UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Mario Karimović

Mobilna aplikacija za podporo CRM procesov

DIPLOMSKO DELO
VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: doc. dr. Rok Rupnik

Ljubljana, 2012

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00272/2012
Datum: 11.04.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MARIO KARIMOVIĆ**

Naslov: **MOBILNA APLIKACIJA ZA PODPORO CRM PROCESOV**
MOBILE APPLICATION TO SUPPORT CRM PROCESSES

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Zasnujte funkcionalnosti mobilne aplikacije za podporo izvajanju CRM procesov v stanju mobilnosti. Mobilno aplikacijo nato za Android platformo razvijte s pomočjo uporabe ogrodja PhoneGap. Posebno pozornost posvetite učinkovitemu uporabniškemu vmesniku.

Mentor:

doc. dr. Rok Rupnik

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani Karimović Mario,

z vpisno številko 63080279,

sem avtor diplomskega dela z naslovom:

Mobilna aplikacija za podporo CRM procesov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom doc. dr. Rok Rupnik
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne Podpis avtorja:

Zahvala

Za pomoč pri izdelavi diplomske naloge se zahvaljujem mentorju doc. dr. Roku Rupniku.

Zahvaljujem se staršem in sestri Marini za omogočen študij in podporo. Izpostavil bi tudi prijatelja Alana in se mu zahvalil za vse nasvete in pomoč v času študija. Zahvala gre tudi Elizabeti, ki mi je polepšala študijske dneve.

Kazalo

Seznam uporabljenih kratic

API Aplication Programming interface; programski vmesnik.

CSS Cascading Style Sheets; slogovni jezik za oblikovanje spletnih strani.

GPS Global Positioning System; sistem globalnega določanja lege.

HTML Hyper Text Markup Language; označevalni jezik za oblikovanje spletnih strani.

IDE Integrated Developmnet Kit; integrirano razvojno okolje.

iOS iPhone Operating System; prenosni operacijski sistem podjetja Apple.

JS JavaScript; objektni skriptni programski jezik.

MMS Multimedia Messaging System; multimedijski sporočilni sistem.

OS Operating System; operacijski sistem.

PDA Personal Digital Assistant; osebni elektronski pomočnik.

PDF Portable Document Format; odprt standard za izmenjavo elektronskih dokumentov.

SDK Software Development Kit; paket programskih orodjih, za razvoj aplikacij za posamezno programsko opremo.

SMS Short Message Service; sistem za pošiljanje kratkih besedilnih sporočil.

SQL Structured Query Language; strukturni povpraševalni jezik za delo s podatkovnimi zbirkami.

UML Unified Modelling Language; Univerzalni modelirni jezik za specifikacijo, vizualizacijo, konstrukcijo in dokumentacijo izdelkov v okviru objektnega razvoja informacijskih rešitev.

W3C World Wide Web Consortium; mednarodni inštitut, kjer člani organizacije, osebje s polnim delovnim časom in javnost sodelujejo te skupaj razvijajo standarde za splet.

WiFi tehnologija brezžičnega prenosa podatkov preko računalniškega omrežja.

Povzetek

Dandanes je mobilni telefon nujno in nepogrešljivo »orodje« sodobnega človeka. Postal je zvesti spremljevalec in osebni pomočnik skoraj vsake osebe. Zato smo se odločili, da za diplomsko delo izdelamo mobilno aplikacijo. Ta bo poenostavila in olajšala delo tržnikom, komercialistom in vsem tistim, ki za svoje delo potrebujejo kar se da enostavno in hitro dostopnost do podatkov o svojih strankah in zgodovini poslovanja z njimi. S pomočjo aplikacije bodo imeli tudi na terenu v vsakem trenutku pregled nad dosedanjo komunikacijo s strankami in opravili, ki jih morajo v določenem terminu opraviti.

V diplomskem delu je opisan razvoj mobilne aplikacije za platformo Android, s pomočjo spletnih tehnologij in ogrodja PhoneGap. Na začetku so opisane uporabljene tehnologije, ki so bile uporabljene za uspešno izvedbo mobilne aplikacije. Sledi predstavitev kaj je CRM oziroma sistem za upravljanje odnosov s strankami in njegov pomen za uspešno in boljše delovanje podjetja.

V osrednjem delu je predstavljeno ogrodje PhoneGap, ki temelji na uporabi označevalnega jezika HTML, stilnih predlog CSS in objektnega skriptnega jezika JavaScript. PhoneGap je medplatformsko ogrodje, ki izvorno kodo prevede v domorodne aplikacije za več operacijskih sistemov.

V nadaljevanju je opisan razvoj aplikacije od zajema zahtev, načrtovanja, izvedbe in same uporabe. Aplikacija je bila razvita s spletnimi tehnologijami in s pomočjo ogrodja PhoneGap smo to prevedli v aplikacijo za izvajanje na mobilni napravi z Android platformo. Skozi uporabniški vmesnik je prikazana in opisana uporaba aplikacije.

V zaključku so predstavljene sklepne ugotovitve in ideje za nadaljnji razvoj, ki se nanašajo predvsem na širitev funkcionalnosti aplikacije.

Ključne besede:

Mobilna aplikacija, CRM, PhoneGap

Abstract

Nowadays a mobile phone is an essential »instrument« of the modern man. It has become a loyal companion and a personal assistant of almost every person. That is why we chose to create a mobile application for our diploma thesis. This application will facilitate work for marketers, commercialists and all the people that require a simple and fast access to their clients' data and business history for their work. With the help of this application they will have on overview of all current customer communication at any moment, even outside of office, and also the tasks they are required to complete in a certain time frame.

In the thesis we described the development of a mobile application for Android platform using web technologies and PhoneGap framework. We start by describing the technologies that were used for a successful implementation of the application. Following is a description of CRM (customer relationship management) and its significance for a better and successful operation of a company.

The central part contains a presentation of PhoneGap, which is based on HTML, CSS and JavaScript. PhoneGap is a cross-platform framework that translates source code into native applications for several operating systems.

Furthermore we describe the development of the application from the identification of requirements, planning, realization and the use itself. It was developed using web technologies and translated via PhoneGap framework into an application for a mobile device running on the Android platform. The use of the app is presented through the user interface.

The conclusion contains our findings and ideas for further development, which refer mostly to expanding the functionality of the application.

Keywords:

Mobile application, CRM, PhoneGap

1. Uvod

Delavec na terenu oziroma tržnik se srečuje z velikimi težavami, ki jih je potrebno bodisi odpraviti ali pa vsaj zmanjšati, za lažje, uspešnejše, boljše poslovanje in zadovoljstvo vseh vpletenih v prodajni proces. Če tržnik na terenu nima potrebnih podatkov, ki so mu nujno potrebna v tistem trenutku in če ne opravi svojega dela v določenem času, to lahko pomeni izgubo za podjetje in posledično tudi škodo ali izgubo stranke. Tržniku na terenu je potrebno omogočiti dostop do vseh potrebnih podatkov. Ta mora dostopati do osebnih podatkov in kontaktnih podatkov stranke, da lažje in učinkoviteje komunicira z njo. Pregled oziroma sledljivost vseh dogodkov oziroma aktivnosti za posamezno stranko. Tako lahko spremlja zgodovino komunikacije s stranko, (klic, elektronska sporočila, ponudbe, pogodbe, naročila in sestanek). Pregled nad prodajnimi priložnostmi in spremljanje uspešnosti le teh je zelo pomembno. To, da lahko tržnik vidi neuspešno zaključene priložnosti iz preteklosti je veliko vredno, saj se iz teh lahko največ nauči in v prihodnosti ne ponovi morebitnih napak iz preteklosti. Skratka tržniku je mobilna aplikacija ne samo v oporo, da je bližje stranki temveč tudi opora pri samem delu. Omogoča mu pregled nad opravili, ki jih mora opraviti in dodajanje novih opravil, ki jih bo opravil v prihodnosti. Z aplikacijo se zmanjša število nepravočasno opravljenih nalog, saj mu aplikacija omogoča enostaven pregled in nadzor nad potekom dela. Z današnjo tehnologijo je enostavno omogočiti podporo tržnikom izven pisarne.

Pametni mobilni telefoni so zvesti spremljevalci skoraj vsake osebe. Posledično je razvoj pametnih mobilnih telefonov v velikem razmahu. Konkurenca je velika in vsi se trudijo omogočiti nove lastnosti oziroma funkcionalnosti, ki bi navdušile uporabnike. Nekoč so mobilne naprave omogočale opravljanje klicev, nato pošiljanje kratkih sporočil SMS. Vse to se je nadgradilo s pošiljanjem slik, zvoka in video posnetkov prek sporočil MMS. Danes pametni mobilni telefoni omogočajo dostop do spleta, elektronske pošte, nadomeščajo glasbene predvajalnike, uporabljamo jih kot fotoaparate, navigacijske naprave in še veliko več. Pametni mobilni telefoni so že pred časom prerasli pomen besede telefon in dosegli funkcionalnosti malega računalnika v žepu.

Ker je trg mobilnih naprav zelo velik, se je pojavilo precejšne število različnih operacijskih sistemov.

Tabela 1.1 prikazuje tržni delež operacijskih sistemov za mobilne naprave v drugem četrtletju leta 2012:

Operacijski sistemi	**2Q12 število**	**Delež 2Q12 %**	**2Q11 število**	**Delež 2Q11 %**
Android	98,529.3	64.1	46,775.9	43.4
iOS	28,935.0	18.8	19,628.8	18.2
Symbian	9,071.5	5.9	23,853.2	22.1
RIM	7,991.2	5.2	12,652.3	11.7
Bada	4,208.8	2.7	2,055.8	1.9
Microsoft	4,087.0	2.7	1,723.8	1.6
Ostali	863.3	0.6	1,050.6	1.0
Skupno	**153,686.1**	**100.0**	**107,740.4**	**100.0**

Tabela 1.1: Svetovna prodaja pametnih telefonov po operacijskem sistemu v 2. četrtletju 2012 (v tisočih); Vir: Gartner (Avgust 2012) [2].

Najbolj zastopani operacijski sistemi v pametnih telefonih so Android, iPhone OS, BlackBerry OS, Windows Mobile, Bada in Symbian.

Zaradi velikega števila uporabnikov pametnih mobilnih telefonov, se tudi razvijalci mobilnih aplikacij zavedajo možnosti in želijo prodreti na trg z različnimi aplikacijami, ki so plačljive ali zastonjske. Težava se pojavi, ko je potrebno razviti aplikacijo, ki bi se izvajala na različnih operacijskih sistemih. Razvijanje aplikacij za tako veliko število operacijskih sistemov je vse prej kot lahko delo in terja veliko investicijo. Za vsako platformo je drugačen razvoj in razvijalci morajo biti seznanjeni z vsakim razvojem posebej. Zato se je pojavilo ogrodje za medplatformski razvoj, ki se drži načela »Razvij enkrat prevedi večkrat«. Z medplatformskim ogrodjem je možno izvorno kodo prevesti v domorodne aplikacije za več operacijskih sistemov.

2. Uporabljene tehnologije

Opis tehnologij, ki so bile uporabljene za uspešno izvedbo mobilne aplikacije.

2.1. Pametni telefon

Pametni telefoni [3] so kombinacija osebnega digitalnega pomočnik (PDA) [4] in mobilne naprave. V začetku je bil glavni cilj mobilnih naprav vzpostaviti povezavo med dvema telefonoma in s tem olajšati medsebojno komunikacijo oseb na različnih lokacijah. Danes s pomočjo tega poleg opravljanja klicev in pošiljanja sms sporočil, predvajamo in zajemamo avdio in vizualne vsebine, uporabljamo GPS navigacijo, dostopamo do elektronske pošte, brskamo po spletu in ne samo, da prikazuje spletne strani optimizirane za mobilne naprave, prikazuje tudi navadne spletne strani, ki niso optimizirane v enaki kvaliteti kot to počne brskalnik nameščen na namiznem računalniku. Seveda je na pametnem telefonu manjši ekran. To je lahko edina slabša izkušnja za uporabnika v primerjavi z brskanjem po spletu na namiznem oziroma prenosnem računalniku. Velika prednost uporabe pametnega telefona je tudi ta, da lahko uporabniki sami nameščajo in uporabljajo zahtevne aplikacije in tako še dodatno razširijo funkcionalnosti telefona.

Današnji pametni telefoni se lahko pohvalijo z zmogljivimi procesorji, brezžično povezavo, senzorji, podporo različnim dokumentom (Pdf, Word, Excel in drugi), elektronski pošti. Dodatne prednosti so medijski predvajalnik, digitalna kamera, fotoaparat in vedno večji zasloni, ki so občutljivi na dotik. S tem, je uporabniška izkušnja vedno boljša. Lahko bi rekli, da je pametni telefon mali računalnik v žepu z dodanimi funkcijami mobilnega telefona.

Največji proizvajalci pametnih telefonov so podjetja HTC, Samsung, Apple, Nokia, Motorola, LG, ZTE, RIM, Huawei Device in TCL Communications.

Tabela 2.1 prikazuje tržni delež prodaje mobilnih telefonov v drugem četrtletju leta 2012:

Podjetje	**2Q12 število**	**Delež 2Q12 %**	**2Q11 Število**	**Delež 2Q11 %**
Samsung	90,432.1	21.6	69,827.6	16.3
Nokia	83,420.1	19.9	98,869.3	22.8
Apple	28,935.0	6.9	19,628.8	4.6
ZTE	17,936.4	4.3	13,070.2	3.0
LG Electronics	14,345.4	3.4	24,420.8	5.7
Huawei Device	10,894.2	2.6	9,026.1	2.1
TCL Communications	9,355.7	2.2	7,938.9	1.9
HTC	9,301.2	2.2	11,016.1	2.6
Motorola	9,163.2	2.2	10,221.4	2.4
RIM	7,991.2	1.9	12,652.3	3.0
Ostali	137,266.4	32.8	152,989.70	35.7
Skupaj	**419,007.90**	**100.0**	**428,661.15**	**100.0**

Tabela 2.1: Svetovna prodaja pametnih telefonov po proizvajalcih v 2. četrtletju 2012 (v tisočih); Vir: Gartner (Avgust 2012) [2].

2.2. HTML

HTML (HyperText Markup Language) je označevalni jezik za prikazovanje spletnih strani in njenih vsebin, ki jih je mogoče prikazati v spletnem brskalniku [5]. Ta predstavlja osnovo spletnega dokumenta in je napisan v obliki HTML elementov. Vsak element je najpogosteje sestavljen is treh delov in sicer iz začetne značke, vsebine in končne značke. Začetna in končna značka pričneta z lomljenim oklepajem <, končata z >, vmes pa je zapisano ime elementa. Končna značke se obvezno zaključi tako, da je pred imenom elementa zapisan še znak /. Primer začetne značke ( ) in končne značke ( ). Poleg poljubne vsebine, se lahko znotraj para značk vnese še druge značke, to imenujemo gnezdenje značk.

Označevalni jezik, poleg prikaza HTML dokumenta v brskalniku, določa strukturo in semantični pomen tega. HTML dokument je znakovna datoteka, kar pomeni, da jo lahko odpremo in urejamo s poljubnim urejevalnikom besedil.

2.3. CSS

CSS (Cascading Style Sheets) je stilna podloga, predstavljena v obliki preprostega slogovnega jezika, ki skrbi za predstavitev spletnih strani v brskalniku [6]. Z njim definiramo stil HTML oziroma XHTML elementov v obliki pravil, kako naj se ti prikažejo na spletni strani. Določamo lahko različne lastnosti elementa, kot so barva, velikost, odmik, poravnava, obroba, pozicija in še veliko drugih atributov. Poleg tega lahko določamo oziroma nadziramo aktivnosti, ki jih uporabnik nad elementi na spletni strani izvaja, kot je npr. prekritje povezave z miško.

Bistvo uporabe stilne podloge je poleg definiranja pravil, pred vsem ločitev strukture strani, ki jo podaja označevalni jezik skupaj z vsebino, od njene predstavitve. S tem omogočimo lažje urejanje in dodajanje stilov ter poskrbimo za večjo preglednost dokumentov, ki temeljijo na HTML sintaksi. Z uporabo tega se prav tako zmanjša ponavljanje kode, saj omogočimo množici strani uporabo istih podlog.

Specifikacija in vzdrževanje CSS je v rokah skupine CSS Working Group znotraj organizacije Word Wide Web Consoritum (W3C). Specifikacijo uradno potrdijo člani W3C in tako nastanejo CSS priporočila. Ki pa niso omejena samo na internetne brskalnike, uporabljajo se tudi v programih za pisanje dokumentov in elektronskih preglednic kakor tudi na napravah, kot so mobilni telefoni.

2.4. JavaScript

Javascript [7] je objektni skriptni programski jezik, ki ga je razvilo računalniško podjetje Netscape, da bi spletnim programerjem pomagal pri ustvarjanju interaktivnih spletnih strani. Razvit je bil neodvisno od Jave, vendar si z njo deli številne lastnosti in strukturo. JavaScript se lahko uporabi v kombinaciji s HTML kodo in s tem poživi stran z dinamičnim izvajanjem. Velika programerska podjetja podpirajo JavaScript in kot odprt jezik ga lahko uporablja vsakdo, ne da bi pri tem potreboval licenco. Sintaksa jezika JavaScript ohlapno sledi programskemu jeziku C in prav tako kot C, JavaScript nima vgrajenih vhodno izhodnih funkcij, zato je njihova izvedba odvisna od gostitelja.

JavaScript se veliko uporablja za ustvarjanje dinamičnih spletnih strani. Podpirajo ga vsi novejši brskalniki. Programsko kodo se vgradi ali pa vključi v HTML, da opravlja naloge, ki niso mogoče samo s statično spletno stranjo. Tako lahko npr. preverja pravilnost vnesenih podatkov, odpira nova okna, opravlja izračune ipd. Na žalost se lahko pojavi težava, saj različni brskalniki izpostavijo različne objekte za uporabo in za podporo vseh brskalnikov je zato potrebno napisati več različic funkcij.

JavaScript se prav tako uporablja izven spletnih strani v različnih orodjih, na primer v datotekah PDV, prav tako ga podpirata tudi operacijska sistema Microsoft Windows in Mac OS X.

Programi imajo svoje objektne modele, ki zagotavljajo dostop do gostiteljevega okolja, samo jedro jezika pa je v vseh programih večinoma enako.

3. CRM

CRM oziroma upravljanje odnosov s strankami je strategija v podjetju, ki pozornost usmerja na odnose s strankami in ne toliko na produkte [8]. Tako se osredotoča na stranke, podatke o stranki, komunikacije s stranko ipd. Ključni cilj CRM sistema je podjetju oziroma zaposlenim zagotoviti celovit pogled nad vsemi podatki o stranki. Bistvo vsakega podjetja je pridobiti stranke, jih obdržati in posledično povečati prihodke in dobičke.

Dva ključna sklopa CRM sistema, ki smo ju tudi v naši mobilni aplikaciji implementirali sta hramba podatkov in prodaja. Tu so podatki o podjetjih in osebah oziroma zaposlenih v podjetjih. Med te podatke sodijo tudi podatki podjetij in oseb, ki še niso stranke, bi pa to utegnile postati. Prav tako se beležijo zapiski telefonskih klicev, elektronska sporočila in sestanki, skratka celotna komunikacija z obstoječimi in potencialnimi strankami. V drugem sklopu, ki se ukvarja s prodajo se beležijo vse poslovne priložnosti bodisi uspešne ali neuspešne. V vsakem trenutku se lahko dostopa do aktualnih ali že zaključenih priložnosti, se vidi vrednosti teh in v kakšni fazi se te nahajajo ali pa v kateri fazi so propadla.

Mobilni CRM zajame podporo trženju, prodaji, po-prodajnim aktivnostim in servisiranju oziroma skrbi za stranke. Praktično predstavlja pisarno integrirano v mobilni napravi, ki uporabnika zalaga z vsemi potrebnimi informacijami in podatki o določeni stranki ali prodaji. Mobilni CRM potrebujejo delavci na terenu, med sestanki in doma. Z razvojem vse bolj naprednih telefonov in dostopom do interneta praktično povsod, je to postala realnost, ki jo je potrebno čim prej sprejeti.

4. PhoneGap

Na trgu je vse večja izbira pametnih telefonov in kot uporabniki smo lahko vsi navdušeni nad tem. Vendar se za razvijalce mobilnih aplikacij tu pojavi težava, saj je izbira platforme vse prej kot lahka. Razvijalcem tako lahko platforme pomagajo pri razvoju aplikacij za več operacijskih sistemov spletne tehnologije in ogrodja kot je PhoneGap. Ogrodje (angl. framework) PhoneGap predstavlja pomemben korak k združitvi na eni strani hitro spreminjajočega se mobilnega trga in na drugi strani številne razvijalce s poznavanjem spletnih tehnologij.

Naloga PhoneGap pri razvoju mobilni aplikacij za več platform je pretvoriti spletne tehnologije (slika 4.1) kot so HTML, CSS in JavaScript v domorodne aplikacije, ki se lahko izvajajo na različnih platformah [1].

Slika 4.1: Spletne tehnologije podprte v ogrodju PhoneGap [10].

PhoneGap je odprtokodna rešitev, ki vključuje odprte standarde in ostaja odprt in neodvisen. Zaradi odprtosti platforme veliko izboljšav prispeva kar zunanja skupnost PhoneGap razvijalcev. Uporabljajo ga lahko razvijalci za razvoj mobilnih aplikacij vseh vrst, ki so lahko plačljive, proste plačila ali odprtokodne. Med uporabniki platforme so tudi velika podjetja, kot so IBM, Alcatel-Lucent, Cisco in Logitech [9].

Aplikacije razvite s spletnimi tehnologijami so hibridne, saj so sestavljene iz domorodnega brskalnika in spletne aplikacije, ki imajo dostop do virov naprav. Različna okna aplikacije so pravzaprav različne spletne strani.

PoneGap (slika 4.2) ni edino orodje, ki omogoča razvoj mobilnih aplikacij za delovanje na več operacijskih sistemih. Alternative so še RhoMobile, Appcelerator, Red Foundry, Corona, appMobi, DragonRAD, Sencha Touch in drugi.

Slika 4.2: PhoneGap logotip [12].

4.1. Prednosti in slabosti

Prednosti razvoja hibridnih aplikacij (angl. hybrid app):

- Hiter razvoj aplikacij za več platform z uporabo spletnih tehnologij HTML, CSS in JavaScript.
- Za razvoj aplikacijo ni potrebno programirati v izvornem jeziku, potrebno je kreirati le osnovni projekt za vsako platformo posebej.
- Če je potrebno aplikaciji v kakšnem primeru zagotoviti popolno izvorno podporo, je to možno z uporabo izvornega razvoja.
- Aplikacije lahko dodatno obogatimo s funkcionalnostmi naprave (GPS, kamera, …).
- Aplikacije lahko ponudimo na spletni trgovini mobilnih aplikacij, saj so te zavite v izvorni razvoj.
- Pri izdelavi izgleda aplikacije so na voljo številne knjižnice, ki poenostavijo in pospešijo razvoj.
- Stroški razvoja aplikacije so cenejši v primerjavi z izvornim razvojem, saj ni potrebno za vsako platformo ponovno razvijati aplikacijo v izvornem jeziku platforme. Potrebne so le prilagoditve za posamezne platforme.

Slabosti razvoja hibridnih aplikacij:

- Za vsako platformo posebej je potrebno namestiti razvojno okolje.
- Ne podpirajo vseh funkcionalnosti mobilnih naprav (v prihodnosti naj bi se to odpravilo).
- Odzivnost aplikacij v primerjavi z izvornimi aplikacijami je slabša, saj je do izvornih kontrol mobilne naprave potreben prehod skozi API, ki upočasni delovanje.
- Razvite dele aplikacije na izvoren način je potrebno prilagoditi izvornemu jeziku vsaki platformi posebej.
- Pri razvoju so razvijalci omejeni, ker brskalniki ne omogočajo kompleksnejših interakcij z uporabnikom.

4.2. Zgodovina

PhoneGap se je prvič pojavil leta 2008 na konferenci iPhoneDevCamp v San Franciscu. Glavni razlog, da so začeli razmišljati v tej smeri je ta, da se vsi programerji za razvoj aplikaciji za iPhone srečajo z objektnim programskim jezikom C. Ker je večje število spletnih programerjev kot tistih, ki uporabljajo objektni programski jezik C, so si postavili vprašanje. Ali bi lahko izdelali ogrodje (angl. framework), ki bi omogočal spletnim programerjem, da bi svoje znanje spletnih tehnologij HTML, CSS in JavaScript uporabili in izdelali mobilno aplikacijo za iPhone in dostopali do sistemskih virov naprav kot je kamera ali kompas?
Podjetje Nitobi je začelo z razvojem ogrodja PhoneGap in ustanovilo skupnost, v kateri so si vsa podjetja prizadevala iti v smeri odprtosti, preglednosti in sodelovanja. Ogrodje je med drugim podprlo tudi podjetje Apple in ga kljub spremenjeni različici 4.0 licenčne pogodbe še danes podpira [11].

Aprila leta 2011 je podjetje Adobe Systems prevzelo podjetje Nitobi in nadaljevalo z razvojem za večino mobilnih platform. Projekt so marca leta 2012 pridružili odprtokodni skupnosti Apache Software Foundation pod imenom Cordova.

Od samega začetka število aplikacij razvitih na ta način strmo narašča. Po uradnih podatkih podjetja Nitobi je bila koda prenesena več kot 600.000-krat. Po neuradnih podatkih na spletni strani PhoneGap navajajo, da je mesečno v povprečju 100.00 prenosov.

4.3. Podprti operacijski sistemi

PhoneGap trenutno podpira razvoj mobilnih aplikacij za operacijske sisteme Apple iOS, Google Android, HP webOS, Microsoft Windows Phone, Nokia Symbian OS, RIM BlackBerry in Samsung Electronics Bada (slika 4.3).

Slika 4.3: Logotipi mobilnih operacijskih sistemov podprtih v ogrodju PhoneGap [13].

4.4. Dostopnost sistemskih virov mobilnih naprav

PhoneGap omogoča dostop do različnih funkcionalnosti mobilnega telefona oziroma naprave [15]. Iz slike 4.4 je razvidno, da pri nekaterih operacijskih sistemih ni omogočen dostop do sistemskih virov naprav. V prihodnosti naj bi se to odpravilo, se še razširilo in tako omogočilo razvijalcem, da še dodatno obogatijo mobilne aplikacije.

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 5.x	Blackberry OS 6.0+	WebOS	Windows Phone 7	Symbian	Bada
Merilnik pospeška	✓	✓	✓	✓	✓	✓	✓	✓	✓
Kamera	✓	✓	✓	✓	✓	✓	✓	✓	✓
Kompas	x	✓	✓	x	x	✓	✓	x	✓
Stiki	✓	✓	✓	✓	✓	x	✓	✓	✓
Datotečni sistemi	✓	✓	✓	✓	✓	x	✓	x	x
Geografska lokacija	✓	✓	✓	✓	✓	✓	✓	✓	✓
Večpredstavnost	✓	✓	✓	x	x	x	✓	x	x
Omrežje	✓	✓	✓	✓	✓	✓	✓	✓	✓
Obvestilo (opozorilo)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Obvestilo (zvok)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Obvestilo (vibracija)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Podatkovno skladišče	✓	✓	✓	✓	✓	✓	✓	✓	x

Slika 4.4: Dostopnost sistemskih virov naprav v ogrodju PhoneGap [14].

4.4.1. Merilnik pospeška

Merilnik pospeška (angl. accelerometer) omogoča zaznavanje gibanja naprave v x, y in z smeri.
Zaznavanje gibanja pomeni, da naprava zazna vsak premik, zato se lahko ta objekt uporabi v različne namene. Eden takšnih primerov bi bil, da se kamera sproži šele takrat, ko je naprava v popolnem mirovanju.

Objekt **accelerometer** vsebuje metode:

- accelerometer.**geCurrentAcceleration**

Metoda sproži senzor gibanja, ki zazna spremembo v gibanju po vseh treh koordinatah (x, y, z) glede na začetno in končno stanje naprave. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

- accelerometer.**watchAcceleration**

Metoda vrača spremembo položaja v intervalu po vseh treh koordinatah (x,y,z). Časovni interval se poda preko parametra freuqency v objektu acceleratorOptions. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

- accelerometer.**clearWatch**

Metoda ustavi spremljanje gibanja naprave tako, da poda enolično referenčno identifikacijo watchID, ki jo kreira accelerometer.watchAcceleration. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

4.4.2. Kamera

Kamera (angl. camera) omogoča dostop do programske opreme za uporabo kamere na napravi.

Objekt **camera** vsebuje metodo:

- camera.**getPicture**

Metodo uporabimo, ko želimo prikazati fotografijo iz galerije oziroma posneti fotografijo. Privzeta nastavitev omogoča, da ima uporabnik možnost posneti fotografijo, ali pa si lahko ogleda fotografijo iz galerije s klikom nanjo. Ko uporabnik zajame fotografijo ali konča ogled, se operacija kamera zaključi in obnovi prejšnja aplikacija od koder je bil izveden klic.

Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango), Bada (1.2) in webOS.

- camera.**cleanup**

Metoda odstrani vse zajete fotografije, ki so bile zajete s kamero in so shranjene na začasni lokaciji za shranjevanje (angl. temporary storage location). Podprta je na platformi iOS.

4.4.3. Avdio in vizualne vsebine

Avdio in vizualne vsebine (angl. capture) omogočajo dostop do zvočnih, slikovnih in video vsebin na napravi.

Objekt **capture** vsebuje metode:

- capture.**captureAudio**

Metoda kliče programsko opremo za zajemanje zvočnega posnetka na napravi. Privzeto je nastavljeno, da ko se zajame 1 posnetek se snemanje zaključi. Lahko se nastavi limit parameter v objektu CaptureAudioOptions in se v eni seji zajame več posnetkov. Ko se doseže določeno število posnetkov, se snemanje zaključi. Nato se sproži CaptureCB, ki vrne polje objektov MediaFile, ki opisuje vsak posnet avdio posnetek. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS in Windows Phone 7 (Mango).

- capture.**captureImage**

Metoda kliče programsko opremo za upravljanje kamere na napravi. Omogoča zajem več fotografij v eni seji, privzeto je nastavljeno, da zajame eno fotografijo. Koliko slik naj zajame, se nastavi v objektu CaptureImageOptions tako, da se poda parameter limit. Ko zajame fotografijo oziroma fotografije se sproži CaptureCB, ki vrne polja objektov MediaFile, ki opisuje vse zajete fotografije. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango) in Bada (2.x).

- capture.**captureVideo**

Metoda kliče programsko opremo za upravljanje kamere, da začne snemanje video posnetka. Omogoča zajem več video posnetkov v eni seji. Zajem video posnetkov se zaključi, ko se doseže podano število posnetkov. Privzeto je nastavljeno, da se operacija zaključi po 1 posnetku, v kolikor se želi v eni seji zajeti več posnetkov, je potrebno podati parameter limit v objektu CaptureVideoOptions. Ko se zajem video posnetkov zaključi, se sproži CaptureCB, le-ta vrne polja objektov MediaFile, ki opisuje vse zajete video posnetke. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS, Windows Phone 7 (Mango) in Bada (2.x).

- capture.**getFormatData**

Metoda vrne informacije objekta MediaFile o zajeti fotografiji, avdio posnetku ali video posnetku. Podprta je na platformah Android, BlackBerry (OS 5.0 ali višje), iOS in Windows Phone 7 (Mango).

4.4.4. Kompas

Kompas (angl. compass) omogoča zaznavanje smeri v katero kaže naprava.

Objekt **compass** vsebuje metode:

- compass.**getCurrentHeading**

Metoda zazna v katero smer kaže naprava in vrne vrednost v stopinjah od 0 do 359,99. Podatki se vrnejo preko objekta CompassHeading. Podprta je na platformah Android, iOS, Windows Phone 7 (Mango pod pogojem, da ima vgrajen kompas), Bada (1.2 in 2.x) in webOS.

- compass.**watchHeading**

Metoda v časovnem intervalu spremlja smer v katero kaže naprava. Vrednosti so v stopinjah od 0 do 359,99. Časovni interval se poda preko parametra freuqency v objektu compassOptions. Podprta je na platformah Android, iOS, Windows Phone 7 (Mango pod pogojem, da ima vgrajen kompas), Bada (1.2 in 2.x) in webOS.

- compass.**clearWatch**

Metoda ustavi zajemanje smeri naprave tako, da se poda enolično referenčno identifikacijo watchID, ki jo kreira compass.watchHeading. Podprta je na platformah Android, iOS, Windows Phone 7 (Mango pod pogojem, da ima vgrajen kompas), Bada (1.2 in 2.x) in webOS.

- compass.**watchHeadingFilter**

Metoda vrača smer v katero kaže naprava, če se ta spremeni za določeno število stopinj. Najmanjše število stopinj, ki se upošteva se poda preko parametra filter v objektu compassOptions. Vrednosti so v stopinjah od 0 do 359,99. Podprta je na platformi iOS.

- compass.**clearHeadingFilter**

Metoda ustavi zajemanje smeri naprave tako, da se poda enolično referenčno identifikacijo watchID, ki jo kreira compass.watchHeadingFilter. Podprta je na platformi iOS.

4.4.5. Omrežje

Omrežje (angl. connection) omogoča pridobivanje informacij o mobilnih in brezžičnih omrežjih. Do teh dostopamo preko vmesnika navigator.network.

Lastnost je **connection.type**, ki vrne status in tip vzpostavljene povezave.
Tip povezave:

- Conecction.**UNKNOWN** (nepoznano omrežje),
- Conecction.**ETHERNET** (ethernet omrežje),
- Conecction.**WIFI** (brezžično omrežje),
- Conecction.**CELL_2G** (2 generacija mobilnih omrežij),
- Conecction.**CELL_3G** (3 generacija mobilnih omrežij),
- Conecction.**CELL_4G** (4 generacija mobilnih omrežij),
- Conecction.**NONE** (nobena povezava ni vzpostavljena).

Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (2.x) in webOS.

4.4.6. Stiki

Stiki (angl. contacts) omogočajo dostop, urejanje in dodajanje stikov v lokalnem imeniku.

Objekt **contacts** vsebuje metode:

- contacts.**create**

Metoda vrne nov objekt Contact, ki vsebuje podatke o novem stiku, ki ga želimo dodati v lokalni imenik stikov na napravi. Ker objekt ni obstojen (angl. persistent) je potrebno za trajno hrambo izvesti še metodo Contact.save. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje) in Bada (1.2 in 2.x).

- contacts.**find**

Metoda izvede iskanje po lokalnem imeniku stikov na napravi in vrne polje objektov tipa Contact. Vrnjeni objekti tipa Contact bodo vsebovali le tiste podatke, ki bodo določeni s parametrom contactFields. V primeru, da ne vsebuje nobenih parametrov, bodo vrnjeni rezultati vsebovali le parameter id. V kolikor je vrednost parametra podan z *, je vrnjen objekt z vsemi parametri. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje) in Bada (1.2 in 2.x).

4.4.7. Lastnosti naprave

Lastnosti naprave (angl. device) omogoča dostop do informacij o programski in strojni opremi naprave.

Objekt **device** vsebuje lastnosti:

- device.**name**

Lastnost vrne naziv modela naprave. Ta se lahko razlikuje med različicami istega modela naprave. Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- device.**cordova**

Lastnost vrne verzijo izvajalnega okolja PhoneGap oziroma Cordova na napravi. Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

- device.**platform**

Lastnost vrne naziv operacijskega sistema nameščenega na napravi. Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- device.**uuid**

Lastnost vrne univerzalni enolični identifikator, ki ga določi proizvajalec naprave. Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- device.**version**

Lastnost vrne različico operacijske sistema nameščenega na napravi. Lastnost je podprta na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

4.4.8. Datotečni sistem

Datotečni sistem (angl. file) nudi objekte za delo z datotekami na datotečnem sistemu.
Objekti, ki so na voljo:

- **DirectoryEntry** (objekt predstavlja mapo v datotečnem sistemu),

- **DirectoryReader** (objekt vsebuje seznam datotek in map znotraj datotečnega sistema),
- **File** (objekt vsebuje atribute datotek),
- **FileEntry** (objekt predstavlja datoteke v datotečnem sistemu),
- **FileError** (objekt se nastavi pri napakah v katerikoli File API metodi),
- **FileReader** (objekt omogoča branje datotek),
- **FileSystem** (objekt predstavlja datotečni sistem),
- **FileTransfer** (objekt omogoča nalaganje datotek na strežnik ali prenos datotek s strežnika),
- **FileTransferErorr** (objekt se nastavi pri napakah v povezavi s prenosom),
- **FileUploadOptions** (objekt se uporablja v povezavi s FileTransfer in določa dodatne parametre pri prenosu na strežnik),
- **FileUploadResult** (objekt ko se uspešno zaključi prenos),
- **FileWriter** (objekt omogoča pisanje v datoteko),
- **Flags** (objekt se uporablja za določanje parametrov objektu DirectoryEntry),
- **LocalFileSystem** (objekt za dostopanje do korenske mape datotečnega sistema),
- **Metadata** (objekt zagotavlja informacije o stanju datotek in map).

Objekti so podprti na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje) in Windows Phone 7 (Mango).

4.4.9. Geografska lokacija

Geografska lokacija (angl. geolocation) zagotavlja inforamcije o napravi kot sta geografska širina in dolžina - koordinate GPS (angl. Global Positioning System). API je osnovan na specifikaciji W3C.

Objekt **geolaciton** vsebuje metode:

- geolocation.**getCurrentPostion**

Metoda vrne trenutno lokacijo naprave kot objekt tipa Position. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- geolocation.**watchPosition**

Metoda začne spremljati spremembe geografske lokacije, ob vsaki spremembi vrne novo vrednost tipa Position. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- geolocation.**clearWatch**

Metoda ustavi spremljanje geografske lokacije tako, da se poda enolično referenčno identifikacijo watchID, ki jo kreira geolocation.watchPosition. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

4.4.10. Večpredstavnost

Večpredstavnost (angl. media) se uporablja za zajemanje in predvajanje zvočnih posnetkov na napravi. Trenutna implementacija ne ustreza W3C standardom.

Objekt **media** vsebuje metode:

- media.**getCurrentPosition** (metoda vrne trenutno pozicijo predvajane zvočne datoteke),
- media.**getDuration** (metoda vrne dolžino zvočne datoteke),
- media.**play** (metoda prične oziroma nadaljuje z predvajanjem),
- media.**pause** (metoda začasno ustavi predvajanje),
- media.**release** (metoda sprosti sistemske vire predvajane datotek),
- media.**seekTo** (metoda premakne pozicijo predvajanja na določeno pozicijio),
- media.**startRecord** (metoda prične s snemanje zvočne datoteke),
- media.**stopRecord** (metoda zaključi s snemanjem zvočne datoteke),
- media.**stop** (metoda ustavi predvajanje zvočne datoteke).

Podprte so na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje) in Windows Phone 7 (Mango).

4.4.11. Opozorila

Opozorila (angl. notification) so lahko grafična, zvočna in opozorila z vibriranjem.

Objekt **notification** vsebuje metode:

- notification.**alert**

Prikaže grafično opozorilo na ekranu. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango), Bada (1.2 in 2.x) in webOS.

- notification.**confirm**

Prikaže grafično opozorilo na ekranu z opcijama potrditi ali zavrniti aktivnost. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

- notification.**beep**

Metoda izvede zvočni signal kot opozorilo. Ima vhodni parameter times, ki pove koliko krat se bo signal ponovil. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

- notification.**vibrate**

Metoda sproži vibriranje. Ima vhodni parameter time, ki pove koliko časa v milisekundah naj naprava vibrira. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 5.0 ali višje), Windows Phone 7 (Mango) in Bada (1.2 in 2.x).

4.4.12. Podatkovno skladišče

Podatkovno skladišče (angl. storage) omogoča dostop do lokalne hrambe podatkov. Programski vmesnik (angl. API) je osnovan na specifikacijah W3C Web SQL *Database in W3C Web *Storage. Nekatere naprave podpirajo specifikacijo, v tem primeru PhoneGap ogrodje uporabi privzeto implementacijo.

Objekt **storage** vsebuje metod:

- **openDatabase**

Z metodo kreiramo novo SQL Llite podatkovno bazo. Ta vrne objekt tipa Database, s katerim lahko naprej upravljamo podatke. Podprta je na platformah Android, iOS, BlackBery WebWorks (OS 6.0 ali višje)in webOS.

4.5. Dogodki

Dogodki (angl. events) so tisti dogodki, ki jih ogrodje PhoneGap lahko zazna. Ko se dogodek zazna se sproži ustrezna metoda. V nadaljevanju bomo predstavili dogodke in na katerih platformah so ti podprti.

Vrste dogodkov:

- **deviceready**
  Zelo pomemben dogodek, ki ga vsaka PhoneGap aplikacija mora imeti. Sproži se, ko se v celoti naloži PhoneGap. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje), Windows Phone 7 in Bada (1.2 in 2.x).
- **pause**
  Dogodek se sproži, ko aplikacije preide v delovanje v ozadju. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7.
- **resume**
  Dogodek, ko aplikacija preide iz delovanja v ozadju v aktivno delovanje. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7. (1.2 in 2.x).
- **online**
  Dogodek se sproži, ko se mobilna naprava poveže v mobilno omrežje. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7.
- **offline**
  Dogodek se sproži, ko mobilna naprava prekine povezavo z mobilnim omrežjem. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7.
- **backbuton**
  Dogodek se sproži, ko uporabnik pritisne na tipko nazaj (anlg. backbutton). Uporabno je takrat, ko želimo povoziti (angl. override) privzeto funkcionalnost. Podprt je na platformah Android, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7.
- **batterycritical**
  Dogodek se sproži, ko aplikacija zazna kritično stopnjo napolnjenosti baterije. Podprt je na platformah Android, iOS in BlackBerry WebWorks (OS 5.0 ali višje)
- **batterylow**
  Dogodek se sproži, ko aplikacija zazna nizko stopnjo napolnjenosti baterije. Podprt je na platformah Android, iOS in BlackBerry WebWorks (OS 5.0 ali višje)
- **batterystatus**

Dogodek se sproži, ko aplikacija zazna, da se je spremenila stopnja napolnjenosti baterije vsaj za 1%. Podprt je na platformah Android, iOS, BlackBerry WebWorks (OS 5.0 ali višje) in Windows Phone 7 (Mango).

- **menubutton**
  Dogodek se sproži, ko uporabnik pritisne tipko meni (angl. menu). Uporabno je takrat, ko želimo povoziti (angl. override) privzeto funkcionalnost. Podprt je na platformah Android in BlackBerry WebWorks (OS 5.0 ali višje).
- **searchbutton**
  Dogodek se sproži, ko uporabnik mobilne naprave na platformi Android pritisne tipko za iskanje (angl. search). Podprt je na platformi Android.
- **startcallbutton**
  Dogodek sproži, ko uporabnik pritisne tipko za klicanje (angl. call). Podprt je na platformi BlackBerry WebWorks (OS 5.0 ali višje).
- **endcallbutton**
  Dogodek se sproži, ko uporabnik pritisne tipko za končanje klica (angl. end call). Podprt je na platformi BlackBerry WebWorks (OS 5.0 ali višje).
- **volumedownbutton**
  Dogodek se sproži, ko uporabnik pritisne tipko za zmanjšanje jakosti zvoka (angl. volume down). Podprt je na platformi BlackBerry WebWorks (OS 5.0 ali višje).
- **volumeupbutton**
  Dogodek se sproži, ko uporabnik pritisne tipko za povečanje jakosti zvoka (angl. volume up). Podprt je na platformi BlackBerry WebWorks (OS 5.0 ali višje).

5. Razvoj aplikacije

V tem poglavju bom predstavil praktični del diplomske naloge, ki zajema načrtovanje in izdelavo mobilne aplikacije s pomočjo spletnih tehnologij in ogrodja PhoneGap za operacijski sistem Android.

5.1. Aplikacija

Mobilna aplikacija CRM je narejena z namenom, da olajša delo tržnikom in jim omogoči enostavno poslovanje s svojimi strankami. Nudi jim v vsakem trenutku celovit pregled nad svojim delom. Tako lahko uporabnik aplikacije pregleduje, kaj se je v preteklosti dogajalo z določeno stranko in katera so njegova naslednja opravila. Aplikacija omogoča popoln nadzor nad potekom tržnikovega dela.

Uporabnik na vsakem koraku uporabe programa dobi informacijo, ki jo potrebuje v tistem trenutku. Primer: ko pregleduje aktivnost sestanek, lahko iz tega pogleda vidi, s kom se sestanek odvija in izvede klic ali pošlje sporočilo, kraj sestanka, pretekle aktivnosti vezane na to priložnost, vidi ostale priložnosti iz preteklosti, ki so vezane na to podjetje. Več o sami uporabi aplikacije v nadaljevanju.

Aplikacija uporabniku omogoča sledeče funkcionalnosti:

- dodajanje in urejanje podjetij ter zaposlenih
- dodajanje in urejanje priložnosti
- pregled aktivnih in opravljeni priložnosti
- dodajanje in urejanje aktivnosti
- pregled aktivnih in opravljenih aktivnosti

5.2. Analiza in načrtovanje

Preden sem pričel z analizo in načrtovanjem je bilo potrebno zajeti zahteve.

Zahteve:

- mobilna aplikacija, ki se lahko uporablja na različnih operacijskih sistemih (Android, iOS, …)
- enostaven in uporabniku prijazen uporabniški vmesnik (angl. user interface)
- možnost dodajanja podjetij ter zaposlenih

- možnost kreiranja novih priložnosti
- možnost urejanja priložnosti
- pregled v kateri fazi se nahaja priložnost
- možnost dodajanja aktivnosti, vezane na priložnosti in kontakte
- novo aktivnost se lahko kreira šele takrat, ko se zaključi pretekla aktivnost.

Po zajetih zahtevah sem uvidel, da bom moral aplikacijo kreirati v novem okolju. Preden sem se spoznal z zahtevami aplikacije, sem imel v mislih izdelati jo za operacijski sistem Android v Javi, saj sem se s tem srečal že med študijem. Ker pa je bila prvotna zahteva ta, da je potrebno kreirati aplikacijo, ki bo delovala na različnih operacijskih sistemih sem se odločil, da jo bom izdelal s pomočjo spletnih tehnologij in ogrodja PhoneGap.

Vsakdo želi kreirati enostaven in uporabniku prijazen uporabniški vmesnik in tu je bilo potrebno vložiti veliko truda in raziskovanja. Najprej sem pričel z raziskovanjem in pregledal veliko število CRM mobilnih aplikacij, kako delujejo, kaj omogočajo ipd. Vsi imajo eno skupno lastnost, to je kompleksnost. Mogoče pa tako kompleksne zadeve ne moreš realizirati na enostaven način? Več o uporabniškem vmesniku v poglavju 5.5.

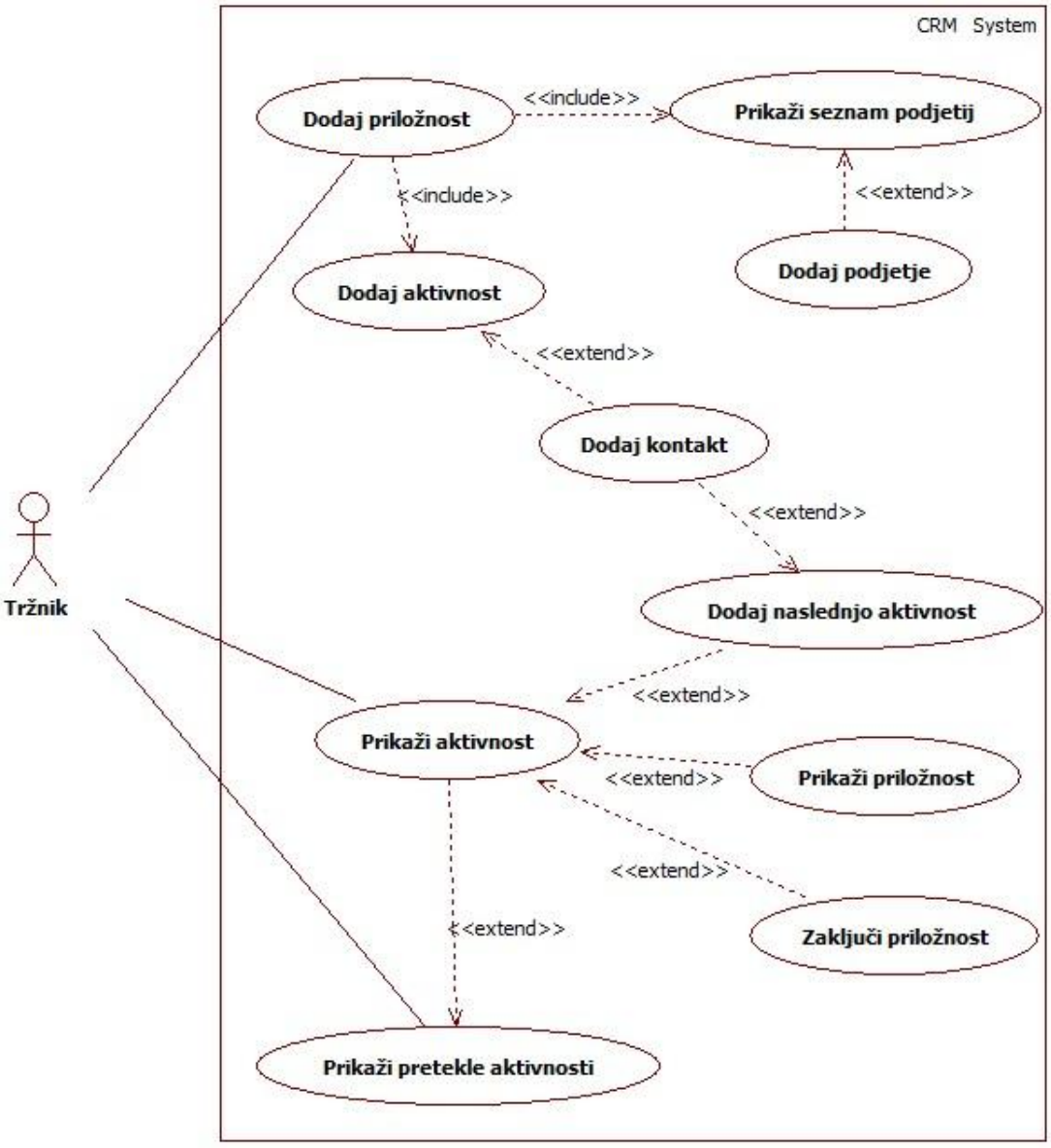
Naslednja pomembna zadeva je bila ta, da je potrebno upoštevati čas, ki ga imajo tržniki na voljo. Ko se tržnik odpravi na teren ne sme izgubljati čas. Kar se da hitro mora imeti vse potrebne podatke na razpolago, brez dodatnih klikov in iskanj.

Potrebno je bilo načrtovati, kako naj se beleži podatek, v kateri fazi se nahaja priložnost oziroma prodaja. Pri nekaterih aplikacijah, ki sem jih pregledoval uporabnik sam določi v kateri fazi je priložnost. To ni ravno zanesljivo, če tržnik ne določi prave faze. V tem primeru lahko narobe določen podatek samo škoduje. V našem primeru tega nismo želeli rešiti na tak način, saj ne želimo uporabniku omogočiti spreminjanja tako pomembnih zadev. Zato smo se dogovorili, da se po opravljeni aktivnosti spremeni faza v kateri se priložnost nahaja. Primer ko tržnik opravi klic se priložnost nahaja na 20%, ko pa zaključi sestanek je priložnost na 40%. Tako ima tržnik pregled nad napredovanjem pri določeni prodaji. S tem, ko aplikacija samodejno spreminja fazo prodaje, na podlagi zaključenih aktivnosti, smo lahko prepričani v točnost te napovedi istočasno pa smo tudi olajšali delo uporabniku.

Mobilna aplikacija deluje na lokalni bazi in za delovanje ne potrebuje nobenih dodatnih pogojev, kot je npr. omrežna povezava. Pri načrtovanju podatkovne baze je bilo potrebno razmišljati dolgoročno, saj je bilo potrebno podpreti sinhronizacijo podatkov s strežnikom, ki se bo izdelala v bližnji prihodnosti.

5.2.1. Diagram primera uporabe

Diagram primera uporabe (angl. use case diagram) v metodologiji UML služi opisu delovanja poslovnega sistema. Na grafičen način prikažemo in zajamemo funkcionalnosti, ki jih sistem omogoča.

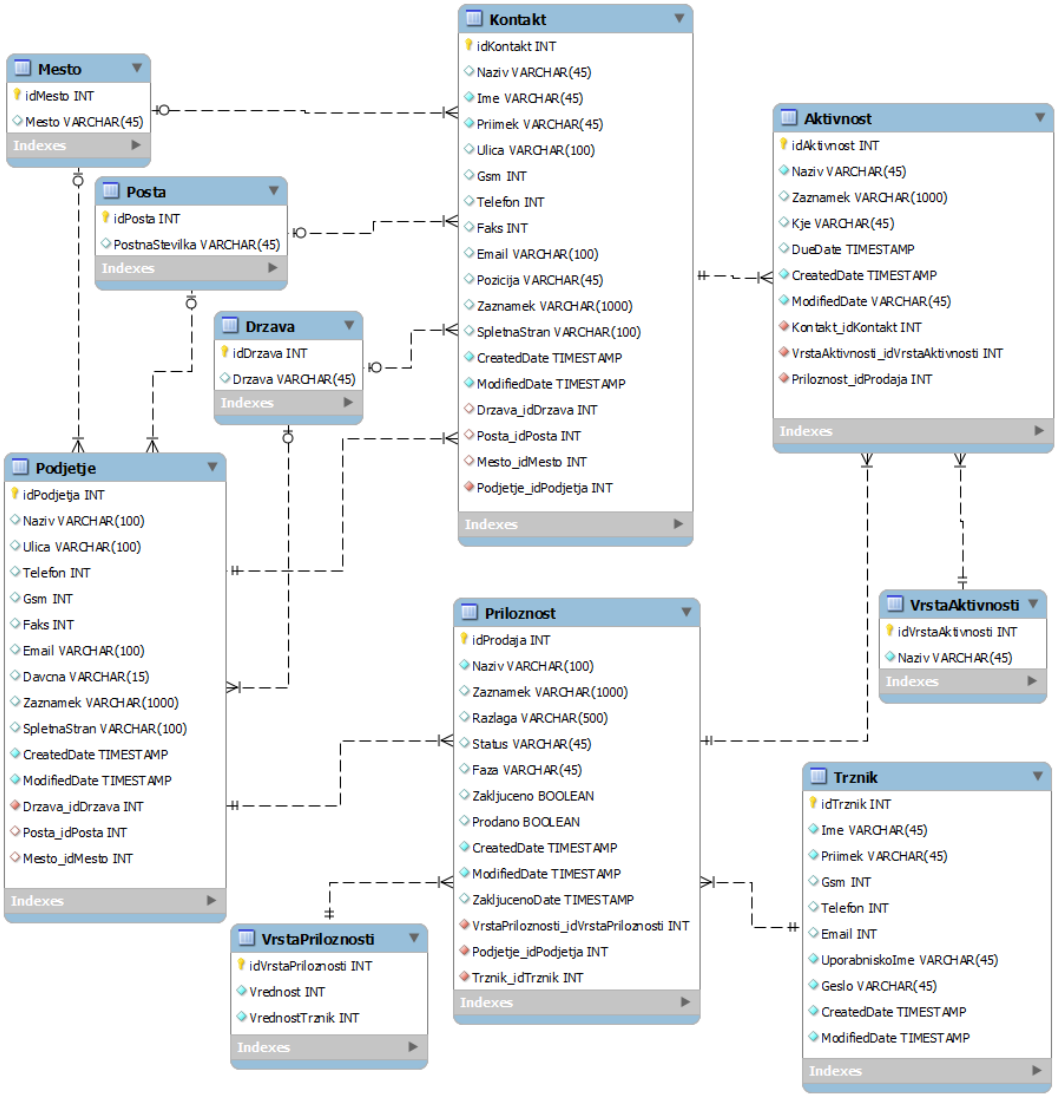


Slika 5.1: Diagram primera uporabe mobilne aplikacije.

Diagram primera uporabe (slika 5.1) prikazuje vse funkcije, ki jih lahko izvrši akter (tržnik).

5.2.2. Podatkovna baza

Osrednji del podatkovnega modela (slika 5.2) so entitete Podjetje, Kontakt, Priložnost in Aktivnost. Entiteti Podjetje in Kontakt sta zadolženi za shranjevanje podatkov o strankah in potencialnih strankah.



Slika 5.2: Podatkovni model mobilne aplikacije.

V entiteti Priložnost so shranjene vse priložnosti, ki so vezane na podjetja. Te so lahko različnih vrst, ko izberemo vrsto določimo vrednost priložnosti in delež, ki pripada tržniku. S tem tržnik ve kolikšna je vrednost priložnosti in kolikšen je njegov delež. Ko se zaključi priložnost mora uporabnik aplikacije izpolniti anketo, ta se shrani v atribut Razlog. Priložnosti so lahko uspešno ali neuspešno zaključene, to nam povedo atributa Zaključeno in Prodano. Atributa Status in Faza sta ključnega pomena, saj tržnik na podlagi teh ve, kolikšen je njegov napredek na posamezni priložnosti.

Entiteta Aktivnost se navezuje na kontakt in na priložnost. Aktivnost je lahko različnih vrst (klic, email, sestanek), v kolikor gre za sestanek je potrebno vnesti še kraj sestanka. Dokler je aktivnost aktivna mora ta imeti vnesen podatek o času dogodka oziroma opomnik, to je atribut DueDate. Prav tako je potrebno vsaki zaključeni aktivnosti vnesti zaznamek.

V tabelah lahko vidimo, da smo uporabili atribut ModifiedDate, ki je tipa TIMESTAMP. Ta beleži čas zadnje spremembe vsakega posega v bazo. To smo uporabili zaradi sinhronizacije s strežnikom, ki se bo implementirala v prihodosti.

5.3. Začetek izdelave aplikacije

Pred začetkom razvoja aplikacije je bilo potrebno pripraviti razvojno okolje za mobilni operacijski sistem Android [16]. V prvi fazi smo namestili razvojno okolje Eclipse IDE 4.2, Android SDK, vtičnik (angl. plugin) ADT in knjižnice PhoneGap. PhoneGap paket, ki vsebuje knjižnice in ustrezno dokumentacijo za vzpostavitev novega projekta sem prenesel z njihove spletne strani.

Ko smo pripravili okolje za razvoj smo ustvarili nov projekt v Eclipse IDE. Vpišemo ime projekta, minimalno verzijo operacijskega sistema na katerem se lahko aplikacija predvaja in ime paketa za identifikacijo projekta. Postopek je enak, kot bi izdelovali izvorno aplikacijo.

Sledilo je nadaljevanje z vstavljanjem PhoneGap knjižnice v naš projekt. V korenski imenik (angl. root directory) našega projekta smo kreirali nove mape:

- **/libs**
- **assets/www**

V mapo **/libs** se je skopirala datoteka **cordova-1.9.0.jar**, ki se nahaja znotraj PhoneGap paketa. Različica datotek se lahko razlikuje. V času, ko sem razvijal aplikacijo je izšla nova različica 2.0.0. V mapo **assets/www** smo skopirali knjižnico **cordova-1.9.0.js**, ki se prav tako nahaja znotraj PhoneGap paketa. V paketu se nahaja še mapa **xml**, ki jo je potrebno z njeno vsebino kopirati v **/res** mapo našega projekta.

Narediti moramo še nekaj sprememb v glavni (angl. main) Java datoteki, ki se nahaja v **/src** datoteki v Eclipse IDE:

- Dodamo **import org.apache.cordova.*;**
- Razred ne razširja več **Activity** ampak **DroidGap.**
- Zamenjamo klic funkcije, ki odpre prvo stran aplikacije **setContentView(R.layout.main)** z **super.loadUrl("file:///android_asset/www/login.html");**

Primer spremenjene Java datoteke:

```
import android.os.Bundle;
import org.apache.cordova.*;

public class mario extends DroidGap {
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
super.loadUrl("file:///android_asset/www/login.html");
}
}
```

Login.html je datoteka oziroma prva vstopna stran v aplikacijo, ki jo je potrebno kreirati v mapi **/www**. Tu po potrebi dodajamo nove HTML datoteke, kot pri izdelavi spletnih strani. V mapi /www se nahajajo tudi knjižnice in CSS datoteke, ki skrbijo za izgleda aplikacije. Na koncu je potrebno nastaviti še dodatne pravice v datoteki **AndroidManifest.xml** in projekt je pripravljen.

5.4. jQuery Mobile

Odprtokodno JavaScript knjižnico jQuery Mobile (slika 5.3), smo uporabili za razvoj uporabniškega vmesnika. JQuery Mobile knjižnica vsebuje razna orodja in funkcije za upravljanje aplikacij nameščenih na napravah, z zaslonom občutljivim na dotik [17]. S pomočjo te je zelo poenostavljen razvoj uporabniškega vmesnika.

Slika 5.3: Logotip odprtokodne JavaScript knjižnice jQuery Mobile [18].

V aplikaciji smo uporabili različico jQuery Mobile 1.1.1.

5.4.1. DateBox

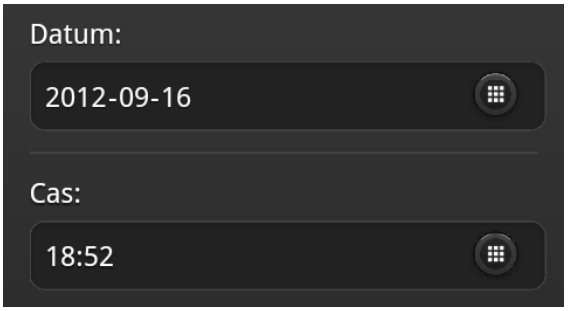
DateBox [19] odprtokodno JavaScript knjižnico smo uporabili za izbiro datuma in časa izvedbe aktivnosti (slika 5.4). Ker so vse konstante v tujem jeziku, je bilo te potrebno prevesti v Slovenščino. Uporaba knjižnice je enostavna, med drugim lahko nastaviš tudi tip izpisa datuma oziroma časa.

Primer vnosnega polja za datum:

```
<input name="date" id="date" type="date" data-role="datebox" data-options='{"mode": "datebox"}'>
```

Primer vnosnega polja za čas:

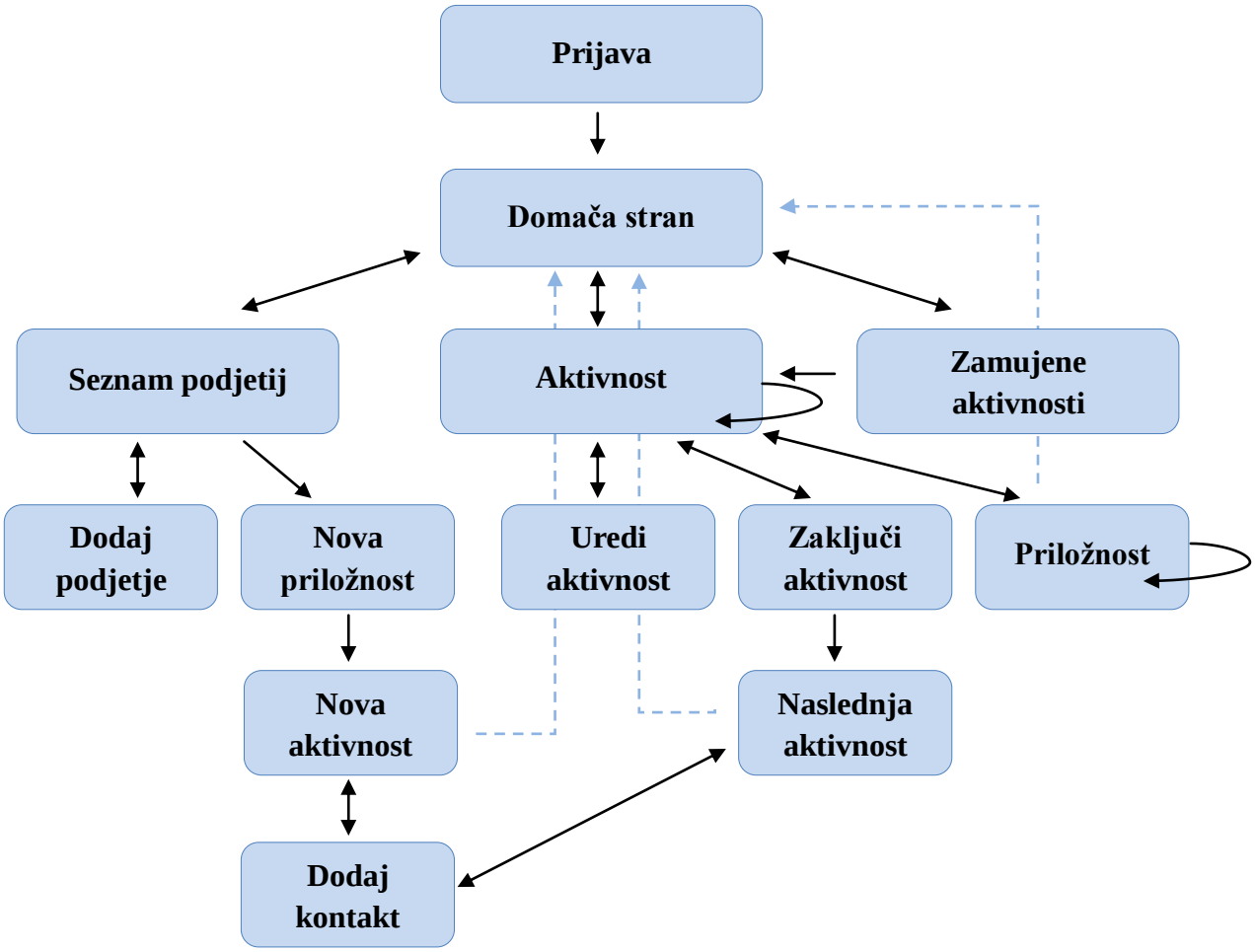
```
<input name="cas" id="cas" type="date" data-role="datebox" data-options='{"mode": "timebox"}'>
```



Slika 5.4: Izgled vnosnega polja za datum in čas.

5.5. Uporabniški vmesnik in uporaba aplikacije

Aplikacija je zasnovana tako, da uporabnika vodi skozi uporabo programa in mu ne dovoli, da zaide s poti. Na sliki 5.5 je videti, kako se posamezna okna povezujejo in sledijo med seboj.

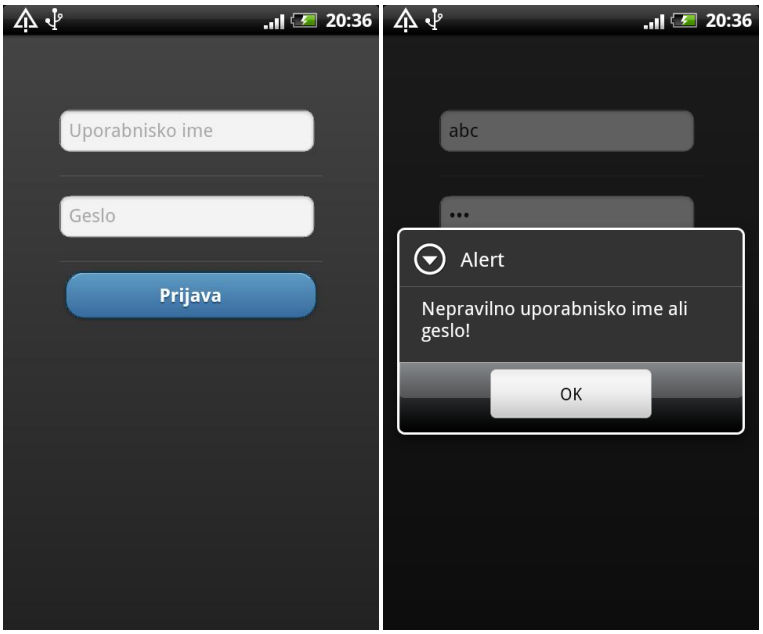


Slika 5.5: Povezava oken med seboj v mobilni aplikaciji.

V nadaljevanju bom skozi uporabniški vmesnik prikazal delovanje mobilne aplikacije.

Prijava

Po zagonu aplikacije se je potrebno najprej prijavit z uporabniškim imenom in geslom za dostop do aplikacije. V primeru neuspešne prijave, aplikacija opozori uporabnika o nepravilnosti uporabniškega imena in gesla (slika 5.6).

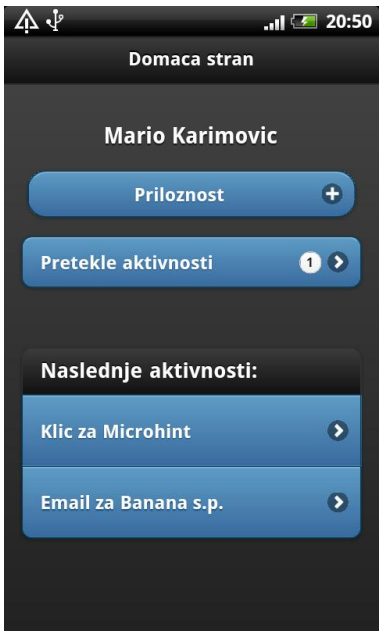


Slika 5.6: Zaslonski maski prijave v aplikacijo ter opozorilo ob neuspešni prijavi.

Domača stran

Ob uspešni prijave se odpre domača stran, na kateri vidimo (slika 5.7):

- **Ime in priimek** prijavljenega uporabnika oziroma tržnika.
- Gumb **Priložnost** je namenjen dodajanju oziroma kreiranju nove priložnosti . Ob kliku nanj se odpre novo okno Seznam podjetij, ki je prvi korak za kreiranje nove priložnosti.
- **Pretekle aktivnosti** prikažejo število zamujenih aktivnosti oziroma število tistih aktivnosti, ki niso bile opravljene v roku. Ob kliku na Pretekle aktivnosti se odpre novo okno Zamujene aktivnosti, kjer se izpiše seznam vseh aktivnosti, ki jim je datum izvedbe potekel.
- **Naslednje aktivnosti** na vhodni strani izpišejo vse aktivnosti, ki jih tržnik mora opraviti na trenutni datum. Tako tržnik ob prijavi nemudoma lahko dostopa do aktivnosti, ki naj bi jih opravil. Ob kliku na posamezno aktivnost, se odpre novo okno Aktivnost, kjer so vidne vse podrobnosti in podatki o tej. Naslednje aktivnosti se izpišejo v seznamu in se razvrstijo oziroma sortirajo po datumu od najstarejše do najnovejše.



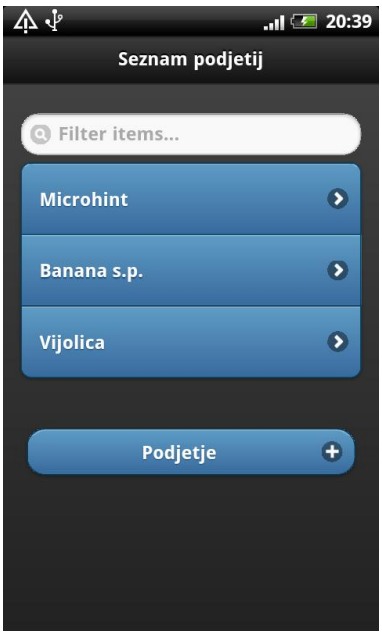
Slika 5.7: Zaslonska maska domače strani.

Domača stran je zasnovana tako, da tržniku poenostavi dostop do njegovih najpogostejših opravil. Z enim klikom in brez posebnega iskanja lahko kreira novo priložnost in pregleda aktivnosti, ki jih ima še za narediti.

Seznam podjetij

Ob kliku na Domači strani na gumb Priložnost se odpre stran s seznamom podjetij (slika 5.8), ki je prvi korak za kreiranje nove priložnosti. Tu je potrebno izbrati ali pa dodati podjetje na katero se bo nanašala nova priložnost.

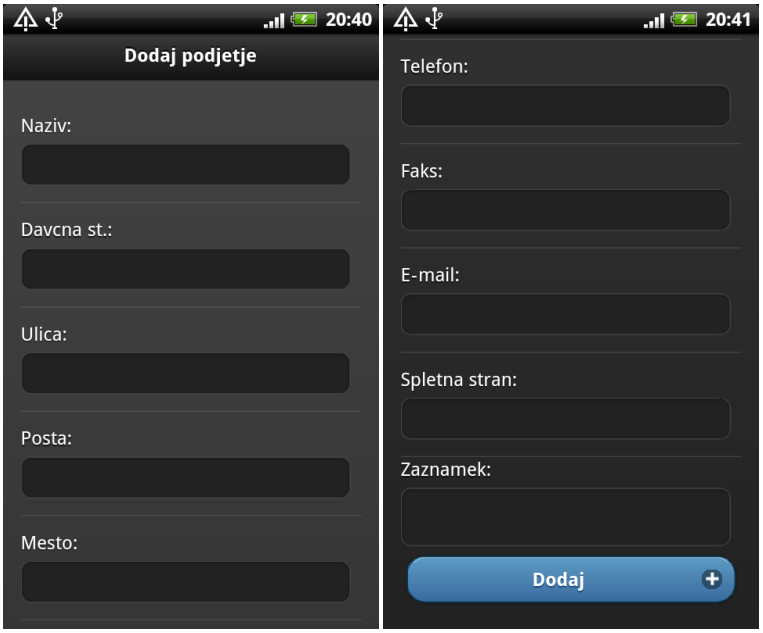
- **Polje za hitro iskanje** »Filter items…« omogoča iskanje podjetij po nazivu. V primeru, da je v lokalni bazi veliko število podjetij je tudi seznam precej dolg in tržniku ni enostavno najti želeno podjetje. Z iskalnikom se skrči nabor podjetij in poenostavi izbiro.
- **Seznam podjetij** prikazuje nazive podjetij iz lokalne baze. Ob kliku na podjetje se odpre novo okno Nova priložnost, ki je vezana na izbrano podjetje.
- Ob kliku na gumb **Podjetje** se odpre novo okno Dodaj podjetje, kjer je potrebno vnesti podatke o podjetju.



Slika 5.8: Zaslonska maska prikaza seznama podjetij.

Dodaj podjetje

Stran Dodaj podjetje (slika 5.9) je namenjena dodajanju novega podjetja v lokalno bazo.



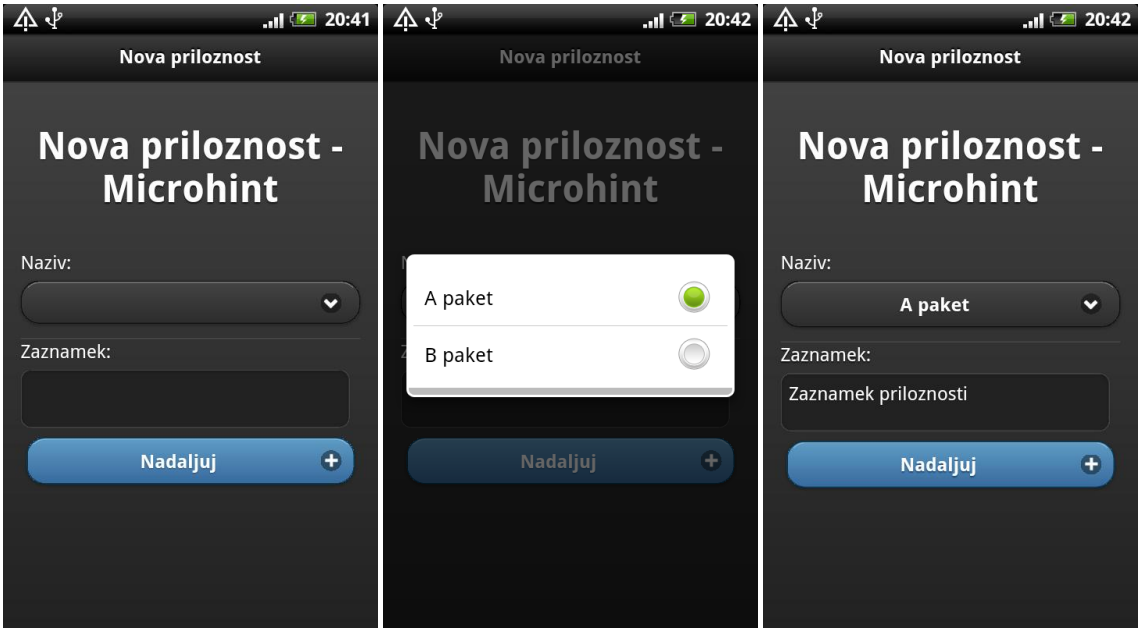
Slika 5.9: Zaslonska maska za dodajanje novega podjetja.

- Potrebno je izpolniti **osnovne podatke o podjetju** (naziv, ulica, pošta, mesto, telefonska številka, …)
- **Zaznamek** je namenjen tržniku, da vnese poljuben zaznamek o samem podjetju.
- Gumb **Dodaj** služi temu, da shrani podjetje v bazo in ob tem kreira info kontakt podjetja. V nadaljnjem postopku je prav tako možno dodati nove kontakte oziroma

zaposlene znotraj podjetja. Ko se podjetje shrani v lokalno bazo, se odpre novo okno Nova priložnost, ki je vezana na pravkar dodano podjetje.

Nova priložnost

Po tem ko smo v prvem koraku izbrali podjetje na katerega bo vezana priložnost je potrebno v drugem koraku dodati novo priložnost (slika 5.10).

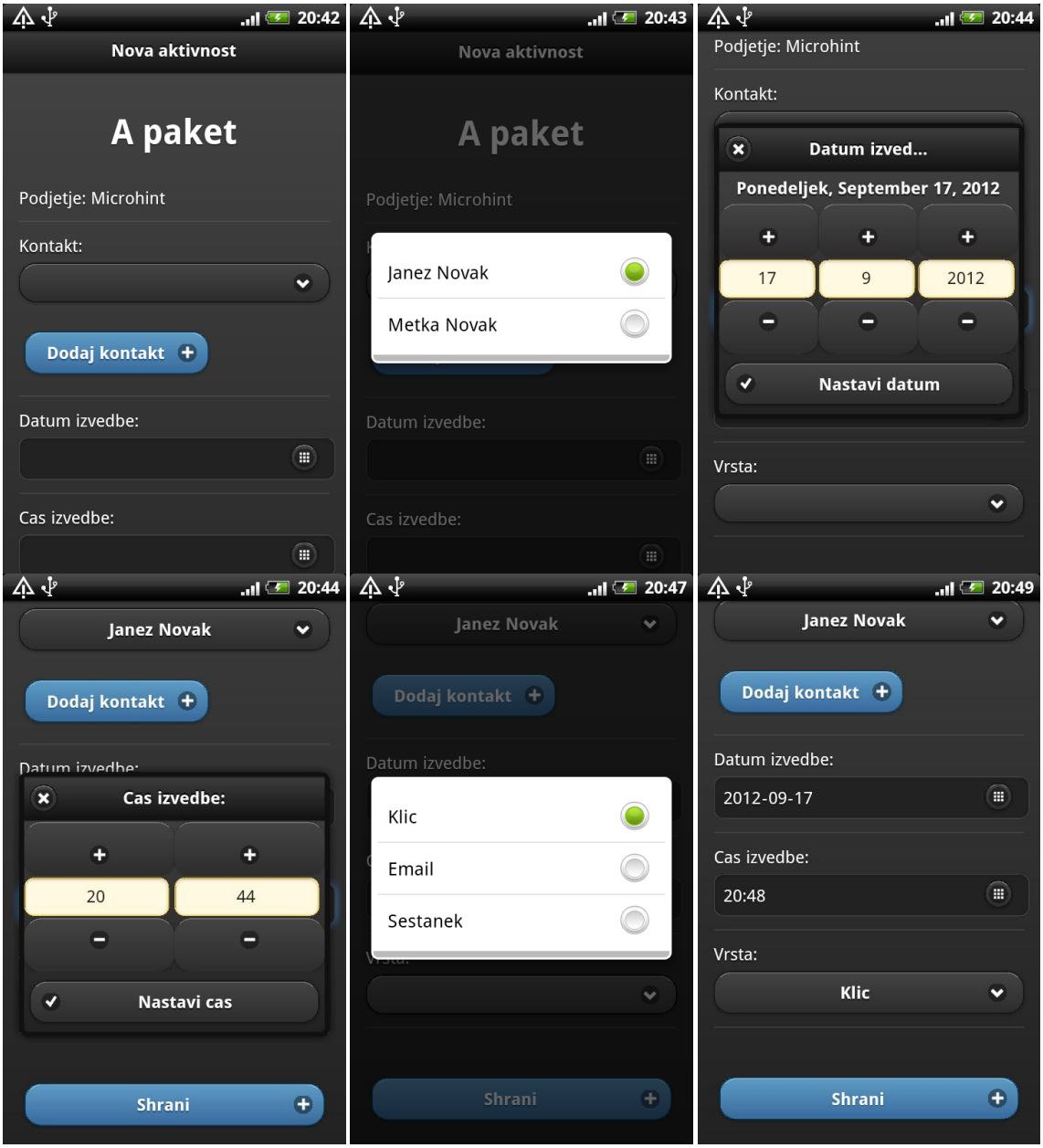


Slika 5.10: Zaslonska maska za kreiranje nove priložnosti.

- Na vrhu se izpiše **Nova priložnost – Naziv podjetja**. Tako tržnik v vsakem trenutku ve, na katero podjetje se navezuje priložnost.
- **Naziv** je potrebno izbrati iz spustnega seznama (angl. drop down). V seznamu se izpišejo vrste priložnosti, ki se razlikujejo med seboj po nazivu in vrednostih. V naši prodaji smo si zamislili, da so v naprej znani možni paketi prodaje in tudi cene oziroma znesek, ki pripada tržniku. S tem ko tržnik izbere vrsto priložnosti, določi naziv in vrednost priložnosti.
- **Zaznamek** je namenjen tržniku, da si zapiše podrobnosti, ki so pomembne za prodajo.
- Gumb **Nadaljuj** vodi na tretji oziroma zadnji korak dodajanja priložnosti. Ko tržnik klikne na gumb Nadaljuj se shrani priložnost in odpre okno Nova aktivnost. To je namenjeno kreiranju naslednje aktivnosti oziroma opravila, ki je vezano na novo nastalo priložnost.

Nova aktivnost

Zadnji in zelo pomemben korak pri kreiranju nove priložnosti je kreiranje nove aktivnosti (slika 5.11).



Slika 5.11: Zaslonska maska za kreiranje nove aktivnosti.

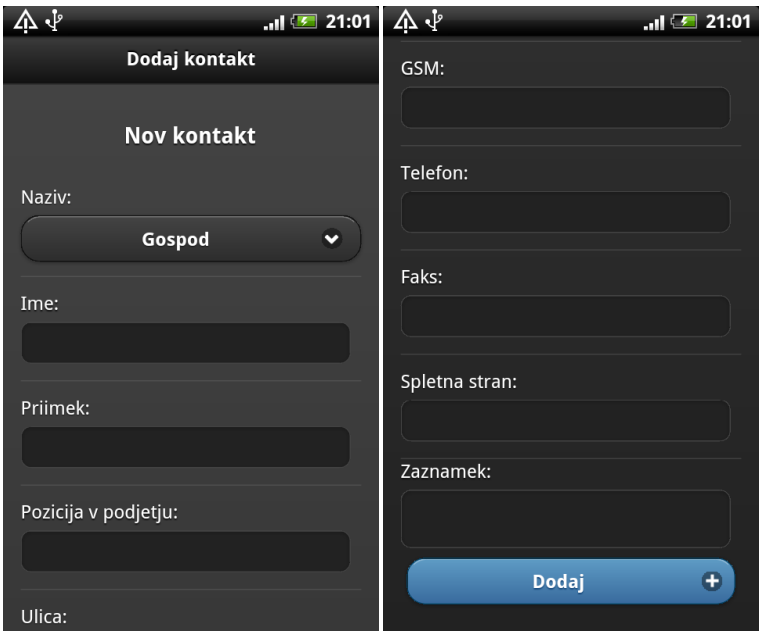
- Na vrhu je izpisan **naziv priložnosti**.
- Pod tem je **naziv podjetja** na katerega se nanaša priložnost.
- **Kontakt** na katerega se navezuje aktivnost se izbere iz spustnega seznama. Tu prikaže vse kontakte iz lokalne baze, ki so zaposleni v podjetju na katerega se navezuje priložnost.

- Ob kliku na gumb **Dodaj kontakt,** se odpre novo okno za dodajanje kontakta. V tem primeru se doda nov kontakt v bazo in v spustnem seznamu kontaktov se doda še nov kontakt, katerega se lahko izbere. Ko se kreira nov kontakt, se tega poveže s podjetjem na katerega je vezana priložnost oziroma aktivnost.
- **Datum izvedbe** je potrebno določiti, saj se aktivnosti ne more kreirati brez datuma izvedbe oziroma opomnika. Ob kliku na datum izvedbe, se odpre v novem pojavnem oknu (angl. popup) možnost izbiranje datuma. Tržnik lahko določi dan, mesec in pa leto izvedbe aktivnosti.
- **Čas izvedbe** je identičen kot datum izvedbe, le da je tu potrebno določiti čas izvedbe na izbrani datum. Odpre se novo pojavno okno, kjer uporabnik določi uro in minuto izvedbe.
- **Vrsta** pomeni za kakšno vrsto aktivnosti gre, to je lahko klic, email ali sestanek. Iz spustnega seznama je potrebno izbrati vrsto aktivnosti. V primeru, da se izbere aktivnost sestanek, se prikaže še dodatno polje v katerega je potrebno vpisati kraj sestanka.
- Ob kliku na gumb **Shrani** se v bazo zapiše aktivnost, ki se bo tržniku prikazala na domači strani pod seznamom naslednjih aktivnosti na datum, ki ga je določil za čas izvedbe aktivnosti. Naziv aktivnosti je vedno sestavljen iz vrste aktivnosti in nazivom podjetja, npr. Klic za Podjetje d.o.o..

Dodaj kontakt

Stran Dodaj kontakt (slika 5.12) je namenjena dodajanju novega kontakta v lokalno bazo.

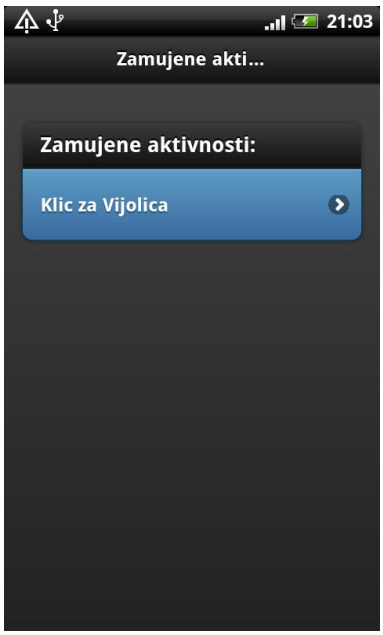
- Potrebno je izpolniti **osnovne podatke o kontaktu** (ime, priimek, ulica, pošta, mesto, telefonska številka, …)
- **Namen zaznamka je**, da se lahko vnese poljuben zaznamek o kontaktu.
- Gumb **Dodaj** služi temu, da shrani kontakt v bazo. Ker aplikacija ve s katere aktivnosti je prišla zahteva za kreiranje novega kontakta, posledično ve za katero podjetje gre. Novo kreirani kontakt poveže z ustreznim podjetjem in ga shrani kot zaposlenega znotraj tega podjetja.



Slika 5.12: Zaslonska maska za dodajanje novega kontakta.

Zamujene aktivnosti

Ob kliku na Pretekle aktivnosti na domači strani, se odpre novo okno Zamujene aktivnosti (slika 5.13).



Slika 5.13: Zaslonska maska zamujenih aktivnosti.

- Prikaže se **seznam aktivnosti**, ki so bile zamujene oziroma niso bile opravljene v roku. Te so razvrščene oziroma sortirane po datumu od najstarejše do najnovejše. Ob

kliku na posamezno aktivnost se odpre novo okno Aktivnost, kjer so vidne vse podrobnosti in podatki o tej.

Aktivnost

Pogled Aktivnost (slika 5.14) prikazuje vse potrebne podatke, ki jih tržnik potrebuje, da lahko uspešno opravi aktivnost. Ko smo razmišljali o tem, kako naj izgledal ta pogled smo imeli v mislih uporabnika, ki je na terenu in nima časa, da bi se ukvarjal še z mobilno aplikacijo. V trenutku mu morajo biti vsi podatki na razpolago, ne glede ali so to pretekle aktivnosti, priložnosti, telefonska številka kontakta,…

Slika 5.14: Zaslonska maska prikaza aktivnosti.

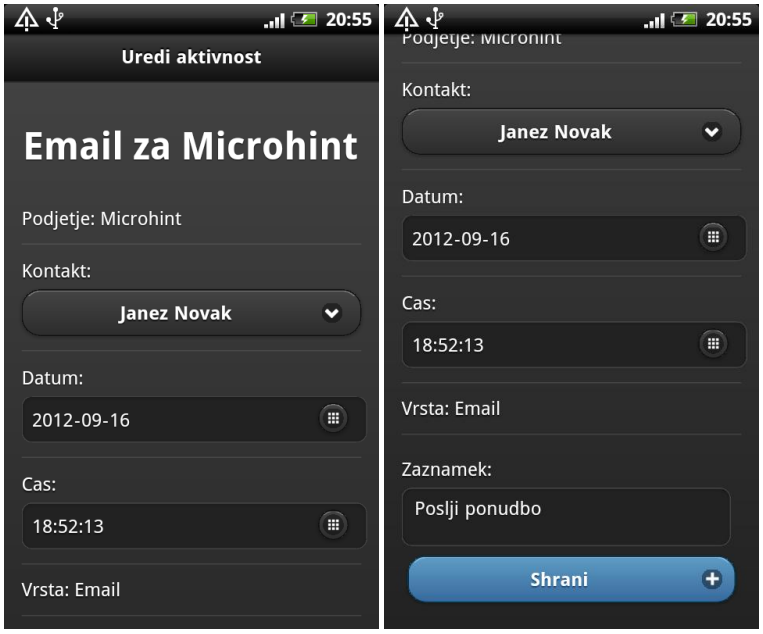
- Na vrhu je **naziv priložnosti**. Ta podatek tržniku pove o kateri zadevi oziroma paketu komunicira s stranko.
- Pod nazivom priložnosti je izpisana **vrednost priložnosti** in vrednost, ki pripada tržniku. Ti podatki so v naprej znani in so shranjeni v lokalni bazi v tabeli VrstePriložnosti. Ko tržnik kreira priložnost mora izbrati za kakšno vrste priložnosti gre. V tem koraku določi vrednost prodaje oziroma priložnosti in posledično delež, ki mu pripada. S tem, ko so ti podatki tržniku v vsakem trenutku vidni, želimo tega spodbuditi oziroma motivirati dober odnos in uspešno komunikacijo in delo s strankami.
- **Dosedanja komunikacija** prikazuje seznam vseh aktivnosti, ki so vezane na trenutno priložnost neodvisno s katerim kontaktom znotraj podjetja so se te zgodile. V vsakem trenutku je tržniku omogočen pregled nad preteklo komunikacijo in s klikom na preteklo aktivnosti vidi podrobnosti o tej. Poleg datuma opravljene aktivnosti in

naziva se tu izpiše še zaznamek aktivnosti. Na tem pogledu je omejen prostor za izpis celotnega zaznamka posameznih aktivnosti, zato je potrebno klikniti na aktivnost in odpre se nov pogled, kjer je izbrana aktivnost prikazana v celoti. Aktivnosti v seznamu so razvrščene oziroma sortirane po datumu od najnovejše do najstarejše.

- **Čas aktivnosti** prikazuje datum izvedbe aktivnosti. V kolikor je aktivnost zamujena oziroma ni bila zaključena v roku se poleg datuma prikaže še ikona s klicajem, ki tržnika dodatno opozori na zamujeno aktivnost. Poleg časa aktivnosti je izpisana **še vrsta aktivnosti**, tako tržnik ve kaj mora narediti. Vidi ali je potrebno opraviti klic, poslati sporočilo, ali pa oditi na sestanek.
- V kolikor je aktivnost vrste sestanek, se pod časom aktivnosti izpiše še **kraj sestanka**, v nasprotnem primeru ta podatek ni pomemben oziroma ga sploh ni in se ne prikaže.
- **Vizitka kontakta** prikazuje podatke o kontaktu na katerega je vezana aktivnosti. Prikazuje ime, priimek, pozicijo v podjetju, telefonsko številko, elektronski naslov in naslov kontakta. Ob kliku na telefonsko številko program pokliče programsko opremo nameščeno na mobilni napravi, ki opravlja klice in tako tržniku omogoči hitro klicanje. Ob kliku na elektronski naslov se izvede klic programske opreme nameščene na mobilni napravi, ki opravlja pošiljanje elektronskih sporočil.
- **Vizitka podjetja** prikazuje podatke o podjetju na katerega je vezana priložnost. Prikazuje naziv, telefonsko številko, elektronski naslov in naslov podjetja. Ob kliku na telefonsko številko ali elektronski naslov se izvede primerna akcija, ki je že opisana na vizitki kontakta.
- **Seznam priložnosti** izpiše vse priložnosti, ki se navezujejo na podjetje, ki je v povezavi s trenutno aktivnostjo. S tem podatkom tržnik lahko vidi vse priložnosti in posledično preteklo komunikacijo, ki je bila opravljena s podjetjem. Poleg datuma zadnje spremembe priložnosti se tu izpiše še naziv, stopnja in faza priložnosti. Stopnja in faza priložnosti tržniku povedo na kakšni stopnji se nahaja priložnost oziroma kakšen je napredek in pa stopnjo že zaključenih priložnosti. Tako lahko tržnik ve npr. na kakšni stopnji je bila pretekla priložnost neuspešno zaključena. Ob kliku na priložnost iz seznama se odpre novo okno Priložnost, kjer so vidne vse podrobnosti izbrane priložnosti.
- Ob kliku na gumb **Uredi** se odpre novo okno Uredi aktivnost za urejanje trenutne aktivnosti. V kolikor je aktivnost, na kateri se nahajamo že zaključena se ta gumb ne prikaže in ni možnosti urejanja že zaključene aktivnosti.
- Ob kliku na gumb **Zaključi** se odpre novo okno Zaključi aktivnost, ki je prvi korak za zaključitev aktivnosti. V primeru, da je trenutna aktivnost že zaključena se ta gumb ne prikaže, saj je ta že zaključena.

Uredi aktivnost

Ko se uporabnik nahaja na podrobnem pogledu aktivnosti, lahko klikne na gumb Uredi za urejanje aktivnosti in odpre se okno Uredi aktivnost (slika 5.15).



Slika 5.15: Zaslonska maska za urejanje aktivnosti.

- Na vrhu je izpisan **naziv aktivnosti**.
- Pod tem je izpisan **naziv podjetja** na katerega se nanaša aktivnost.
- **Kontakt** na katerega se navezuje aktivnost se lahko spremeni oziroma izbere iz spustnega seznama. Tu prikaže vse kontakte iz lokalne baze, ki so zaposleni v podjetju na katerega se navezuje trenutna aktivnost.
- **Datum izvedbe** in **čas izvedbe** se prav tako lahko spremeni in tržniku omogoči npr. prestaviti aktivnost na drug datum.
- **Vrsta** prikazuje za kakšno vrsto aktivnost gre.
- Tržnik prav tako lahko uredi **zaznamek.**
- Ob kliku na gumb **Shrani** se shranijo ustvarjene spremembe in se prikaže predhodno urejena aktivnost.

Priložnost

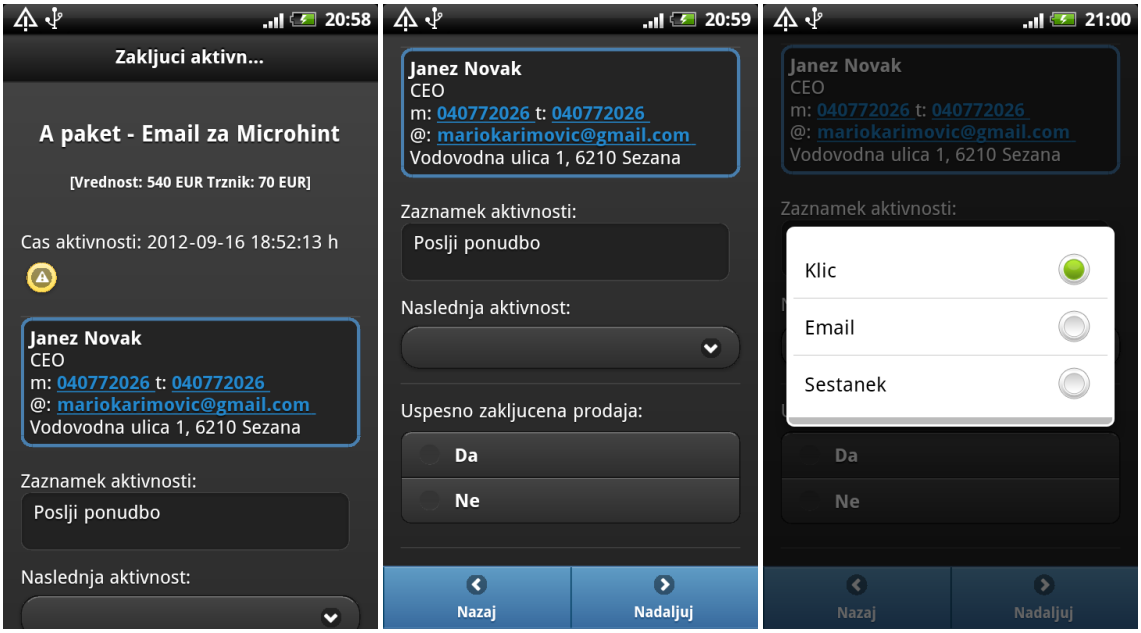
Na podrobnem pogledu aktivnosti se prikaže seznam priložnosti. Ob kliku na eno izmed priložnost iz seznama se odpre novo okno Priložnost (slika 5.16), kjer so izpisane vse podrobnosti o tej.

Slika 5.16: Zaslonska maska prikaza priložnosti.

- Na vrhu je izpisan **naziv priložnosti**.
- Pod nazivom priložnosti je izpisana **vrednost priložnosti** in pa **vrednost**, ki pripada tržniku.
- Tu je izpisana še **stopnja** in **faza** v kateri se nahaja priložnost.
- **Zaznamek**, ki je bil vnesen ob kreiranju priložnosti.
- **Vizitka podjetja** prikazuje podatke o podjetju na katerega je vezana priložnost. Prikazuje naziv, telefonsko številko, elektronski naslov in naslov podjetja. Ob kliku na telefonsko številko program pokliče programsko opremo nameščeno na mobilni napravi, ki opravlja klice in tako tržniku omogoči hitro klicanje. Ob kliku na elektronski naslov se izvede klic programske opreme nameščene na mobilni napravi, ki opravlja pošiljanje elektronskih sporočil.
- V seznamu **vsa komunikacija**, so izpisane vse dosedanje aktivnosti vezane na trenutno priložnost. Aktivnost v seznamu je opisana z datumom izvedbe, nazivom in zaznamkom aktivnosti. Ob kliku na posamezno aktivnost se odpre novo okno Aktivnost, kjer so vidne vse podrobnosti izbrane aktivnosti.
- **Ostale priložnosti** prikazuje seznam ostalih priložnosti, ki so vezane na isto podjetje kot je sedanja priložnost. Poleg datuma zadnje spremembe priložnosti se tu izpiše še naziv, stopnja in faza priložnosti. Stopnja in faza priložnosti tržniku povedo na kakšni stopnji se nahaja priložnost oziroma kakšen je napredek in stopnja priložnosti.
- Na dnu je gumb **Domov** in ob kliku nanj se odpre domača stran aplikacije.

Zaključi aktivnost

Ko uporabnik oziroma tržnik želi zaključiti aktivnost, mora na podrobnem pogledu aktivnosti klikniti na gumb Zaključi. Odpre se mu okno Zaključi aktivnost (slika 5.17), ki je zelo podobno podrobnemu pogledu aktivnosti. Aktivnost lahko zaključi šele takrat, ko napove naslednjo aktivnost ali zaključi priložnost (uspešno ali neuspešno).



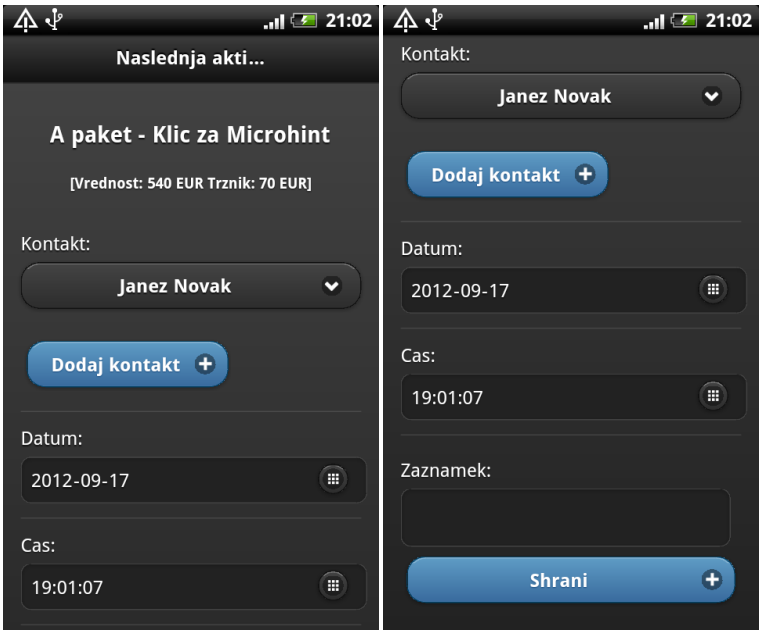
Slika 5.17: Zaslonska maska za zaključitev aktivnosti.

- Na vrhu je **naziv priložnosti** in **naziv aktivnosti**.
- Pod nazivom je izpisana **vrednost priložnosti** in **vrednost**, ki pripada tržniku.
- **Čas aktivnosti** prikazuje datum izvedbe aktivnosti. V kolikor je aktivnost zamujena oziroma ni bila zaključena v roku, se poleg datuma prikaže še ikona s klicajem, ki tržnika dodatno opozori na zamujeno aktivnost.
- V kolikor je aktivnost vrste sestanek, se pod časom aktivnosti izpiše še **kraj sestanka**.
- **Vizitka kontakta** prikazuje podatke o kontaktu na katerega je vezana aktivnosti. Prikazuje ime, priimek, pozicijo v podjetju, telefonsko številko, elektronski naslov in naslov kontakta.
- **Zaznamek** je obvezen podatek, ki ga je potrebno vnesti preden se zaključi in kreira naslednja aktivnost ali pa zaključi priložnost. Tu mora tržnik vnesti zapisnik komunikacije, ki je bila storjena s stranko.
- Tržnik lahko kreira **naslednjo aktivnost**, ki se bo kreirala, ko se bo sedanja aktivnost zaključila. To stori tako, da iz spustnega seznama izbere vrsto naslednje aktivnost (klic, email, sestanek).

- Seveda lahko tržnik **zaključi prodajo** oziroma priložnost. To stori tako, da obkljuka potrditveno polje (angl. checkbox). Izbere lahko DA, kar pomeni uspešno zaključena prodaja ali pa NE, kar pomeni neuspešno zaključena prodaja.
- Na dnu je gumb **Nazaj** in ob kliku nanj se odpre predhodno odprto okno, podroben pogled aktivnosti.
- Da se uspešno zaključi aktivnost, je potrebno klikniti na gumb **Nadaljuj**. Ob kliku nanj se preveri, če je tržnik vnesel zaznamek in izbral naslednjo aktivnost oziroma, če je zaključil podajo.

Naslednja aktivnost

S tem ko je tržnik zaključil aktivnost je moral napovedati naslednjo aktivnost v kolikor ni zaključil prodaje. Ko to stori se odpre okno Naslednja aktivnost, ki je identična oknu Nova aktivnost. Razlika je le v tem, da je tu že določeno kakšne vrste bo naslednja aktivnost, saj je to storil tržnik že v predhodnem koraku.



Slika 5.18: Zaslonska maska za kreriranje naslednje aktivnosti.

- Na vrhu je izpisan **naziv priložnosti** in **naziv aktivnosti**. Naziv aktivnosti je vedno sestavljen iz vrste aktivnosti in nazivom podjetja, npr. Klic za Podjetje d.o.o..
- **Kontakt** na katerega se navezuje aktivnost se izbere iz spustnega seznama. Tu prikaže vse kontakte iz lokalne baze, ki so zaposleni v podjetju na katerega se navezuje priložnost.
- Ob kliku na gumb **Dodaj kontakt** se odpre novo okno za dodajanje kontakta. V tem primeru se doda nov kontakt v bazo in v spustnem seznamu kontaktov se doda še nov

kontakt, katerega se lahko izbere. S tem ko se kreira nov kontakt, se tega poveže s podjetjem na katerega je vezana priložnost oziroma aktivnost.

- **Datum izvedbe** je privzeto nastavljen na tisti datum ko je predhodna aktivnost bila zaključena in izbrana naslednja aktivnost. Seveda datum izvedbe se lahko spremeni. Ob kliku na datum izvedbe se odpre v novem pojavnem oknu možnost izbiranje datuma. Tržnik lahko določi dan, mesec in leto izvedbe aktivnosti.
- **Čas izvedbe** se prav tako privzeto nastavi na tisti čas, ko je bila izbrana naslednja aktivnost. Tu lahko tržnik spremeni čas izvedbe, odpre se novo pojavno okno, kjer se določi uro in minuto izvedbe.
- V primeru, da je bila izbrana naslednja aktivnost sestanek, se prikaže še dodatno polje v katerega je potrebno vpisati **kraj sestanka.**
- V kolikor želi lahko uporabnik vnese še **zaznamek**.
- Ob kliku na gumb **Shrani** se v bazo zapiše nova naslednja aktivnost, ki se bo tržniku prikazala na domači strani pod seznamom naslednjih aktivnosti na datum, ki ga je določil za čas izvedbe aktivnosti.

6. Sklepne ugotovitve in ideje za nadaljnje delo

V diplomskem delu je opisan razvoj mobilne aplikacije, s pomočjo spletnih tehnologij in ogrodja PhoneGap. Kot lastnik pametnega telefona z nameščenim operacijskim sistemom Android, sem posledično izdelal aplikacije za ta operacijski sistem. Aplikacija je namenjena tržnikom na terenu in z njeno pomočjo bodo lahko svoje delo opravljali lažje in bolje.

Med razvojem aplikacije smo testiranje opravili v brskalniku, saj je izvorna koda napisana z označevalnim jezikom HTML, stilne predloge CSS in objektnega skriptnega jezika JavaScript. Ko smo zaključil z razvojem, sem spletno aplikacijo vključil v ogrodje PhoneGap in jo prevedli v domorodno oziroma hibridno aplikacijo za operacijski sistem Android. Na mobilno napravo je bilo nato potrebno prenesti še datoteko s končnico .apk in namestiti aplikacijo. Tako smo pričeli še s testiranjem na mobilni napravi. Seveda smo testiranje opravljali tudi na simulirani mobilni napravi Android s pomočjo vtičnika AVD. Temu smo se hitro odpovedali, saj je bilo testiranje zelo počasno.

Testiranje v brskalniku je pokazalo samo eno pomanjkljivost. V primeru spustnega seznama se ne odvija vse, kot bi se moralo. Ob kliku na spustni seznam, se prikaže nabor vrednosti v seznamu in ob izbiri prve vrednosti v seznamu, se ne posodobi stanje na izbrano vrednost. V primeru, da se izbere drugo vrednost, se stanje posodobi na izbrano vrednost in deluje brez težav. To je težava knjižnic jQuery Mobile, s pomočjo katere smo izdelali uporabniški vmesnik. Ker pa ne delamo spletno aplikacijo, smo to težavo zanemarili in preverili delovanje na mobilni naprav, kjer se ta težava ni pojavila.

Težave, ki so se pojavile ob testiranju na mobilni napravi:

- Odzivnost ni takšna, kot bi si želeli. V primerjavi s testiranjem v brskalniku je bilo delovanje za malenkost počasneje.
- Ob kliku na vnosno polje, kjer je potrebno vnesti tekst se spači videz vnosnega polja.
- Ko aplikacija od uporabnika zahteva, da vnese datum oziroma čas, je uporabljena odprtokodna knjižnica DateBox, ki je del jQuery Mobile knjižnice. Tu se prav tako lahko zgodi, da se ob kliku na izbiro datuma oziroma časa spači vnosno polje.

Vse navedene težave s katerimi smo se srečali, se nanašajo na uporabniški vmesnik. Sklepam, da je težava v izdelavi uporabniškega vmesnika, s knjižnicami jQuery Mobile.

Aplikacija omogoča vse, kar je bilo podano v zahtevah. Toda tu je še veliko prostora za napredek. Ideje za nadaljnji razvoj:

- Vzpostaviti povezavo aplikacije s strežnikom in omogočiti sinhronizacijo. To bi bila nujno potrebna rešitev, da lahko aplikacija preživi na trgu in se sooči z konkurenco.
- Grafični izpis in pa analiza uspešnosti tržnika. V bazi se nahajajo vsi podatki, ki so potrebni za uspešno analizo, potrebno je le te prebrati in jih predstaviti na enostaven in uporabniku prijazen način.
- Prodaja v aplikaciji je vnaprej zastavljena in tržnikom ne omogoča spreminjanja vrednosti prodaje. Potrebno bi bilo omogočiti nastavitve, kjer bi lahko urejali konstante in tako dodatno obogatili aplikacijo. V primeru, da se spremeni vrednost prodaje je sedaj potrebno spremenit izvorno kodo in ponovno naložiti aplikacijo na mobilno napravo.

Aplikacija predstavlja nekaj novega in z nadaljnjim razvojem in odpravo težav z uporabniškim vmesnikom, bi ta lahko dosegla pozitivni šok na trgu.

Slike

Tabele

Viri

[1] T. Myer, Beginning PhoneGap, John Wiley & Sons, Inc., Indianapolis, Indiana, 2012.

[2] (2012) Gartner Says Worldwide Sales of Mobile Phones Declined 2.3 Percent in Second Quarter of 2012. Dostopno na: http://www.gartner.com/it/page.jsp?id=2120015

[3] (2012) Smartphone. Dostopno na: http://en.wikipedia.org/wiki/Smartphone

[4] (2012) Personal digital assistant. Dostopno na: http://en.wikipedia.org/wiki/Personal_digital_assistant

[5] (2012) HTML. Dostopno na: http://sl.wikipedia.org/wiki/HTML

[6] (2012) CSS. Dostopno na: http://sl.wikipedia.org/wiki/CSS

[7] (2012) JavaScript. Dostopno na: http://sl.wikipedia.org/wiki/JavaScript

[8] (2012) Upravljanje odnosov s strankami. Dostopno na: http://sl.wikipedia.org/wiki/Upravljanje_odnosov_s_strankami

[9] (2012) PhoneGap: s spletnimi tehnologijami do mobilnih aplikacij. Dostopno na: http://tehnik.mobitel.si/phonegap-s-spletnimi-tehnologijami-do-mobilnih-aplikacij/

[10] (2012) Grafična predstavitev podprtih spletnih tehnologij s strani PhoneGap ogrodja. Dostopno na: http://devlup.com/mobile/cross-platform-mobile-development-tools/2416/

[11] (2012) PhoneGap. Dostopno na: http://en.wikipedia.org/wiki/PhoneGap

[12] (2012) PhoneGap logotip. Dostopno na: http://phonegap.com/about/artwork

[13] (2012) Grafična predstavitev podprtih platform. Dostopno na: http://phonegap.com/developer

[14] (2012) Grafična predstavitev podprtih funkcionalnosti. Dostopno na: http://phonegap.com/about/feature

[15] (2012) PhoneGap API Reference. Dostopno na: http://docs.phonegap.com/en/2.0.0/index.html

[16] (2012) Getting Started with Android. Dostopno na: http://docs.phonegap.com/en/2.0.0/guide_getting-started_android_index.md.html#Getting%20Started%20with%20Android

[17] (2012) jQuery Mobile. Dostopno na: http://jquerymobile.com/

[18] (2012) jQuery Mobile logotip. Dostopno na: http://en.wikipedia.org/wiki/File:Jquery-mobile-logo.png
[19] (2012) jQuery Mobile DateBox. Dostopno na: http://dev.jtsage.com/jQM-DateBox/