Tallinna Ülikool
Digitehnoloogiate instituut

Ülevaade andmete hoiustamisest ja haldamisest Androidi tarkvaraarenduses

Bakalaureusetöö

Autor: Paul Kirspuu
Juhendaja: Jaagup Kippar

Autor: ....,, ... ,, 2016

Juhendaja: ....,, ... ,, 2016

Instituudi direktor: ....,, ... ,, 2016

Tallinn 2016

Autorideklaratsioon

Deklareerin, et käesolev bakalaureusetöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

....

(kuupäev) (autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Paul Kirspuu (sünnikuupäev: 26.05.1993)

1. annan Tallinna Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Ülevaade andmete hoiustamisest ja haldamisest Androidi tarkvaraarenduses“, mille juhendaja on Jaagup Kippar, säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tallinnas,

....

(kuupäev) (allkiri)

Sisukord

Lühendite loetelu

BaaS – *Backend as a Service* ehk rakenduse funktsionaalsus teenusena

Cloud Code – arendajapoolne lisafunktsionaalsusloogika võrguserveri raamistikule

JSON – *JavaScript Object Notation* ehk tekstivormis kirjutatud lihtsustatud andmetüüp, mis põhineb JavaScripti programmeerimiskeele alamhulgal

NoSQL – *Not Only SQL* ehk andmebaasi päringukeel, mis põhineb mitterelatsioonilistel andmetüüpidel, näiteks võtme-väärtuse ning dokumendipõhised andmed

REST – *Representional State Transfer* ehk võrgurakenduste arhitektuuristruktuuri stiil ühe põhiomadusena kasutada andmete loomise, lugemise, muutmise ning kustutamise funktsionaalsusi

SQL – *Structured Query Language* ehk relatsiooniliste andmebaaside jaoks loodud struktuurpäringukeel

Sissejuhatus

Viimasel dekaadil on ulatuslikult suurenenud andmehulgad, mida mobiilirakendused käitlevad. Populaarne operatsioonisüsteem Android on suuteline hoiustama ja haldama erinevat tüüpi andmevooge, arendaja valik on seejuures, millist vahendit ühe või teise tarkvara välja töötamisel kasutusele võtta.

Käesoleva bakalaureusetöö probleemiks on vähene eestikeelse informatsioon andmete hoiustamisest ja haldamisest Androidi platvormil, mistõttu püütakse see tühimik lahendada ülevaate koostamisega antud teemal.

Töö tulemusena luuakse emakeelne materjal Androidi tarkvaraarenduses kasutatavatest andmekäitluse võimalustest ning süsteemidest, mis seda hõlbustavad. Võrreldakse populaarsemaid vahendeid, mida hetkel teenuseturul pakutakse ja tuuakse välja eelised ühe või teise raamistiku ees.

Käesoleva bakalaureusetöö käigus otsitakse vastuseid järgnevatele küsimustele:

- Millised tehnilised võimalused on andmeid hoiustada Androidi operatsioonisüsteemis?
- Kuidas hallata erinevaid andmetüüpe antud platvormil?
- Millised on populaarseimad võrguandmebaaside pakkujad ning mis on nende eelised ja puudused?
- Miks ja millises olukorras eelistada üht andmekäitluse vahendit teise ees?

1 Androidi andmekäitlus laiemalt

Androidi tarkvaraarenduses on mitmeid viise andmete käitlemiseks ning need jagunevad laiemalt kolme suuremasse gruppi. Esimene andmekäitluse variant on rakendusesisene ehk informatsiooni on võimalik kasutada ilma võrguühenduseta. Teine viis on rakendustevaheline, mis võimaldab teavet erinevate programmide vahel jagada ja seeläbi üksteisega suhelda. Kolmas suurem andmehalduse võimaluste grupp on võrguülene ehk andmed on kättesaadavad nii rakenduses kui ka võrguteenuse kaudu näiteks veebis kuvamiseks ja läbi kasutajaliidese nende käitlemiseks.

1.1. Rakendusesisesed andmed

Sellesse kategooriasse kuuluvad andmed, mida hoiustatakse vaid seadmes endas. Android pakub sellist moodust kasutada soovivate rakenduste jaoks lahendusi nagu Shared Preferences ehk eesti keeles jagatud eelistused, Internal Storage ehk sisemälu ning SQLite Databases ehk SQLite tüüpi andmebaasid (Android Developers, kuupäev puudub).

1.1.1. Sisemälu

Sisemise mälu kasutamisel salvestatakse failid seadmesse, vaikimisi saab neid andmeid kasutada vaid konkreetne rakendus ning mitte ükski teine osapool, ei teised rakendused ega ka kasutaja ise. Programmi eemaldamisel kaovad ka sellega seotud sisemällu salvestatud failid (Android Developers, kuupäev puudub).

Sisemälul on funktsionaalsus, mis võimaldab salvestada andmeid ka vahemällu. Seda on mõistlik kasutada juhul, kui andmed on väikese mahuga ja pole rakenduse toimimise seisukohast hädavajalikud, näiteks logifailid. Vahemälu on võimalik kasutajal endal ruumipuuduse korral puhastada, kuid arendaja ei tohiks eeldada, et see on vaid seadme omaniku ülesanne. Tuleb veenduda, et vahemällu salvestatud failide suurused jääksid mõistlikesse piiridesse, hea tava on umbes 1MB rakenduse kohta. Rakenduse eemaldamisel kaovad ka vahemällu salvestatud failid (Android Developers, kuupäev puudub).

1.1.2. SQLite andmebaas

Android võimaldab kasutada informatsiooni hoiustamiseks ja haldamiseks andmebaasi tüüpi SQLite. Sinna salvestatud andmeid on võimalik kasutada rakenduse piires, igal klassil on teabele ligipääs, kuid teised programmid ning kasutaja ise andmebaasis olevaid kirjeid luua, lugeda, muuta ega kustutada ei saa. Andmebaasi kasutamine on otstarbekas, kui andmed on korduvad ehk struktuurilt sarnased, näiteks kontaktinformatsioon või mängu skoor (Android Developers, kuupäev puudub).

1.2. Rakendustevahelised andmed

Ka siia kategooriasse kuuluvad seadmes hoiustatavad andmed, kuid mida on võimalik rakenduste vahel jagada. Android pakub sellist viisi kasutada soovivate rakenduste jaoks lahendusi nagu eelnevaski peatükis mainitud Shared Preferences ehk eesti keeles jagatud eelistused, External Storage ehk välismälu ja Content Provider ehk sisuteenuse pakkuja (Android Developers, kuupäev puudub).

1.2.1. Jagatud eelistused

Jagatud eelistused on eelkõige mõeldud rakenduse seadistuste salvestamiseks, kuid tihtipeale kasutatakse seda ka üksteisega suhtlevate mobiiliprogrammide puhul. Et need andmed kättesaadavaks teha, on vaja lisada loodavale rakendusele juurde funktsionaalsus, mis teeb jagatud eelistused loetavaks kõikidele programmidele. See võib kujutada endas aga potentsiaalset turvariski, kuna pahavaralised rakendused saavad seeläbi võimaluse andmeid kurjasti ära kasutada. Kui on teada kindel programm, kellega soovitakse andmeid jagada, tuleb mõlemasse tarkvarasse sisse kirjutada vastav seadistus, et viimased on omavahel niiöelda „sõber-rakendused“ (Meier, 2012).

1.2.2. Välismälu

Väline mälu kujutab endast andmete salvestamist kas sisemällu või eemaldatavale välismälu seadmele, näiteks SD kaardile. Informatsioon salvestatud välise mälu funktsionaalsust kasutades võimaldab vaikimisi andmeid luua, lugeda, muuta ja kustutada ükskõik millise rakenduse ning ka kasutaja enda poolt. Välisele mälule salvestamiseks tuleb mobiiliprogrammil saada õigus seda teha ehk enne paigaldust küsitakse kasutajalt, kas viimane on nõus

tingimusega, et rakendus toimetab seadme mälus ning saab modifitseerida andmeid. Keeldumise puhul jääb alla Androidi versiooni Marshmallow jooksutavatele seadmetele programm paigaldamata, kaasaarvatud ja üle selle versiooni omavatel seadmetel on funktsionaalsus, mis võimaldab kasutajal ise valida, milliseid õigusi lubada ja milliseid mitte. Ühe või teise õiguse keelamine võib aga programmi tööd halvata, samas pahavaraliste rakenduste eest on võimalik ennast kaitsta.

1.2.3. Sisuteenuse pakkuja

Content Provider ehk eesti keelde tõlgituna sisuteenuse pakkuja võimaldab ligipääsu kesksele operatsioonisüsteemi andmebaasile läbi kasutajaliidese. Peamine kasutusvaldkond antud vahendil on rakendustevahelise suhtluse loomine ning nendevahelise andmete jagamise koordineerimine (Meier, 2012).

Sisuteenuse pakkuja kaudu on võimalik õiguste andmisel, mis on näiteks Androidi enda rakendustel nagu kalender ja kontaktisirvija vaikimisi lubatud, küsida teiste rakenduste poolt loodud andmeid ning neid ka vajadusel muuta. Seda kõike on mugav läbi viia ilma omapoolse spetsiaalse serveri- ega rakenduseloogikata (Meier, 2012).

1.3. Võrguülesed andmed

See grupp kätkeb endas lugematul arvul teenusepakkujaid, kes võimaldavad andmete hoiustamist serveris ja nende haldamist kasutajaliideses. Android ise võrguandmebaaside loomisesse suuresti panustanud ei ole, küll aga on teinud koostööd, et teenusepakkujad saaksid mugavalt oma rakendusi operatsioonisüsteemiga integreerida. Androidi tarkvaraarenduses kasutatavad põhilisemad võrguülesed andmed võib jagada laiemalt nelja suuremasse gruppi: Real-Time ehk reaalaja-, Big Data ehk analüütilise suunitlusega, Leaderboards ehk mänguga seotud ning informatiivse väärtusega andmed.

1.3.1. Analüütilised ja reaalaja-andmed

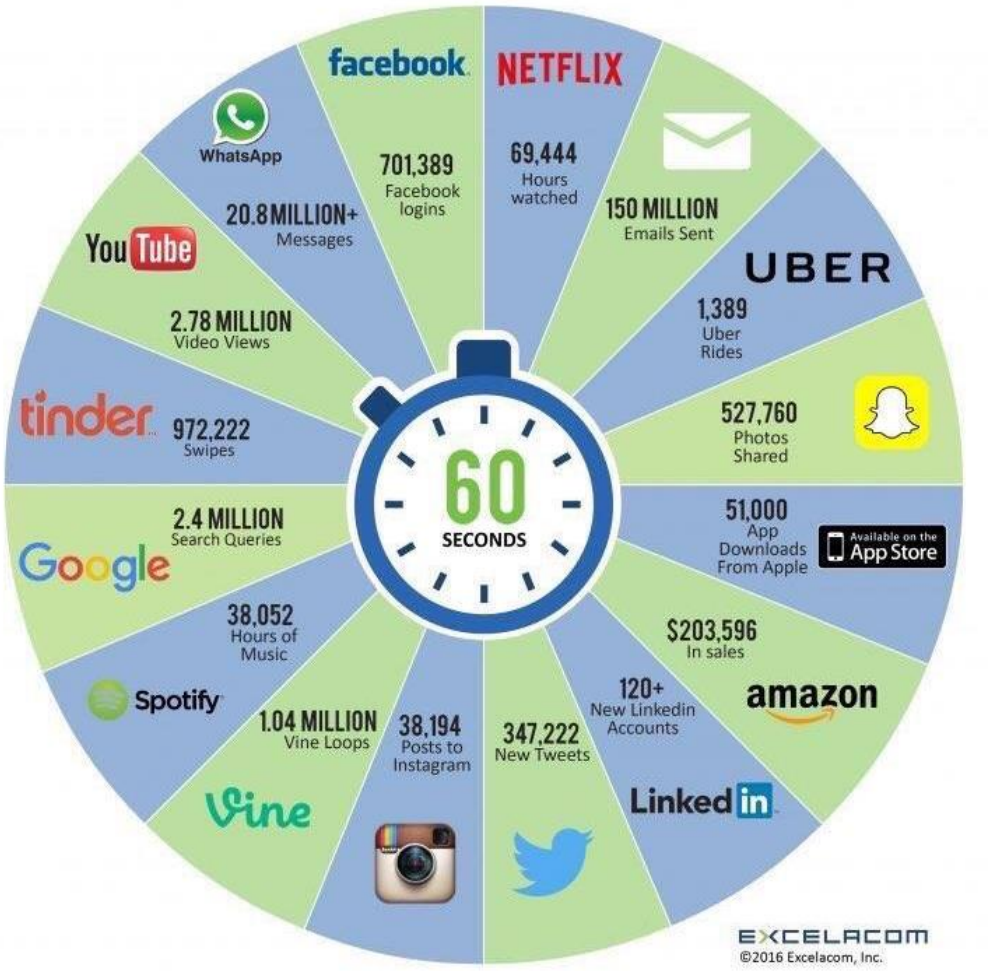
Big Data ja Real-Time andmevood käivad küll enamjaolt käsikäes, kuid antud töös võetakse need luubi alla kui kaks eraldiseisvat andmetüüpi.

Reaal-aja andmed on Androidi tarkvaraarenduses võrdlemisi uus nähe, mistõttu on ka teenusepakkujaid antud valdkonnas vähe. Reaalajas muutuvaid andmeid kasutavad näiteks mitme kasutajaga mängud, sotsiaalmeedia- ning vestlusrakendused. Relatsioonilise struktuuriga andmebaasid sellist võimekust ei oma, seega kasutatakse nimetatud rakenduste väljatöötamisel NoSQL tüüpi andmekandjaid.

Big Data on lihtsamalt öeldes termin kirjeldamaks võimet saada aru ja kasulikult ära kasutada üha suurenevat andmemassi, mida tänapäeval genereeritakse. On hulk andmeid, mille analüüsimisel on võimalik parandada ja edasi viia oma ettevõtmist, olgu see siis näiteks äri, tööstuse või teadusuuringu valdkonnas (Marr, kuupäev puudub).

Informatsiooni genereeritakse täna suuremal hulgal kui kunagi varem. Kui võtta andmed, mis tekitati kuni aastani 2000 ning tuua need tänapäevasesse mastaapi, siis võib öelda, et sama hulk toonast teavet luuakse täna iga mõne minuti järel. Andmete mahust ja selle muutumisest ajas annab hea ülevaate veel see, et kogu maailmas olevast informatsioonist 90% on tekitatud paari aasta eest (Marr, kuupäev puudub).

Uute andmete loomine ja millisel kiirusel need liiguvad, pidades silmas, millised olid vastavad näitajad dekaad tagasi, on fenomenaalne. Igal minutil saadetakse üle maailma 200 miljonit elektroonilist kirja, vajutatakse pea 2 miljonit laiki sotsiaalmeedia keskkonnas Facebook, saadetakse üle 300 000 lühisõnumit Twitteris, laetakse üles 100 tundi videomaterjali YouTube keskkonda ning 200 000 pilti Facebooki. Samal ajal tehakse miljarditesse küündivaid päringuid otsingumootorites, ainuüksi 3,5 miljardit antud valdkonna maailma juhtivas Google internetikeskkonnas (Marr, kuupäev puudub). Järgneval illustratsioonil võib näha veel huvitavaid näitajaid aastal 2016 toimuvas interneti minutis (vt Joonis 1).



Joonis 1. Mis juhtub internetiminutis aastal 2016? (LeBoeuf, 2016).

1.3.2. Mänguandmed

Nutiseadmetele mõeldud mängude andmevood on kontseptsioonilt sarnased kõikide teiste platvormide mängudele. Järgnevalt nimekiri peamistest osadest, mida mängu juures on võimalik talletada (Google Developers, kuupäev puudub):

- *Leaderboards* ehk skooride pingeread
- *Achievements* ehk saavutused
- *Real-time Multiplayer* ehk reaalajas mitme kasutajaga mängude andmed
- *Events and Quests* ehk mänguspetsiifilised sündmused ning ülesanded
- *Saved Games* ehk salvestatud mänguseisud
- *Turn-based Multiplayer* ehk korrapõhiste mitme kasutajaga mängude andmed

Kõikide nende andmevoogude hoiustamiseks ja haldamiseks pakub Google omalt poolt Play Games Services teeki, mis integreerub nii veebipõhiste mängudega kui ka rakendustega muu hulgas platvormidel Android ning iOS.

1.3.3. Informatiivsed andmed

Informatiivse teabe alla võib liigitada kõik muu, mis eelnevate alapealkirjade alla ei kuulu. Nendeks võivad olla lihtsad päringud andmebaasist ühekordseks kasutamiseks, näiteks kasutaja registreerimiseks ning autentimiseks või kirjeldava väärtusega rakenduste andmed, mis võivad olla nii lokaalsed kui ka uuendamise eesmärgil võrguülesed.

Andmekäitlusvahendid informatiivse teabega tegelemiseks ei pea olema nišitooted, mis omavad suurejoonelisi funktsionaalsuseid. Samuti pole nõudlust kindla struktuuri järele, seega sobivad nii relatsioonilised kui ka mitterelatsioonilised NoSQL tüüpi andmebaasid, arendaja saab teha valiku vastavalt konkreetse tarkvaralahenduse vajadustele. Selliste andmevoogudega rakenduste puhul soovitatakse eelkõige kasutada dokumendipõhiseid hoidlaid, kuna need on paindlikud ning struktuuri muutuste suhtes vastuvõtvamad kui võtme-väärtuse põhised või relatsioonilised andmebaasid.

2 Võrguandmebaaside platvormid

Käesolevas peatükis tehakse ülevaade populaarsematest Androidi andmehalduse ja -hoiustamisega tegelevatest teenusepakkujatest võrdluse vormis. Juttu tuleb peamiselt NoSQL tüüpi andmebaasidele keskenduvatest teenustest, kuid mainimata ei jää ka relatsioonilise struktuuriga hoidlad. Valikusse võetakse hetkeseisuga kõige populaarsematest platvormidest kaks dokumendipõhist ning kaks võtme-väärtuse tüübile keskenduvat võrguandmebaasi platvormi.

NoSQL määratlusse kuuluvatest andmebaasidest võib pidada kõige populaarsemaks dokumendipõhist infokandjat ning siinkohal ei tähenda mõiste dokument mitte tekstiformaati, vaid andmestruktuuri. Andmed ladustatakse üldjuhul XML või JSON kujul ning platvorm on võimeline täitma funktsioone nagu informatsiooni lugemine, loomine, kirjutamine ja kustutamine. Dokumendipõhine andmebaas on midagi vahepealset relatsioonilisi ja võtme-väärtuse andmekandjaid silmas pidades – esimene neist olles range struktuuriga, teine põhimõtteliselt struktuuritu (Harrison, 2015).

Teise NoSQL määratlusena võetakse ette võtme-väärtuse tüüpi andmebaasid. Andmed on kujutatud paaridena, on olemas võti ning sellele vastav väärtus. Üldiselt on informatsioon taolistes andmebaasides hoiustatud primitiivsete programmeerimiskeeltest tuntud tüüpidena, näiteks tekstiliste, numbriliste või massiividena. Võtme-väärtuse kujul andmete hoidmise puhul on märksõnadeks kiirus ja andmete pärimise lihtsus, tahaplaanile jääb relatsioonilistest andmebaasidest tuntud range struktureeritus (Seeger, 2009).

Parse oli ja mõningal puhul veel on üks suurima tarbijaskonnaga andmebaasiteenuseid. Arendajate kurvastuseks teatas Facebook 2016. aasta jaanuaris, et sulgeb nimetatud veebiraamistiku. Samal ajal teatati, et ei jäeta oma kasutajaid lageda taeva alla, vaid käivitatakse projekt nimega Parse Server. Teenust kirjeldatakse kui avatud lähtekoodiga versiooni Parse rakendusest, mida on võimalik käivitada igas Node.js rakendust joosta suutval platvormil. Arendajad on nüüd sunnitud üritama olemasolevad rakendused migreerida uute vahenditega või liikuda sootuks uue teenusepakkuja juurde – kuna asi on segane ning hetkel väga kiires arengus, siis antud bakalaureusetöös Parse järeltulijat Parse Server ei käsitleta.

2.1. Couchbase

Mitterelatsiooniliste andmebaaside järsk populaaruse kasv toimus pisut enne dekaadivahetust, CouchDB oli selleks ajaks juba arendanud ja üles ehitanud kasutusvalmis NoSQL baasiraamistiku. Membase, samade andmetüüpidega tegelev konkureeriv teenus tegutses samal ajal oma andmebaasiga, mis kujutas endast Memcached rakendusraamistikku. Memcached baseerub andmete lugemiseks mõeldud vahemälu komponendist, seda kasutatakse näiteks MySQL andmebaasides vähendamaks informatsiooni käitlemisel tekkivaid andmemahte. Membase võttis Memcached tehnoloogia ning ehitas sinna peale funktsionaalsuse, mis võimaldaks andmeid lisaks lugemisele ka lisada, muuta ja kustutada. Üheks näiteks Membase kasutusest on omaaegne populaarne mäng Farmville (Harrison, 2015).

Vaatamata CouchDB tehnilistele saavutustele olla üks esimesi suuri tegijaid NoSQL andmebaaside maastikul tuli ühel hetkel tunnistada, et neil puudub turul ellujäämiseks oma nišš. Aasta 2011 esimesel poolel kuulutati välja Membase ja CouchDB ühinemine. Tegevuse käigus sündis firma nimega Couchbase ning esimese asjana annetati selleks hetkeks valmis kirjutatud serveritarkvara CouchDB Apache kogukonnale. Seeläbi pandi alus projektile Couchbase, mis nüüdseks sisaldab endas CouchDB poolt pärandatud andmetüübile JSON põhinevat raamistikku ning sellega integreeritud Memcached funktsionaalsustega võtme-väärtuse kihti (Harrison, 2015).

Couchbase sobib hästi rakendustele, mis tegelevad analüütiliste või informatiivsete andmete käitlemisega.

2.1.1. Couchbase N1QL

NoSQL operatsioonilised andmebaasid pole tuntud koostööd tegema SQL keelega, kuniks aastani 2015, mil Couchbase tutvustas enda poolt arendatud omasuguste seas esmakordselt nähtud lahendust. N1QL, *Non-First Normal Form Query Language* eesti keelde otsetõlgituna palju ei ütle, kuid antud mõiste võiks defineerida järgmiselt – esimene andmepäringu keel, mis seob omavahel kogu andmetüübi JSON paindlikkuse ning keele SQL võimsuse (Couchbase, kuupäev puudub).

2.1.2. Eelised

- **Avatud lähtekood**
  Couchbase on Apache Licence, Version 2.0 alla kuuluv avatud lähtekoodiga serveriteenus (DB-Engine, kuupäev puudub).
- **N1QL**
  Arendajatel on variant vajadusel kasutada Couchbase poolt kohandatud SQL andmepäringuvormi võimaluste rohkust. (Couchbase, kuupäev puudub).
- **Dokument / võtme-väärtus**
  Couchbase pakub varianti kasutada andmetüübina nii dokumendi kui ka võtme-väärtuse funktsionaalsust (Harrison, 2015).
- **XDCR**
  Couchbase omab funktsionaalsust *cross datacenter replication* ehk eestikeeli andmete duplikeerimine teisele serverisüsteemile varundamise eesmärgil (Couchbase, kuupäev puudub).
- **Andmete killustatus**
  Sharding funktsionaalsus võimaldab rakenduse laienemisel kasutada andmevoogude käitlemiseks mitut masinat (DB-Engine, kuupäev puudub).
- **Võrguühenduseta oleku toetus**
  Kasulik funktsionaalsus rakendustele, mis soovivad töötada ka ilma võrguühenduseta, kuid samal ajal selle olemasolul andmeid sünkroniseerida (Harrison, 2015).

2.1.3. Puudused

- **Puudub tugi konsoolis andmetega töötamiseks**
  Couchbase serveris hoiustatavaid andmeid on võimalik käidelda vaid läbi nende poolt loodud kasutajaliidese (Harrison, 2015).
- **N1QL**
  Couchbase poolt kohandatud SQL andmepäringu vorm on nii eelis kui ka puudus. Raamistiku lisafunktsionaalsus päringute kaudu informatsiooni kättesaamiseks võib osutuda esialgu keeruliseks, kuna on mõnevõrra erinev traditsioonilistest SQL lausetest. Seda võib eriti valusalt tunda suuremahuliste andmepäringute koostamisel (G2 Crowd, kuupäev puudub).

- **Turvalisusseadistuste paindlikkus**
  Couchbase ei paku kasutajatele õiguste määramist dokumentide loomiseks, lugemiseks, muutmiseks ja kustutamiseks. Eksisteerivad vaid äärmused, kas ligipääs on või mitte (G2 Crowd, kuupäev puudub).

2.2. MongoDB

Aastal 2007 lõid Google poolt ära ostetud omaaegse reklaamifirma DoubleClick omanikud ja vanemarendajad uue ettevõtte nimega 10gen. Nende eesmärgiks oli luua PaaS ehk teiste sõnadega seletatuna platvorm teenusena raamistik, sarnane toonase teenusega Google App Engine. Nimetatud raamistiku ehitamiseks oli andmete hoiustamiseks vajadus skaleeritava ja paindliku serveritarkvara järele. Kuna 10gen aastal 2007 oma nõudmistele vastavat kandidaati ei leidnud, otsustati andmebaas luua ise – valminud toode sai nimeks MongoDB. Järgmisel aastal muudeti kurssi ning fookus langes täielikult enda üllitise arendamisele, aastal 2009 väljastati andmebaasi tarkvara avatud lähtekoodiga litsensi alt laiemale kogukonnale kasutamiseks (Harrison, 2015).

MongoDB on dokumendipõhine andmebaas, kus informatsioon hoiustatakse BSON kujul. BSON on binaarselt kodeeritud versioon JSON andmetüübist, mis võimaldab andmeid lugeda ja kirjutada otse kettalt ning samuti toetab informatsiooni hoiustamise võimalusi nagu kuupäevad, kellaajad ning binaarsed andmed. MongoDB andmebaas pakub Javascriptil põhinevaid päringufunktsionaalsusi, mille kaudu on lihtne andmeid käidelda. Dokumendid võivad sisaldada ühe või rohkem väljasid ning andmetüübid võivad varieeruda, olgu nendeks siis alamdokumendid, binaarne informatsioon või massiivid. Selline paindlikkus annab arendajatele võimaluse rakenduse vajadustest lähtuvalt andmemudelit lihtsalt ja kiirelt muuta. Samal ajal on olemas funktsionaalsus andmemudel kinnitada, kui selleks peaks vajadus tekkima (Harrison, 2015). Suurte andmehulkade käitlemiseks ei pruugi üks masin toimingute toetamiseks olla piisav ning selle probleemi lahendamiseks pakub MongoDB *sharding* funktsionaalsust. See kujutab endast masinate lisamist andmevoo laiali jaotamiseks, et informatsiooni lugemine ja kirjutamine sujuvalt toimiks. Couchbase kasutab andmete organiseerimiseks sarnast lahendust nimega vBuckets (Couchbase, kuupäev puudub), (MongoDB, kuupäev puudub).

MongoDB sobib hästi rakendustele, mis tegelevad informatiivsete andmete haldamise ja hoiustamisega, väiksema koormuse puhul ka analüütiliste andmete käitlemisega.

2.2.1. Eelised

- **Avatud lähtekood**
  MongoDB on Free Software Foundation's GNU AGPL v3.0 alla kuuluv avatud lähtekoodiga serveriteenus (DB-Engine, kuupäev puudub).
- **BSON andmetüüp**
  Võimalik lugeda ja kirjutada andmeid otse kettalt (Harrison, 2015).
- **Andmepäringu tööriistad**
  MongoDB raamistik pakub kõrgelt optimiseeritud tööriistu nagu päringuplaneerija ning -efektiivsuse tõstja (Harrison, 2015).
- **Andmete killustatus**
  Sharding funktsionaalsus võimaldab rakenduse laienemisel kasutada andmevoogude käitlemiseks mitut masinat (DB-Engine, kuupäev puudub).
- **Programmeerimiskeelte toetus**
  MongoDB teek toetab kõiki enimlevinud programmeerimiskeeli C, C++, C# ja.NET, Java, JavaScript, Perl, PHP, Python, Motor, Ruby, Scala ning veel teisigi, kokku 27 keelt (DB-Engine, kuupäev puudub).
- **Kiirus**
  Omavahel seotud andmed asuvad enamasti ühes kollektsioonis, seega käivad päringud kiiresti. Kuid seda vaid juhul, kui andmed on dokumendi vormis – relatsiooniliste mudelite emuleerimise korral ei suuda MongoDB oma kiirust säilitada (Harrison, 2015).
- **Andmestruktuurid**
  MongoDB on paindlik ning ei nõua üheseid andmestruktuure üle kõikide objektide (Harrison, 2015).

2.2.2. Puudused

- **Suure hulga andmete käitlemine**
  Kui rakenduses toimuv andmete loomine ja muutmine on massiivne ning pidev, võib MongoDB funktsionaalsus taustal informatsiooni laiali laotada olla aeglane (G2 Crowd, kuupäev puudub).
- **Teenuse omapoolne kasutajaliides puudub**
  Kuivõrd avatud lähtekoodiga tasuta pakutavad kasutajaliidesed võivad olla ühe funktsionaalsuse poolest head ja jäävad vajaka mõne teise omadusega, tekib vajadus täiusliku graafilise kasutajaliidese järele. Selle mure lahendamiseks on küll olemas tasulised vahendid, kuid need kipuvad olema kallid (G2 Crowd, kuupäev puudub).
- **Andmekollektsiooni võti**
  Mitme masina vahel andmete laiali laotamiseks vajaliku võtme valik on äärmiselt oluline kuna seda pole võimalik hiljem muuta (G2 Crowd, kuupäev puudub).

2.3. MongoDB vs Couchbase

Käesolevas alapeatükis võetakse kokku kahe eelnevalt tutvustatud dokumendipõhise andmebaasi eelised ja puudused ning tuuakse välja punktid, millised kasutusjuhud on ühe serveriteenuse eelistamiseks teise ees (Kerby, 2015).

- Kui on plaanis kasutada serveriteenust rakendusega osaliselt ilma võrguühenduseta, tuleks valik teha **Couchbase** kasuks.
- Rakenduse funktsionaalsusloogika jooksutamisel ainult serveripoolel langeb kaalukauss **MongoDB** poole.
- Andmete kättesaadavuse mugavus ja jaotamise toetamine on samuti **MongoDB** eeliseks.
- Andmete jaotamist toetab ka **Couchbase**, kuid valik tuleks langetada teenuse poolt lisaks nimetatud omadusele veel andmete terviklikkuse pärast.
- Kui rakenduses on vaja dünaamilisi päringuid ning eelistatud on defineerida indeksid ise, on parim valik **MongoDB**.
- Andmete aeg-ajaliseks muutmiseks kasutatavate eelnevalt defineeritud päringute jooksutamise jaoks on **Couchbase** õige vahend.

2.4. Firebase

Firebase pakub rakenduse arendamiseks funktsionaalsusi nagu andmete hoiustamine, kasutaja identifitseerimine, staatilise majutuse pakkumine, interaktiivne kasutajaliides ning võrguühenduseta töö (Firebase, kuupäev puudub). Teenus sobib hästi reaalaja-andmetega seotud programmide jaoks, kuna on üles ehitatud sündmusepõhiselt. See tähendab, et kui näiteks SQL tüüpi andmebaasist teabe küsimise peale tagastatakse andmed ning ühendus sulgetakse, siis sündmusepõhine andmevahetus toimub pidevalt. Ühendus on seega katkematu ja iga kord, kui baasis toimub informatsiooni lisamine, muutmine või kustutamine, kasutaja seadmes andmed uuendatakse (Drucker, kuupäev puudub).

Firebase andmebaasis pole tabeleid kui niisuguseid, pole isegi kollektsioone ega dokumente, on vaid üks suur JSON tüüpi objekt. Andmeid on seejuures aga võimalik rakenduse jaoks mugavalt organiseerida ning ükskõik kui sügavaks struktureerida. Informatsioonile ligipääs on tagatud kasutajaliidese kaudu, kust on vajadusel võimalus teavet käsitsi juurde lisada, olemasolevaid kuvada, neid muuta ning kustutada. Kuna andmed Firebases on alati struktureeritud hierarhiliselt, on võimalik kuitahes sügavalt neid õigete võtmete ja väärtuste kaudu veebiaadressiga kätte saada, seda nii brauseris kui ka rakenduse sees (Drucker, kuupäev puudub).

Firebase sobib hästi rakendustele, mis tegelevad analüütilise suunitlusega või reaalaja-andmete haldamise ja hoiustamisega. Mõningate mööndustega sobib võrguraamistik ka mänguandmete käitlemiseks, kuid seda pigem vaid väiksemate nõudlustega mängude puhul.

2.4.1. Eelised

- **Ühekäsuline käivitamine ja tagasivõtmine**
  Lihtne teostada andmete edastamist ja samal ajal ka vajadusel vana versiooni kasutusele võtmist läbi administraatori paneeli (Manglani, 2016).
- **Google toetus**
  Firebase omandati Google poolt oktoobris, aastal 2014. Suurfirma alla kuulumine annab teenusele usaldusväärsuse võrreldes konkurentidega.

- **Hästi dokumenteeritud**
  Suurepärane dokumentatsioon ja alustamisjuhised on ajavõit ning vähekogenenud arendajale suureks abiks õppimisel.
- **Platvormide rohkus**
  Firebase toetab platvorme nagu iOS, Android, JavaScript ning REST (Manglani, 2016).
- **Reaalaja-andmebaas**
  Võimalus ehitada rakendusi, mille sisuks on pidev andmevoogude reaalajas liikumine (Manglani, 2016).
- **Võrguühenduseta oleku toetus**
  Kasulik funktsionaalsus rakendustele, mis soovivad töötada ka ilma võrguühenduseta, kuid samal ajal selle olemasolul andmeid sünkroniseerida (Manglani, 2016).
- **Autentimine paljude teenustega ilma serveripoolse koodita**
  Arendaja ei pea omapoolset loogikat populaarsemate teenustega autentimiseks lisama – Firebase omab sisseehitatud funktsionaalsust autentimiseks muu hulgas ka e-maili ja parooliga või loob anonüümsele kasutajale külastusaja jooksul kehtiva unikaalse identifitseerimisvõtme (Manglani, 2016).

2.4.2. Puudused

- **Teavituste saatmise funktsionaalsus puudub**
  Probleemi lahendamiseks on võimalik kasutada vastavaid teenuseid, näiteks Batch (Manglani, 2016).
- **JSON puu võib vajada harjumist**
  Andmetüübina kasutatav kuitahes sügavaks struktureeritav JSON objektiga ümber käimine võib esialgu osutuda keeruliseks (Manglani, 2016).
- **Puudub Cloud Code lisamise võimalus**
  Alternatiivina saab kasutada oma serverit ja jooksutada vajaminevad lisafunktsionaalsused seal (Manglani, 2016).
- **Platvormil Javascript SDK puudub võrguühenduseta oleku toetus**
  Funktsionaalsus töötab operatsioonisüsteemide iOS ja Android arendustarkvaradel, kuid mitte Javascripti omal.

2.5. AWS Mobile Hub

Suurfirma Amazon poolt pakutav mobiilirakenduste serveriteenus pole pelgalt vaid andmete käitlemiseks mõeldud vahend. Ametlikult 2015. aasta oktoobris käivitatud AWS Mobile Hub arendustarkvara pakub lihtsat võimalust lisada oma mobiilirakendusele funktsionaalsusi ning samal ajal nende keskkonnas seda ka arvukal hulgal seadmetel testida (Amazon Web Services, kuupäev puudub).

Mobiilirakendustele keskenduv Amazoni loodud BaaS ehk tagarakendus teenusena funktsionaalsuste hulka kuuluvad muu hulgas kasutajate autentimine, vahendid andmete analüüsimiseks, sisufailide(näiteks videod ja pildid) jagamine, teavituste saatmine, vahend rakenduse testimiseks, lisafunktsionaalsuste lisamine ning kasutaja andmete hoiustamine (Amazon Web Services, kuupäev puudub).

AWS Mobile Hub sobib hästi rakendustele, mis tegelevad informatiivsete või analüütilise suunitlusega andmete haldamise ja hoiustamisega.

2.5.1. Eelised

- **Andmebaasi tüübi valik**
  Erinevalt enamustest turul olevatest BaaS tüüpi teenuste pakkujatest võimaldab AWS Mobile Hub valida andmete hoiustamiseks kas relatsioonilist või NoSQL andmebaasi (Pratima, 2015).
- **AWS Device Farm**
  Amazon pakub nii automaatset kui ka kaugjuhtimisel arendaja käe läbi toimuvat mobiilirakenduse testimist ning seda omal valikul tehtud seadmete peal (Pratima, 2015).
- **Cloud Code**
  Lisafunktsionaalsuste juurde panemiseks pole tarvis ehitada serveripoolset rakendust, vaid on võimalik läbi AWS Lambda kõike seda mugavalt läbi viia (Pratima, 2015).
- **Hinnapoliitika**
  AWS Mobile Hub tervikliku teenusena on arendajatele kasutamiseks tasuta, raha tuleb välja käima hakata, kui rakendus läheb massidesse. See tähendab seda, et teenus ise ning ka selle funktsionaalsused on mingites piirides tasuta, kuid suuremate kasutusmahtude vajadusel on tarvis kukrut kergendada (Amazon Web Services, kuupäev puudub).

2.5.2. Puudused

- **Kaelamurdev dokumentatsioon**
  Vähekogenenud programmeerimishuvilistele võib osutuda keeruliseks suuremahulises dokumentatsioonirägastikus orienteerumine (Manglani, 2016).
- **Funktsionaalsuste hind**
  Kvaliteetse teenuse eest tuleb välja käia ka vastav summa, mis ei pruugi olla paljudele asjaarmastajatele taskukohane (Amazon Web Services, kuupäev puudub).
- **Limiteeritud programmeerimiskeelte valik**
  Teenuse ülesseadmiseks on võimalik kasutada programmeerimiskeeli Java, Python või Node.js (Amazon Web Services, kuupäev puudub).

2.6. Firebase vs AWS Mobile Hub

Siinses alapeatükis võetakse kokku kahe eelnevalt kirjeldatud serveriraamistiku plussid ja miinused ning võrreldakse, millised rakendustetüübid sobivad paremini Firebase või AWS Mobile Hub teenuse kasutamiseks.

Algajale arendajale võib esialgu osutuda keeruliseks kohaneda Amazoni poolt pakutava teegi dokumentatsioonis ning selle ülesseadmisel oma rakendusele. Kui aga pühendada piisavalt aega ja tahtmist, ei tohiks raamistiku tööle saamine internetis leiduvate õpetuste toel kuigi raskeks osutuda. **Firebase** on üles ehitatud märksa arendajasõbralikumalt ning selle teenuse dokumentatsioonis ära eksimine on vähetõenäolisem kui AWS Mobile Hub puhul.

Kui üheks prioriteetideks on rakenduse testimine erinevate seadmete peal, tuleb igal juhul kasuks suurepärane **AWS Mobile Farm**, mille sarnast ükski teine serveriraamistiku teenus täna ei paku.

Kasutajate autentimine on mõlema teegi puhul läbimõeldud ning funktsionaalsus lihtsasti kasutatav. AWS Mobile Hub puhul käib see läbi erinevate populaarsemate sotsiaalteenuste nagu Google, Facebook ja Twitter ning sama pakub ka **Firebase**. Siinkohal aga selle vahega, et nende teenusega on võimalik autentida ka lisaks eelnimetatud võimalustele kasutajaid e-maili aadressi ja parooliga ning isegi anonüümseks jääda soovivaid kliente.

Firebase andmetüübiks on JSON ning võtme-väärtuse põhiselt üles ehitatud puu on lihtsasti hallatav, kui informatsiooni on väikestes kogustes. Puu kasvades võib osutuda keeruliseks selle struktuuri ümber muuta ning mänguruum antud andmetüübi puhul on küllaltki väike. **AWS Mobile Hub** kasutab andmebaasi DynamoDB, mis on kordades rohkemate funktsionaalsustega, alustades skaleerimisest ja automaatsest andmete varundamisest ning lõpetades erinevate andmetüüpide toetamisega. Neist viimane ongi kõige märkimisväärsem, võimaldades arendajal kasutada enda vajadustele vastavalt skalaarseid väärtusi nagu numbrilised, tekstilised, binaarsed või NULL ning lisaks veel dokumendipõhiseid või võtme-väärtuse kujul olevaid andmetüüpe (Patra, 2015).

3 Andmete hoiustamis- ja haldamisvõimaluste kasutamine

Peatükis võetakse läbi keerulisemate andmekäitlusvahendite kasutusjuhud, kirjeldatakse nende toimimismehhanisme ja funktsionaalsusi ning jagatakse viiteid eelnevas peatükis tutvustatud serveriraamistike ülesseadmise ja kasutamise kohta.

3.1. SQLite

Struktureeritud ja paiksete andmete hoiustamiseks on parim valik Androidi enda poolt pakutav SQLite – avatud lähtekoodiga, võimekas, kuid kergekaaluline SQL tüüpi andmebaasi teek, mille abil saavad tarkvaraarendajad täieliku kontrolli salvestatu üle. Nimetatud andmebaasi teek on sisse ehitatud omakorda C infoobjektide kogumisse, mis kuulub vaikimisi Androidi tarkvarasse (Meier, 2012).

SQLite pole omaette jooksev protsess paljude teiste hulgas, mida rakendus käivitab, vaid on integreeritud kui teek. See vähendab rakenduse ooteaega, sõltuvust välistest vahenditest ning lihtsustab andmevahetuse lukustamist ja sünkroniseerimist. SQLite on tuntud oma usaldusväärsuse poolest, seda tüüpi andmebaasi on kasutanud ja kasutavad ka tänapäeval suur hulk elektrooniliste seadmete tootjaid, muu hulgas näiteks MP3 mängijates, pihuarvutites ning mobiilsetes telefonides (Meier, 2012).

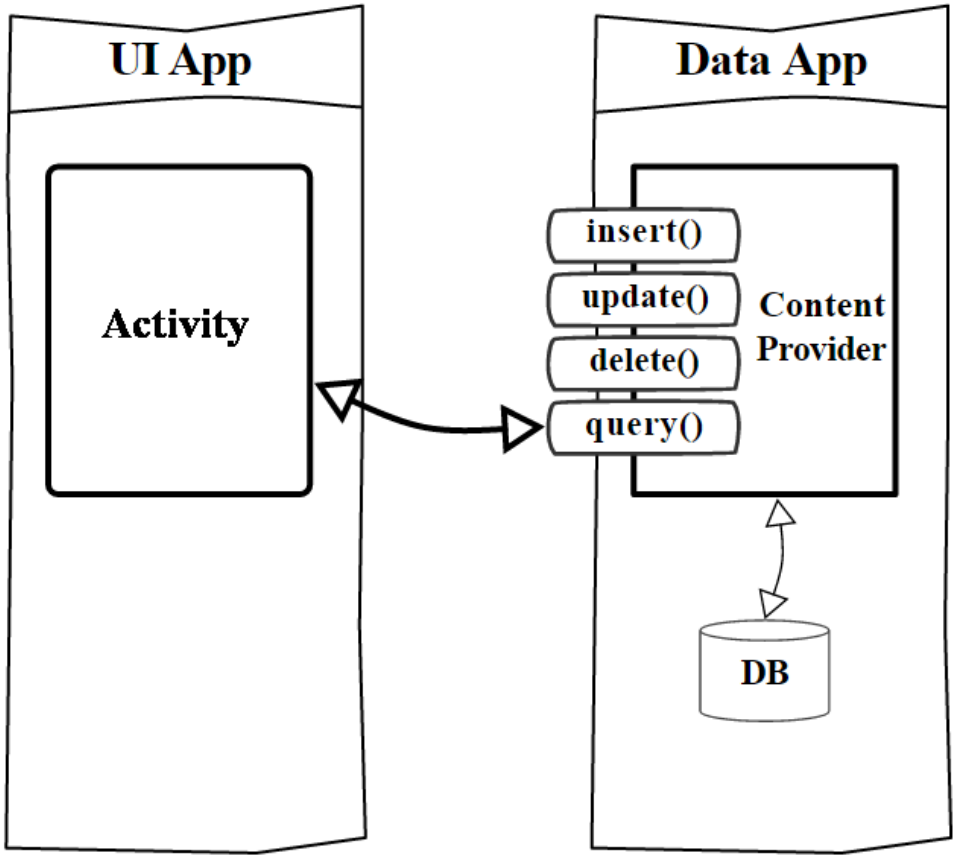
Lihtne, kuid võimas SQLite erineb traditsioonilistest andmebaasidest andmetüüpide määratluse poolest. Veeru väärtuse määramisel näiteks pole tarvis teha muud, kui see lihtsalt päisesse kirjutada. See tähendab, et tulbas paiknevad andmeväärtused ei pea olema ühte tüüpi – selle asemel kirjutatakse hoopis iga andmekild uude ritta eraldi kontrollimata, kas viimane on siis tekstiline või numbriline väärtus (Meier, 2012).

3.2. Sisuteenuse pakkuja

Nagu esimeses peatükis põgusalt kirjeldatud, on *content provider* üks peamisteks vahenditeks rakendustevahelise suhtluse jaoks. Sisuteenuse pakkuja on keskse mehhanismi rollis, mis võimaldab õiguste olemasolul kollektiivselt programmide vahel andmeid hoiustada ning neid hallata. Funktsionaalsus võimaldab rakendusel luua päring operatsioonisüsteemis talletatud andmetele kasutades ühtset ressursi-indikaatorit, mis on laiemale üldsusele tuntud kui internetibrauseri akna ülal asetsevasse kasti kirjutatav saidiaadress. Päringu tegemisel ei tea

küsiv rakendus, milline programm talle andmetega vastab – valiku teostab Android süsteem järgides konkreetsele sisuteenuse pakkujale sätestatud õigusi (Mednieks, Dornin, Meike, & Nakamura, 2011).

Sisuteenuse pakkuja kaudu on võimalik andmeid hoiustada sõltuvalt vajadusele failides, võrguteenuses või SQLite andmebaasis ning käidelda andmeid küsiva rakenduse poolt neljal põhilisel moel: lisada, vaadata, muuta ja kustutada (vaata Joonis 2). Kasutuses olev andmetüüp koosneb kahest märksõnast, meedia- ja alamtüüp ning need on eraldatud üksteisest kaldkriipsuga, näiteks *image/png*. Meediatüüp *image* näitab, et andmekillu sisuks on pildifail, *png* täpsustab, et tegemist on *Portable Network Graphic* alamtüübiga (Rittmeyer, 2012).



Joonis 2. Tüüpiline sisuteenuse pakkuja kasutamine (Jones, 2013).

3.3. MongoDB Androidis

Androidi toe saabumine MongoDB andmebaasi raamistikule on vaid aja küsimus, hetkel on võimalik kasutada MongoDB funktsionaalsusi läbi programmeerimisekeele Java, millele ka mobiiliplatvorm üles on ehitatud. Kuna pole soovitatav rakendust otse andmebaasiga suhtlema panna, oleks tark luua keskne serveriraamistik. Selle ehitamisel tuleb kasutada mõningaid abivahendeid ning üheks enimkasutatavaks lahenduseks on üles seada veebiserverisse tagarakendus, mis on siis niiöelda vahemehe rollis andmete käitlemisel. Keskse serveriraamistiku kaudu on samal ajal võimalik suurendada platvormide arvu, mis teenusega suhelda saavad.

Alustamiseks on soovitatav järgida mõnd teemakohast õpetust, siinkohal soovitatakse kasutada näiteks töö käigus leitud juhendit Node.js ning MongoDB abil Androidi rakendusele registreerimissüsteemi loomise kohta. Õpetuse link on leitav kasutatud materjalide nimekirjas (DataOps, 2015).

3.4. AWS Mobile Hub

Praeguseks hetkeks ligi kuus kuud vana mobiilse serveriraamistiku AWS Mobile Hub kohta on juhendeid ja õpetusi internetis kesiselt. Töö käigus leiti üht funktsionaalsust tutvustav video, mis on üsna keerulise ülesehitusega platvormiga hea koht alustamiseks. Videos näidatakse AWS Lambda ülesseadmise protseduuri mobiilsele rakendusele Mobile Hub abiga, mis lihtsamalt öeldes tähendab programmile lisafunktsionaalsuste juurde panemist läbi raamistiku vahendi ilma tarkvara koodi puutumata. Viide õpetusele asub kasutatud kirjanduse nimekirjas (Chud, 2016).

3.5. Firebase

Firebase pakub omalt poolt seitsmekäigulist õpetust lihtsa veebipõhise suhtlusrakenduse loomiseks, mida on võimalik nende keskkonnas ise ka järele proovida. Androidi juhend on samuti olemas, küll mitte täieliku näitrakenduse kujul, kuid see-eest põhifunktsionaalsused kirjeldatakse selgelt koodiridade kaupa ära.

3.6. Couchbase

Võib julgelt öelda, et Couchbase serveriraamistiku ülesseadmisel täiendavaid juhendeid internetist juurde otsida vaja ei ole. Teenusepakkuja on loonud õpetuse alustamaks rakenduse ehitamist nullist ning lisanud ka iga funktsionaalsuse kohta eraldi väikese näite koodijupi ja seletava teksti kujul.

Kokkuvõte

Käesolev bakalaureusetöö keskendus Androidi tarkvaraarenduses käibel olevate andmete hoiustamise ja haldamise vahendite valikust eestikeelse ülevaate koostamisele. Eesmärgi saavutamisel lähtuti eelkõige algajale arendajale laiema ülevaate tegemisest, millised on levinumate andmekäitlusvahendite kasutusjuhud ning miks eelistada üht teatud olukorras teisele.

Ülevaate koostamisel jaotati Androidi tarkvaraarenduses kasutatavad andmekäitluse vahendid kolme rühma: rakendusesisesed, rakendustevahelised ja võrguülesed. Viimases tehti laiem käsitlus omakorda andmetüüpidest, mida rakendused internetiühenduse korral kasutavad. Nendeks olid reaalaja-, analüütilise suunitlusega, mänguga seotud ning informatiivse väärtusega andmed. Võrguteenuste pakkujate kohta loodi ülevaade lühikirjelduse ning plusside ja miinuste välja toomisega. Viimane peatükk keskendus keerulisemate vahendite täpsemale kirjeldamisele, toimimismehhanismidele ja kasutusnäpunäidete andmisele.

Töö käigus leiti, et Android omalt poolt pakub baaslahendusi lihtsamate rakenduste andmete käitlemiseks, kuid võrguraamistikku omatootena veel välja andnud ei ole. Küll on operatsioonisüsteemi arendusmeeskond pidanud vajalikuks toetada serveriteenuste pakkujaid integreerimaks oma toode Androidiga. Seetõttu on turg sellel alal vägagi lai, erinevatele funktsionaalsustele keskenduvaid teenusepakkujaid palju ning arendajatel on suurepärane võimalus valida loodava rakenduse nõudmistele sobiv vahend.

Summary

Title: Overview of Data Storage and Management Options in Android Software Development

The purpose of this Bachelor Thesis was to create an overview of data storage and management options in Android software development, mainly aiming towards beginner developers. Objective was to review which storage options Android platform provides and to review use cases of the most popular data management and storage service platform providers available.

Creating this overview, data storage and management options in Android software development were divided into three groups – application-based data, applications-shared data and network data. Network data options were further divided into four – Real-Time, Big Data, Leaderboards and data for informative purposes. Data management and storage service providers overview consisted of a description of what they are about and what are the pros and cons of each service. Last chapter focused on giving additional information and use cases about some of the more complicated data storage options introduced in previous chapters.

As a result it was found that Android provides basic solutions for application data management but has not developed a server service of its own. Nevertheless, Android development team is focused on co-operating with service providers to help them integrate their platforms with the mobile operation system. A broad selection of services with different niches are available and application developers get to choose which one is best in their particular case.

Kasutatud kirjandus

Harrison, G. (2015). *Next Generation Databases: NoSQL, NewSQL and Big Data*. New York: Apress.

Meier, R. (2012). *Professional Android 4 Application Development*. Indianapolis: John Wiley & Sons, Inc.

Mednieks, Z., Dornin, L., Meike, G. B., & Nakamura, M. (2011). *Programming Android*. Sebastopol: O'Reilly Media, Inc.

Android Developers. (kuupäev puudub). *Storage Options | Android Developers*. Loetud aadressil http://developer.android.com/guide/topics/data/data-storage.html

Android Developers. (kuupäev puudub). *Content Provider Basics | Android Developers*. Loetud aadressil http://developer.android.com/guide/topics/providers/content-provider-basics.html

Manglani, K. (2016, 11. veebruar). *Top 5 Parse Alternatives*. Ray Wenderlich. Loetud aadressil http://developer.android.com/guide/topics/providers/content-provider-basics.html

Marr, B. (kuupäev puudub). *Big Data Possibilities.* Advanced Performance Institute. Loetud aadressil http://www.ap-institute.com/big-data-possibilities.aspx

LeBoeuf, K. (2016, 29. veebruar). *2016 Update: What Happens In One Internet Minute?.* Excelacom. Loetud aadressil http://www.excelacom.com/resources/blog/2016-update-what-happens-in-one-internet-minute

Google Developers. (kuupäev puudub). *Leaderboards | Play Games Services | Google Developers*. Loetud aadressil https://developers.google.com/games/services/common/concepts/leaderboards

Seeger, M. (2009). *Key-Value stores: a practical overview* (kursusetöö). Loetud aadressil http://blog.marc-seeger.de/assets/papers/Ultra_Large_Sites_SS09-Seeger_Key_Value_Stores.pdf

Couchbase. (kuupäev puudub). *N1QL reference*. Loetud aadressil http://developer.couchbase.com/documentation/server/current/n1ql/index.html

Couchbase. (kuupäev puudub). *Cross Datacenter Replication (XDCR)*. Loetud aadressil http://docs.couchbase.com/admin/admin/XDCR/xdcr-intro.html

Couchbase. (kuupäev puudub). *About sharding data*. Loetud aadressil http://docs.couchbase.com/developer/dev-guide-3.0/sharding.html

MongoDB. (kuupäev puudub). *Sharding - MongoDB Manual 3.2*. Loetud aadressil https://docs.mongodb.org/manual/sharding

DB-Engines. (kuupäev puudub). *Couchbase vs. MongoDB comparison*. Loetud aadressil http://db-engines.com/en/system/Couchbase%3BMongoDB

G2 Crowd. (kuupäev puudub). *Couchbase reviews | G2 Crowd*. Loetud aadressil https://www.g2crowd.com/products/couchbase/reviews

G2 Crowd. (kuupäev puudub). *MongoDB reviews | G2 Crowd*. Loetud aadressil https://www.g2crowd.com/products/mongodb/reviews

Kerby, D. (2015, 14. oktoober). *Why MongoDB is the Way to Go*. Loetud aadressil https://dzone.com/articles/why-mongodb-is-worth-choosing-find-reasons

Pratima. (2015, 22. oktoober). *Amazon announces AWS Mobile Hub and it makes Parse, App Engine look tame*. Algos + Machines. Loetud aadressil https://algosandmachines.com/2015/10/22/amazon-announces-aws-mobile-hub-and-it-makes-parse-app-engine-look-tame

Amazon Web Services. (kuupäev puudub). *What Is AWS Mobile Hub?*. Loetud aadressil http://docs.aws.amazon.com/mobile-hub/latest/developerguide/overview.html

Firebase. (kuupäev puudub). *Features - Firebase*. Loetud aadressil https://www.firebase.com/features.html

Drucker, B. (kuupäev puudub). *Firebase Tutorial: Building a Realtime App with Firebase*. Airpair. Loetud aadressil https://www.airpair.com/firebase/posts/firebase-building-realtime-app

Patra, C. (2015, 13. november). *Amazon DynamoDB: ten things you really should know*. CloudAcademy Blog. Loetud aadressil http://cloudacademy.com/blog/amazon-dynamodb-ten-things

DataOps. (2015, 5. oktoober). *Android Login Registration System with Node.js and MongoDB – Server #1*. DataOps. Loetud aadressil http://dataops.co/android-login-registration-system-with-node-js-and-mongodb

Chud, A. (2016, 10. märts). *Demo: Cloud Code in a Mobile App (AWS Lambda + AWS Mobile Hub)*. YouTube. Vaadatud aadressil https://www.youtube.com/watch?v=JnVdqsCDJAU

Rittmeyer, W. (2012, 12. mai). *Android Tutorial: Content Provider Basics*. Grokking Android. Loetud aadressil http://www.grokkingandroid.com/android-tutorial-content-provider-basics