. Prevencia chýb

   Systém by mal byť navrhnutý tak, aby čo najviac predchádzal chybám. Či už vylúčením podmienok náchylných na chyby alebo vyžiadaním potvrdenia od používateľa na vykonanie akcie.

6. Radšej rozpoznanie ako spomínanie si

   Všetky prvky systému by mali byť jasne viditeľné. Keď používateľ prechádza z jednej časti rozhrania do druhej, nemal by byť nútený spomínať si na informácie z predchádzajúcej časti, naopak, všetky informácie by mali byť viditeľné všade tam, kde sú potrebné.

7. Flexibilita a efektívnosť používania

   Systém by mal skúseným používateľom umožňovať efektívnejšie používanie prostredníctvom skratiek, skrytých pred menej zdatnými používateľmi.

8. Estetický a minimalistický dizajn

   Rozhranie by malo obsahovať len potrebné informácie, pretože menej dôležité informácie by znížili ich viditeľnosť.

9. Pomoc používateľom rozpoznať, diagnostikovať a zotaviť sa z chýb

   Chybové správy by mali byť prezentované jednoduchým jazykom, jasne definujúc problém aj riešenie.

10. Pomoc a dokumentácia

    V ideálnom prípade je systém dostatočne zrozumiteľný, nepotrebujúc žiadne dodatočné vysvetlenie. Niekedy je však potrebné používateľom pomôcť pochopiť systém, a to poskytnutím dokumentácie.

Týchto heuristických pravidiel sa treba držať nie len pri testovaní, ale najmä vo fáze návrhu používateľského rozhrania systému.

Kvalitatívne testovanie s používateľmi je vykonávané potenciálnymi budúcimi používateľmi. Toto testovanie prebieha tak, že participantom zadáme úlohy a následne sledujeme, kedy rozhranie systému nepodporuje splnenie úlohy, prípadne kladie prekážky. Na základe toho sa identifikujú potrebné zmeny používateľského rozhrania. Poznáme niekoľko typov kvalitatívneho testovania s používateľmi.

- Prieskumy používateľov

  Od participantov sa vyžaduje vyplnenie dotazníka, a teda toto testovanie môže byť uskutočnené aj online.

- Etnografické pozorovanie

  Participantov pozorujeme v ich prirodzenom prostredí bez akejkoľvek interakcie a počas toho si zaznamenávame priebeh

- Využitie inžinierstva

  Testovanie sa uskutočňuje v kontrolovanom prostredí, v takzvanom laboratóriu použiteľnosti, ktoré simuluje reálne prostredie. Zvyčajne sa vykoná prieskum a tiež pozorovanie.

V tejto práci som sa rozhodla aplikovať kvalitatívne testovanie s používateľmi využitím inžinierstva, pretože oproti testovaniu bez používateľov nám poskytuje náhľad do správania používateľa. Testovanie je ideálne vykonávať s potenciálnymi budúcimi požívateľmi v prostredí, v ktorom sa bude aplikácia reálne používať, prípade v laboratóriu použiteľnosti. V mojom prípade však nemám možnosť testovania budúcich používateľov v takomto prostredí. Testovanie bude preto vykonávané s participantmi rôznych vekových skupín a technických zručností. V dôsledku pandemických opatrení, súvisiacich s vírusom Covid-19, budú všetky testovania prebiehať online, pričom participanti sa budú nachádzať vo svojom domácom prostredí.

Zoznam participantov testovania, spolu s informáciou o pracovnej oblasti a platforme ich mobilného zariadenia, je nasledovný.

- Bc. Terézia Škorupová
  - 27 rokov
  - Študentka masmediálnej komunikácie
  - Používateľka platformy iOS
- Mgr. art. Ivan Šveda
  - 46 rokov

  - Marketingový manažér
  - Vzdelanie v umeleckom obore
  - Používateľ platformy iOS
- Jaroslav Šturm
  - 24 rokov
  - Technický špecialista vysielacích technológií
  - Stredoškolské vzdelanie
  - Používateľ platformy iOS
- Mgr. Richard Rác
  - 28 rokov
  - Marketingový špecialista
  - Pedagogické vzdelanie
  - Používateľ platformy iOS
- Sofia Mandelíková
  - 21 rokov
  - Študentka architektúry
  - Používateľka platformy Android

Participantom boli zadané nasledovné testovacie scenáre spolu s doplňujúcimi otázkami.

- Scenár 1
  1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
  2. Prihláste sa do aplikácie.
  3. Zistite, aká je vaša aktuálna verzia aplikácie.
  4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
  5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.
- Scenár 2
  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných kampaní.

3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.
4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.

- Scenár 3

1. Ste prihlásený/á v aplikácii.
2. Nájdite zoznam všetkých reklamných plôch.
3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
6. Zistite počet nájdených plôch.
7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.

- Scenár 4

1. Ste prihlásený/á v aplikácii.
2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.
4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
5. Vytvorte novú fotografiu tohto nosiča.
6. Zobrazte všetky fotografie tohto nosiča.
7. Vymažte ľubovoľnú fotografiu tohto nosiča.
8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
9. Aplikujte vykonané zmeny danej reklamnej plochy.
10. Odhláste sa z aplikácie.

- Doplňujúce otázky

1. Ako sa vám páčila aplikácia?
2. Zdalo sa vám používanie dostatočne intuitívne?
3. Zmenili by ste niečo?

Testovanie odhalilo niekoľko problémov použiteľnosti prototypu, avšak väčšina súvisela s nedostatočnou viditeľnosťou ovládacích prvkov alebo textu. Výsledky testovania sumarizuje tabuľka 2.1. Zobrazuje problémy použiteľnosti identifikované počas testovania so všetkými participantmi, ako aj návrh riešenia týchto problémov. Obsahuje tiež informáciu o závažnosti jednotlivých problémov, ktorá je označená číslom 1 až 3. Toto číslo vyjadruje, ako veľmi bol používateľ obmedzený pri riešení danej testovacej úlohy.

Problém	**Závažnosť**	**Riešenie**
Neintuitívne zobrazenie zoznamu reklamných plôch prislúchajúcich zvolenej kampani. Správnym riešením bolo kliknutie na danú kampaň, participanti sa však snažili nájsť riešenie v možnostiach filtrovania plôch.	3	Pridanie možnosti filtrovania reklamných plôch podľa kampane.
Nevýrazné tlačidlo pre potvrdenie parametrov filtrovania, umiestnené v pravom dolnom rohu dialógu v podobe ikonky ✓.	2	Zmena podoby tlačidla. Ikonka ✓ bude nahradená textom „OK“.
Nevýrazná informácia o počte zobrazených výsledkov filtrovania reklamných plôch.	1	Táto informácia má veľmi nízku dôležitosť, preto nie je potrebné ju zdôrazňovať, aby nezatienila dôležitejšie prvky rozhrania.

Tabuľka 2.1: Zhrnutie výsledkov testovania použiteľnosti prototypu.

2.2 Výber architektúry

Android aplikácie majú omnoho zložitejšiu štruktúru ako desktopové aplikácie, ktoré majú zvyčajne len jeden prístupový bod a bežia ako jeden monolitický proces [4]. Základné stavebné prvky Android aplikácií sú takzvané aplikačné komponenty, definované v aplikačnom manifeste, ktorý reprezentuje súbor s názvom AndroidManifest.xml. Prostredníctvom aplikačných komponentov dokáže používateľ alebo systém vstúpiť do aplikácie. Poznáme tieto základné typy komponentov, ktoré môžu byť spúšťané individuálne. [60]

- Activity

Activity komponent reprezentuje obrazovku s UI, umožňujúcim komunikáciu aplikácie s používateľom. Android aplikácia má jednu alebo viacero activity komponentov, ktoré sú navzájom nezávislé. Táto nezávislosť umožňuje, v prípade že to daná aplikácia povolí, spustenie akéhokoľvek activity komponentu inou aplikáciou. [60]

- Fragment

  Aplikačný komponent typu fragment predstavuje časť UI, ktorá môže byť použitá opakovane. Fragmenty existujú len v rámci prislúchajúceho activity komponentu, ktorým sú riadené. [61]

- Service

  Service je komponent, ktorý beží na pozadí a neposkytuje UI. Využíva sa pre rôzne druhy úloh, ktoré sú časovo náročné alebo vykonávajú prácu pre externé procesy. Service komponent môže byť spustený iným aplikačným komponentom, alebo sa naň môže tento komponent naviazať, čo umožní vzájomnú komunikáciu. [60]

- Broadcast receiver

  Pomocou broadcast receiver komponentu dokáže aplikácia reagovať na celosystémové vysielacie oznámenia, a to tak, že umožňuje do aplikácie doručovať udalosti mimo bežnú používateľskú interakciu. Na tieto udalosti reaguje rôzne, môže to to byť napríklad oznámenie o slabej batérii alebo dokončení sťahovania dát. Je možné použiť systémový broadcast receiver, alebo si vytvoriť vlastný. [60]

- Content provider

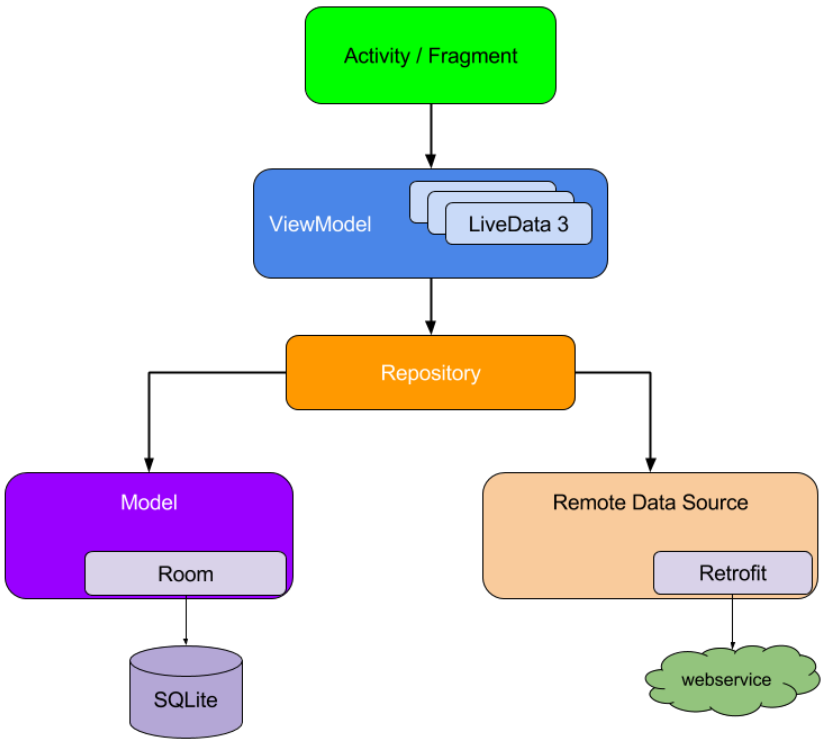
  Komponent typu content provider je aplikačné rozhranie slúžiace na správu a zdieľanie aplikačných dát. Tieto dáta môžu byť uložené v súbore, na webe, v databáze SQLite, alebo na inom perzistentnom úložisku, ku ktorému má aplikácia prístup, a pomocou content provider komponentu môžu tieto dáta modifikovať aj iné aplikácie. Príkladom takýchto dát sú kontaktné informácie, ku ktorým môže aplikácia v prípade povolenia prístupu pomocou tohto komponentu pristupovať. [60]

Aplikačné komponenty by mali byť od seba nezávislé a nemal by sa v nich uchovávať stav ani dáta. Tieto komponenty totiž môžu byť kedykoľvek odstránené Android operačným systémom, a to napríklad z dôvodu malej pamäte. Preto by sme sa mali držať princípu oddelenia zodpovednosti vytvárania UI z modelu. [4]

Princíp oddelenia zodpovednosti napríklad znamená, že do tried typu activity a fragment, ktoré sú zodpovedné iba za rozhranie, by sme nemali pridávať všetok kód. Ďalším princípom je vytváranie UI z modelu, ktorý by mal byť

ideálne perzistentný. Komponenty typu model sa starajú o aplikačné dáta a sú nezávislé od životného cyklu aplikácie. [4]

V tomto riešení je na získanie aplikačných dát použité REST API súkromného servera zadávajúcej spoločnosti. Obrázok 2.7 ďalej znázorňuje platformou Android odporúčanú architektúru, ktorou som sa riadila.



Obr. 2.7: Android aplikačná Architektúra. Prevzaté z [4]

Na obrázku 2.7 vidíme architektonický vzor odporúčaný pre vývoj Android aplikácií typu Model-View-ViewModel, označovaný skratkou MVVM. Účelom vrstvy zvanej Model je abstrahovanie dátových zdrojov. Vrstva View, realizovaná komponentami typu activity alebo fragment, je zodpovedná za informovanie vrstvy ViewModel o používateľských vstupoch. Komponenty typu ViewModel a Repository sú pre túto architektúru špecifické, a preto si ich ozrejmíme osobitne. Architektonický vzor MVVM svojou modularitou zabezpečuje dobrú testovateľnosť a udržateľnosť implementácie. [62]

Obrázok 2.7 tiež znázorňuje, ako majú aplikačné komponenty navzájom komunikovať, pričom každý komponent by mal byť závislý len od komponentu, ktorý je priamo pod ním. Vďaka architektúre, ktorá lokálne perzistuje dáta, môžu používatelia ihneď pristupovať k aplikačným dátam aj po tom, čo na určitý čas opustili aplikáciu. [4]

ViewModel

Komponenty typu ViewModel poskytujú dáta pre UI komponenty, či už activity alebo fragment, ku ktorým prislúchajú. Obsahujú logiku na spracovanie údajov a komunikujú s modelom. [4]

Repository

Úložiskový modul zvaný repository sa stará o dátové operácie a poskytuje API na jednoduché získanie dát z iných častí aplikácie. Repository sprostredkováva rôzne dátové zdroje, ktoré v aplikácii používame. Tieto dátové zdroje môžu byť napríklad webové služby, perzistntné modely alebo rýchla vyrovnávacia pamäť. [4]

2.3 Výber technológií

Pri voľbe technológií som sa takisto riadila odporúčaniami platformy Android. V porovnaní s technológiami znázornenými na obrázku 2.7 som sa rozhodla v rámci ViewModel komponentov namiesto typu LiveData použiť Flow.

LiveData

LiveData je pozorovateľný komponent, ktorý informuje fragmenty o obdržaní dát a tiež rešpektuje životný cyklus týchto komponentov. LiveData umožňuje iným komponentom sledovať zmeny dátových objektov, a to bez pevnej závislosti s týmito objektami. Namiesto LiveData som sa však rozhodla použiť komponent Flow. [4]

Flow

Flow je podobný typu Sequence, pretože oba sa vyznačujú takzvanou lenivou produkciou prvkov. To znamená, že prvok sa vytvorí až v momente, keď je potrebný. Rozdiel medzi týmito dvoma typmi je v tom, že Flow podporuje korutiny, čo je v kotline alternatíva asynchrónneho spracovávania k použitiu Rxjava. To znamená, že použitím typu Flow dokážeme jednoducho produkovať prvky z asynchrónnych operácií, ako napríklad databázové dotazy. [63]

Retrofit

Čo sa týka získania dát zo vzdialeného dátového zdroja, jednou z možností je použitie knižnice Retrofit. Retrofit je typovo bezpečný http klient, ktorý

môžeme použiť k prístupu na beckendový server. Môžeme ju použiť za predpokladu, že tento server poskytuje REST API, čo je v našom prípade splnené. [4]

Room

Použitím technológie Retrofit získame dáta, avšak tie nie sú nikde uložené. Na to, aby sa nemuseli pri každom vytvorení fragmentu všetky potrebné dáta získavať opäť zo vzdialeného zdroja, potrebujeme dáta lokálne perzistovať, k čomu nám slúži objektovo mapovacia knižnica Room. Tá dokáže kontrolovať databázové dotazy podľa dátovej schémy, a vďaka tomu sa chybné dotazy odchytia už počas kompilácie. [4]

Kapitola 3

Realizácia

Táto kapitola popisuje vývoj aplikácie. Najprv si priblížime proces implementácie, štruktúru kódu a pomocné nástroje, ktoré boli použité počas vývoja. Následne opíšeme testovanie tejto aplikácie pomocou jednotkových testov.

3.1 Implementácia

Aplikácia bola vyvíjaná v prostredí Android Studio, poskytovaným spoločnosťou Google. Android Studio disponuje inteligentným editorom, ktorý výrazne zjednodušil proces implementácie. Pri tvorbe rozloženia grafických prvkov obrazoviek je možné využiť vizuálny editor [64], avšak v tejto práci sa obrazovky vytvárali iba prostredníctvom XML zdrojov. Vývojové prostredie Android Studio tiež ponúka okrem spúšťania na identifikovaných fyzických zariadeniach aj možnosť spustenia aplikácie na emulátoroch rôznych typov [65]. Táto funkcia bola využitá na testovanie používateľského rozhrania, a to spúšťaním aplikácie na rôznych typoch obrazoviek mobilných zariadení. Organizácia kódu je nasledovná.

- Data

  V tomto adresári sa nachádzajú komponenty zodpovedné za získavanie a prezistenciu dát. Nachádza sa tu definícia dátového modelu, ako aj implementácia úložiskového modulu repository, ktorý poskytuje API na získanie dát z iných častí aplikácie. V časti preference sa ukladajú používateľské prihlasovacie údaje, ako aj preferencie používateľa. Je tu tiež realizované vkladanie závislostí pomocou aplikačného rámca Koin.

- UI

  Sekcia UI je ďalej organizovaná podľa jednotlivých obrazoviek. Obrazovka má príslušný fragment, ViewModel, prípadne fragment dialógu danej obrazovky, či iné pomocné triedy obrazovky. Viaceré obsahujú

napríklad triedu typu Adapter, ktorá umožňuje generovanie grafických prvkov počas behu aplikácie.

- Utils

  Utils obsahuje triedy s pomocnými metódami. Tie sú jednoducho využiteľné viacerými časťami aplikácie, a to vďaka tomu, že dané triedy sú reprezentované statickými singleton objektami. V rámci tohto adresára sa nachádza metóda na hešovanie a tiež rozširujúca funkcia na automatické pridávanie označenia triedy, ktoré je potrebné zadávať pri logovaní.

Koin

V aplikácii je realizované vkladanie závislosti, v angličtine označované ako dependency injection. Je to technika umožňujúca tvorbu závislostí medzi komponentami bez toho, aby na seba priamo odkazovali [66]. Prináša tak zníženie závislosti medzi komponentami, ako aj zjednodušenia testovania. V tomto riešení je vkladanie závislostí realizované pomocou aplikačného rámca Koin [67]. Ten sa postará o to, aby sa po spustení aplikácie vytvoril graf závislostí, pričom tento graf sa vytvorí z module súborov, ktoré sa vyskytujú v adresári di.module. Príkladom takéhoto súboru je PreferenceModule.kt, ktorý vytvorí jednu inštanciu preferencií spolu so synchronizovaným prístupom k tejto inštancii.

Zdrojový kód 3.1: Ukážka použitia Koin.

```
val preferenceModule = module {
    single<IPreferenceRepository> {
        PreferenceRepository(get())
    }
}
```

SharedPreferences

Rozhranie SharedPreferences poskytuje API na prístup a modifikáciu preferencií typu kľúč a hodnota [68]. Toto rozhranie môžeme použiť na uloženie malého množstva takýchto dát. SharedPreferences garantuje silnú konzistenciu, čo ale môže mať za následok spomalenie aplikácie. V tejto aplikácii sa SharedPreferences používa na uloženie prihlasovacích údajov, identifikácie zariadenia, ako aj na uloženie preferencie o automatickej synchronizácii iba cez WiFi.

3.1.1 Získanie a prezistencia dát

Za získanie a perzistenciu dát je zodpovedný úložiskový modul repository, ktorý poskytuje API prístupu k dátam iným častiam aplikácie a využíva dátový model.

Repository

Rrepository získava dáta zo vzdialeného zdroja, a to na základe REST API požiadaviek, ktoré sú zadefinované v súboroch adresára api. V rámci tejto diplomovej práce sa však z dôvodu uvedenom v kapitole 1.5 využíva mockované API. Ďalšou úlohou repository je uloženie dát do databázy a poskytnutie prístupu k týmto dátam.

Model

Model reprezentuje REST API požiadavky a odpovede. V modeloch sa využíva anotácia @Json, ktorá slúži k serializácii do formátu JSON, ako aj deserializáciu z tohto formátu, a to prostredníctvom knižnice Moshi. Napríklad uvedená ukážka zobrazuje model s názvom LoginRequestModel, ktorý serializuje požiadavku na prihlásenie do JSON formátu, a z toho následne vytvorí požiadavku, pričom to isté platí aj pre odpoveď.

Zdrojový kód 3.2: Ukážka dátového modelu.

```
data class LoginRequestModel(
    @Json(name = "username")
    val username: String,
    @Json(name = "password")
    val passwordHash: String
)
```

3.1.2 Používateľské rozhranie

Čo sa týka definície rozloženia grafických prvkov obrazoviek v XML súbo-roch, boli použité rôzne podporované typy rozložení zabudované do platformy Android. Zvolené boli podľa vhodnosti pre rozloženie danej obrazovky, prípadne boli v rámci obrazovky kombinované viaceré typy rozložení. Použité typy rozložení boli nasledovné.

Lienar Layout

Pomocou rozloženia Lienar Layout dokážeme zadefinovať lineárne usporiadanie grafických prvkov, či už vertikálne alebo horizontálne.

V rámci UI tejto aplikácie bol Lienar Layout vhodný pre obrazovku účtu. Ďalej je uvedená ukážka z XML zdroja tejto obrazovky, spolu so znázornením na obrázku 3.1.

Zdrojový kód 3.3: Ukážka použitia Linear Layout.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginHorizontal="16dp" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="8dp"
        android:text="@string/about_app_title"
        android:textAppearance=
            "?android:attr/textAppearanceListItem" />

    <TextView
        android:id="@+id/text_view_version"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/text_view_build"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:layout_marginVertical="20dp"
        android:background="#c0c0c0" />
    ...
</LinearLayout>
```

Obr. 3.1: Ukážka rozloženia Linear Layout z obrazovky účtu.

Constraint Layout

Constraint Layout je založený na definovaní vzťahov medzi elementami. Tento typ rozloženia je možné vytvoriť aj pomocou vizuálneho editora, avšak v tejto práci boli všetky obrazovky vytvorené priamo v XML súbore.

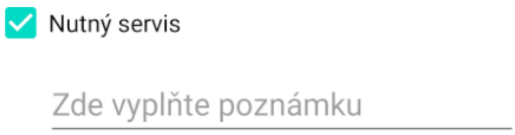
Toto rozloženie bolo použité v rámci obrazovky detailu reklamnej plochy v časti pridávania poznámok, ako môžeme vidieť na ukážke XML zdroja spolu s ukážkou danej časti obrazovky 3.2.

Zdrojový kód 3.4: Ukážka použitia Constraint Layout.

```
<androidx.constraintlayout.widget.ConstraintLayout
    android:id="@+id/expandable_view_notes"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:visibility="gone"
    android:layout_marginTop="8dp"
    android:layout_marginStart="10dp"
    app:layout_constraintTop_toBottomOf=
        "@+id/btn_arrow_notes"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent">
    ...
    <CheckBox
        android:id="@+id/check_box_service"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/note_service"
        app:layout_constraintTop_toBottomOf=
            "@id/edit_text_store_temporary_closed"
        app:layout_constraintStart_toStartOf="parent"/>

    <EditText
        android:id="@+id/edit_text_service"
        android:layout_width="fill_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="30dp"
        android:visibility="gone"
        android:inputType="text"
        android:hint="@string/note_hint"
        app:layout_constraintTop_toBottomOf=
            "@id/check_box_service"
        app:layout_constraintStart_toStartOf="parent"/>
    ...
</androidx.constraintlayout.widget.ConstraintLayout>
```



Obr. 3.2: Ukážka Constraint Layout z obrazovky detailu reklamnej plochy.

Za obrazovky a interakciu s používateľom sú zodpovedné komponenty typu activity a fragment, pričom v tejto aplikácii bol potrebný iba jeden activity komponent a každej obrazovke prislúcha jeden fragment.

3.1.3 Testovanie

Testovanie v Android je možné vykonávať niekoľkými spôsobmi. Pre platformu android môžeme používať klasické testovacie nástroje ako sú JUnit testy, Mockito a podobne. Zaujímavým nástrojom pre testovacie Android aplikácií je Espresso framework. Je to nástroj na testovanie aplikácií, kde napísaný kód za nás preklikáva aplikáciu a zároveň kontroluje prvky na zadaných pozíciách, či obsah nejakých textov. Týmto spôsobom dobre napísanými testami môžeme pokryť veľa scenárov a ušetriť si veľa manuálneho testovania.

Zdrojový kód 3.5: Ukážka .

```
@Test
    fun loginTest() {
        val appCompatEditText = onView(
            allOf(
                withId(R.id.username),
                childAtPosition(
                    allOf(
                        withId(R.id.container),
```

```
                childAtPosition(
                    withId(R.id.nav_host_fragment),
                    0
                )
            ),
            0
        ),
        isDisplayed()
    )
)
appCompatEditText.perform(replaceText("meno"),
↪  closeSoftKeyboard())
...
```

V uvedenom úryvku kódu testu písaného pomocou Espresso frameworku môžeme napríklad vidieť, ako zadávame text "meno"do poľa üsername". Kód môže vyzerať trochu krkolomne, pretože tu potrebujeme získať z UI konkrétne textové pole, ktoré budeme vypĺňať, čo môže byť celkom zdĺhavé. Ak si však s písaním dáme dostatok práce, tak tým môžeme pokryť veľké množstvo prípadov použitia aplikácie a vyhnúť sa tým chybám typu zlé UI, tlačidlo nereaguje a podobne.

Záver

Cieľom tejto práce bolo navrhnúť a implementovať Android mobilnú aplikáciu, ktorá by komunikovala so serverom spoločnosti zameranej na prevádzkovanie reklamných médií. Aplikácia by mala slúžiť na evidenciu prenájmu reklamných plôch a zároveň na dokumentáciu inštalácie a technického servisu reklám a reklamných nosičov. Mali byť analyzované potreby zákazníkov a na základe toho definované funkčné a nefunkčné požiadavky na aplikáciu. Ďalej mal prebehnúť softvérový návrh vrátane papierového modelu, voľba architektúry a technológií. Prototyp aplikácie mal byť otestovaný, a na základe týchto testov mali byť navrhnuté úpravy a implementovaná výsledná aplikácia.

V analýze som sa venovala charakteristikám rôznych typov aplikácií. Následne som vykonala analýza požiadaviek na aplikáciu. Zadefinovala som rôzne skupiny, ako aj rôzne metódy zberu požiadaviek, a nakoniec som popísala výslednú formuláciu funkčných a nefunkčných požiadaviek. Na základe týchto požiadaviek boli identifikované prípady použitia.

Následne som vykonala analýzu aplikačnej platformy. V rámci tejto časti som porovnala rôzne mobilné platformy, a na základe toho som vybrala platformu Android s minimálnou podporovanou verziou 4.4 . Ďalej som tiež porovnala programovacie jazyky Kotlin a Java. Výsledkom tohto porovnania bol výber jazyka Kotlin.

Zoznámila som sa s REST API súkromného servera zadávajúcej spoločnosti, avšak pri analýze tohto API bolo identifikovaných niekoľko nedostatkov. Z tohto dôvodu vznikla diskusia so zadávajúcou spoločnosťou, na základe ktorej začala zadávajúca spoločnosť meniť API. Nestihla však zapracovať všetky zmeny, a z toho dôvodu bola v rámci tejto diplomovej práce aplikácia testovaná iba na mockovanom REST API.

Na základe analýzy som zvolila vhodnú architektúru, technológie a vytvorila návrh používateľského rozhrania, ktorého výstupom bol lo-fi prototyp. Na tomto prototype som realizovala testovanie použiteľnosti, a k identifikovaným problémom som navrhla úpravy.

Následne som aplikáciu spolu s navrhnutými úpravami implementovala a

podrobila automatizovaným testom. Vhodné by bolo vykonanie akceptačného testovania, avšak z dôvodu nedodaných zmien implementácie REST API zadávajúcou spoločnosťou sa testovanie zatiaľ realizované nebolo.

Výsledkom tejto práce je funkčná aplikácia, ktorá však ešte bude vyžadovať úpravy na základe meniaceho sa API, prípade tiež v dôsledku zmeny požiadaviek zo strany zadávateľa.

Literatúra

[1] Mobile Operating System Market Share Worldwide on 2020. *StatCounter [online]. [cit. 2. 5. 2021]*. Dostupné z: https://gs.statcounter.com/os-market-share/mobile/worldwide/#yearly-2020-2020-bar

[2] Mobile Operating System Market Share Worldwide May 2012 - May 2021. *StatCounter [online]. [cit. 20. 5. 2021]*. Dostupné z: https://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-201205-202105

[3] Mobile & Tablet Android Version Market Share Worldwide on 2020. *StatCounter [online]. [cit. 2. 5. 2021]*. Dostupné z: https://gs.statcounter.com/android-version-market-share/mobile-tablet/worldwide/#yearly-2020-2020-bar

[4] Guide to app architecture. *developer.android.com [online]. 2021 [cit. 2. 6. 2021]*. Dostupné z: https://developer.android.com/jetpack/guide

[5] Serrano, N.; Hernantes, J.; Gallardo, G.: Mobile Web Apps. *IEEE Software*, ročník 30, č. 5, 2013: s. 22–27, doi:10.1109/MS.2013.111.

[6] Codenames, Tags, and Build Numbers. *source.android.com [online]. 2021 [cit. 20. 5. 2021]*. Dostupné z: https://source.android.com/setup/start/build-numbers

[7] McLaughlin, M.: Android Versions Guide: Everything You Need to Know. *Lifewire [online]. 2021 [cit. 21. 5. 2021]*. Dostupné z: https://www.lifewire.com/android-versions-4173277

[8] Gurbaxani, V.; Whang, S.: The Impact of Information Systems on Organizations and Markets. *Commun. ACM*, ročník 34, č. 1, Leden 1991, ISSN 0001-0782, doi:10.1145/99977.99990. Dostupné z: https://doi.org/10.1145/99977.99990

[9] Main, J.: The winning organization. *Fortune*, 1988. Dostupné z: https://money.cnn.com/magazines/fortune/fortune_archive/1988/09/26/71058/index.htm

[10] Markgraf, B.: Importance of Information Systems in an Organization. *Chron [online]. 2019 [cit. 18. 4. 2021].* Dostupné z: https://smallbusiness.chron.com/importance-information-systems-organization-69529.html

[11] Sørensen, C.; De Reuver, M.; Basole, R. C.: Mobile platforms and ecosystems. *Journal of Information Technology*, 2015. Dostupné z: https://doi.org/10.1057/jit.2015.22

[12] Zimmerman, J. B.: Mobile computing: Characteristics, business benefits, and the mobile framework. *University of Maryland European Division-Bowie State*, ročník 10, 1999: str. 12.

[13] Deepak, G.; Pradeep, B.: Challenging issues and limitations of mobile computing. *International Journal of Computer Technology & Applications*, ročník 3, č. 1, 2012: s. 177–181.

[14] Flora, H.; Wang, X.; Chande, S.: An Investigation on the Characteristics of Mobile Applications: A Survey Study. *I.J. Information Technology and Computer Science*, ročník 11, 10 2014: s. 21–27, doi:10.5815/ijitcs.2014.11.03.

[15] Malavolta, I.: Beyond Native Apps: Web Technologies to the Rescue! (Keynote). In *Proceedings of the 1st International Workshop on Mobile Development*, Mobile! 2016, New York, NY, USA: Association for Computing Machinery, 2016, ISBN 9781450346436, str. 1–2, doi:10.1145/3001854.3001863. Dostupné z: https://doi.org/10.1145/3001854.3001863

[16] Fling, B.: *Mobile design and development: Practical techniques for creating mobile sites and Web apps.* O'Reilly Media, Inc., 2009, ISBN 978-0-596-16644-5.

[17] Joorabchi, M. E.; Mesbah, A.; Kruchten, P.: Real Challenges in Mobile App Development. In *2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*, 2013, s. 15–24, doi:10.1109/ESEM.2013.9.

[18] Popa, M.: Audit Process during Projects for Development of New Mobile IT Applications. *Informatica Economica*, ročník 14, č. 3, 2010.

[19] Andreou, A. S.; Panayidou, D.; Andreou, P.; aj.: Preserving Quality in the Development of Mobile Commerce Services and Applications. In *Automation, Control, and Information Technology*, 2005, s. 1–8.

[20] Pocatilu, P.; Boja, C.; aj.: Quality characteristics and metrics related to m-learning process. *Amfiteatru Economic*, ročník 11, č. 26, 2009: s. 346–354.

[21] Garofalakis, J. D.; Stefani, A.; Stefanis, V.; aj.: Quality Attributes of Consumer-Based m-Commerce Systems. In *ICE-B*, 2007, s. 130–136.

[22] Rabi'u, S.; Ayobami, A. S.; Hector, O.: Usability characteristics of mobile applications. In *Proceedings of International Conference on Behavioural & Social Science Research (ICBSSR), Kampar, Malaysia.(Indexed by Thomson Reuters)*, ročník 2, 2012.

[23] Kolokolov, V.; Baumann, P.; Santini, S.; aj.: Flexible development of variable software features for mobile business applications. In *Proceedings of the 17th International Software Product Line Conference co-located workshops*, 2013, s. 67–73.

[24] Wasserman, A. I.: Software Engineering Issues for Mobile Application Development. FoSER '10, New York, NY, USA: Association for Computing Machinery, 2010, ISBN 9781450304276, str. 397–400, doi:10.1145/1882362.1882443. Dostupné z: `https://doi.org/10.1145/1882362.1882443`

[25] Salmre, I.: Characteristics of Mobile Applications. *Chapter*, ročník 2, 2004: s. 19–36.

[26] Šilhavý, R.; Šilhavý, P.; Prokopová, Z.: Requirements gathering methods in system engineering. *Recent Researches in Automatic Control - 13th WSEAS International Conference on Automatic Control, Modelling and Simulation, ACMOS'11*, May 2011: s. 106–110.

[27] Aurum, A.; Wohlin, C.: *Engineering and Managing Software Requirements*. Springer-Verlag Berlin Heidelberg, 2005, ISBN 978-3-540-28244-0, 19–49 s.

[28] Carlshamre, P.; Karlsson, J.: A usability-oriented approach to requirements engineering. *Proceedings of the Second International Conference on Requirements Engineering*, 1996.

[29] Lynch, W.: All You Need to Know About Use Case Modeling. *Medium [online]. 2019 [cit. 16. 4. 2021].* Dostupné z: `https://warren2lynch.medium.com/all-you-need-to-know-about-use-case-modeling-828756da3215`

[30] Jacobson, I.: Object-Oriented Development in an Industrial Environment. OOPSLA '87, New York, NY, USA: Association for Computing Machinery, 1987, ISBN 0897912470, str. 183–191, doi:10.1145/38765.38824. Dostupné z: `https://doi.org/10.1145/38765.38824`

[31] Silva, N.: Use Case Diagram Relationships Explained with Examples. *Creately [online]. 2020 [cit. 16. 4. 2021]*. Dostupné z: https://creately.com/blog/diagrams/use-case-diagram-relationships/

[32] Keizer, G.: Microsoft's enterprise phone strategy flops as revenue evaporates. *Computerworld [online]. 2017 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/3193644/microsofts-enterprise-phone-strategy-flops-as-revenue-evaporates.html

[33] Vaughan-Nichols, S. J.: Ubuntu switches to GNOME desktop and gives up smartphone hopes. *ZDNet [online]. 2017 [cit. 6. 5. 2021]*. Dostupné z: https://www.zdnet.com/article/ubuntu-switches-to-gnome-desktop-and-gives-up-smartphone-hopes/

[34] Sayer, P.: BlackBerry hands its brand to TCL, maker of its last smartphones. *Computerworld [online]. 2016 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/3151183/blackberry-hands-its-brand-to-tcl-maker-of-its-last-smartphones.html

[35] Rosenbaum, D.: BlackBerry returns with a surprisingly decent business phone: The KEYone. *Computerworld [online]. 2017 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/3192659/blackberry-returns-with-a-surprisingly-decent-business-phone-the-keyone.html

[36] Palmer, J.: iPhone vs. Android: Which is better for you? *Tom's Guide [online]. 2021 [cit. 6. 5. 2021]*. Dostupné z: https://www.tomsguide.com/face-off/iphone-vs-android

[37] Mearian, L.: iOS vs. Android: When it comes to brand loyalty, Android wins. *Computerworld [online]. 2018 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/3262051/ios-vs-android-when-it-comes-to-brand-loyalty-android-wins.html

[38] Vaughan-Nichols, S. J.: iPhone vs. Android: Which is better for you? *Computerworld [online]. 2018 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/2468474/iphone-vs-android-which-is-better-for-you.html

[39] Vaughan-Nichols, S. J.: What went wrong with iOS 6 Wi-Fi. *ZDNet [online]. 2012 [cit. 6. 5. 2021]*. Dostupné z: https://www.zdnet.com/article/what-went-wrong-with-ios-6-wi-fi/

[40] Hamblen, M.: Most Android devices lack latest security patches. *CSO [online]. 2017 [cit. 6. 5. 2021]*. Dostupné z: https://www.csoonline.com/article/3184690/most-android-devices-lack-latest-security-patches.html

[41] Price, D.: Do iPhones get viruses? *Macworld [online]. 2020 [cit. 6. 5. 2021]*. Dostupné z: https://www.macworld.co.uk/feature/do-iphones-get-viruses-3655942/

[42] Kan, M.: Google Play faces cat-and-mouse game with Android malware. *Computerworld [online]. 2017 [cit. 6. 5. 2021]*. Dostupné z: https://www.computerworld.com/article/3184211/google-play-faces-cat-and-mouse-game-with-android-malware.html

[43] Support different platform versions. *developer.android.com [online]. 2020 [cit. 17. 5. 2021]*. Dostupné z: https://developer.android.com/training/basics/supporting-devices/platforms

[44] Cass, S.: The top programming languages: Our latest rankings put Python on top-again - [Careers]. *IEEE Spectrum*, ročník 57, č. 8, 2020: s. 22–22, doi:10.1109/MSPEC.2020.9150550.

[45] Java™ Platform Overview. *Oracle Java Documentation [online]. 2021 [cit. 19. 5. 2021]*. Dostupné z: https://docs.oracle.com/javase/8/docs/technotes/guides/index.html

[46] Urma, R.-G.: Alternative Languages for the JVM. A look at eight features from eight JVM languages. *oracle.com [online]. 2014 [cit. 19. 5. 2021]*. Dostupné z: https://www.oracle.com/technical-resources/articles/java/architect-languages.html

[47] Rathod, P.: Google announced Kotlin priority programming language for developing Android applications. *DEV Community [online]. 2019 [cit. 19. 5. 2021]*. Dostupné z: https://dev.to/prathaprathod/google-announced-kotlin-priority-programming-language-for-developing-android-applications-58l4

[48] Bose, S.; Mukherjee, M.; Kundu, A.; aj.: A comparative study: java vs kotlin programming in android application development. *International Journal of Advanced Research in Computer Science*, ročník 9, č. 3, 2018: str. 41.

[49] Scott, J.: How programming languages got their names. *DEV Community [online]. 2020 [cit. 19. 5. 2021]*. Dostupné z: https://dev.to/scottydocs/how-programming-languages-got-their-names-207e

[50] Gotseva, D.; Tomov, Y.; Danov, P.: Comparative study Java vs Kotlin. In *2019 27th National Conference with International Participation (TELECOM)*, IEEE.

[51] Develop Android apps with Kotlin. *Google Developers [online]. [cit. 19. 5. 2021]*. Dostupné z: https://developer.android.com/kotlin

[52] Singh Khatri, V.: Kotlin vs Java: Important Differences That You Must Know. *Hackr.io [online]. 2020 [cit. 24. 5. 2021]*. Dostupné z: https://hackr.io/blog/kotlin-vs-java

[53] Navdeep Singh, G.: Kotlin vs Java: Which is Better for Android App Development? *XenonStack [online]. 2020 [cit. 24. 5. 2021]*. Dostupné z: https://www.xenonstack.com/blog/kotlin-andriod/

[54] What is a REST API? *IBM [online]. 2021 [cit. 11. 6. 2021]*. Dostupné z: https://www.ibm.com/cloud/learn/rest-apis

[55] User Interface Design. *Interaction Design Foundation [online]. [cit. 17. 5. 2021]*. Dostupné z: https://www.interaction-design.org/literature/topics/ui-design

[56] Pavlíček, J.: Návrh uživatelského rozhraní. Praha, FIT ČVUT, 2020.

[57] Stevens, E.: How To Define A Problem Statement: Your Guide To The Second Step In The Design Thinking Process. *CareerFoundry [online]. 2019 [cit. 18. 5. 2021]*. Dostupné z: https://careerfoundry.com/en/blog/ux-design/stage-two-design-thinking-define-the-problem/

[58] Cox, A.: Business Requirements vs Functional Requirements? Who Cares? *Netmind [online]. 2017 [cit. 18. 5. 2021]*. Dostupné z: https://netmind.net/en/business-vs-functional-requirements-who-cares/

[59] Nielsen, J.: Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 1994, s. 152–158.

[60] Application Fundamentals. *developer.android.com [online]. 2021 [cit. 2. 6. 2021]*. Dostupné z: https://developer.android.com/guide/components/fundamentals

[61] Fragments. *developer.android.com [online]. 2020 [cit. 2. 6. 2021]*. Dostupné z: https://developer.android.com/guide/fragments

[62] Mishra, R.: Android Architecture Patterns. *GeeksforGeeks [online]. 2021 [cit. 8. 6. 2021]*. Dostupné z: https://www.geeksforgeeks.org/android-architecture-patterns/

[63] Introducing Flow. *developer.android.com [online]. [cit. 3. 6. 2021]*. Dostupné z: https://developer.android.com/codelabs/advanced-kotlin-coroutines#7

[64] Build a UI with Layout Editor. *developer.android.com [online]. 2020 [cit. 25. 6. 2021]*. Dostupné z: https://developer.android.com/studio/write/layout-editor

[65] Run apps on the Android Emulator. *developer.android.com [online]. 2021 [cit. 25. 6. 2021]*. Dostupné z: https://developer.android.com/studio/run/emulator

[66] Bhavya, K.: A quick intro to Dependency Injection: what it is, and when to use it. *freeCodeCamp [online]. 2018 [cit. 25. 6. 2021]*. Dostupné z: https://www.freecodecamp.org/news/a-quick-intro-to-dependency-injection-what-it-is-and-when-to-use-it-7578c84fa88f/

[67] What is Koin? *insert-koin.io [online]. [cit. 25. 6. 2021]*. Dostupné z: https://insert-koin.io/docs/reference/introduction

[68] SharedPreferences. *developer.android.com [online]. 2021 [cit. 25. 6. 2021]*. Dostupné z: https://developer.android.com/reference/android/content/SharedPreferences

Dodatok **A**

Zoznam použitých skratiek

IS Informačný Systém

OS Operačný systém

UI User Interface

UX User Experience

SSL Secure Sockets Layer

HTML HyperText Markup Language

API Application programming interface

REST Representational state transfer

JVM Java Virtual Machine

Lo-fi Low fidelity

MVVM Model-View-ViewModel

XML eXtensible Markup Language

JSON JavaScript Object Notation

Dodatok B

Testovanie prototypu aplikácie

Táto príloha opisuje detailný opis priebehu testovania lo-fi prototypu aplikácie so všetkými participantmi.

B.1 Bc. Terézia Škorupová

- Scenár 1

  1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
     - Prebehlo bez problémov.
  2. Prihláste sa do aplikácie.
     - Prebehlo bez problémov.
  3. Zistite, aká je vaša aktuálna verzia aplikácie.
     - Prebehlo bez problémov.
  4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
     - Prebehlo bez problémov.
  5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.
     - Prebehlo bez problémov.

- Scenár 2

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných kampaní.
     - Prebehlo bez problémov.

3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.
   - Prebehlo bez problémov.
4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
   - Prebehlo bez problémov.
5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.
   - Participantka najprv skúšala zobraziť tento zoznam pomocou filtrovania reklamných plôch. Keďže takúto možnosť filtrovania nenašla, vrátila sa do zoznamu reklamných kampaní, kde sa jej podarilo splniť zadanie kliknutím na zvolenú kampaň.

- Scenár 3

1. Ste prihlásený/á v aplikácii.
2. Nájdite zoznam všetkých reklamných plôch.
   - Prebehlo bez problémov.
3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
   - Prebehlo bez problémov.
4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
   - Participantka najprv skúšala zobraziť tento zoznam pomocou filtrovania reklamných plôch, kde našla možnosť filtrovania pomocou názvu. Po vysvetlení, že reťazec sa môže vyskytovať vo viacerých názvoch, sa vydala správnou cestou vyhľadávania podľa reťazca.
5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
   - Participantka bez problémov našla možnosť filtrácie podľa stavu plôch, avšak dialóg filtrovania bez uloženia zmien ihneď zavrela. Po upozornení na neuložené parametre filtrovania mala problém nájsť potvrdzovacie tlačidlo vo forme ikonky ✓v spodnej časti dialógu. Toto tlačidlo sa jej podarilo nájsť až po usmernení.
6. Zistite počet nájdených plôch.
   - Participantka chcela najprv zistiť tento počet pomocou čísla v názve plôch, po chvíli si však všimla informáciu uvedenú nad zoznamom plôch.
7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.

  - Prebehlo bez problémov.

- Scenár 4

  1. Ste prihlásený/á v aplikácii.
  2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
     - Prebehlo bez problémov.
  3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.
     - Participantka chvíľu hľadala reklamné nosiče, následne to však prebehlo bez problémov.
  4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
     - Prebehlo bez problémov.
  5. Vytvorte novú fotografiu tohto nosiča.
     - Prebehlo bez problémov, avšak participantka vyhodnotila odkaz na galériu nosiča ako možnosť pridania už vytvorenej fotografie uloženej v mobilnom zariadení.
  6. Zobrazte všetky fotografie tohto nosiča.
     - Prebehlo bez problémov.
  7. Vymažte ľubovoľnú fotografiu tohto nosiča.
     - Prebehlo bez problémov.
  8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
     - Participantka chvíľu hľadala sekciu s poznámkami, následne to však prebehlo bez problémov.
  9. Aplikujte vykonané zmeny danej reklamnej plochy.
     - Prebehlo bez problémov.
  10. Odhláste sa z aplikácie.
      - Prebehlo bez problémov.

- Doplňujúce otázky

  1. Ako sa vám páčila aplikácia?
     - Aplikácia sa jej páčila.
  2. Zdalo sa vám používanie dostatočne intuitívne?
     - Áno, používanie sa jej zdalo dostatočne intuitívne.
  3. Zmenili by ste niečo?
     - Iba čo sa týka už zmienených problémov.

B.2 Mgr. art. Ivan Šveda

- Scenár 1

  1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
     - Prebehlo bez problémov.
  2. Prihláste sa do aplikácie.
     - Prebehlo bez problémov.
  3. Zistite, aká je vaša aktuálna verzia aplikácie.
     - Prebehlo bez problémov.
  4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
     - Prebehlo bez problémov.
  5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.
     - Prebehlo bez problémov.

- Scenár 2

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných kampaní.
     - Prebehlo bez problémov.
  3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.
     - Participant si najprv nevšimol informáciu o termíne pod názvom kampane a snažil sa ju nájsť na iných miestach. Po chvíli sa však vrátil na zoznam kampaní a informáciu o termíne ukončenia našiel.
  4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
     - Prebehlo bez problémov.
  5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.
     - Participant najprv skúšal zobraziť tento zoznam pomocou filtrovania reklamných plôch. Keďže takúto možnosť filtrovania nenašiel, vrátil sa do zoznamu reklamných kampaní, kde sa mu podarilo splniť zadanie kliknutím na zvolenú kampaň.

- Scenár 3

1. Ste prihlásený/á v aplikácii.
2. Nájdite zoznam všetkých reklamných plôch.
   - Prebehlo bez problémov.
3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
   - Prebehlo bez problémov.
4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
   - Prebehlo bez problémov.
5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
   - Participant bez problémov našiel možnosť filtrácie podľa stavu plôch, avšak chvíľu mu trvalo nájsť potvrdzovacie tlačidlo vo forme ikonky ✓v spodnej časti dialógu filtrovania.
6. Zistite počet nájdených plôch.
   - Participant chcel najprv zistiť tento počet pomocou čísla uvedeného v názve plôch, po chvíli si však všimol informáciu uvedenú nad zoznamom plôch.
7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.
   - Participant si znova nebol istý potvrdením zmien v dialógu filtrovania, inak prebehlo bez problémov.

- Scenár 4

1. Ste prihlásený/á v aplikácii.
2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
   - Prebehlo bez problémov.
3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.
   - Prebehlo bez problémov.
4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
   - Prebehlo bez problémov.
5. Vytvorte novú fotografiu tohto nosiča.
   - Prebehlo bez problémov.
6. Zobrazte všetky fotografie tohto nosiča.
   - Prebehlo bez problémov.
7. Vymažte ľubovoľnú fotografiu tohto nosiča.

   - Prebehlo bez problémov.
8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
   - Prebehlo bez problémov.
9. Aplikujte vykonané zmeny danej reklamnej plochy.
   - Participant si chvíľu nebol istý možnosťou potvrdenia pomocou tlačidla v tvare ikonky ✓v hornej časti obrazovky.
10. Odhláste sa z aplikácie.
    - Prebehlo bez problémov.

- Doplňujúce otázky

1. Ako sa vám páčila aplikácia?
   - Aplikácia sa mu páčila a pochválil dobrú štruktúrovanosť a jednoduché nájdenie informácií.
2. Zdalo sa vám používanie dostatočne intuitívne?
   - Áno, používanie sa mu zdalo dostatočne intuitívne. Dodal, že mierne problémy mu robilo nedostatočné pochopenie pojmov z danej domény.
3. Zmenili by ste niečo?
   - Iba čo sa týka už zmienených problémov.

B.3 Jaroslav Šturm

- Scenár 1

1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
   - Prebehlo bez problémov.
2. Prihláste sa do aplikácie.
   - Prebehlo bez problémov.
3. Zistite, aká je vaša aktuálna verzia aplikácie.
   - Prebehlo bez problémov.
4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
   - Prebehlo bez problémov.
5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.

  - Prebehlo bez problémov.

- Scenár 2

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných kampaní.
     - Prebehlo bez problémov.
  3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.
     - Prebehlo bez problémov.
  4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
     - Prebehlo bez problémov.
  5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.
     - Participant najprv skúšal zobraziť tento zoznam pomocou filtrovania reklamných plôch. Keďže takúto možnosť filtrovania nenašiel, vrátil sa do zoznamu reklamných kampaní, kde sa mu podarilo splniť zadanie kliknutím na zvolenú kampaň.

- Scenár 3

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných plôch.
     - Prebehlo bez problémov.
  3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
     - Prebehlo bez problémov.
  4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
     - Participant najprv skúšal zobraziť tento zoznam pomocou filtrovania reklamných plôch, kde našiel možnosť filtrovania pomocou názvu. Po chvíli sa však vydal správnou cestou vyhľadávania podľa reťazca.
  5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
     - Prebehlo bez problémov.
  6. Zistite počet nájdených plôch.
     - Prebehlo bez problémov.
  7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.

  - Prebehlo bez problémov.

- Scenár 4

  1. Ste prihlásený/á v aplikácii.
  2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
     - Prebehlo bez problémov.
  3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.
     - Prebehlo bez problémov.
  4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
     - Prebehlo bez problémov.
  5. Vytvorte novú fotografiu tohto nosiča.
     - Prebehlo bez problémov.
  6. Zobrazte všetky fotografie tohto nosiča.
     - Prebehlo bez problémov.
  7. Vymažte ľubovoľnú fotografiu tohto nosiča.
     - Prebehlo bez problémov.
  8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
     - Prebehlo bez problémov.
  9. Aplikujte vykonané zmeny danej reklamnej plochy.
     - Prebehlo bez problémov.
  10. Odhláste sa z aplikácie.
      - Prebehlo bez problémov.

- Doplňujúce otázky

  1. Ako sa vám páčila aplikácia?
     - Aplikácia sa mu páčila.
  2. Zdalo sa vám používanie dostatočne intuitívne?
     - Áno, používanie sa mu zdalo dostatočne intuitívne.
  3. Zmenili by ste niečo?
     - Iba čo sa týka už zmienených problémov.

B.4 Sofia Mandelíková

- Scenár 1

  1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
     - Prebehlo bez problémov.
  2. Prihláste sa do aplikácie.
     - Prebehlo bez problémov.
  3. Zistite, aká je vaša aktuálna verzia aplikácie.
     - Prebehlo bez problémov.
  4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
     - Prebehlo bez problémov.
  5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.
     - Prebehlo bez problémov.

- Scenár 2

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných kampaní.
     - Prebehlo bez problémov.
  3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.
     - Prebehlo bez problémov.
  4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
     - Prebehlo bez problémov.
  5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.
     - Participantka skúšala zobraziť tento zoznam pomocou filtrovania reklamných plôch.

- Scenár 3

  1. Ste prihlásený/á v aplikácii.
  2. Nájdite zoznam všetkých reklamných plôch.
     - Prebehlo bez problémov.

3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
   - Prebehlo bez problémov.
4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
   - Prebehlo bez problémov.
5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
   - Prebehlo bez problémov.
6. Zistite počet nájdených plôch.
   - Participantka chcela najprv zistiť tento počet pomocou čísla v názve plôch, po chvíli si však všimla informáciu uvedenú nad zoznamom plôch.
7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.
   - Prebehlo bez problémov.

- Scenár 4

1. Ste prihlásený/á v aplikácii.
2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
   - Prebehlo bez problémov.
3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.
   - Prebehlo bez problémov.
4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
   - Prebehlo bez problémov.
5. Vytvorte novú fotografiu tohto nosiča.
   - Prebehlo bez problémov.
6. Zobrazte všetky fotografie tohto nosiča.
   - Prebehlo bez problémov.
7. Vymažte ľubovoľnú fotografiu tohto nosiča.
   - Prebehlo bez problémov.
8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
   - Prebehlo bez problémov.
9. Aplikujte vykonané zmeny danej reklamnej plochy.

   - Prebehlo bez problémov.
10. Odhláste sa z aplikácie.
   - Prebehlo bez problémov.

- Doplňujúce otázky

1. Ako sa vám páčila aplikácia?
   - Aplikácia sa jej páčila.
2. Zdalo sa vám používanie dostatočne intuitívne?
   - Áno, používanie sa jej zdalo dostatočne intuitívne.
3. Zmenili by ste niečo?
   - Iba čo sa týka už zmienených problémov.

B.5 Mgr. Richard Rác

- Scenár 1

1. Účet v aplikácii máte založený. Poznáte svoje prihlasovacie meno, ale zabudli ste heslo. Obnovte si ho.
   - Prebehlo bez problémov.
2. Prihláste sa do aplikácie.
   - Prebehlo bez problémov.
3. Zistite, aká je vaša aktuálna verzia aplikácie.
   - Prebehlo bez problémov.
4. Zistite, koľko času ubehne do najbližšej synchronizácie dát aplikácie.
   - Prebehlo bez problémov.
5. Nastavte si automatickú synchronizáciu dát aplikácie iba v prípade, ak je vaše zariadenie pripojené cez Wi-Fi.
   - Prebehlo bez problémov.

- Scenár 2

1. Ste prihlásený/á v aplikácii.
2. Nájdite zoznam všetkých reklamných kampaní.
   - Prebehlo bez problémov.
3. Zistite predpokladaný termín ukončenia kampane s názvom „Kampaň 6“.

   - Prebehlo bez problémov.
4. Označte ľubovoľnú kampaň ako ukončenú alebo neukončenú.
   - Prebehlo bez problémov.
5. Nájdite zoznam všetkých reklamných plôch prislúchajúcich zvolenej reklamnej kampani.
   - Participant hľadal riešenie na rôznych miestach. Po nejakom čase sa však vrátil do zoznamu reklamných kampaní, kde sa mu podarilo splniť zadanie kliknutím na zvolenú kampaň.

- Scenár 3

1. Ste prihlásený/á v aplikácii.
2. Nájdite zoznam všetkých reklamných plôch.
   - Prebehlo bez problémov.
3. Zistite, v akom meste sa nachádza plocha s názvom „Plocha 2“.
   - Prebehlo bez problémov.
4. Vyhľadajte zoznam všetkých plôch, v ktorých názve sa vyskytuje textový reťazec „loc“.
   - Participant najprv skúšal zobraziť tento zoznam pomocou filtrovania reklamných plôch, kde našiel možnosť filtrovania pomocou názvu. Po chvíli sa však vydal správnou cestou vyhľadávania podľa reťazca.
5. Z tohto zoznamu vyhľadajte všetky plochy, ktoré sú v stave „v procesu“.
   - Prebehlo bez problémov.
6. Zistite počet nájdených plôch.
   - Prebehlo bez problémov.
7. Zrušte všetky aplikované obmedzenia na zoznam plôch tak, aby boli znovu zobrazené všetky plochy.
   - Prebehlo bez problémov.

- Scenár 4

1. Ste prihlásený/á v aplikácii.
2. Zobrazte detail reklamnej plochy s názvom „Plocha 2“.
   - Prebehlo bez problémov.
3. Vyhľadajte všetky reklamné nosiče prislúchajúce danej ploche, v ktorých názve sa vyskytuje textový reťazec „osi“.

   - Prebehlo bez problémov.
4. Zistite počet vytvorených fotografií nosiča s názvom „Nosič 1“.
   - Prebehlo bez problémov.
5. Vytvorte novú fotografiu tohto nosiča.
   - Prebehlo bez problémov.
6. Zobrazte všetky fotografie tohto nosiča.
   - Prebehlo bez problémov.
7. Vymažte ľubovoľnú fotografiu tohto nosiča.
   - Prebehlo bez problémov.
8. K zvolenej ploche pridajte poznámku o tom, že predajňa je trvale uzavretá. K poznámke tiež pridajte ľubovoľný textový opis.
   - Prebehlo bez problémov.
9. Aplikujte vykonané zmeny danej reklamnej plochy.
   - Prebehlo bez problémov.
10. Odhláste sa z aplikácie.
    - Prebehlo bez problémov.

- Doplňujúce otázky

1. Ako sa vám páčila aplikácia?
   - Aplikácia sa mu páčila.
2. Zdalo sa vám používanie dostatočne intuitívne?
   - Áno, používanie sa mu zdalo dostatočne intuitívne.
3. Zmenili by ste niečo?
   - a. Tlačidlo na aplikovanie vykonaných zmien v detaile plochy označiť textom „Uložiť“ namiesto ikonky ✓.

Dodatok C

Obsah priloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `dp.zip .2 thesis` ..... zdrojová forma práce vo formáte LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce vo formáte PDF
  - `thesis.ps` ..... text práce vo formáte PS