jamk

Käyttäjän monitorointi AWS Cloud-Watch RUM-palvelun avulla

Mikko Hämäläinen

Opinnäytetyö, AMK
Toukokuu 2024
Tietojenkäsittelyn tutkinto-ohjelma (AMK)

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Hämäläinen, Mikko

Käyttäjän monitorointi AWS CloudWatch RUM-palvelun avulla

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2024, 57 sivua

Tietojenkäsittelyn tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Sovelluskehityksen kasvu on ollut räjähdysmäistä viime vuosina, erityisesti pilvipalvelupohjaisten sovellusten. Sen myötä on käyttäjäkokemuksen ymmärtäminen noussut entistä tärkeämpään rooliin. Käyttäjäkokemuksen ymmärtäminen on kriittisen tärkeää, jotta pystytään ymmärtämään tärkeimmät seikat sovellusta suunniteltaessa ja toteuttaessa. Tämän ymmärtäminen auttaa sovelluskehittäjiä luomaan entistä eheämpiä ja helppokäyttöisiä sovelluksia.

Tutkimuksen lähtökohtana oli ottaa selvää, miten CloudWatch RUM-palvelu otetaan käyttöön sovelluksessa ja millaista analytiikkaa CloudWatch RUM kerää sovelluksesta. Opinnäytetyön toimeksiantajana toimi Skillwell oy, joka on jyväskyläläinen ohjelmistokehitysyritys. Heidän tavoitteenansa oli selvittää millaista dataa CloudWatch RUM kerää ja miten se auttaisi parantamaan heidän toimintaansa sovelluskehityksen saralla.

Tutkimus on toteutettu laadullisena kehittämistutkimuksena. Tutkimuksen lähteinä on käytetty kotimaisia sekä ulkomaisia artikkeleita ja kirjallisuutta, jotka ovat alan ammattilaisten laatimia. Tutkimustuloksena saatiin selvitys, miten CloudWatch RUM konfiguroidaan ja otetaan osaksi sovellusta sekä millaista metriikkaa RUM kerää ja kuinka se havainnollistetaan. Tutkimusdata kerättiin Skillwellin websocket chat-sovelluksen testiversiosta.

Avainsanat (asiasanat)

AWS, käyttäjänmonitorointi, CloudWatch RUM, AWS Well-Architected, sovelluksen suorituskyky, business intelligence.

Muut tiedot (salassa pidettävät liitteet)

jamk

Description

Hämäläinen, Mikko

User monitoring with the AWS CloudWatch RUM

Jyväskylä: JAMK University of Applied Sciences, May 2024, 57 pages.

Degree Programme in Business Information Technology. Bachelor´s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

Application development has exploded in recent years, especially for cloud-based applications. As a result, understanding the user experience has become increasingly important. Understanding the user experience is critical to understanding the key considerations when designing and implementing an application. Understanding this will help application developers to create more cohesive and user-friendly applications.

The starting point for this research was to understand how the CloudWatch RUM service is deployed in an application and what analytics CloudWatch RUM collects from the application. The thesis was commissioned by Skillwell oy, a software development company based in Jyväskylä. Their goal was to find out what kind of data CloudWatch RUM collects and how it could help them improve their application development activities.

The research was conducted as a qualitative development study. The sources of the research were domestic and foreign articles and literature written by professionals in the field. The outcome of the research was an explanation of how CloudWatch RUM is configured and integrated into the application, what kind of metrics RUM collects and how it is visualized. The research data was collected from a test version of Skillwell's websocket chat application.

Keywords/tags (subjects)

AWS, user monitoring, CloudWatch RUM, AWS Well-Architected, application performance, business intelligence.

Miscellaneous (Confidential information)

Sisältö

Kuviot

Käsiteluettelo

AWS

Amazon.com:n kehittämä ja ylläpitämä pilvipalvelualusta

AWS CloudWatch

Pilvipalvelu, jonka avulla voidaan monitoroida pilven resursseja ja sovellusten suoritusta.

AWS CloudWatch RUM

Pilvipalvelu, jonka avulla avulla monitoroidaan sovelluksen suorituskykyä keräämällä käyttäjädataa käyttäjäistunnoista reaaliajassa.

AWS Well-Architected

Viitekehys, jossa kuvataan pilvipalveluiden keskeisimmät suunnitteluperiaatteet ja käytänteet miten optimoida pilvipalveluiden suorituskykyä.

APM

Käytänne, jonka avulla seurataan sovelluksen suorituskykyä käyttäen monitorointiohjelmistoja ja telemetriadataa.

RUM

Suorityskyvyn monitorointi prosessi, jonka avulla kerätään yksityiskohtaista dataa käyttäjän ja sovelluksen välisestä vuorovaikutuksesta

Pilvipalvelu

Verkon kautta tarjottu tietotekninen palvelu

1 Johdanto

Sovelluskehityksen kasvu on ollut räjähdysmäistä lähivuosina ja isossa nosteessa ovat sovellukset, jotka toimivat pilvipalveluiden päällä. Pilvipalvelut ovat paljon joustavimpia ratkaisuja kehittää ja ylläpitää sovelluksia kuin vaikkapa perinteiset omakustanteiset konesalit, joiden kautta palvelut pyörivät. Pilvipalvelut eivät kuitenkaan takaa automaattisesti täydellistä käyttäjäkokemusta ja optimaalisesti toimivaa sovellusta.

Pelkät pintapuoliset kyselyt käyttäjiltä eivät auta havaitsemaan sovelluksen ongelmia riittävän nopeasti vaan tarvitaan ratkaisu, jonka avulla voidaan seurata käyttäjien toimintaa ja tapahtumia reaaliajassa. Näiden palvelujen avulla saadaan parempi kuva mitkä tekijät vaikuttavat sovelluksen suoritukseen käyttäjätasolla. Tähän tarkoitukseen Amazon Web Services (AWS) on luonut AWS CloudWatch RUM-palvelun, joka auttaa keräämään ja monitoroimaan asiakaspuolen dataa sovelluksen suorituskyvystä käyttäjäistuntojen perusteella. Kerätty data sisältää muun muassa sivujen latausajat, asiakaspuolen virheilmoitukset sekä käyttäjien käyttäytymisen sovelluksessa. (Use CloudWatch RUM n.d)

Tämän opinnäytetyön tarkoituksena oli saada selville, miten AWS CloudWatch RUM palvelu otetaan käyttöön sovelluksessa ja millaista tietoa RUM:in avulla voidaan kerätä. Työn toimeksiantajana toimiva Skillwell halusi tietää miten RUM toimii ja kuinka he voivat käyttää sitä omissa sovelluksissaan ja sen kautta siirtää tätä osaamista palvellakseen asiakkaitaan paremmin. Tulosten tarkoituksena oli luoda Skillwellille parempi ymmärrys, kuinka he voivat tarjota asiakkailleen parempia palveluita ja sovellusten ylläpitoa RUM:in avulla.

Tutkimus ja sen tulokset on suunnattu henkilöille, joilla on opintojen, töiden tai mielenkiinnon kannalta tarve ymmärtää sovellusten monitoroinnista ja sovellusten kehittämisestä käyttäjäkokemuksen näkökulmasta. Aihe valikoitui mielenkiinnosta pilvipalveluita kohtaan sekä halusta saada syvempää ymmärrystä AWS:n tarjoamista palveluista työelämää varten. Tulokset avaavat lukijoille ymmärryksen miksi monitorointia kannattaa tehdä ja mitä kaikkea tietoa sen avulla voidaan saavuttaa. Tarkoituksena on herättää ajatuksia, miten käyttäjän toiminnalla on myös vaikutuksia sovelluksen toimintaan sekä antaa ymmärrys minkä takia sovelluksia monitoroidaan käyttäjäläheisesti.

2 Tutkimusasetelma

Tässä kappaleessa käydään läpi tutkimuksen tavoitteet ja rajaus sekä perehdytään tutkimuksen eri osa-alueisiin. Kappaleessa käydään läpi opinnäytetyön toimeksiantaja, tutkimuskysymykset ja perehdytään työn tutkimusmenetelmiin.

2.1 Toimeksiantaja

Opinnäytetyön toimeksiantajana toimii Skillwell Oy. Skillwell on vuonna 2018 perustettu jyväskyläläinen ohjelmistokehitysyritys, joka tarjoaa asiakkaillensa erilaisia digitaalisia ratkaisuja, pilvipalveluosaamista sekä erilaisia integraatioratkaisuja. Skillwell on AWS:n sertifioitu kumppani, jonka ansiosta Skillwellin ratkaisut pohjautuvat AWS-pilvipalveluratkaisuihin. (Skillwell n.d.) Vuonna 2021 Skillwellin liikevaihto oli 0,9 miljoonaa euroa ja seuraavana vuonna 2022 se oli 1,1 miljoonaa euroa (Finder 2022).

2.2 Tutkimuksen tavoitteet ja aiheen rajaus

Tutkimuksen tavoitteena on selvittää millaista analytiikkaa AWS CloudWatch RUM-palvelulla pystytään keräämään. Työssä käytettävä data tulee Skillwellin omasta Websocket-testisovelluksesta. Datan sisältö ei ole kuitenkaan tärkeää tutkimuksessa vaan huomata millaista analytiikkaa CloudWatch RUM pystyy datan avulla muodostamaan. Käyttäjäkokemuksen ymmärtäminen on elintärkeää sovelluskehityksessä, sillä sen avulla voidaan kehittää käyttäjäystävällisemmäksi. CloudWatch RUM:in keräämällä analytiikalla voidaan tarkastella miten käyttäjän käyttäytyminen vaikuttaa sovelluksen suoritukseen eri tilastoina sekä taulukko- ja kaaviomuodoissa.

Toimeksiantajan näkökulmasta tutkimuksessa on tarkoitus selvittää mitä tietoja Skillwell voi CloudWatch RUM:n avulla kerätä omissa sovelluksissaan. Yrityskeskeisestä näkökulmasta on tärkeä huomioida palvelun antamat hyödyt yritykselle. Tässä tapauksessa miten analytiikan avulla voidaan tehdä ja miten käyttäjien toiminta vaikuttaa sivun käyttäjäkokemukseen.

Aihe on ajankohtainen, sillä pilvipalveluiden käyttöönoton kasvu on lähivuosina ollut kovassa nousussa. Myös käyttäjien kokemuksen ymmärtäminen ei tule koskaan menettämään tärkeyttään sovelluskehityksen saralla. Mitä paremmin käyttäjän kokemusta ymmärretään, sen helpompi on luoda laadukkaita sovelluksia, joissa käyttäjäkokemus on laadukasta.

2.3 Tutkimuskysymykset

Tutkimus pohjautuu CloudWatch RUM-ympäristöön ja sinne saatu data tulee Skillwellin omasta testisovelluksesta. RUM käsitteenä käydään teoriassa läpi ja tutkimuksessa tarkastellaan CloudWatch RUM-palvelun ympäristöä, millaista analytiikkaa sinne kerääntyy ja miten käyttöönotto tapahtuu. Tutkimuskysymykset ovat seuraavanlaiset:

- Mikä on RUM?
- Miten CloudWatch RUM otetaan käyttöön sovelluksessa?
- Mitä tietoa/analytiikkaa CloudWatch RUM:lla voidaan kerätä?

2.4 Tutkimusmenetelmä

Toimeksiantajan ongelmiin nähden ja tutkimuskysymysten pohjalta muodostui opinnäytetyön tutkimusmenetelmäksi laadullinen kehittämistutkimus. Laadullisella tutkimuksella pyritään ymmärtämään tutkittavaa ilmiötä ja saada vastauksia kysymykseen: ”Mistä on kyse?”. Laadullisessa tutkimuksessa aineisto kerätään havaintojen ja haastattelujen avulla, jotta ilmiöstä saadaan ymmärrys. (Kananen 2015, 34.) Tutkittavaa palvelua halutaan ymmärtää syvällisemmin ja miten siitä voidaan saada paras hyöty irti (Bister 2019, 44). Kanasen (2017, 36) mukaan laadullisella tutkimuksella halutaan ymmärtää ilmiötä uudella tavalla. Tarkoituksena on saada yhdestä tutkimusyksiköstä mahdollisemman paljon irti eli tapaukseen pureudutaan syvällisesti pelkän pintaraapaisun sijaan. (Kananen 2017, 36.)

Laadullisen tutkimuksen tutkimusprosessi on joustava, koska ilmiötä ei tunneta tarkkaan. Laadullisessa tutkimuksessa aineisto ohjaa tutkimusta, sillä etukäteen ei voida tietää tarvittavaa aineistomäärää. Tämä johtuu aineistosta tehtävien tulkinnoista ja tuloksista, sillä ne määrittävät onko lisäaineiston keruulle tarvetta. Hyvänä esimerkkinä voidaan verrata tilanteeseen, jossa opiskelija on

tullut takaisin ulkomaan vaihdosta. Opiskelijalta kysytään kysymyksiä vaihdossa olleesta ajasta, joihin opiskelija vastaa omien kokemustensa perusteella. Kyselijät kysyvät vastauksiin tarkentavia kysymyksiä, jonka pohjalta yleisölle rakentuu mielikuva, millaista on opiskella yliopistossa ulkomailla. (Kananen 2017, 35.)

2.5 Tiedonhaku

Tutkimuksen tietoperusta on laadittu siten, että lukijan on helppo ymmärtää tutkimuksessa käsiteltäviä palveluita, termejä ja teknologioita. Tietoperustassa kuvataan pilvipalveluiden monitorintiin liittyviä palveluita ja käsitteitä. Tietoperustassa tuodaan esille palveluiden ja käsitteiden hyvät ja huonot puolet sekä niihin liittyviä eroavaisuuksia. Lähteet kerättiin kolmenlaisista lähteistä, joita olivat alan ammattilaisten verkossa julkaistut artikkelit, kirjallisuus ja AWS:n sekä muiden ammattilaisten kehittäjädokumentaatiot. Lähteet on valittu luotettavuuden ja kirjoittajien ammattitaidon perusteella.

Alan ammattilaisten laatimien artikkeleiden avulla tietoperustaa saatiin syvennettyä. Artikkelit toivat erilaisia näkökulmia ja laajensivat tietoperustassa käytäviä asioita, jotka loivat laajempaa ymmärrystä dokumentaatioista saatuun tietoon. Artikkelit valittiin huolellisesti ottaen huomioon kirjoittaja, julkaisuja ja julkaisuajankohta, jotta artikkelin laatu nähtiin tarpeeksi luotettavaksi. Tämä vaati kriittistä havainnointia, sillä artikkeleita on monia ja niiden laatu vaihteleva.

Kirjallisuuden kohdalla luotettavuuden menettelytapa toimi samoilla kriteereillä kuten artikkelien luotettavuuden arvioinnissa, tarjoten syvällisen katsauksen aiheeseen. Luotettavuudessa keskityttiin erityisesti julkaisuajankohtaan, sillä IT-alan kehitys on nopeaa ja kirjallisuuden julkaisutahti on netissä oleviin artikkeleihin verrattuna huomattavasti hitaampaa. Kirjallisuudessa olevalla tiedolla on täten riski vanhentua nopeammin.

Dokumentaatiot olivat tärkeä tiedonlähde, sillä niissä kehittäjät itse kertoivat teknologioista, joista saatiin tutkimukselle relevanttia tietoa. Dokumentaatiot tarjosivat tärkeimmät ja ajankohtaisimmat tiedot teknologiasta. Dokumentaatioita käytettäessä otettiin kuitenkin huomioon puolueellisuus, joita kehittäjät ovat voineet vahingossa tai tahallisesti tuoda esille, sillä kyseessä on kehittäjien itse laatima tuote.

3 Pilvipalvelut

Tässä kappaleessa käsitellään pilvipalveluiden määritelmää, mitkä ovat pilvipalveluiden keskeiset ominaispiirteet ja minkälaisia pilvipalveluntarjoajia löytyy.

3.1 Mitä ovat pilvipalvelut?

Pilvipalvelu on malli, jolla mahdollisestaan kaikkialle ulottuva, helposti käytettävä, on-demand-pohjalla oleva kokoelma erilaisia laskentaresursseja (esim. verkkoja, palvelimia ja tallennustilaa) joita voidaan ottaa käyttöön nopeasti, vaivattomasti ja ilman palveluntarjoajan vuorovaikutusta asiakkaaseen. (Wittig & Wittig 2023, 4.)

Mell ja Grance (2011) kertovat pilvipalveluiden viisi keskeistä ominaisuutta:

- **On-demand itsepalvelu**

  Kuluttaja voi hankkia erilaisia palveluja, kuten palvelin aikaa tai verkkotallennustilaa, tarpeensa mukaan ilman minkäänlaista ihmisten välistä vuorovaikutusta palveluntarjoajien kanssa. (Mell & Grance 2011, 2).

- **Laaja verkkoyhteys**

  Valmius saada palveluita käyttöön verkon kautta standardisoitujen mekanismien avulla, joiden ansiosta asiakaspuolen laitteiden kuten kannettavien tietokoneiden, mobiilipuhelimien ja tablettien avulla on helpompi ottaa pilvipalvelut käyttöön (Mell & Grance 2011, 2).

- **Resurssien yhdistely**

  Palveluntarjoajan resurssit yhdistetään palvelemaan useampia kuluttajia käyttämällä monivuokralaismallia, missä palveluntarjoajan fyysiset ja digitaaliset resurssit jaetaan dynaamisesti eri käyttäjien vaatimusten mukaan. Asiakkaalla ei ole tietoa resurssien tarkasta sijainnista, mutta hän voi määritellä sijainnin korkeammalla tasolla (esim. maa, osavaltio tai datakeskus). Esimerkkejä resursseista ovat tallennustilat, prosessointi, muisti ja verkon kaistanleveys. (Mell & Grance 2011, 2.)

- **Nopea elastisuus**

  Kapasiteettia voidaan eri tapauksissa tarjota ja vapauttaa elastisesti, joissakin tapauksissa automaattisesti, jotta niiden skaalautuvuus ulos- ja sisäänpäin vastaa kysynnän tuottamaan kuormaan (Mell & Grance 2011, 2).

- **Mitattava palvelu**

  Pilvipalvelujärjestelmät kontrolloivat ja optimoivat resurssin käyttöä hyödyntämällä erilaisia mittaustapoja, jotka ovat eri palveluille omanlaisensa. Resurssien käyttöä voidaan monitoroida, kontrolloida ja raportoida mikä luo läpinäkyvyyttä asiakakan ja palveluntarjoajan välille. (Mell & Grance 2011, 2.)

3.2 Pilvipalvelutarjoajat

AWS

AWS (Amazon Web Services) on vuonna 2006 Amazonin lanseeraama kattava ja alati kehittyvä pilvipalvelualusta ja pilvipalveluiden markkinajohtaja. AWS koostuu erilaisista palvelumalleista, joita ovat infrastruktuuripalvelut (IaaS), alustapalvelut (PaaS) ja ohjelmistopalvelupaketit (SaaS). AWS tarjoaa erilaisia työkaluja, kuten laskentatehoa, tietokantatallennusta ja sisällönjakelupalveluja. AWS oli yksi ensimmäisistä yrityksistä, joka otti käyttöön pay-as-you-go-pilvipalvelumallin, joka skaalautuu tarjoamaan käyttäjille eri palveluita asiakkaan tarpeiden mukaan. (Kirvan, Barney & Gillis 2024)

AWS Well-Architected

AWS Well-Architected on ohjeistus, jonka avulla pilviarkkitehdit voivat rakentaa turvallisen, suorituskykyisen, joustavan ja tehokkaan infrastruktuurin erilaisille sovelluksille ja suoritusmäärille. AWS Well-Architected koostuu kuudesta pilarista: toiminnallinen huippuosaaminen (engl. operati-

onal excellence), turvallisuus (engl. security), luotettavuus (engl. reliability), suorituskyvyn tehokkuus (engl. performance efficiency), kustannusoptimointi (engl. cost optimization) ja kestävyys (engl. sustainability). (AWS Well-Architected n.d.)

Hyvin rakennetut sovellukset ja ympäristöt kasvattavat huomattavasti liiketoiminnan menestyksen todennäköisyyksiä. Näistä syistä AWS kehitti Well-Architected Frameworkin, jonka kuuden pilarin avulla pystytään vertailemaan pilviarkkitehtuurien hyviä ja huonoja puolia kun arkkitehtuuria aletaan kehittämään ja valitsemaan siihen tulevia palasia AWS:n ympäristössä. (Bonso 2023.)

AWS:n omilla verkkosivuilla Well-Architected kuusi pilaria määritellään seuraavasti:

- **Toiminnallinen huippuosaaminen**

  Toiminnallisen huippuosaamisen pilari keskittyy hallinnointiin ja palveluiden monitorointiin AWS:n palveluissa sekä prosessien ja proseduurien kehittämiseen johdonmukaisesti. Tärkeimmät aiheet mitä tämä pilari koskettaa ovat muutosten automatisointi, kutsuihin vastaaminen ja standardien määrittäminen päivittäisille toiminnoille, jotta niiden hallinnointi olisi helpompaa. (AWS Well-Architected n.d.)

- **Turvallisuus**

  Turvallisuuden pilari keskittyy systeemien ja tiedon turvaamiseen. Isoimpina tekijöinä turvallisuuden pilarissa toimii luottamuksellisuus sekä että data pysyy eheänä, käyttäjien lupien hallinta eri AWS:n palveluissa ja valvontatoimien pystyttämisen, jotta voidaan havaita erilaiset turvallisuuteen liittyvät tapahtumat järjestelmissä. (AWS Well-Architected n.d.)

- **Luotettavuus**

  Luotettavuuden pilari keskittyy varmistamaan, että kaikki suorituskuormat suorittavat niille osoitetut työtehtävät ja kuinka palautua tilanteesta, jossa on epäonnistuttu vastaamaan vaadittuun kysyntään. Tärkeimpinä tekijöinä luotettavuuden pilarissa toimivat hajautetusti suunniteltu systeemi, suunnitelma miten palaudutaan erilaisista ongelmista ja vaihtuviin muutoksiin mukautuminen. (AWS Well-Architected n.d.)

- **Suorituskyvyn tehokkuus**

Suorituskyvyn tehokkuuden pilari keskittyy tietotekniikan- ja laskennanresurssien jäsentelyyn, optimointiin sekä niiden järkevään jakamiseen ympäristössä. Tärkeimpinä tekijöinä toimivat resurssien tyypin ja koon järkevä optimointi tehtävän kuormaan nähden, tehokkuuden monitorointi sekä tehokkuuden ylläpitäminen, kun liiketoiminalliset tarpeet muuttuvat. (AWS Well-Architected n.d.)

- **Kustannusoptimointi**

Kustannusoptimoinnin pilarin tärkeimpänä tehtävänä on turhien kulujen välttäminen. Tärkeimpinä tekijöinä toimivat rahallisen kulutuksen ymmärtäminen ajan myötä ja varojen jakamiseen liittyvä valvonta, resurssien oikean tyypin ja määrän valinta, sillä samassa palvelussa eri vaihtoehdot maksavat eri määriä ja palvelujen skaalaus ilman että tarvitsee kuluttaa yli varojen, jotta päästään liiketaloudellisiin tavoitteisiin. (AWS Well-Architected n.d)

- **Kestävyys**

Kestävyyden pilari keskittyy pilvipalveluiden käytöstä syntyneisiin ympäristöllisten vaikutusten minimoimiseen. Tärkeimpinä tekijöinä toimivat jaettu visio kestävyyden mallista, ymmärrys pilvipalveluiden vaikutuksesta ympäristöön ja tarvittavien resurssien käytön täyden potentiaalin hyödyntäminen, jotta loppupään vaikutukset vähenisivät. (AWS Well-Architected n.d)

Kuvio 1. AWS:n Well-Architected Frameworkin muodostavat pilarit (AWS Well-Architected Framework — 6 Pillars Explained 2023, muokattu)

4 Pilvipalveluiden monitorointi

Tässä kappaleessa käsitellään mitä on sovelluksen suorituskyvyn seuranta, miksi sitä tehdään ja mitä se tuo. Kappaleessa syvennytään AWS:n omaan sovelluksen suorituskyvyn seurannan palveluihin ja kuinka ne esiintyvät AWS:n Well-Architected frameworkin pilareissa ja kuinka ne palvelevat tätä ohjekehystä.

4.1 Sovelluksen suorituskyvyn seuranta

Sovelluksen suorituskyvyn seuranta (engl. application performance monitoring lyh. APM) on kokoelma erilaisia työkaluja ja prosesseja, joiden avulla eri IT-ammattilaiset pystyvät varmistamaan, että sovellukset täyttävät työntekijöiden, yhteistyökumppaneiden ja asiakkaan vaatimat kriteerit suorituskyvyn, luotettavuuden sekä hyvän käyttäjäkokemuksen osalta. (Brush, Lockhart & Demaitre 2022.) Sovelluksen suorituskyvyn seuranta antaa mahdollisuuden saada seurata eri mittareilla sovelluksen suorituskykyä verkkopalveluiden, jonojen ja tietokantojen pyyntöjen, virheiden ja viiveiden seuraamiseksi. (APM n.d.)

Sovellusten suorituskyvyn seuranta kuuluu yleisemmän, toisiinsa liittyvän termin sovellusten suorituskyvyn hallinnan (engl. application performance management) alle. Sovelluksen suorituskyvyn seuranta keskittyy ainoastaan suorituskyvyn valvomiseen, kun sovelluksen suorituskyvyn monitorointi keskittyy laajemmin sovelluksen suorituskykytasojen hallintaan. Lyhyesti sanottuna monitorointi on osa hallinnointia. (Brush ym. 2022.)

Mitä APM:ssa (engl. application performance monitoring) mitataan?

Kasvava digitalisaatio on kiihdyttänyt sovelluskehitystä räjähdysmäisesti, mutta samalla huomioon otettava huomioon sovellusten saatavuus käyttäjille, jossa suurina tekijöinä ovat sovellusten suorituskyky ja niiden optimointi. Tehokas APM auttaa sovelluskehittäjiä pitämään sovellusten suorituskyvyn korkealla tasolla sekä pitämään käyttäjät tyytyväisinä (What is APM? n.d). On tärkeää ymmärtää mitä tietoa APM:n avulla halutaan saada selville. APM avulla pystytään monitoroimaan sovelluksen ympäristöä eri toimijoilla. Näiden toimijoiden monitorointi tyyppejä ovat muun muassa:

- **Digitaalisen kokemuksen seuranta (engl. Digital experience monitoring)**

  Digitaalisen kokemuksen seurannan avulla kerätään suorituskykyyn liittyviä metriikoita, kuten latausaikoja, vastausaikoja ja seisonta-aikaa käyttäjän laitteen käyttöliittymästä. Digitaalisen kokemuksen seuranta tukee myös reaalikäyttäjän seurantaa (engl. real user monitoring lyh. RUM) mikä seuraa oikean käyttäjän toimintaa sovelluksessa sekä miten käyttäjän toiminta vaikuttaa sovellukseen.

- **Sovelluksen seuranta (engl. Application monitoring)**

  Sovelluksen seuranta kattaa koko sovelluskokonaisuuden tarkkailun, johon kuuluvat muun muassa sovelluskehys (ohjelmointikieli esim. Java), käyttöjärjestelmä, tietokanta, API:t, väliohjelmistot (engl. middleware), web-sovelluspalvelin ja käyttöliittymä. Sovelluskokonaisuuden monitorointi auttaa löytämään koodista syntyvät suorituskyvynpullonkaulat, sillä se sisältää kooditason jäljityksen. (What is application performance management? n.d.)

- **Tietokannan seuranta (engl. Database monitoring)**

  Tietokannan seurannan avulla tarkastellaan tietokannan kyselyitä tai proseduureja sovelluksen seurannan lisäksi (What is application performance management? n.d).

- **Saatavuuden seuranta (engl. Availability monitoring)**

  Saatavuuden seuranta monitoroi sovelluksen varsinaista saatavuutta ja laitteiston suoritusta, sillä vaikka sovelluksella ei olisi sillä hetkellä käyttäjiä, sillä pystytään mittaamaan, miten laitteisto kuormittuu sovelluksesta. (What is application performance management? n.d.)

Tiivistettynä APM:n avulla kerätään pääasiassa kolmea eri datakategoriaa: mittarit (metrics), jäljet (traces) ja lokitiedostot (log files). Mittareiden avulla pyritään ymmärtämään eri tapahtumien tila, joiden avulla pystytään seuramaan sovelluksen tilan muutoksia ja havaitsemaan ilmeneviä ongelmia. Jälkien avulla pystyttään käymään jokainen askel läpi sovellukseen kohdistuvissa kutsuissa.

Jäljet auttavat havainnollistamaan kutsun koko matkan sovelluksen tietoverkon lävitse, jonka ansiosta pystytään havaitsemaan laajan kokoelman ongelmia sovelluksessa, kuten vaikka turvallisuuteen liittyviä uhkia. Lokitiedostot luodaan automaattisesti joko sovelluksen puolesta tai käyttöjärjestelmän puolesta. Lokitiedostot pitävät sisällään tietoja käyttäjän käyttäytymisestä ja kaikista tapahtumista mitä sovelluksessa on tapahtunut. Niiden avulla on helppo pureutua juurisyihin, esimerkiksi selvittää miksi jokin asia tapahtui sovelluksessa tai miksi jonkun mittarin arvo on epänormaalilla tasolla verrattuna normaaliin. (Brush ym. 2022.)

Suoritusdatan keräämisen lisäksi nämä toimijat myös luovat käyttäjistä tapahtumaprofiilin, joka luodaan keräämällä ja jäljittämällä jokainen tapahtuma mitä käyttäjä on tehnyt omassa käyttöliittymässään (What is application performance management? n.d.) Hyvin toteutettu APM auttaa pitämään sovellukset suorituskykyisinä ja saatavina mikä suoraan heijastuu käyttäjien tyytyväisyyteen. Hyvällä monitoroinnilla saadaan pidettyä kustannukset kurissa, koska ongelmien ratkominen ja havainnointi helpottuu vähentäen sovelluksen mahdollisia seisonta-aikoja ja kehitystyö suoraviivaistuu, kun ongelmiin pystyttään reagoimaan jo kehitysvaiheessa. (What is APM? n.d).

Kuvio 2. Monitoroinnin osa-alueet mitä APM koskettaa (Jaffe 2020, muokattu)

Observability ja APM

Tarkkailtavuus (engl. observability) ja APM termeinä käytetään usein paljon saman asian kuvaamiseen, mutta näiden kahden termin välillä on kuitenkin eroavaisuuksia. Brush ja muut määrittelevät nämä kaksi termiä seuraavasti; APM on datan keräämistä, jonka pohjalta voidaan seurata suoritusta ja havaita ongelmia. Tarkkailtavuus käsitteenä määritellään lähes identtisesti pintatasolla, mutta tarkkailtavuudella pystytään tarkastelemaan sovelluksen tilaa sen järjestelmästä saadulla tiedolla. Pienemmän tason ohjelmistoissa ja järjestelmissä nämä termit ovat kuitenkin täysin erottamattomia toisistaan, johtuen käytössä olevista rajallisista resursseista ja kokonaisuuden pienestä koosta. (Brush ym. 2022.)

O'Donnellin (n.d) mukaan tarkkailtavuuden ja APM:n valinnan välillä kannattaa ottaa huomioon sovelluksen monimutkaisuus sekä muut mahdolliset haasteet ja tarpeet. Tarkkailtavuutta kannattaa harkita silloin kun kyseessä on monimutkainen kokonaisuus, jonka kokonaisuutta pitää ymmärtää suorituskykyä pidemmälle. Tärkeää on myös ymmärtää, miten kokonaisuus toimii reaaliajassa sekä olla valmis tutkimaan asiaa pidemmälle.

APM:n kannattaa harkita silloin kun halutaan keskittyä enemmän sovelluksen suorituskyvyn optimointiin, tunnistaa ja ratkaista kooditason ongelmia sekä halutaan löytää räätälöity ratkaisu käyttäjän kokemusta ja suorituskyvyn seurantaa varten. (O'Donnell n.d.)

Tiivistäen voidaan sanoa, että kun kyseessä on monimutkainen ratkaisu, jota tarvitsee ymmärtää syvällisesti kannattaa harkita tarkkailtavuutta. APM:ää kannattaa harkita, kun halutaan keskittyä sovelluksen suorituskyvyllisiin asioihin sekä koodi tason ongelmiin ja ratkaisuihin, jotka vaikuttavat suorituskykyyn.

4.2 RUM

Colmenares (2024) käsittelee artikkelissaan Amazon CloudWatch RUM-palvelua ja sen käyttöönottoon liittyviä seikkoja sekä pureutumalla syvemmälle RUM (real user monitoring) käsitteeseen. Nykyään datan keruu sovelluksista on astumassa entistä suurempaan rooliin, samalla myös datan ymmärtämisen painoarvo kasvaa. Datan ymmärtäminen auttaa sovelluskehittäjiä huomaamaan

erilaiset kehityskohdat sovelluksessa, sekä luomaan parempaa ymmärrystä, kuinka käyttäjän toiminta vaikuttaa sovelluksen toimintaan. (Colmenares 2024.)

Dataa voidaan koota eri RUM ratkaisuilla kuten esimerkiksi CloudWatch RUM:lla, jotka auttavat sovelluskehittäjiä luomaan parempia ratkaisuja (Colmenares 2024; Hawes 2023). CloudWatch RUM:in avulla voidaan valvoa ja kerätä loppukäyttäjien toiminnan pohjalta valvottuja metriikoita lähes reaaliajassa. Kerätty data voi olla verkkosivun lataamisajat tai käyttäjän käyttäytyminen sivulla. (Colmenares 2024.) Hawes (2023) mukaan RUM-ratkaisut auttavat keräämään dataa jokaisesta istunnosta, jonka avulla voidaan hahmottaa käyttäytymismalleja ja niiden pohjalta tehdä parannuksia sovellukseen. Käyttäjän istunnolla on paljon vaikutusta sovelluksen suoritukseen. Istunnot poikkeavat toisistaan valtavasti, joku käyttäjä voi kokeilla sivulla kaikkia mahdollisia toimintoja, kun taas toinen käyttäjä voi tehdä sivulla vaan tarvittavat toiminnot. (Hawes 2023.) Tämä data on erittäin arvokasta sovelluskehittäjille, sillä tämä data auttaa kehittäjiä hahmottamaan kehityskohdat, joiden korjaamisen myötä auttaa käyttäjiä saamaan paremman käyttökokemuksen (Colmenares 2024). Yhteenvetona voidaan todeta, että RUM auttaa kehittäjiä tunnistamaan sovelluksen kehityskohteita, vahvistamaan käyttäjäkokemusta sekä luomaan laadukkaasti toimiva sovellus. Samalla on tärkeää myös sovelluskehittäjien ymmärtää miten kerättyä dataa pitää tulkita

RUM:in vahvuudet

RUM:in vahvuuksiin kuuluu datan analysointi ja kuinka tämä data auttaa näyttämään tarkasti mitä kehitettävää sovelluksessa on (Knupfer 2024). RUM mahdollistaa sovelluskehittäjiä havaitsemaan datan ongelmat sovelluksessa paljon ennen kuin ne ilmenevät käyttäjällä. Tämä auttaa kehittäjiä luomaan parempia ratkaisuja koodi- että käyttöliittymätasolla. Näiden huomioiden avulla pystytään ymmärtämään käyttäjän realistista kokemusta sovelluksessa. (Hawes 2023.)

Knupferin (2024) mukaan RUM auttaa sovelluskehittäjiä sekä yhtiöitä optimoimaan myös omaa toimintaansa liiketalouden kannalta. RUM auttaa optimoimaan käytettyjä resursseja, kuten miten koodin rakenne vaikuttaa sovellukseen ja miten sitä voitaisiin parantaa tai miten käytössä olevien palvelimien määrä ja teho heijastuu tarvittuun kapasiteettiin. RUM auttaa myös suoraviivaista-

maan päätöksen tekoa datan avulla mitä RUM:in kautta saadaan. Näin saadaan hallittua kustannuksia sekä hahmottamaan mitä palveluita ja ratkaisuja voidaan optimoida paremmiksi, ettei makseta enemmän kuin tarvitsisi. (Knupfer 2024.)

RUM:in heikkoudet

Parhaissa RUM-ratkaisuissa kaikki käyttäjien istunnoissa tehdyt toiminnot tallennetaan, että saadaan laaja kuva käyttäjien saamasta kokemuksesta sovelluksessa ja sen pohjalta tarkempi kuva kehityskohteista. Tästä huolimatta kaikki RUM-ratkaisut eivät pysty tätä tarjoamaan. Korkeasti skaalautuvilla ratkaisuilla on mahdollisuus kerätä kaikki käyttäjädata, joilla voidaan muodostaa laaja kokonaiskuva käyttäjäkokemuksesta, kun taas vähemmän skaalautuvat ratkaisut joutuvat tekemään kokoelman vain sillä tietyllä määrällä dataa, jota ne voivat maksimissaan kerätä. (Hawes 2023.)

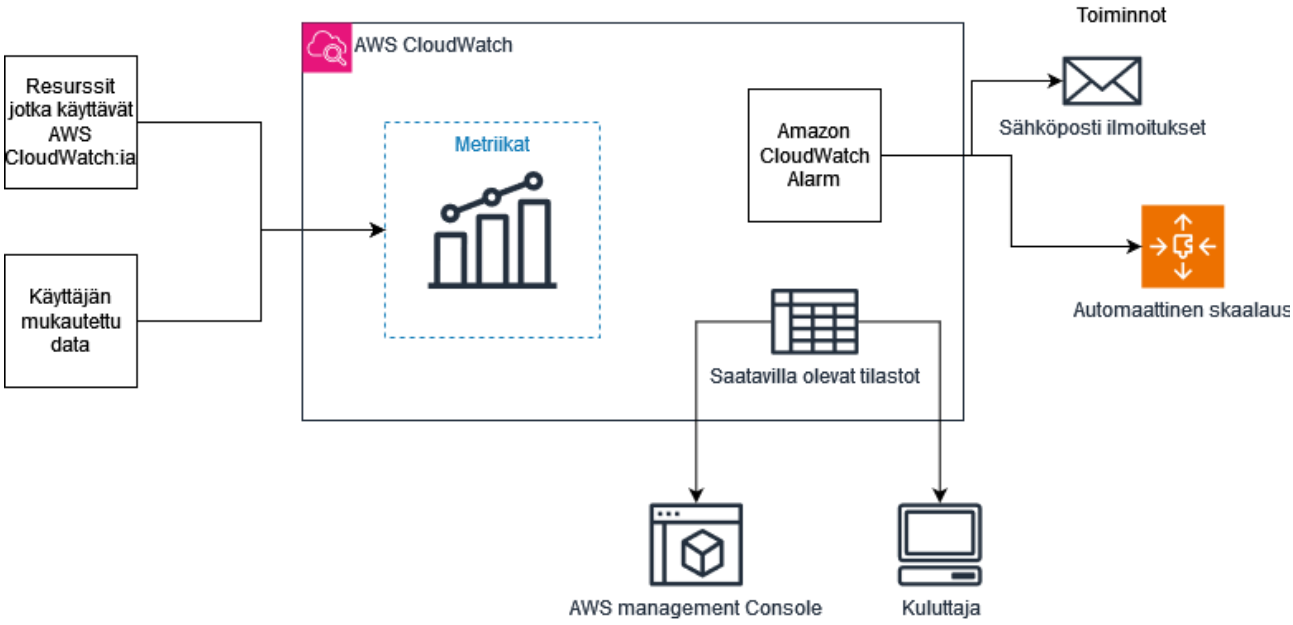
Kerner (2022) ja Stackifyn julkaisu (2023) kertovat, vaikka RUM-palveluilla pystytään saavuttamaan paljon, löytyy niistä silti heikkouksia, kuten että RUM on sovelluksen esituotantovaiheessa lähes hyödytön, sillä sovelluksella on silloin harvemmin käyttäjäkuormaa (Kerner 2022; What Is Real User Monitoring? How It Works, Examples, Best Practices, and More 2023). RUM ei myöskään kerro mitään palvelinpuolen suorituksesta. RUM on tarkoitettu seuraamaan käyttäjäpuolen liikennettä ja kuormaa, mihin palvelimien suorituskyky ei kuulu laisinkaan. RUM tarvitsee myös käyttäjäliikennettä toimiakseen. Esimerkiksi sivustot, jotka ovat testausvaiheessa, jolloin sivuston elementtejä testataan yksittäisillä käyttäjillä eivät ole järkeviä kohteita käyttää RUM:ia, sillä sivustojen liikenne ja kuorma ei antaisi realistista kuvaa oikeasta käyttäjäkuormasta, täten antaen lähes hyödytöntä dataa realistista käyttäjäkuormaa ajatellen. (Kerner 2022.) RUM:lla ei myöskään pysty vertaamaan oman sivuston tuloksia kilpailijan tuloksiin, mikä tekee vertailun kilpailijan sivuston käyttäjädataan vaikeaksi.

4.3 AWS CloudWatch

AWS CloudWatch on palvelu, jonka kautta käyttäjät pystyvät monitoroimaan mitä heidän pystyttämissään AWS:n resursseissa ja sovelluksissa tapahtuu reaaliajassa. CloudWatch pystyy kokoamaan

ja seuraamaan pystytettyjen palveluihin liittyvää metriikkaa. CloudWatch:ssa käyttäjä pystyy luomaan omia mittaristoja seuraamaan käyttäjälle tärkeitä metriikoita. Mikäli palvelu tarvitsisi lisää resursseja suorittaakseen tehtäväänsä moitteettomasti, CloudWatch:n avulla voidaan luoda muistutuksia, kun tietty kynnysarvo ylittyy esimerkiksi virtuaalikoneen kapasiteetissa. (What is Amazon CloudWatch? N.d)

Chai ja Wigmore (2021) kertovat artikkelissaan tarkemmin CloudWatch:in käytöstä. Käyttäjä pystyy pääsemään käsiksi CloudWatch:in toimintoihin sovelluksen ohjelmointirajapinnan (engl. application programming interface lyh. API), komentokehoitteen, AWS:n omien ohjelmistokehityspaketin tai AWS:n nettisovelluksen, AWS Management Consolen kautta. CloudWatch:in käyttöliittymä tarjoaa käyttäjälle sen hetkiset tilastot eri palveluista, jotka käyttäjä pystyy näkemään kaaviomuodossa. (Chai & Wigmore 2021.) AWS:n dokumentaatio (What is Amazon CloudWatch? N.d) sekä Chai ja Wigmore (2021) kertovat kuinka käyttäjät voivat itse asettaa ilmoituksia, jotka lähettävät muistutuksen, kun monitoroidussa palvelussa ylittyy tietty kynnysarvo.



Kuvio 3. Esimerkki miten AWS CloudWatch:in arkkitehtuuri toimii (How Amazon CloudWatch works n.d., muokattu)

CloudWatch:in hyödyt

CloudWatch tarjoaa parhaan tuen yrityksille, jotka käyttävät AWS:n tarjoamia resursseja ja palveluita. CloudWatch tarjoaa käyttäjilleen helposti lähestyttävän käyttöliittymän lisäksi monia muita hyötyjä. Näihin kuuluu muun muassa helppokäyttöisyys, kaiken AWS monitorintidatan näkemisen yhdestä paikasta, näkemyksen sovelluksessa liittyviin syy-seuraussuhteisiin ja muihin yhteyksiin suorituskyvyn kannalta sekä kehittää ja parantaa AWS:n tarjoamia, että kehittäjien omia resursseja. (Chai & Wigmore 2021.)

CloudWatchin heikkoudet

CloudWatchin heikkouksiin kuuluu korkeampi hinta verrattuna kolmannen osapuolien monitorointipalveluihin, vaikka CloudWatch:sta löytyy ilmaiset tasot ne tarjoavat huomattavasti rajoitetumman kokoelman monitorointiin liittyviä resursseja ja työkaluja. Muita CloudWatch:iin liittyviä heikkouksia/haasteita ovat monien vakio AWS-metriikkojen päivittymisnopeus, joka on pienimmillään yhden minuutin intervalleissa. Monimutkaisemmat integraatiot CloudWatch:ssa ovat rajoittuneita usein vain AWS:n omiin resursseihin ja CloudWatch:in syvempi osaaminen ja ymmärrys voi olla pidemmän oppimisprosessin takana. (Chai & Wigmore 2021.)

4.4 AWS CloudWatch RUM

AWS CloudWatch RUM-palvelulla pystytään keräämään ja näkemään käyttäjäpuolen tekemien toimintojen vaikutus web-sovelluksen suorituskykyyn lähes reaaliajassa. Dataa mitä voidaan kerätä ja visualisoida ovat muun muassa sivujen lataamisajat, käyttäjäpuolen häiriöt ja käyttäjien käyttäytyminen sivustolla. Datan avulla voidaan selvittää mitä selaimia ja laitteita käyttäjät käyttävät. (Use CloudWatch RUM n.d.)

CloudWatch RUM auttaa käyttäjiä visualisoimaan poikkeamia sovelluksen suorituksessa ja löytämään relevanttia dataa eri ongelmiin. Näihin kuuluvat virheviestit, käyttäjäistunnot sekä pinojäljet (engl. stack traces). RUM:in avulla voidaan auttaa ymmärtämään loppukäyttäjän käyttäytymisen vaikutus sovellukseen. Näitä muuttujia ovat muun muassa käyttäjien määrä, maantieteellinen sijainti ja käytetyt selaimet. (Use CloudWatch RUM n.d.)

Colmenares (2024) kertoo laajemmin CloudWatch RUM:iin liittyvistä sekoista. RUM:ssa käsiteltävä data pysyy tallennettuna 30 päivää, jonka jälkeen se poistetaan. CloudWatch RUM on suunniteltu

toimimaan saumattomasti web-sovelluksissa. Tämän ansiosta web-sovellukset toimivat parhaan kykynsä mukaan ilman että CloudWatch RUM haittaisi sen suoritusta. Tämänkaltainen toiminnallisuuden ja suorituskyvyn yhteistoiminta on merkittävässä roolissa CloudWatch RUM:in suunnittelufilosofiassa. (Colmenares 2024.)

5 Tutkimus

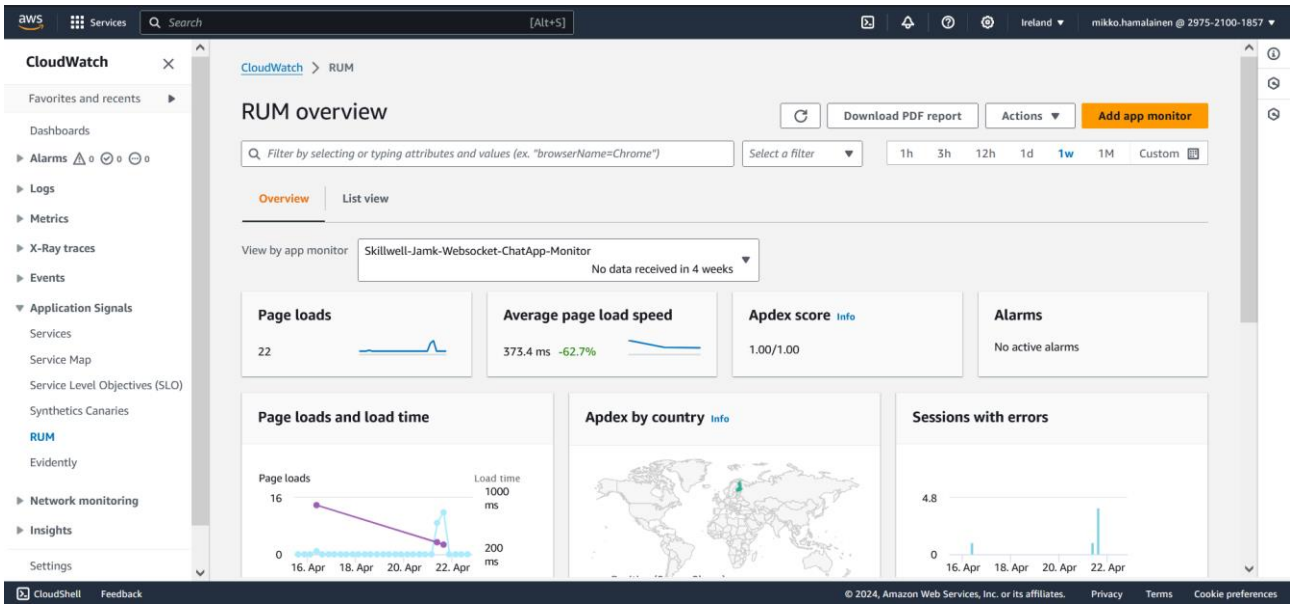
Tässä kappaleessa kerrotaan millaista analytiikkaa CloudWatch RUM:lla saadaan kerättyä. Kappaleessa käydään myös RUM:in käyttöönotto, mitä kaikkia asetuksia voidaan käyttöönotossa muokata ja miten RUM konfiguroidaan Skillwellin omassa sovelluksessa.

Testiympäristö

Datan keräämistä analytiikkaa varten testiympäristönä käytettiin Skillwellin tuotantovaiheessa olevaa Websocket chat-sovellusta, missä käyttäjä voi kirjautua sisälle sovellukseen, luoda chat-huoneita ja kirjoittaa viestejä chat-huoneisiin. RUM liitetään sovellukseen, jonka jälkeen kaikki käyttäjän tekemät toiminnot sovelluksessa luovat dataa RUM:iin ja sen pohjalta saadaan erilaista analytiikkaa. Tämän analytiikan avulla pystytään määrittelemään miten sovelluksen suorituskyky reagoi käyttäjiin ja käyttäjien käyttäytymiseen sovelluksessa.

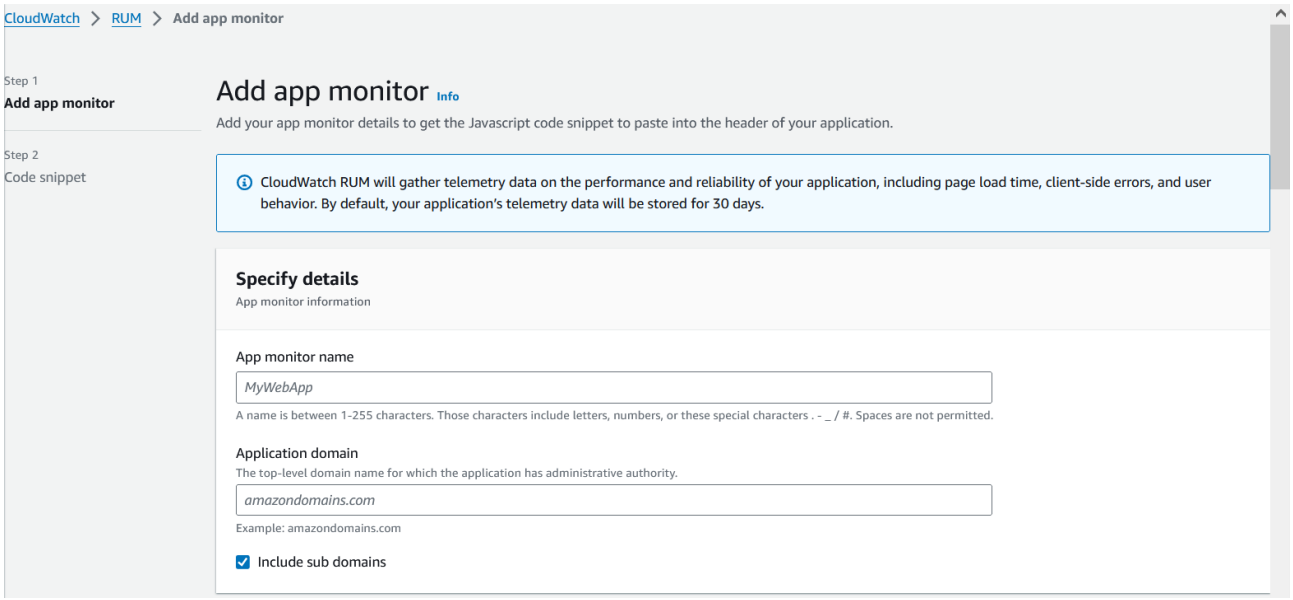
5.1 CloudWatch RUM käyttöönotto sovelluksessa

Jotta RUM voidaan ottaa käyttöön sovelluksessa, pitää ensiksi luoda sovelluksen monitoroija (engl. app monitor). Tämä luodaan valitsemalla ”Add app monitor”.



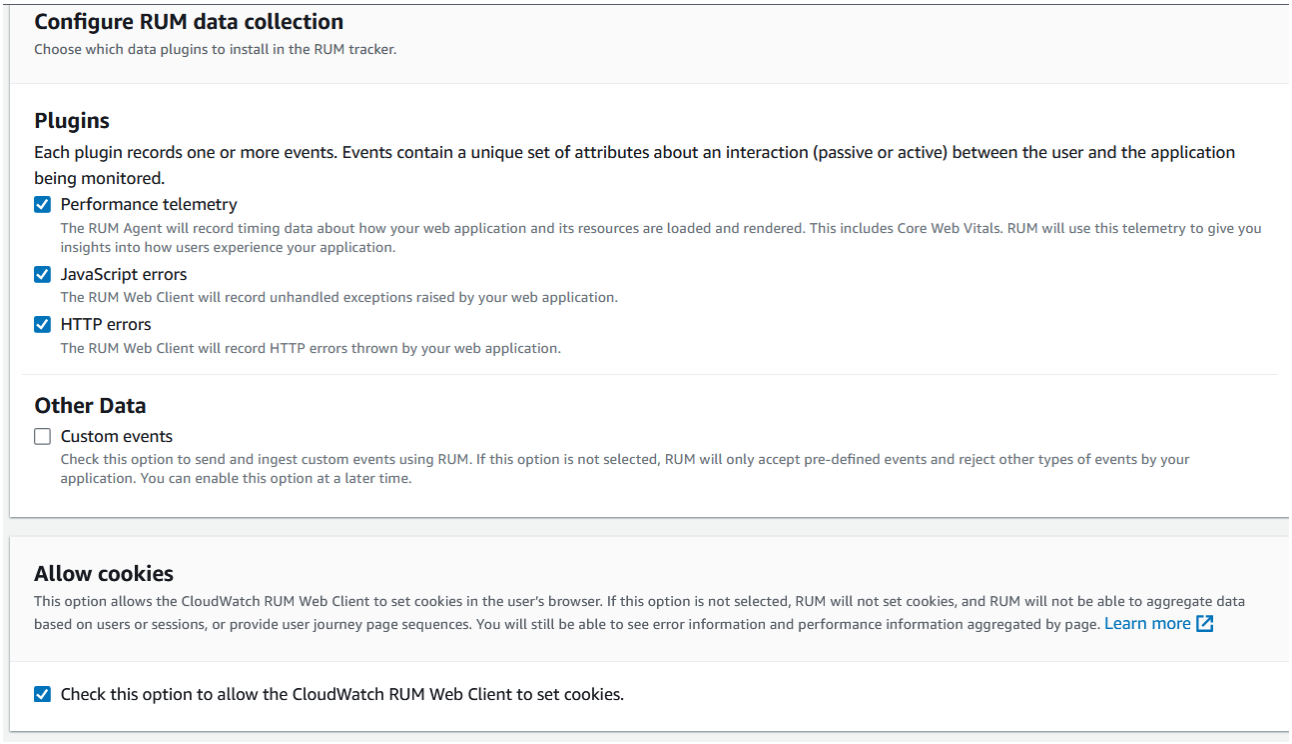
Kuvio 4. CloudWatch RUM-palvelun aloitus-näkymä

”Add app monitor” välilehdeltä annetaan monitoroijalle nimi ja sovelluksen osoite mistä tietoja kerätään. Myös sovelluksen aliosoitteista voidaan kerätä tietoja painamalla ”Include sub domains” vaihtoehto päälle (kts. kuvio 5).



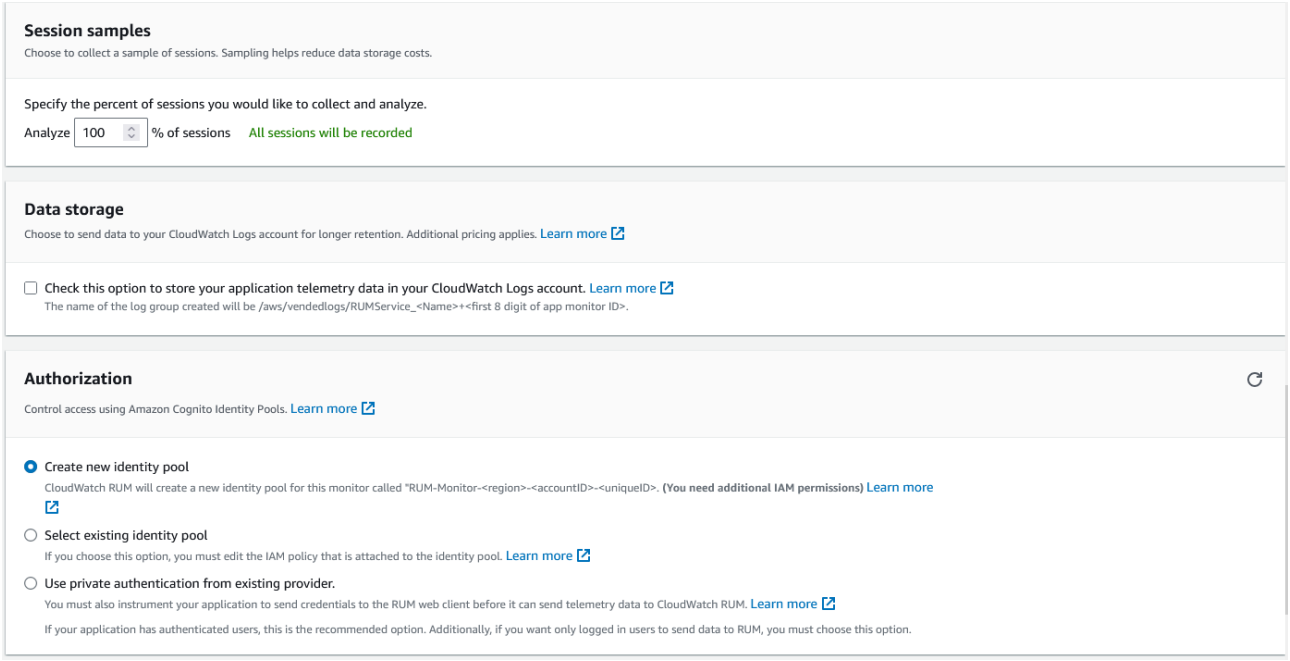
Kuvio 5. CloudWatch RUM-palveluun uuden sovelluksen monitoroijan luominen

Tämän jälkeen konfiguroidaan millaista dataa CloudWatch RUM kerää sovelluksesta (kts. kuvio 6). Cloudwatch RUM:iin voidaan asentaa eri lisäosia, jotka keräävät erilaista dataa sovelluksesta. Vaihtoehtoina löytyvät suoritus telemetria (engl. performance telemetry) jonka avulla CloudWatch RUM kertoo, miten sovellus ja sovelluksen resurssit latautuvat sekä renderöityvät, JavaScript virheilmoitukset (engl. JavaScript errors) joka seuraa kaikkia JavaScript-ohjelmointikieleen liittyviä käsittelemättömiä virheilmoituksia ja HTTP virheilmoitukset (engl. HTTP errors) joka seuraa HTTP-kutsuista tulevia virheilmoituksia. (Step 2: Create an app monitor n.d) HTTP (Hypertext Transfer Protocol) on protokolla, joka on suunniteltu tiedonsiirtoon verkkolaitteiden välillä (What is HTTP? n.d). Käyttäjä voi myös luoda CloudWatch RUM:iin omia tapahtumia (engl. custom events) joista CloudWatch RUM kerää dataa. Omien tapahtumien avulla RUM:iin voidaan määrittää jokin tietty monitoroitava tapahtuma, josta halutaan saada enemmän tietoa siinä tapauksessa missä RUM ei valmiiksi keräisi kyseisestä tapahtumasta tarpeeksi tietoa.



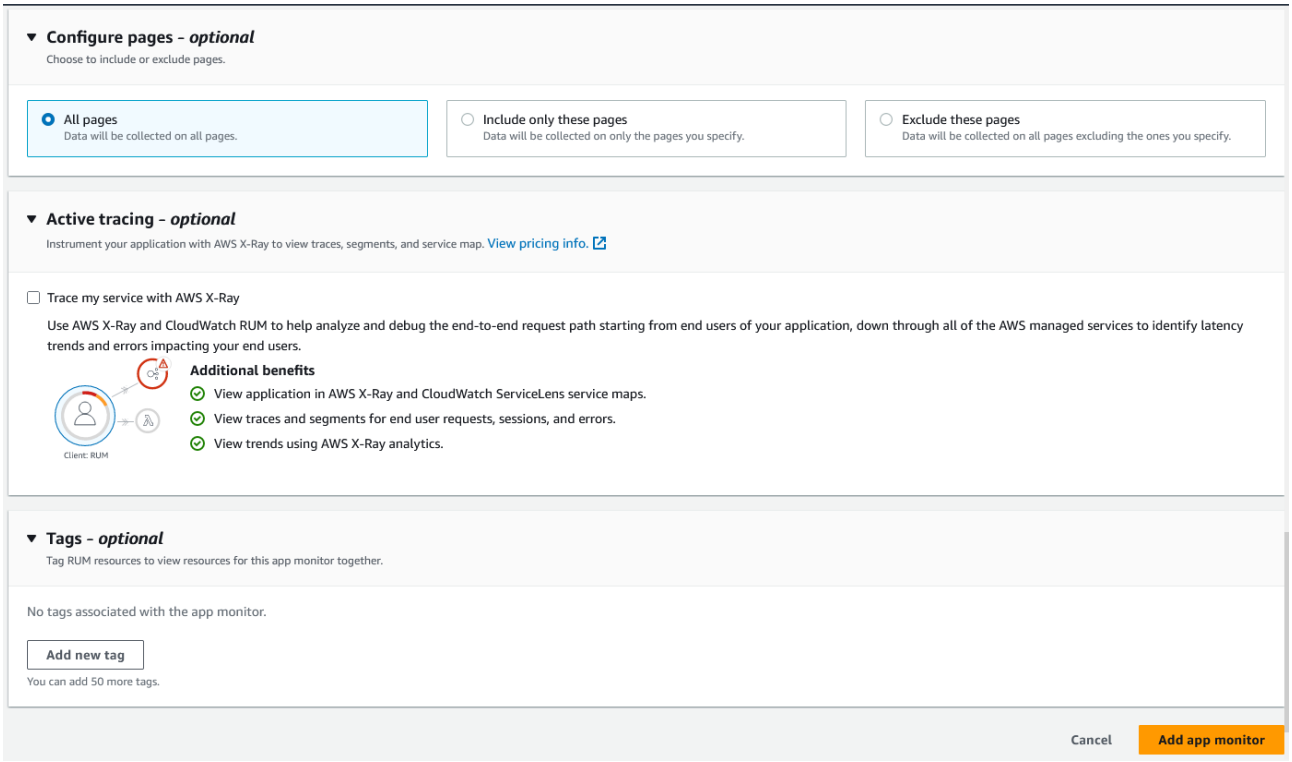
Kuvio 6. CloudWatch RUM-palvelun sovelluksen monitorijan keräämän datan määrittely

Seuraavaksi määritellään CloudWatch RUM-palvelun evästeasetukset (kts. kuvio 6). CloudWatch RUM kerää käyttäjistä tietynlaista dataa vakiona. Ottamalla evästeet käyttöön, CloudWatch RUM kerää käyttäjistä käyttäjätunnuksen ja istuntotunnuksen, joista käyttäjätunnus generoidaan satunnaisesti RUM:in puolesta. Mikäli käyttäjä hyväksyy evästeiden käytön sovelluksessa, jossa RUM on käytössä, alkaa RUM keräämään käyttäjästä dataa, jotka voidaan nähdä RUM:in omasta käyttöliittymästä. Mikäli evästeiden käyttöä sivulla ei hyväksytä, RUM ei pysty keräämään kyseisestä käyttäjästä dataa. Tätä dataa on yhteenlaskettu data perustuen käyttäjätunnuksiin kuten kuinka monta uniikkia käyttäjää on ja kuinka moni käyttäjistä, jotka kokivat virheilmoituksen sovelluksessa. Yhteenlaskettu data istuntotunnuksista, kuten istuntojen määrä ja niiden istuntojen määrä, joissa tapahtui virheilmoitus ja viimeisenä käyttäjän liikkuminen sivustolla, johon kerätään kaikki sivut, joissa käyttäjä on istunnon aikana käynyt. (Data protection and data privacy with CloudWatch RUM n.d).



Kuvio 7. Sovelluksen istuntojen analysoinnin määrittely, sovelluksen telemetrian pidemmän ajan tallennuksen määrittely ja käyttäjien todentaminen

Seuraavaksi valitaan, kuinka iso määrä istunnoista kerätään ja analysoidaan (kts. kuvio 7). Tämä auttaa hallitsemaan datan säilömisestä tulleita kustannuksia. Analysoitavien istuntojen kerättävä määrä määritellään prosenttiluvuilla. Tämän jälkeen valitaan, halutaanko RUM:in keräämää dataa säilöä pidemmäksi aikaa, sillä RUM tallentaa oletuksena datan 30-päivän ajaksi, jonka jälkeen data poistetaan. Pidempää tallennusta varten RUM:in data kopioidaan käyttäjän CloudWatch Logs tilille, josta käyttäjä voi itse määritellä ajan sille, kuinka kauan RUM:in keräämää dataa säilytetään. (Step 2: Create an app monitor n.d.) Seuraavaksi määritellään, kerätäänkö dataa vain vahvistetuista käyttäjistä vai myös vahvistamattomista käyttäjistä. Tämä tapahtuu identity poolien kautta, jotka ovat osa Amazon Cognito-palvelua. Identity poolit ovat tapa todentaa käyttäjiä ja eri asetuksilla voidaan antaa käyttäjille eri valtuuksia sovelluksessa. (Using identity pools (federated identities) n.d.)



Kuvio 8. Sovelluksen monitoroijan valinnaiset asetukset

Viimeisenä tulee sovelluksen monitoroijan valinnaiset asetukset (kts. kuvio 8). Nämä eivät ole pakollisia asetuksia, mutta niiden avulla pystyy räätälöimään sovelluksen monitoroijan tiettyä monitorointitarkoitusta varten. Ensimmäisenä voidaan määritellä miltä sivuilta sovelluksessa dataa kerätään. Seuraavaksi on mahdollista ottaa käyttöön aktiivinen jäljitys (engl. active tracing), jossa otetaan käyttöön AWS X-Ray palvelu RUM:in rinnalle keräämään suoritusdataa sovelluksesta sekä muista sovellukseen liittyvistä palveluista. Viimeisenä palvelulle voidaan antaa tageja, joiden avulla on helpompi tunnistaa ja hallinnoida palveluita järkevästi. Kun kaikki asetukset on tehty, valitaan ”Add app monitor”, joka luo kyseisen monitoroijan.

Kun sovelluksen monitoroijan on pystytetty AWS-ympäristöön, asennetaan RUM-client sovellukseen. Tämä tapahtuu ajamalla ”npm install --save aws-rum-web”-komento sovelluksen kehitysympäristössä. Tällä komennolla asennetaan tarvittavat riippuvuudet (engl. dependencies) sovellukseen, jotta RUM-clientin voi ottaa käyttöön sovelluksessa. AWS:n management console tarjoaa valmiin koodinpätkän, jossa on sovelluksen monitoroijan asetusten mukainen konfiguraatioskripti valmiina. Konfiguraatioskriptin on mahdollista saada TypeScript, JavaScript sekä HTML muodoissa, käyttäjä voi itse valita missä muodossa hän voi ottaa RUM-clientin käyttöön sovelluksessa. Tämä

skripti upotetaan sovelluksen main-tiedostoon, joka toimii sovelluksen suorittamisen aloituspisteenä, jotta RUM-client voi alkaa keräämään sovelluksesta dataa.

```
import { AwsRum, AwsRumConfig } from 'aws-rum-web';

try {
  const config: AwsRumConfig = {
    allowCookies: true,
    endpoint: "https://dataplane.rum.us-west-2.amazonaws.com",
    identityPoolId: "us-west-2:00000000-0000-0000-0000-000000000000",
    sessionSampleRate: 1,
    telemetries: ['errors', 'performance']
  };

  const APPLICATION_ID: string = '00000000-0000-0000-0000-000000000000';
  const APPLICATION_VERSION: string = '1.0.0';
  const APPLICATION_REGION: string = 'us-west-2';

  const awsRum: AwsRum = new AwsRum(
    APPLICATION_ID,
    APPLICATION_VERSION,
    APPLICATION_REGION,
    config
  );
} catch (error) {
  // Ignore errors thrown during CloudWatch RUM web client initialization
}
```

Kuvio 9. Esimerkki RUM-clientin konfiguraatioskriptistä TypeScript muodossa

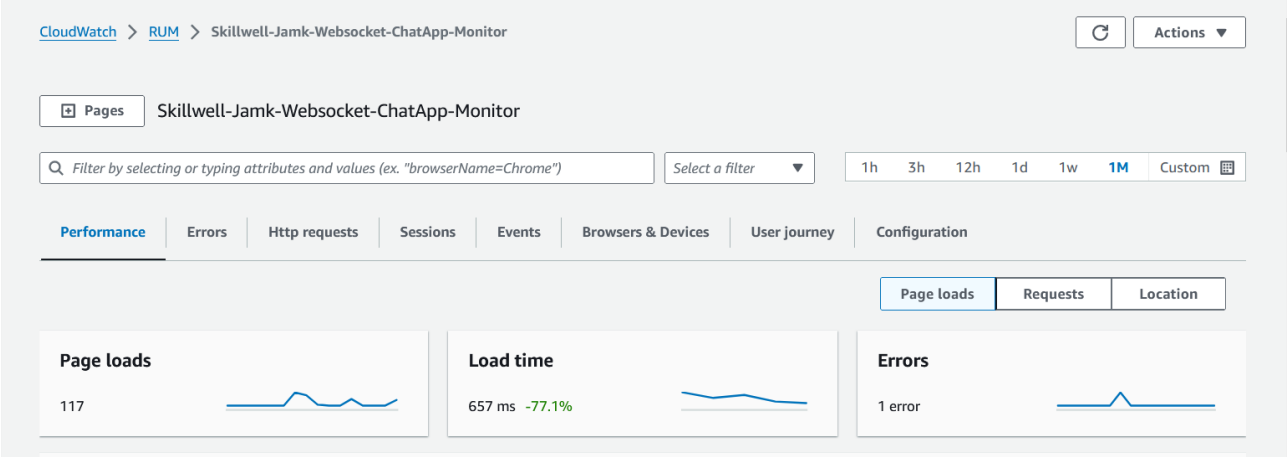
RUM-clientin konfiguraatioskriptissä on kerrottuna kaikki muuttujat (kts. kuvio 9), mitkä määriteltiin sovelluksen monitoroijan konfigurointivaiheessa. AWSRumConfig-objektissa on määritelty evästeiden käyttö, osoite mistä RUM kerää dataa, identity poolin tunnus, istuntojen tallennuksen määritys tallennetaanko istuntoja vai ei. Jos istunnot tallennetaan, annetaan muuttujalle arvo 1. Jos istuntoja ei tallenneta, annetaan arvoksi 0 ja viimeisenä on mistä sovelluksessa kerätään telemetriaa. AWSRumConfig-objektin jälkeen tulevat RUM-clientin sovelluksen tunnus, versio ja millä

AWS:n alueella (engl. region) sovellus pyörii. Nämä sijoitetaan AwsRum-luokkaan AWSRumConfig-objektin kanssa. Loppuun tulee vielä virheidenkäsittely mahdollisia virheilmoituksia varten.

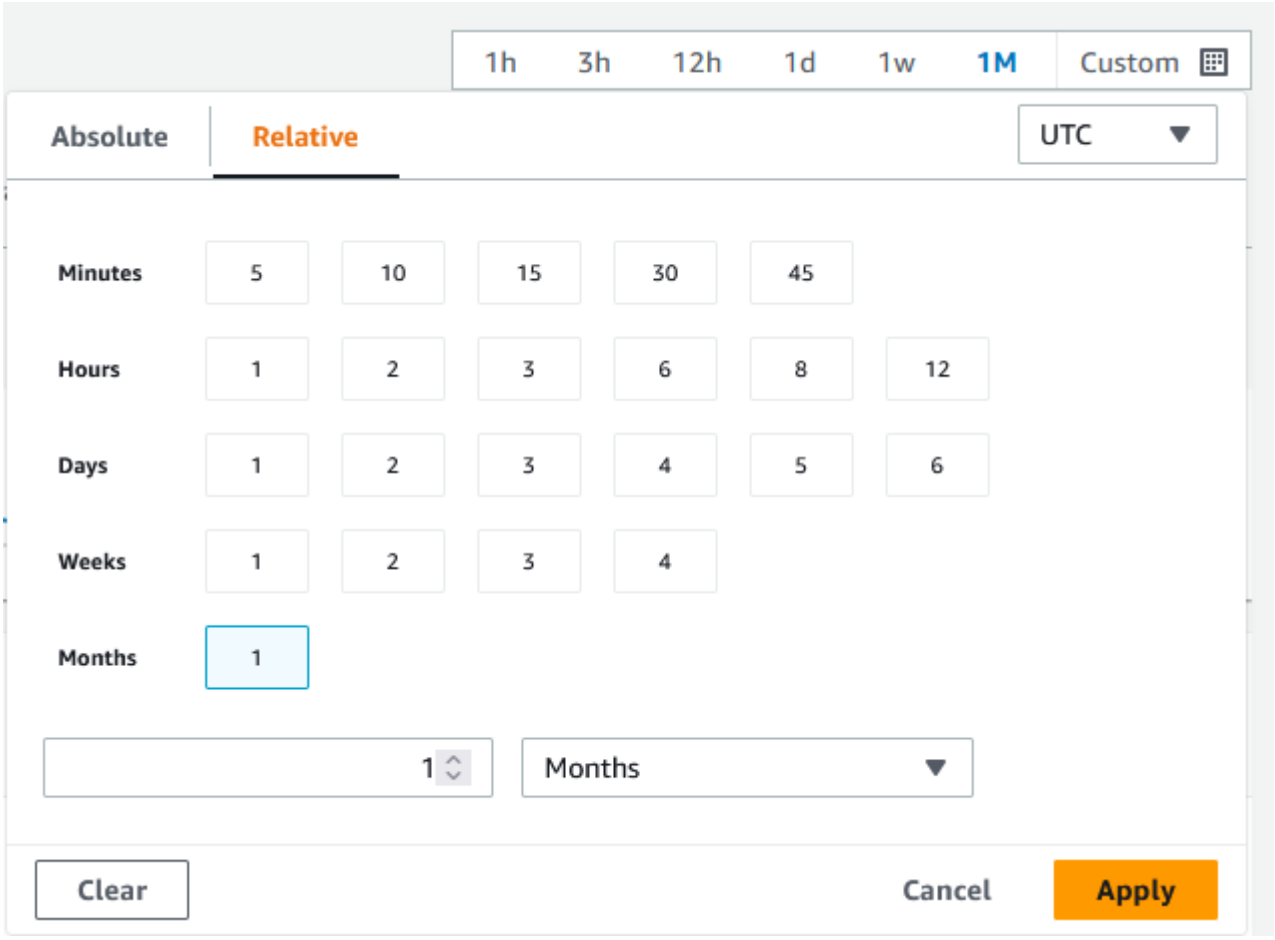
5.2 RUM:in keräämä analytiikka

Suoritusnäkymä

Kun RUM on konfiguroitu ja sovellus saatu pyörimään valitaan luotu sovelluksen monitoroija. Sovelluksen monitoroija avautuu suoritusnäkymään (engl. performance) mistä pääsee näkemään RUM:in keräämää analytiikkaa sovelluksesta (kts. kuvio 10). Näkymässä pystyy määrittämään, kuinka pitkältä ajalta näytetään kerättyä dataa. Keräämisajanjakson pystyy määrittämään tunneista päivään, viikkoon tai kuukauteen. keräämisajanjakson pystyy myös määrittämään omien tarpeidensa mukaan valitsemalla ”Custom” (kts. kuvio 11). Myös eri kategorioiden suoritusdataa pystyy tarkastelemaan valitsemalla ”page loads”, ”request” tai ”location” (kts. kuvio 10).



Kuvio 10. Performance-näkymä RUM:ssa



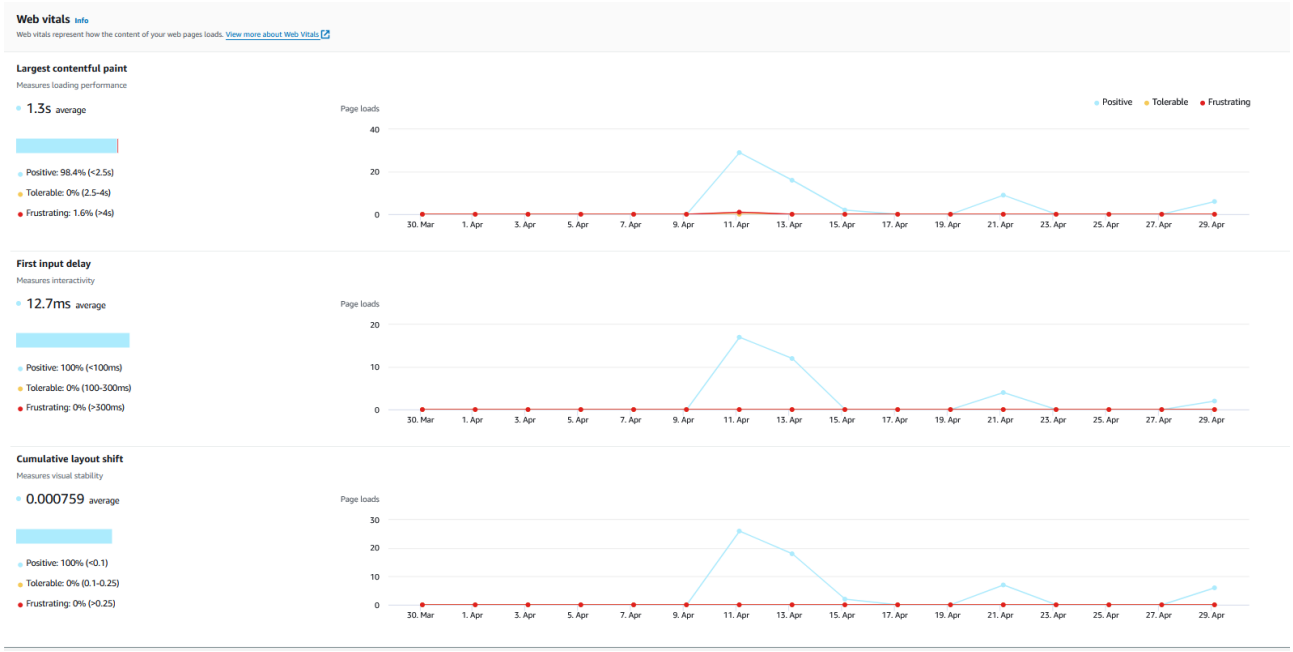
Kuvio 11. Kerätyn datan keräämisajanjakson määrittäminen

Page loads metriikka

Suoritusnäkymästä näkyy kaikki sovelluksen sivun latauksiin liittyvä metriikka, kun valittuna on ”page loads” vaihtoehto. Alempana suoritusnäkymä välilehdellä on taulukko, jossa on koottuna sivun latausaikojen keskiarvot määritetyltä ajanjaksolta (kts. kuvio 12). Pystysarakkeessa nähdään päivämäärät miltä ajalta data on kerätty ja vaakasarakkeessa näkyvät kellonajat. Tämä näkymä on määritetty näyttämään datan keruun huhtikuulta 2024. Taulukosta voidaan nähdä, millaisia latausaikoja sivu on suorittanut. Taulukon yläpuolella on nähtävissä suurin latausviive ja virheiden määrä. Taulukon yläpuolella on annettu värikoodein latausajoille eri arvot. Vaaleanvihreä väri kertoo, että sivun lataus on onnistunut alle kahdessa millisekunnissa mikä on suorituksen kannalta hyvä. Oranssi väri kertoo sivun latautuvan kahden ja kahdeksan millisekunnin välillä mikä tarkoittaa suorituksen kannalta siedettävää aikaa. Punainen väri kertoo sivun latautuvan kauemmin kuin kahdeksan millisenkuntia mikä alkaa olemaan rasite.

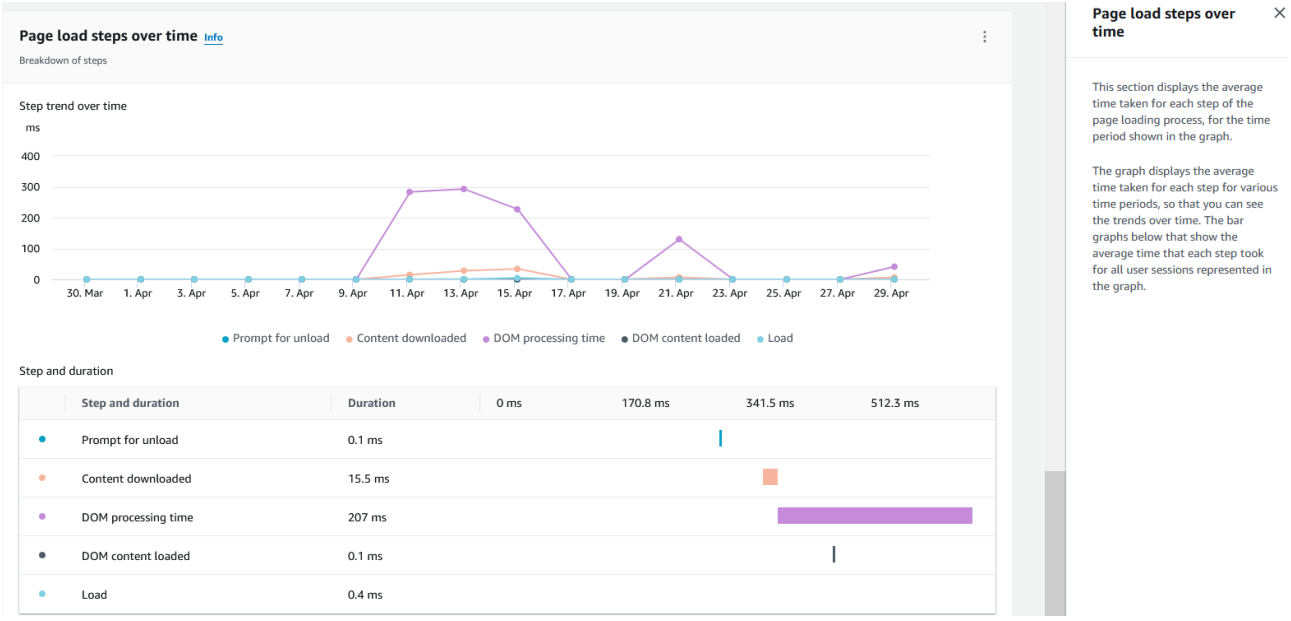
Kuvio 12. Sivun latausaikoja kuvaava taulukko

Seuraavaksi suoritusnäkymässä tulee vastaan sovelluksen Web vitalsit (kts. kuvio 13). Web vitals on Googlen luoma yhtenäinen ohjeistus nettisivujen arvioimaan nettisivujen laatumerkkejä, joiden avulla voidaan tarjota hyvä käyttäjäkokemus. Web vitalsien tavoitteena on yksinkertaistaa suorituskyvyn mittaamiseen käytettävien työkalujen löytäminen ja auttaa sovelluskehittäjiä keskittymään tärkeimpiin mittareihin. (Walton 2023.) Ensimmäisenä taulukossa mitataan sivun latausnopeutta (Largest contentful paint). Hyvän käyttökokemuksen takaamiseksi latausajan pitäisi olla alle 2.5 sekunnin (Walton 2023). Toisena taulukossa mitataan sivun interaktiivisuutta (First input delay). Tämä kuvaa käyttäjän kokemusta sivuston vuorovaikutuksesta ja reagointikyvystä. Tätä mitataan kuinka kauan sivustolla kestää reagoida käyttäjän tekemään vuorovaikutukseen sivulla, esimerkiksi kuinka kauan sivustolla kestää tuoda esiin lista, kun sille kuuluvaa painiketta klikataan. Hyvän käyttäjäkokemuksen takaamiseksi tämä viive pitäisi olla alle 100 millisekuntia. (Osmani & Djirdeh 2022.) Viimeisenä taulukossa mitataan visuaalista vakautta (Cumulative layout shift). Viisuaaline vakaus on käyttäjäkokemusta ajatellen erittäin tärkeä osa-alue, sillä sivun asettelun odottamaton muutos voi vaikuttaa käyttäjäkokemukseen häiritsevästi. Esimerkiksi käyttäjä voi kadottaa kohdan mistä hän oli viimeksi lukenut, kun odottamaton muutos sivun asettelussa tapahtuu. Hyvän käyttäjäkokemuksen takaamiseksi visuaalista vakautta mittaava luku pitäisi olla alle 0.1. (Mihajlija & Walton 2023.)



Kuvio 13. Sovelluksen latausaikoja, interaktiivisuutta ja visuaalissta vakautta mittaavat taulukot, jotka heijastuvat käyttäjäkokemukseen

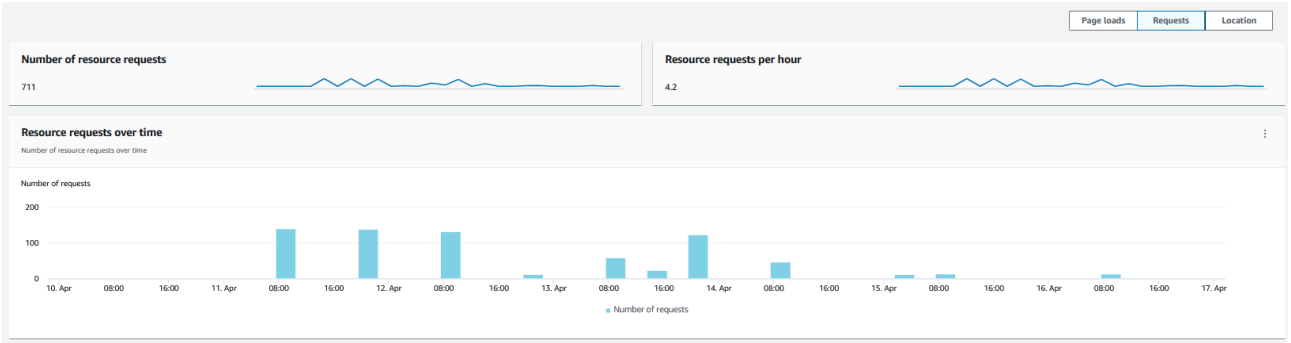
Suoritusnäkymän viimeisenä taulukkona on sivun lataamisen eri vaiheita ja niihin kulunutta aikaa mittavaa taulukko (kts. kuvio 14). Viivadiagrammi kuvaa kuinka kauan aikaa tietty vaihe on sivun latauksessa ottanut sekä milloin tämä vaihe on tapahtunut. Alempi pylväsdiagrammi kuvaa taas, kuinka kauan jokainen sivun lataukseen liittynyt vaihe on kestänyt kaikki käyttäjäistunnoissa keski-määrin.



Kuvio 14. Sivun lataamisen eri vaiheisiin kulunut aika

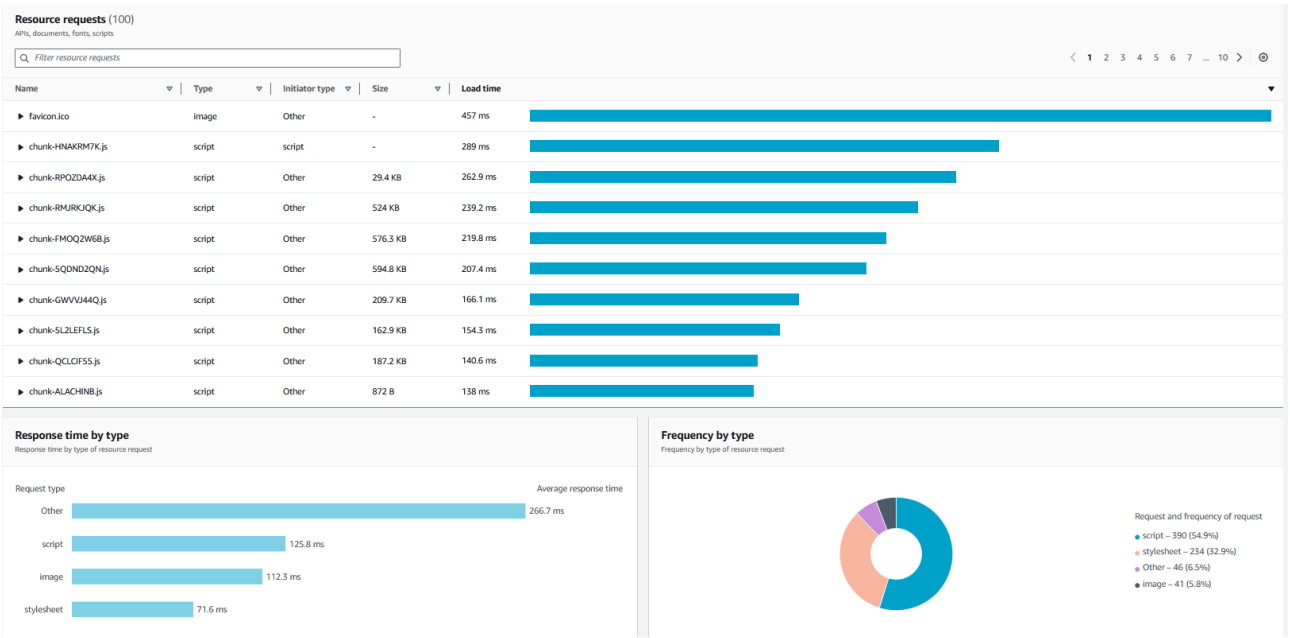
Requests metriikka

Kun suoritusnäkymässä on valittuna ”requests”, näyttää suoritusnäkymä tällöin sovelluksen kutsuihin liittyvää metriikkaa (kts. kuvio 15). Ensimmäisenä yläpalkissa tulevat vastaan resurssikutsujen määrä sovelluksessa sekä resurssikutsujen määrä per tunti. Alempana pylväsdiagrammissa havainnollistetaan resurssien kutsumäärät ja niiden ilmentyminen mitatulla ajanjaksolla.



Kuvio 15. Sovelluksen resurssien kutsumäärät

Alempana olevassa ”Response request”-listauksessa nähdään kutsuttavan resurssin nimi, tyyppi, käynnistimen tyyppi (engl. initiator type), koko ja latausaika. Vasemmassa alalaidassa olevassa pylväsdiagrammissa on havainnollistettu eri resurssityyppien keskimääräinen vastausaika ja oikealla alalaidassa olevassa rinkiläkaaviossa on havainnollistettu kuinka useasti eri resurssityypit ilmenevät kutsuissa.

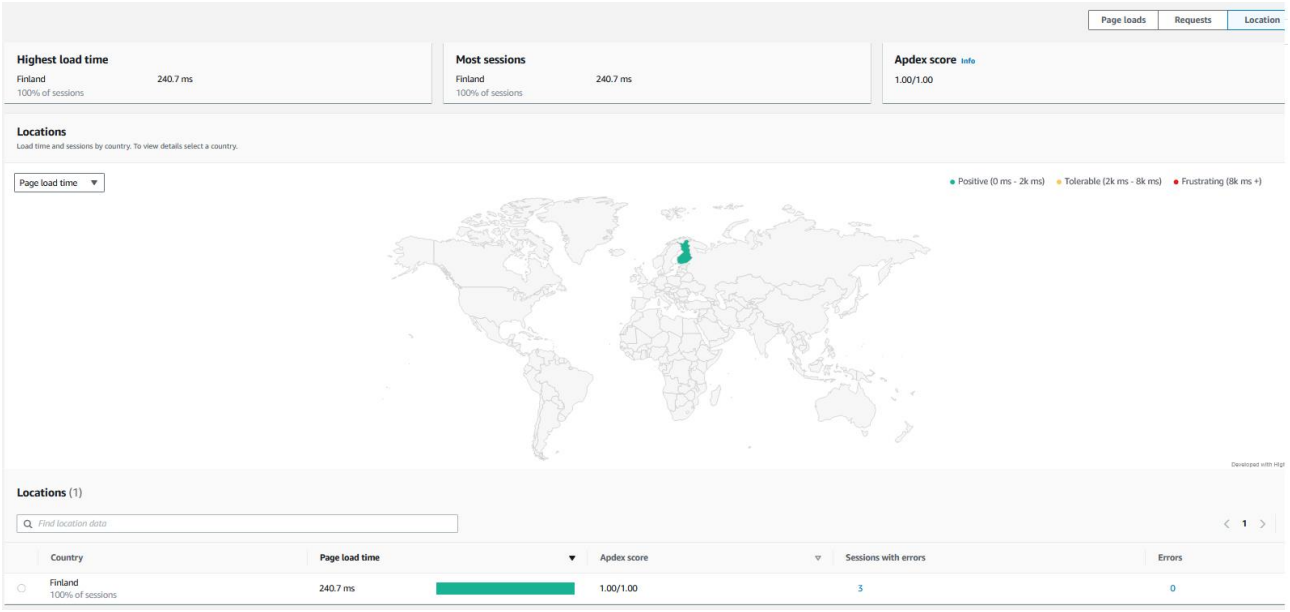


Kuvio 16. Resurssikutsujen tyypit ja niiden ilmentyminen

Sijainti metriikka

Kun suoritusnäkymässä valitaan ”Locations”, näyttää suoritusnäkymä tällöin maailmankartan missä on korostettuna maat, joissa on sivun latauksia tapahtunut (kts. kuvio 17). Ylälaidan palkeissa nähdään missä maassa on ollut korkeimmat latausajat ja kuinka monta prosenttia ne vievät kaikista istunnoista, missä maissa on ollut eniten istuntoja ja kuinka monta prosenttia maan istunnot vievät kaikista istunnoista ja viimeisenä Apdex-pisteet, joka on standardi sovellusten vasteaikojen raportoimiseksi, vertailemiseksi ja arvioimiseksi. Apdex-pisteytys auttaa ymmärtämään loppukäyttäjän tyytyväisyyttä. Pisteytys tapahtuu 0 ja 1 välillä. 1 tarkoittaa erittäin tyytyväinen ja 0 vähiten tyytyväinen. (How CloudWatch RUM sets Apdex scores n.d.) Maailmankartasta pystyy valitsemaan sivujen lataukset tai istunnot ja valinnan mukaa kartta päivittyy sen valinnan mukaan.

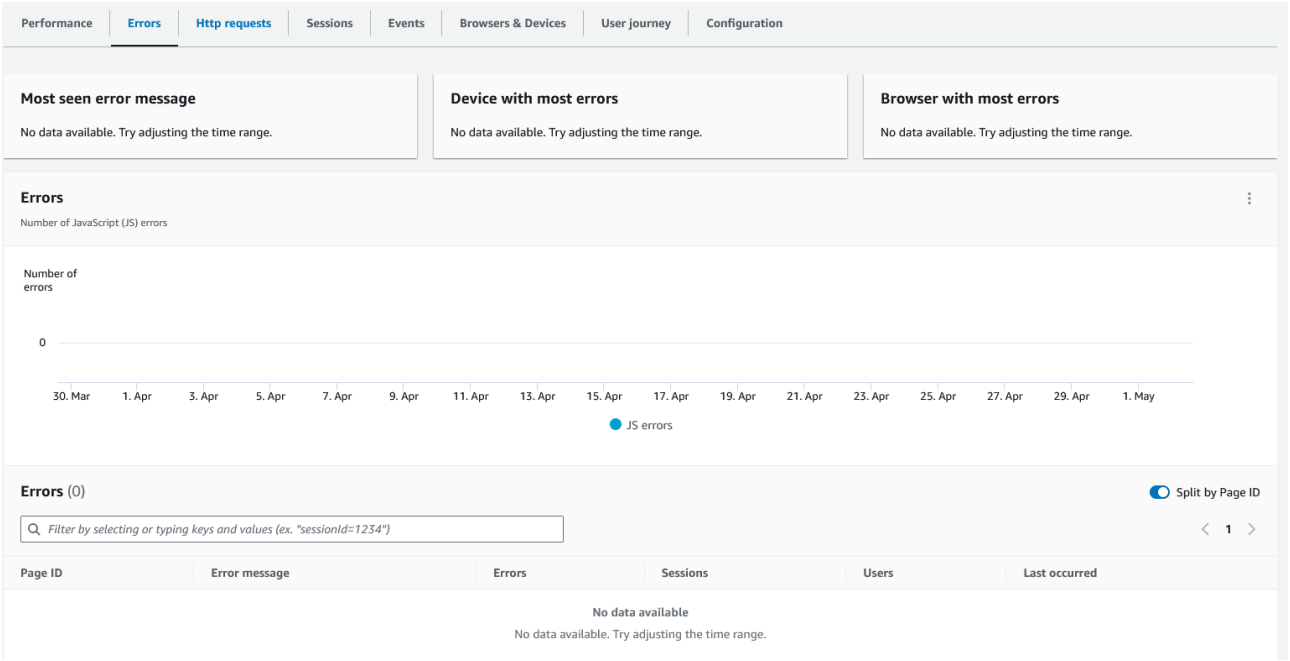
Kartan alla ”Locations”-listassa nähdään maan nimi, kauanko sivun lataus on siellä kestänyt, maan Apdex-pisteet, istunnot joissa on tapahtunut virheilmoitus ja virheilmoitukset.



Kuvio 17. Sijainnit missä sivujen latauksia on suoritettu

Virheilmoitusnäkymä

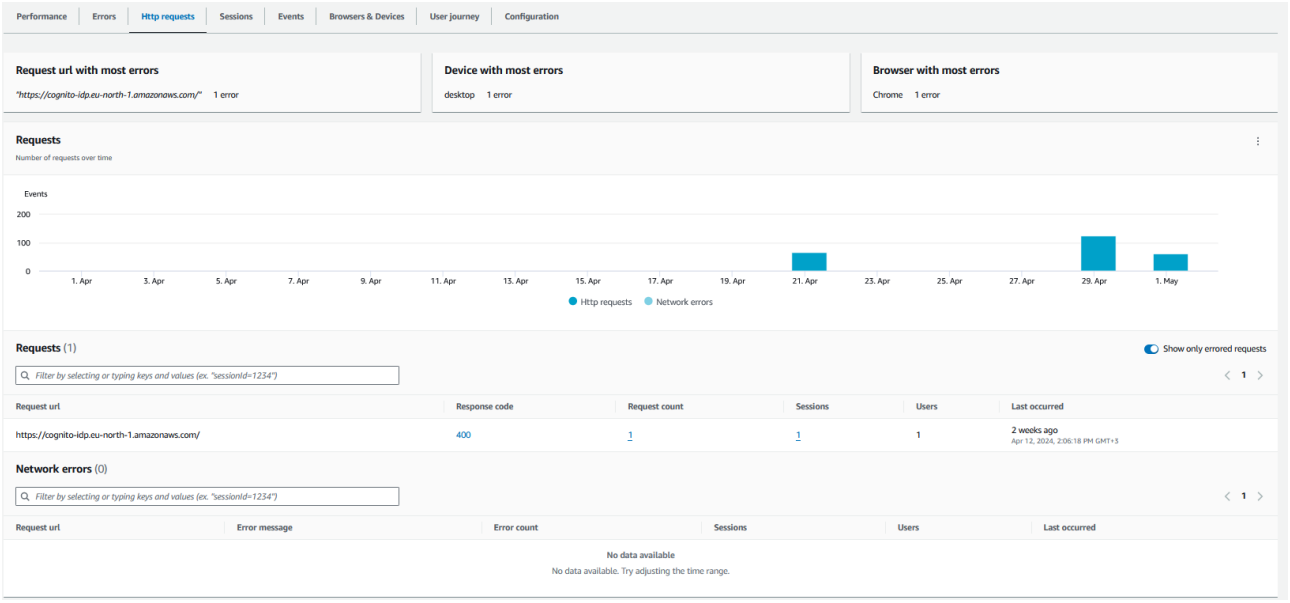
Virheilmoitusnäkymä käsittelee sovelluksessa tulleiden virheilmoituksia (engl. errors) ja niiden määrää (kts. kuvio 18). Virheilmoitusnäkymän ylälaidassa on kolmessa palkissa esillä mitä virheilmoituksia on eniten tullut vastaan, millä laitteella virheilmoituksia on tullut eniten ja millä selaimella virheilmoituksia on tullut eniten. Alempana viivadiagrammissa näytetään, kuinka paljon JavaScript-virheilmoitukisa on tullut kullakin päivänä. Viivadiagrammin alapuolella on lista mistä pystyy näkemään sivun tunnuksen, jossa virheilmoitus tapahtui, itse virheilmoitusviestin, muut virheilmoitukset, istuntojen määrä missä virheilmoitus tapahtui, käyttäjien määrä, joilla virheilmoitus tapahtui sekä milloin viimeksi kyseinen virheilmoitus on tapahtunut.



Kuvio 18. Errors-näkymä RUM:ssa

Http-kutsut-näkymä

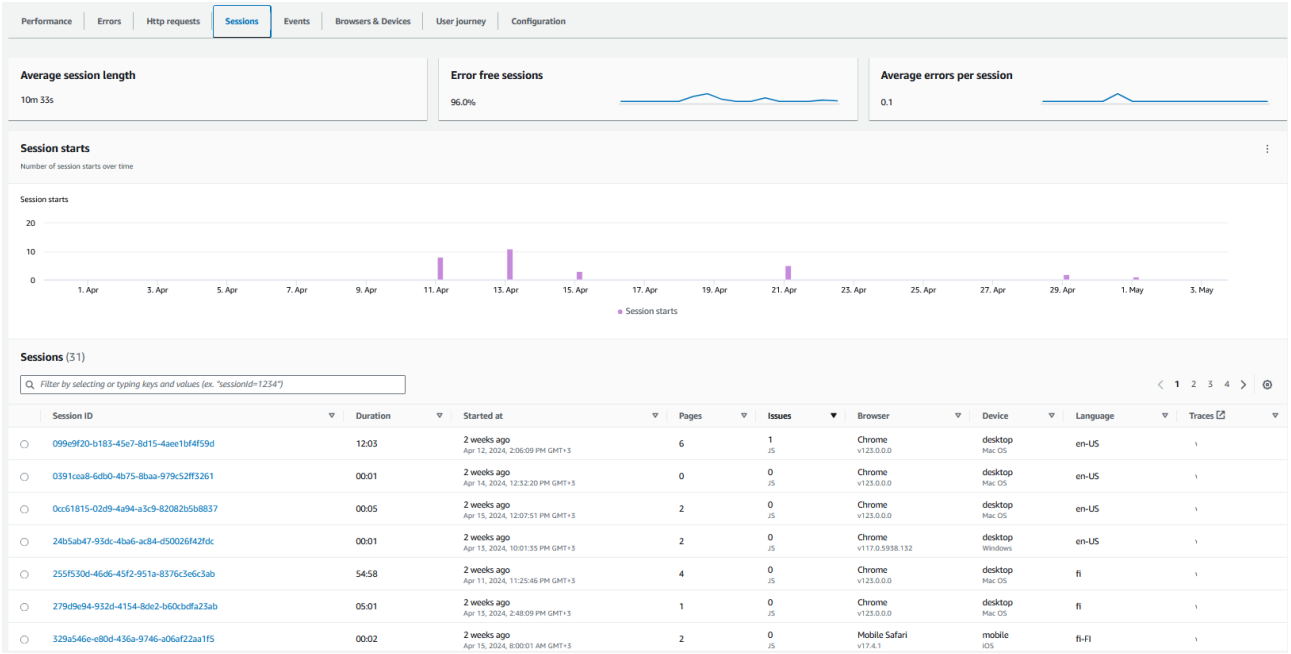
Http-kutsut-näkymässä (engl. http requests) käsitellään sovellukseen kohdistuneita http-kutsuja ja niihin liittyviä virheilmoituksia (kts. kuvio 19). Näkymässä ylälaidassa olevissa palkeissa näkyy mistä URL-osoitteen kutsuista on tullut eniten virheilmoituksia, laite millä on tullut eniten virheilmoituksia, ja selain millä on tullut eniten virheilmoituksia. Pylväsdiagrammi havainnollistaa http-kutsujen määrän tiettyinä päivinä sekä verkkovirheiden määrän. ”Requests”-listassa näkyy mikä oli kutsun URL-osoite, vastauskoodi, kutsujen määrä, istuntojen määrä, käyttäjien määrä sekä milloin kyseinen kutsu on viimeksi ilmentynyt.



Kuvio 19. Http-kutsuihin ja niiden virheilmoituksiin liittyvä näkymä

Istunnot-näkymä

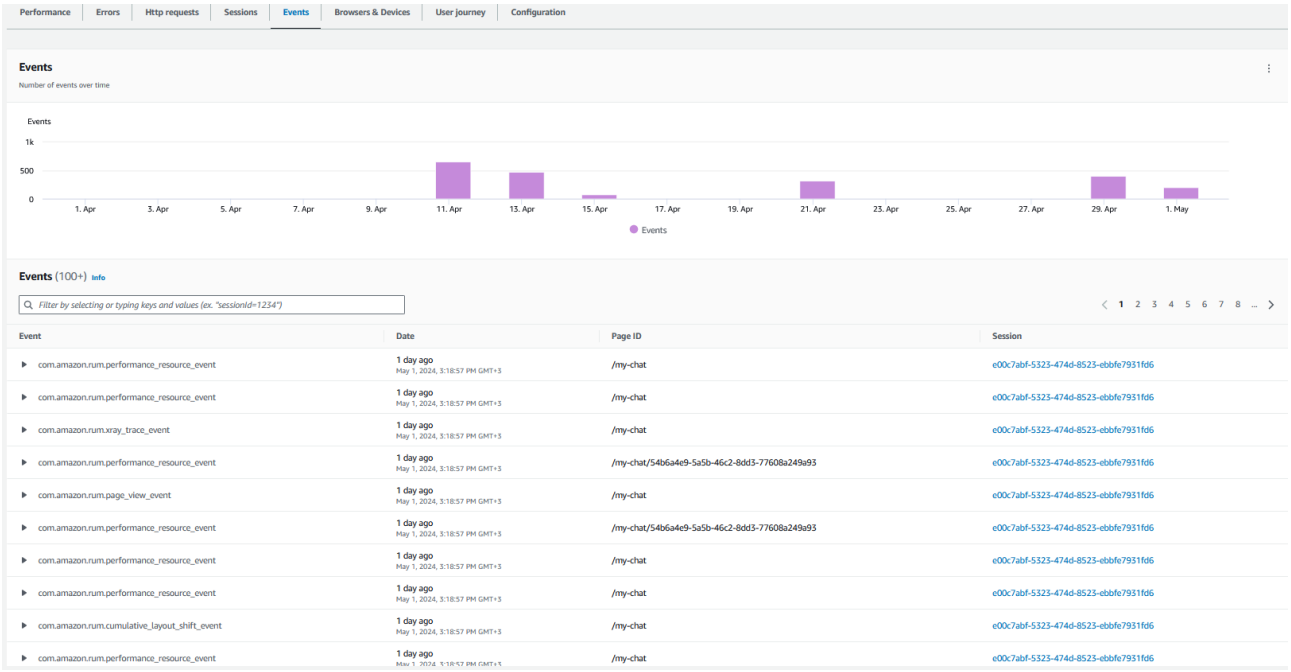
Istunnot-näkymässä pystytään tarkastelemaan sovelluksen käyttäjien istuntojen dataa (kts. kuvio 20). Yläpalkeissa pystytään näkemään kuinka pitkiä istunnot ovat keskimäärin, kuinka monta prosenttia istunnoista on virheilmoitusvapaita ja virheilmoitusten ilmentyminen keskimäärin per istunto. Pylväsdiagrammi havainnollistaa kuinka paljon istuntoja on aloitettu kullakin päivänä. Alempana ”Sessions”-listassa näkyvät kaikki tapahtuneet istunnot. Listassa nähdään istunnon tunnus, pituus, aloitusaika, kuinka monta sivua istunnossa käsiteltiin, ongelmien määrä, selain, laite, selaimen kieli sekä istunnon jäljitys.



Kuvio 20. Istunnot-näkymä

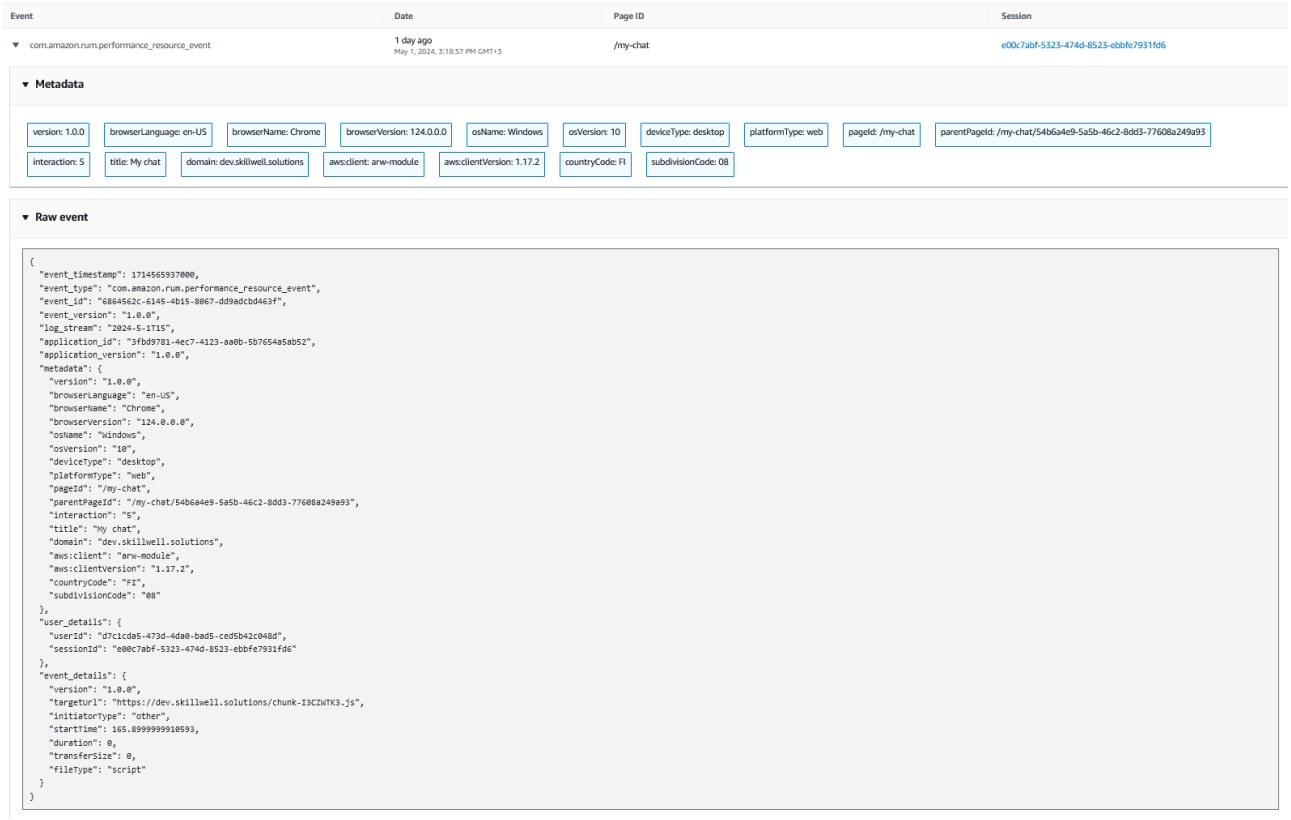
Tapahtumat-näkymä

Tapahtumat-näkymässä pystytään tarkastelemaan kaikkia tapahtumia mitä sovelluksessa on suoritettu (kts. kuvio 21). Pylväsdiagrammi havainnollistaa kuinka paljon tapahtumia on tapahtunut sovelluksessa tiettyinä päivinä. Alempana ”Events”-listassa pystytään näkemään mitä tapahtumia on suoritettu, niiden suorituspäivämäärä, sivun tunnus missä tapahtuma on suoritettu ja istunto missä tapahtuma on suoritettu.



Kuvio 21. Tapahtumat-näkymä

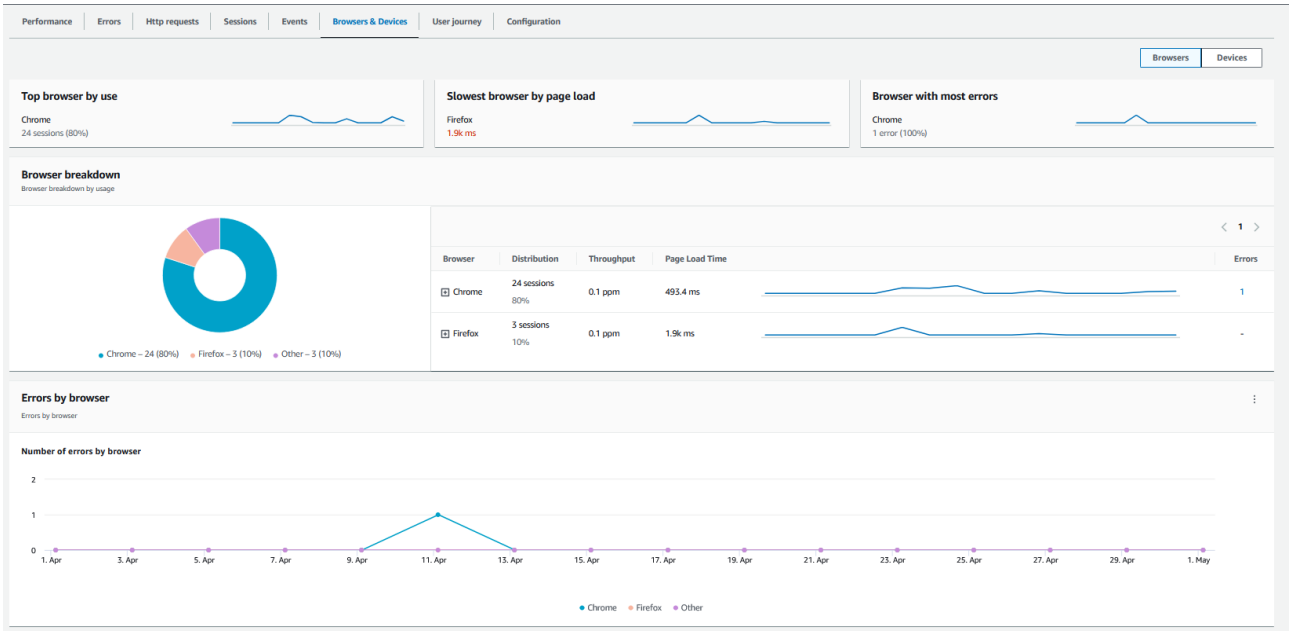
Jokaisen tapahtuman pystyy laajentamaan painamalla tapahtuman edessä olevaa nuolta, joka avaa laajemmat tiedot tapahtumasta (kts. kuvio 22). Tiedoissa näkyy tapahtuman metadata, jossa kuvaillaan tapahtuman tiedot ja ominaisuudet. Metadatan alapuolella näkyy raakatapahtuma eli RUM:in luoma objekti, jota ei olla mitenkään muunneltu. Tästä voidaan havainnoida kaikki tapahtumaan liittyvät muuttujat.



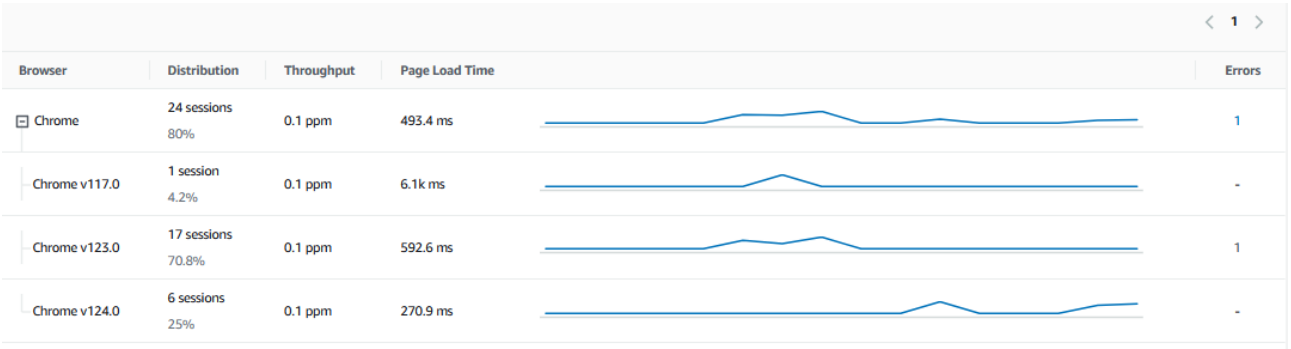
Kuvio 22. Yksittäisen tapahtuman metatiedot ja raakadata

Selaimet ja laitteet-näkymä

Selaimet ja laitteet-näkymässä pystytään näkemään suoritusmetriikkaa eri selainten ja laitteiden välillä (kts. kuvio 23). Suoritusmetriikka mittaristot ovat selaimille ja laitteille sama ja ne voi valita näkymän oikeasta yläkulmasta. Yläpalkeissa voidaan nähdä mikä on käytetyin selain, millä selaimella on hitain sivun latausaika ja millä selaimella tulee eniten virheilmoituksia. Rinkilädiagrammissa havainnollistetaan, kuinka selaimia käytetään kokonaisuudessaan. Rinkilädiagrammin vieressä olevassa listauksessa näkyy, kuinka monessa istunnossa selainta on käytetty, mikä niiden läpimenoaika on, kuinka nopeat sivujen latausajat selaimilla on sekä virheilmoitukset. Painamalla selaimen nimen edessä olevaa plusmerkkiä voidaan avata selaimen eri versioiden vertailu (kts. kuvio 24).

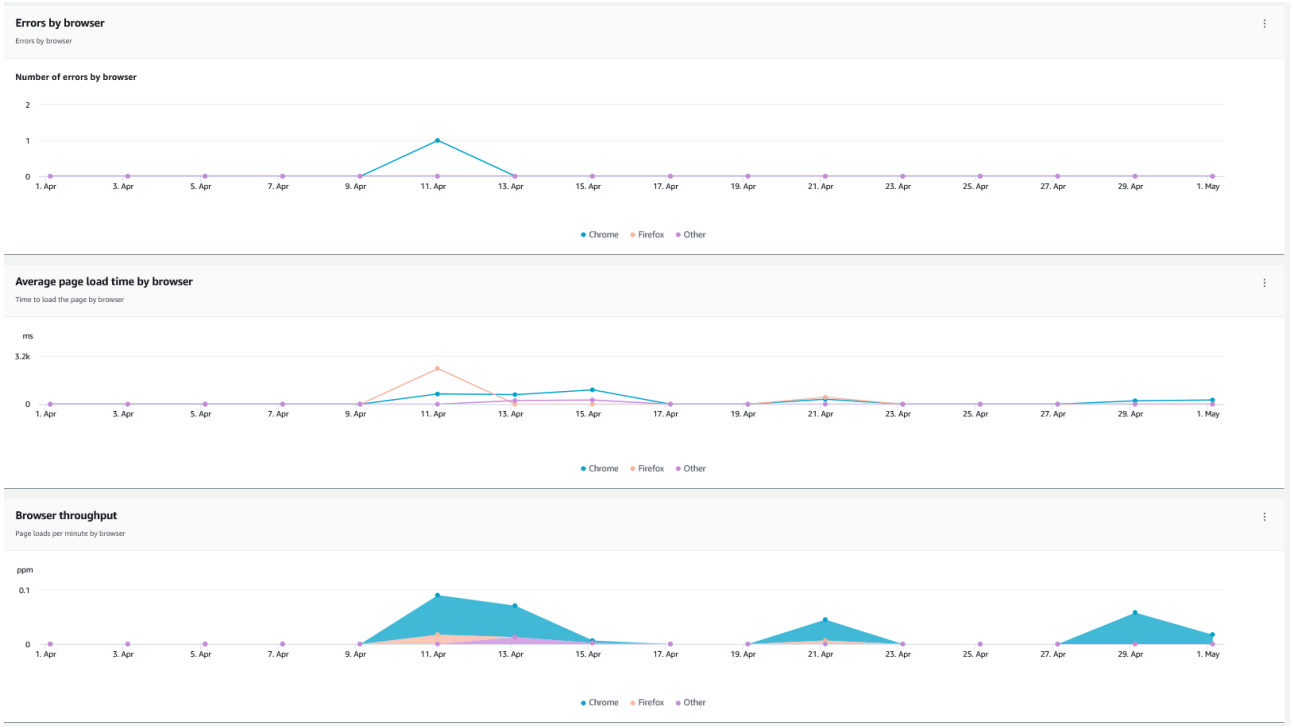


Kuvio 23. Selaimet ja laitteet-näkymän data kun valittuna on selain-tiedot



Kuvio 24. Selaimen eri versioiden erot sivun latausajoissa

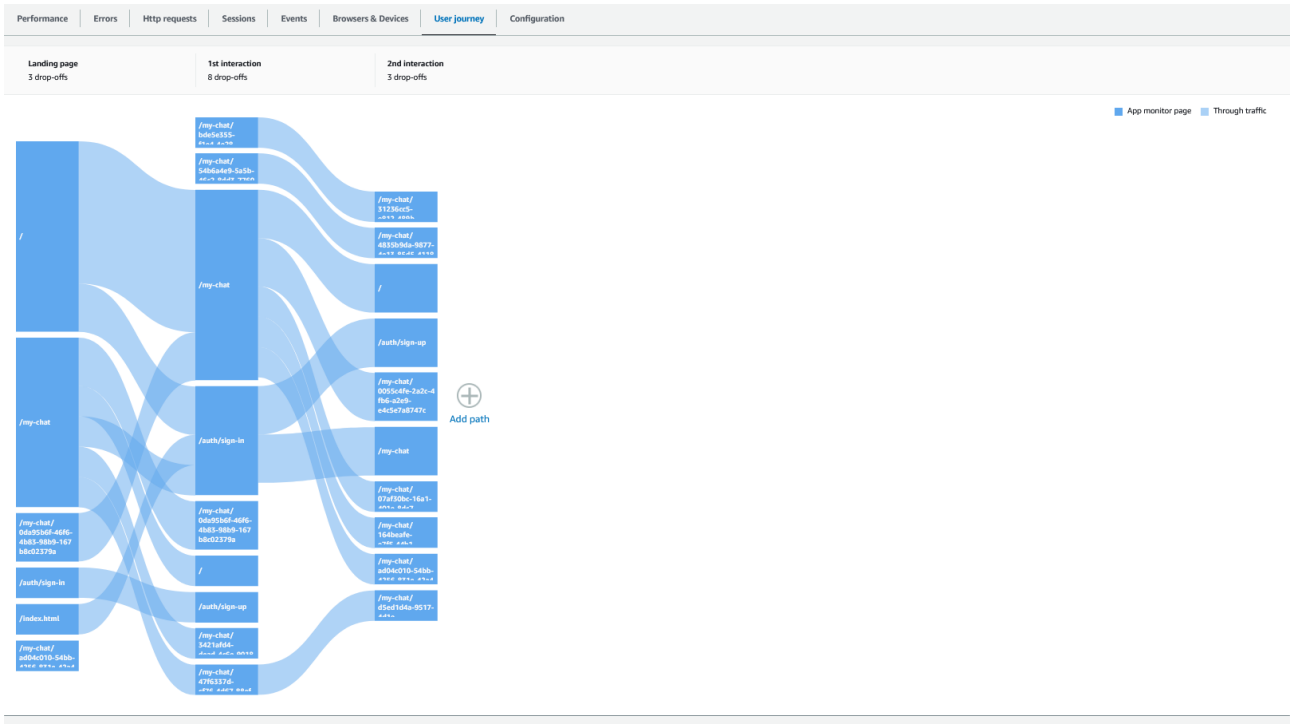
Seuraavaksi näkymässä tulevat viivadiagrammit, jotka kuvaavat selaimien virheilmoitusten määriä, selainten välisten sivun latausaikojen eroja ja selainten välisiä läpimenoaikoja eri päiviltä (kts. kuvio 25).



Kuvio 25. Selaimista kerättyä metriikkaa

Käyttäjän matka-näkymä

Käyttäjän matka-näkymän kaaviossa on havainnollistettu miltä eri sivuilta käyttäjät ovat tietyille sivuille päätyneet (kts. kuvio 26). Kaaviossa ensimmäisenä on aloitussivu ja palkeissa sivun nimi, mistä käyttäjä on aloittanut sivuston käytön. Tämän jälkeen nähdään minkä sivujen kautta käyttäjä on kulkenut sivustolla. Jokaisen siirtymän yläpuolella voidaan nähdä, kuinka moni käyttäjä on lähtenyt sivulta pois. Tummansinisellä nähdään monitoroitavat sivut ja vaaleansinisellä nähdään käyttäjien liikenne eri sivujen välillä. Kaaviossa olevasta plusmerkistä voi lisätä uuden kulkureitin kaavioon.



Kuvio 26. Kaavio kuvaamaan käyttäjän matkaa sivustolla

6 Tulokset

Tässä kappaleessa käydään tutkimuksesta tulleita tuloksia ja johtopäätöksiä sekä vastataan tutkimuskysymyksiin.

6.1 Mikä on RUM?

RUM eli real user monitoring on sovelluksen suorituskykyä mittaava osa-alue, joka keskittyy käyttäjän vaikutukseen sovelluksessa. RUM täten palvelee tietoperustassa esitettyä Well-Architected viitekehystä, missä suorituskyvyn tehokkuus esiintyy yhtenä pilarina laadukkaan pilvipalveluinfrastruktuurin luomisessa. RUM:in avulla pyritään mittaamaan käyttäjien kokemusta sovelluksessa ja saada ymmärrys, miten käyttäjät kokevat sovelluksen käytettävyyden. RUM auttaa havaitsemaan nopeasti mahdolliset käyttäjiä koskevat ongelmat, jonka ansiosta tämä auttaa sovelluskehittäjiä luomaan käyttäjäystävällisempiä sovelluksia.

6.2 Miten CloudWatch RUM otetaan käyttöön sovelluksessa?

Alussa luotiin monitoroitavalle sovellukselle sovelluksen monitoroija, jonka avulla RUM pystyi keräämään dataa sovelluksen käytöstä, joka osoittautui suoraviivaiseksi prosessiksi. Monitoroijalle annettiin nimi ja osoite mistä kerätä dataa. Tämän jälkeen määriteltiin mitä dataa haluttiin kerättävän, joihin RUM:ssa oli annettu jo valmiita vaihtoehtoja, joista kaikki otettiin käyttöön. Mahdollisuutena oli myös määritellä omia tapahtumia, joiden tarkoitus on käyttäjän itse määritellä mitä tietoa sovelluksesta kerättään. Tämä vaihtoehto antaisi mahdollisuuden muokata monitoroitavia kohteita sovelluksessa. Omia tapahtumia ei otettu käyttöön, sillä testisovellusta varten RUM:in keräämät oletusmetriikat olivat tarpeeksi päteviä ymmärtämään millaista dataa RUM kerää.

Kriittinen kohta RUM:in konfiguroinnissa on evästeiden käyttö. Ilman evästeiden käyttöä RUM pystyy keräämään tietoa istunnoista ja käyttäjien toiminnoista, mutta data ei olisi niin tapauskohtaisen tarkkaa, kun se voisi olla evästeiden kanssa, joten evästeet otettiin käyttöön. Datan lähettämistä varten määritettiin tarvittavat Identity Poolit ja IAM rooli oikeudet. RUM-client hyödyntää näitä saadakseen oikeudet lähettää dataa sovelluksen monitoroijalle. Lopuksi valinnaisiin asetuksiin ei koskettu sen enempää, aktiivinen jäljitys laitettiin päälle, jotta sovellukseen liittyviin ongelmiin saadaan Amazon X-Rayn kautta lisää ymmärrystä.

Kun sovelluksen monitoroija oli luotu, saatiin siitä valmis skripti, joka upotettiin testisovelluksen koodiin. Sitten kehitysympäristössä ajettiin ”npm install --save aws-rum-web”-komento, joka asensi tarvittavat riippuvuudet sovellukseen, jotta RUM-client pystyi keräämään dataa sovelluksesta.

6.3 Mitä tietoa/analytiikkaa CloudWatch RUM:lla voidaan kerätä?

Testisovelluksen käytön jälkeen päästiin tarkastelemaan millaista dataa RUM kerää. Kerätty data oli keräämishetkestä hetkestä eteenpäin 30-päivää käytössä eikä sitä vanhempaa dataa päässyt enää tarkastelemaan, kuten tietoperustan Cloudwatch RUM osiossa oli esitetty. RUM:in keräämä data oli hyvin monipuolista, erittäin selkeästi havainnollistettu ja tarkasti kerättyä. Tähän tietenkin vaikutti määrittely siitä, millaista dataa haluttiin kerättävän, joka määritettiin RUM:in konfiguroinnissa. On tärkeää myös ottaa huomioon, että ilman evästeiden käyttöä data ei välttämättä olisi ollut yhtä yksityiskohtaisen tarkkaa. Kerätty data oli kuitenkin hyvin vajaata, joka johtui testisovelluksen pienestä käyttäjämäärästä. Tämä havainnollistaa tietoperustassa esitetyn heikkouden RUM:in käytöstä kehitysvaiheessa olevissa sovelluksissa.

Kuvio 27. CloudWatch RUM:in keskeisimmät ominaisuudet

Käyttäjän toimintaan liittyvää metriikkaa RUM keräsi erittäin laajasti monelta osa-alueelta. Tietoperustassa CloudWatch RUM osiossa esitetyt kerätyn datan analysoitavat kohteet päästiin todentamaan tutkimuksessa hyvin. Kuviosta 27 pystytään näkemään CloudWatch RUM:in keskeiset ominaisuudet sen käyttöön liittyen. Kerätty data koostui pääosin suorituskykyyn, virheilmoituksiin, kutsuihin, istuntoihin ja laitteisiin, että selaimiin liittyvästä metriikasta. RUM:iin kerätyn metriikan avulla pystytään ymmärtämään paremmin, kuinka sovellusta käytetään ja millaisia kehityskohteita sovelluksesta löytyy, kuten tietoperustan RUM:in osiossa käsiteltiin.

Suorituskyvyssä metriikka keskittyi sivun latausaikoihin, resurssien pyyntöihin sekä sijainnin vaikutukseen sovelluksessa päivittäisellä tasolla erilaisina diagrammeina ja kaavioina. Virheilmoituksissa ja Http-kutsuissa metriikka keskittyi ilmoitusten ja kutsujen määrään sekä kuinka paljon tiettyä virheilmoitusta on esiintynyt. Istunnoissa metriikka keskittyi istuntojen pituuteen, istuntojen määrän kuvaamiseen pylväsdiagrammina sekä istuntoihin liittyviin virheisiin. Istuntojen metriikan pohjalta pystyttiin havaitsemaan erilaisia käyttäytymismalleja liittyen käyttäjien toimintaan sovelluksessa kuten tietoperustan RUM osiossa oli todettu. Tapahtumissa metriikka keskittyi tapahtumien määrään pylväsdiagrammina, tapahtumien kirjaamiseen mitä sovelluksessa on suoritettu sekä tapahtumien tyyppeihin.

Lähes kaikissa käyttäjän toimintaa mittaavilla osa-alueilla käyttäjän käyttämä selain ja laite olivat metriikan kohteina. Tämä metriikka sisälsi, kuinka paljon virheilmoituksia tietty laite tai selain on kohdannut, millä laitteella tai selaimella oli ollut pisimmät latausajat sovelluksessa ja miten selaimen versio vaikutti asiaan ja kuinka paljon mitäkin laitetta tai selainta oli käytetty.

RUM pystyi keräämään dataa erittäin monelta käyttäjään liittyvältä osa-alueelta, joka on yksi RUM:in keskeisiä ominaisuuksia. Käyttäjästä itsestään kerättiin tietoa selaimesta ja laitteesta, jotka olivat käytössä testisovellusta käytettäessä. Tietoperustassa esitettiin, kuinka datan avulla pystytään näkemään millainen vaikutus käyttäjän valitsemalla selaimella ja laitteella oli sovelluksen suoritukseen. Tämän pohjalta pystytään optimoimaan sovellusta monelle eri alustalle ja parantamaan sovelluksen käytettävyyttä. RUM loi myös kaavion liittyen käyttäjien liikkumisesta sovelluksessa. Kaavio näytti miltä sivulta sovelluksessa käyttäjät aloittivat sovelluksen käytön ja kuinka moni käyttäjä lähti sivulta pois missäkin vaiheessa.

Johtopäätökset

Tutkimuskysymyksillä oli tarkoitus selvittää CloudWatch RUM-palvelun keräämän datan mahdollisuudet ja rajoitteet sekä millainen käyttöönottoprosessi palvelulla on. Kerätyllä datalla pystyi havaitsemaan monia seikkoja sovelluksen suorituskyvystä. Data auttoi visualisoimaan kuinka paljon kuormaa sovellus kohtaa ja miten se vaikutti virheilmoitusten yleistymiseen sekä sivun latausten nopeuksiin. Hyvin rakennetut kaaviot sekä niihin liitetyt kynnysarvot auttavat aloittelevaakin kehittäjää ymmärtämään miten sovellus suorittaa. CloudWatch RUM:in tapa visualisoida kerätty data

on erittäin helposti tulkittavissa, jonka ansiosta palvelu toimii mainiona työkaluna kehittäjille. Palvelu tarjoaa syvällisen metriikan käyttäjien kokemuksista sovelluksessa, mikä auttaa huomaamaan esimerkiksi sovelluksen vikakestoisuuden suurten käyttäjämäärien rasituksessa. Tämä tieto auttaa kehittäjiä pohtimaan sovelluksen kehityskohteita käyttäjän näkökulmasta.

Käyttöönotto oli tehty erittäin suoraviivaiseksi prosessiksi mikä alentaa kynnystä ottamaan palvelu käyttöön entisestään. Konfiguroinnissa kaikki vaihtoehdot ja vaiheet oli selkeät sekä niistä oli mahdollista saada enemmän informaatiota tarkennuksia varten AWS:n dokumentaatiossa. Käyttöönoton suurimpia vahvuuksia oli RUM-clientin skriptin luominen konfiguroinnin pohjalta. Valmiin skriptin ansiosta käyttöönotossa ei tarvitse miettiä skriptin muotoilua tai muuttujien arvoja. Toisena vahvuutena nähtiin, kuinka nopeasti palvelu alkoi keräämään dataa heti käyttöönottoprosessin päätyttyä, mikä oli sillä samalla hetkellä, kun skripti upotetaan koodiin, ajetaan asennuskomento ja sovellus laitetaan käyntiin.

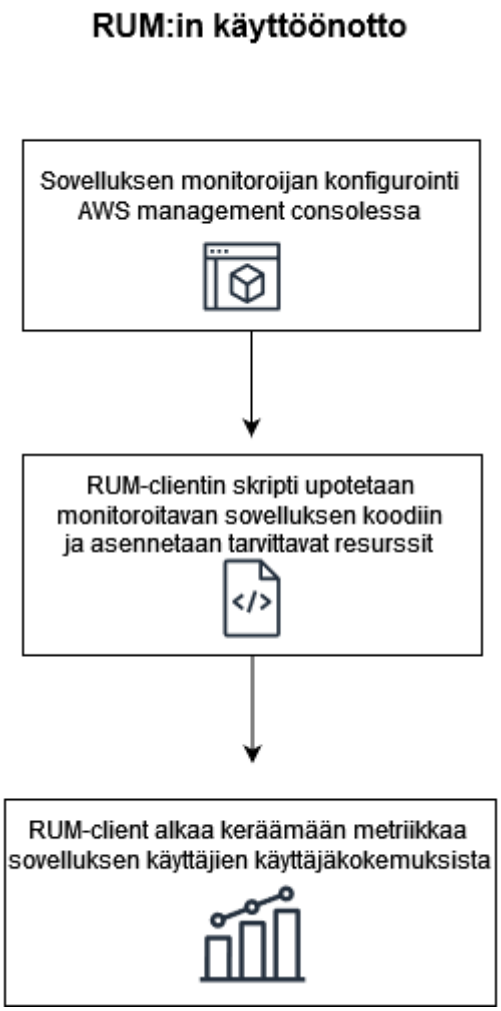
7 Pohdinta

Tässä kappaleessa käydään läpi tutkimuksen tavoitteiden saavuttamista sekä tutkimuksesta tulleita jatkokehitys mahdollisuuksia. Kappaleessa käydään lisäksi tutkimuksen luotettavuutta sekä yleistettävyyttä ja eettisyyttä.

Tutkimuksen tavoitteena oli ymmärtää mikä on RUM, miten CloudWatch RUM otetaan käyttöön sovelluksessa ja millaista analytiikkaa RUM:lla pystytään keräämään. Tuloksina tutkimuksesta saatiin ymmärrys RUM:in konfiguroinnista ja siihen liittyvistä seikoista, mitä metriikkaa RUM kerää, mitä niistä pystytään näkemään ja mitä RUM käsitteenä on.

Tutkimuksessa päästiin hyvin käsittelemään RUM:in käyttöönoton vaiheet ja konfiguraatiot. Kuviossa 28 on havainnollistettu RUM:in käyttöönotto, joka oli erittäin johdonmukainen ja selkeä prosessi. Käyttöönotossa kaikki vaiheet olivat selkeästi selitetty ja järkevässä järjestyksessä. Kuten kuviossa 28 esitetään, käyttöönoton vaiheita ei ollut montaa ja RUM-clientin asentaminen sovellukseen datan keräämistä varten oli yksinkertainen ja helppo tehdä. Nopean asennuksen jälkeen RUM oli valmis keräämään dataa sovelluksesta ilman turhia käynnistys- tai latausaikoja.

RUM:in keräämän datan ymmärtäminen saatiin myös hyvin todennettua tutkimuksessa testisovelluksen käytön myötä, joka loi tarkasteltavaa metriikkaa RUM:iin. Kerätyn datan ymmärtäminen oli erittäin mutkatonta, sillä RUM:in käyttöliittymässä kerätylle datalle oli eri kategoriat sekä mahdolliset kynnysarvot, joiden avulla dataa pystyttiin tulkitsemaan vaivattomasti. Kuviot ja kaaviot olivat käyttöliittymässä selkeät ja johdonmukaiset, joista pystyi helposti tekemään päätelmiä eri suorituskyvyllisiin seikkoihin liittyen. Kerätyn datan määrä olisi kuitenkin saanut olla suurempi, jota tutkimuksessa ei pystytty toteuttamaan. Tutkimuksessa käytetty data antoi mahdollisuuden tutkia millaista dataa RUM kerää, mutta olisi suositeltavampaa, että kerätyn data määrän olisi suurempi, jotta RUM:in toiminnasta saataisiin kokonaisvaltaisempi kuva.



Kuvio 28. Yksinkertaistettu kuvaus RUM:in käyttöönotosta

Testisovellus missä RUM:ia käytettiin, oli tutkimushetkellä tuotantovaiheessa, mikä vaikutti RUM:in metriikoiden luotettavuuteen, koska sovelluksessa ei ollut kunnon käyttäjäkuormaa. RUM on kehitysvaiheessa olevissa sovelluksissa huono, sillä kerätty data ei ole verrattavissa oikeaan kuormitukseen, kun verrataan sovellukseen, joka on valmiina julkaistu käyttäjille. Tämä tuli ilmi jo tietoperustan kokoamisvaiheessa. Täten RUM:in keräämää metriikkaa ei voida tässä tutkimuksessa yleistää, mutta kaikki taulukot ja diagrammit RUM:ssa antoivat parempaa ymmärrystä, miten kerätty data näkyy ja visualisoidaan RUM:ssa.

Luotettavuus ja eettiset kysymykset

Tutkimusta voidaan pitää luotettavana, sillä tutkimuksessa tarkisteltiin RUM:in hyviä ja huonoja puolia. Teoriapohjaan kerätty materiaali oli kerätty alan ammattilaisten kokoamista julkaisuista

sekä CloudWatch RUM:iin pohjautuvat lähteet olivat pääosin AWS:n omasta kehittäjädokumentaatiosta. Pitää kuitenkin ottaa huomioon, että IT-ala on alati kehittyvä ja nopeaa tahtia etenevä ala, jossa vuodenkin ero voi muuttaa palveluiden ulkoasua tai toiminnallisuuksia huomattavasti. Tutkimustulokset ovat myös muiden hyödynnettävissä ja sovellettavissa, vaikka työn toimeksiantajana toimikin Skillwell. Tuloksien tarkoitus oli tuottaa laajempi ymmärrys CloudWatch RUM-palvelusta, josta on myös monelle muulle sovelluskehitysalan yritykselle käytännöllistä tietoa. Käyttäjäkokemuksen tärkeys on ratkaisevassa roolissa sovelluskehityksessä ja CloudWatch RUM tarjoaa siihen hyvät työkalut. CloudWatch RUM:ia voidaan tosin käyttää vain AWS:n pilvipalveluiden kautta, sillä se on AWS:n lanseeraama palvelu.

Tutkimusta tehdessä noudatettiin hyvää tieteellistä käytäntöä sekä JAMKin eettisiä periaatteita. Yksi suuri eettinen kysymys työssä oli RUM:in evästeet. Ilman evästeitä RUM ei pysty keräämään yhtä yksityiskohtaista dataa käyttäjistä, mikä tietyllä tavalla vaikeuttaa käyttäjäkokemuksen hahmottamista ja ymmärtämistä sovelluskehityksen näkökulmasta. Täten evästeitä ei saisi ilman käyttäjän hyväksyntää asentaa sivulle, sillä se rikkoisi EU:n tietosuoja-asetuksia. Tässä työssä testisovellus ei ollut kuluttajien käytössä julkisesti, mutta RUM:in käytön tulevaisuutta ajatellen tästä pitää olla selkeä linjaus mitä tietoja kerätään ja olla läpinäkyvä käyttäjälle mitä tietoja hänestä kerätään. Kuitenkin evästeiden avulla kerätystä datasta saisi paljon hyödyllistä informaatiota sovelluskehittäjille.

Jatkokehitysmahdollisuudet

Tutkimuksessa keskityttiin ymmärtämään, miten RUM otetaan käyttöön ja millaista analytiikkaa RUM pystyy keräämään. CloudWatch RUM tarjoaa myös mahdollisuuden monitoroida käyttäjän itse määrittämiä omia tapahtumia sovelluksessa, joihin tässä tutkimuksessa ei perehdytty syvemmin. Jatkokehityksen kannalta omien tapahtumien tärkeys nähtiin tärkeäksi.

Omat tapahtumat (engl. custom events) ovat käyttäjän itse määrittämiä tapahtumia ennalta määritettyjen tapahtumien monitoroinnin lisäksi. Jokaiselle tapahtumatyypille määritetään nimi ja datan tyyppi, jota kyseinen tapahtuma kerää sovelluksesta. Jokainen tapahtuma saa olla kooltaan enintään 6KB. Omia tapahtumia voidaan kerätä vain, jos RUM:in konfiguraatiossa ne on määritetty

käyttöön. Tapahtumalle täytyy määritellä nimi tai tyyppi, joka kuvaa tapahtumaa. Nimen merkkimäärä voi olla 1–256 merkkiä pitkä ja nimi voi olla aakkosnumeeristen merkkien, alaviivojen, väliviivojen ja pisteiden yhdistelmä. Tapahtuman lisätietoihin kuuluu mitattavan datan tiedot mitä tallennetaan CloudWatch RUM:iin ja se on oltava objektimuodossa sisältäen eri arvoja. (Send custom events n.d.) Kuviossa 28 on esimerkki omasta tapahtumasta, jossa monitoroidaan käyttäjän tekemiä toimintoja sivustolla. RUM saa omat tapahtumat tässä muodossa sovelluksesta ja sen jälkeen esittää ne tapahtumalle kuuluvassa näkymässä RUM:in käyttöliittymässä,

Jatkokehityksen kannalta omien tapahtumien luominen auttaisi yksilöimään haluttua dataa RUM:ssa ja helpottaisi sovelluskehittäjiä perehtymään yksityiskohtaisemmin käyttäjäkokemuksen parantamiseen. Ennalta määritetyissä metriikoissa ei aina välttämättä ole juuri sitä dataa kerättynä mikä olisi hyödyllistä tietyn ongelman ratkaisemiseen sovelluksessa käyttäjän kokemusta ajatellen.

Send a custom event using the `recordEvent` API, NPM example

```
awsRum.recordEvent('my_custom_event', {
        location: 'IAD',
        current_url: 'amazonaws.com',
        user_interaction: {
            interaction_1 : "click",
            interaction_2 : "scroll"
        },
        visit_count:10
    }
)
```

Kuvio 29. Esimerkki luodun tapahtuman skriptistä (Send custom events n.d)

Lähteet

APM. N.d. Datadog:in dokumentaatiosivusto. Viitattu 20.2.2024. https://docs.datadoghq.com/tracing/

AWS Well-Architected Framework — 6 Pillars Explained. 2023. Medium. Viitattu 14.4.2024. https://medium.com/@vasanthabalaji/aws-well-architected-framework-6-pillars-explained-83af8e29ce0d

AWS Well-Architected. N.d. Amazon Web Services verkkosivut. Viitattu 21.2.2024. https://aws.amazon.com/architecture/well-architected/

Bister, T. 2019. Tietojenkäsittelyn opinnäytetyö: viittoja ja karttoja tutkimisen ja kehittämisen teille. Viitattu 18.4.2024. https://janet.finna.fi, Jyväskylän ammattikorkeakoulu.

Bonso, J. Päivitetty 30.6.2023. AWS Well-Architected Framework – Six Pillars. Tutorialsdojo. Viitattu 14.4.2024. https://tutorialsdojo.com/aws-well-architected-framework-six-pillars/

Brusch, K., Lockhart, E. & Demaitre, E. Päivitetty elokuussa 2022. What is APM? Application performance monitoring guide. TechTarget. Viitattu 20.2.2024. https://www.techtarget.com/searchenterprisedesktop/definition/Application-monitoring-app-monitoring

Chai, W. & Wigmore, I. Päivitetty toukokuussa 2021. AWS CloudWatch. TechTarget. Viitattu 15.4.2024. https://www.techtarget.com/searchaws/definition/CloudWatch

Colmenares, V. 2024. AWS CloudWatch RUM (Real User Monitoring). Medium. Viitattu 12.3.2024. https://vicolmeheredia.medium.com/aws-cloudwatch-rum-real-user-monitoring-2fa5a0f2e2b7

Data protection and data privacy with CloudWatch RUM. N.d. Amazon Web Services dokumentaatio. Viitattu 24.4.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-privacy.html#CloudWatch-RUM-cookies

Hawes, C. Päivitetty 12.12.2023. What is real user monitoring (RUM)? Dynatrace. Viitattu 12.3.2024. https://www.dynatrace.com/news/blog/what-is-real-user-monitoring/

How Amazon CloudWatch works. N.d. Amazon Web Services dokumentaatio. Viitattu 15.4.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch_architecture.html

How CloudWatch RUM sets Apdex scores. N.d. Amazon Web Services dokumentaatio. Viitattu 2.5.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-apdex.html

Jaffe, J. 2020. Importance of Application Performance Monitoring Tools For Mobile Apps. Viitattu 17.4.2024. https://bluewhaleapps.com/blog/importance-of-apm-tools-for-mobile-apps

Kananen, J. 2015. Opinnäytetyön kirjoittajan opas: Näin kirjoitan opinnäytetyön tai pro gradun alusta loppuun. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. 2017. Laadullinen tutkimus pro graduna ja opinnäytetyönä. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kerner, S.M. Päivitetty elokuussa 2022. real user monitoring (RUM). TechTarget. Viitattu 13.3.2024. https://www.techtarget.com/searchitoperations/definition/real-user-monitoring-RUM

Kirvan, P., Barney, N. & Gillis, A. Päivitetty tammikuussa 2024. Amazon Web Services (AWS). TechTarget. Viitattu 3.3.2024. https://www.techtarget.com/searchaws/definition/Amazon-Web-Services

Knupfer, F. Päivitetty 18.1.2024. What is real user monitoring (RUM)? New Relic. Viitattu 15.3.2024. https://newrelic.com/blog/best-practices/what-is-real-user-monitoring

Mell, P. & Grance, T. 2011. The NIST Definition of Cloud Computing. NIST Special Publications. Viitattu 20.2.2024. https://www.govinfo.gov/app/details/GOVPUB-C13-74cdc274b1109a7e1ead7185dfec2ada

Mihajlija, M & Walton, P. Päivitetty 2023. Cumulative Layout Shift (CLS). Web.dev. Viitattu 30.4.2024. https://web.dev/articles/cls

O'Donnell, J. N.d. Observability vs. APM: What to Know on Your Monitoring Journey. Logz.io. Viitattu 5.4.2024. https://logz.io/blog/observability-vs-apm/

Osmani, A & Djirdeh, H. Päivitetty 2022. Optimize First Input Delay. Web.dev. Viitattu 30.4.2024. https://web.dev/articles/optimize-fid

Send custom events. N.d. Amazon Web Services dokumentaatio. Viitattu 8.5.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-custom-events.html

Skillwell Oy: taloustiedot. 2022. Finder. Viitattu 16.4.2024. https://www.finder.fi/IT-konsultointi+IT-palvelut/Skillwell+Oy/Jyv%C3%A4skyl%C3%A4/yhteystiedot/3232195

Skillwell. N.d. Skillwellin Tietoa meistä-sivu. Viitattu 16.4.2024. https://www.skillwell.fi/fi/tietoa-meista

Step 2: Create an app monitor. N.d. Amazon Web Services dokumentaatio. Viitattu 25.4.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-get-started-create-app-monitor.html

Use CloudWatch RUM. N.d. Amazon Web Services blogi-sivusto. Viitattu 12.2.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM.html

Using identity pools (federated identities). N.d. Amazon Web Services dokumentaatio. Viitattu 25.4.2024. https://docs.aws.amazon.com/cognito/latest/developerguide/identity-pools.html

Walton, P. Päivitetty 2023. Web Vitals. Web.dev. Viitattu 30.4.2024. https://web.dev/articles/vitals

Wittig, A. & Wittig, M. 2023. Amazon Web Services In Action, Third Edition: An In-Depth Guide To AWS. Manning Publications. Viitattu 14.2.2024. https://scholar.google.fi

What is Amazon CloudWatch? N.d. Amazon Web Services blogi-sivusto. Viitattu 12.2.2024. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html

What is APM (Application Performance Monitoring)? N.d. Amazon Web Services verkkosivut. Viitattu 29.3.2024. https://aws.amazon.com/what-is/application-performance-monitoring/

What is application performance management (APM)? N.d. IBM verkkosivut. Viitattu 29.3.2024. https://www.ibm.com/topics/application-performance-management

What is HTTP? N.d. Cloudflare. Viitattu 22.4.2024. https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/

What Is Real User Monitoring? How It Works, Examples, Best Practices, and More. 2023. Stackify. Viitattu 13.3.2024. https://stackify.com/what-is-real-user-monitoring