TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Karl Kallas 175937IADB

Reklaamiserveri täiendamine automatiseeritud turunduse jaoks ettevõtte Bytelogics Inc. näitel

Bakalaureusetöö

Juhendaja: Toomas Lepikult
PhD

Kaasjuhendaja: Keith Siilats
MA

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Karl Kallas

07.01.2021

Annotatsioon

Käesoleva bakalaureusetöö eesmärk oli parandada ettevõtte Bytelogics DSP (*Demand Side Platform*) reklaamiserveri automatiseeritud turunduse efektiivsust ehk tõsta veebidomeenidel näidatavatele reklaamidele tulevate klikkide osakaalu turunduseelarvet suurendamata. Töö sisuks oli efektiivsust parandava metoodika leidmine ning teostamiseks tehniliste lahenduste väljatöötamine ja olemasoleva serveri täiendamine uute lahendustega.

Töö käigus seati eesmärgid, tehti kindlaks olemasoleva serveri puudused, pandi paika töö maht, selleks vajalikud täiendused ja arendati välja lahendused. Tulemusena esitatakse lahendused, mis arendati OpenRTB (*Open Real Time Bidding*) standardeid järgides. Pidades kinni ka *Google Authorized Buyers* reklaame vahendava teenuse kriteeriumitest ning reklaamioskjoni serverite arhitektuurilistest praktikatest. Lahenduseks loodud metoodika ning arenduste abil tõusis Bytelogics automatiseeritud turunduse efektiivsus kahekordselt.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 34 leheküljel, 7 peatükki, 15 joonist.

Abstract

Improving the Ad Server for Programmatic Advertising on the Example of Bytelogics Inc.

The aim of this thesis was to improve the ad server of Bytelogics DSP (*Demand Side Platform*) by increasing its efficiency of programming advertising without raising the advertising budget. The efficiency is measured in the clicks that ads recieve. Bytelogics DSP ad server had previously not met its performance expectations because it was lacking data about the best performing web domains that show online ads.

This thesis covers the work that includes finding the right methodology and technology to increase the efficiency and therefore improve the existing DSP server. Programmatic advertising that makes up the majority of online digital advertising helps its users to bid on ad spaces in real time. By collecting the data about domains and users from ad requests that are sent from ad exchanges to DSP servers it is possible to create specific pretargeting groups that receive the ads.

To fulfill the aim of the thesis it was important to point out the deficiencies of the existing ad server, improvements it needed, scope and objectives of the work and also to develop the solutions. The presented outcome was developed by following OpenRTB standards and Google Authorized Buyers ad exchange practices. Also the overall conventions of programmatic advertising.

The scope of the work was to create a data pipeline for collecting data from bid requests and also to create an analysis on the collected data to form a *whitelist* of domains that have the best ad performance on them. The developed solution helped to double the efficiency of Bytelogics DSP programmatic advertising.

The thesis is in Estonian and contains 34 pages of text, 7 chapters, 15 figures.

Lühendite ja mõistete sõnastik

Ad Exchange	Reklaamioksjonil reklaame vahendav teenus
Automatiseeritud turundus	Digireklaamide automaatne suunamine täpsematele kasutaja gruppidele, kasutades selleks kogutud andmeid
DMP	*Data Managment Platform*, andmehaldus platvorm DSP serveriga koostööks
DSP	*Demand Side Platform*, reklaamioksjonil osalev reklaame nõudev platvorm
Google Authorized Buyers	Google poolt arendatud Ad Exchange teenus
HTML	*Hypertext Markup Language*, hüperteksti märgistuskeel
HTTP	*Hypertext Transfer Protocol*, protokoll teabe edastamiseks arvutivõrkudes
IAB	*Interactive Advertising Bureau*, interaktiivse turunduse büroo
JSON	*Javascript Object Notation*, JavaScripti programmeerimiskeele alamhulgal põhinev lihtsustatud andmevahetusvorming
MVP	*Minimal Viable Product*, minimaalne töötav toode ehk toote algeline kuju, mis on küllaldane tagasiside kogumiseks
MySQL	Relatsiooniline andmebaasi haldamise süsteem
OpenRTB	*Open Real Time Bidding*, reaalajaoksjonite tarbeks loodud standardid
PHP	*Hypertext Preprocessor*, skriptimiskeel serveripoolsete lahenduste loomisel
Reklaamiserver	Automatiseeritud turunduses osalev server
RTB	*Real Time Bidding*, reklaamipindadele reaalajas toimuv oksjon
Whitelist	List asjadest, mida saab pidada aktsepteeritavaks või usaldusväärseks

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesoleva bakalaureusetöö eesmärgiks on täiendada ettevõtte Bytelogics Inc. reklaamiserverit automatiseeritud turunduse efektiivsuse tõstmiseks. Automatiseeritud turundus on digireklaamide automaatne suunamine täpsematele kasutaja gruppidele, kasutades selleks kogutud andmeid [39].

Bytelogics Inc. on Ameerika Ühendriikides baseeruv ettevõte, mille värskeim projekt on reklaamioksjonitel reklaame nõudva teenuse ehk DSP (*Demand Side Platform*) loomine. 2020. aasta septembris valminud esimene MVP (*Minimum Viable Product*) versioon võimaldas ettevõttel alustada esimest automaatselt toimivat kampaaniat reklaamikohtade ostmiseks veebilehtedel.

Paraku ei vastanud turundustegevuse efektiivsus aga ootustele ja reklaamidele saadi oodatust vähem klikke. MVP versioonis panustati oksjonitel reklaamikohtade saamiseks kõikidel võimalikel veebidomeenidel, sest puudusid automatiseeritud turundust toetavad ja ülevaadet loovad andmed hästi toimivate reklaame näitavate domeenide kohta. Lahenduseks loodud andmete kogumis kanali ning nende analüüsil saadud tulemuse integreerimine serverisse kuulubki autori loodud täienduste hulka ning moodustab põhilise osa alljärgnevast lõputööst.

Käesoleva lõputöö raames otsustati parandada automatiseeritud süsteemi efektiivsust pidades kriteeriumiks turundusse paigutatud eelarvet mitte suurendada. Sihiks seati olemasolevat süsteemi toetava andmete kogumis kanali ning andmehaldus võimaluste loomine. Bytelogics reklaamiserveri täiendusteks vajalike lahenduste väljatöötamine, metoodika leidmine ja tehniline elluviimine on käesoleva töö autori panus ettevõtte turunduse efektiivsuse kasvu ning on ka alljärgneva töö peamiseks sisuks. Töös on esitatud Bytelogics DSP täiendusteks väljatöötatud lahendused, mis näitavad, kuidas muuta automatiseeritud turundust efektiivsemaks peamiselt PHP ja MySQL tehnoloogiaid kasutades.

Käesoleva bakalaureusetöö põhiosa moodustavad viis peatükki. Esimses peatükis kirjeldatakse töö ülesande püstitust. Teises peatükis luuakse ülevaade automatiseeritud turunduse taustast ja ajaloost, teema päevakohasusest ning eelistest reklaame nõudvatele ettevõtetele ja internetikasutajatele. Kolmandas peatükis kirjeldatakse automatiseeritud turunduse lähtekohti võttes arvesse andmehalduse rolli, OpenRTB protokolli ja Google Authorized Buyers'i teenuse suuniseid. Lisaks tutvustatakse DSP serveri suhtlust hajussüsteemis ja DSP serveri andmehaldust. Neljas peatükk annab ülevaate Bytelogics DSP kohta, toob välja selle puudulikkused ning kirjeldab täienduste loomise protsessi. Viiendas peatükis esitletakse leitud lahendusi ning tuuakse välja järeldused ja võimalikud edasiarendused.

2 Ülesande püstitus

Bytelogics DSP reklaamiserveri turundustegevuse efektiivsus ei vasta ootustele ja reklaamidele saadakse vähe klikke, sest puuduvad automatiseeritud turundust toetavad andmed hästi toimivate reklaame näitavate domeenide kohta.

Probleemist lähtuvalt seatakse töö eesmärgiks täiendada olemasolevat Bytelogics DSP reklaamiserverit luues DMP'le (*Data Management Platform*) omase funktsionaalsusega andmekogumis ning -analüüsi lahendused. Eesmärk on analüüsida automatiseeritud turunduses praktiseeritavaid standardeid ning tehnoloogiaid. Töö fookus on luua kanal andmete kogumiseks ja kogutud andmete analüüsil teha nö valge list (*whitelist)* domeenidest, mida server kasutab, et optimeerida ja muuta turunduskampaaniad efektiivsemaks läbi teadlikult parema tootlikkusega domeenidele ja seeläbi täpsemale sihtgrupile reklaamimise.

3 Automatiseeritud turundus

Peatükk loob ülevaate automatiseeritud turunduse taustast ning päevakohasusest. Lisaks kirjeldab millist mõju avaldab tehnoloogia erinevatele osapooltele. Viimane alapeatükk selgitab teemaga seotud piiranguid ning õigusi.

3.1 Teema taust ja ajalugu

Tehnoloogia eksponentsiaalne areng ning eelkõige interneti, personaalarvutite ja nutitelefonide kasutamise laialdane levik on loonud aluse uute ning järjest efektiivsemate turundus meetodite tekkeks. Suur andmete hulk, mis üle 4 miljardi internetikasutajaga tekib, on tänapäeval väga oluline turundust vedav jõud [1]. Üheks selliseks turundus meetodiks on automatiseeritud turundus.

Automatiseeritud turundus on osa internetiturundusest, mis sündis 90ndate aastate keskel. Esimese reklaami, mis kujutas endast ristküliku kujulist pilti kirjaga „*Have you ever clicked your mouse right HERE?*“, ostis domeenile www.hotwired.com Ameerika telekommunikatsiooni ettevõte AT&T. Sellel ajal müüdi reklaamipinda veebilehtedel sarnaselt ajakirjadega. Lehe omanikud rentisid välja selleks ettenähtud osa leheküljest, millel näidati reklaame. Reklaam oli staatiline olenemata sellest, kes parasjagu domeeni külastas, sest informatsioon reklaami kohta oli muutmata kujul veebilehe enda serveris kuni kampaania lõpuni. [2]

Kuna domeene tekkis aina juurde ning enamus reklaamipinda renditi manuaalselt ning pigem eksklusiivsete lepingutega, siis jäi palju nendest müümata. Ometi saadi aru, et tegu on väärtusliku osaga ärist ning probleemi lahenduseks hakati neid kohti pakkuma soodushindadega oksjonitele, milles osalesid nii reklaamipinna pakkujad kui ka need, kes enda reklaame soovisid näidata. [2] Selleks loodi 1995. aastal spetsiaalsed reklaamiserverid, mis hoidsid reklaamipindade aadresse ning reklaamide infot enda serverites ja vahendasid neid vastavalt igale veebilehe visiidile. Seega muutusid reklaamid personaliseerituks ning dünaamiliseks. [4]

Selleks et siduda andmeid reklaamide tootlikuse ja sihtruppide vahel loodi veebilehtedele pikslid ja küpsised, mis lehe HTML (*Hypertext Markup Language*) koodi integreerimisel said hakata koguma andmeid internetikasutajate ning -sessioonide kohta. [5]

Andmete kogumine, analüüs ning töötlemine on loonud viimase kümnendi jooksul soodsa pinnase automatiseeritud turunduse arenguks. Selline turundusmeetod võimaldab optimeerida reklaamipinda nõudvatel osapooltel iga nende reklaami kuvamise, vaatamise, kui ka kliki hinna. [5]

3.2 Teema päevakohasus

Paljudes tööstusvaldkondades muutub üha kasulikumaks minna üle automatiseeritud süsteemidele, säästes sellega kulusid ning optimeerides töö efektiivsust. Treenides andmete peal mudeleid või neid analüüsides usaldatakse järjest enam otsused automaatsete süsteemide teha. [6]

Ameerika Ühendriikides panustavad pea kolmveerand ettevõtetest 2019 ja 2020 aastate näitel enamuse turunduseelarvest digitaalsesse turundusse. 2019 aastal möödus esmakordselt ka digitaalturunduse kulutuste osakaal traditsioonilise meedia omast, millest üle poole (53,1%) on displeireklaamid nagu näiteks pildid või videod. Displeireklaamid on automatiseeritud turunduse leivanumber, ligi 85% kõikidest sellistest reklaamidest internetis on näidatud tänu automatiseeritud süsteemidele. [6]

Võrreldes televisiooni- või raadiokanalitega, läbi mille reklaamimiseks 2019 ja 2020 aastal ettevõtted vähem kulutasid (2019 mõlema kombineeritud langus ligi -3% ning 2020 langus ligi -2,5%), on digitaalse turunduse trend kasvav (2019 +13,2% kasv ja 2020 + 9,5% kasv) [6].

Match2One turundusplatvormi näitel on Rootsi transpordi firma Flygbussarna, kes 2018 aastal peale 30 aastat tegutsemist internetiturundusse otsustas investeerida, teeninud investeeringust tagasi 800% tänu automatiseeritud turunduse efektiivsusele [7].

Kui muidu on automatiseeritud turunduse teenust brändidele pakkunud selleks ettenähtud reklaamiagentuurid, siis trend on kaldunud nö majasiseste ehk ise kasutatavate platvormide poole. 2018 aasta andmetel ostsid 45% brändidest ise reklaame

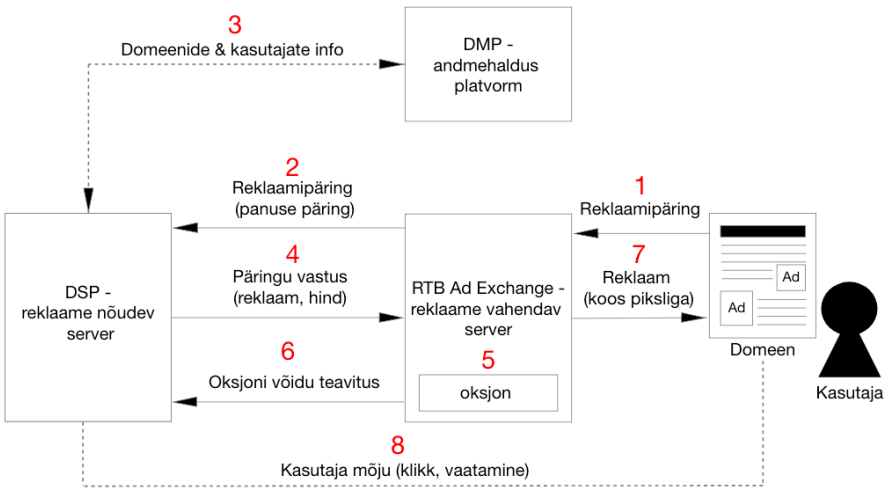
automatiseeritult ning aastaks 2022 ennustatakse tõusu ettevõtte siseste lahenduste kasutamisel kuni üle poolte brändide ehk 62% poolt [2].

3.3 Automatiseeritud turunduse eelised

3.3.1 Eelised reklaame nõudvatele ettevõtetele

Automatiseeritud turundus võimaldab teha reklaamipindadele pakkumisi reaalajas, töödeldes samal ajal läbi andmeid eelnevate reklaamide tootluse kohta, näiteks reklaami klikkide või vaatamiste arvu kohta [8]. Lisaks annab see suure kontrolli eelarve üle ja hõlbustab kampaaniate efektiivsust monitoorida. Arvutades müügitulu kokku ning vaadates paljud neist on läbi turunduskampaania tulnud, on võimalik arvutada välja iga kliki ning isegi reklaami näitamise hind (CPC - *cost-per-click*, CPM - *cost-per-mill* ehk hind tuhande näitamise kohta) [9].

Üheks põhiliseks automatiseeritud turunduse meetodiks on RTB (*Real Time Bidding*) ehk reaalajas oksjon [8]. Iga veebilehe külastus pannakse müüki ehk oksjonile, mis tähendab, et vähem kui 100-200 millisekundi jooksul, kui veebileht laeb, toimub lehel olevale reklaamipinnale oksjon, mille kõige kõrgema hinna pakkuja võidab [10]. Kogu protsessi jooksul saavad pakkumise reklaame vahendavalt serverilt (*Ad Exchange*) kõik oksjonil osalevad reklaame nõudvad DSP serverid (*Demand Side Platform*), kes algoritme kasutades otsustavad millise hinnaga ja millist reklaami sellele konkreetsele lehekülastusele näidata tahavad (Joonis 1). [12]



Joonis 1. Reaalajas oksjon [12]

Üheks kõige suuremaks reklaamide vahendajaks on Google poolt loodud teenus Authorized Buyers (kunagine Google DoubleClick AdX), millega iga oksjonil osalev DSP server suhtleb [11]. Reklaamipäringu saamisel võetakse arvesse näiteks domeen, millel külastaja parasjagu on ning kust reklaamipäring tuleb. Analüüsitakse näiteks kas sellelt domeenilt on tulnud palju klikke eelnevatele reklaamidele või kui paljud kasutajad on ostnud toodet sellel leheküljel näidatud reklaamide kaudu. Algoritm arvutab milline oleks võimalikult odav kuid reaalne hind, et oksjon võita või siis otsustab, et seda reklaamikohta ei ole tarvis ja loobub. [12]

Tänu andmetele iga reklaami näitamise kohta on ettevõttel võimalik luua täpne ning kalkuleeritud ülevaade turundusprotsessist, sest kõik tulemused on reaalajas nähtavad ning analüüsitavad [4]. See on ka suurim eelis, mida automatiseeritud turundus omab traditsioonilise turunduse ja ka teistsuguste interneti turunduskanalite ees. Lisaks annab see võimaluse suunata reklaame spetsiifilistele kasutajagruppidele ja domeenidele [12].

3.3.2 Eelised internetikasutajatele

Ameerikas loeb uudiseid 43% inimestest internetist, olgu selleks sotsiaalmeedia või uudisteportaal [13]. Sarnane trend on ka Eesti loetuima ajalehe „Postimees“ lugejatega [14]. Põhjuseks on info kättesaadavus, mitmekesisus ning tegevuse efektiivsus. Kuna kõik toimub kiirelt ja mugavalt ei jõua kasutaja huviorbiiti kuidas leheküljed ennast tegelikult majandavad.

Selline arusaam tuleneb ka üldisest teadmatusest internetiturunduse tausta ning reklaaminduse osapoolte väärtushinnangute kohta. Ei ole juhitud internetikasutajatele piisavalt tähelepanu, et isegi kui pealtnäha tundub kõik tasuta ja maksma ei pea, siis veebilehtede sisu tarbimine ei ole tasuta ja suur osa sisutootjaid sõltub näiteks reklaamipinna müügi eest saadud rahadest. [15]

Küll aga teadlikumad kasutajad tunnevad veebis olles muret enda andmete üle ning arusaam, et neid andmeid kasutatakse reklaamide personaliseerimiseks on neile valdavalt vastuoluline. [8]

Teisalt on välja toodud, et usaldusväärsete ning kvaliteetsete brändide reklaamide nägemine veebis meelestab kasutajaid positiivselt ning selles nähtakse pigem kasu. See on paradoksaalne situatsioon, kuid samas tõestab, et sihtgruppidele suunatud

automatiseeritud turundus on kasutajate jaoks vastuvõetav. Personaliseeritud toodete reklaamid ning lehtede sisu loovad kasutajatele parema tunde kui asjakohatud reklaamid.[8]

Et sellised turundus strateegiad saaksid eksisteerida peab olema pidev koostöö ning kommunikatsioon turundustehnoloogia ettevõtete, seadusandlike organite ning kasutajate vahel [16]. Veebilehtedel on tähtis informeerida lisaks küpsiste kohta kasutajaid ka tegevuse eesmärgi kohta, näiteks teavitada et sellest sõltub veebilehe sissetulek. Kasutajad ei taha maksta reklaamivaba interneti eest, kuid nende jaoks on oluline, et seda saaks usaldada [15], [17]. Selle tagab ettevõtete läbipaistvus andmekasutus ja turundus metoodikates ning kasutajate õigus sekkuda enda andmete kogumisse.

Kuna negatiivsust tekitavad asjakohatud ja ebakvaliteetsed reklaamid, siis on kompromiss kasutajatele kasutada reklaame filtreerivaid rakendusi, mis lasevad defineerida millised reklaamid tulevad temani ning millised mitte. Lisaks on olemas rakendused, mis näitavad milliseid ning kui palju andmeid kolmandad osapooled kasutaja kohta on kogunud. [18]

3.4 Õigused ja piirangud

Järjest enam pööratakse tähelepanu ka sellele kuidas ning millised andmed ettevõtete valduses ringlevad. Kiire tehnoloogilise arenguga turundusvaldkonnas ei ole sugugi lihtne alati kindlaid piire tõmmata ning tihe töö seadusandlike organisatsioonide ning tehnoloogia ettevõtete vahel toimub pidevalt [19].

Sellest kuidas 2018 aastal välja tulnud GDPR (*General Data Protection Regulation*) on mõjutanud internetiturundajaid ning tarbijaid, on kirjutatud palju, kuid peamine mõte on üks – iga indiviid peab omama kontrolli enda kohta salvestud andmete kohta [25]. Leheküljed, mis koguvad andmeid nii enda kui ka kolmandate osapoolte jaoks, saavad seda teha ainult kasutaja nõusolekuga, milleks kuvatakse külastajatele kõik reeglid nii küpsiste kui ka pikslite kohta ja samuti antakse võimalus neist loobumiseks. Oluline on, et selgitatakse sealjuures ka nende vajalikkust ning kasutamise eesmärki. [24]

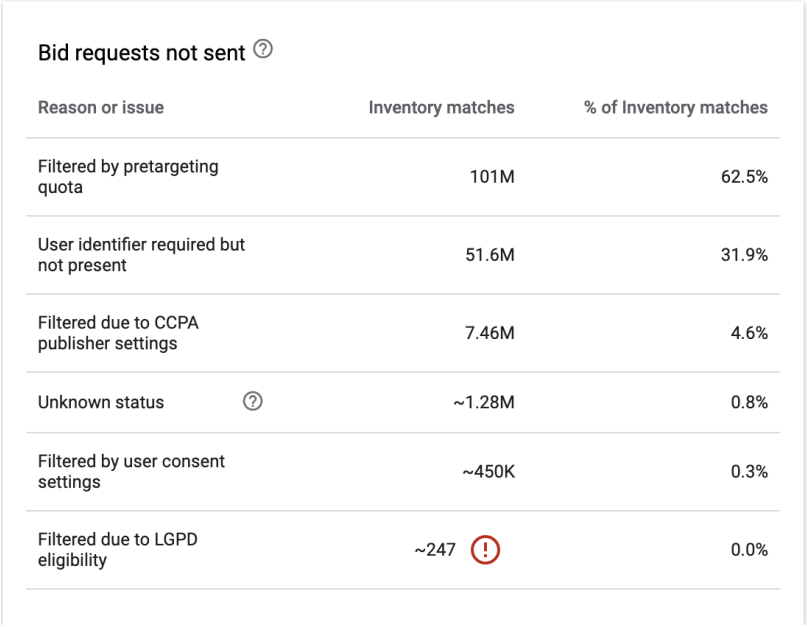
Sarnase eesmärgiga on loodud USA's CCPA (*California Consumer Privacy Act*), mille suurim erinevus GDPR'iga on see, et kasutajal on õigus kõiki toiminguid andmete osas

(kustutamine, nende kogumisest keeldumine või juba kogutud andmete nägemine) seadistada mitte enne vaid tagant järele [23].

Lisaks veebilehtedelt otse info hankimisele ringleb automatiseeritud turunduses veel muul moelgi andmeid. Reaalajas toimuvatel oksjonitel vahendavad reklaame pakkuvad serverid andmeid reklaamipäringute kaudu DSP serveritele, mida veebilehed kasutaja nõusolekul kogunud on. [23]

Selleks et kasutajal ei katkeks ühendus enda kohta salvetatud andmetega, on loonud reklaamitehnoloogia ettevõtted koostöös IAB (*Interactive Advertising Bureau*) Euroopa filiaaliga läbipaistvus ja nõusoleku raamistiku (*Transparency and Consent Framework* ehk TCF). Kuna andmeid käsitlevaid osapooli on palju, nõuab raamistik reklaamipinda müüvatel lehtedel luua võimaluse kasutajal näha kelleni tema andmed välja jõuavad ning soovi korral saata teade nende kustutamise osas. [20], [24]

Lisaks filtreerivad paljud reklaame vahendavad serverid reklaamipäringud (Joonis 2), mis ei vasta GDPR või CCPA nõuetele ja ei saada neid DSP serveritele edasi või muudavad saadetud andmed vastavalt nõuetele [22]. Google Authorized Buyers keelab lisaks ka andmete, mis reklaamipäringutega kaasas on, edasi müümist. Lubatud on neid andmeid aga kasutada oksjonitel osalemiseks ja strateegiate optimeerimiseks [11].



Joonis 2. Ekraanitõmmis filtrist Google Authorized Buyers keskkonnas

4 Automatiseeritud turunduse lähtekohad

Peatükk iseloomustab automatiseeritud turunduse juurde kuuluvaid tehnoloogiaid. Esimene alapeatükk kirjeldab andmehalduse komponente ning rolli. Teises ja kolmandas on välja toodud kuidas OpenRTB standard ja Google Authorized Buyers reklaame vahendav server mõjutavad automatiseeritud turunduse osapoolte kommunikatsiooni ning arhitektuuri. Viimaks tutvustatakse põhilised tehnilised lähtekohad, mis on DSP serveri arendusel olulised silmas pidada.

4.1 Andmehalduse roll automatiseeritud turunduses

Andmed on alus automatiseeritud süsteemide toimimiseks, nii on ka automatiseeritud turunduses. Andmeanalüüs võimaldab optimeerida turundusse panustatud kulutusi ning tõsta reklaamide tootlikust [12].

Veebis kogutud andmeid käsitletakse vastavalt sellele, kes neid omab ja kes kasutab. Peamiselt jagatakse need kolme kategooriasse. On olemas esimese osapoole andmed, mida kogub ja omab iga ettevõte enda leheküljelt (või ka väljaspoolt veebi) ise. Kuna see on iga ettevõtte enda käsutuses, võib selle kaudu luua väga detailse ning põhjaliku profiili enda kliendist. [26]

Veel on kasutuses nö teise osapoole andmed, mis on mõne muu ettevõtte andmebaasis, kuid informatsiooni jagamiseks loovad erinevad osapooled koostöösuhteid, näiteks võib lennufirma paluda ülevaadet autorendi klientuuri osas, kuna tõenäoliselt on sihtgrupp väga sarnane. [26], [27]

Viimane kategooria on kolmanda osapoole andmed, mida koguvad firmad võimalikult laiahaardeliselt, et need struktureerida ja sellest loodud info pealt kasu teenida [26]. Sellised on tavaliselt suured andmehaldus platvormid (DMP – *data mangment platform*), mis pakuvad teenust erinevatele automatiseeritud turunduse osapooltele, näiteks DSP serveritele internetikasutaja sihtgruppide kohta [5].

Andmeid koguvad aga automatiseeritud turunduse kõik osapooled. Veebilehed, kes reklaamipinda müüvad küsivad kasutajatelt luba koguda sessiooni andmeid kasutades selleks ettenähtud küpsiseid või piksleid. [26] Üldiselt on need kolmanda osapoolega seotud ning saadetakse päringuna reklaame vahendavatele serveritele (*Ad Exchange*) või andmehaldus platvormidele. Kuid samade vahenditega kogutakse ka esimese osapoole andmeid. [28]

Lehekülje HTML'i lisatakse spetsiaalne kood mis reageerib vastavalt kasutaja tegevusele veebilehel. Laiemalt on levinud küpsised ning pikslid. HTML'is näevad need välja `<script>`, `<img>` või `<iframe>` kujul ning tavaliselt saadavad pikslid koos informatsiooniga ka reklaamipäringuid ja küpsised salvestavad informatsiooni kasutaja arvutisse. [12]

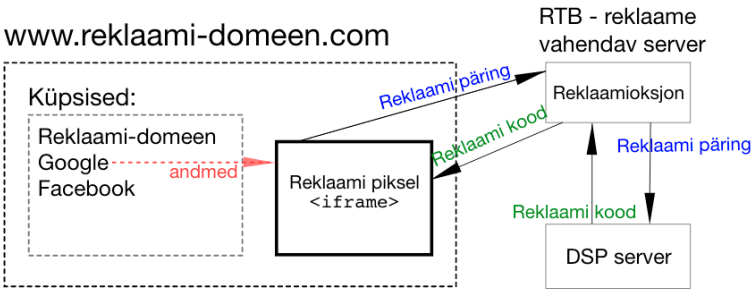
Automatiseeritud turundamist praktiseerides tuleb suur kogus andmeid ka päringute kaudu reaalajas toimuvalt oksjonilt DSP serveritele [23]. Need andmed sõltuvad sellest, mida antud külastatav domeen on lubanud kasutaja sessiooni kohta koguda. Seal võib olla informatsiooni näiteks domeeni kohta, kasutatava seadme kohta ja kasutaja geolokatsiooni kohta. Lisaks on päringus ka kirjas minimaalne hind, et reklaami oksjonil osaleda. [12]

Ühtlasi kogutaksegi andmeid, mis kirjeldavad reklaamide tootlust, näiteks CTR (*click-through-rate* ehk reklaami klikkide osakaal) ja vaatamiste arv, mida DSP algoritmid ka reaalajas kasutavad, et oksjonil panustamiseks hinda arvutada [28]. Teisalt korjatakse ka andmeid, mis kirjeldavad veebisessioonide semantilist poolt ning mis aitavad iseloomustada kas domeene, millel reklaame näidatakse või kasutajaid endeid. [26]

4.2 OpenRTB protokolli suunised

OpenRTB on mittetulundusühingu IAB poolt välja töötatud automatiseeritud turunduse raamisitik reklaamioskjonitel osalejatele. OpenRTB eesmärk on kasvatada reaalajas oksjonite kasutust turundustööstuses võimaldades selleks avatud standardeid, mis lihtsustavad kommunikatsiooni reklaami pakkuvate serverite ning reklaami nõudvate serverite vahel. Oluline on et kõik osapooled mõistaksid just kui ühte keelt. Esimene OpenRTB API kasutamise spetsifikatsioon arendati välja 2010 aastal. Kõige uuem versioon on OpenRTB 3.0, mis valmis aastal 2020. [29]

Automatiseeritud turundus toimub läbi serverite omavahelise suhtluse, kus osalevad reklaame vahendav server (*Ad Exchange*) ning DSP server suheldes läbi REST API [10]. Kui kasutaja internetis veebilehe avab, millel asuvad ka reklaamideks ettenähtud kohad, siis saadab reklaamikoha HTML koodis olev piksel päringu reklaame vahendavale serverile (Joonis 3), millel on omakorda ühendus DSP serveritega [5]. Iga kvalifitseeruv server saab vahendaja poole pealt HTTPS protokollil põhineva POST reklaamipäringu (*Bid Request*) koos päringu kehas olevate andmetega kasutaja, domeeni, miinimum osalemis hinna ja palju muu kohta (Lisa 1 – Reklaamipäringu JSON) [30].



Joonis 3. Reklaamipinda pakkuva domeeni piskli päring

Kiire andmete töötluse järel peab DSP server päringule vastama otsusega (*Bid Response*) (Lisa 2 – Päringu vastuse JSON). Vastuse kehas on välja toodud, kas server osaleb oksjonil reklaamikoha peale või mitte ja kui palju sellele panustab. Oluline on veel vastuses reklaam ise, mis esineb *AdMarkup* kujul. Olemuselt meenuteb see HTML'i kuhu on sisestatud näiteks pilt ning link, kuhu reklaam klikkides viib. Üldiselt on reklaami koodi lisatud ka jälgimis (1 korda 1 suuruses) piksel, mis annab DSP serverile märku kui reklaami nähakse. All on toodud näide reklaami koodist veebilehel, esimene osa sellest on jälgimispiksel, teine ja kolmas osa on edasiviiv link ja pilt. [30]

```
<img width=\"1\" height=\"1\"
src="https://biddergoogle.bytelogics.io/view?bidId=1-
1606494996.019018-2018&cost=${AUCTION_PRICE}\" style=\"display:none;\"
/>
<a href=\"https://biddergoogle.bytelogics.io/click?bidId=1-
1606494996.019018-2018\">
<img src=\"https://biddergoogle.bytelogics.io/images/ad.gif\"
width=300 height=250/>
</a>
```

Joonis 4. Reklaami kood – piksel, link, pilt

Levinum andmeedastus formaat reaalajas oksjonite serverite vahel on JSON (*JavaScript Object Notation*) ning reklaame vahendav server peab päringu päises märkima vastavalt „*Content-Type:application/json*“. Põhjus, miks päisesesse selle kohane märge läheb on selles, et kasutusel on ka Protobuf formaat ja ka binaarkoodis saadetav päringu keha, kuid need on vähem levinumad. [30]

OpenRTB võimaldab vahendada andmeid ka kompresseeritud ning kodeeritud viisil. Selline lähenemine mõjutab oluliselt serverite vahelise suhtluse ning päringute kiirust. Reklaame vahendav server saab märku anda võimalusest suhelda kompresseeritud viisil märkides päisesse „*Accept-Encoding: gzip*“ ja pakkumisi vastu võttev server on sellisel juhul kohustatud märkima „*Content-Encoding: gzip*“. Lisaks peab vahendav server lisama päringu päisesse kasutatava OpenRTB versiooni, mis informeeriks sellega suhtluses olevaid servereid, näiteks „*X-openrtb-version: 2.5*“. [30]

4.3 Google Authorized Buyers suunised

Google Authorized Buyers on reklaame vahendav teenus, mis annab võimaluse turundajatel osaleda oksjonitel ning panustada reklaamikohtadele automatiseeritult [11].

Et teenust kasutada tuleb DSP serverit arendaval ettevõttel Google esindajatega ühendust võtta, kes testivad ühendusse astuva reklaame nõudva serveri turvalisust ning võimekust. Google jaoks on oluline, et URL mille kaudu serverite suhtlus toimub oleks turvaline ehk kasutaks HTTPS protokolli. Selleks tuleb lisada DSP serverisse SSL sertifikaat. [31]

Lisaks kontrollitakse suhtluses oleva serveri füüsilist asukohta, et optimeerida suhtlust Google serveritega. Kui soovitakse reklaamida näiteks nii ida pool kui lääne pool USA’d, siis peab olema server mõlemas kohas. Testitakse ka serveri võimekust mahulisele reklaami päringute hulgale vastata, sellega pannakse paika maksimaalne QPS (*query-per-second*) ehk päringud sekundis, mida teenus saadab. [31]

Google soovib saada päringutele vastused 80 kuni 870 millisekundi jooksul, peale mida loobutakse vastust ootamast ning reklaami nõudev server oksjonikohta ei saa. Kui Google tuvastab probleeme serveriga ühenduses, näiteks suur osa päringuid ei vasta õigeaegselt, langetavad nad QPS’i kuni ühendus normaliseerub. [31]

Google teenus saadab päringuid sellega ühenduses olevatele DSP serveritele, kasutades selleks nende enda Authorized Buyers RTB protokolli või eelnevas peatükis kirjeldatud OpenRTB protokolli [31]. Peamine erinevus on selles, et Google enda protokoll on samuti Protobuf formaadis. Erinevalt JSON'ist ei ole Protobuf tekstil põhinev ning võimaldab defineerida soovitud andmeskeemi ning väljade tüüpe [32].

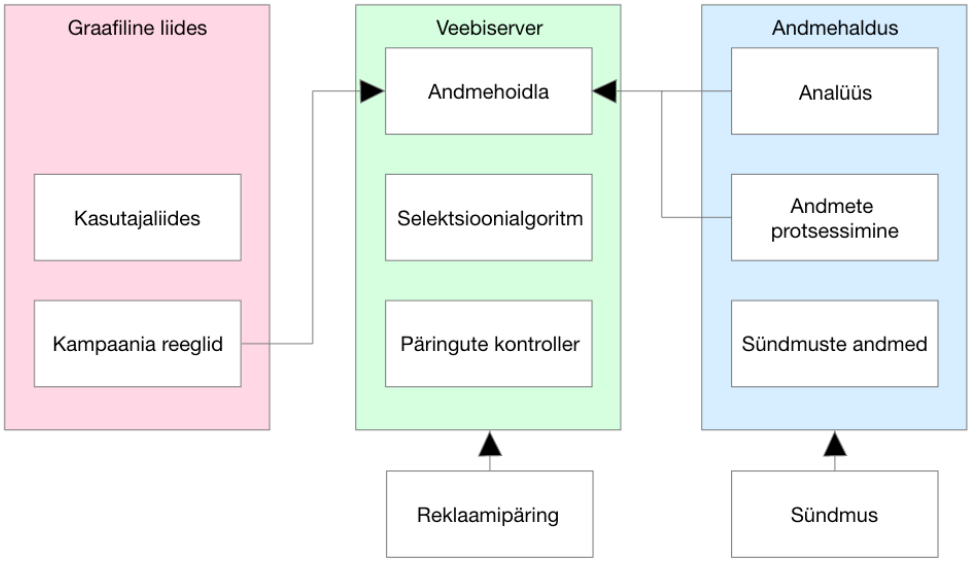
Google Authorized Buyers teenus võimaldab konfigureerida nii mõningaidki parameetreid enne oksjonil osalemist, mida nimetatakse eelsihtimiseks (*pretargeting*). See on abiks õige turundus taktika leidmisel ning sihtgrupi fokusseerimisel. Teenuse portaalis on võimalik valida millises geolokatsioonis soovid reklaame näidata, lisaks on võimalik määrata millistel lehtedel seda teha. Näiteks saab veel määrata platvormi, kus reklaame näidata: mobiilirakendustes, veebisbrauserites või mõlemates ja ka defineerida kindlad domeenid, millelt reklaamipäringuid soovitakse. [31]

4.4 DSP ja automatiseeritud turundus

DSP on platvorm, mille abil saavad reklaame nõudvad ettevõtted automatiseeritult panustada reklaamikohtadele reklaamioksjonitel. DSP server kuulub automatiseeritud turunduse hajussüsteemi, mis võtab vastu reklaame vahendavate serverite päringuid ning vastab neile. [5]

Reklaame valiva DSP komponendid on tavaliselt (Joonis 5):

- Graafiline liides kampaaniate manageerimiseks.
- Veebiserver, mis võtab vastu reklaamipäringud ja neid selekteerib.
- Üks või mitu andmehoidlat. [33]



Joonis 5. DSP rakenduse komponendid

4.4.1 DSP suhtlus hajussüsteemis

DSP serveritele on omased REST arhidektuuril põhinevad API kontrollerid, mis reklaamide päringuid vastu võtavad ja neile vastused tagasi saadavad [33]. Üldiselt nõuavad reklaame vahendavad serverid vastust 100 millisekundi ringis, seega minimaalne latentsus on kriitiline nõue [12]. Selleks peab serveri osa, mis vastab päringule reklaami koodi (*AdMarkup*) ja hinnapakkumisega olema võimeline päringuid sõeluma ja neid struktueerima väga kiirelt [33].

Madala latentsuse saavutamiseks tuleb lisaks programmi koodile silmas pidada füüsilisi ja riistvaralisi piiranguid. Selleks soovitatakse, et DSP serverid asuksid geograafiliselt lähedal reklaame vahendavatele serveritele. Soovituslik on ka veebiserveri programmi koormuse jaotur (*load balancer*), mis hajutaks sisse tulevaid päringuid. Oluliseks peetakse veel, et päringute alustatud serveriprotsessid skaleeruksid vastavalt KPI'dele (*key performance indicator*) [33]. KPI'de hulka kuuluvad indikaatorid nagu protsessori töövõime, paralleelsete protsesside arv, protsesside kasutatav mälu, andmete kirjutamis kiirus ja lugemiskiirus jne. Sellised parameetrid pannakse vastavalt serveri riistvaralistele näitajatele paika serveri programmi konfiguratsioonifailides. [34]

Hea tava hoida sisse tulevaid päringuid kontrolli all, on integreerida serveri API lõpp-punkti maksimum päringute kontrollimise klapp (*throttle*), mis ei lase päringu sisu

sõeluda ja vastab kohe 204 (sisu puudub) HTTP koodiga, mis tähendab, et DSP server ei soovi oksjonil osaleda. Kuna reklaame vahendavate serverite jaoks on oluline hoida kommunikatsioonikanal üleval, on parem vastata ülekoormuse korral mitte osalemisega, kui lasta serveril päringutest umbe joosta ja 500 (serveri viga) HTTP koodiga vastata. [31]

4.4.2 DSP server ja andmehaldus

Põhilise osa DSP serveri tööst moodustavad selektsioonialgoritmid, mis reklaamipakkumiste otsuseid langetavad [36]. Küll aga ei tee algoritmid otsuseid randoomselt vaid nende sisendid sõltuvad eelnevalt kogutud andmetest. Andmed, mida server kasutab jagunevad tavaliselt kiireloomuliselt kättesaadavateks ning sügavamal analüüsil põhinevateks suurandmeteks. [35]

Väiksema latentsuse saavutamiseks on DSP serverite ja andmebaasi vahel üldiselt vahemälu kiht, mis talletab ning loeb kiiresti ajakriitilist informatsiooni. Sellised ajakriitilised andmed võivad olla näiteks eelnevate reklaamioksjonite hinnad, reklaamide klikid, vaatamised jne. [35]

Paraku pole kõik info kogu aeg vahemälust kättesaadav ja selleks on vähemalt sama oluline andmebaasi enda disain. Selleks on DSP serveritel nii enda andmebaas kuid on tavaks ka ühendus DMP platvormiga, mis andmeanalüüsil põhinevat informatsiooni jagab. DMP võib olla ka ettevõtte sisene kuid ostetakse sisse ka välise teenusena. [35]

Analüütilist andmehoidlat ehk DMP'd ei kasutata üldiselt reaalajas iga päringu puhul vaid see on ette nähtud suurema hulga andmete töötlemiseks [12]. Selle abil on võimalik luua näiteks kasutaja segmente, domeenide kategooriaid või muid turunduskampaaniaid iseloomustavaid gruppe. Protsessi käigus saadud andmeanalüüsi tulemusel optimeeritakse DSP tööd soovitud tulemuste saavutamiseks. [2]

Kuna andmehulgad on suured ning info liikumine peab olema kiire ja efektiivne, siis on oluline teada, milliseid standardeid reklaame vahendavad serverid kasutavad. Näiteks OpenRTB reklaamipäringu formaadi põhjal saab andmebaasi tabelite disainimisel selekteerida, millised on kriitilised andmeväljad, mis on tarvis salvestada. Andmebaaside disaini puhul on peamine silmas pidada mälu kasutuse efektiivsus ja samal ajal jõudlust. [35]

DSP serveri arendusel on seega tähtis disainida andmebaasid ning andmetöötlus meetodid nii, et oleks lihtne kõik andmed omavahel ühendada. Kuna need võivad pärineda mitmest kohast – mitu DMP'd, DSP enda andmebaas jne. Peab olema suutlikus viia omavahel kokku reklaamid ja nende klikid, vaatamised ning mis kõige olulisem, reklaamide konversioonid ehk millistest reklaamipakkumistest on lõpuks ettevõte tulu teeninund. [35]

5 Bytelogics DSP

Käesolevas peatükis tutvustatakse Bytelogics DSP rakendust. Tuuakse välja Bytelogics DSP puudused ja kirjeldatakse lahenduste leidmise protsessi.

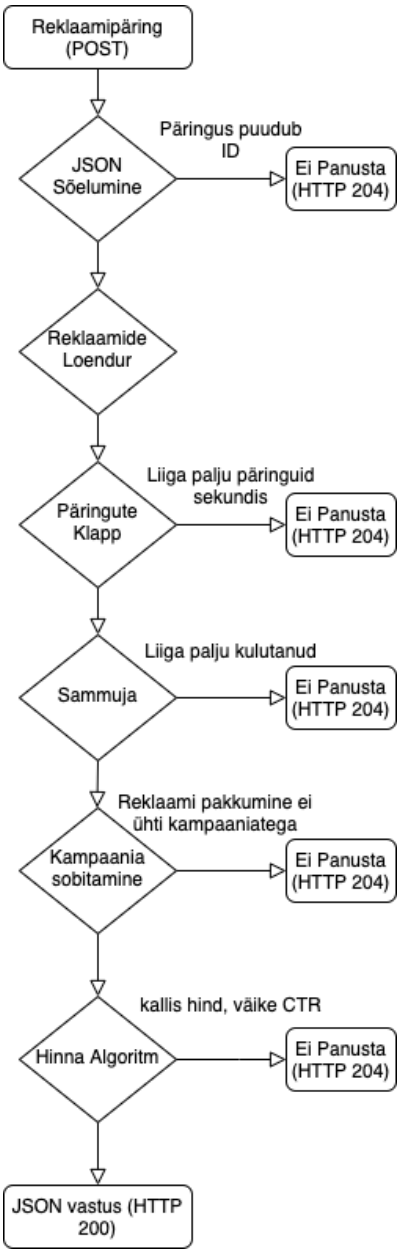
5.1 Bytelogics DSP ülevaade

Bytelogics DSP töötab virtuaalmasinas Google Cloud pilveteenuse abil. Seal on see Google Authorized Buyers reklaame vahendatavatele serveritele geograafiliselt lähedal ja see võimaldab minimaliseerida latentsust serverite kommunikatsioonis.

Bytelogics DSP on hajussüsteem, mis järgib teenusepõhise arhitektuuri mudelit. See on kirjutatud PHP programmeerimiskeeles MVC (*Model-View-Controller*) raamistikule tuginedes. Kogu süsteem koosneb kesksest MySQL andmebaasist ning sellega ühenduses olevast UI'st (*User Interface*) ehk kasutajaliidesest ning *Bidder* (panustamis) serverist. UI võimaldab DSP teenuse kasutajal kampaaniaid hallata ning *Bidder* server on mõeldud kogu automatiseeritud turunduse loogika jaoks. *Bidder* server on UI lahendusest eraldi ning selle eesmärk on suhelda Google Authorized Buyers reklaame pakkuva serveriga rakendusliidese ehk API (*Application Programming Interface*) abil.

Reklaamipäringuid võtab *Bidder* serveris vastu kontroller, mis suhtleb mudeli abil nii MySQL andmebaasi kui ka serveri vahemälus töötava Memcached teenusega. Memcached salvestab vahemällu kiiresti vajaminevad andmed ja tagab sellega kiire andmetöötluse. Andmete puudumisel vahemälus pärib serveri mudel andmed siiski kõvakettal olevast andmebaasist. Päringuid töötlev *Bidder* klass on loodud *singleton* disainil, mis võimaldab tervel rakendusel korraga kasutada mälus olevaid sisse tulnud reklaamipäringu andmeid.

Bytelogics DSP tähtsaim funktsionaalsus ongi *Bidder* serveril. Selle eesmärk on osaleda automatiseeritud turunduses filtreerides andmete ning algoritmide abil sissetulevad reklaamipäringud ning arvutada turunduskampaaniale vastav reklaamipakkumine. Protsessi aitab paremini mõista Joonis 6, millel on iga filtri või algoritmi töö lihtsustatud viisil illustreeritud.



Joonis 6. Reklaamipäring läbi *Bidder* klassi selektsioonifiltrite

5.2 Bytelogics DSP puudulikkus

Bytelogics DSP esmane ehk MVP (*Minimal Vibale Product*) versioon töötas test kampaaniate peal, kasutades selleks limiteeritud ressursse (väike QPS ja turunduseelarve), et vigade ilmnemisel suuri kahjusid ei tekiks. Selektsioonifiltrid kasutasid andmeid reklaamide tootluse kohta nagu klikid, vaatamised jne.

Küll aga panustas DSP server reklaamidele üsna kinnisilmi, sest pakkumised tulid kõikidelt ettejuhtuvatelt domeenidelt, kes reklaamipinda pakuvad. Selline praktika ei ole

tavaliselt aga vettpidav ning kasutab eelarvet ebaotstarbekalt. Näiteks võivad paljud domeenid, mis reklaamipinda müüvad olla ebakvaliteetsed ja näidata oma lehel korraga mitukümmend reklaami, mille seast silma paistmine ei ole võimalik. Või siis on reklaamitava toote sihtgrupp koondunud hoopis teistele domeenidele ja seeläbi läheb osa ressursse tühja kohta turundamiseks. Bytelogics DSP lahendusel puudusid valged listid (*whitelist*) domeenidest, mis on kõige parema tootlikkusega ning millele panustatakse kas lihtsalt rohkem või koguni 100 protsendiliselt. [38]

Seega oli projekti plaanis teada, et edasised arendused olid vajalikud ja senisel MVP tasemel ei olnud automatiseeritud turundus rakendatud piisavalt efektiivselt. Turunduse optimeerimiseks ja *whiteliest* koostamiseks oli tarvis rohkem teada ettevõtte sihtgrupi kohta ning veebilehtede kohta, mida nad külastavad. Selleks oli aga tarvis andmeid, mida analüüsida.

Andmed olid kättesaadavad ettevõtte kodulehelt kasutades esimese osapoole piksleid ning lisaks tuli andmeid ka reklaamipäringutega. Samas puudusid selleks spetsiaalselt välja arendatud andmekogumis kanalid ja andmebaasi disain. Bytelogics DSP serveril puudus DMP, mis sisestaks DSP'le reklaamiotsuste tegemiseks analüüsitud andmeid.

Probleemist lähtuval seati eesmärgiks luua DMP'le sarnase funktsionaalsusega andmekogumis ning -analüüsi täiendused olemasoleva DSP lahenduse *Bidder* serverisse. Fookus oli luua kanal andmete kogumiseks ja kogutud andmete analüüsil teha *whitelist* domeenidest, mida server kasutaks, et optimeerida ja muuta turunduskampaaniad efektiivsemaks läbi teadlikult domeenidele ja seeläbi täpsemale sihtgrupile reklaamimise.

5.3 Täienduste protsessi kirjeldus

Täienduste välja mõtlemine, metoodika valimine, nende valmis programmeerimine ja integreerimine oli töö autori teha ning eeldas enne arenduse alustamist põhjalikku uurimist DSP serverite ja RTB oksjonite tavadest [33]. Lisaks vajas süvenemist automatiseeritud turunduses kasutatavate standardite ning tehnoloogiate dokumentatsioonidesse. Kuna töö autoril puudus pikaajaline kogemus DSP rakendustega töötamisel, nõustas autorit ka projekti vanemarendaja, kes autorit lahenduste integreerimisel juhendas.

Automatiseeritud turunduse ja sinna juurde kuuluvate RTB reklaamioksjonite praktikaid tutvustasid spetsiaalselt arendajatele mõeldud dokumentatsioonid nagu näiteks Google Authorized Buyers suunised [31], OpenRTB standard [30] ja vabavaralised näited koos kirjelduste ja õpetustega [38]. Nendest dokumentatsioonidest sai peamised teadmised tehniliste eelduste osas ja reklaamipäringute sisu ning atribuutide kohta, mis on väärtuslikud salvestamiseks ning hiljem analüüsis kasutamiseks.

Bytelogics DSP juurdearendusel oli äärmiselt oluline teha selgeks ka MySQL andmebaasi disaini põhipraktikad, et vältida vigu rakenduses, kus igasugune liigne tegevus raiskab päringutele vastamise aega ning seeläbi ka turunduskampaania eelarvet. Selleks saadi vajalikud teadmised O'Reilly keskkonnas olevalt MySQL jõudluse optimiseerimiseks mõeldud õpetustest [37]. Tarvis oli kasutada MySQL päringukeelt ka analüütika ja äriteabe vahendina, mille abil kogutud andmetest vajalikk informatsioon kätte saadi. Kuna autoril varasem kogemus MySQL keele analüütilise kasutamise osas puudus, oli abiks selle alane Udemy veebikoolitus, kus kõik vajalikud põhitõed enne läbi sai töödatud [39].

Kuna kogu serveri töö on väga ajakriitiline, siis nõudis arendusprotsess iga serveri koodi lisatud komponendi töö ajalist testimist, selleks kasutas autor Xdebug Profiling tehnoloogiat koos tööriistaga *Analyze Xdebug profiler snapshot*, mis asus arenduskeskkonnas PHP Storm. Samuti kasutas töö autor serveri töö testimiseks Apache Benchmark tehnoloogiat, et testida serveri koormustaluvust saates sellele korraga suur arv päringuid. Testimine toimus arendustööga paralleelselt.

Lõpliku analüüsi tulemusel saadud domeenide *whitelist*'i filtreeriva funktsionaalsuse rakendas autor esialgu *Bidder* serveri koodi, tehes andmebaasi tabelisse reklaamipäringu saabumisel domeene võrdleva päringu, kuid antud lahendus võttis päringu töötlemisel liiga palju aega. Seega pidas autor oluliseks lisada serveri kiiruse säilitamiseks domeenide *whitelist*'i Google Authorized Buyers *pretargeting* konfiguratsiooni kasutades selleks Google API'd. Sellisel juhul toimub domeenide selektsiooni enne Bytelogics DSP'le päringu saatmist.

6 Bytelogics DSP täiendused

Peatükk loob ülevaate Bytelogics DSP täiendustest. Kirjeldatud on valitud metoodika ning kasutatud tehnoloogiad. Peatükis on toodud välja järeldused ning täienduste tulemused. Lisaks annab peatükk ülevaate potentsiaalsetest edasiarendustest.

6.1 Metoodika

Bytelogics DSP puudulikkuse parandamiseks lähtus autor käesoleva töö 3. peatükis välja toodud automatiseeritud turunduse ja DSP serverite arhitektuurilistest lähekohtadest [12], [35]. Peamiselt on täienduste aluseks tehnilised ja metoodilised põhitõed – reklaamipäringutest saadud andmete iseloom, mida kirjeldab OpenRTB standard [30] ning Google Authorized Buyers dokumentatsioon [31], mida kasutab autor päringute sõelumise funktsionaalsuse täiendamises ning andmebaasi disaini puhul peatükkides 6.2 ja 6.3.

Turunduse optimeerimis eesmärgi saavutamiseks kasutatakse sellekohaste andmete kogumist ning töötlust. Andmete kogumise tehnoloogiaid arendades säilitatakse serveri ja andmebaasi kiire töö, mille tagavad vastavad nõuanded andmete salvestus ning äriteabe saamise tehnoloogiate õpetustes [37], [39].

Optimeerimise otstarbeks valitud metoodikat ehk domeenide *whitelist'*i koostamist kasutatakse lähtudes selle olulisuse esile tõstmisest vabavaralise näide soovituste põhjal [36]. Lahenduse programmeerimisel viiakse kokku DSP serveri, MySQL tehnoloogilised teadmised, andmete iseloom ning reklaamioksjonil osalemise kriteeriumid.

Arenduse jooksul loodud lahenduste kooskõla reklaame pakkuva serveri Google Authorized Buyers kriteeriumitega minimaalse latentsuse osas testis autor kasutades serveri koodi kiirust testivate tööriistadega nagu Apache Benchmark ning Xdebug profile snapshot. Andmebaasi tabelite ning päringute disainimisel testis autor andmebaasi töö kiirust kasutades MySQL päringukeeles olemasolevat *explain* käsku, et terve lahendus võimaliku minimaalse ajaga toimiks.

6.2 Päringute sõelumine

Bytelogics DSP saab reklaamipäringuid Google Authorized Buyers platvormilt, mis kasutab klientidega suhtlemiseks lisaks enda standardile ka OpenRTB standardit. Seega oli peamine abivahend reklaamipäringu sisu ja reeglite aru saamise puhul OpenRTB 2.5 dokumendi peatükk 3.2 [30].

Dokumendi kirjelduse kohaselt olenevad päringu JSON kehas olevad andmed paljuski sellest, milliselt domeenilt reklaamipäring alguse saab. Alati ei pruugi kõik väljad olla täidetud ja see on veebilehe enda otsustada. Samuti oleneb andmete iseloom sellest, kas reklaamipäring tuleb mobiilirakendusest või veebibrauserist [30].

Alapeatüki 3.2.13 abil sai selgeks, et *whitelist* koostamise puhul oleks oluline salvestada iga veebibrauserist sissetuleva päringu puhul objekt *site*. Mobiilirakenduse puhul see näiteks puudub ja selle asemel on objekt *app*, mille väärtus on rakenduse nimi. *Site* on massiiv, mille sees on erinevad veebilehte kirjeldavad atribuudid. Oluliseks nende hulgast salvestada nägi autor *domain* ja *page* objekte. Teine väärtus erineb esimesest selle poolest, et seal on juures ka URL'i (*Uniform Resource Locator*) järel olev täpne teekond (*path*).

Alapeatükk 3.2.20 aitas tuua välja, millist informatsiooni saab reklaamipäringust kasutaja kohta. Selleks on päringu kehas massiiv *user.* Praktikas on selle sees peale *id* midagi väga harva, sest erinevad piirangud ning seadused keelavad isikut identifitseerivaid andmeid edastada. OpenRTB dokument viitas ka sellele, et *id* väli on sõltuv reklaamioksjoni serverist, seega oli abiks ka Google Authorized Buyersi dokumentatsioon, mis kirjeldab, et päringuga kaasa antav *id* on *google id,* mille omistab Google igale internetikasutajale küpsiste abil lähtuvalt veebibrauserist. Ometi pole ka see väli alati kaasas, kuna kasutajal on õigus küpsiseid kustutada [31].

Antud töö fookuses autor midagi rohkemat reklaamipäringust salvestamiseks vajalikuks ei pidanud. Kuna eelnevalt oli pidanud *Bidder* klass kontrollima päringu sisu olemasolu ning päringu *id* olemasolu, siis oli autoril tarvis selleks olemasolevat meetodit täiendada lisades koodi, mis eraldab reklaamipäringu JSON objektist parameetrid *site* ja *user* (Joonis 7). *Site* on olemas iga brauserist tuleva päringu puhul, seega ei ole seda tarvis tingimuslikult salvestada vaid sümbol „@" märgib PHP koodis, et väärtus on alati olemas. Küll aga massiivi sees olevad *domain* ja *page* vajavad kontrolli, mida saab PHP's

teha muutuja salvetamisel kasutades kolmepoolse operaatori süntaksit, milles *if* ja *else* asendatakse kooloni ning küsimärgiga [40]. Kontrolli vajab ka *user* objekti *id* ning kuna väärtused on andmebaasis *nullable*, siis nende puudumisel päringus omistatakse vastavatele muutujatele koodís *null*.

```
private function parseJson($this->input) {
        if (empty($this->input)) {
            return(false);
        }
        $this->bidRequest = json_decode($this->input, true);
        if (empty($this->bidRequest)) {
            return(false);
        }
        $this->bidRequestId = @$this->bidRequest['id'];
        if ( ! $this->bidRequestId ) {
            $this->error("parseRequest: no id in request", 0.1);
            return(false);
        }
        $site = @$this->bidRequest['site'];
        $domain = isset($site['domain']) ?
$site['domain'] : null;
        $this->page = isset($site['page']) ?
             $site['domain'] : null;
        if ($this->page != null) {
            $this->url = parse_url($this->page);
            $this->domain = $this->url['host'];
            $this->path = isset($this->url['path']) ?
            substr($this->url['path'],1,255) : '';
            $this->bidRequest['path'] = $this->path;
        } else {
            $this->domain = strtolower($domain);
        }
        $user = @$this->bidRequest['user'];
        $this->google_id = isset($user['id']) ? $user['id'] : null;
        return(true);
    }
```

Joonis 7. Reklaamipäringu JSON keha sõeluv funktsioon

6.3 Andmebaasi ettevalmistus

6.3.1 Tabelite loomine

Andmebaasi tabelite loomise puhul oli peamine lähtekoht luua võimalikult kiiresti päritavad tabelid ning võimalikult efektiivselt mälu kasutada. Põhjuseks suur sissetulevate päringute ja seeläbi ka salvestamiseks vajalike andmete hulk. Selleks

vajalike praktikate kohta sai teadmisi O'Reilly MySQL jõudluse optimeerimise õpetustest [37]. Õpetustest sai töö autor abi ka päringute ja tabelite disaini efektiivsuse ja kiiruse testimise praktika osas, milleks on EXPLAIN käsk. See kirjeldab iga päringu ridade lugemise arvu, indeksite kasutamist jne. Päringute programmeerimine ja andmetabelite loomine käisid algselt käsikäes, sest tarvis oli testida nende kiirust.

Andmetabelite disain loodi vastavalt päringust sõelutud vajaminevatele andmetele, mida illustreerib ka autori loodud ERD (*Entity-Relationship Model*) skeem (Lisa 3), mis sai loodud enne andmebaasi programmeerimist kuid täiendati paralleelselt programmeerimisega. Skeemil on rohelisega märgitud autori poolt loodud tabelid ning punasega varem loodud tabel, mida samuti autori tööks tarvis oli. Sinna tabelisse on varem kogutud andmed, mis tulevad kasutajate Bytelogics kodulehele saabumise korral *Google Cookie Matching* teenuse abil. Tabelis on kokku viidud Bytelogics kasutaja *id* ja Google poolt kasutajale antud *google id.* Sama *google id* on üldiselt kaasas ka reklaamipäringutes ning see võimaldab luua ülevaate reklaamidest, läbi mille tulevad Bytelogics kasutajad ning eraldada nende seast internetikasutajad, kes veel ei ole reklaamile klikkinud ega Bytelogics kasutajaks saanud.

Lisa 4 on välja toodud tabelite loomise MySQL kood. Oluline oli testida ning luua õige andmebaas enne lokaalselt, produktsioonis olevate tabelite mahud kasvavad kiirelt ning sinna indeksite lisamine võib suurte tabelite puhul liiga kaua aega võtta. Tabelite loomisel pöörati seega tähelepanu indeksite loomisele – *Domain, User, Path, Site* tabelite puhul sai loodud komposiitindeksid, mis kiirendavad vajaminevaid päringuid märkimisväärselt. Lisaks luues komposiitindeksi tabeli väärtuse ja primaarvõtmega, saab MySQL andmebaasimootor InnoDB aru, et tegu on unikaalse väärtusega ja kontrollib seeläbi duplikaatide sisestamist automaatselt.

6.3.2 Andmete salvestamine

Kuna kriteeriumiks oli andmete kiire lugemine ning kirjutamine, siis oli vajalik tabelite disainimisega paralleelselt luua andmeid kirjutavad käsud. Selleks kirjutas autor MySQL programmeerimiskeeles protseduuri (Lisa 5.), mis salvestab sissetulevate reklaamipäringute andmed tabelitesse. Tavaliselt on suuremates ettevõtetes serveri koodi programmeerijad eraldi andmebaasi arendajatest, seega peamiselt annab salvestusprotseduuride kirjutamine andmebaasi, mitte serverisse, andmete salvestamise kontrolli andmebaasi arendajale.

Salvestusprotseduur on kiirem kui eraldi MySQL käsud, sest andmebaas kutsutakse välja mitme korra asemel ühe korra. Veel muudab see tööd kiiremaks läbi selle, et andmebaas salvestab korduvkasutatava protseduuri vahemällu [37].

Protseduur kasutab indekseeritud tabelite väljasid, et kontrollida ega sissetulevaid andmeid juba ei eksisteeri. Transaktsioonid erinevate andmebaasi päringute ümber võimaldavad lukustada antud tabelivälja. Näiteks kui sama *id*'ga tabelivälja tahab muuta samal ajal mitu transaktsiooni, siis see ei ole võimalik. Kuna *bid* tabel, mis salvestab andmed sissetulevad päringu koha on alati uus, siis selle ümber ei ole transaktsiooni vaja ega duplikaadi kontrolli vaja.

MySQL protseduuri teostab PHP serveri MVC raamistiku mudel, mille kutsub välja kontroller. Raamistiku poolt on mudeli koodis SQL päringute väljakutsumiseks meetod, mis ootab sisendiks puhast MySQL koodi tekstina. Selle kasutamiseks oli tarvis autoril integreerida JSON'i sõelumis meetodi ja päringu loenduri vahele lisa funktsionaalsus, mis mudelit kasutab ja päringust sõelutud *Bidder* klassi muutujad andmebaasi salvestab (Joonis 8). Muutujad, mille ees on *this* märksõna, on *singleton* klassi muutujad ehk seeläbi ka globaalsed muutujad. Kuna *Model* on *Bidder* klassis initsialiseeritud, siis on kõikjal klassi koodis selle meetoditele ligipääs. Alljärgnev funktsioon kutsub välja *execSQL* meetodi, mille sisendiks läheb andmebaasi kirjutatud MySQL protseduur koos kõikide muutujate massiiviga, mis päringu salvestamiseks on tarvis.

```
private function bidToDb() {
    $bidRequestId = substr($this->bidRequestId, 0, 22);
    // save 5% of raw json
    if (rand(0,100) < 5) {
        $rawJson = $this->input;
    } else {
        $rawJson = '';
    }
    $row = $this->Model->executeSQL("call pBid(?,?,?,?,?)",
        array('sssss',
            $this->str($this->google_id),
            $this->str($this->domain),
            $this->str($this->path),
            $this->str($rawJson),
            $this->str($bidRequestId)
            ));
    if(isset($row["bid_id"])) {
        $this->bidDbId=$row["bid_id"];
    } else {
        $this->error("Bid request not stored: ".$bidRequestId);
    }
    return(true);
}
```

Joonis 8. Andmebaasi protseduuri väljakutsuv funktsioon

Mudeli funktsiooni sisendiks mineva massiivi muutujate ümber on abifunktsioon `$this->str()`, mis kõikide tekstide ülakomad MySQL päringu teostamiseks kõlblikuks teeb. Funktsiooni alguses on seatud piirang, mis ei salvestaks kõikide sisse tulevate päringute JSON'it. See väli salvestatakse andmebaasi testimise eesmärgil, et omada ülevaadet sissetulevate päringute olemuse kohta. Vajadusel saab seda koodis muuta ja lõplikus DSP versioonis võiks olla 0%. Funktsioon tagastab alati tõese väärtuse, mis tähendab, et kui midagi valesti läheb, siis päringu informatsiooni ei salvestada, kuid selektsiooni algoritm saab endiselt oma tööd teha.

6.4 Analüütika ja äriteave

Eeltoodud integratsioonid võimaldasid hakata koguma andmebaasi reklaamipäringute andmeid. Andmete kogumine ning analüüs on kestev protsess. Mida suurem kogus andmeid, seda paremad võimalused on täpsema ning sügavama äriteabe ammutamiseks andmetest. Käesoleva töö ajalise limiidi tõttu seati andmete kogumise perioodiks kaks nädalat. Tunnis saatis Google Authorized Buyers reklaamipäringuid ligikaudu 100000,

mis tähendab, et kahe nädala jooksul kogutud ligikaudu 33 miljonit päringut on piisav minimaalne kogus analüüsi ja *whitelist'*i koostamiseks.

Andmete kogumise protsessi käigus tegi autor kogutud andmetabelid nädala kaupa osadeks. Esimese nädala kogutud *bid* sisendid olid ühes tabelis, mille nimi muudeti *bid week_1* ja teise omad *bid_week_2.* Põhjus seisnes äriteabe MySQL koodi välja mõtlemises reaalsete andmete peal ning ka selles, et produktsioonis olev andmebaas liialt koormust ei saaks. Selle jaoks kopeeris autor käsurea abil esimese nädala andmetabelid teise masinasse, kus need lokaalsesse andmebaasi lisas, et selle peal arendada. Kopeerimiseks oli vajalik sisestada terminali allolev käsk.

```
mysqldump -u User -pPassword bidder
--column-statistics=0
--single-transaction
--result-file="/Users/karlkallas/bidder/mysqlDumps/bid_week_1.sql" bid
--skip-create-options
```

Joonis 9. MySQL tabeli kopeerimise käsk

Lõplike andmete peal päringute jooksutamiseks oli tarvi tabelid tagasi kokku liita, mille jaoks kasutas autor MySQL andmebaasi tabelite partitsioone, mis võimaldab mitu tabelit omavahel kokku panna. Protseduuri illustreeriv näide *bid* partitsioonitabeli loomisest on toodud all. Partitsioonide suurused on vastavalt lisatava tabeli maksimaalse *id* ümardamisel saadud.

```
create or replace table bid_partition;

alter table bid_partition partition by range (id) (
    partition p1 values less than (16000000),
    partition p2 values less than (33000000),
    partition p values less than (maxvalue)
    );

alter table bid_partition exchange partition p1 with table bid_week_1;
```

Joonis 10. Partitsioonitabelite loomise kood

Lõpliku MySQL *whitelist* päringu koostamisel oli veel tarvis filtreerida välja kõik unikaalsed Bytelogics lehel käinud *google_id*'ga kasutajad *matches* tabelist (Joonis 11). Põhjusel, et tabelisse oli sattunud erineva *bytelogics_id* kuid sama *google_id*'ga

kasutajad, mis tuleneb sellest kui samas veebibrauseris mitu kasutajat käib, kuna *google_id* on brauseri küpsise põhine. Sellist olukorda võivad tekitada näiteks ühiskasutatavad arvutid raamatukogudes.

```
-- Matches unique
create or replace
    view v_matches_unique as
    select distinct matches.google_id AS google_id
from matches;
```

Joonis 11. Unikaalsete kasutajate filtreerimise kood

Lõplike partitsioonide ja unikaalsete kasutajate peal loodud MySQL kood, mille abil domeenidest *whitelist* loodi on loodud kasutades *join* käsku, mis ühendab vajaminevad tabelid primaarvõtme ning võõra võtme vahel (Joonis 12). *User* tabel on ühendatud ühise välja *google_id* abil. Kuna eesmärgiks on leida domeenid, mis kõige paremini Bytelogics DSP automatiseeritud turunduses toimivad, tuleb leida domeenid, millel näidatud reklaamide kaudu on kasutajad jõudnud Bytelogics lehele ehk millistel veebilehtedel klikitakse reklaamidele. Selleks on tehtud andmebaasi päring, mis grupeerib kõik kahe nädala jooksul kogutud domeenid. *Join* käskude abil saab päringusse tabeleid liites filtreerida välja *v_matches_unique* kasutades domeenid, millel näidatud reklaamid on lõppenud klikiga. *Having* käsk päringu lõpus on klikiroboteid ennetav kontrollimise mehhanism, mis eeldab, et domeeni on külastanud vähemalt 10 unikaalset kasutajat. Antud töö fookuses on see piisavalt efektiivne, et saada kätte töötav *whitelist.*

```
-- Whitelist domains
select d.domain as domain_name, count(distinct b.user_id) as
user_counts
from bid_partition b
    join site_partition s on b.site_id = s.id
    join user_partition u on b.user_id = u.id
    join v_matches_unique m on u.google_id = m.google_id
    join domain_partition d on s.domain_id = d.id
where d.domain != ''
group by d.id
having user_counts > 10
order by count(distinct b.user id) desc;
```

Joonis 12. Domeenide *whitelist* koostamise MySQL kood

6.5 Analüüsi tulemuste rakendamine

MySQL on oluline töövahend andmetest äriteabe ammutamiseks, pealtnäha lihtne päring võib anda palju sisulist ja kasulikku informatsiooni automatiseeritud turunduse tulemuste kohta. Autori kirjutatud päring andis tulemuseks tabeli domeenide nimedest, mis on reastatud lehekülje külastatavuse arvu järgi (Joonis 13). Seega on tootlikumad domeenid, mis reklaame näitavad tabeli ülal osas.

domain_name	user_counts 1
www.foxnews.com	669
nypost.com	433
www.cnn.com	206
people.com	134
www.nbcnews.com	131
www.cheatsheet.com	100
www.usatoday.com	93
www.foxbusiness.com	88
www.newsweek.com	82
www.cnbc.com	77
heavy.com	76
pagesix.com	72
www.forbes.com	71
thehill.com	69
www.nytimes.com	68

Joonis 13. MySQL analüüsi tulemuste tabel ehk *whitelist*

Kuna eelnevalt panustas Bytelogics DSP kõikidele ettejuhtuvatelt domeenidelt tulevatele reklaamipäringutele, siis antud analüüsi tulemus andis võimaluse optimeerida DSP tööd tehes panustamine efektiivmsemaks, kontrollides millistele domeenidele reklaamipakkumised tulevad. Google Authorized Buyers API kaudu sai autor programmaatiliselt muuta *pretargeting* ehk eelsihtimise konfiguratsioone, milles on võimalik määrata millistelt domeenidelt reklaamipakkumised tulevad. Selleks tuli saata PUT päring, mille sisendiks läks JSON keha koos massiiviga *placements,* mille sees olid kõik domeenid, mis *whitelist'*i leidmisel saadi (Joonis 14).

```
{
  "placements": [
    {
      "token": "www.foxnews.com",
      "type": "PRETARGETING_PLACEMENT_SITE"
    },
    {
      "token": "nypost.com",
      "type": "PRETARGETING_PLACEMENT_SITE"
    }
      ...
   ]
}
```

Joonis 14. *Whitelist* domeenide uuendamise päringu JSON

Peale konfiguratsiooni päringu saatmist muudeti Google Authorized Buyers serveris reklaamipäringute saatmine ainult hea tootlikusega ehk autori poolt lisatud *whitelist* domeenidele.

6.6 Järeldused

Käesolevas töös tehtud Bytelogics DSP andmehalduse täiendused, andmekogumis kanali loomine PHP serveris ja MySQL andmebaasis koos andmete analüüsiga aitavad parandada ettevõtte automatiseeritud turunduse tootlikust. Täienduste ja andmete töötlemise tulemuse rakendamisel panustab Bytelogics DSP hästi tootvatele domeenidele ja seeläbi ei kuluta turunduseelarvet enam juhuslikult vaid järgides kindlat strateegiat. See võimaladab optimeerida reklaamimise hinda, makstes reklaamide näitamise eest sama summa kuid saades vastu rohkem klikke.

Esimene nädal *whitelist* domeenidele panustades tõusis reklaami nädalane klikkide arv koguni kahe kordselt. Kuna analüüsi tulemusel leiti domeenid, millel näidatud reklaamidelt on tulnud olemasolevad Bytelogics kasutajad, oli võimalik fokusseerida kogu automatiseeritud turundus nendele domeenidele, seeläbi näidata reklaami ka suurema tõenäosusega täpsemale kasutaja sihtgrupile, kes veel ei ole Bytelogics kasutaja.

Bidder server salvestas eelnevalt autori loodud täiendustele reklaamide klikke, mille abil reklaamioksjonile hinda arvutati. Tabelist pärides õnnestus testida täienduste eesmärgi saavutamist (Joonis 15), milleks oli optimeerida Bytelogics DSP automatiseeritud turundust ja tõsta reklaamide tootlust. Tootluse efektiivsuse tõusu tõestas võrdlus sisse

tulevate klikkide arvuga enne ja peale täiendusi, kuna päringute arv nädalas jäi samaks kuigi klikkide arv reklaamidele tõusis vastavalt – nädal millal panustati veel kõikidele domeenidele, saadi 102 klikki, nädal millal panustati *whitelist* domeenidele, saadi 238 klikki.

```
select count(*) from clicks
where date between '2020-11-02' and '2020-11-06';

select count(*) from clicks
where date between '2020-11-09' and '2020-11-13';
```

Joonis 15. Klikkide tabelite päringud enne ja peale täienduse

Kui arvestada, et reklaamipäringuid tuli päevas ligi 100000, siis kahekordne tootluse parandamine tähendas, et varasemalt saadi viie päeva ehk 50000 päringu kohta ligikaudu 5000 (500000 / 102) reklaami näitamise kohta 1 klikk. Optimeerides Bytelogics DSP tööd *whitelist* domeenidele parandati tulemus ligikaudu 2000 reklaami näitamise kohta 1 klikk.

Loodud täiendused kinnitasid, et DSP serveri täiendamine DMP ehk andmehalduse funktsionaalsusega tõstab töö efektiivsust ning seeläbi optimeerib turunduseelarve kasutust.

6.7 Võimalikud edasiarendused

Andmehaldus ja -töötlus on väga laiahaardeline valdkond, mille osad küündivad andmekogumis tehnoloogiate arendusest andmete analüüsini. Samuti on automatiseeritud turundus, DSP serverid ja reklaamide reaalajaoksjonid väga sisutihedad valdkonnad. Käesoleva töö kontekstis tehtud täiendused iseloomustavad läbi reklaamide tootlikuse tõstmise andmete kogumise ning nendest informatsiooni leidmise olulist rolli automatiseeritud turunduses ja DSP rakenduses.

Igasugune optmieerimise protsess on oma olemuselt erinevate meetodite katsetamine ning nende tulemuste võrdlemine. See on aega ja kannatust nõudev protsess, mis ei pruugi alati tulemust parandada, kuid on sellegi poolest oluline.

Bytelogics DSP automatiseeritud turunduse meetodite optimeerimist saab laiendata kasutades sügavamaid statistilisi mudeleid kogutud andmete töötlemiseks. Kõik töös

tehtud täiendused on loonud head eeldused mõelda edasi näiteks masinõppe mudelite kasutamisele andmeanalüüsis. Loodud andmekogumise kanal, kogutud andmete hulk ning *whitelist* koostamine on edasisteks arendusteks hea lähtekoht. Näiteks saab *whitelist* domeenid omakorda jagada gruppideks, kasutades selleks mõnda masinõppe grupeerimis meetodit, mis teeb reklaami pakkuvatele domeenidele panustamise veelgi täpsemaks.

7 Kokkuvõte

Automatiseeritud turundus, mis moodustab suure osa tänapäeval laialt kasutusele võetud digitaalturundusest, võimaldab veebidomeenidel olevatele reklaamipindadele teha pakkumisi kasutaja veebilehe avamisel reaalajas. Kogudes reklaamipäringutest andmeid reklaame näitavate veebidomeenide ja kasutaja kohta luuakse turundus sihtgruppe, mis reklaamimise internetis võimalikult efektiivseks teevad.

Käesoleva bakalaureusetöö eesmärk oli välja töötada lahendused, mis parandaksid ettevõtte Bytelogics DSP reklaamiserveri automatiseeritud turunduse efektiivsust. Eesmärgiks oli tõsta veebidomeenidel näidatavatele reklaamidele tulevate klikkide osakaalu turunduseelarvet suurendamata.

Käesoleva töö raames käsitleti automatiseeritud turunduse efektiivsuse tõstmiseks valminud andmete kogumise ja haldamise süsteemi, mis integreeriti reklaamioskjonitel osalevasse serverisse PHP programmeerimiskeele ning MySQL päringukeele abil. Selleks valmistati ette andmebaas, mis olulised andmed salvestas ning võimaldaks viia neid kokku varem kogutud andmetega. Täiendati ka olemasoleva serveri funktsionaalsust päringute sõelumiseks ning andmebaasiga suhtluseks.

Lahendus võimaldas salvestada reklaamioksjonite päringutest sõelutud andmeid pidades kinni Google Authorized Buyers reklaame vahendava teenuse ning OpenRTB standardi suunitlustest. Peamine kriteerium reklaame vahendava serveriga suhtluseks oli reklaamipäringutele vastamine ligikaudu 100 millisekundiga. Selleks testiti serverisse loodud lahendusi Apache Benchmark ning Xdebug profile snaphsot tehnoloogiad kasutades. Samuti testiti andmebaasi töökiirust ning lähtuti andmebaasi disaini puhul mälu optimaalsest kasutamisest.

DSP serverisse integreeritud andmete kogumise lahenduse tulemusel loodi MySQL päringukeelt kasutades ka andmeanalüüsi protseduur, tänu millele saadi äriteavet domeenide kohta, millel näidatud reklaamide kaudu on Bytelogics kasutajad peamiselt tulnud. Selle tulemusel tehti domeenide valge list ehk *whitelist*, milles olevad veebilehed

iseloomustavad Bytelogics kasutajaid kõige rohkem ning mille kaudu ka tulevasi potentsiaalsed kasutajaid oodatakse. *Whitelist* sisendati automatiseeritud süsteemi, et ainult nendele domeenidele reklaame näidata ja seeläbi turunduse efektiivsust tõsta.

Lõplik lahendus, mis panustas reklaamidele ainult *whitelist* domeenidel saavutas töö jaoks seatud eesmärgi, tõstes reklaamide klikkide arvu kahekordselt ning muutes seeläbi automatiseeritud turunduse efektiivsemaks. Sealjuures jäi turundusse panustatud eelarve suurus samaks ja seetõttu on täiendused ettevõttele kasulikud.

Kasutatud kirjandus

[1] How Much Data is Created on the Internet Each Day? Microfocus. [WWW] https://blog.microfocus.com/how-much-data-is-created-on-the-internet-each-day/ (01.11.2020)

[2] What is programmatic advertising? Match2One. [WWW] https://www.match2one.com/blog/what-is-programmatic-advertising (15.10.2020)

[3] NetGravity Launches AdServer, the Premier Advertising Management System Software for Worl Wide Web Publishers. NetGravity. [WWW] https://web.archive.org/web/19980526003204/http://www.netgravity.com/news/pressrel/launch.html (17.10.2020)

[4] How Ad Server Works. Epom. [WWW] https://epom.com/blog/ad-server/how-ad-server-works (15.10.2020)

[5] Ad Tech 101. Epom. [WWW] https://epom.com/blog/ad-server/ad-tech-101 (15.10.2020)

[6] 2020 Global Digital Ad Trends. PubMatic. [WWW] https://pubmatic.com/wp-content/uploads/2019/12/2020-Global-Digital-Ad-Trends.pdf (01.11.2020)

[7] Customer story – Flygbussarna transport service Sweden. Match2One. [WWW] https://www.match2one.com/customer-stories/flygbussarna-transport-service-sweden/ (20.10.2020)

[8] Pedro Palos-Sanchez , Jose Ramon Saura , Felix Martin-Velicia. A study of the effects of programmatic advertising on users' concerns about privacy overtime. (2019) [WWW] https://www.sciencedirect.com/science/article/abs/pii/S014829631830540X (21.10.2020)

[9] Media Buying Models. Singular. [WWW] https://www.singular.net/blog/media-buying-models/ (26.10.2020)

[10] Authorized Buyers Overview. Google support. [WWW] https://support.google.com/authorizedbuyers/answer/6138000?hl=en (01.11.2020)

[11] Authorized Buyers Program Guidelines. Google. (2020) [WWW] https://www.google.com/doubleclick/adxbuyer/guidelines/ (02.11.2020)

[12] Jun Wang, Weinan Zhang, Shuai Yuan. Display Advertising with Real-Time Bidding (RTB) and Behavioural Targeting. (2017) [WWW] https://arxiv.org/pdf/1610.03013.pdf (01.11.2020)

[13] About one-in-five U.S. adults say they get their political news primarily throuch social media. Journalism. [WWW] https://www.journalism.org/2020/07/30/americans-who-mainly-get-their-news-on-social-media-are-less-engaged-less-knowledgeable/pj_2020-07-30_social-media-news_00-01/ (14.11.2020)

[14] Postimehe tiraaž langes esmakordselt alla 40 000. Eesti Rahvus Ringhääling. (04.05.2019) [WWW] https://www.err.ee/936276/postimehe-tiraaz-langes-esmakordselt-alla-40-000 (14.11.2020)

[15] Ignorance Around Advertising Is Putting the Free Internet at Risk. Ad Tech Daily. (23.10.2020) [WWW] https://adtechdaily.com/2020/10/23/ignorance-around-advertising-is-putting-the-free-internet-at-risk/ (17.11.2020)

[16] IAB & IAB Tech Lab Respond With Support for OpenRTB and IAB Europe's Transparency & Consent Framework. IAB Tech Lab. (19.10.2020) [WWW] https://iabtechlab.com/iab-and-tech-lab-respond-with-support-for-open-rtb-and-iab-europes-transparency-consent-framework/ (10.11.2020)

[17] Alexander Bleier. The Imporance of Trust for Personalized Online Advertising. (2015) [WWW] https://www.sciencedirect.com/science/article/pii/S0022435915000263 (11.11.2020)

[18] Introduction to Filter Lists. AdBlock. [WWW] https://help.getadblock.com/support/solutions/articles/6000066909-introduction-to- filter-lists#:~:text=To%20access%20filter%20lists%20in,left%20of%20the%20options%20p age.

[19] OpenRTB Advisory. IAB Tech. (08.02.2018) [WWW] https://iabtechlab.com/wp-content/uploads/2018/02/OpenRTB_Advisory_GDPR_2018-02.pdf (8.11.2020)

[20] AdCOM Specification v1.0. IAB Tech Lab. (2020) [WWW] https://github.com/InteractiveAdvertisingBureau/AdCOM/blob/master/AdCOM%20v1.0%20FINAL.md#object_regs (05.11.2020)

[21] Ad Manager and Ad Exchange program policies. Google Support. [WWW] https://support.google.com/admanager/answer/9835267?hl=en&ref_topic=28145 (10.11.2020)

[22] Data Privacy. Unity Ads. [WWW] https://unityads.unity3d.com/help/programmatic/data-privacy (10.11.2020)

[23] When Wading into the Bidstream Beware the Currents. Martech Advisor. (03.10.2019) [WWW] https://www.martechadvisor.com/articles/ads/when-wading-into-the-bidstream-beware-the-currents/ (11.11.2020)

[24] Matt Savare. Fear of Brave? An Analysis of GDPR Challenges to Behavioral Advertising. (13.11.2018) [WWW] https://www.lowenstein.com/news-insights/publications/client-alerts/fear-of-brave-an-analysis-of-gdpr-challenges-to-behavioral-advertising (09.11.2020)

[25] Principles Relating to Processing of Personal Data. Intersoft Consulting. https://gdpr-info.eu/art-5-gdpr/ (05.11.2020)

[26] 1st, 2nd, 3rd Party Data – What It All Means? AdSquare. [WWW] https://www.adsquare.com/1st-2nd-and-3rd-party-data-what-it-all-means/ (10.11.2020)

[27] Contextual Marketing: The Real Business of Internet. Harvard Business School. [WWW] https://hbswk.hbs.edu/archive/contextual-marketing-the-real-business-of-the-internet (07.11.2020)

[28] The Seven-Step Ad Tech Guide. Data & Marketing Association. (2020) [WWW] https://dma.org.uk/uploads/misc/seven-step-ad-tech-guide-v9.pdf (12.11.2020)

[29] OpenRTB (Real-Time Bidding) [WWW] https://iabtechlab.com/standards/openrtb/ (04.11.2020)

[30] OpenRTB API Specification Version 2.5. IAB Tech. (2016) [WWW] https://iabtechlab.com/wp-content/uploads/2016/07/OpenRTB-API-Specification-Version-2-5-FINAL.pdf (04.11.2020)

[31] Reaalajaoksjoni suunitlused. Google Developers. [WWW] https://developers.google.com/authorized-buyers/rtb/start (14.10.2020)

[32] Protocol buffers. Google Developers. [WWW] https://developers.google.com/protocol-buffers (01.11.2020)

[33] Infrastructure Options for Serving Advertising Workloads. Google. [WWW] https://cloud.google.com/solutions/infrastructure-options-for-serving-advertising-workloads (01.11.2020)

[34] How to Monitor Server Performance. Heroix. [WWW] https://blog.heroix.com/blog/how-to-monitor-server-performance (14.11.2020)

[35] Infrastructure Options for Data Pipelines In Advertising. Google. [WWW] https://cloud.google.com/solutions/infrastructure-options-for-data-pipelines-in-advertising (02.11.2020)

[36] RTB4Free Overvire. RTB4Free. [WWW] http://rtb4free.com/details_docker.html (10.10.2020)

[37] Jeremy D. Zawodny, Derek J. Balling, Baron Schwartz, Peter Zaitsev, Arjen Lentz, Vadim Tkachenko. High Performance MySQL, 2nd Edition. (2008) [WWW] https://www.oreilly.com/library/view/high-performance-mysql/9780596101718/ch04.html (05.10.2020)

[38] DIY DSP. RTB4Free. [WWW] http://www.rtb4free.com/dsp_blog1a.html (10.10.2020)

[39] Turunduse automatiseerimine ja seda abistav tarkvara. Zaproo. [WWW] https://www.zaproo.ee/artiklid-ja-arvustused/turunduse-automatiseerimine-ja-seda-abistav-tarkvara/ (01.12.2020)

[40] Shorthand comparisons in PHP. Stitcher. [WWW] https://stitcher.io/blog/shorthand-comparisons-in-php (03.11.2020)

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Karl Kallas

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Reklaamiserveri täiendamine automatiseeritud turunduse jaoks ettevõtte Bytelogics Inc. näitel„ mille juhendaja on Toomas Lepikut
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

07.01.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – Reklaamipäringu JSON

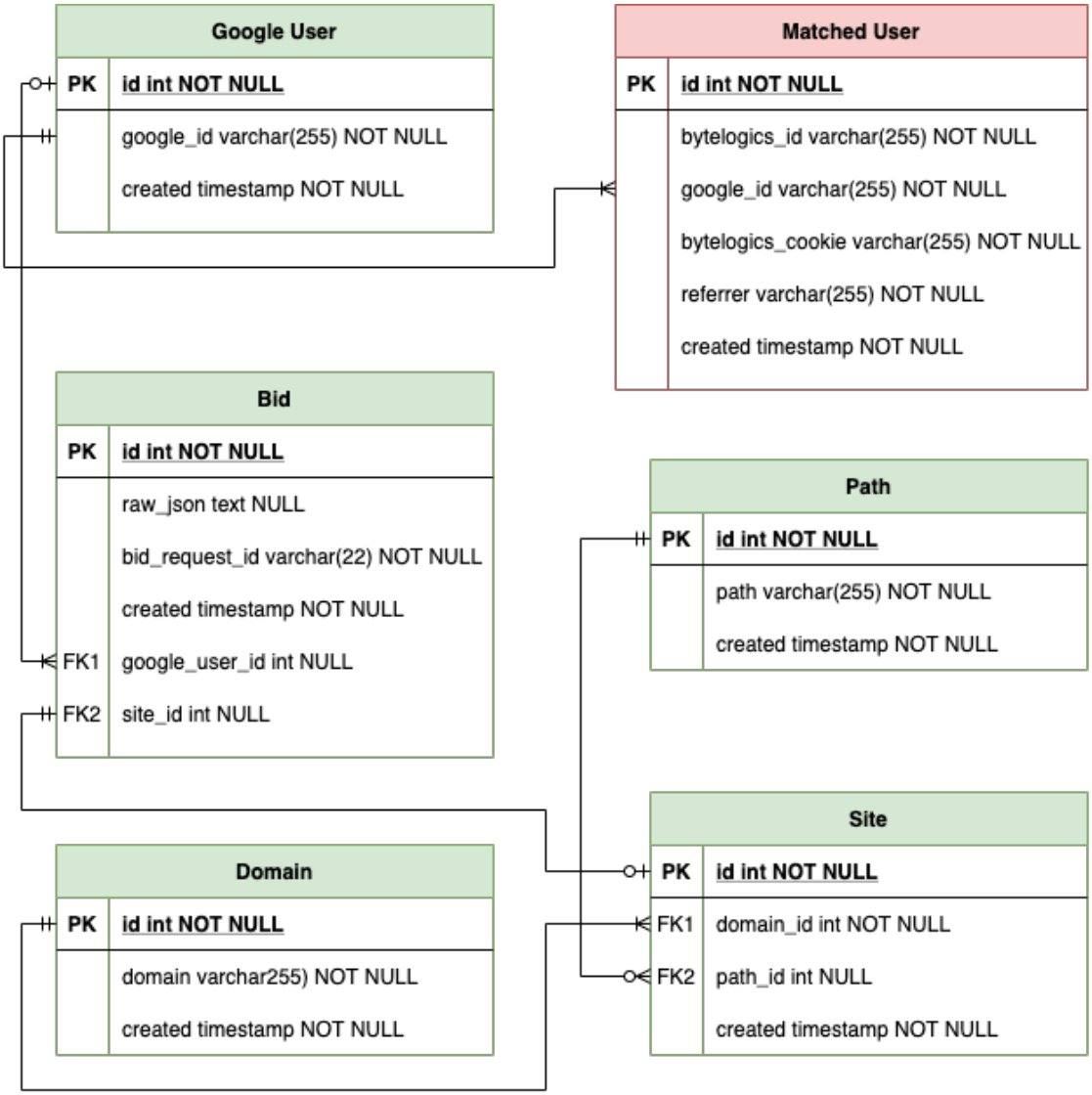

```
{
    "id": "TESTX0FJusRtn3k05A72F",
    "imp": [
        {
            "id": "1",
            "banner": {
                "w": 300,
                "h": 250,
                "pos": 1
            },
            "tagid": "2538099118",
            "bidfloor": 1.8799999999999999,
            "bidfloorcur": "USD",
            "secure": 1,
            "metric": [
                {
                    "type": "click_through_rate",
                    "value": 0.0012041396694257855,
                    "vendor": "EXCHANGE"
                }
            ],
            "ext": {
                "billing_id": [
                    "104443512217"
                ],
                "publisher_settings_list_id": [
                    "15540914462735936663",
                    "16194453266193410891"
                ],
                "allowed_vendor_type": [
                    42,
                    144
                ],
                "ampad": 3
            }
        }
    ],
    "site": {
        "page": "https://www.ebay.com/itm/Toyota-4-runner-nightshade-
2020",
```

```
        "publisher": {
            "id": "pub-5335449554482979",
            "ext": {
                "country": "US"
            }
        },
        "content": {
            "livestream": 0,
            "language": "en"
        },
        "mobile": 1,
        "ext": {
            "amp": 0
        }
    },
    "device": {
        "ua": "Mozilla/5.0 (iPhone; CPU iPhone OS 13_6)",
        "ip": "76.182.0.0",
        "geo": {
            "country": "USA",
            "region": "NC"
        },
        "make": "apple",
        "model": "iphone",
        "os": "iOS",
        "osv": "13.6"
    },
    "user": {
        "id": "TESTgoogle_id"
    },
    "at": 1,
    "tmax": 122,
    "cur": [
        "USD"
    ],
    "bcat": [
        "IAB14-4",
        "IAB10-1",
    ],
    "ext": {
        "google_query_id": "TEST2fwi2SGIMbh9sda23DsdWasfpiej1"
    }
}
```

Lisa 3 – Reklaamipäringu vastuse JSON


```
{
    "id": "TESTX0FJusRtn3k05A72F",
    "seatbid": [
        {
            "seat": "1",
            "bid": [
                {
                    "id": "1-1606494996.019018-2018",
                    "adid": "1",
                    "price": 0.20000000000000001,
                    "nurl":
      "https://biddergoogle.bytelogics.io/win?bidId=1-
      1606494996.019018-2018&cost=${AUCTION_PRICE}
                    "adm": "<img width=\"1\" height=\"1\"
src=\"https://biddergoogle.bytelogics.io/view?bidId=1-
1606494996.019018-2018&cost=${AUCTION_PRICE}\" style=\"display:none;\"
/>
<a href=\"https://biddergoogle.bytelogics.io/click?bidId=1-
1606494996.019018-2018\">
<imgsrc=\"https://biddergoogle.bytelogics.io/images/green.gif\"
width=300 height=250/></a>",
                    "adomain": [
                        "bytelogics.com"
                    ],
                    "cat": [
                        "IAB3-3"
                    ],
                    "impid": "1",
                    "iurl":
"https://biddergoogle.bytelogics.io/images/green.gif",
                    "burl":
"https://biddergoogle.bytelogics.io/win?bidId=1-1606494996.019018-
2018&cost=${AUCTION_PRICE}
                    "cid": "104443512217",
                    "crid": "1",
                    "w": 300,
                    "h": 250,
                    "ext": {
                        "bidder_name": "Bytelogics"
                    }
            }...
```

Lisa 4 – Andmebaasi tabelite täiendus ERD



Lisa 5 – Andmebaasi tabelid MySQL

```
-- Bid request table
create table if not exists bid
(
    id int auto_increment primary key,
    raw_json text null,
    user_id int null,
    site_id int null,
    created timestamp default CURRENT_TIMESTAMP not null,
    bid_request_id varchar(22) not null
);

create index bid_bid_request_id_index
    on bid (bid_request_id);

create index bid_site_id_index
    on bid (site_id);

create index bid_user_id_index
    on bid (user_id);

-- Google User table
create or replace table user
(
    id int auto_increment primary key,
    google_id varchar(255) not null
);

create or replace index user_google_id_id_index
    on user (google_id, id);

-- Domain table
create table if not exists domain
(
    id int auto_increment primary key,
    domain varchar(255) not null
);

create index domain_domain_id_index
    on domain (domain, id);

-- Path table
create table if not exists path
(
    id int auto_increment primary key,
    path varchar(255) not null);
```

```
create index path_path_id_index
    on path (path, id);

-- Site table
create table if not exists site
(
    id int auto_increment primary key,
    path_id int null,
    domain_id int not null
);

create index site_domain_id_index
    on site (domain_id);

create index site_domain_id_path_id_query_id_id_index
    on site (domain_id, path_id, id);
```

Lisa 6 – Andmete salvestusprotseduur

```
create procedure pBid(IN ua1 varchar(255),
                      IN google_id1 varchar(255),
                      IN domain1 varchar(255),
                      IN path1 varchar(255),
                      IN raw_json1 text,
                      IN bid_request_id1 varchar(22));
BEGIN
    START TRANSACTION;
    do @user_id:=(select id from user where google_id=google_id1);
    if (@user_id is null) then
        insert user (google_id) values (google_id1);
        do @user_id:=last_insert_id();
    end if;
    commit;
    START TRANSACTION;
    do @domain_id := (select id from domain where domain=domain1);
    if (@domain_id is null) then
        insert domain (domain) values (domain1);
        do @domain_id:=last_insert_id();
    end if;
    commit;
    START TRANSACTION;
    do  @path_id := (select id from path where path=path1);
    if (@path_id is null) then
        insert path (path) values (path1);
        do @path_id:=last_insert_id();
    end if;
    commit;
    START TRANSACTION;
    do @site_id := (select id from site where domain_id=@domain_id and
            query_id=1 and  path_id = @path_id);
    if (@site_id is null) then
        insert site (domain_id,query_id,path_id)
        values (@domain_id,1,@path_id);
        do @site_id:=last_insert_id();
    end if;
    commit;

    insert bid (raw_json, user_id, site_id, bid_request_id)
    values (raw_json1, @user_id, @site_id, bid_request_id1);

    select  last_insert_id() bid_id;
END;
```