TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Varvara Doilova 142844

TTÜ ÕPPEKESKKONNA AINED.TTU.EE LOGIDE VISUALISEERIMINE JA ANALÜÜS

Bakalaureusetöö

Juhendaja: Ago Luberg
Magistrikraad

Tallinn 2017

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: Varvara Doilova

24.05.2017

Annotatsioon

Andmete kogumine on olnud alati oluline. Andmete kogumise alustamiseks ei pea olema mingit kindlat põhjust, vaid loodetakse, et neist saab kunagi kasu. Logimine ehk andmete kogumine on oluline, kuna nendes andmetes on palju informatsiooni programmi tööst, mida saab kasutada programmi nõrkade külgede leidmiseks ja vigade parandamiseks. Logide analüüsimisel, eriti suure andmehulga puhul, on otstarbekam kasutada diagramme ja graafikuid, kuna nendest on lihtsam aru saada.

Käesoleva töö eesmärgiks on luua võimalus TTÜ õppekeskkonna ained.ttu.ee logide visualiseerimiseks ja analüüsimiseks. Logide analüüsimisel taheti näha õppekeskkonna kasutamist nädala lõikes, populaarseid kursusi ning iga kursuse jaoks populaarseid päevi koos kellaajaga. Samuti tunti huvi kui palju unikaalseid kasutajaid on igat kursust vaadanud.

Bakalaureusetöö tulemuseks on TTÜ serveris töötav veebipõhine logide analüüsimise võimalus, mida on võimalik kasutada edaspidi TTÜ õppekeskkonna ained.ttu.ee logide analüüsimiseks. Sama rakendust kasutades saab analüüsida ka teisi logisid, näiteks TTÜ Git'i logisid. Logide analüüsimisel on valminud vaade (*dashboard*) diagrammidega, mille alusel on võimalik analüüsida õppekeskkonna kasutamist nädala lõikes ja on võimalik erineval viisil analüüsida kursusi, näiteks ühe kaupa või mitut kursust koos.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 40 leheküljel, 6 peatükki, 23 joonist, 1 tabelit.

Abstract

TTU learning system ained.ttu.ee logs visualization and analysis

Data collection has always been important. The goal of data collection is to benefit from analyzing the contents of the data. Different programs and systems use logging to store events in a timeline. Those events can give information on how the system has been used, what are the problematic usages etc. In order to understand better, visualization can be used to present events in the logs.

The aim of this work is to come up with a solution how to visualize and analyze the logs of Tallinn University of Technology's Moodle learning system ained.ttu.ee. Some concrete examples of visualization have be presented, for example the heatmap on how the system has been used over one week in average.

The thesis gives an overview of the logging in general. Then 10 different existing log analysis and visualization tools have been compared. The author chooses Elasticsearch to be used for manipulating the data. All the used components and programs have been described. The document describes the installing and configuration process of Elasticsearch.

The data from the learning system is presented in 2 separate ways. The first (test) data was provided as plain text file with tab separated fields. The second data (from the live system) is provided directly from the database. The document describes how to handle both input sources.

After setting up the visualization tool Kibana, the author describes how to create a dashboard of different types of diagrams and visualization. The document points out how to create heatmap, tag cloud, vertical bar chart. Each is describe in detail using Moodle data as example.

As the result of the thesis, Elastic Stack has been setup to visualize data from Moodle logs. The document gives an overview of the process and also has examples on how to

create some different visualization. The same system could be extended to visualize data from another log, for example Git.

The thesis is in estonian and contains 40 pages of text, 6 chapters, 23 figures, 1 tables.

Lühendite ja mõistete sõnastik

API	*Application Programming Interface*, rakendusliides
backup	Andmete varundamine
Dashboard	Visuaalide kogum
DSL	*Domain–specific language*, domeenispetsiifiline keel
exception	Erind
HTTP	*Hypertext Transfer Protocol*, hüperteksti edastusprotokoll
IP–aadress	*Internet Protocol* – aadress, interneti protokoll Internetis arvutite ja seadmete eristamiseks
JSON	*JavaScript Object Notation*, lihtsustatud andmevahetusvorming
localhost	Kohalik host on standartne hostinimi, mille aadress viitab samale arvutile.
Master–processor	Ülemprotsessor
MySQL	Relatsioonilise andmebaasi haldamise süsteem
plugin	Laiendus, pistikprogramm
RESTful	*Representational State Transfer*, tarkvaraarhitektuuri stiil
SQL	*Structured Query Language*, struktureerutud päringukeel
TCP	*Transmission Control Protocol*, edastusohje protokoll
UDP	*User Datagram Protocol*, kasutajadatagrammi protokoll
Unix Timestamp	Ajatempli formaat
URI	*Uniform Resource Identifier*, unifitseeritud ressursi identifikaator
Worker–processor	Alluvprotsessor

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Tänapäeval on peaaegu igal programmil olemas logid. Neid salvestatakse erinevatel põhjustel ja erineval viisil. Logid võivad olla salvestatud näiteks failina või andmebaasis tabeli kujul. Logide analüüsimine on oluline, kuna selle tulemuste põhjal saab ülevaate programmi tööst, esinevatest vigadest ja nõrkadest kohtadest, mis lihtsustab programmi parendamist. Kõige parem on logisid analüüsida visuaalsel kujul, kuna inimese jaoks on piltide mõistmine lihtsam kui teksti mõistmine.

TTÜ veebilehel ained.ttu.ee on kahe aasta jooksul kogunenud üle nelja miljoni logirida, kuid täpsem analüüs selle kohta, mida tudengid lehel teevad ja milliseid lehti vaatavad, on puudu. Selline ülevaade annab võimaluse teha veebilehel uuendusi, parandusi ja täiendusi, näiteks tudengi sisselogimisel ilmub ette tema kõige külastatum leht. Samuti saab sellist analüüsi kasutada õppeainete parendamiseks, näiteks õppejõud saab näha milliseid materjale külastavad tudengid kõige rohkem, mis järelikult meeldivad.

Käesoleva bakalaureusetöö eesmärgiks on luua võimalus logide visualiseerimiseks ja analüüsimiseks. Logide analüüsimisel on huvitav näha õppekeskkonna kasutamist nädala lõikes, ülevaadet populaarsete kursuste kohta ning millistel kellaaegadel käiakse ained.ttu.ee veebilehel. Samuti on huvitav näha iga kursuse kohta millistel nädalapäevadel kui palju erinevaid kasutajaid käib seda kursust vaatamas.

Teises peatükis kirjeldatakse logide olemasolu ja logimise tähtsust ning miks on vaja logisid analüüsida ja visualiseerida. Kolmandas peatükis pannakse paika nõuded, viiakse läbi tehnoloogiate analüüs, seejärel selgitatakse struktuuri ja programmide kirjeldusi. Neljandas peatükis on toodud välja programmide installeerimised, seadistamised ja kasutamisjuhendid. Viiendas peatükis kirjeldatakse skriptitud väljade, visuaalide ja *dashboard*i loomine ning analüüs.

2 Teoreetiline taust

Antud peatükis on kirjeldatud mis on logi ja selle tähtsus. Samuti on räägitud logimise analüüsist ja visualiseerimisest.

2.1 Logi ja logimise tähtsus

Logi on kronoloogiliselt jäädvustatud sündmuste andmestik, mis on salvestatud kas tekstilisel kujul logifailidena või andmebaasis. Logifail hoiab nimekirja sündmustest, protsessidest, tegevustest, teadetest ja kommunikatsioonist erinevate tarkvararakenduste ja operatsioonisüsteemide vahel [1] . Paljud logifailid on salvestatud tavalise teksti kujul, mis minimeerib faili suurust ja annab võimaluse avada faile tavalise tekstiredaktoriga. Kui hoida logisid struktureeritud kujul, näiteks JSON objektidena, siis ühe kirje jaoks on vaja rohkem ruumi ja sellega kaasneb ka faili suurenemine. Logifaili laiendiks kasutatakse tavaliselt *.log*, et rõhutada faili sisu, kuid sageli kasutatakse laiendit *.txt* [2] . Kuna logifaili sisu on tekstikujul, siis seda on tavaliselt raske töödelda ja analüüsida, ilma eelneva töötluseta. Seega sellised logid, mis vajavad sagedat töötlust, salvestatakse andmebaasi.

Logimine on sarnane inimeste ajaloo kirjutamisega. Üks tähtsamaid põhjuseid, miks inimesed peavad teadma ajalugu, on kunagi tehtud vigade vältimine. Sama kehtib ka logimise kohta. Vanade logide alusel võib leida mustreid, kuidas tekkisid vead või kuidas käituvad kurjategijad. Mustrid aitavad tulevikus vältida selliseid probleeme [3] .

Logid aitavad avastada vigu programmides ja neid parandada. Vea tekkimisel saab logidest leida, millised tegevused olid tehtud enne vea tekkimist ja millised tegevused põhjustasid seda. Logides on salvestatud niinimetatud juhend vea tekitamiseks, mis aitab tarkvaraarendajatel kiiresti seda parandada.

2.2 Logide analüüs

Logide analüüsi kasutatakse sageli süsteemi jõudluse analüüsimiseks, kus näiteks uuritakse süsteemi päringutele vastamise kiirust, mälu kasutust, ressursside kasutust, andmebaasi probleeme ja andmebaasi päringuid [4] .

Logi analüüsi sagedus sõltub analüüsi eesmärkidest. Kui võtta üldiselt, siis analüüsi on vaja läbi viia perioodiliselt. Näiteks, kui analüüsi eesmärgiks on programmi turvalisuse tagamine, siis tasub seda teha vähemalt kord päevas [5] .

2.3 Logide visualiseerimine

Idee piltide kasutamisest andmetest arusaamiseks on kasutusel juba läbi sajandite, alustades kaartidest ja graafikutest 17. sajandil ja kuni sektordiagrammi leiutamiseni 1800 aastate alguses [6] .

Andmete visualiseerimine kirjeldab abstraktse informatsiooni esitamist graafilisel kujul. See aitab tuvastada seaduspärad, tendentsid ja korrelatsioonid, mis vastasel juhul oleks jäänud märkamatuks traditsioonilistes aruannetes, tabelites ja arvutustabelites [7] .

Uurimused näitavad, et inimesed reageerivad visuaalsetele efektidele paremini kui mõnele teisele ärritajale. Inimese aju töötleb visuaalset informatsiooni 60 000 korda kiiremini kui teksti. Tegelikult visuaalsed andmed hõlmavad 90 protsenti informatsioonist, mida edastatakse ajju [7] .

Seega ka suuremahuliste keeruliste andmete mõistmiseks on samuti otstarbekas kasutada diagramme ja graafikuid, mitte tabeleid ja raporteid. Visualiseerimine on universaalne viis andmetest kiiresti ja lihtsalt arusaamiseks. Visualiseerimine aitab näha üldist pilti [6] .

3 Tehnoloogia

Järgnevas peatükis on selgitatud nõuded, tehnoloogia valik ja selle kirjeldus.

3.1 Nõuded

Logide visualiseerimiseks ja analüüsimiseks on vaja programmi, mis peab vastama järgmistele nõuetele:

- Programm peab pakkuma head visualiseerimisvõimalust ehk ilusad modernsed graafikud ja diagrammid, dünaamiliselt muudetavad, eksporditavad.
- Programm peab olema tasuta kätte saadav ilma ajaliste piiranguteta.
- Programm peab olema avatud lähtekoodiga.
- Programm peab olema laiendatav ehk võimalus juurde panna pistikprogramme (*plugin*).
- Programm peab olema veebipõhine.

3.2 Tehnoloogia valik

Logide analüüsimiseks otsitakse alapeatükis 3.1 püstitatud nõuetele vastavat programmi. Tehnoloogia valimisel on koostatud võrdlustabel (Tabel 1), milles on võrreldud 10 programmi kuue hindamiskriteeriumi alusel.

Analüüsimisel jälgiti kuut hindamiskriteeriumit:

- visualiseerimine – programmis on võimalik näha andmeid visualiseeritud kujul. Visuaalid on uudsed ja ilusad, on mitmeid visuaalitüüpe, diagrammid ja

graafikud on dünaamilised. Hindamisviis on skaalal 0–5, 5 kui programm vastab eelpool kirjeldatule, 0 (null) kui visualiseerimisvõimalus puudub või on piiratud.

- tasuta versiooni olemasolu – programmi on võimalik kohe alla laadida, ei ole prooviaega määratud ega tasumisele kuuluvat summat programmi aktiveerimiseks. Hindamisviis on „+”, kui programm on tasuta, või „–”, kui ei ole.

- avatud lähtekoodiga – vajadusel saab programmile lisada funktsionaalsust, luua laiendusi, muuta disaini. Hindamisviis on „+”, kui programm on avatud lähtekoodiga, või „–”, kui ei ole.

- laiendused (*plugins*) – programmil on olemas valmislaiendused, mida saab vajadusel juurde panna või eemaldada. Hindamisviis on „+”, kui programmil on laiendused, või „–”, kui ei ole.

- veebipõhine – programmi põhitöö käib läbi veebiliidese, mis töötab kasutaja brauseris. Hindamisviis on „+”, kui programm on veebipõhine, või „–”, kui ei ole.

- viimane uuendus – mida hilisem on viimane uuendus, seda parem. See näitab, kui vana on programm. Hindamisviis on kuupäeva märkimine kujul päev.kuu.aasta või sellise täpsusega, mis on olemas.

Tabelis (Tabel 1) horisontaalis on paika pandud programmide hindamiskriteeriumid ja vertikaalis on esitatud programmid, mida võrreldakse. Võrdluses osales 10 populaarsemat logide töötlemisprogrammi [18] – [20] . Tabel on sorteeritud viimase kriteeriumi, milleks on programmi viimane uuendus, järgi kahanevalt.

Tabelis (Tabel 1) on esitatud kahe erivärviga analüüsitulemus ja valge taustvärviga on programmide nimed ja kriteeriumid. Lahter on rohelise taustvärviga juhul, kui vastav programm rahustab vastavat kriteeriumit ning punase taustvärviga on märgitud kriteeriumi mitte täitmine. Visualiseerimise kriteeriumi täitmine on hinnatud skaalal 0 – 5, kus 0 tähendab visualisatsiooni võimaluse puudumist või väga piiratud ja 5 tähendab, et visualiseerimisvõimalused on väga head.

Väga head visualiseerimistulemused tähendab, et on palju erinevaid visualiseerimistüüpe, näiteks sektor– ja tulpdiagrammid, visuaalid on modernsed ja dünaamilised, visuaale saab ise juurde lisada.

Antud võrdlusest (Tabel 1) on näha, et programmid Elastic Stack ja Graylog 2 vastavad kõikidele kriteeriumitele, kuid Elastic Stacki visualiseerimisvõime on hinnatud kõrgema hindega ja viimane uuendus on hiljem tehtud. Sellest selgub, et Elastic Stack täidab kõik nõuded, mis olid esitatud peatükis 3.1.

Võrdlustabeli analüüsi tulemusel on valitud välja programm Elastic Stack, mis ei ole iseenesest üks programm, vaid Elastic programmide pakett kolmest programmist, mida täpsemalt kirjeldatakse peatükis 3.3.

Tabel 1. Programmide võrdlustabel.

Programm / Kriteerium	**Visuali–seerimine**	**Tasuta versiooni olemasolu**	**Avatud lähte – koodiga**	**Laiendused (*plugins*)**	**Veebi – põhine**	**Viimane uuendus**
Elastic Stack [8]	5	+	+	+	+	04.05.2017
Splunk [9]	5	–	–	+	+	24.04.2017
Graylog 2 [10]	4	+	+	+	+	04.04.2017
GoAccess [11]	3	+	+	–	+	07.03.2017
WebLog Expert [12]	2	+	–	–	–	08.12.2016
AWStats [13]	1	+	+	+	-	05.08.2016
Sawmill [14]	5	–	–	–	+	08.2016
Visual Log Parser [15]	0	+	+	–	–	30.09.2013
Webalizer [16]	3	+	–	–	+	26.08.2013
LogParser [17]	2	+	–	+	–	20.04.2005

3.3 Elastic Stack

Elastic Stack on kolmest avatud lähtekoodiga toote kollektsioon – Elasticsearch, Kibana, Logstash või Elasticsearch, Kibana, Beats.

Elasticsearch on Elastic Stacki keskmeks, mis on andmebaas logide salvestamiseks ja otsingute teostamiseks. Logstash võtab vastu logid, transformeerib need ja ekspordib andmed erinevatesse sihtpunktidesse, näiteks Elasticsearch'i. Kibana on visualiseerimise kiht, mis töötab Elasticsearch'i peal [21] .

Beats on kerge andmete edastaja, mida paigaldatakse kasutaja serveritesse esindajatena (*agent*) teatud tüübiga andmete saatmiseks Elasticsearch'i. Beats kasutab vähem süsteemi ressursse kui Logstash [22] , [23] .

Elastic Stacki programmide paigaldamisel peab arvestama sellega, et versioonid peavad kõikidel programmidel olema sama, kuna erinevate versioonidega programmid ei pruugi koos töötada. Elastic Stacki pidevalt uuendatakse ja uuendused ilmuvad sageli, seega peab viitama ka nende versiooni uuendamisele. Enne uuele versioonile üleminemist on vaja tutvuda Eastic'u dokumentatsiooniga, et uuendus õigesti viia läbi ja midagi ei läheks katki [24] .

Antud töös on kasutatud Elasticsearch, Logstash ja Kibana.

3.3.1 Elasticsearch

Elasticsearch on JSONil baseeruv, hajutatud, RESTful otsimis– ja analüüsimismehhanism, mis on ehitatud otsingumootori teegile nimega Lucene [25] . Tänu hajutatud arhitektuurile võib Elasticsearch laieneda kuni tuhande serverini ja talletada andmeid petabaidi suurusjärgus [26] .

Uuringud näitavad, et samal andmehulgal, kus SQL päringud võivad võtta üle 10 sekundi vastuse tagastamiseks, tagastab Elasticsearch vastuse vähem kui 10 millisekundiga [26] . Elasticsearch päringud kirjutatakse lihtsas keeles Query DSL, mis on JSON stiilis domeenispetsiifiline keel ehk DSL [27] .

Indekseerimisoperatsiooni ajal Elasticsearch konverteerib töötlemata andmed, milleks võivad olla logifailid või teatefailid, sisemisteks dokumentideks ja salvestab need põhi

andmestruktuuri, mis on sarnane JSON objektiga. Iga dokument on lihtne võtmete ja väärtuste kogum, kus võtmeteks on *string* ehk sõne ja väärtuseks on üks andmetüüpidest – sõne, number, kuupäev või massiiv [26] .

Elasticsearch ei ole relatsiooniline andmebaas ning ei luba päringutes kasutada ühendamist ja alampäringuid, seega enne andmete sisestamist Elasticsearch'i on vaja viia läbi andmete denormaliseerimine. Elasticsearch täidab täistekstotsinguid väga kiiresti, kuna päringud toimuvad indeksite alusel, mis oluliselt vähendab päringu täitmisel loetud andmete hulka [26] .

3.3.2 Logstash

Logstash on andmete kogumismootor, mis võimaldab reaalajas andmete konveiertöötlust. Logstash saab dünaamiliselt ühendada andmed erinevatest allikatest ja normaliseerida need vastavalt nõuetele [28] .

Logstash oli loodud logide töötlemiseks, kuid selle võimalused ei piirdu ainult logidega. See võtab vastu andmed kõikides suurustes ja formaatides. Logstash'ile on loodud üle 200 pistikprogrammi (*plugin*) ja see on paindlik uue pistikprogrammi loomiseks. Logstash on avatud lähtekoodiga programm, kuid probleemiks on suur hulk pistikprogramme, mille arendus jäi pooleli ja esineb palju vigu. Vigased pistikprogrammid võivad rikkuda struktureeritud teadet, mida Logstash genereerib [28] .

Logstash on süsteem, mis võtab vastu, töötleb ja väljastab logid struktureeritud kujul. Logstash töötleb tavalist informatsiooni rida nii, et tulemuseks on struktureeritud ja täiendatud andmed JSON formaadis. Logstash'i üks peamine kasutusviis on dokumentide indekseerimine andmeladudes, mis nõuavad struktureeritud andmeid, selliseks on näiteks Elasticsearch [29] .

Logstash'i andmetöötlemiskonveier koosneb kolmest etapist: sisendid, filtrid, väljundid. Sisendid genereerivad sündmusi, filtrid muudavad neid ja väljundid saadavad need sündmused teisse kohta. Kõige sagedasemad sisendid on failid failisüsteemist, süsteemilogid ja serverist andmete lugemine [30] .

Filtrid on vahepealsed töötlemisseadmed Logstash'i konveieris, mida saab kombineerida tingimusavaldistega, et saavutada tulemus. Logstash'is sagedasemad filtrid on [30] :

- *grok* – juhusliku teksti parsimine ja struktureerimine.
- *mutate* – väljade ümbernimetamine, eemaldamine, lisamine, muutmine ja asendamine.
- *drop* – sündmuse täielik eemaldamine, milleks võib olla *debug*–sündmused.
- *copy* – sündmuse kopeerimine.
- *geoip* – geograafilise asukoha informatsiooni lisamine IP–aadressi järgi.

Väljundid on Logstash'i konveieri viimane etapp, kus töödeldud andmed võivad liikuda erinevatesse kohtadesse. Populaarsemaks andmete edasisaatmiskohaks on Elasticsearch ja failid kettal [30] .

3.3.3 Kibana

Kibana on analüüsimis– ja visualiseerimisplatvorm disainitud töötamaks Elasticsearch'iga. Kibanat kasutatakse Elasticsearch'is salvestatud andmete otsimiseks, uurimiseks ja visualiseerimiseks. Kibana lihtsustab suure andmehulga mõistmist. Kibanal on lihtne, brauseril baseeruv liides, mis lubab kiiresti luua dünaamilist visualisatsioonide kollektsiooni ehk *dashboard*it, mis näitab reaalajas Elasticsearch'i päringutes toimunud muutusi [31] .

3.4 Elasticdump

Elasticdump on importimis– ja eksportimisvahend Elasticsearch'i jaoks, mis on kirjutatud programmeerimiskeeles JavaScript. Elasticdump on avatud lähtekoodiga programm, mis nõuab Elasticsearch'i versiooni 1.0.0 või kõrgemat [32] .

Elasticdump, nagu ka Elasticsearch, loob indekseid andmete importimisel. Objektide kirjutamisel luuakse uus objekt või kui sama unikaalse identifikaatoriga objekt on juba

olemas, siis see kirjutatakse üle. Faili edastamisel ülekirjutamist ei toimu, vaid teatatakse veast, et fail sellise nimega eksisteerib [32] .

3.5 X–Pack

X–Pack on Elastic Stacki laiendus, mis sisaldab turvalisuse, hoiatuste, jõudluse jälgimise, raportite koostamise ja graafikute loomise funktsioone. X–Packi komponente on võimalik ükshaaval juurde ühendada [36] .

X–Pack sisaldab endas ka masinõpet, mis võimaldab avastada võimalikke anomaaliaid sisestatud andmetes. X–Packi masinõpe funktsioonidesse on sisse ehitatud rollid, mis lihtsustab kontrolli selle üle, millistel kasutajatel on õigus vaadata ja hallata ülesandeid, andmevoogusid ja tulemusi [37] .

Kaubamärgi Elastic all on kolm sellist pistikprogrammi (*plugin*), Shield – turvalise ligipääsu andmetele tagamiseks, Watcher – hoiatuste tegemiseks, Marvel – jõudluse jälgimiseks, mis olid alguses loodud iseseisvate laiuendustena kuni versiooni 2.4.1, mille viimane uuendus oli 28 september 2016. Nende pistikprogrammide (*plugin*) uuemad versioonid kuuluvad nüüd X–Pack alla, mille viimane versioon antud töö koostamise ajal on 5.4.0. X–Packi versioon peab ühtima Elasticsearch'i versiooniga [37] .

Antud töös ei ole X–Packi kasutatud, kuna see on tasuline laiendus ja iga klient saab individuaalse hinnapakkumise. X–Packil on kolm tellimuse tüüpi: *Basic*, *Gold*, *Platinum*, mis erinevad omavahel komponentide hulga poolest. Ainult *Platinum* tellimusega saab kõik X–Packi komponendid [38] .

X–Packis on komponent, mis tagab turvalise ligipääsu andmetele. Selle alternatiivina leidis autor avatud lähtekoodiga (*open source*) programmi Nginx, mis oli kasutusele võetud.

3.6 Nginx

Nginx on avatud lähtekoodiga HTTP ja pöördproksi server (*reverse proxy*), ning üldine TCP/UDP proksi ehk proksiserver (*general TCP/UDP proxy*) [33] . Nginx on orienteeritud suurele jõudlusele, paralleelsusele ja madalale mälukasutusele [34] . Nginx omab ühte ülemprotsessorit (*master–processor*) ja mitut alluvprotsessorit (*worker–processor*). Ülemprotsessori peamiseks eesmärgiks on konfiguratsioonide lugemine ja hindamine ning alluvprotsessorite ülalpidamine. Alluvprotsessorite eesmärk on tegelikke päringute töötlemine. Nginx'i ja selle moodulite töötamisviis on määratud konfiguratsioonifailis [35] .

3.7 Struktuur

Selles alampeatükis on kirjeldatud Elastic Stack ja teiste programmide koostöö.

Joonisel (Joonis 1) on näidatud struktuuri ülesehitus. Töö põhistruktuuriks on andmete sisestamine Logstash'i ja Elasticsearch'i saatmine, seejärel on Elasticsearch'is salvestatud andmed nähtavad läbi veebiliidese Kibana.

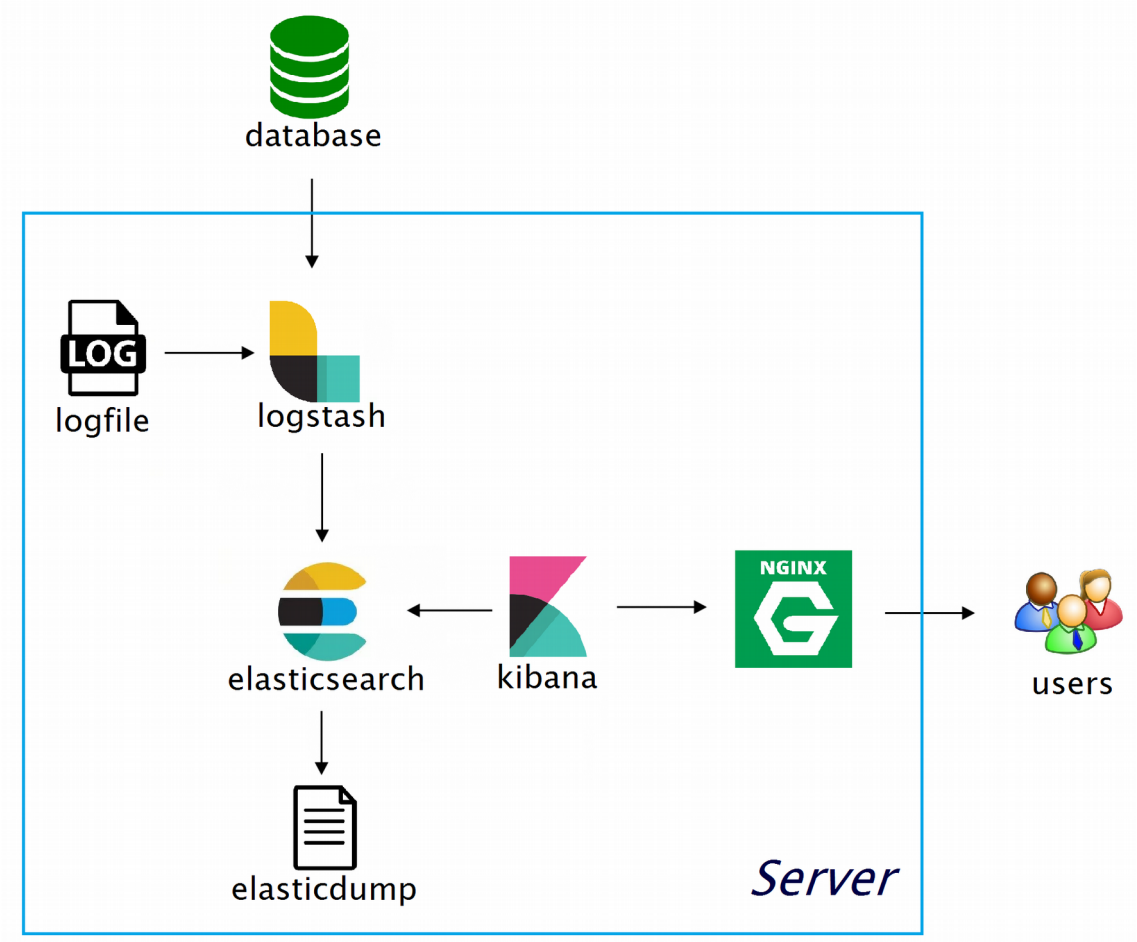
Logstash võtab vastu erinevat tüüpi sisendeid. Antud töös on sisendina kasutatud logifaili ja välisandmebaasi. Logifail asub samas serveris Logstash'iga, andmebaas aga asub teises serveris, eraldi Logstash'ist. Logstash töötleb andmeid ja saadab need Elasticsearch'i, kus need salvestatakse indekseeritult. Läbi Kibana saab Elasticsearch'is salvestatud andmeid uurida, visualiseerida, analüüsida.

Elasticsearch, Logstash ja Kibana on eraldi teenused, mis võivad asuda erinevates serverites. Käesolevas töös on need pandud tööle ühes serveris.

Kuna Logstash, Elasticsearch ja Kibana on pandud tööle serveris *localhost'*ile, siis selleks, et tagada väljastpoolt ligipääs sellele, on pandud juurde Nginx server, mis tagab turvalise ligipääsu ainult graafilisele liidesele Kibanale. Kibanas on võimalus eksportida ja importida salvestatud diagramme ja graafikuid ning *dashboard*e ehk visuaalide kogumit, aga Elasticdump võimaldab lisaks eksportida ja importida kogu ülejäänud informatsiooni nagu skriptitud väljad, indeksid, sisestatud andmed. Elasticdump võimaldab teha niiöelda *backup'*i Elasticsearch'is salvestatud kõikidest andmetest,

milleks on kõik logiandmed, Kibanas loodud skriptitud väljad, visualisatsioonid ja *dashboard*id. Elasticsearch on selle struktuuri keskmeks, kuna seal on salvestatud kõik andmed, seal toimuvad otsingud ja sinna on salvestatud kõik Kibanas loodud objektid.

Käesolevas töös Logstash võtab vastu kas logifaili, mis asub programmiga ühes serveris, või loob ühenduse välisserveriga, kus asub andmebaas logidega, ja pärib sealt andmeid.



Joonis 1. Struktuur.

4 Metoodika

Selles peatükis on kirjeldatud algandmete struktuuri, programmide installeerimist, seadistamist ja käivitamist.

4.1 Esialgsed andmed

TTÜ õppekeskkonna ained.ttu.ee logid on salvestatud andmebaasi, kus on kogunenud üle nelja miljoni kirjet.

Esialgsed andmed on testkeskkonna andmebaasist tehtud päringu tulemus ja esitatud failina, mille sisu on struktureerimata kujul. Logifailis on 89 836 kirjet.

Sturktureeritud andmete näiteks võib tuua küsimustiku, milles on antud küsimused koos võimalike vastustega. Sellist tüüpi andmeid on lihtsam analüüsida. Struktureerimata andmete näiteks võib tuua avatud intervjuu vormis küsitluse, mis sarnaneb vabas vormis vestlusele, kus intervjueerija ei esita konkreetset küsimust, mis eeldab lühivastust, vaid suunab vastajat teatud teemadest rääkima [39] .

Seega, andmed, mida analüüsitakse on struktureerimata kujul salvestatud logifailidesse. Esialgsed andmed on MySQL andmebaasist päritud ja tabulaatoriga eraldatud. Logifail koosneb kirjetest, mis on eraldatud uue reaga. Iga kirje ehk rida koosneb 21 väärtusest, mis on omavahel eraldatud tabulaatoriga (Joonis 2).

```
80411   \\core\\event\\user_graded  core    graded  user    grade_grades
    -31338  u   1   232 50  6   197 6   187 0
a:3:{s:6:"itemid";i:-31338;s:10:"overridden";b:0;s:10:"finalgrade";d:1;}
    1461059727  \N  193.40.250.236  \N
```

Joonis 2. Logifaili kirje näide.

TTÜ õppekeskkonnas ained.ttu.ee töötab Moodle, mis on avatud lähtekoodiga õppekeskkond ja kursuste haldussüsteem [40] . Moodle veebilehel on kirjeldatud logide tabeli (Joonis 3) väljade tähendusi [41] , [42] , [43] .

Field	Type	Null	Key	Default	Extra
id	bigint(10)	NO	PRI	NULL	auto_increment
eventname	varchar(255)	NO			
component	varchar(100)	NO			
action	varchar(100)	NO			
target	varchar(100)	NO			
objecttable	varchar(50)	YES		NULL	
objectid	bigint(10)	YES		NULL	
crud	varchar(1)	NO			
edulevel	tinyint(1)	NO		NULL	
contextid	bigint(10)	NO	MUL	NULL	
contextlevel	bigint(10)	NO		NULL	
contextinstanceid	bigint(10)	NO		NULL	
userid	bigint(10)	NO	MUL	NULL	
courseid	bigint(10)	YES	MUL	NULL	
relateduserid	bigint(10)	YES		NULL	
anonymous	tinyint(1)	NO		0	
other	longtext	YES		NULL	
timecreated	bigint(10)	NO	MUL	NULL	
origin	varchar(10)	YES		NULL	
ip	varchar(45)	YES		NULL	
realuserid	bigint(10)	YES		NULL	

Joonis 3. Andmebaasi tabeli skeem.

4.2 Programmide installeerimine

Töös on kasutatud Elasticsearch, Logstash, Kibana, Nginx, Elasticdump. Programme on kirjeldatud peatükis 3.

Programmide Elasticsearch, Logstash, Kibana alla laadimisel peab jälgima, et nende versioon oleks kõikidel sama. Antud töös on kasutatud versioon 5.3.1.

Elasticsearch ja Logstash vajavad töötamiseks programmeerimiskeelt Java, soovitatav on kasutada Java 8 [44] .

Java ja Nginx installeerimiseks on kasutatud juhendit [45] . Selles juhendis on samuti kirjeldatud Elastic Stacki installeerimine, kuid seal on kasutatud vana versioon.

Elastic Stacki ehk Elasticsearch, Kibana ja Logstash installeerimiseks on kasutatud Elastic kodulehel olevaid juhendeid [46] , [47] , [48] .

Elasticdump töötamiseks on vaja Node.js pakettide haldurit ehk npm, mille alla laadimiseks on kasutatud juhendit [49] . Node.js installeerimisel peab jälgima, et selle tulemusel loodud Node.js kodukausta nimi oleks *node,* mitte *nodejs,* kuna Elasticdump on seadistatud Node.js kodukausta otsimiseks kasutama *node.* Elasticdumpi installeerimiseks on kasutatud juhend [32] .

4.3 Seadistamine

Nginx seadistamiseks on kasutatud juhendit [45] .

Järgnevalt on kirjeldatud Logstash'i kahte konfigureerimisviisi. Esimesel juhul Logstash'i sisendiks on fail ja teisel juhul sisendiks on andmebaas välisserverist.

4.3.1 Logstash logifailist lugemine

Logstash'i konfiguratsioonifail koosneb kolmest *plugin*ast: *input, filter, output*. *Input* ehk sisendis määratakse faili või failide asikoht sättega *file*, milles määratakse failide tüüp, asukoht ja alguspunkt (Joonis 4).

```
file {
    type => "logs"
    path => "/home/logs/moodle_logs.log"
    start_position => "beginning"
}
```

Joonis 4. Logstash'i failist lugemise konfiguratsioonifaili sisendi seadistus.

Teine konfiguratsioonifaili osa on *filter*, kus määratakse muster, millele allub logifaili sisu ja väljade konverteerimised.

Logstash'is on andmete struktureerimiseks erinevaid laiendusi [50] . Antud töös on kasutatud *plugin* Grok [51] .

Grok töötleb andmed rea kaupa, seega kui rida ei allu Grok–i mustrile, siis vaikimisi jäetakse see rida vahele, kuid on võimalik seadistada nii, et rida ei jäeta vahele, aga see rida jääb struktureerimata kujule. Grok–i abil võib ka mustri asemel määrata kaust, kus on üks või mitu mustrifaili.

Grok näites (Joonis 5) on esitatud osa mustrist. Mustri sisuks on väljade nimed ja andmetüübid. Antud näites ei ole näidatud kõiki välju. *Match* on see osa, mis kirjeldab mustrit. Selle sees on esitatud *message.* Grok-is näites on väljas *message* esitatud muster, millele peab alluma andmerida ja *message* on ka välja nimeks, mis luuakse andmerea töötlemise tulemusena, kuhu salvestatakse esialgne andmerida muutumatul kujul. Mustris on määratud väljad koos andmetüübiga ja kuna logifailis on rea andmed eraldatud tabulaatoriga, siis ka mustris on väljad eraldatud tabulaatoriga.

Grok mustri väli %{INT:id}, kus *INT* on välja tüüp *integer* ehk täisarv ja *id* on välja nimi, mida luuakse. Väli *%{GREEDYDATA:timecreated:number}*, kus *GREEDYDATA* on välja tüüp, mis võtab vastu suvalise andmejuppi: sõne, arv, ajatempel. Viimane element *number*, mis on välja nime *timecreated* järel, on samuti välja tüüp, kuid see on prioriteetsem ehk alguses vaadatakse, kas pärast välja nime on määratud tüüp, kui on, siis püütakse andmed teisendada sellesse tüüpi ja kui ei, siis tekib viga, selle asemel kasutatakse välja nime ees olevat tüüpi. Kui välja nime ees olev tüüp ka ei sobi, siis see andmerida ei allu mustrile.

Väljas *(?<crud>[crud])* on märgitud < > nurksulgudesse välja nimi ja selle järgi kohe muster, millele peab selle välja sisu alluma. *[crud]* tähendab, et välja sisuks peab olema üks tähemärk valikust, näiteks *c* [51] .

tag_on_failure abil salvestatakse vea olukorras märge väljasse *tags. add_field* abil lisatakse uus väli nimega *eventName* ja väärtusega *grok*, mis näitab ära, et andmed on sisse loetud *plugin*a Grok kasutamisel.

```
grok {
    match=> {
        "message"=>
             "%{INT:id}   %{GREEDYDATA:objectid:int}(?
<crud>[crud])%{GREEDYDATA:timecreated:number} %
{GREEDYDATA:ip:ip}"
             }
    tag_on_failure => [ ]
    add_field=> {
        "eventName"=>"grok"
    }
}
```

Joonis 5. Grok mustri esitamise näide.

Filtri teine osa on *if*-laused, millega eemaldatakse kõik väljad, mille väärtuseks on tühiobjekt. Antud andmebaasi skeemi järgi on *nullable*-väljadeks: o*bjecttable, objectid, courseid, relateduserid, other, origin, ip, realuserid*.

Filtri viimaseks osaks on ajatempli teisendamine loetavale kujule ja IP-aadressi konverteerimine (Joonis 6). Ajatempliks on väli *timecreated*, mis on *Unix Timestamp* formaadis. Välja *ip* alusel lisatakse informatsioon geograafilisest asukohast [52] .

```
date {
    match => [ "timecreated","UNIX" ]
    target => "timecreated"
}
geoip {
    source => "ip"
}
```

Joonis 6. Logstash'is ajatempli teisendamine ja IP–aadressi töötlemine.

Logstash'i konfiguratsioonifaili viimane osa on *output*, kus on määratud andmete saatmiskoha Elasticstack'i andmed: veebiaadress ja indeksi nimi, mida Elasticsearch kasutab indeksi loomisel (Joonis 7).

```
elasticsearch {
    hosts => "http://localhost:9200"
    index => "moodle-test_1"
}
```

Joonis 7. Logstash'i failist lugemise konfiguratsioonifaili *output*–i seadistus.

4.3.2 Logstash andmebaasist lugemine

Teine Logstash'i seadistus on andmete otse andmebaasist lugemiseks. Selleks on vaja panna peale Logstash'i *plugin* ja vastava andmebaasi draiverid, antud juhul on kasutusel MySQL andmebaasi draiverid. Pärast allalaadimist asuvad draiverid kaustas */usr/share/java/mysql.jar* [53] .

Logstash'i *plugin* peaks viimase Logstash'i versiooni alla laadimisel kohe olemas olema, aga kui seda ei ole, siis saab alla laadida juhendi järgi [54] .

Logstash sisendiks on andmed välisandmebaasist, selleks on vaja luua andmebaasi ühendus. *Input'*is (Joonis 8) peab seadistama *jdbc*, kus on vaja määrata eelnevalt alla laetud draiveri asukoht, mis käib sätte *jdbc_driver_library* alla. Sätesse *jdbc_driver_class* määratakse draiveri klass. Järgmine osa on andmebaasiga ühenduse loomine. Selleks on vaja määrata sätte *jdbc_connection_string* alla andmebaasi aadress koos pordi ja andmebaasi nimega ning *jdbc_user* alla andmebaasi kasutajanimi ja *jdbc_password* alla parool [57] .

Kuna andmebaasis, millest tehakse päring, on umbes neli miljonit rida, siis on vaja piirata sisse lugemisridade arvu, kuna Logstash püüab kõik andmed alguses mällu lugeda ja seejärel hakab neid töötlema: sätte *jdbc_fetch_size* parameetriks määratakse 1000 rida.

Eelviimane rida Logstash'i sisendis *jdbc*–s on *schedule* ehk käivitamiseplaan, mis määrab Logstash'i käivitamissagedust. Antud konfiguratsioonis on määratud igaminutiline uuendus [57] . Iga viie minuti tagant uuendamiseks peab esimese tärni juurde lisama */5*, kus 5 on minutite arv: "*/5 * * * *", iga viie tunni tagant uuendamiseks: "0 */5 * * *". Igal viiendal nädalapäeval uuendamiseks peab *schedule*

olema määratud järgmiseks: “0 0 * * 5”, kus viimane arv viis tähendab reedet. Nädalapäevade indeksid on 0-6, kus 0 on pühapäev ja 6 on laupäev. Selleks, et Logstash taaskäivitaks andmete uuendamine iga viie kuu tagant ei ole spetsiaalset märget, seega on vaja lihtsalt määrata millistel kuudel peab toimuma taaskäivitus: “0 0 1 5,10 *”, kus 5,10 tähendab, et uuendus toimub maikuus (5) ja oktoobris (10) kesköösel, mida näitab teine element. Kolmas element, mis on üks, määrab, et taaskäivitus toimub esimesel kuupäeval. Kui ühe asemel määrata tärn (*), siis taaskäivitus toimub määratud kuu igal päeval [58] , [59] .

Eelviimane Logstash’i sisendi osa on andmebaasist päringu tegemine. Antud juhul tehakse *select*–päring, kus võetakse kõik väljad, tabelist nimega *logs*.

Viimase osana määratakse SQL päringute piirang sätetega *jdbc_paging_enabled*, mis näitab, et SQL päring jagatakse mitmeks alampäringuks, et pärida andmed tabelist osade kaupa, ja *jdbc_page_size*, kus määratakse SQL päringuga päritavate andmete maksimaalse arvu [60] , [61] .

```
jdbc {
    jdbc_driver_library => "/usr/share/java/mysql.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"

    jdbc_connection_string => "jdbc:mysql://193.40.111.120:3306/db_name"
    jdbc_user => "username"
    jdbc_password => "password"
    jdbc_fetch_size => "1000"

    schedule => "* * * * *"
    statement => "SELECT * FROM logs"
    jdbc_paging_enabled => "true"
    jdbc_page_size => "50000"
}
```

Joonis 8. Logstash’i sisendi seadistus andmebaasist lugemiseks.

Andmebaasi parooli võib määrata failina, kus on salvestatud parool. Samuti võib päringu jaoks luua eraldi faili ja määrata selle asukoht kasutades *statement* asemel *statement_filepath* ja selle väärtuseks on faili nimi jutumärkides [62] .

Logstash'i *filter* on failist lugemise ja andmebaasist lugemise puhul erinev, kuna failist lugemisel pidi andmed struktureerima, aga andmebaasist lugemisel saab Logstash kasutada andmebaasi tabeli skeemi: väljade nimesid ja nende tüüpe.

Kuupäeva ja IP-aadressi konverteerimine jääb samaks kui oli määratud eelmises alampeatükis 4.3.1 .

Logstash'i väljundi o*utput* (Joonis 9) koosneb kahest Logstash'i väljundi *plugin*ast: *stdout* ja *elasticsearch*. Logstash'i *plugin* s*tdout* kasutatakse andmete printimiseks konsooli, mis on kasulik näiteks sisse loetud andmete jälgimiseks. *Stdout*is on määratud koodeki *plugin codec,* mis dekodeerib JSONi formaadis andmevoo, kus iga JSON objekt on ühel real ja objektid on omavahel eraldatud uue reaga [55] , [56] . Logstash *plugin elaticsearch*'i seadistusse on lisaks Elasticsearch'i URI ja indeksi nimele lisatud dokumendi tüüp ja dokumendi rea unikaalne identifikaator [57] .

```
stdout {
    codec => json_lines
}
elasticsearch {
    hosts => "http://localhost:9200"
    index => "jdbc_test1"
    document_type => "data"
    document_id => "%{id}"
}
```

Joonis 9. Logstash'i andmebaasist lugemise konfiguratsioonifaili *output* seadistus.

4.4 Programmide käivitamine

Operatsioonisüsteemis Ubuntu pannakse programmid tööle teenustena. Esimesena peab olema tööle pandud Elasticsearch, kuna sellega püüavad ühendust saada Logstash, et andmeid sinna edastada, ja Kibana, et andmeid lugeda. Elasticsearch'i käivitamiseks on kasutatud Joonis 10 näidatud käsku. Kibana ja Nginx käivitamine käib samal põhimõtel kui Elasticsearch.

```
sudo -i service elasticsearch restart
```

Joonis 10. Elasticsearch käivitamiskäsk.

Juhul, kui on vaja näha näiteks Kibana vahepealseid logisid, siis on vaja avada Kibana kodukaustas käsurida ja panna tööle Kibana fail, mis asub *bin*-kaustas. Sama loogika järgi saab panna tööle ka Elasticsearch, kui on vaja logisid näha.

Kibanaga võib tekkida probleem, et see ei lähe tööle teenusena, siis saab niiöeda lükata programmi töö tagaplaanile läbi käsurea (Joonis 11) [64] .

```
sudo nohup bin/kibana &
```

Joonis 11. Kibana käivitamiskäsk.

Kui Elasticsearch ja Kibana juba töötavad, siis saab panna tööle ka Logstash'i. Selle käivitamine erineb Elasticsearch'i ja Kibana käivitamisest. Logstash'i käivitamisel peab määrama konfiguratsioonifail (Joonis 12) [65] . Logstash'i automaatseks uuendamiseks lisatakse käsule parameeter *--config.reload.automatic* [66] .

```
/usr/share/logstash/bin/logstash
-f
/etc/logstash/conf.d/logstash-1.conf
--config.reload.automatic
```

Joonis 12. Logstash'i käivitamiskäsk.

Kuna Logstash teeb muudatusi kaustas */usr/share/logstash*, siis peab arvestama sellega, et kasutajal peavad olema vastavad õigused.

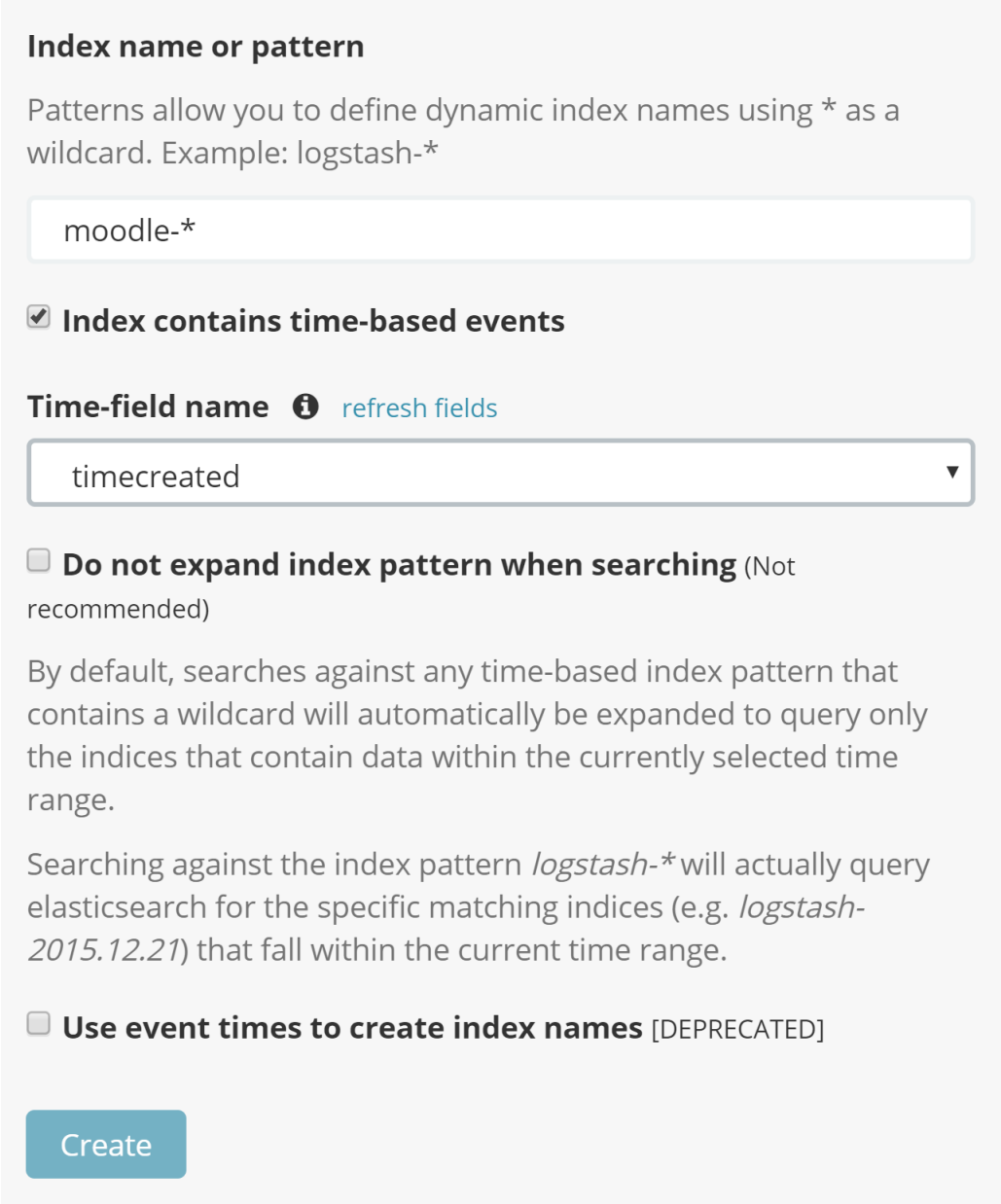
4.5 Tööga alustamine Kibanas

Kui kõik programmid on tööle pandud, siis selleks, et Kibanas näha andmeid on vaja luua Kibana veebiliideses *Index Pattern* ehk indeksi muster, mille alusel leitakse andmed Elasticsearch'ist ja kuvatakse Kibanas.

Indeksi mustri loomiseks peab brauseris avama Kibana veebiliidese ja sisse logima. Kui ei ole loodud veel ühtegi mustrit, siis vaikimisi ilmub vaade indeksi mustri loomiseks (Joonis 13).

Juhul, kui avaneb mingi muu vaade, siis vasakus menüüs peab valima *Managament* ning avanenud vaates valima *Index Patterns*. Selles vaates peab vajutama nuppule *+Add New,* mis asub üleval vasakul, millega avaneb indeksi loomise vaade.

Välja nimega *Index name or pattern* peab kirjutama Logstash'i konfiguratsioonifailis määratud indeksi nime, näiteks *moodle-test1,* või mustrit, näiteks *moodle-**. Väli *Index contains time-based events* peab olema märgistatud, mis tähendab, et andmetes on olemas vähemalt üks aja väli, mille alusel saab andmeid sisse lugeda Kibanasse. Antud juhul on valitud *timecreated*, mis on logi kirje loomise ajatempel.



Joonis 13. Kibanas indeksi loomise vorm.

Logiandmete vaatamiseks Kibanas peab vasakul menüüst valima *Discover*, mis otsib andmeid Elasticsearch'ist indeksi alusel, mis oli Kibanas loodud. Kuna andmed on sisestatud spetsiifilise ajatempli järgi, siis võib olla on vaja määrata ka vastav ajavahemik paremal üleval menüüs viimane valik on *Time Range.*

4.6 Elasticdumpi kasutamine

Elasticdumpi abil saab Elasticsearch'is salvestatud indekseid eksportida ja importida JSON kujul. Kuna Kibanas loodud indeksid, visualisatsioonid, *dashboard*'id salvestatakse Elasticsearch'is indeksina *.kibana*, siis ka neid saab importida ja eksportida.

Elasticdump käsk antud juhul koosneb programmi nimest *elasticdump* ja kolmest parameetrist: *input, output, type*.

Elasticdump sisendiks ja väljundiks saavad olla nii fail kui ka URI, tüübiks saab olla *analyzis, mapping, data.*

Andmete eksportimisel on sisendiks Elasticsearch'i URI, mis käib parameetri *input* alla. Väljundiks on faili nimi *.json* laiendusega, mis käib parameetri *output* alla. Tüübiks on määratud *data* parameetri *type* alla.

Kui väljundis määratud fail on olemas, siis visatakse erind (*exception*), vastasel juhul luuakse fail käsus määratud nimega kausta, kuhu on käsurida navigeeritud ehk kui soovitakse fail salvestada kindlasse kohta, siis peab väljundis ehk *output* määrama absoluutne *path*.

Kibanas salvestatud visualisatsioonide ja *dashboard*ite eksportimiseks on vaja *input*-parameetris lisada URI lõppu Elasticsearch'i indeks */.kibana*, kus Kibana objektide andmed on salvestatud (Joonis 14) [67] . Eksportimisel luuakse fail nimega *kibanaindex.json*.

```
elasticdump
    --input=http://localhost:9200/.kibana
    --output=kibanaindex.json
    --type=data
```

Joonis 14. Kibana visuaalide ja *dashboard*ite eksportimine.

Andmete importimiseks on vaja sisendisse määrata vastava faili nimi ja väljundisse Elasticsearch URI, mille lõpus määrata vajalik indeks, kuhu on vaja andmed importida (Joonis 15) [67] .

```
elasticdump
    --input=kibanaindex.json
    --output=http://localhost:9200/.kibana
    --type=data
```

Joonis 15. Andmete importimine failist Elasticsearch'i.

Elasticdump võimaldab eksportida Elasticsearch'is salvestatud logid struktureeritud kujul ehk JSON formaadis. Selleks peab sisendiks määrama Elasticsearch URI koos vajaliku indeksiga, väljundiks võib määrata näiteks faili nime ja tüübiks on *data* ehk andmed. Kui jätta indeks üldse määramata, siis eksporditakse kogu informatsioon mis on Elasticsearch'is salvestatud.

5 Visualiseerimine ja analüüs

Antud peatükis on kirjeldatud tegevusi Kibanas, milleks on skriptitud väljade loomine, visualisatsioonide loomine ja *dashboardi* loomine.

5.1 Väljade lisamine

Logiandmete teisendamisel on olemas kaks kuupäeva välja, kus üks *timestamp* on rea Logstash'i abil sisse lugemise aeg ja teine on *timecreated* ehk logi rea loomise aeg. Kuupäev koos ajaga ehk ajatempel ei ole alati parim viis andmete grupeerimiseks, sest neid võib olla lõpmata palju erinevaid. Heaks grupeerimiseks ajajärgi on piirata hulgad, milleks võivad olla nädalapäev või tund.

Kibanas on võimalus lisada skriptitud välju. Vaikimisi skriptimiskeeleks on Painless. Leidub ka pistikprogramme (*plugin*) teiste keelte kasutamiseks. Võimalikud keeled on Groovy, JavaScript, Python, mis on paindlikumad. Vähem paindlikud, aga võib-olla efektiivsemad on Lucene Expressions, Mustache, Java [68] .

Käesoleva töö eesmärke täitmiseks peab lisama juurde kaks välja, mida ei ole logides olemas, kuid mis tuletatakse logirea loomise ajatemplist. Lisatavad väljad on nädalapäev ehk *weekday* ja tund ehk *hour*.

Väljade loomisel on kasutatud keelt Painless. Skriptitud väli luuakse ainult ühe indeksi andmete jaoks.

5.1.1 Väli *weekday*

Peamenüüst vasakul valida *Management*, *Index Pattern* ja valida ilmunud indeksite nimekirjast vajalik indeks, antud juhul *moodle-**. Siit edasi on vaja valida *Scripted fields* ja vajutada nupule *Add Scripted Field*.

Nädalapäeva välja loomisel määrab autor nimeks *weekday*. Eelpool oli mainitud, et keele valikuks on Painless, kuna see on vaikimisi olemas ja sobib antud juhul hästi. Välja tüübiks peab valima *number*, mis on vaikimisi valitud. Üks võimalikest tüüpidest on *date*, aga see ei sobi, kuna tulemusena väli *weekday* ei pea käituma kui *date* ehk kuvama lihtsalt ajatempli asemel päeva nime, vaid ridu peab saama jagada *weekday* alusel ehk seitsme nädalapäeva ja määramata päeva vahel.

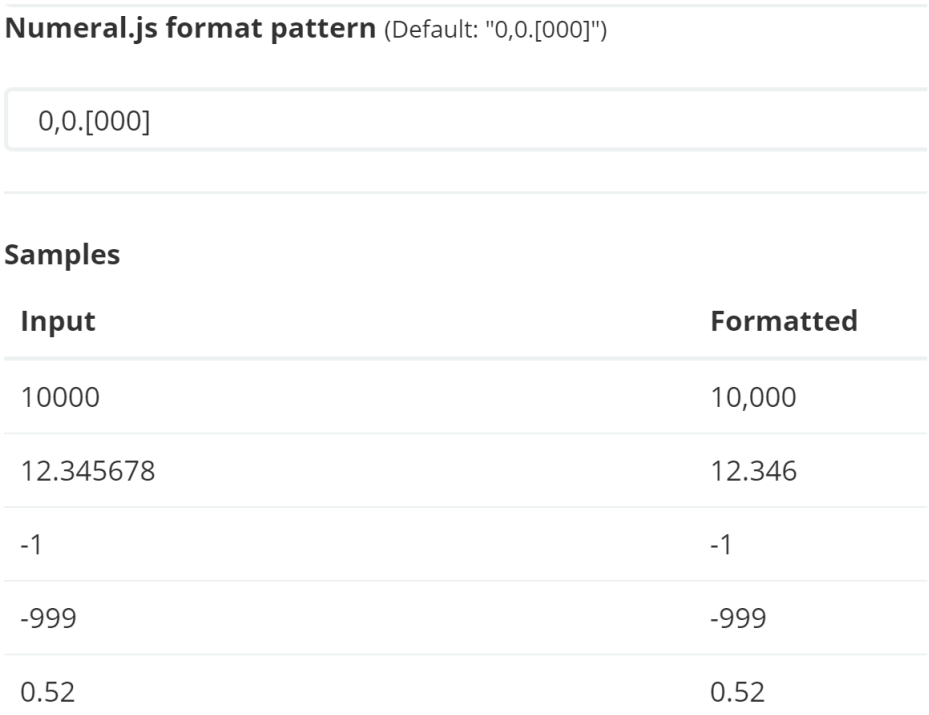
Väljundformaat on *String*, kuna väli hakkab sisaldama nädalapäeva nimetuse sõna. Kõige tähtsamaks osaks välja loomisel on *Script*, mis määrab ära välja sisu. Esimene samm on määrata, millise olemasolevatest väljadest võtta aluseks. Antud juhul on valitud logirea loomise ajatempel *timecreated*, mis on *date*-tüüpi ning sellest eraldatakse nädalapäev numbrilisel kujul funktsiooniga *dayOfWeek* ja salvestatakse muutujasse nimega *x*. Skripti teine osa on numbrist päeva nime kätte saamine ehk *if*-klausli abil saab määrata kui *x* on 1, siis tagasta *Monday*. Kuna Elastic Stack on ingliskeelne, siis ka päeva nimed on kirjutatud inglise keeles, soovi korral saab ka teises keeles kirjutada. Viimase alternatiivina on märgitud *Unknown*, mida kasutatakse juhul, kui tekib mingi ootamatu olukord, kus ei ole ajatemplit märgitud. Lisas 1 on välja toodud skriptitud välja skripti kood

Välja *weekday* kasutamiseks on erisus võrreldes teiste väljadega, kuna selle välja tüübiks on *number*, kuid numbri asemel on välja väärtus salvestatud *String*i kujul. See erinevus on selle pärast, et kuupäevast saab kätte nädalapäeva numbri ja käsitsi asendatakse vastav number vastava päeva nimega. Sellest tulenevalt on vaja visualisatsioonide loomisel kasutada *Advanced* valikut välja juures, mis asetseb tavaliselt paremal all nurgas. Avanenud vaates peab lahtrisse nimega *JSON Input* kirjutama sellise rea: *{"valueType":"string"}*, mis näitab ära, et välja tüüp on sõne ehk *String*.

5.1.2 Väli *hour*

Järgmise skriptitud välja lisamine toimub samal põhimõtel kui *weekday*. Valida indeks, millele on vaja lisada skriptitud väli, minna vaatesse *Scripted Fields* ja valida *Add Scripted Field*. Kuna välja sisuks on tund, mil logirida oli loodud, siis välja nimeks on *hour* ehk tund. Samal põhimõttel on valitud skriptimiskeeleks Painless. Välja tüübiks ja

formaadiks on määratud *number*, kuna välja sisuks peab olema tund numbrina lõigul 0 – 23. Sellise formaadi valikul ilmub lahter nimega *Numeral.js format pattern*, mille vaikimisi väärtuseks on *0,0.[000]*. Selle lahtri allpool on näidatud, millised on lubatud väärtused sellise mustri jaoks (Joonis 16). Kuna tunni näitamiseks on vaja ainult ühte täisarvu, siis määrame selle mustri nulliks ehk lahtrisse on vaja jätta ainult üks number 0.



Joonis 16. Skriptitud välja numbri formaat.

Uue välja loomise tähtsaim osa on skript, kus on vaja näidata, et tuleb kasutada välja *timecreated*, mis on *date* tüüpi ning millest eraldatakse päeva tund funktsiooniga *hourOfDay*. Kui kõik vajalikud lahtrid on täidetud, siis lehe lõpus on nupp *Create field*, mis salvestab uue välja.

Alampeatükis 5.2.1 on näitena kirjeldatud skriptitud väljadega diagrammi loomist.

5.2 Diagrammide loomine

Selles alampeatükis kirjeldatakse kolme visualiseerimistüübi diagrammi loomist.

5.2.1 *Heatmap chart*–tüüpi diagramm

Heatmap ehk soojuskaart on maatriksi kujul esitatud andmed, kus individuaalsed väärtused on esitatud värvina.

Järgnevalt luuakse diagramm populaarsete kellaaegade näitamiseks nädalapäevade järgi (Joonis 18). Paremal menüüst valida *Visualize*, seejärel valida visualisatsioonitüüp *Heatmap chart* ja vajalik indeks, antud juhul *moodle–**. Avaneb vaade, kus on graafiku seadistamiseks võimalik valida kolme *buckets* ehk ämbri tüübi seadistust, milleks on *X–Axis* ehk x–teljel, *Y–Axis* ehk y–teljel ja *Split Chart*, mis võimaldab kas x–telje või y–telje järgi tükeldada diagrammi.

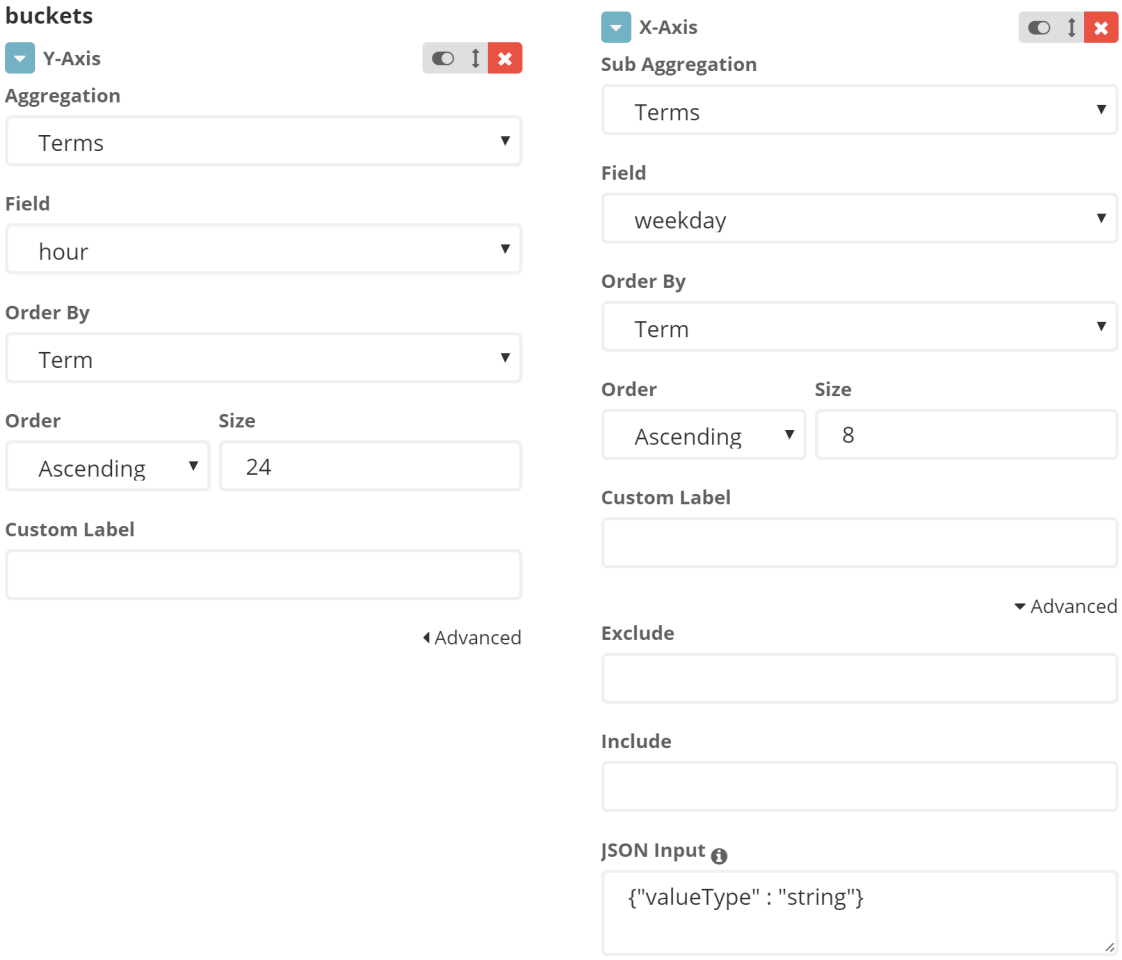
Antud graafiku loomiseks läheb vaja kahte *buckets* ehk ämbrit, milleks on *Y–Axis* ja *X-Axis* (Joonis 17). Esimesena seadistatakse *X–Axis*. Valides *X–Axis* ära, ilmub lahter nimega *Aggregation* ehk koondamine. *Aggregation*i all on esitatud 8 valikut: *Date Histogram*, *Histogram, Ragne, Data Range, IPv4 Range, Terms, Filters, Significant Terms*. Sellest nimekirjast on vaja valida *Terms*. Agregeerimistüübi valimisel ilmub mitu uut välja: *Field*, kust valitakse üks olemasolevatest väljades indeksis, *Order By*, mille vaikimisi väärtuseks on *metric: Count*, mis tähendab, et sorteerimine selle välja järgi käib esinemissageduse järgi, *Order*, mille vaikimisi väärtus on *Descending*. Samuti esinevad väljad nagu *Size*, mis määrab ära kui palju kuvatakse elemente graafikul, selle vaikimisi väärtus on 5 ehk graafikule valitakse 5 esimest unikaalset elementi antud väljast, mida kuvatakse graafikul ja viimane väli on *Custom Label* ehk kasutaja antud nimi väljale.

Kuna tegemist on x–teljega, siis *field* ehk väljaks on vaja valida *weekday*, *Order by* määrame *Term* ning sorteerimisjärjekorra tüübiks on *Ascending* ehk kasvavalt, *size* on vaja määrata 8, kuna selle väljal on 8 võimalikku väärtust 7 nädalapäeva ja *unknown*-väärtus.

Kuna *weekday* on skriptitud väli ja selle tüüp ei ole üheselt määratud ehk välja tüüp on number, kuid välja sisu on sõne. Seega on vaja avada *Advanced* seadistuse ja *JSON Input* välja sisestada objekt, mis ütleb, et välja tüübiks on sõne: *{"valueType":"string"}*.

Sellel etapil, kui x–teljega on lõpetatud, saab rakendada muudatused graafikus vajutades nuppule *Apply changes*.

Nüüd ei ole näha *buckets* ehk ämbrite valikut, vaid on nupp *Add sub–buckets*, millele klikkides ilmub ämbrite valik. Antud graafiku valmimiseks on vaja lisada *Y–Axis* –tüüpi ämber. Ilmub sama lahter agregeerimisvõimalustega, millest on taas vaja valida *Terms*. Edasi peab valima väljaks *hour*, selle sorteerimiseks valida *Term*i järgi kasvavalt. *Size* ehk ämbri suuruseks on 24, kuna välja *hour* võimalikud väärtused on 0-23. *Y-Axis* seadistus on korras ja saab rakendada muudatused vajutades vastavale nupule.

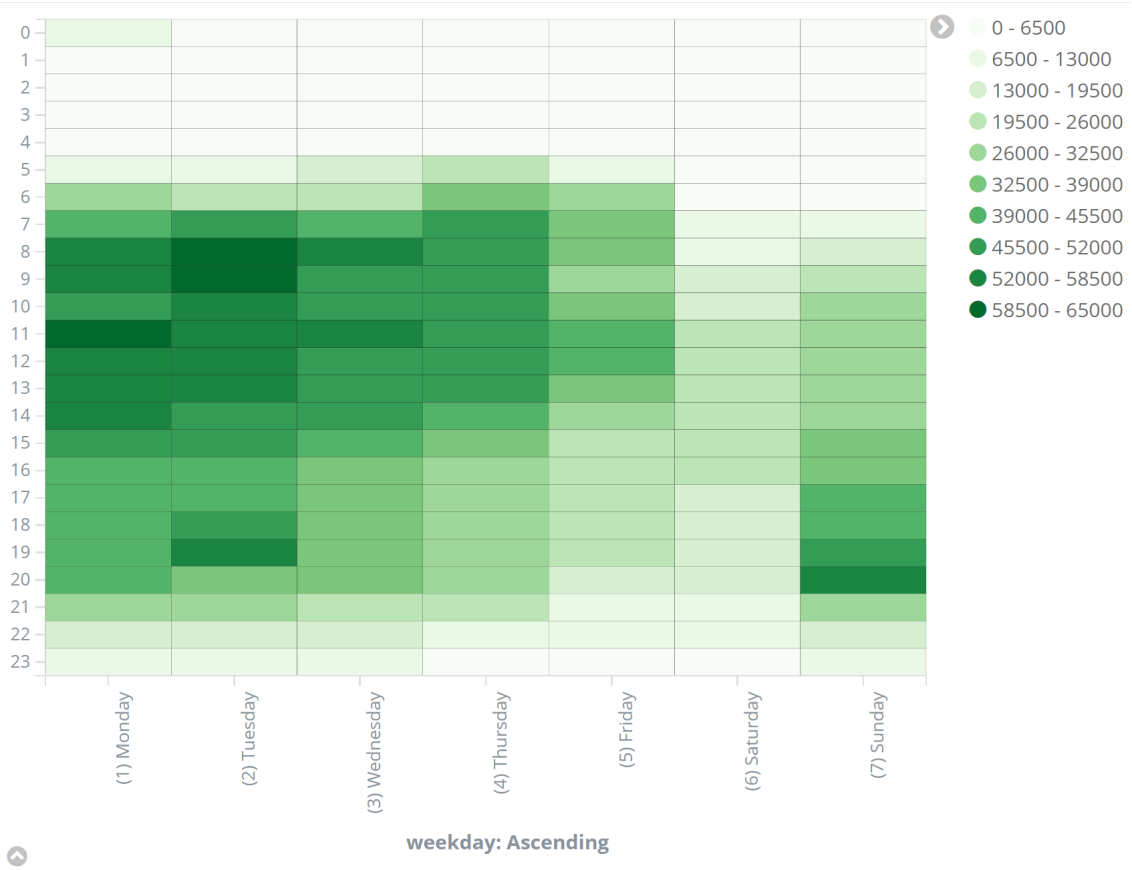


Joonis 17. *Heatmap*–tüüpi diagrammi koostamise vorm.

Kõik senised seadistused on tehtud vaates *Data*. Selle kõrval asub vaade valikutega ehk *Options*, kus saab määrata legendi asukoha, värviskeemi, värviskaala, värvide hulga. Värviskeemis on viis valikut: sinised värvitoonid, rohelised, punased,

hallid värvitooinid ning skeem kollasest punasesse. Värviskaala funktsioone on kolm: lineaarne, logaritmiline ja ruutjuur. Kui olemasolevad valikud ei rahulda vajadusi, siis saab käsitsi määrata vahemikud. Vaikimisi on määratud, et legend asub paremal pool, värvivalik on roheline, värviskaala on lineaarne ning värvide hulk on 4. Antud diagrammi loomisel on vaja värvide hulgaks määrata 10, mis teeb diagrammi detailsemaks.

Visualisatsiooni salvestamiseks on vaja üleval paremal pool asuvast menüüst valida *Save,* sisestada ilmunud lahtrisse diagrammi nimi ning vajutada lahtri all asuvale nuppule *Save*. Loodud diagrammi tulemus on esitatud Joonis 18.



Joonis 18. *Heatmap*–tüüpi visualisatsioon: Populaarsus tundide järgi erinevatel nädalapäevadel.

5.2.2 *Tag cloud*–tüüpi diagramm

Tag cloud ehk sildipilv on visualisatsiooni tüüp, kus kuvatakse tekstandmed, tavaliselt selleks on üksikud sõnad, niinimetatud pilvena. Iga sõna suurus sõltub tema esinemise tihedusest andmetes.

Järgnevalt luuakse visuaal 25 populaarsema kursuse näitamiseks (Joonis 19). Paremal menüüst valida *Visualize*, seejärel valida visualisatsioonitüüp *Tag cloud* ja vajalik indeks, antud juhul *moodle–**. Avaneb vaade, kus on graafiku seadistamiseks võimalik valida üks *bucket* ehk ämber, milleks on *Tags.*

Ämbri *Tags* valimisel ilmub vorm, mille esimene lahter on *Aggregation* ehk agregeerimistüüp ja selles on ainult üks valik *Terms.* Lahtrisse *Field* valitakse väli *courceid.* Andmed sorteeritakse sama välja esinemistiheduse järgi kahanevalt ja sellest võetakse 25 esimest elementi ehk 25 populaarsemat kursust.

Options–vaate alt on võimalik märkida linnuke välja *Show Label* ette, oleks näidatud visuaali nimetus ja saab seadistada visuaali kolme parameetrit: *Text Scale, Orientations, Font Size.*

Joonis 19. *Tag cloud*–tüüpi visuaal 25 populaarsema kursuse *id* näitamine.

5.2.3 *Vertical bar chart*–tüüpi diagramm

Vertical bar chart ehk tulpdiagramm on ideaalne valik olukorras kui on väga palju nõudmisi. See sobib hästi ajaliste andmete ja andmete, mis ei ole seotud ajaga, kuvamiseks.

Järgnevalt luuakse diagramm iga nädalapäeva iga tunni kohta unikaalsete kasutajate arvu kuvamiseks kursuste vahemikude kohta (Joonis 22). Paremal menüüst valida *Visualize*, seejärel valida visualisatsioonitüüp *Vertical bar chart* ja vajalik indeks, antud juhul *moodle–**. Avaneb vaade, kus on diagrammi seadistamiseks võimalik valida *metrics* ja *buckets*. Seadistuse *metriks* all on *Y–Axis* ehk y–telje seadistus, mida võib olla mitu. Vaikimisi on seadistatud *Y–Axis*ses lahtri *Aggregation*i väljaks *Count*, mis seadistab graafiku y–teljel andmeridade arvu näitamiseks. Diagrammi (Joonis 22) loomiseks on vaja agregeerimistüübiks valida *Unique Count* ja väljaks on määratud *userid* ning *Custom Label*'iks on pandud *Unikaalsete kasutajate arv* (Joonis 20). Sellise seadistusega on y–teljel kuvatud unikaalne kasutajate arv.

Joonis 20. *Vertical bar chart*–tüüpi diagrammi seadistusvormi *metric* osa.

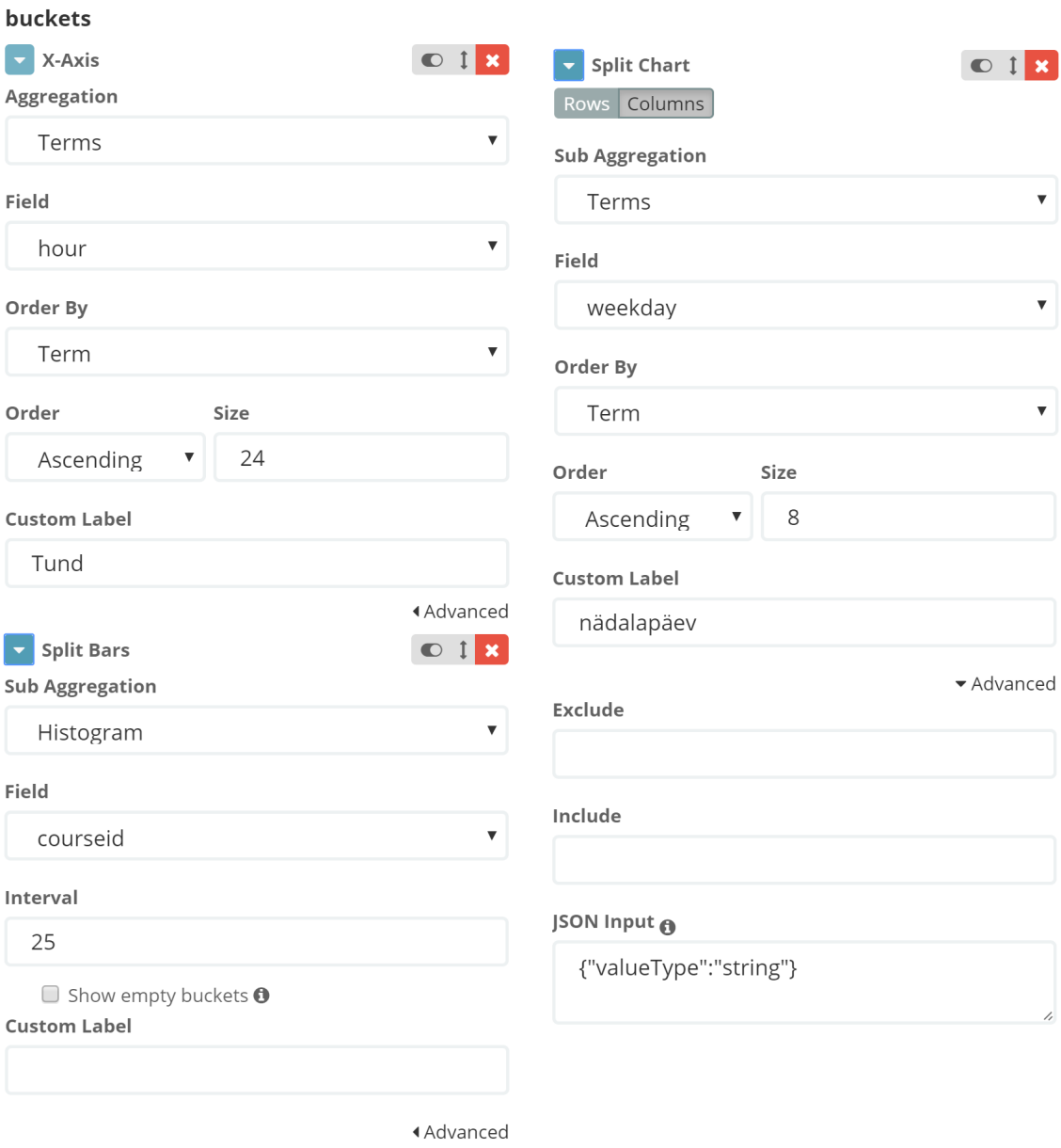
Graafiku edasiseks kujundamiseks peab seadistama kõiki kolme *buckets* ehk ämbri tüübi seadistust, milleks on *X–Axis* ehk x–telje andmed, *Split Bars*, mis võimaldab muuta graafik mitmetasemeliseks, ja *Split Chart*, mis võimaldab kas x–telje või y–telje järgi tükeldada diagrammi (Joonis 21).

X–Axis seadistuses agregeerimistüübiks ehk *Aggregation* lahtris peab valima *Terms,* väljaks valitakse *hour,* kasvavas järjekorras sorteerimiseks on *Order By* lahtrisse valitud *Term* ja lahtrisse *Order* ehk järjekord on valitud *Ascending.* Kuna tunde on 0–23, siis väljasse *Size* määratakse 24, *Custom Label*isse ehk asendusnimeks on määratud *Tund.*

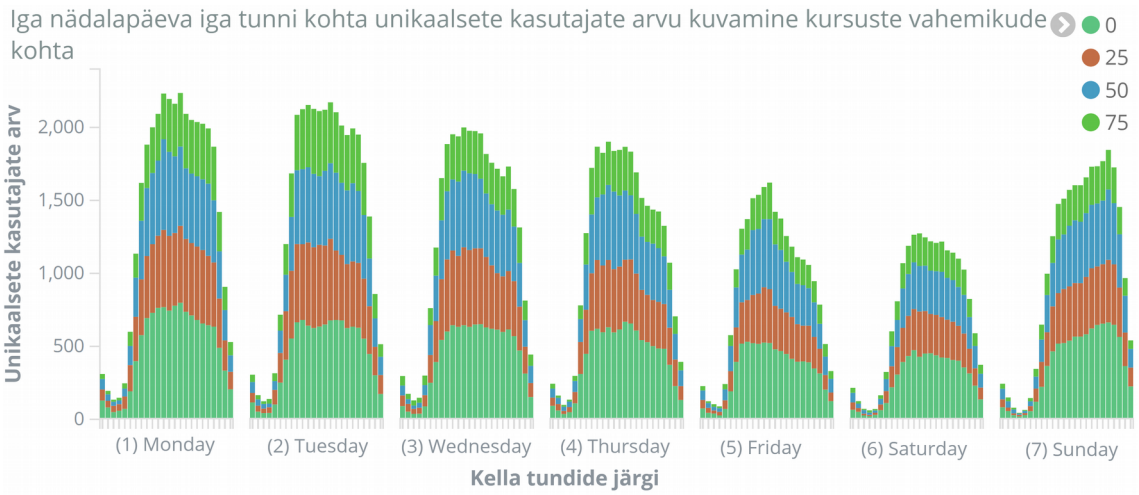
Teiseks ämbriks ehk *bucket* on *Split Chart,* mis peab diagrammi tükeldama *Column*s järgi ehk x–telje järgi. *Field* ehk välja lahtrisse valitakse nädalapäev ehk *weekday,* mida sorteeritakse *Term* järgi kasvavas järjekorras ja mille suuruseks on 8. Väljasse *Custom Label* sisestatakse asendusnimi *nädalapäev.* Kuna kasutatud on väli *weekday,* siis peab *Advanced* seadistustes määrama *JSON Input* välja sisuks *{"valueType":"string"}.*

Kolmas *bucket* ehk ämber on *Split Bars,* mis muudab diagrammi mitmetasemeliseks. Selle esimeseks seadistuseks on *Sub Aggregation* ehk alamagregeerimistüüp, milleks peab valima *Histogram,* väljaks on valitud *courseid.* Viimaseks seadistuseks on *Interval* ehk intervall, mis jagab välja courseid väärtused vahemikudeks, kus ühe vahemiku suurus on 25. Kui kõik seadistused on tehtud, siis saab rakendada muudatused.

Diagrammi tulemus on esitatud Joonis 22, kus igas tulbas näidatud kõikide kursuste vahemikude unikaalne kasutajate arv suhteliselt. Lisas 2 on näidatud üks päev suurendatult.



Joonis 21. *Vertical bar chart*–tüüpi diagrammi seadistusvormi *buckets* osa.



Joonis 22. *Vertical bar chart*–tüüpi diagramm iga nädalapäeva iga tunni kohta unikaalsete kasutajate arvu kuvamiseks kursuste vahemikude kohta. Legendis on näidatud värvile vastavat kursuste vahemiku, mis on märgitud vahemiku esimese kursuse unikaalne identifikaator.

5.3 *Dashboard*i loomine

Dashboard on vaade, kus on kindel valik graafikuid esitatud. Töölaud võimaldab ühendada mitu graafikut, mis näitab logiandmeid erinevatelt vaatenurkadest, kuid kui ühel graafikul valida mingi kindel millegi identifikaator, siis kogu töölaua visualisatsiooni graafikud muutuvad ka.

Dashboardi loomiseks on vaja vasakust menüüst valida *Dashboard* ja avanenudt vaates peab vajutama plussnupule (+). Tulemuseks ilmub salvestatud visualisatsioonide nimekiri. Sellest nimekirjast on vaja valida sobivad diagrammid ühe kordse pealkirja peale klikkimisega ja diagramm ilmub lehe lõpus asuvale *Dashboard*ile. Diagrammi suurust saab muuta paremal all nurgast venitades. Diagrammi üleval paremal nurgas on nupp, mis võimaldab muuta diagrammi asukohta *Dashoard*il. *Dashboard* võimaldab ühendada erinevate indeksite visualisatsioone.

Kui *Dashbord* on valmis, siis paremal pool üleval menüüs on valik *Save*, millele klikkides ilmub lahter *dashboard*i nime jaoks ja linnukese lahter, mis pakub võimalust salvestada *dashboard*iga antud ajavahemikku.

Salvestatud *dashboard*il on üleval menüü, kus saab valida *Add* ja lisada juurde visualisatsioone. Olemasoleva *dashboard*i muutmisel saab seda salvestada uue objektina või salvestada muudatused samas objektis.

Logiandmetes on kursuse id ehk unikaalseks identifikaatoriks on null (0) siis, kui tegemist on süsteemse muudatusega, näiteks kasutaja lisamine, ning kursuse id on üks (1) kui tegemist on õppekeskkonna pealehega. Kuna analüüsida tahetakse ainult reaalseid aine kursuseid, siis peab need andmed välja filtreerima.

Dashboard ühendab erinevaid diagramme ja filtreerimisel muutuvad dünaamiliselt kõik graafikud. Selleks, et diagrammid näitaksid informatsiooni ainult kursuste kohta, ilma süsteemitegevusteta ja pealeheta, on vaja määrata filtris seda, et kui kursuse unikaalne identifikaator on 0 või 1, siis neid ei kuvata. Selleks pilvediagrammil valida kursuse *id* null (0), mille alusel filtreeritakse andmed ära ja kuvatakse ainult selle kursuse andmed. Veebilehe üleval otsingu lahtri alla tekib märge filtrist. Selle kõrval on link *Actions*, kust on vaja valida *Disable*, mis lülitab filtrimise välja ehk kuvatakse kõik andmed ilma filtrita. Seejärel valida pilvediagrammil kursuse *id* üks (1), ja lülitada 0-kursuse filter tagasi, milleks on vaja liigutada kursor filtrile ja ilmuvas valikus märgistada esimest kastikest. Viimaseks sammuks on filtrite inverteerimine, milleks on vaja *Actions* valikus vajutada *Invert*–linki. *Dashboard* on vaja uuesti salvestada, et filtrid ei kaoks ära.

Dashboardi loomise tulemus on esitatud Joonis 23, mille loomisel on kasutatud peatükkides 5.2.1, 5.2.2, 5.2.3 loodud diagramme.

Kui üks kord vajutada vasakul menüüs *Dashboard'*ile ilmub viimati vaadatud *dashboard*, siis teist korda vajutades menüüs ilmub nende nimekiri ja uue lisamise võimalus.

5.4 Analüüs

Moodle õppekeskkonna ained.ttu.ee on kogunenud 4 392 279 rida andmeid. Logiandmeid on kogutud perioodil 28 august 2015 kuni 15 mai 2017. Seega logid sisaldavad infot nelja semestri kohta. Üldiselt on suurema unikaalse identifikaatoriga kursus on uuem, millest võib järeldada, et väiksema identifikaatoriga kursusel on vähem kasutajaid.

Analüüs on tehtud *dashboard*i alusel (Joonis 23), mis on loodud peatükis 5.3.

Diagrammi Populaarsed kellaajad nädalapäeva järgi on näha, et õppekeskkonda ained.ttu.ee kasutatakse kõige rohkem esmaspäeval kell 11 ja teisipäeva hommikul kell 9 ning kõige vähem populaarne nädalapäev on laupäev.

Graafikul on näha, et esmaspäeval kasutatakse õppekeskkonda aktiivselt kella 7–21, kus kõige aktiivsem periood on hommikul 8 ja päeval 11–14. Esmaspäevast neljapäevani on õppekeskkonna kasutus suurenenud ajavahemikus 7–20. Reedesel päeval on aktiivsusperiood kaks korda lühem, aktiivsusperiood jääb ajavahemikku 7–14.

Reedel ja laupäeval on keskkonna kasutus kõige väiksem, kuna tegemist on nädalalõpuga ja tudengid õpivad vähem. Pühapäeval õhtupoole aktiivsus keskkonnas kasvab kuni kell 20.

Kuna diagrammil on näha, et õppekeskkonda kasutatakse kõige rohkem tööpäevadel, siis võib järeldada, et seda kasutatakse nii ülikooli tundide ajal kui ka kodus. Populaarsed ajad õhtul on teisipäeval kell 18–19 ja pühapäeval kell 19–20, mille alusel võib järeldada, et järgmisel päeval, kolmapäeval ja esmaspäeval, on kodutöö tähtaeg.

Kursuste populaarsuse diagrammi alusel (25 populaarsemat kursust), on näha, et kõige populaarsem on kursus unikaalse identifikaatoriga 83 ja temast järgmine kõige populaarsem on unikaalse identifikaatoriga 51. Kolmas kõige populaarsem kursus on identifikaatoriga 71. Seega õppekeskkonna ained.ttu.ee kolm kõige populaarsemat kursust on unikaalse identifikaatoriga 83, 51 ja 71, mis on järjestatud populaarsuse järgi.

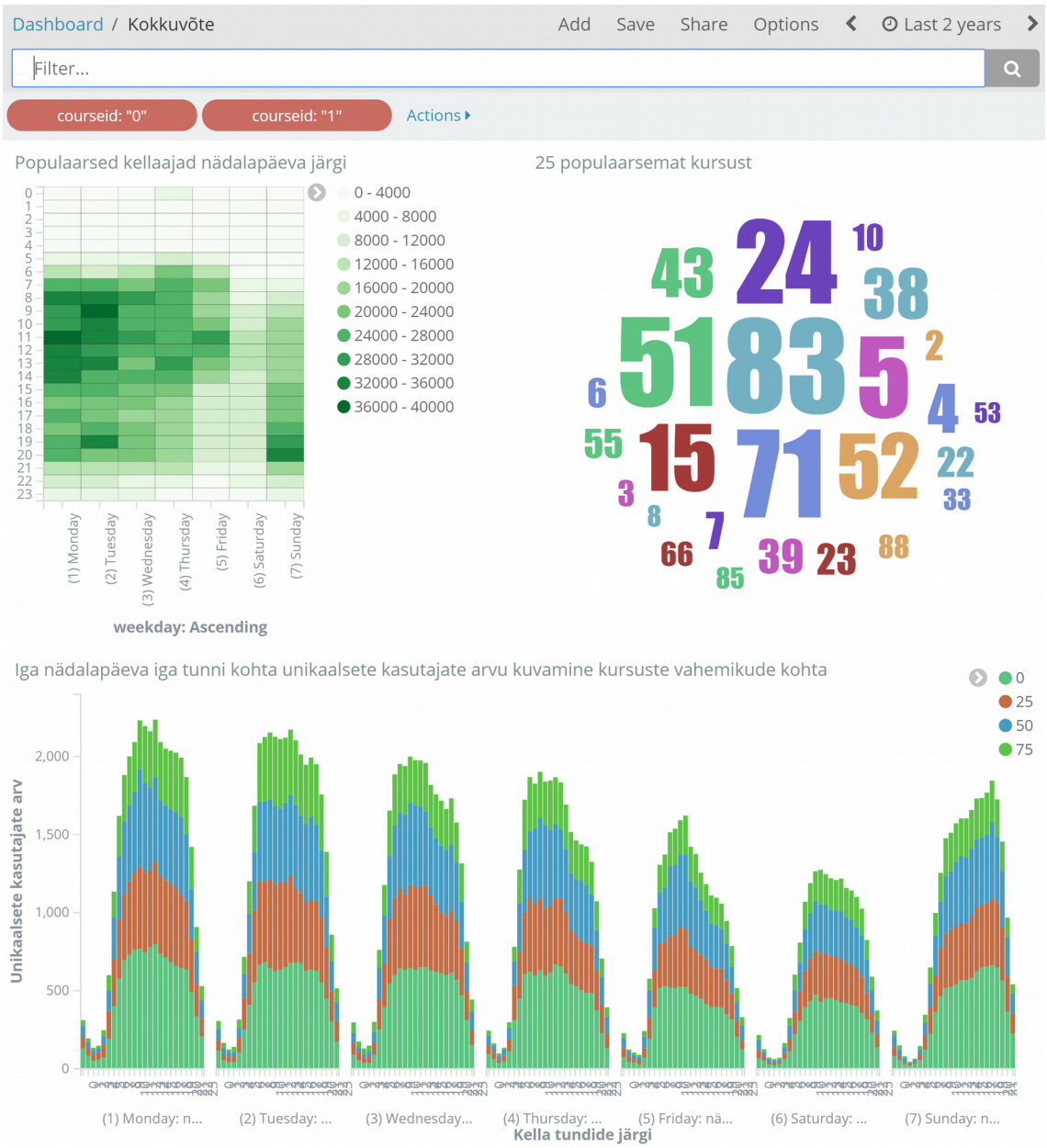
Logiandmetes ei ole esitatud kursuse unikaalse identifikaatori ja nime seost, seega õppekeskkonna ained.ttu.ee veebilehte uurides, leidis autor, et kursus unikaalse

identifikaatoriga 83 on aine ITX0040 Andmeturve ja kursus unikaalse identifikaatoriga 51 on aine IDK1011 Programmeerimise algkursus ning kursus unikaalse identifikaatoriga 71 on aine ITI0011 Programmeerimise põhikursus Javas.

Diagrammil nimega Iga nädalapäeva iga tunni kohta unikaalsete kasutajate arvu kuvamiseks kursuste vahemikude kohta on esitatud andmed kursuste vahemike jaoks ehk kursused on jaotatud nelja kategooria vahel kursuse unikaalse identifikaatori alusel, kus esimene vahemik on 0–24, teine vahemik on 25–49, kolmas vahemikuks on 50–74 ja 75–99. Esimesest vahemikust on filtreerimise teel eemaldatud kursused unikaalse identifikaatoriga 0 ja 1, kuna need ei ole õppeaine kursused.

Esimese kursuste vahemiku jaoks on pühapäev kõige populaarsem päev. Pühapäeval on külastab seda kursust kõige rohkem erinevaid kasutajaid, 269 erinevat kasutajat, kell 20. Selle alusel võib, järeldada, et nendes kursustes, mis asuvad antud vahemikus, on kodutöö tähtaeg määratud esmaspäevaks. Teise kursuste vahemiku jaoks on populaarseim kolmapäev, kus erinevate kasutajate hulk on kell 11 suurim, erinevaid kasutajaid 197. Kolmanda vahemiku jaoks on esmaspäev kõige populaarsem, kus kell 11 on kõige rohkem erinevaid kasutajaid (269). Neljanda vahemiku jaoks on populaarsemaks päevaks teisipäev, kus kell 19 on erinevaid kasutajaid kõige rohkem (200).

Järeldusena võib tuua, et erinevate kursuste vahemikes on populaarne päev erinev. Esimese ja kolmanda kursuste vahemikes on kõige rohkem erinevaid kasutajaid, mõlemas on 269 erinevat kasutajat. Teise ja neljanda kursuse vahemike jaoks on erinevate kasutajate arv peaaegu sama.



Joonis 23. *Dashboard*i näide kursuse unikaalse identifikaatorite filtritega.

6 Kokkuvõte

Käesoleva töö eesmärgiks oli TTÜ õppekeskkonna ained.ttu.ee logide visualiseerimine ja analüüsimine.

Logide analüüsimisel pidi saama ülevaate õppekeskkonna kasutamist nädala vältel ning ülevaade populaarsetest kursustest. Iga kursuse kohta taheti näha, millistel kellaaegadel ja päevadel käiakse kursust vaatamas ning taheti teada iga kursuse kohta unikaalsete kasutajate arvu erinevatel päevadel.

Töö tulemuseks on TTÜ serveris töötav veebipõhine logide analüüsimisvahend, mille abil analüüsiti TTÜ õppekeskkonna logisid. Logide analüüsimiseks on loodud vaade kolme diagrammiga, mille alusel on analüüsitud õppekeskkonna kasutamissagedus ja mille alusel saab analüüsida igat kursust eraldi või mitme kaupa.

Logide analüüsimisel leiti, et õppekeskkonda ained.ttu.ee kasutatakse kõige rohkem tööpäevadel ajavahemikus 7 – 20. Kõige suurem aktiivsus õppekeskkonnas on esmaspäeval kell 11 ja teisipäeval kell 9. Samuti on järeldatud, et kodutööde tähtajaks määratakse kõige sagedamini esmaspäeva ja kolmapäeva, kuna nendel päevadel on õhtuti suurem aktiivsus ajavahemikus 18 – 20.

Analüüsi tulemusel selgus, et kolm populaarsemat kursust on Andmeturve, Programmeerimise algkursus ja Programmeerimise põhikursus Javas.

Kasutatud kirjandus

[1] Log File. [WWW] https://www.techopedia.com/definition/5445/log-file (06.05.2017)

[2] Christensson, P. Log File Definition, TechTerms. [WWW] https://techterms.com/definition/logfile (06.05.2017)

[3] Vaarandi, R. (2005). Tools and Techniques for Event Log Analysis : doctoral thesis. Tallinn University of Technology, Tallinn.

[4] Parsons, T. 5 Ways to Use Log Data to Analyze System Performance. [WWW] https://blog.logentries.com/2014/06/5-ways-to-use-log-data-to-analyze-system-performance/ (06.05.2017)

[5] Parsons, T. How often should you look at your event and system logs? Daily, weekly, or just when there is a problem? [WWW] https://blog.logentries.com/2012/05/how-often-should-you-look-at-your-event-and-system-logs-daily-weekly-or-just-when-there-is-a-problem/ (06.05.2017)

[6] Data Visualization. What it is and why it matters. [WWW] https://www.sas.com/en_id/insights/big-data/data-visualization.html (16.05.2017)

[7] See Your Business Data in a Whole New Light. [WWW] https://www.oracle.com/solutions/business-analytics/data-visualization.html (16.05.2017)

[8] The Open Source Elastic Stack. [WWW] https://www.elastic.co/products (20.05.2017)

[9] What is Splunk? [WWW] https://www.splunk.com/ (16.05.2017)

[10] Overview. [WWW] https://www.graylog.org/features (16.05.2017)

[11] Why GoAccess? [WWW] https://goaccess.io/ (16.05.2017)

[12] WebKog Exparet. [WWW] https://www.weblogexpert.com/lite.htm (16.05.2017)

[13] What is AWStats. [WWW] http://www.awstats.org/ (16.05.2017)

[14] SAWMILL ANALYTICS - The Professional Solution. [WWW] http://www.sawmill.co.uk/(20.05.2017)

[15] Visual Log Parser. [WWW] https://visuallogparser.codeplex.com/ (16.05.2017)

[16] What is it? [WWW] http://www.webalizer.org/ (16.05.2017)

[17] Analyze your Web Server Data and be empowered with LogParser and Log Parser Lizard GUI. [WWW] http://www.hanselman.com/blog/AnalyzeYourWebServerDataAndBeEmpoweredWithLogParserAndLogParserLizardGUI.aspx (16.05.2017)

[18] Log analyzers Comparisons. [WWW] http://www.awstats.org/docs/awstats_compare.html (20.05.2017)

[19] Kili, A. 4 Good Open Source Log Monitoring and Management Tools for Linux. [WWW] https://www.tecmint.com/best-linux-log-monitoring-and-management-tools/ (20.05.2017)

[20] Marty, R. Visual Log Analysis – The Beauty. [WWW] https://www.slideshare.net/zrlram/visual-log-analysis-defcon-2006 (20.05.2017)

[21] The Complete Guide to the ELK Stack. [WWW] https://logz.io/learn/complete-guide-elk-stack/ (16.05.2017)

[22] Lightweight Data Shippers. [WWW] https://www.elastic.co/products/beats (16.05.2017)

[23] What is the difference between Logstash and Beats?. [WWW] https://www.elastic.co/guide/en/beats/filebeat/1.1/diff-logstash-beats.html (22.05.2017)

[24] Upgrading the Elastic Stack. [WWW] https://www.elastic.co/guide/en/elastic-stack/current/upgrading-elastic-stack.html (16.05.2017)

[25] Elasticsearch. [WWW] https://www.elastic.co/products/elasticsearch (16.05.2017)

[26] Vanderzyden, J. What is Elasticsearch, and How Can I Use It? [WWW] https://qbox.io/blog/what-is-elasticsearch (16.05.2017)

[27] Introducing the Query Language. [WWW] https://www.elastic.co/guide/en/elasticsearch/reference/current/_introducing_the_query_language.html (16.05.2017)

[28] Logstash Introduction. [WWW] https://www.elastic.co/guide/en/logstash/current/introduction.html (16.05.2017)

[29] Levy, T. 5 Logstash Pitfalls You Need to Avoid. [WWW] https://logz.io/blog/5-logstash-pitfalls-and-how-to-avoid-them/ (16.05.2017)

[30] How Logstash Works. [WWW] https://www.elastic.co/guide/en/logstash/master/pipeline.html (16.05.2017)

[31] Introduction. [WWW] https://www.elastic.co/guide/en/kibana/current/introduction.html (16.05.2017)

[32] Elasticsearch-dump. [WWW] https://github.com/taskrabbit/elasticsearch-dump (16.05.2017)

[33] nginx. [WWW] https://nginx.org/en/ (16.05.2017)

[34] Selvin, V. (2016) HOW TO SECURE ELASTICSEARCH AND KIBANA. [WWW] https://mapr.com/blog/how-secure-elasticsearch-and-kibana/ (16.05.2017)

[35] Beginner's Guide. [WWW] http://nginx.org/en/docs/beginners_guide.html (16.05.2017)

[36] Introduction. [WWW] https://www.elastic.co/guide/en/x-pack/current/xpack-introduction.html (18.05.2017)

[37] Getting Started. [WWW] https://www.elastic.co/guide/en/x-pack/current/ml-getting-started.html (18.05.2017)

[38] Subscriptions that Go to Work for You. [WWW] https://www.elastic.co/subscriptions (18.05.2017)

[39] Niglas, K. (2013) Andmete esmane töötlemine, analüüsimine ja esitamine : õppematerjal. Tallinna Ülikool, Tallinn.

[40] Moodle. [WWW] https://moodle.org/ (20.05.2017)

[41] Event 2. [WWW] https://docs.moodle.org/dev/Event_2 (06.05.2017)

[42] General developer forum. [WWW] https://moodle.org/mod/forum/discuss.php?d=138539 (21.05.2017)

[43] moodle. [WWW] https://github.com/moodle/moodle/blob/master/lib/accesslib.php (21.05.2017)

[44] Java Virtual Machine. [WWW] https://www.elastic.co/guide/en/elasticsearch/guide/master/_java_virtual_machine.html (20.05.2017)

[45] Anicas, M. (2015) How To Install Elasticsearch, Logstash, and Kibana (ELK Stack) on Ubuntu 14.04. [WWW] https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elk-stack-on-ubuntu-14-04 (02.04.2017)

[46] Install Elasticsearch with Debian Package. [WWW] https://www.elastic.co/guide/en/elasticsearch/reference/current/deb.html (02.04.2017)

[47] Install Kibana with Debian Package. [WWW] https://www.elastic.co/guide/en/kibana/current/deb.html (02.04.2017)

[48] Installing Logstash. [WWW] https://www.elastic.co/guide/en/logstash/current/installing-logstash.html (02.04.2017)

[49] Install npm on Ubuntu 16.04. [WWW] https://www.rosehosting.com/blog/install-npm-on-ubuntu-16-04/ (18.05.2017)

[50] Filter plugins. [WWW] https://www.elastic.co/guide/en/logstash/current/filter-plugins.html (20.05.2017)

[51] grok. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-filters-grok.html (20.05.2017)

[52] geoip. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-filters-geoip.html (20.05.2017)

[53] Basmayor, M. (2009) Setting Up MySQL/JDBC Driver on Ubuntu. [WWW] https://marksman.wordpress.com/2009/03/01/setting-up-mysqljdbc-driver-on-ubuntu/ (20.05.20117)

[54] Working with plugins. [WWW] https://www.elastic.co/guide/en/logstash/2.3/working-with-plugins.html#listing-plugins (15.05.2017)

[55] stdout. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-outputs-stdout.html (20.05.2017)

[56] json_lines. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-codecs-json_lines.html (20.05.2017)

[57] Mohan, V. (2016) Migrating MySql Data Into Elasticsearch Using Logstash. [WWW] https://qbox.io/blog/migrating-mysql-data-into-elasticsearch-using-logstash (15.05.2017)

[58] Natarajan, R. (2011) How to Run Cron Every 5 Minutes, Seconds, Hours, Days, Months. [WWW] http://www.thegeekstuff.com/2011/07/cron-every-5-minutes/ (15.05.2017)

[59] jdbc. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-inputs-jdbc.html (15.05.2017)

[60] jdbc_fetch_size. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-inputs-jdbc.html#plugins-inputs-jdbc-jdbc_fetch_size (20.05.2017)

[61] jdbc_paging_enabled. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-inputs-jdbc.html#plugins-inputs-jdbc-jdbc_paging_enabled (20.05.2017)

[62] Levy, T. (2015) INSERT INTO LOGSTASH SELECT DATA FROM DATABASE. [WWW] https://www.elastic.co/blog/logstash-jdbc-input-plugin (15.05.2017)

[63] replace. [WWW] https://www.elastic.co/guide/en/logstash/current/plugins-filters-mutate.html#plugins-filters-mutate-replace (15.05.2017)

[64] Diener, D. (2016) Running Bash Commands in the Background the Right Way [Linux]. [WWW] https://www.maketecheasier.com/run-bash-commands-background-linux/ (15.05.2017)

[65] Logstash Configuration Examples. [WWW] https://www.elastic.co/guide/en/logstash/current/config-examples.html (15.05.2017)

[66] Reloading the Config File. [WWW] https://www.elastic.co/guide/en/logstash/current/reloading-config.html (15.05.2017)

[67] Kibana 4 - Import and Export Visualizations and Dashboards with Elasticdump. [WWW] http://air.ghost.io/kibana-4-export-and-import-visualizations-and-dashboards/ (05.05.2017)

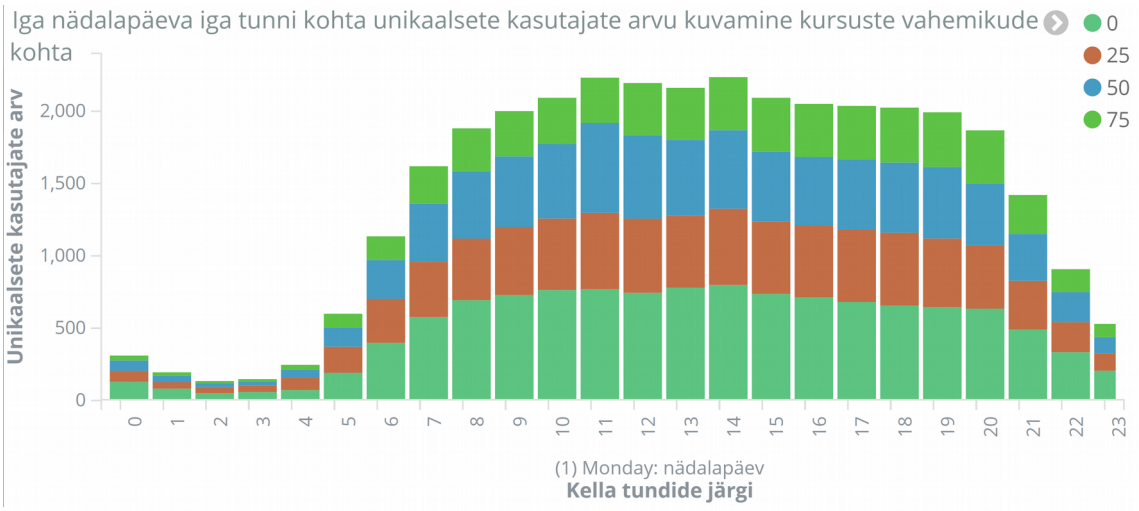
[68] Scripting. [WWW] https://www.elastic.co/guide/en/elasticsearch/reference/5.4/modules-scripting.html (16.05.2017)

Lisa 1 – Skriptitud välja *weekday* skripti kood

```
int x = doc['timecreated'].date.dayOfWeek;
if (x == 1)
    '(1) Monday';
else if (x == 2)
    '(2) Tuesday';
else if (x == 3)
    '(3) Wednesday';
else if (x == 4)
    '(4) Thursday';
else if (x == 5)
    '(5) Friday';
else if (x == 6)
    '(6) Saturday';
else if (x == 7)
    '(7) Sunday';
else
    'Unknown'
```

Joonis 24. Skriptitud välja *weekday* skripti sisu.

Lisa 2 – Diagramm iga nädalapäeva iga tunni kohta unikaalsete kasutajate arvu kuvamiseks kursuste vahemikude kohta – ühe päeva näide suurendatud kujul



Joonis 25. Diagramm iga nädalapäeva iga tunni kohta unikaalsete kasutajate arvu kuvamiseks kursuste vahemikude kohta filtreeritud esmaspäeva järgi.