Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

LAURI TUUMI

Google Sheets- ja Google Apps Script -alustan käyttö sovelluskehityksessä

LAVA-varastonhallintajärjestelmä

TIETOJENKÄSITTELYN TUTKINTO-OHJELMA
2022

Tekijä(t)
Sukunimi, Etunimi
Tuumi, Lauri

Julkaisun laji
Opinnäytetyö, AMK

Päivämäärä
joulukuu 2022

Sivumäärä
39

Julkaisun kieli
suomi

Julkaisun nimi
Google Sheets- ja Google Apps Script -alustan käyttö sovelluskehityksessä

Tutkinto-ohjelma
Tietojenkäsittelyn Tutkinto-ohjelma

Tämän työn aiheena oli tutkia Google Sheets -alustan soveltuvuutta ohjelmistokehitysalustana. Työ perustuu ohjelmistokehitysprojektiin, jossa yritykselle Suomen Euromaster oy on kehitetty varastonhallintajärjestelmä.

Tämän työn tavoite oli avartaa lukijan tietämystä Google Sheets alustan hyvistä ja hunoista puolista järjestelmäkehityskäytössä. Työn sisältö keskittyy tekijän kokemuksiin projektin suorittamisesta ja sen haasteista. Työssä myös kerrotaan hieman varastonhallintajärjestelmien perusrpiirteistä ja ominaisuuksista.

Työn tuloksissa pohditaan suoritetun projektin onnistumista työssä käytettyjen työkalujen ja toimintatapojen osalta. Työssä pohditaan, mitä höytyjä ja haittoja käytetyissä alustoissa on ollut.

Työn esittämän projektin mittaan käytetyistä alustoissa esiintyi huomattavia ennakoimattomia puutteita ja ongelmia. Projekti saatiin kuitenkin suoritettua ja lopputulos oli asiakkaan tarpeisiin toimiva järjestelmä, vaikka ei kuitenkaan täydellinen sellainen. Työn lopputuloksena Google Sheets -alustaa on vaikea suositella varsinaisen järjestelmän kehitykseen.

Avainsanat
Varastonhallintajärjsetelmä, Google Sheets, Järjestelmäkehitys, Laskentataulukko

Author(s) Tuumi, Lauri	Type of Publication Bachelor's thesis	Date December 2022
	Number of pages 39	Language of publication: Finnish

Title of publication
Using the Google Sheets platform for software development

Degree program
Bachelor of Business Information Systems

The subject of this thesis was to examine the feasibility of using the Google Sheets platform to develop an information system. The thesis is based on a software development project in which a warehouse management system (WMS) was developer for the corporation Suomen Euromaster oy.

The goal of this thesis was to expand the readers knowledge on the good and bad aspects of using the Google Sheets platform for developing an information system. The main focus of the thesis is on the developers experience of executing the project and the issues that have risen while doing so. The thesis also describes the developed system in enough detail for the reader to understand the basic concepts of the system and the functionality it provides.

The conclusions of the thesis consider the success of the project with regards to the tools and methods used to develop the final product. The thesis also explores the advantages and disadvantages of the tools that have been used.

Many unforeseen issues came up as the development of this project progressed. Regardless the project was completed, and it resulted in a warehouse management system that worked for the needs of the customer, even though it wasn't perfect.

Keywords
Warehouse Management System, Google Sheets, Systems Development, Spreadsheet

SISÄLLYS

1 JOHDANTO

Googlen pilvessä toimivat toimisto-ohjelmat ovat nykyään osa monen yhtiön työntekijöiden päivittäisiä työkaluja. Työkalujen kehittyesssä Google on panostanut työkaluihin kuuluvien automaatio-ominaisuuksien kehitykseen.

Jotkut inttävät, että laskentataulukko ei ole tietokanta. Tämän opinnäytetyön varastonhallintajärjestelmäprojektissa en käyttänyt Google Sheets-taulukkoa vain tietokantana vaan myös käyttöliittymänä järjestelmään, jonka toimintaa suoritetaan Googlen Apps Script –ohjelmointikehykseen (tästä eteenpäin GAS) luoduin skriptein.

Työssä esiteltävän järjestelmän tilaajana on toiminut Suomen Euromaster Oy (tästä eteenpäin Euromaster), sen kehitys konseptitaulukosta toimivaan järjestelmään on suoritettu noin kolmessa viikossa ja se on tällä hetkellä Euromasterin kokoonpanolaitoksella tuo tantokäytössä.

Tämän työn tavoite on tutkia Sheets- ja GAS-alustan toimivuutta sovelluskehityksessä. Kerron tässä opinnäytetyössä mainituilla alustoilla kehitetyn järjestelmän synnystä, kehityksestä, rakenteesta ja kehitys- sekä ylläpitotyössä kohtaamistani ongelmista.

2 OPINNÄYTETYÖN TAVOITE

Tämän opinnäytetyön tavoite on tutkia työn ohjelmistokehitysprojektin prosessia ja tuloksia ja sitä, miten Googlen valmiiden järjestelmien adaptoiminen tietojärjestelmäksi onnistui, sekä mitä etuja ja haittoja näiden järjestelmien käytöstä oli. Varastohallintajärjestelmä, jonka kehitystyö on tämän opinnäytetyön aiheena, on

laaja, vaikkakin pikaisesti kehitettyö. Järjestelmän synnystä ja rakenteista kerrottaan tässä työssä laajasti, muttei läheskään tyhjentävästi. Järjestelmän kehityksen ja rakenteiden pääpiirteiden esittelemisen jälkeen työssä syvennytään käytettyjen alustojen ja kehitystyön arvioimiseen.

3 LAVA

LAVA on varastonhallintajärjestelmä, jonka olen kehittänyt työnantajalleni Euromasterille pikaisena toimeksiantona vuoden 2021 syksynä. Järjestelmän nimi viittaa varastossa käytettäviin lavoihin, mutta on myös lyhenne sanoista *Laurin Varasto*. Järjestelmä on kehitetty Euromasterin kokoonpanolaitokselle, jossa kootaan renkaita ja vanteita pyöriksi. Renkaita, vanteita ja koottuja pyöriä varastoidaan kokoonpanolaitoksella. Vaikka järjestelmä on kehitetty kokoonpanolaitokselle, on se kuitenkin pienin muutoksin otettavissa käyttöön melkeinpä minkälaisessa varastossa.

3.1 Yleistä varastonhallintajärjestelmistä

Varastonhallintajärjestelmien — eli WMS-järjestelmien (Warehouse Management System) — tehtävä on hallita materiaalin kulkua ja varastointia varastossa. Järjestelmissä on toimintoja materiaalin vastaanottamiseen, lähettämiseen, varastointiin ja varastosta keräämiseen. WMS-järjestelmiä käytetään varaston hallinnan tehostamiseen. (Ramaa, Subramanya & Rangaswamy, 2012, 14)

Ramaa, Subramanya ja Rangaswamy (Ramaa, Subramanya & Rangaswamy, 2012, 14) esittävät, että yksi WMS-järjestelmien päätavoite on jo tietojärjestelmissä olemassa olevien tietojen uudelleen syöttämisen karsiminen. Esimerkiksi verkkokaupan varastonhallintajärjestelmän tulisi ohjata artikkelien keräämistä, paketointia ja lähettämisetä automaattisesti asiakkaiden tilausten perusteella. Varastohenkilöstön tehtäväksi jäisi vain prosessin fyysisten osuuksien suoritus ja kuittaaminen järjestelmään.

Ramaa, Subramanya ja Rangaswamy (Ramaa, Subramanya & Rangaswamy, 2012, 14) määrittelevät WMS-järjestelmille niiden kehittyneisyyden mukaan kolme tyyppiä:

- **Perus WMS** – Perus WMS tukee vain varastosaldojen ja sijaintien hallintaa. Järjestelmää käytetään pääasiassa tapahtumien rekisteröimiseen. Järjestelmän tieto on yksinkertaista ja keskittyy pääasiassa varaston tavaran kulkuun. Järjestelmässä saattaa olla tuki RF-terminaalien käyttöön.
- **Edistynyt WMS** – Yllämainitun Perus varastonhallintajärjestelmän toiminnallisuuden lisäksi edistynyt WMS pystyy suunnittelemaan resurssien käyttöä ja aktiviteetteja edistääkseen tavaroiden kulkua varastossa.
- **Monimutkainen WMS** – Yllämainitun edistyneen järjestelmän lisäksi monimutkainen järjestelmä tarjoaa vielä edistyneempiä ominaisuuksia varaston tai varastojen joukon toiminnan optimointiin. Monimutkainen WMS hallitsee ja tehostaa tavaroiden logistista ketjua laajasti ja tarjoaa tarkkaa tietoa varaston tavaroiden sijainnista ja kulusta.

Tässä työssä esiteltävä LAVA-järjestelmä sijoittuu ehdottomasti tyyppiin *perus WMS*. LAVAn toiminnallisuudet rajoittuvat tavaroiden hyllyttämiseen (varastopaikkaan sijoittamiseen) ja keräilyyn (varastopaikasta poistamiseen). Tapahtumista kerätään tiedot lokiin, jonka avulla varaston toiminnasta voidaan tehdä alkeellista analyysiä.

3.2 LAVAn alkuperä

Tarve järjestelmälle syntyi jo vuoden 2021 kesällä, kun Euromasterin kokoonpanolaitoksen varasto irrotettiin samassa rakennuksessa toimivasta keskusvarastosta. Ennen irrottamista kokoonpanolaitoksella ei varsinaisesti ollut omaa varastoa, vaan kokoonpanolaitoksen varastointia hallitsi keskusvarasto ja sen varastonhallintajärjestelmä.

Ennen irrottamista kokoonpanolaitos oli myös Euromasterin myyntijärjestelmässä keskusvaraston osa, mutta irrottamisen jälkeen siitä tuli oma toimipisteensä.

Erottamisen jälkeen kokoonpanolaitos jäi varastotietojärjestelmän kannalta niin sanotusti tyhjän päälle.

Kokoonpanolaitoksen tietojärjestelmäaukko vaati nopean paikkauksen. Google Sheets otettiin tässä kohtaa käyttöön varastonhallintaan. Sheets-alustaan tallennettuun taulukkoon oli otettu keskusvaraston järjestelmästä raportti kokoonpanolaitoksen irrottamisen jälkeen kokoonpanolaitokselle siirtyvän varaston osuuden tilasta (paikat, tuotteet ja määrät), joten sitä lähdettiin käyttämään varaston hallitsemiseen.

Käyttöön otettua taulukkoa hallittiin täysin manuaalisesti. Tämä tarkoitti, että tuotetta paikkaan lisätessä haki varaston työntekijä taulukosta hakutoiminnolla rivin, jossa oli lisättävää tuotetta oikealla paikalla ja lisäsi käsin laskien rivin määrää lisättävällä määrällä. Mikäli riviä kyseiselle tuotteelle ja paikalle ei löytynyt, niin sellainen piti täysin käsin kirjoittaen lisätä. Kuten lukija voi kuvitella, niin edellä mainittu toimintatapa oli kaukana tehokkaasta ja erittäin virhealtis.

3.2.1 Vaatimukset tietojärjestelmälle

Kokoonpanolaitoksen tietojärjestelmävaatimukset eivät olleet suuria. Järjestelmän hallinnoima varasto ei myöskään ole suuri. Järjestelmän tulisi kyetä varastoimaan tuotteita varastopaikoille niin, että yhdellä varastopaikalla voi olla useampaa tuotetta samaan aikaan. Tuotteita pitäisi pystyä hyllyttämään ja keräilemään helpolla käyttöliittymällä. Järjestelmällä tehdyt tapahtumat pitäisi myös kirjata, sillä tapahtumatietoja tarvittaisiin varaston operaattorin laskutuksen laskemiseksi.

3.2.2 Järjestelmän valinta

Kokoonpanolaitoksen tietojärjestelmäkehityksen polkua suunnitellessa päädyttiin kahteen mahdolliseen vaihtoehtoon:

1. keskusvarastossa käytössä olevan varastonhallintajärjestelmän lisensointi ja käyttöönotto kokoonpanolaitoksella
2. Sheets-taulukon jatkokehittäminen pienimuotoiseksi varastonhallintajärjestelmäksi käyttäen GAS-skriptejä.

Keskusvaraston käyttämän varastonhallintajärjestelmän lisensointi ja käyttöönotto olisi ollut yksinkertainen päätös, mutta iso projekti. Isoksi projektiksi se olisi kehkeytynyt siitä syystä, että keskusvaraston järjestelmän käyttöönotto olisi vaatinut huomattavasti uudelleentoteutettua integraatiota myyntijärjestelmään.

Integraatioiden toteuttamisen kustannukset ja kehitysaika olisivat olleet liiallisia kokoonpanolaitoksen suhteessa pienen varaston tarpeiden ja järjestelmän tuottaman hyödyn näkökulmasta, vaikka keskusvarastossa käytössä oleva järjestelmä olisikin kaikki kookoonpanolaitoksella tarvittavan järjestelmän vaatimuksen täyttänytkin. Tietojärjestelmä haluttiin kokoonpanolaitokselle käyttöön mahdollisimman pian ja vähin kustannuksin, joten taulukon jatkokehittämiseen tyydyttiin ja LAVA-järjestelmän kehitys alkoi.

3.3 LAVAn kehitys

LAVA-järjestelmän kehitys lähti toden teolla käyntiin lokakuun alussa 2021. Kehityksen päätavoite oli, että järjestelmä saataisiin kokoonpanolaitoksella käyttöön marraskuun 2021 alusta lähtien, eli alle kuukauden sisällä. Kehityksen tahti oli siis ripeä, jopa erittäin ripeä normaalin tietojärjestelmän kehityksen aikamittoihin verrattuna.

Kehitin järjestelmää yksin, kuitenkin loppukäyttäjiltä palautetta pyytäen, esimerkiksi käyttöliittymän suunnittelussa. Järjestelmän kehitystä helpotti GAS-alustan hyvä dokumentaatio ja laaja valikoima kysymyksiä ja vastauksia ongelmiin liittyen Googlen foorumeilla ja Stack Overflow -palvelussa.

Tietorakenteiden luominen ja käyttöliittymän jäsenteleminen sujui jouhevasti. Järjestelmän toimintojen ohjelmoiminen GAS-ohjelmistokehykseen oli myös sujuvaa. Järjestelmän ensimmäisissä versioissa, joissa skriptit olivat toimivia, niiden suoritus oli hidasta. Skriptien suorituskyvystä ja muista Sheets- ja GAS-alustan kipukohdista kerrotaan omassa luvussaan tarkemmin.

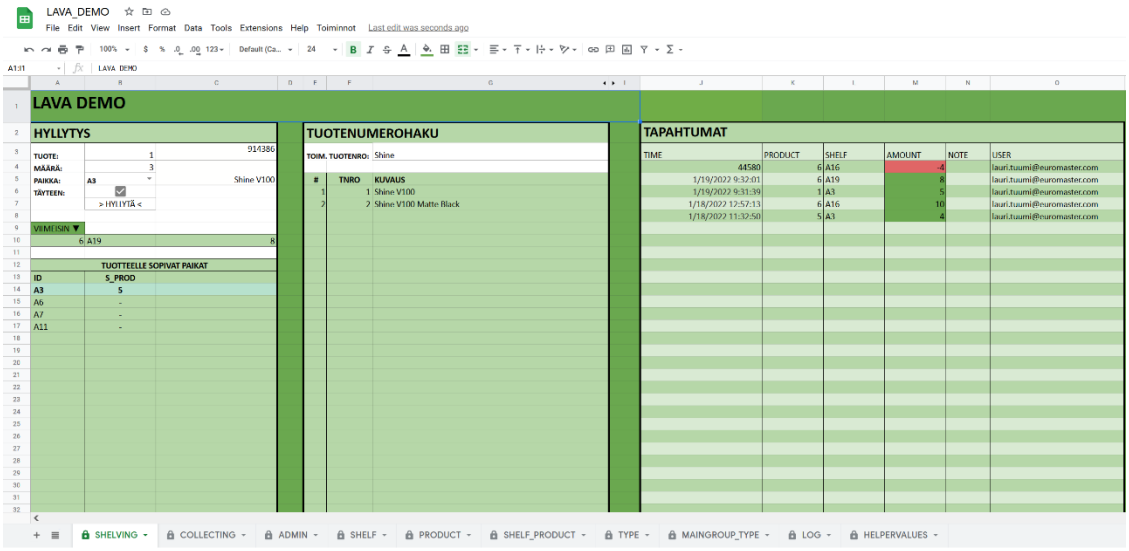
4 LAVAN RAKENNE

LAVAn taulukosto on jaettu moneen yksittäisen Sheets-taulukon välilehteen. Välilehdet voidaan jakaa kahteen ryhmään: *tauluvälilehdet* ja *käyttöliittymävälilehdet*. Tauluvälilehtiin tallennetaan dataa ja käyttöliittymävälilehdissä annetaan ohjelmalle parametreja ja käynnistetään toimintoja. Näiden lisäksi on myös välilehti *HELPERVALUES*, jossa ei ole dataa eikä käyttöliittymää, vaan viitteitä ja kaavoja, joita käytetään ohjelman toimintojen skripteissä.

Ohjelman loppukäyttäjille on sallittu vain ja ainoastaan käyttöliittymien käyttöön tarvittavien solujen muuttaminen. Rajoituksilla on pyritty varmistamaan, etteivät loppukäyttäjät esimerkiksi vahingossa ylikirjoita kaavoja tai tule muuttaneeksi järjestelmään tallennettua dataa.

4.1 Hyllytyskäyttöliittymä

SHELVING-välilehdellä suoritetaan hyllytyksiä järjestelmään, jotta saadaan lisättyä tietyn tuotteen määrää tietyllä hyllyllä. Hyllytyksen tiedot syötetään välilehden vasemmassa laidassa olevan otsikon *HYLLYTYS* alla oleviin tyhjiin soluihin (kuva 1).



Kuva 1. Kuvakaappaus LAVAn hyllytyskäyttöliittymästä

Käyttäjän syötettyä tuotenumeron näyttää ohjelma alla olevassa *TUOTTEELLE SOPIVAT PAIKAT* -listassa paikkakoodeja, johon annettua tuotetta saa hyllyttää. Listassa näkyy myös, onko ehdotetussa paikassa jo olemassa kyseistä tuotetta. Käyttäjän pitää alla olevasta listasta valita jokin paikka, johon tuotetta hyllyttää.

Mikäli syötetty tuotenumero löytyy järjestelmästä, niin hyllytyksen syötekenttien oikealle puolelle tuodaan *PRODUCT*-välilehdeltä syötetyn tuotteen toimittajan tuotenumero näkyviin. Toimittajan tuotenumeron alapuolelle tulee myös tuotteen tekstikuvaus.

Kohtaan *TÄYTEEN* käyttäjä merkitsee täpän, mikäli kohdepaikka tulee täyteen, eikä siihen pysty hyllyttämään enää lisää tuotetta. Hyllytyksen skripti merkitsee tällöin kyseisen paikan *SHELF*-välilehteen (sarakkeeseen *FULL*) täynnä olevaksi. Täynnä olevaan paikkaan ei voida hyllyttää tuotetta.

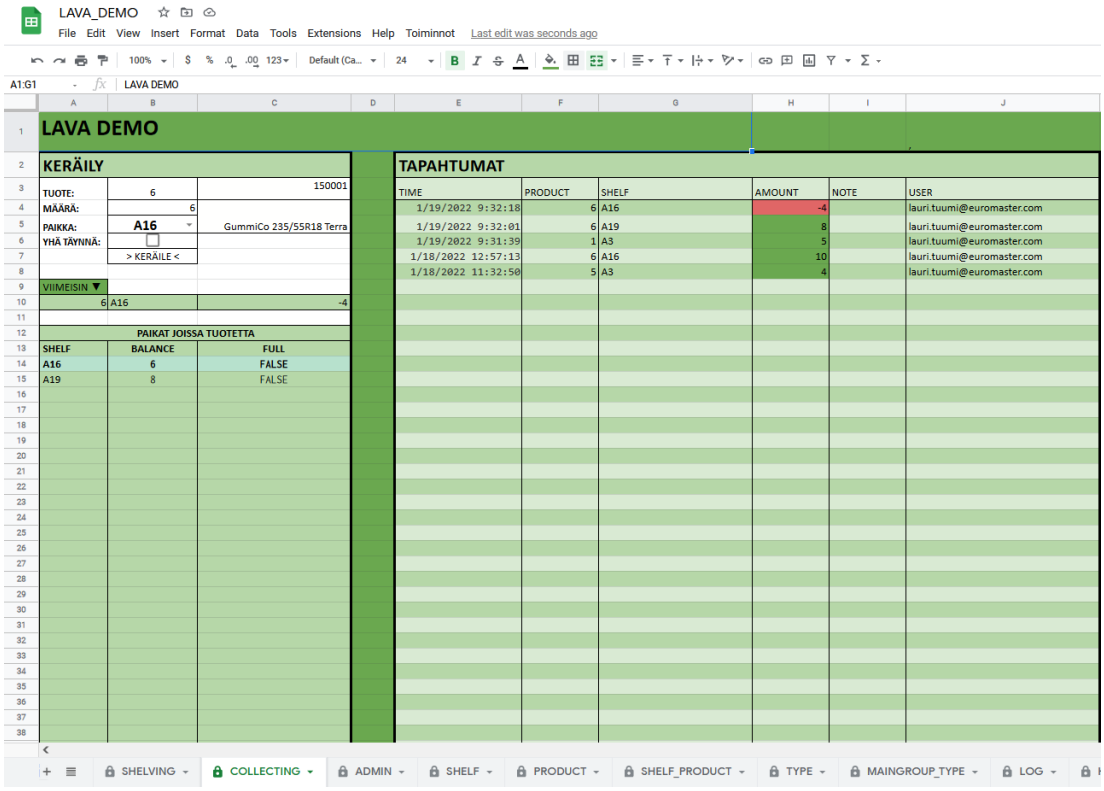
Aktivoidakseen hyllytystapahtuman tulee käyttäjän muokata solun B7 arvoa, jossa ennestään lukee “> *HYLLYTÄ* <”. Muutoksen jälkeen skripti lisää syötetylle paikalle syötettyä tuotetta syötetyn määrän verran ja asettaa paikan täydeksi, mikäli käyttäjä on *TÄYTEEN*-valintaan asettanut täpän. Tapahtumasta tallentuu tieto lokiin välilehdelle *LOG*.

HYLLYTYS-osion oikealla puolella on osio *TUOTENUMEROHAKU*. Tämän osion avulla käyttäjä voi tuotteen toimittajan tuotenumeron tai kuvauksen perusteella hakea tuotteita tuoterekisteristä *(PRODUCT-välilehti).* Tämä toiminto on suunniteltu tilannetta varten, jossa hyllyttäjällä on vain toimituksen pakkausluettelo. Pakkausluetteloissa on kuvauksia ja toimittajan tuotenumeroita, mutta LAVAn käyttämää tuotenumeroa ei niistä löydy.

Hyllyttäjä voi syöttää *TOIM. TUOTENRO* -kenttään joko osan tai koko toimittajan tuotenumero tai tuotteen kuvauksen osan tai tuotteen koko kuvauksen. Alapuolelle ilmestyy listaan haun löytämät tuotteet. Haetun tuotteen LAVAn tuotenumeron voi hyllytyksen *TUOTE*-kenttään hakea syöttämällä haetun tuotteen järjestysnumeron alla olevasta listasta ja pilkun perään (esim. *5,*) *TUOTE*-kenttään. Tuotenumeron ilmestyttyä *TUOTE*-kenttään hyllyttäjä voi suorittaa hyllytyksen loppuun normaaliin tapaan.

4.2 Keräilykäyttöliittymä

COLLECTING-välilehdellä suoritetaan keräilyjä (kuva 2). Keräilykäyttöliittymä toimii lähes identtisesti hyllytyskäyttöliittymän kanssa, mutta vain päinvastoin. Tuotteen määrän lisäämisen sijaan tuotetta vähennetään tietyltä hyllyltä.



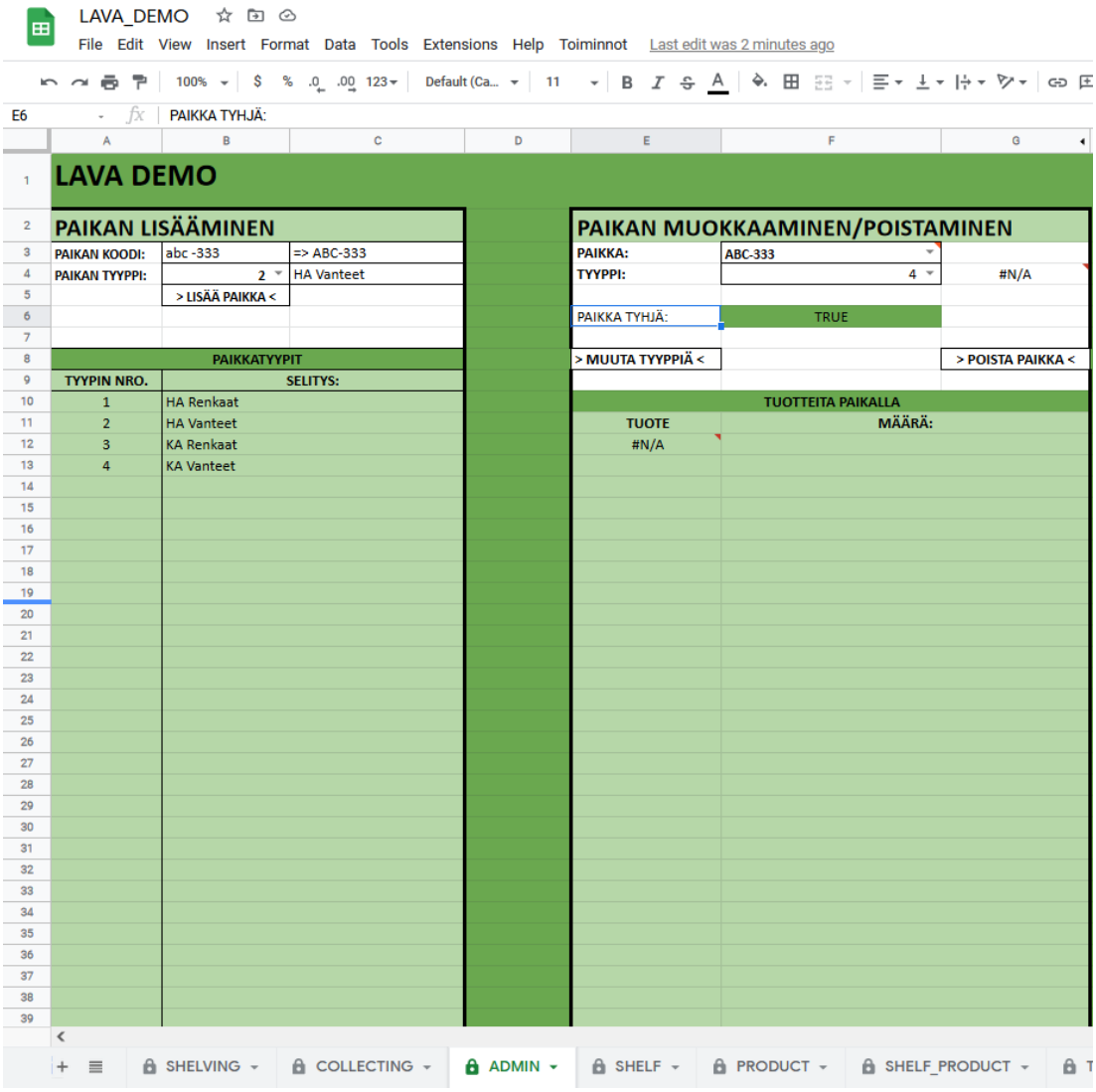
Kuva 2. Kuvakaappaus LAVAn keräilykäyttöliittymästä

Hyllyttämisestä poiketen keräilykäyttöliittymässä on täppä *YHÄ TÄYNNÄ*. Käyttäjän kuuluu asettaa tähän täppä, mikäli keräiltävä paikka on yhä täynnä, vaikka siitä keräillään tuotetta. Jos täppää ei laiteta, niin keräily asettaa aina paikan tilan ei-täydeksi.

4.3 Hallintakäyttöliittymä

ADMIN-välilehdeltä suoritetaan paikkojen hallinnan toimintoja (kuva 3). Paikoille voidaan suorittaa kolmenlaista eri toimintoa:

- Voidaan lisätä paikka.
- Paikka voidaan poistaa.
- Paikan tuotetyyppi voidaan muuttaa.



Kuva 3. Kuvakaappaus LAVAn hallintakäyttöliittymästä

Paikan lisääminen suoritetaan välilehden vasemmanpuoleisesta osiosta antamalla paikalle koodi (esim. *ABC-123*) ja tyyppi, jonka jälkeen lisäämisen skripti laukaistaan muokkaamalla solua, jossa lukee "> *LISÄÄ PAIKKA* <". Paikan lisäämisen alapuolella näkyy listaus paikkojen tyyppien koodeista ja niiden selityksistä.

Paikan koodin kenttään voi syöttää välilyöntejä ja pieniä kirjaimia, mutta koodi muutetaan aina merkkijonoksi, jossa on vain isoja kirjaimia, eikä lainkaan välilyöntejä. Ainoa rajoitus paikan lisäämiselle on, että kyseisellä koodilla ei saa olla jo olemassa olevaa paikkaa.

Paikan muokkaaminen ja poistaminen suoritetaan lisäämiskäyttöliittymän oikealla puolella olevasta osiosta. Paikan poistaminen ja tyypin muuttaminen molemmat vaativat, että paikalla ei ole yhtään tuotetta.

Paikan koodin valittua näkyy osion alapuolella listassa tuotteet ja määrät, joita kyseisellä paikalla on. Tämä on lisätty helpottamaan paikan tyhjentämistä, sillä keräilykäyttöliittymästä ei pysty valitsemaan paikkaa ja näkemään, mitä tuotteita valitulla paikalla on.

Paikan poistamista varten pitää valita paikka, varmistaa että se on tyhjä ja laukaista paikan poisto muokkaamalla solua, jossa lukee "> *POISTA PAIKKA* <". Paikan tyypin muokkausta varten pitää paikan valitsemisen lisäksi valita uusi tyypin koodi paikan valinnan alapuolelta, jonka jälkeen muokkaus aktivoidaan muokkaamalla solua, jossa lukee "> *MUUTA TYYPPIÄ* <".

Hallintakäyttöliittymän toiminnot on suunniteltu yksinkertaisiksi, jotta järjestelmää käyttävä varastotyöntekijä voisi itsenäisesti lisätä ja muokata paikkoja ilman pääkäyttäjän apua.

4.4 SHELF-taulukko

Välilehti nimeltä *SHELF* toimii LAVA-järjestelmän hyllypaikkarekisterinä (taulukko 1). Välilehdellä on kolmella sarakkeella atomisia arvoja (Arvoja, jotka ovat raakaa dataa, eivätkä esimerkiksi kaavan luomaa tai muualta tuomaa dataa):

***ID*-**sarakkeessa on paikan koodi. Koodi voi käytännössä olla mikä tahansa isoilla kirjaimilla kirjoitettu välilyöntimerkitön merkkijono. ID toimii hyllytaulussa kuten relaatiotietokannan taulun avain.
***FULL*-**sarakkeeseen tallennetaan totuusarvomuuttuja, joka kerto, onko paikka täynnä vai ei.
***TYPE*-**sarake määrittää, minkä tyyppisille tuotteille rivin hyllypaikka on tarkoitettu. Sarake 'viittaa' *välilehden* TYPE *sarakkeeseen* TYPE.

Taulukko 1. Ote *SHELF*-välilehdeltä (atomiset arvot maalattu vihreällä)

ID	FULL	TYPE	S_PROD
A1	FALSE	1	-
A2	TRUE	4	-
A3	FALSE	2	5
A4	TRUE	3	-

Sarake ***S_PROD*** sisältää kaavan, joka hakee nykyisistä varaston saldoista hyllytyskäyttöliittymään syötetyn tuotteen saldon kyseisellä rivillä. Mikäli hyllytyskäyttöliittymään on syötetty esimerkiksi tuote 123, niin *S_PROD* sarake näyttää tuotteen 123 saldot kullakin paikalla, jossa tuotetta on.

4.5 PRODUCT-taulukko

Välilehti nimeltä *PRODUCT* sisältää LAVA-järjestelmän tuoterekisterin (taulukko 2). Välilehdellä on viidellä sarakkeella atomisia arvoja:

- ***PRODUCT***-sarakkeessa on tuotteen numero. Tuotteen numeron tulee olla kokonaisluku ja se toimii tuotetaulussa kuten relaatiotietokannan taulun avain.
- ***SUPPLIER_ID***-sarakkeessa on tuotteen toimittajan tuotenumero, joka on merkkijono.
- ***MAINGROUP_ID***-sarakkeessa on tuotteen pääryhmän koodi kokonaislukuna tallennettuna. Myyntijärjestelmässä, josta taulu on otettu, määrää pääryhmä tuotteen kategorian, kuten esrimerkiksi henkilöautonrengas tai TPMS-tarvike.
- ***SUBGROUP_ID***-sarakkeessa on tuotteen alaryhmän koodi kokonaislukuna tallennettuna. Järjestelmässä, josta taulu on otettu, määrää alaryhmä tuotteen valmistajan, kuten Michelin tai Nokian renkaat.
- ***DESCRIPTION***-sarakkeessa on tuotteen sanallinen kuvaus merkkijonona.

Taulukko 2. Ote *PRODUCT*-välilehdeltä (atomiset arvot maalattu vihreällä)

PRODUCT	SUPPLIER_ID	MAINGROUP_ID	SUBGROUP_ID	DESCRIPTION	TYPE
1	914386	150	439	Shine V100	2
2	459719	150	439	Shine V100 Matte Black	2
3	374315	163	439	Fälgar CC1	4
4	122672	162	439	Fälgar CC5	4

Sarakkeeseen ***TYPE*** haetaan pystyhakukaavalla (*VLOOKUP*) *MAINGROUP_ID* -sarakkeen arvolla *MAINGROUP_TYPE* -välilehdeltä tuotteen tyypin numeron. Eli tuotteen pääryhmä määrää tuotteen tyypin ja pääryhmien tyypit löytyvät välilehdeltä *MAINGROUP_TYPE*.

Ohjelman perustoimintojen, eli hyllytyksen ja keräilyn kannalta ainoat välttämättömät sarakkeet ovat *PRODUCT* ja *TYPE*. *SUPPLIER_ID* ja *DESCRIPTION* vain tarjoavat käyttäjälle lisätietoja tuotteista ja mahdollistavat tuotteiden helpomman etsimisen hyllytyskäyttöliittymässä.

4.6 SHELF_PRODUCT-taulukko

Kuten nimestä voi päätellä, *SHELF_PRODUCT*-välilehti yhdistää hyllypaikkoja ja tuotteita ja täten tallentavat varaston tilaa (taulukko 3). Välilehden sarakkeet *SHELF* ja *PRODUCT* toimivat taulussa kuten relaatiotietokannan yhdistelmäavain, jossa *SHELF* viittaa *SHELF*-välilehden avaimeen ja *PRODUCT* viittaa *PRODUCT*-välilehden avaimeen. *SHELF* ja *PRODUCT* määräävät, että rivin tuotetta on rivin paikalla. Tuotteen ja paikan yhdistelmän (*ID*) on oltava uniikki.

Taulukko 3. Ote *SHELF_PRODUCT*-taulukosta (atomiset arvot maalattu vihreällä)

SHELF	PRODUCT	BALANCE	DESCRIPTION	ID	FULL
A1	3	10	Fälgar CC1	A1^3	FALSE
A3	5	4	GummiCo 225/75R15 SupraGrip	A3^5	FALSE
A16	6	6	GummiCo 235/55R18 Terra	A16^6	FALSE
A3	1	5	Shine V100	A3^1	FALSE

Ainoa välilehden ‘oma’ atominen arvo on *BALANCE*, johon tallennetaan kyseisen paikkatuoteyhdistelmän tuotteiden määrä. Hyllyttäessä *BALANCE*-sarakkeen arvoa lisätään ja keräillessä sarakkeen arvoa vähennetään. *BALANCE*-sarakkeeseen ei tallenneta lainkaan nolla-arvoja, vaan hyllypaikan tullessa tyhjäksi poistetaan koko rivi taulusta. Samaa periaatetta noudattaen, puuttuvaan paikkatuoteyhdistelmään hyllyttäessä luodaan kokonaan uusi rivi.

Sarakkeen *DESCRIPTION* arvo haetaan *PRODUCT*-välilehdeltä yksinkertaisesti pystyhaulla rivin *PRODUCT*-sarakkeen arvon avulla. Tämä sarake ei tue mitään varsinaista toimintaa, mutta se helpottaa käyttäjää hahmottamaan nopeammin, mitä tuotetta milläkin rivillä on.

ID-sarake sisältää paikkatuoteyhdistelmän ‘yhdistelmäavaimen’ arvon. Arvo koostuu sarakkeen *SHELF* arvosta, jonka loppuun on lisätty sirkumfleksi (^) ja *PRODUCT*-sarakkeen arvo. Mikäli Sheets-alustalla pystyisi suorittamaan pystyhakuja käyttämällä useampaa arvoa avaimena, niin tätä saraketta ei tarvittaisi. Ominaisuuden puuttuessa on tämä sarake pakko olla, jotta *SHELF*- ja *PRODUCT*-sarakkeiden arvojen yhdistelmällä voidaan hakea välilehden riviä, kuten relaatiotietokannassa yhdistelmäavaimella.

Käyttäjän hyllyttäessä tai keräillessä etsii järjestelmä tuotteen ja paikan yhdistelmän rivin *ID*-sarakkeen avulla käyttäen sisäänrakennettua *MATCH*-kaavaa. Jos hyllyttäessä riviä ei löydy, se tarkoittaa, että riviä ei ole olemassa, joten on luotava uusi.

FULL-sarakkeen arvo haetaan *SHELF*-välilehdeltä pystyhakukaavalla *PRODUCT*-sarakkeen arvoa avaimena käyttäen. *FULL*-sarakkeen arvo on tuotu *SHELF_PRODUCT*-välilehteen, jotta sen avulla voidaan keräilykäyttöliittymässä näyttää *PAIKAT JOSSA TUOTETTA* -listassa, jotta käyttäjä näkee paikan tilan keräilyä suunnitellessaan. Arvon voisi hakea keräilykäyttöliittymässä erikseen pystyhakukaavalla, mutta arvo on helpompi hakea SHELF-välilehden riville jo valmiiksi.

4.7 TYPE-taulukko

TYPE-välilehden tarkoitus on tallettaa sanalliset kuvaukset kaikista käytössä olevista tuotetyypeistä (taulukko 4). Välilehti ei itsessään ole rekisteri tyypeistä, sillä sen *TYPE*-sarakkeen arvot haetaan *MAINGROUP_TYPE*-välilehdestä etsimällä sen *TYPE*-sarakkeen kaikki nollaa isommat uniikit arvot. Sarakkeeseen *DESCRIPTION*

pääkäyttäjä syöttää tyypin sanallisen kuvauksen. *TYPE*-välilehden arvoja käytetään toistaiseksi ainoastaan hallintakäyttöliittymässä, jotta loppukäyttäjä pääsee näkemään, mitä tuotteiden tyyppikoodit tarkoittavat.

Taulukko 4. Ote *TYPE*-taulukosta (atomiset arvot maalattu vihreällä)

TYPE	DESCRIPTION
1	HA Renkaat
2	HA Vanteet
3	KA Renkaat
4	KA Vanteet

4.8 MAINGROUP_TYPE-taulukko

MAINGROUP_TYPE-välilehti yhdistää tuotteiden pääryhmät LAVAn käyttämiin tyyppikoodeihin (taulukko 5). LAVAn suunnittelussa olisi voitu päättää käytettävän myyntijärjestelmän käyttämiä pääryhmiä sellaisenaan. Esteenä tälle oli kuitenkin se, että myyntijärjestelmässä esimerkiksi henkilöautorenkaita on kuudessa eri ryhmässä. LAVAn hallinnoimassa varastossa niitä kaikkia pitäisi pystyä sijoittamaan samoille paikoille, joten piti kehittää pääryhmiä yhdistävä tekijä, eli tyyppi.

Taulukko 5. Ote *MAINGROUP_TYPE*-taulukosta (atomiset arvot maalattu vihreällä)

MAINGROUP	MG DESCRIPTION	TYPE
10	HA Kesä	1
20	HA Talvi	1
50	KA	3
55	KA pin.	3

MAINGROUP_TYPE-välilehden listaus on tuotu myyntijärjestelmästä suoraan. *MAINGROUP*-sarakkeessa on pääryhmän numerokoodi. *MG_DESCRIPTION*-sarake sisältää pääryhmän sanallisen kuvauksen. Kuvausta ei LAVAssa varsinaisesti käytetä missään, mutta se helpottaa hahmottamaan pääryhmien tarkoitusta.

TYPE-sarakkeen arvo määrittelee kunkin pääryhmän tyypin. *TYPE*-sarakkeessa voi myös olla arvo 0, joka tarkoittaa, että kyseisen pääryhmän tuotteita ei tämän LAVA-instanssin varastoon kuuluisi tulla. *TYPE*-sarakkeen 0 ei kuitenkaan tarkoita, että

kyseisen pääryhmän tuotteita ei saisi hyllytettyä järjestelmään. Sen sijaan tyypin 0 pääryhmän tuotteita voi hyllyttää mihin tahansa paikkaan, mutta niille ei ole mitään määrättyjä paikkoja kyseisessä varastossa.

4.9 LOG-taulukko

LOG-välilehteen tallentuu merkintä kaikista järjestelmän keräily- ja hyllytystapahtumista (taulukko 6). Välilehden sarakkeet ovat:

- ***TIME***, johon tallentuu tapahtuman suoritusaika
- ***PRODUCT***, johon tallentuu tapahtumassa käytetyn tuotteen koodi
- ***SHELF***, johon tallentuu tapahtumassa käytetty hyllypaikka
- ***AMOUNT***, johon tallentuu tapahtumassa käytetty tuotteen määrä. Sarakkeen arvon positiivisuus tai negatiivisuus kertoo, onko kyseessä hyllytys- vai keräilytapahtuma
- ***NOTE***, johon tallentuu huomioita tapahtumasta. Esimerkiksi hyllytyksen asettaessa paikan täydeksi tallentuu tapahtuman lokimerkinnän NOTE-sarakkeeseen merkintä [T]. (T = Täyteen)
- ***USER***, johon tallentuu tapahtuman tekijän sähköpostiosoite.

Taulukko 6. Ote *LOG*-taulukosta (atomiset arvot maalattu vihreällä). *TIME*-sarakkeessa näkyy varsinaisessa käyttöliittymässä aika muodossa ‘kk/pp/vvvv t:m:s’

TIME	PRODUCT	SHELF	AMOUNT	NOTE	USER
44579	5	A3	4		lauri.tuumi@euromaster.com
44580	6	A16	10	[EL][T]	lauri.tuumi@euromaster.com
44580	1	A3	5		lauri.tuumi@euromaster.com
44580	6	A19	8	[EL]	lauri.tuumi@euromaster.com

NOTE-saraketta on ajateltu käytettävän niin, että siihen voidaan lisätä erinäisiä hakasulkujen ympäröimiä koodeja, kuten [T] ja [EL], jotka merkitsevät jotain tapahtuman ominaisuutta. Koodien avulla voidaan Sheetsin QUERY-kyselykaavan predikaattien LIKE-avainsanan avulla kyselemään lokimerkintöjä jonkin tai useamman koodin läsnäolon perusteella.

Tapahtuman tekijän sähköpostiosoite tallentuu lokimerkintöihin ainoastaan, jos käyttäjä on osa samaa Google Workspaces -organisaatiota. Mikäli käyttäjä ei ole osa samaa organisaatiota, esimerkiksi normaali ei-kaupallinen Gmail-tunnus, niin lokin *USER*-sarake jää valitettavasti tyhjäksi. (Google 2021)

4.10 HELPERVALUES-välilehti

HELPERVALUES-välilehteen on koottu kaikki toimintojen skriptien vaatimat arvot (taulukko 7). A-sarakkeessa on arvojen nimet ja B-sarakkeessa on itse arvot. D-sarakkeessa on vielä lisätietoa arvon merkityksestä.

Taulukko 7. Ote HELPERVALUES-välilehden hyllytyksen arvojen osiosta. Taulukon arvot esittävät tilaa, jonka kuvan 1 syötteet ovat saaneet aikaan. Vain sarakkeet A, B ja D on näytetty

SHELVING		
S_PROD	1	HYLLYTYKSEEN SYÖTETTY TUOTE
S_AMOUNT	3	HYLLYTYKSEEN ANNETTU MÄÄRÄ
S_SHELF	A3	HYLLYTYKSEEN ANNETTU HYLLY
S_MAKE_FULL	TRUE	HYLLYTYKSEEN ANNETTU HYLLYN TÄYTTYMINEN
S_PROD_TYPE	2	HYLLYTYKSEEN ANNETUN TUOTTEEN TYYPPI (-1 JOS TUOTETTA EI LÖYDY TUOTEREKISTERISTÄ)
S_STO_CODE	A3^1	HYLLYTYKSEEN ANNETUN PAIKAN JA TUOTTEEN YHDISTELMÄN KOODI ('**PAIKKA^TUOTE**')
S_STO_ROW_NUM	5	HYLLYTYKSEEN ANNETUN PAIKKATUOTEYHDISTELMÄN RIVINUMERO *SHELF_PRODUCT*-VÄLILEHDELLÄ
S_CURR_BAL	5	HYLLYTYKSEEN ANNETUN PAIKKATUOTEYHDISTELMÄN SALDO *SHELF_PRODUCT*-VÄLILEHDELLÄ
S_NEW_BAL	8	HYLLYTYKSEEN ANNETUN PAIKKATUOTEYHDISTELMÄN UUSI SALDO, KUN YLLÄ OLEVAAN SALDOON LISÄTÄÄN HYLLYTYKSEEN ANNETTU MÄÄRÄ
S_SHELF_ERR	FALSE	TARKISTUS, ONKO HYLLYTYKSEEN ANNETTU HYLLY OIKEASTI HYLLYTYSKÄYTTÖLIITTYMÄN HYLLYLISTASSA (FALSE = EI VIRHETTÄ)
S_SHELF_ROW_NUM	4	HYLLYTYKSEEN ANNETUN HYLLYPAIKAN RIVIN NUMERO *SHELF*-VÄLILEHDELLÄ (KÄYTETÄÄN HYLLYN TÄYDEKSI MERKITSEMISEEN)
S_BUTTON_VALUE	> HYLLYTÄ <	HYLLYTYKSEN LAUKAISEVAN KENTÄN ARVO LAUKAISEMISEN AIKAAN
S_ACTIVE	FALSE	TARKISTUS, ONKO HYLLYTYSOPERAATIO KÄYNNISSÄ (KERÄILY KESKEYTYY, MIKÄLI SEN ALKAESSA HYLLYTYS ON KESKEN)

Arvoille on annettu A-sarakkeessa nimet, sillä ne ladataan toimintojen käynnistyessä skripteissä JavaScriptin Dictionary-tyyppiseen muuttujaan. Dictionary-tyyppiset muuttujat tallentavat avain-arvopareja, joiden arvoa voi hakea avaimen (nimen) avulla (Flexiple 2022).

Arvojen nimeäminen ja sijoittaminen Dictionary-tyyppiseen muuttujaan mahdollistaa vapaan HELPERVALUES-välilehteen rivien lisäämisen ja sen rivien uudelleenjärjestämisen ilman, että skriptien koodia tarvitsee muuttaa (vrt. kovien viitteiden käyttö). Välilehden kaavat on myös kehitetty niin, että samalta välilehdeltä arvoja käyttävät kaavat hakevat arvot pystyhakua käyttämällä, jolloin rivejä voi vapaasti uudelleenjärjestellä ja lisätä rikkomatta kaavoja.

Välilehden sisältämien lihavoitujen otsikoiden alla on kaikki kyseisen otsikon toimintaan liittyvät arvot. Arvot voivat joko olla viitteitä käyttöliittymävälilehden soluihin, tai kaavoja, joissa käyttöliittymien arvoja on käytetty.

5 LAVAN OHJELMAKOODI

LAVAn ohjelmakoodi on kirjoitettu JavaScript-kielellä. Hyllyttämisen ja keräilemisen funktiot toimivat kutakuinkin seuraavalla tavalla:

1. Ladataan *HELPERVALUES*-välilehdestä arvot ja sijoitetaan ne muuttujiin.
2. Tarkistetaan, että kyseisen toiminnon käyttämät arvot eivät ole virheellisiä. Virheellisten arvojen (tai virhetilan) löytyessä näytetään käyttäjälle virheilmoitus ja lopetetaan toiminnon suoritus.
3. Suoritetaan toiminto tietyin ehdollisuuksin. Esimerkiksi: Mikäli hyllyttäessä paikkatuoteyhdistelmää ei löydy, luodaan *SHELF_PRODUCT*-tauluun uusi rivi ja asetetaan sen määrä käyttäjän syöttämäksi määräksi.

LAVAssa on myös muita funktiota, kuten hyllypaikan poistaminen, muokkaaminen ja lisääminen. Kaikki funktiot seuraavat yllä olevaa toimintaperiaatetta, hakine dataa

HELPERVALUES-välilehdeltä, tarkistaen datan ja tehden muutoksia taulukkoon datan perusteella.

Funktiot eivät suorituksen aikana hae taulukosta tietoa muualta kuin *HELPERVALUES*-välilehdeltä. Funktiot eivät myöskään tee mitään laskennallisesti vaativaa käsittelyä *HELPERVALUES*-välilehdeltä ladatuille tiedoille, sillä laskennat on suoritettu jo välilehden kaavoissa. Funktiot ovat pääasiassa ladattujen tietojen ehtolausein tarkastelemista ja tarkastelujen perusteella tiedon poistamista, sijoittamista tai muuttamista tauluihin.

Tiedon keskittäminen ja laskutoimitusten suorittaminen *HELPERVALUES*-välilehdessä oli tärkeä ratkaisu luoda, sillä kehityksen alkuvaiheessa ongelmaksi kehkeytyi se, että monista eri taulukoista datan lataaminen skrpiteihin suorituksen aikana vei erittäin paljon suoritusaikaa, joka *Apps Script* -alustalla on rajattua.

6 SHEETS- JA GAS-ALUSTAN TUOMAT VAIKEUDET SOVELLUSKEHITYKSEEN

6.1 Funktioden suoritutsaikarajoitukset

6.1.1 Ongelma

GAS-alustalla suoritettavien toimintojen suoritusaika on rajattu 30 sekuntiin per suoritus (Google, 2021). Useimmissa operatiivisista tietojärjestelmissä 30 sekuntia on pitkä aika suorittaa minkäänlaista toimintoa. GAS-alustassa suoritusaikoja pidentää se, että skriptin suorittava taho ei välttämättä Googlen infrastruktuurissa ole muokattavan taulukon ‘lähellä’.

Skriptien avulla taulukoita muokattaessa varsinaiset taulukkoa muokkaavat funktiot, kuten solun arvon muuttaminen, saavat aikaan kutsun pilvessä varastoidun taulukon rajapintaan. Nämä kutsut ovat hitaita ja niiden määrää tulisi rajoittaa. (--Hyde 2020.)

Mikäli GAS-skriptit käsittelisivät suoritusalustassaan paikallisesti sijaitsevaa dataa, laskettaisiin LAVA-järjestelmän suorittamien operaatioiden suoritusaikaa sekunnin kymmenesosissa kymmenien sekuntien sijasta.

Vaikka taulukko on Googlen pilvessä, sekä skriptit suoriutuvat myös Googlen pilvessä, rajapintaan tehdyt kutsut saattavat silti kestää useita sekunteja. Mikäli funktion muuttaman arvon muutos saa aikaan muutoksia kaavojen tuloksissa, pitää kyseisten kaavojen tulokset myös laskea uudelleen ennen kuin data voidaan vapauttaa seuraavan rajapintakyselyn käytettäväksi.

Alkuperäistä taulukkoa muuttavaa kyselyä seuraavien rajapintakyselyiden pitää odottaa kaikkien alkuperäisen muutoksen aikaansaamien kaavojen uudelleenlaskentojen tuloksia. Kaavojen uudelleenlaskenta peräkkäisten taulukon muokkaus- ja lukuoperaatioiden välillä saattaa rajusti pidentää suoritettavan funktion suoritusaikaa sen joutuessa odottamaan kaavojen jatkuvaa uudelleenlaskentaa.

LAVA-järjestelmän hyllytys- ja keräilyfunktioiden ensimmäisissä toteutuksissa funktioiden suoritusrajat ylittyivät. Suoritusten tultua pakkolopetetuksi jäi osia ohjelmakoodista suorittamatta. Pakkolopetuksen konkreettinen seuraus oli usein se, että tapahtuman lokimerkintä jäi lisäämättä *LOKI*-välilehdelle. Funktioiden suoritusrajat ylittyivät alkuperäisissä hyllytys- ja keräilyfunktioissa siksi, että useat niissä käytetyt muuttujat laskettiin taulukon eri alueilta erillisin rajapintakyselyin haetuista arvoista.

6.1.2 Ratkaisu

Hyllytys ja -keräilyfunktioiden nopeuttamiseen paras ratkaisu oli pyrkiä yhdistämään rajapintakyselyitä tekeviä funktiota niin, että rajapintakyselyitä suoritettaisiin mahdollisimman vähän.

Kyselyiden vähentäminen onnistui keräämällä mahdollisimman suuri osa funktioiden tarvitsemista arvoista erilliseen taulukon välilehteen, josta ne pystytään yhdellä

kutsulla lataamaan skriptiin hakemistomuuttujaksi. Arvot sijoitettiin taulukkoon nimeltä *HELPERVALUES*.

HELPERVALUES-taulukko sisältää kaikki keräily tai -hyllytysfunktioiden tarvitsemat laskettavat tai taulukosta haettavat arvot. Kaikki toimintojen tarvitsemat arvot saadaan välilehdeltä ladattua skriptiin yhdellä rajapintakyselyllä.

LAVAn ensimmäisissä versioissa skriptien tarvitsemia arvoja ladattiin käyttöliittymästä ja selvitettiin ohjelmallisesti esimerkiksi tekemällä koodissa hakuoperaatioita skriptiin taulukosta ladattujen isojen alueiden datasta. *HELPERVALUES*-välilehden myötä kaikki laskeminen saatiin keskitettyä välilehteen, jossa haku- ja laskuoperaatiot tapahtuvat jatkuvasti käyttäjän syöttäessä dataa (vrt. funktioiden koodissa laskeminen) ja josta kaikki arvot saadaan skriptin käynnistyttyä ladattua yhdellä rajapintakyselyllä.

6.1.3 Vaikutus alustojen käytettävyyteen ohjelmistokehityksessä

Suoritusaikarajoitukset vaikuttivat aluksi ongelmalta, joka voisi estää koko LAVA-järjestelmän kehittämisen Sheets- ja GAS-alustalla, mutta skriptien tekemien rajapintakyselyiden keskittäminen *HELPERVALUES* -välilehden avulla vähensi suoritusaikaa niin, että pakkolopetuksen riskiä ei ole. Tulevaisuudessa tosin on tärkeää pitää suoritusaikarajoitukset mielessä GAS-skriptejä kehittäessä. Skriptit, jotka lataavat tietoa Sheets-taulukoista ja mahdollisesti myös muiden rajapintojen kautta saattavat helposti kehkeytyä liian pitkiksi suorittaa ja täten olla käytännössä käyttökelvottomia.

Harmittavaa GAS-alustan suoritusaikarajoituksissa on se, että rajoitukset ovat absoluuttisia reaaliajan mukaan, eivätkä esimerkiksi skriptin käyttämän prosessorin laskenta-ajan mukaan. Skripti, joka joutuu 90% suoritusajastaan odottamaan erinäisten kyselyiden valmistumista saattaa tulla pakkolopetetuksi, vaikka suoritusajasta suurin osa on ovat vähää suoritustehoa vaativaa tulosten odottamista.

Suoritusaikarajoitukset hankaloittavat kehittämistä, mutta ovat kuitenkin ymmärrettävä osa jaettujen resurssien käyttöä. Rajoittaminen on mielestäni eritoten ymmärrettävää GAS-alustan kohdalla, sillä palvelun käyttäjää ei laskuteta skriptien pilvessä suorittamisesta erikseen, vaan kaikki suorittaminen kuuluu käyttäjän olemassa olevaan Google Workspaces –tilaukseen.

6.2 Käyttäjien ja skriptien oikeudet

6.2.1 Ongelma

Mikäli olet joskus käyttänyt tai kehittänyt Excel-taulukkoa, jossa on makron käynnistävä hiirellä painettava nappula, saattaa sinulle herätä kysymys siitä, miksi LAVAn toimintojen käynnistäminen tapahtuu tiettyjen solujen arvoa muuttamalla, eikä nappia painamalla. Syy tälle ei ole nappiominaisuuden puuttuminen Sheets-taulukoista.

Vähäisempi syy nappien käyttämättömyydelle on se, että nappeja pitää painaa tietokoneella käyttäessä hiirellä tai kosketusnäyttöä käyttäessä sormella. Sheets-taulukossa nappeja ei voi sijoittaa mitenkään niin, että ne olisivat näppäimistöllä painettavissa.

Tärkeämpi syy solun arvon muuttamisen käyttämiseen ‘nappina’ on se, että nappien suorittamat skriptit suoriutuvat nappia painavan käyttäjän oikeuksilla. LAVAn taulukon solut käyttöliittymää lukuun ottamatta on suojattu loppukäyttäjien muutoksilta. Suojaaminen saa aikaan sen, että loppukäyttäjien oikeuksilla käynnistetyt skriptit eivät pysty suojattua dataa muuttamaan (Tanaikech 2020).

6.2.2 Ratkaisu

Käyttäjien pitäisi siis pystyä käynnistämään skriptejä eskaloiduin oikeuksin muuttaakseen suojattua dataa. Suojatun datan muuttamiseksi on LAVAssa

hyödynnetty GAS-alustan ominaisuutta nimeltä Installable Triggers (tästä eteenpäin asennettavat laukaisimet).

Asennettavat laukaisimet ovat GAS-projektiin liitettäviä laukaisimia, jotka suorittavat skriptejä tapahtuman perusteella, tai tiettyihin aikoihin. Asennettavat laukaisimet suoriutuvat sen käyttäjän oikeuksilla, joka on asennettavan laukaisimen GAS-projektiin luonut. (Google 2022)

Asennettaviin laukaisimiin on mahdollista lisätä laukaisin, joka suorittaa aina taulukon dataa muutettaessa tietyn funktion. LAVAssa on luotu tällainen *onEdit*-laukaisin, jonka skripti tarkistaa, täyttääkö solun muutos tietyt kriteerit ja sen perusteella mahdollisesti käynnistää järjestelmän toimintoja.

Laukaisimessa solun muutoksesta ei tarkisteta, mitä soluja on muutettu. Muutostapahtumista sen sijaan tarkistetaan, mikä solun arvo on ollut ennen muutosta. Muutosta edeltävä arvo määrää, saako muutos aikaan jonkin toiminnon suorittamisen.

Mikäli solun arvo ennen muutosta on ollut "> *HYLLYTÄ* <", niin tapahtuma saa aikaan hyllytystoiminnon. Hyllytystoiminnon loputtua laukaisimen skripti myös palauttaa muutetun solun arvon takaisin aikaisempaan, jotta kyseisen solun muutosta voidaan jälleen käyttää toiminnon laukaisemiseen. Toimintaperiaate on sama kaikissa toiminnoissa, vain solun alkuperäinen teksti eroaa.

6.2.3 Vaikutus alustojen käytettävyyteen ohjelmistokehityksessä

Skriptien suorittaminen taulukon omistajan oikeuksilla oli ongelma, joka oli pikaisen nettihakemisen jälkeen helppo ongelma ohittaa. Mielestäni tämä 'ongelma' ei erikseen vaikuta alustan käytettävyyteen sovelluskehityksessä.

Mahdollisuus nappien käyttöön myös tilanteissa, joissa skripti pitäisi suorittaa taulukon omistajan oikeuksin, helpottaisi esteettisten käyttöliittymien luomista. Jos nappeja voisi sijoittaa taulukkoon niin, että ne olisivat näppäimistöllä aktivoitavissa ja ne voisivat suorittaa toimintoja taulukon omistajan oikeuksin, niin voisi LAVAssa

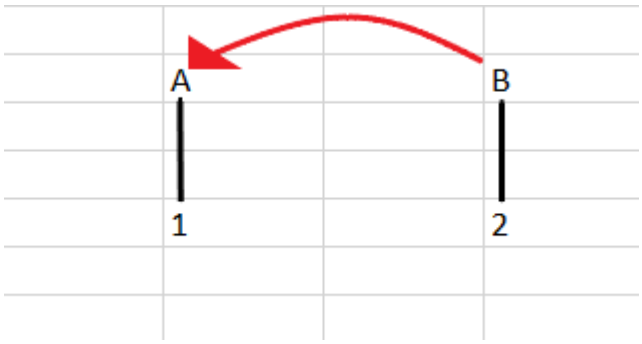
käyttää nappeja, mutta niiden käyttämättömyys ei vaikeuttanut kehitystyötä, eikä vaikuta negatiivistesti järjestelmän varsinaiseen toimintaan.

6.3 Solujen siirtäminen

6.3.1 Ongelma

Loppukäyttäjät eivät pysty siirtämään soluja, jotka ovat heiltä suojattuja. Loppukäyttäjän tosin pystyvät siirtämään heille sallittuja käyttöliittymien soluja muiden käyttöliittymän solujen päälle. Käyttäjille on LAVA-järjestelmän käytössä muutaman kerran käynyt niin, että he ovat hiirtä käyttäessä vahingossa vetäneet solun toisen solun päälle.

Käyttöliittymän solujen siirtäminen aiheuttaa ongelmia viitteissä, jotka viittavat soluun, jonka päälle toinen viitetty solu on siirretty. Esimerkkitilanteena solun 1 kaava viittaa soluun A ja solun 2 kaava viittaa soluun B (kuva 3). Jos solun B siirtää solun A päälle, muuttuu solun 2 kaava viittamaan uuteen paikkaan (A), mutta solun 1 kaavassa sen sijaan viite soluun A muuttuu viitevirheeksi (#REF) (kuva 4). Virhe sattuu vain silloin, kun siirretään solu, johon on viitattu, toisen solun päälle, johon on myös viitattu.



Kuva 3. Tilanne kun solu B siirretään solun A päälle. Solu 1 viittaa soluun A ja solu 2 viitta soluun B

Kuva 4. Tilanne kun solu B on siirretty solun A päälle. Solun 2 viite pysyy siirretyssä solussa B, mutta solun 1 viite virheytyy.

Viitevirhe saattaa virheyttää skritptien toimintaa ja jopa mahdollistaa virheellisten tietojen pääsyn datatauluihin. Viitevirheet pitää käydä käsin korjaamassa kaikista virheytyneistä viitteistä. Virheitä pitää useimmiten korjata useita, sillä viitteiden ketjussa joka ikinen viite virheytyy. Jos viiteketju päättyy soluun, jonka päälle on siirretty toinen solu, virheytyy jokainen viiteketjun solu. Kaikki solut on käsin korjattava.

Soluja siirtäessä myös data validation (data validaatio) siirtyy uudelle paikalle. Esimerkiksi käyttäjän vahingossa siirtäessä solun, johon syötetään paikka, siirtyy myös soluun asetettu sääntö, että siihen pystyy syöttämään vain sopivan paikan. Jos kyseisen solun siirtää *MÄÄRÄ*-solun päälle, niin tulee käyttöliittymästä käyttökelvoton, koska määrään ei enää pysty asettamaan numeroa kuten pitäisi

6.3.2 Ratkaisu

Viitteiden virheentymisongelman voi ratkaista sillä, että käyttöliittymän soluihin viitatessa käyttää kovan viitteen sijasta INDIRECT-kaavaa, jolle viite annetaan merkkijonona. Merkkijonona annettu viite ei dynaamisesti reagoi solujen siirtoihin, jolloin viite osoittaa aina oikeaan soluun, eikä virheenny siirtojen takia.

INDIRECT-kaava ei ratkaise data validaation siirtymisen ongelmaa. Data validaation siirtämisen saa tosin korjattua helposti, käyttäjän pitää vain siirtää data validoitu solu takaisin oikealle paikalleen. Välttämättä käyttäjä ei osaa näin tehdä, mutta ongelma on muutaman sekunnin työ pääkäyttäjän suorittaa.

Solujen siirtämiseen tai data validaation paikan lukitsemiseen ei ole mitään ratkaisua Sheetsissä. Solujen formatointia ei myöskään saa lukittua käyttäjiltä. Formatoinnin lukitsemattomuus tarkoittaa, että käyttäjän kopioidessa arvoja muista soluista, tulee kyseisten solujen fontit, korostukset ja värit käyttöliittymän soluihin. Formatoinnin muuttuminen on yleensä vain kosmeettinen vika, tosin käyttäjä saattaa kopioidessaan muuttaa solun tiedon tallennusmuodon esimerkiksi kokonaisluvusta tekstiksi.

6.3.3 Vaikutus alustojen käytettävyyteen ohjelmistokehityksessä

Solujen vahinkosiirtojen mahdollisuus rikkoa järjestelmän käyttöliittymiä on mielestäni huomattava negatiivinen asia järjestelmän kehityksessä Sheets- ja GAS-alustalla. LAVA on alun perin suunniteltu niin, että hyllyttämisiä ja keräilyjä suoritettaisiin vain numeronäppäimistöä käyttäen, jolloin hiirellä tai tablettia käyttäen sormella solujen siirtäminen vahingossa olisi mahdotonta. Valitettavasti LAVAn käyttökohteessa ei numeronäppäimistöjä ole saatu toimimaan varastohenkilöiden käytössä olevilla tableteilla.

Käyttöliittymässä, jossa solujen ei numeronäppäimistökäytön sujuvuuden takia tarvitse olla vierekkäin, voisi solujen väliin jättää muutaman solun kokoisen suojatun turva-alueen. Käyttäjä saisi virheen aikaiseksi siis vain jos hän erikseen vetäisi hiirellä tai sormella käyttöliittymän solun turva-alueen yli toisen käyttöliittymäsolun päälle. Tämä tilanne tosin ei ole mahdoton ja käyttäjän on mahdollista sekoittaa turva-aluein suunniteltu käyttöliittymä.

Siirretyn solun korjaaminen on nopea työ, mutta käyttäjä ei välttämättä osaa tehdä sitä itse. Mikäli pääkäyttäjään ei saada yhteyttä vahinkosiirron jälkeen saattaa järjestelmä olla käyttökelvoton pidemmänkin aikaa. Pitkään kestävä järjestelmän vikatila ei ole suotuisaa kriittisissä järjestelmissä.

Mielestäni vahinkosiirtojen ongelman vaikutus alustojen käytettävyyteen on huomattava, sillä siihen ei ole mitään varmaa ratkaisutapaa. On tapoja tehdä ongelman

ilmenemisestä hyvin epätodennäköistä, mutta vahinkosiirtoja ei valitettavasti voi mitenkään estää kokonaan.

6.4 Toimintojen käynnistymättömyys

6.4.1 Ongelma

LAVAssa on ilmennyt sellaista ongelmaa, että toiminnot eivät ole käynnistyneet, vaikka käyttäjä on käynnistyssolun arvoa muuttanutkin. Välillä skriptien suoritusten lokiin tallentuu jokin geneerinen virhe, jonka vihreilmoitus kertoo, että skripti ei juuri sillä hetkellä saanut yhteyttä taulukkoon. Toisinaan tätä sattuu useammin niin, että suoritusten lokiin ei tallennu yhtään mitään.

Lokiin tallentuvia virheentymisiä tapahtuu ehkä noin kerran viikossa, mutta lokista löytymättömiä virhetilanteita, joissa skriptit eivät käynnisty, tapahtuu useammin. Lokista löytymättömiä virheitä ei kuitenkaan järjestelmää kehittäessä ilmennyt lainkaan.

Valitettavasti taulukon kehittäjänä minulla ei varsinaisesti ole mitään keinoa itsenäisesti selvittää, miksi skriptit vain eivät jossain kohtaa käynnisty. Jos ongelma olisi skriptin sisältämän koodin virheentyminen, pystyisin sitä itse tutkimaan ja korjaamaan. Oletettavasti skriptien ohjelmakoodissa ei ole mitään vikaa, sillä GAS kyllä tallentaa ohjelmakoodin aina tiedot lokiin, kun ohjelmakoodi kohtaa virhetilanteen.

6.4.2 Ratkaisu

Valitettavasti skriptien selittämättömälle käynnistymättömyydelle en ole keksinyt mitään ratkaisua. Jokin selitys, ellei ratkaisu, voisi mahdollisesti löytyä, mikäli ongelmasta oltaisiin yhteydessä Googleen, joka voisi vielä syvemmin tutkia ongelmaa.

Onneksi skriptien käynnistämättömyys aiheuttaa vain hieman lisää työtä ohjelman käyttöön. Kun toiminnon aktivointisolun on muuttanut ja skritpi ei käynnistykkään, saa sen korjattua painamalla CTRL+Z, joka palauttaa aktivointisolun valmiiseen tilaan.

Loppukäyttäjät ovat havainneet, että ongelma ilmenee useimmiten silloin, kun useampi käyttäjä on muokkaamassa taulukkoa samaan aikaan. Se, että useampi käyttäjä saisi tämän ongelman aikaan, selittäisi sen, että tätä ongelmaa ei ole järjestelmää kehittäessä ilmentynyt.

6.4.3 Vaikutus alustojen käytettävyyteen ohjelmistokehityksessä

Skriptien käynnistymättömyys aiheuttaa pääasiassa ärtymystä loppukäyttäjille, sillä käyttäjät joutuvat ensin odottamaan, onko toiminnon käynnistymisessä vain hetkellinen viive vai jääkö se kokonaan suorittamatta. Odottelun jälkeen käyttäjä joutuu CTRL+Z -näppäinyhdistelmällä palauttamaan aktivointisolun normaalitilaan ja yrittämään aktivointia uudestaan.

7 SHEETS- JA GAS-ALUSTAN EDUT SOVELLUSKEHITYKSESSÄ

7.1 Käyttäjien hallinta

Useimpien tietojärjestelmien toiminnassa vaaditaan jonkinasteista käyttäjienhallintaa. Täysin uusi, alusta asti luotu järjestelmä vaatisi täysin uuden käyttäjienhallintakokonaisuuden luontia tai vähintäänkin järjestelmän yhdistämistä

olemassa olevaan käyttäjärekisteriin. Uuden käyttäjähallintakokonaisuuden luominen tai olemassa olevaan yhdistäminen vaatisivat molemmat suuren määrän kehitystyötä ja testaamista, mukaan lukien käyttäjien ja oikeuksien integrointia osaksi järjestelmän toimintaa.

Euromaster, jolle LAVA-järjestelmä on kehityetty, käyttää Microsoftin Active Directory Domain Domain Services (ADDS) -toimialuejärjestelmää. Toimialuejärjestelmässä hallitaan Euromasterin järjesetelmien käyttäjiä. Euromasterin Google Workspaces on yhdistetty yhtiön ADDS-järjestelmään niin, että käyttäjät voivat toimialuetunnuksillaan kirjautua Google palveluihin.

Sheets on Google palvelu, joten Sheets taulukon käyttäminen tehdään kirjautuneena Google Workspaces-tunnuksille, jotka ovat sidottu ADDS-tunnuksiin. Tämä integraatio mahdollistaa sen, että LAVA-järjestelmässä ei ole tarvinnut kehittää erillistä käyttäjärekisteriä, kirjautumisetoimintoa tai tapaa hallita käyttäjien oikeuksia.

Google Workspaces-tunnuksien oikeuksia on taulukon sisällä rajattu *taulukoiden suojelu* –ominaisuudella. Ominaisuus mahdollistaa taulukon välilehtien tai jopa vain välilehtien tiettyjen alueiden muokkausoikeuden rajaamisen vain tietyille käyttäjille. LAVAssa käyttäjiä ei ole rajoitettu sen hienommin, kuin loppukäyttäjien sallitaan muuttavan vain käyttöliittymävälilehdissä käyttöliittymän käyttöön vaadittavia soluja.

7.2 Nopea muutosten käyttöönotto

LAVAa kehittäessä kehtiystyötä nopeutti Sheets-taulukon ja GAS-skriptien helppo muokattavuus. Laskentataulukolle oli nopeaa hahmotella käyttöliittymänäkymiä. Taulukon loppukäyttäjälle jakamalla sai lähes välittömästi palautetta tehdyistä muutoksista.

GAS -koodinkin päivittäminen oli sujuvaa, sillä muutokset eivät vaatineet muuta kuin koodin muuttamista ja tallentamista GAS -työkalussa. Tämän jälkeen koodimuutoksen seuraukset olivat välittömästi käytettävissä.

8 SHEETS JA APPS SCRIPT KEHITYSALUSTANA

Sheets ja GAS toimivat tässä projektissa odotetusti vaatimuksissa määritellyn toiminnan luomiseen. Lähdin tietoisesti kehittämään järjestelmää Sheets-alustalla, vaikka jo alusta lähtien arvelinkin, ettei kehitystyö tule olemaan täysin mutkatonta (onko se ikinä?). Ohjelmakoodin kirjoittaminen GAS-alustalla oli mielekästä, vaikka JavaScript-kielellä olen ohjelmoinut viimeksi kaksi vuotta sitten hyvin yksinkertaisia toimintoja.

8.1 Hyvät puolet

Useimpien tietojärjestelmien toiminnassa vaaditaan jonkinasteista käyttäjienhallintaa. Täysin uusi, alusta asti luotu järjestelmä vaatisi täysin uuden käyttäjienhallintakokonaisuuden luontia tai vähintäänkin järjestelmän yhdistämistä olemassa olevaan käyttäjärekisteriin. Uuden käyttäjähallintakokonaisuuden luominen tai olemassa olevaan yhdistäminen vaatisivat molemmat suuren määrän kehitystyötä ja testaamista, mukaan lukien käyttäjien ja oikeuksien integrointia osaksi järjestelmän toimintaa.

Euromaster, jolle LAVA-järjestelmä on kehityetty, käyttää Microsoftin Active Directory Domain Domain Services (ADDS) -toimialuejärjestelmää. Toimialuejärjestelmässä hallitaan Euromasterin järjesetelmien käyttäjiä. Euromasterin Google Workspaces on yhdistetty yhtiön ADDS-järjestelmään niin, että käyttäjät voivat toimialuetunnuksillaan kirjautua Google palveluihin.

Sheets on Google palvelu, joten Sheets taulukon käyttäminen tehdään kirjautuneena Google Workspaces-tunnuksille, jotka ovat sidottu ADDS-tunnuksiin. Tämä

integraatio mahdollistaa sen, että LAVA-järjestelmässä ei ole tarvinnut kehittää erillistä käyttäjärekisteriä, kirjautumisetoimintoa tai tapaa hallita käyttäjien oikeuksia.

Google Workspaces-tunnuksien oikeuksia on taulukon sisällä rajattu *taulukoiden suojelu* –ominaisuudella. Ominaisuus mahdollistaa taulukon välilehtien tai jopa vain välilehtien tiettyjen alueiden muokkausoikeuden rajaamisen vain tietyille käyttäjille. LAVAssa käyttäjiä ei ole rajoitettu sen hienommin, kuin loppukäyttäjien sallitaan muuttavan vain käyttöliittymävälilehdissä käyttöliittymän käyttöön vaadittavia soluja.

Sheets-taulukkoon tietorakenteiden hahmottelu, muokkaaminen ja toteutus on mielestäni sujuvaa. Vaikka pitäisi kehittää isompaa järjestelmää ‘oikeilla’ työkaluilla, niin voisin nähdä itseni käyttämässä Sheets-taulukkoa ja simppeleitä GAS-skriptejä tietorakenteiden ja ohjelmakoodin vaatimusten hahmottamiseen.

Ohjelman käyttöliittymä on omasta ja loppukäyttäjien mielestä hyvä. Käyttöliittymässä on kaikki ohjelman käyttöön vaadittavat tiedot ja toiminnot. Sheetsin käyttöliittymän huonoista puolista enemmän seuraavassa luvussa.

Käytännössä Sheets-taulukolla ja GAS-skripteillä saa luotua mitä tahansa yksinkertaisia tietoja käsitteleviä järjestelmäkokonaisuuksia. Yksinkertaiset kaavat ovat tehokkaita ja niitä kekseliäästi hyväksikäyttämällä saa monimutkaiseltakin vaikuttavaa logiikkaa luotua ilman ensimmäistäkään riviä GAS-koodia. Esimerkiksi LAVAn hyllytyskäyttöliittymä ei suorita yhtäkään riviä itse kehitettyä ohjelmakoodia, ennen kuin käyttäjä aktivoi toiminnon käyttöliittymän ‘napista’.

8.2 Huonot puolet

Ensimmäinen paha puute Sheetsissä ja GAS-skripteissä on se, että Sheets-taulukko ei ole tietokanta, vaikka sitä tietokannan kaltaisesti käyttäisi. Taulukon soluille ei pysty asettamaan tarkkoja muotosääntöjä. Taulukossa ei myöskään ole mahdollista varsinaisesti luoda relaatiotietokannan kaltaisia viitteitä taulusta toiseen.

Relaatiotietokannan kaltaista toimintaa pystyy luomaan kaavoin ja GAS-skriptein, mutta monille ‘oikean’ tietokannan perusominaisuuksille, kuten viitatun tietueen poistamisen estolle ei ole mitään sisäänrakennettuja ominaisuuksia Sheetsissä; kaikki pitää itse simuloida GAS-koodissa. Oikeissa relaatiotietokannoissa hyvin määritelty tietokanta toimii datan vahtina suojellen dataa käyttäjien ja ohjelmien luomilta datan integriteettiä vahingoittavilta käskyiltä.

Toinen suuri huono puoli Sheetsissä itsessään on se, että käyttöliittymää ei pysty suojaamaan kaikilta käyttäjän virheiltä. Ideaalissa tilanteessa loppukäyttäjät pystyisivät muutamaan koko LAVAn taulukosta vain ja ainoastaan käyttöliittymän solujen arvoja. Loppukäyttäjät pitäisi pystyä estämään

- liikuttelemasta soluja
- muuttamasta solujen datatyyppiä
- muuttamasta solujen formatointia
- muutamasta solujen ehdollista muotoilua.

Olen LAVA-järjestelmän ollessa käytössä joutunut korjailemaan monenlaisia erilaisia ongelmia. Käytännössä kaikki ongelma ovat johtuneet siitä, että käyttäjät ovat epähuomiossa laittaneet vääriä arvoja vääriin paikkoihin, kuten määrän ja tuotenumeron syöttäminen käyttöliittymään päinvastoin kuin pitäisi.

Kuitenkaan käyttäjän virheet eivät useimmiten ole käyttäjän syytä. Sen sijaan käyttäjän virheet voidaan useimmiten todeta järjestelmän syyksi sen salliessa käyttäjän tehdä jotain, mikä johtaa virheisiin.

9 LOPUKSI

Vaikka Sheetsiä ja GAS-skriptejä käyttäen LAVA-järjestelmän kehitys oli ripeää ja helppoa, en silti suorittaisi samanlaista projektia uudestaan. Järjestelmässä on yhä

pitkän jatkokehittämisen jälkeen muutamia puutteita, joiden läpikäymiseen ei valitettavasti tämän työn laajuuteen mahtunut.

Saatan tosin tulevaisuudessa käyttää projektissa käyttämiäni alustoja jonkin muun projektin tietorakenteiden ja toimintojen hahmottamiseen, sekä yksinkertaisten toimintojen luomiseen ja testaamiseen. Tuotantoympäristöön tosin valitsisin silti sinne sopivampia osia tietojärjestelmän kehitykseen.

Nykymaailmassa on monia sovelluskehitysalustoja, jotka mahdollistavat käyttöliittymien ja toiminnallisuuden luomisen suhteessa vähällä kehitystyöllä. Tämän työn projektissa olen Sheetsiä ja GAS-skriptejä sellaisena käyttänyt, vaikka kyseisten työkalujen kehittäjät eivät luultavasti ole ikinä suunnitelleet työkalujansa niin käytettävän.

Ongelmista huolimatta LAVA on toistaiseksi tuotantokäytössä Euromasterin kokoonpanolaitoksen varastolla. Projekti oli tavoitteisiinsa nähden onnistunut ja on edistänyt kokoonpanolaitoksen toimintaa, vaikka en kehittäjänä itse vastaavaan projektiin lähtisikään enää uudestaan.

LÄHTEET

Flexiple. 2022. How to create a JavaScript Dictionary? Viitattu 23.01.2022. https://flexiple.com/javascript-dictionary/#:~:text=Caveats%20%2D%20JavaScript%20Dictionary-,Are%20there%20dictionaries%20in%20JavaScript%3F,to%20dictionaries%20and%20work%20alike

Google. 2021. Apps Script dokumentaatio. Viitattu 29.11.2022. https://developers.google.com/apps-script/reference

Google. 2022. Apps Script opas asennettaviin laukaisimiin. Viitattu 23.11.2022. https://developers.google.com/apps-script/guides/triggers/installable

--Hyde. 2020. Vastaus toisen käyttäjän esittämään kysymykseen. Viitattu 1.12.2022. https://support.google.com/docs/thread/42793831/script-function-in-sheets-are-very-slow?hl=en

Ramaa, A., Subramanya, K.N. & Rangaswamy, T.N. 2012. Impact of Warehouse Management System in a Supply Chain. International Journal of Computer Applications osa 54.

Tanaikech. 2020. User Runs Script for Range Protected by Owner using Google Apps Script. https://tanaikech.github.io/2020/11/05/user-runs-script-for-range-protected-by-owner-using-google-apps-script/