Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Ali Abdollahi & Himel Rahman

Puoluerekisteri.fi-verkkopalvelu

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Ohjelmistotuotanto

Insinöörityö

11.11.2019

Tekijät	Ali Abdollahi Himel Rahman
Otsikko	Puoluerekisteri.fi-verkkopalvelu
Sivumäärä Aika	76 sivua + 2 liitettä 11.11.2019
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja viestintätekniikka
Ammatillinen pääaine	Ohjelmistotuotanto
Ohjaajat	projektipäällikkö Mikko Laitinen lehtori Simo Silander

Insinöörityössä suunniteltiin ja toteutettiin Oikeusministeriölle verkkopalvelu puoluerekisterin ylläpitämiseksi ja uusien puolueiden perustamiseksi sähköisesti. Projekti on osa Oikeusministeriön demokratiaverkkopalveluita ja sen päätarkoitus on antaa mahdollisuus sähköisesti allekirjoittaa puolueen kannatusilmoituksia.

Työssä tutustuttiin työympäristöön suuressa kansainvälisessä IT-alan yrityksessä. Lisäksi siinä sai runsaasti kokemusta asiakassuhteiden ylläpitoisesta ja yleisesti käytetyistä projektinhallintatyökaluista ja -menetelmistä.

Puoluerekisteri.fi-palvelun käyttöliittymä toteutettiin TypeScript-, Angular-, SASS-, ReactiveX-, ja HTML5-teknologioilla. Palvelinpuoli puolestaan toteutettiin hyödyntäen Java-, Spring Boot-, Maven-, Hibernate- ja MySQL-teknologioita.

Jatkuva integraatio oli tärkeässä osassa projektin kulkua. Käytimme Docker-konttiteknologiaa Linux-palvelimillamme ajoympäristöjen virtualisointiin ja Jenkinsiä automaattisiin kasauksiin ja testauksiin.

Insinöörityössä käydään yksityiskohtaisesti läpi järjestelmän sähköinen puolueen rekisteröintiprosessi, joka alkaa yhdistyksen asiamiehen täyttämästä puoluehakemuksesta, aina Oikeusministeriön tekemään puolueen rekisteröintipäätökseen. Prosessi perustuu pitkälti lakiin, joten toteutuksessa oli noudatettava äärimmäistä tarkkuutta.

Projekti valmistui ennen aikataulua, joten asiakkaan toteuttamaan hyväksymistestaukseen jäi runsaasti aikaa, kuten myös tuotantoasennukselle. Asiakas oli tyytyväinen lopputulokseen.

Avainsanat	Spring Boot, Angular, Oikeusministeriö, REST, Docker

Authors	Ali Abdollahi Himel Rahman
Title	Puoluerekisteri.fi Web Service
Number of Pages Date	76 pages + 2 appendices 11 November 2019
Degree	Bachelor of Engineering
Degree Program	Information and Communication Technology
Professional Major	Software Engineering
Instructors	Mikko Laitinen, Project Manager Simo Silander, Principal Lecturer

This thesis covers the design and development of a webservice of the Ministry of Justice. The purpose of the webservice is to maintain the Finnish political party registry and to enable the registration on new parties electronically. The project is part of the Ministry of Justice's initiative to create new democratic web services and its main purpose is to make it possible for organizations to collect party endorsement cards online.

The front end of Puoluerekisteri.fi was implemented using TypeScript, Angular, SASS, ReactiveX, and HTML5. The main technologies used for the back end were Java, Spring Boot, Maven, Hibernate and MySQL.

Continuous integration played an important part during the project lifecycle. We used Docker containers on Linux servers to virtualize the staging environments, as well as Jenkins to automate project builds and testing.

The thesis goes through the process of registering a party though the webservice step-by-step. From the delegate of an organization filling an application, to the final decision made by the Ministry of Justice. Because the process is mainly based on the law, the implementation had to be done with extreme diligence.

The project was completed earlier than scheduled, which extended the amount of time allocated for deployment into production, while also allowing the customer more time for acceptance testing. The customer was satisfied with the final product.

Keywords	Spring Boot, Angular, Department of Justice, REST, Docker

Sisällys

Lyhenteet

API	Application programming interface. Ohjelmointirajapinta.
JAR	Java Archive. Java-arkistotiedosto.
JDBC	Java Database Connectivity. Java-rajapinta tietokantayhteyksille.
JDK	Java Development Kit. Javan ohjelmistokehityspaketti.
JNDI	Java Naming and Directory Interface. Java-rajapinta hakemistopalveluihin.
JPA	Java Persistance API. Kokoelma Java-rajapintoja tiedon tallentamiseen tietokantaan.
JSON	JavaScript object notation. JavaScript-olioihin perustuva tiedonvälitysformaatti.
JWT	JSON Web Token. JSON-pohjainen käyttöoikeustietue.
HQL	Hibernate query language. Hibernate-kirjaston tietokantakyselykieli.
HTML	Hypertext mark-up language. Hypertekstin merkintäkieli.
HTTPS	Hypertext Transfer Protocol Secure. HTTP- ja TLS/SSL-protokolien yhdistelmä.
LDAP	Lightweight Directory Access Protocol. Verkkoprotokolla hakemistopalvelujen käyttöön.
OM	Oikeusministeriö.
ORK	Oikeusrekisterikeskus.
ORM	Object-relational mapping. Oliomallin mukaisen esityksen kuvaus relaatiomallin mukaiseksi esitykseksi.

POJO	Plain Old Java Object. Yksinkertainen Java-olio.
PURE	Puoluerekisteri.fi.
REST	Representational State Transfer. Ohjelmointirajapinnan arkkitehtuurimalli.
SAMPO	ORK:n sähköisen asioinnin alusta.
SQL	Structured query language. Standardoitu kyselykieli relaatiotietokanta hauille.
SSH	Secure Shell Host. Protokolla kahden tietokoneen väliseen salattuun yhteyteen.
SSL/TLS	Secure Sockets Layer/Transport Layer Security. Protokolla, jolla suojellaan sovellusten tietoliikennettä verkossa. SSL on TLS:n edeltäjä.
VIA	Valtion yhteinen integraatioalusta (keskitetty sanomanvälityspalvelu).
VRK	Väestörekisterikeskus.
VTJ	Väestötietojärjestelmä.
WSDL	Web Services Description Language. XML-pohjainen rajapintojen kuvauskieli.
WYSIWYG	What you see is what you get. Termi ohjelmistolle, jonka sisältö näyttää muokattaessa samalta kuin sen lopputulos.
XML	Extended mark-up language. Tiedonvälitykseen tarkoitettu merkintäkieli.

1 Johdanto

Tämän projektin hankintayksikkönä toimii Oikeusrekisterikeskus. Projektin taustalla on demokraattisen päätöksentekojärjestelmän perusperiaate, puolueen rekisteröiminen. Yhdistyksen rekisteröiminen puolueeksi tapahtuu oikeusministeriön kautta, rekisteröintihakemuksen avulla. Hakemukseen tulee liittää vähintään 5000 kannattajakorttia, joiden allekirjoittajina toimivat eduskunta-, kunta- tai europarlamenttivaaleissa äänioikeutetut kansalaiset [1]. Kannattajakorttien keräämiseen on aikaa yksi vuosi.

Tällä hetkellä kannattajakorttien kirjoittaminen ja kerääminen sekä käsittely ovat mahdollista vain manuaalisesti, kynää ja paperia hyödyntäen. Puoluerekisteri.fi-hankkeen ja projektin taustalla on tavoite laajentaa puolueen rekisteröitymiseen liittyvä hakemusprosessi sähköiseen ympäristöön ja mahdollistaa myös kannattajakorttien digitaalinen kerääminen.

Raportissa projektin määrittelyosuudessa keskitytään yleisesti projektin hallintaan ja asiakkaan vaatimuksiin projektista. Projektilähtökohtiin kuului hyödyntää moderneja ketteriä toimintatapoja. Projektiryhmän, kehitystiimin, substanssiasiantuntijoiden ja muiden sidosryhmien tulee tietää ketterän menetelmän (Scrum) periaatteet sekä siihen liittyvät roolit ja vastuut. Tuoteomistajat, projektin asiantuntijat ja substanssiasiantuntijat ovat asiakkaan edustajia, jotka toimivat sisällön ja toiminnallisuuden asiantuntijoina ja yhteistyössä toimittajan kanssa luovat toiminnallisia ratkaisuja toteutusvaiheeseen.

Raportti muilta osin keskittyy kehitystyökaluihin ja käytettyihin tekniikoihin unohtamatta kokonaisuutta. Toimittaja (CGI) suositteli asiakkaalle erinäisiä hyväksi todettuja teknologioita, kuten Docker-konttipohjaista kokoonpanoa ja Java-pohjaista toteutusta sekä Spring Boot -sovelluskehystä palvelinpuolelle. Käyttöliittymä toteutettiin Angular 5 -sovelluskehyksellä ja Angular Material -komponenttikirjastolla. Teknologiavalintoja käytiin asiakkaan kanssa yhdessä läpi, jonka jälkeen ehdotuksemme hyväksyttiin.

Raportissa käsitellään myös kokonaisprosessia ja toimintoja eri käyttäjäroolien kautta. Puoluerekisteri.fi-palvelua käyttää kokonaisuudessaan neljä toisistaan erilaista käyttä-

jää, joilla on täysin erilaiset toiminnot. Kannattaja vierailee sivulla tutkimassa ja kannattamassa eri yhdistyksiä. Yhdistyksen asiamiehenä on mahdollista aloittaa puolueen rekisteröintiprosessi. Väestörekisterikeskuksen edustaja vahvistaa kerättyjen kannattajakorttien lukumäärän. Oikeusrekisterikeskuksen ja Oikeusministeriön työntekijät hoitavat hallinnolliset toiminnot, kuten puoluehakemusten hyväksymisen.

Sovellus valmistui muutaman kuukauden odotettua nopeammin, mikä tarkoitti sitä, että meille jäi aikaa hioa toimintoja ja keskittyä tuotannon asennukseen. Vaalit kuitenkin muuttivat suunnitelmamme mennä tuotantoon tammikuun ensimmäinen päivä vuonna 2019. Sovellus on tästä syystä hyllyllä odottamassa lakimuutosta, joka mahdollistaa kannattajakorttien keräämisen sähköisesti.

Raportti pyrkii antamaan kokonaisen kuvan modernista web-kehityksestä valtiolle tehtävän sovelluksen kautta. Valtiolle tehtäviin projekteihin liittyy erinäisiä vaatimuksia ja tavoitteenamme onkin selvittää näitä käsitteitä ja antaa ymmärrystä ratkaisuista ja toimintaperiaatteista.

2 Projektin määrittely

2.1 Vaatimusmäärittelyt (Tekniset vaatimukset)

Asiakkaan tavoitteena on kehittää puolueen rekisteröitymiseen liittyvää prosessia ja kannattajakorttien keräämistä. Kehitys tulee tapahtumaan uuden sähköisen palvelun kehittämisen ja käyttöönottamisen myötä.

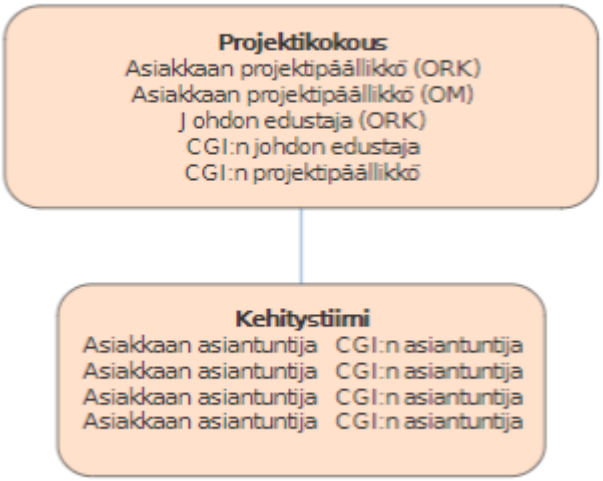
Puoluerekisteri.fi-hankkeen tavoitteena on tarjota saavutettava ja helppokäyttöinen digitaalinen palvelu, joka tarjoaa helpon ja nopean väylän niille yhdistyksille, jotka haluavat rekisteröityä puolueiksi ja niille kansalaisille, jotka haluavat allekirjoittaa kannattajakortteja. Prosessin digitalisoimisen myötä sekä kannattajakortteja keräävän tahon että rekisteröitymishakemuksia käsittelevän viranomaisen työ nopeutuu, ja mahdollinen päätös saadaan aikaan nykyistä nopeammin. Tavoitteena ei ole siirtyä kokonaan sähköiseen prosessiin, vaan manuaalisia työvaiheita on edelleen mahdollista hyödyntää esimerkiksi uuden sähköisen prosessin rinnalla.

Projektin tavoitteeksi asiakas on asettanut sen, että kehitystyö toteutetaan kustannustehokkaasti, laadukkaasti ja sovitussa aikataulussa.

2.2 Projektiorganisaatio

Projektin organisaatioon kuului CGI:n puolelta kolmen kehittäjän kehitystiimi, pääarkkitehti ja projektipäällikkö. Asiakkaan puolelta projektissa oli mukana projektipäälliköt Oikeusministeriöstä, Oikeusrekisterikeskuksesta ja Väestörekisterikeskuksesta. Lisäksi asiakkaalla oli myös tekninen vastaava ja työn valvojana toimi Suomen vaalipäällikkö.

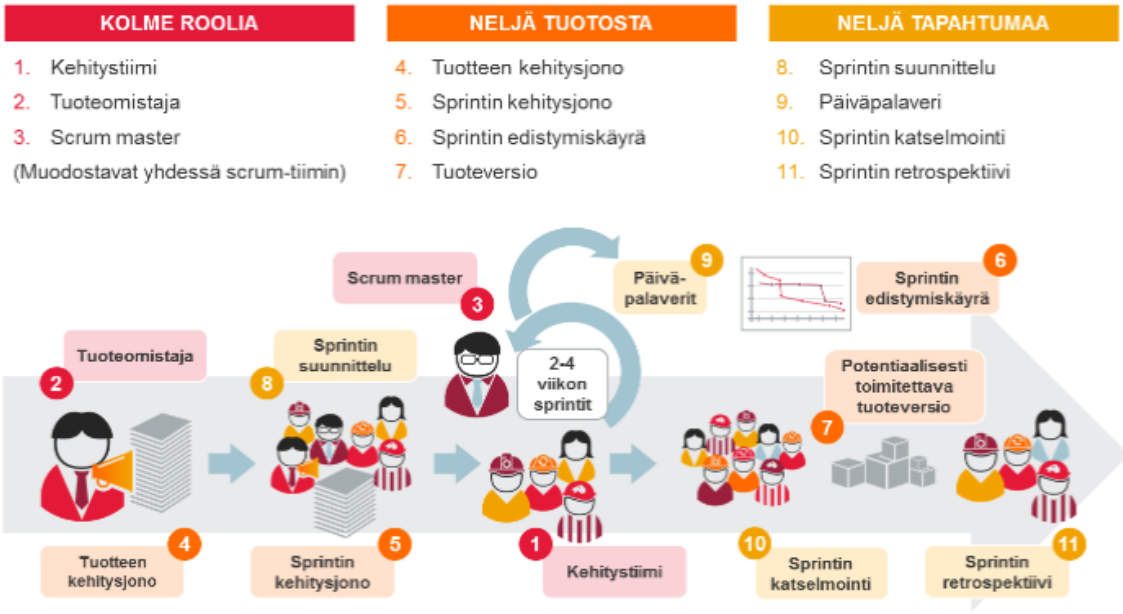
Alkuperäisen organisaatiokaavion mukaan, projektin kehitystiimiin olisi kuulunut asiantuntijoita myös asiakkaan puolelta (ks. Kuva 1), mutta lopulta tämä ei toteutunut.



Kuva 1. Projektin alustava organisaatiokaavio.

2.3 Scrum

Scrum on työnhallinnan menetelmä, jota voidaan käyttää osana erilaisia, pääasiassa ketteriä, projektimenetelmiä. Scrum perustuu Lean-periaatteisiin, nopeisiin kehityssykleihin ja jatkuvaan vuoropuheluun liiketoiminnan ja Scrum-tiimin välillä. Lean-periaatteen ytimenä on poistaa kaikki prosessit ja vaiheet, jotka eivät tuo lisäarvoa.

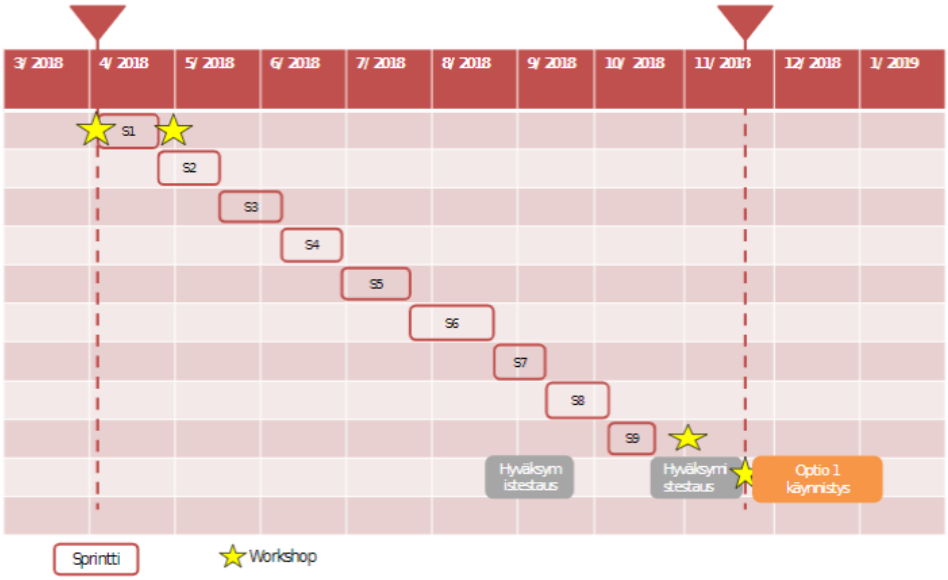


Kuva 2. Scrum-menetelmän toimintamalli.

Noudatimme Puoluerekisteri.fi-projektissa Scrumin mukaista toimintamallia (ks. kuva 2) parantaaksemme projektin ennakoitavuutta ja minimoidaksemme riskejä. Sprintti oli tyypillisesti kaksi viikkoa pitkä ajanjakso, jonka aikana potentiaalisesti toimitettava tuoteversio kehitettiin. Sprinttiin sisältyi päivittäinen Daily, eli päiväpalaveri, jossa jokainen tiimin jäsen kertoi tiimin kesken, mitä oli saanut aikaiseksi edellisenä päivänä sekä mitä hänellä oli aikomuksena tehdä kuluvana päivänä. Päiväpalaverissa käytiin läpi myös mahdollisia ongelmatilanteita ja pohdittiin yhdessä ratkaisuja niihin. Sprintin lopuksi demonstroitiin asiakkaalle tuotteen nykytilanne ja otettiin heitä vastaan palautetta ja parannusehdotuksia.

2.4 Aikataulutus

Projekti koostui yhdeksästä sprintistä, jotka sijoittuivat aikavälille 5.4.2018 – 18.10.2018. Sprinttien sisällöt oli suunniteltu asiakkaan kanssa jo ennen projektin alkua (ks. taulukko 1) ja sprinttien jälkeen pidettiin retrospektiivi, jossa keskusteltiin päättyvän sprintin saavutuksista. Sprintin demot oli myös sijoitettu sprinttien loppuun, ne kestivät yleensä noin puolesta tunnista tuntiin. Aikatauluun kuului myös työpajoja asiakkaan kanssa, missä heiltä pyydettiin mielipiteitä eri toiminnallisuuksien toteutukseen.



Kuva 3. Sprinttien ja työpajojen kalenteri.

Asiakkaan toteuttama hyväksymistestaus alkoi syyskuun lopussa ja kesti noin 3 viikkoa. Tähän aikaan kuului asiakkaan puolelta testaaminen ja CGI:n puolelta toimenpiteet huomioihin liittyen. Koska projektin toteutus tuli jo valmiiksi sprintillä 7, siirtyi hyväksymistestaus odotettua aikaisempaan.

Taulukko 1. Toiminnallisuudet eriteltyinä sprintti kohtaisesti.

Sprintti	**Toiminnallisuudet**	**Ajanjakso**
S1	Sovelluspohjat Hakemuslistaus Hakemuksen syöttö	5.4 -> 26.4.2018
S2	Hakemuksen tallentaminen tietokantaan kaikkine tietoineen Asiamiehen tallentaminen Asiamiehen näkymä Tunnistautuminen	26.4 -> 17.5.2018
S3	Kannatuksen tallentaminen puolueelle Kannattajan näkymä / profiili hänen kannatuksineen Puolueen kannatustilanteen näyttäminen	17.5 -> 7.6.2018
S4	Oikeusministeriön-käyttäjän näkymä Muistiinpanokenttien toiminta Oikeusministeriön-käyttäjän toimintojen toteutusta (mm. keruun rajojen säätö)	7.6 -> 28.6.2018

S5	Sähköposti-ilmoitusten toteutus Keruun loppumisen prosessit (aika loppuu / saatu kokoon tarpeeksi kannattajia) Uusien asiamiesten nimeäminen	28.6 -> 26.7.2018
S6	Oikeusministeriön-käyttäjien palvelun tekstikenttien muokkauseditori Tuki eri selaimille (Internet Explorer, Safari) Kannattajakorttien käsittely keruun keskeytyessä, saavuttaessa tavoitteensa tai aikarajan tullessa umpeen	26.7 -> 23.8.2018
S7	Etusivun ulkonäön muokkaus Hakemuksen tilojen muutokset Muut workshopeissa sovitut muutokset	23.8 -> 13.9.2018

2.5 Käyttötapaukset

Keskeinen osa projektin dokumentaatiosta olivat käyttötapaukset. Käyttötapaus kuvaa yhtä toimintokokonaisuutta sovelluksen käyttäjän tai järjestelmän kannalta ottamatta kantaa tekniseen toteutukseen. Siinä eritellään toimintokokonaisuuden eri toiminnot, niihin tarvittavat oikeudet ja mahdolliset esi- ja jälkiehdot. Lisäksi käyttötapaus sisältää vaihtoehtojen kuvauksia ja niihin liittyviä käsittelysääntöjä ja mahdollisia poikkeuksia. Kuvassa 4 näkyy tyypillisen käyttötapauksen rakenne.

Tarkoitus
Kannatuksen antaminen puolue-statusta varten kannatuksia keräävälle yhdistykselle

Toiminnon numero	Toiminnon kuvaus	Oikeusryhmä
1.	Puolueen kannatus	Sisään kirjautunut käyttäjä

Esiehdot
1. Kannatusten keräys käynnissä 2. Käyttäjä ei ole vielä kannattanut puoluehakemusta 3. Kannattaja on äänioikeutettu.

Jälkiehdot

Tekstikuvaus
Puoluehakemuksen sivulla voi kannattaa puoluehakemusta jos keräys on käynnissä ja käyttäjä ei ole aiemmin kannattanut puoluehakemusta.

Nro	Vaihtoehdon kuvaus
1	Käyttäjä kannattaa puoluetta puoluehakemuksen sivulta (käsittelysääntö 1) (poikkeukset 1, 2 ja 4). Jos kannatus onnistuu normaalisti, käyttäjälle annetaan palaute (käsittelysääntö 2). Jos kannatus epäonnistuu, käyttäjälle annetaan palaute (poikkeus 3).

Nro	Poikkeukset
1	Jos käyttäjä ei ole kirjautunut palveluun sisään, palvelu ohjaa käyttäjän kirjautumaan palveluun sisään.
2	Jos käyttäjä ei ole äänioikeudellinen, antaa järjestelmä virheilmoituksen.
3	Kannatuksen yhteydessä ongelman sattuessa ilmoitetaan kannatuksen epäonnistumisesta.
4	Jos käyttäjä on aiemmin äänestänyt kyseistä puoluetta, kannattaminen ei ole mahdollista.

Nro	Käsittelysäännöt
1	Järjestelmä tarkastaa, että hakemus on tilassa "Kerää kannatusta", aikaa on jäljellä, kannattaja on äänioikeudellinen ja hän ei ole kannattanut kyseistä puoluetta aiemmin. Järjestelmä nostaa päivittäistä ja kumulatiivista kannatusta yhdellä.
2	Onnistuneesta kannatuksesta tulee huomatus ja mahdollisuus siirtyä takaisin etusivulle.

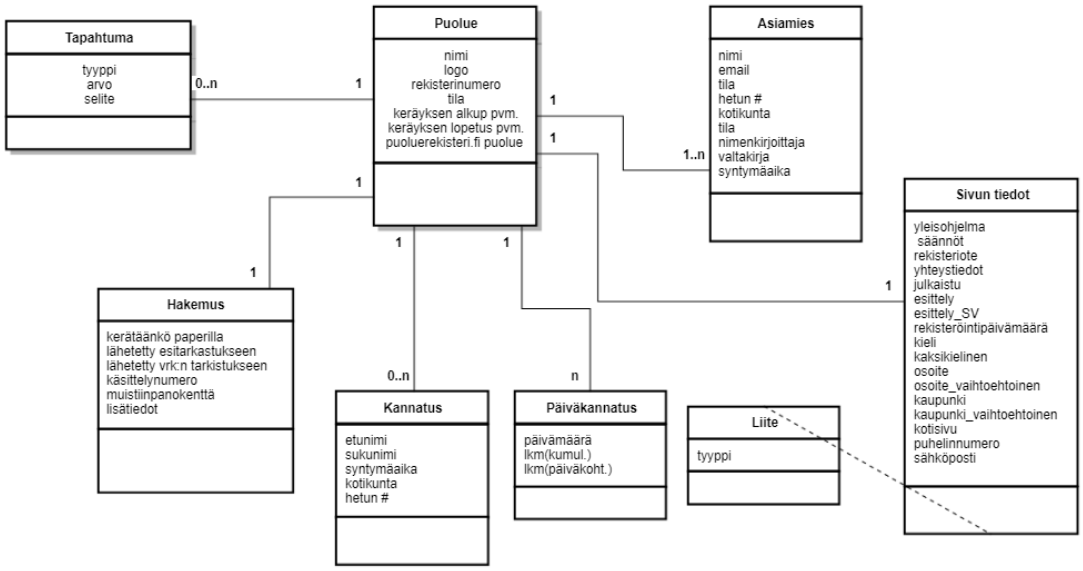
Kuva 4. Kyseinen käyttötapaus koskee yhdistyksen kannatusta.

Ajantasaiset käyttötapaukset ovat erittäin tärkeitä asiakkaille. Asiakas käytti projektin toteutuksen aikana käyttötapauksia eräänlaisina käyttöohjeina hyväksymistestauksessa, sekä kävi tarkastamassa säännöllisesti, että kehittäjinä olimme toteuttaneet toiminnot halutulla tavalla. Käyttötapaukset ovat myös tärkeitä sovelluksen ylläpidettävyyden kannalta. Tulevaisuudessa mahdolliset projektin jatkokehittäjät voivat tutustua sovelluksen eri toimintoihin ja ymmärtää niiden käyttötarkoituksen.

Puoluerekisteri.fi-projektissa kehittäjät olivat vastuussa käyttötapausten kirjoittamisesta ja päivittämisestä, joka yleensä tehtiin heti varsinaisen teknisen toteutuksen jälkeen. Käyttötapauksen ajantasaisuus oli esiehto toteutustiketin sulkemiselle.

2.6 Relaatiomalli

Relaatiomalli on kaavio, joka esittää tietokantaan tallennettava dataa listoina, jotka esittävät tietokantatauluja. Listat on linkitetty keskenään kuvaamaan tietokantataulujen välisiä suhteita. Projektin alussa laadittiin puoluerekisteri.fi-palvelusta relaatiomalli, jonka pohjalta suunniteltiin järjestelmän tietokantarakenne. Relaatiomallia päivitettiin projektin edetessä aina, kun tietokantaan tehtiin muutoksia.



Kuva 5. Projektin relaatiomalli.

3 Arkkitehtuuri

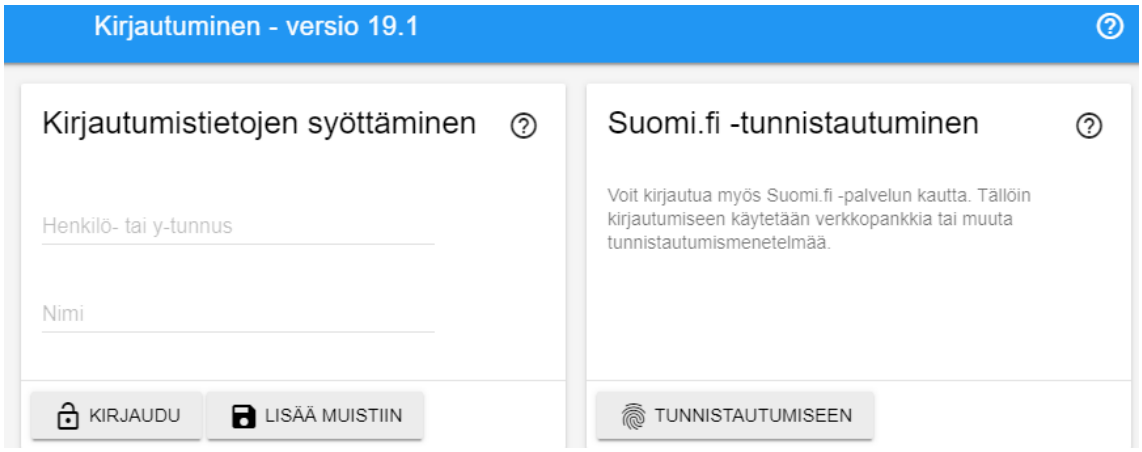
Puoluerekisteri.fi koostuu kahdesta osasta: sähköisestä asioinnista ja OM-hallintapaneelista. Kummatkin sisältävät palvelinpuolen ja selainpuolen. OM-hallintapaneelin palvelinpuoli sisältää suurimman osan ohjelmalogiikasta ja kaikki tietokantaoperaatiot. Hallintapaneelin selainpuoli on Oikeusministeriön ja Oikeusrekisterikeskuksen työntekijöiden sisäiseen käyttöön tehty verkkopalvelu. Hyödyntäen hallintapaneelia he voivat varmistaa, että hakemusprosessit etenevät ripeästi ja lain vaatimalla tavalla sekä pitää puoluerekisteri.fi-sivustoa ajan tasalla.

Sähköisen asioinnin käyttäjiä ovat henkilöt, jotka haluavat perustaa puolueen tai kannattaa järjestelmässä olevaa puoluehakemusta. Toisin kuin hallintapaneeli, joka on intranetsivu, SA-puoli on avoin kaikille. Tämän takia palvelimien suhteen tuli ottaa valmiuksia mahdollisia hyökkäyksiä ja häiriöaikaa varten. Sähköisen asioinnin palvelimet on kahdennettu, eli kuormanjakaja jakaa tulevat kutsut kahden palvelimen kesken. Yleisesti palvelimen kaatuminen voi aiheuttaa yrityksille suuriakin kuluja [2]. Kahdennuksella voidaan myös välttää tämän kaltaisia tilanteita, sillä niin kauan kuin toinen palvelin on toimintakunnossa, sovellusta voidaan käyttää.

SA-puolen palvelin sisältää eri päätepisteitä, joihin kutsuja lähetetään selainpuolelta. Jotkut päätepisteet ovat julkisia, mitkä eivät vaadi kirjautumista sisään, kuten etusivun tiedot. Saavutettuaan päätepisteen, sähköisen asioinnin palvelin välittää kutsut hallintapaneeliin palvelimeen ja välittää tiedot takaisin sähköisen asioinnin selainpuolelle, joka taas vuorostaan näyttää ne käyttäjälle. Kommunikointi SA-puolen ja OM-hallintapaneelin kautta tapahtuu VIA:n välityksellä. VIA on Valtorin sanomanvälityspalvelu valtion sovelluksille. Kun käyttäjä sähköisen asioinnin puolella selaa julkisia puoluehakemuksia, lähtee tästä kutsu sähköisen asioinnin palvelimelle, joka välitetään VIA:n kautta OM-hallintapaneeliin palvelimelle (ks. liite 1).

SA-puolella on tukenaan ORK:n sähköisen asioinnin alusta SAMPO. SAMPO koostuu monista moduuleista, joita jokaista ajetaan omissa Docker-konteissaan. Firewall on sovelluspalomuuri, joka perustuu Nginx-verkkopalvelimeen, modsecurity-liitännäiseen ja OWASP ModSecurity CRS -sääntöihin. Nginx määrittää, mitkä http-kutsut ovat sallittuja ja mitkä vastaukset pakataan gzip-muotoon, jolla nopeutetaan kutsujen palautumista. Modsecurity sisältää palvelimen turvallisuusasetukset. Se tarkistaa sisään tulevien kutsujen sisällöt, eikä päästä läpi kutsuja, jotka eivät vastaa näitä sääntöjä.

Oikeusministeriön palvelut käyttävät yhteistä Suomi.fi-kirjautumista. Tätä varten kehitettiin Login-kontti, joka hoitaa tunnistautumisen vähäisellä konfiguraatiolla. Kehitysympäristössä Login tarjoaa kirjautumisen millä vain tunnuksella ja kehitysversion Suomi.fi-kirjautumisesta.



Kuva 6. Testinäkymä kirjautumiselle. Vasemmalla on vapaa kirjautuminen tunnuksella, oikealla Suomi.fi-tunnistautuminen.

Koska eri osia ja yhteyksiä on paljon, tarvitaan jotain, joka hoitaa moduulien välisen kommunikaation. Gateway käyttää reititykseen Netflixin Zuul-kirjastoa, jonka avulla voidaan rakentaa dynaamisia reittejä, helpottaa monitorointia ja parantaa turvallisuutta [3]. Gateway-konfiguraatioihin on kovakoodattu oletusosoitteet, joita on mahdollista muokata ympäristömuuttujilla (ks. esimerkkikoodi 1).

```
routes:
  login:
    url: http://localhost:10280
  codeset:
    url: http://localhost:10580
  asiantuntijapalvelu:
    url: http://localhost:10980
  payment:
    url: http://localhost:8080
  pepu:
    url: http://localhost:11880
```

Esimerkkikoodi 1. Gateway-osoitteiden konfiguraatio.

4 Käytetyt työkalut

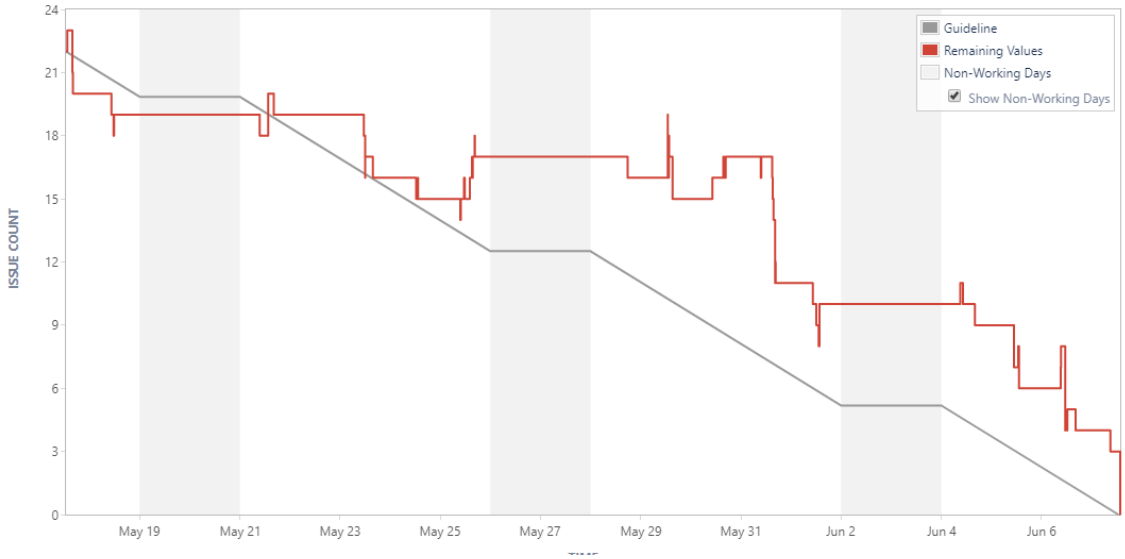
4.1 Projektinhallinta

Projektin hallintaan hyödynnettiin Atlassianin SaaS (Software as a service) -tuoteperheeseen kuuluvia ohjelmistoja. Ohjelmat on asennettu tietoturvan takia Valtroin ylläpitämälle palvelimelle, ja niiden käyttö vaatii VPN-yhteyden.

4.1.1 Jira

Jira on tiketinhallintajärjestelmä, joka mahdollistaa virheiden seurannan ja ketterän projektinhallinnan.

Puoluerekisteri.fi-projektissa käytimme Jiraa hallitsemaan toteutustikettejä ja suunnittelemaan sekä seuraamaan Sprintin tilannetta. Jirasta sai myös raportteja Sprinteistä, kuten burndown-kaaviolla, joissa näkyy tikettien tilanne Sprintin ajanjaksona. Taulukon harmaa viiva kuvaa ihanteellista tilannetta ja punainen viiva todellista tilannetta.



Kuva 7. Projektin kolmannen sprintin burndown-kaavio.

4.1.2 Bitbucket

Bitbucket on verkkopohjainen versiohallintajärjestelmä lähdekoodille ja kehitysprojekteille, jotka käyttävät Git-versionhallintajärjestelmää.

Käytimme projektissa Bitbucketia GIT-repositoryn hallintaan. Kehittäjät tekivät jokaisesta toteutustiketistä oman GIT-haaran, josta tehtiin lopuksi ”pull request”, joka vaati projektin arkkitehdin ja muiden kehittäjien hyväksynnän, ennen kun sen sai yhdistää projektin päähaaraan. Näin vältettiin huonoja koodauskäytäntöjä ja kehittäjien syntaksivirheitä.

4.1.3 Confluence

Confluence on yhteistyöohjelmisto, jolla voidaan luoda projekteille Wiki-sivustoja ja ylläpitää projektien dokumentaatiota.

Puoluerekisteri.fi-projektin kaikki dokumentaatio säilytettiin Confluence-wikissä. Sillä säilytettiin muun muassa projektin käyttötapaukset, tietoa eri ympäristöistä ja käyttäjätunnuksista sekä tuotannon asennusohjeet. Lisäksi toteutustikettien testaustaulukot säilytettiin ja ylläpidettiin Confluence-wikissä.

4.2 Palvelinpuoli

Palvelinpuolen pohjana on kaksi Java Spring -applikaatiota, jotka noudattavat model-view-controller-periaatteita.

4.2.1 Java Spring Boot -sovelluskehys

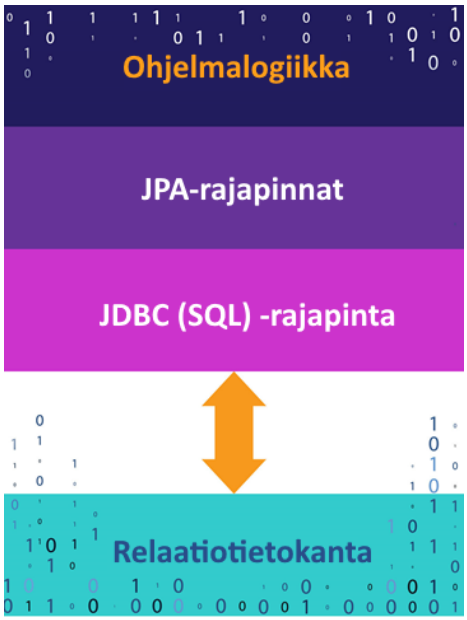
Spring Boot on avoimen lähdekoodin sovelluskehys Java-alustalle, ja se on rakennettu suositun Spring - Java EE -sovelluskehyksen pohjalta. Spring Boot pyrkii vähentämään Springin vaatimaa konfigurointia ja on kehitetty yksinkertaistamaan ja nopeuttamaan web-sovelluksen kehitystä. Spring Boot sisältää myös valmiiksi sisäänrakennetun web-palvelimen, monta Java-kirjastoa ja komentorivityökalun, jotka helpottavat ja nopeuttavat kehitystä.

Spring Security on sovelluskehys, joka tarjoaa Java Spring -applikaatioille työkalut käyttäjien autentikoimiseen. Security tarjoaa muitten Springin moduulien tapaan mahdollisuuden räätälöidä toiminnallisuutta omien tarpeiden mukaan. Käytimme sovelluskehystä kutsujen suodattamiseen sisäänrakennettua ketjua pitkin. Kutsujen tulee täyttää jokaisen ketjun osan vaatimuksen, muuten ketju lähettää koodin 401.

Kontrollerit ovat Spring-sovelluskehyksen tapa luoda päätepisteitä kutsuja varten. Kontrolleriin määritetään osoite, johon kutsut ohjataan, ja resurssi, jonka sen tulee palauttaa. MVC-mallin mukaan kontrolleri myös lähettää tiedon mallille, joka prosessoi ja palauttaa vastauksen takaisin, jonka jälkeen data siirretään näkymään käyttöliittymälle.

4.2.2 Hibernate-sovelluskehys

Hibernate ORM on olio-relaatiomuunnoksen toteuttava sovelluskehys Java-ohjelmointikielelle. Se tarjoaa oman implementaation JPA-rajapinnasta yhdistämään POJO Java -olioita (Plain Old Java Objects) relaatiotietokantaan. Yksinkertaisesti, jokainen POJO-olio vastaan yhtä riviä tietokantataulussa. JPA (Java Persistance API) on määritelmä, joka kuvaa kokoelman rajapintoja Java-olioiden tilan säilyttämiseksi ohjelman ajon loputtua (Persistance) käyttäen JDBC-rajapintaa (Java Database Connectivity). JDBC-rajapinta määrittää, kuinka asiakasohjelma (client) yhdistää tietokantaan. Kuva 8 havainnollistaa ohjelmalogiikan, JPA -rajapintojen, JDBC -rajapinnan ja relaatiotietokannan väliset suhteeet.



Kuva 8. Havainnollistava kuva ohjelmakoodin ja tietokannan yhteydestä JPA:n kautta [4].

Hibernate tarjoaa myös oma SQL-tietokantakielen tyylisen tietokantakielen nimeltä HQL. HQL-kielellä voidaan kirjoittaa tietokantahakuja Java-olioiden luokkamuutuuttujilla, jotka puolestaan palauttavat Java-olioita. Hibernate liittää Java-oliot tietokantatauluihin annotaatioilla, jotka määräytyvät JPA-määritelmän mukaisesti, vaikka on myös mahdollista käyttää Hibernaten omia implementaatioita annotaatioista. Annotaatioilla määritellään mm. tietokantataulun nimi, olion `ID`-atribuutti ja olion relaatiosuhteet muihin olioihin (ks. esimerkkikoodi 2).

```
@Entity
@DynamicInsert
@DynamicUpdate
@Table(name = "party")
public class DParty extends AbstractAuditingEntity implements Serializable,
Comparable<DParty> {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "name", length = 100)
    private String name;

    @Lob
    @Column(name = "logo")
    private byte[] logo;

    @Column(name = "company_number", length = 9)
    private String companyNumber;
```

```
    @Enumerated(EnumType.ORDINAL)
    @Column(name = "status", nullable = false)
    private EPartyStatus status;

    @OneToMany(fetch = FetchType.LAZY, cascade = CascadeType.ALL, mappedBy =
"party")
    @JoinColumn(name = "party_id")
    private List<DDelegate> delegates = new ArrayList<>();

    @OneToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "site_info_id")
    private DSiteInfo siteInfo;

    @OneToMany(fetch = FetchType.LAZY, cascade = CascadeType.ALL, mappedBy =
"party")
    private List<DDailyEndorsement> dailyEndorsements = new ArrayList<>();

    @OneToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "application_id")
    private DApplication application;
```

Esimerkkikoodi 2. Domain-luokka Hibernate-annotaatioilla.

4.2.3 Maven

Maven on Apache Software Foundationin kehittämä riippuvuuksien ja kasauksien hallintatyökalu Java-alustalle. Pom.xml-tiedosto määrittää versionumerot, eri kasausproseduurit ja liitännäisten konfiguraatiot. Kasausten avuksi Maven sisältää lukuisia vaiheita, jotka määrittävät kasauksen elinkaaren ja tavoitteita, joita liitännäiset tarjoavat. Tavoite on yksittäinen toiminto liitännäisen sisällä. Ajamalla komennon `mvn clean:clean` voimme kutsua Mavenia suorittamaan liitännäisen `clean` tavoitteen `clean`, joka tyhjentää generoidut tiedostot työhakemistossa. Esimerkiksi testausvaihe voidaan aloittaa ajamalla komento `mvn test`. Maven suorittaa kaikki tavoitteet, jotka liittyvät testiin ja sitä edeltäviin vaiheisiin. Erilaisilla liitännäisillä on erilaisia tavoitteita ja ne voivat kuulua erilaisiin elinkaaren vaiheisiin.

Maven tekee helpoksi hallita monitasoisia riippuvaisuuksia monien moduulien välillä. Maven sisältää kahdentyyppisiä riippuvuuksia, suoria ja transitiivisia. Suoria riippuvaisuuksia ovat esimerkiksi Spring-viitekehyksen komponentit, kuten Spring Boot, josta sovelluksemme ovat riippuvaisia (ks. esimerkkikoodi 3). Transitiiviset riippuvuudet ovat niitä, joita sovelluksen suorat riippuvuudet vaativat. Transitiiviset riippuvuudet lisätään automaattisesti suoran riippuvuuden mukana. Suoraa riippuvuutta lisätessä tulee määrittää `groupId`, joka erottaa sen muista projekteista. `ArtifactId`, joka on projektin

JAR-tiedoston nimi ilman versiota ja versionumero, joka on yleensä yhdistelmä numeroita ja pisteitä.

```
<dependency>
    <groupId>org.javassist</groupId>
    <artifactId>javassist</artifactId>
    <version>3.18.2-GA</version>
</dependency>
```

Esimerkkikoodi 3. Javassist-riippuvaisuus pom.xml-tiedostossa.

Maven-repositoryt ovat säilöjä, joihin Maven tallentaa JAR-tiedostot, liitännäiset ja muut projektikohtaiset resurssit. Säilöjä on kolmea tyyppiä. Lokaaliin säilöön tallennetaan kaikki ladatut resurssit, ja se sijaitsee kehittäjän omalla koneella, jotta kaikkea ei tarvitse ladata kasauksen yhteydessä verkosta. Keskussäilö on Maven-yhteisön ylläpitämä ja sisältää suuren määrän käytetyimpiä kirjastoja. Keskussäilöön yhdistäminen vaati Internetyhteyden ja on ensimmäinen paikka, josta Maven etsii resursseja, kun niitä ei lokaalista löydy. Joskus on tapauksia, että keskussäilöstä ei löydy haluttua riippuvaisuutta, mikä vaatii silloin säilön lisäyksen manuaalisesti. Säilö lisätään pom.xml-tiedostoon ja sille määritellään id ja osoite, josta resurssit löytyvät (ks. esimerkkikoodi 4).

```
<repositories>
    <repository>
        <id>spring-repo</id>
        <url>https://repo.spring.io/release</url>
    </repository>
</repositories>
```

Esimerkkikoodi 4. Spring-säilön osoitteen lisääminen pom.xml-tiedostoon.

4.2.4 Flyway

Flyway on avoimen lähdekoodin tietokantamigraatiotyökalu, joka mahdollistaa tietokannan versiohallintaan. Tietokannan versiohallinnalla voidaan varmistaa, että kaikilla kehittäjillä on ajantasainen tietokantarakenne ja että yhtäaikaiset rakenteen muutokset eivät aiheuta ongelmia. Tämän ansiosta Flyway soveltuu hyvin ketterään kehitykseen. Tietokantamigraatiot mahdollistavat tietokannan rakenteen palauttamisen uuteen tietokantaan. [5.]

Flyway-migraatiotiedostot kirjoitetaan SQL-komentoina, ja ohjelman ajon alussa Flyway käy kaikki nämä tiedostot läpi, suorittaa uudet ja nostaa suorituksen jälkeen tietokannan skeemaversiota (ks. kuva 9). Jos migraatiotiedostossa esiintyy virhe suorituksen aikana, niin Flyway palauttaa aiemman version.

installed_rank	version	description	type	script	checksum	installed_by	installed_on	execution_time	success
1	1	Initial Setup	SQL	V1__Initial_Setup.sql	1996767037	axel	2016-02-04 22:23:00.0	546	true
2	2	First Changes	SQL	V2__First_Changes.sql	1279644856	axel	2016-02-06 09:18:00.0	127	true
3	2.1	Refactoring	JDBC	V2_1__Refactoring		axel	2016-02-10 17:45:05.4	251	true

Kuva 9. Flyway skeemaversio -tietokantataulu.

4.3 Selainpuoli

4.3.1 TypeScript

TypeScript on avoimen lähdekoodin ohjelmointikehityskieli, joka on Microsoftin kehittämä ja ylläpitämä. TypeScript perustuu ECMAScript 2015 -kehityskieleen, jota paremmin tunnetaan nimellä JavaScript, ja on rakennettu sen pohjalta. TypeScriptin avulla voidaan käyttää tiettyjä toiminnallisuuksia, joita JavaScriptissa ei ole mahdollista käyttää.

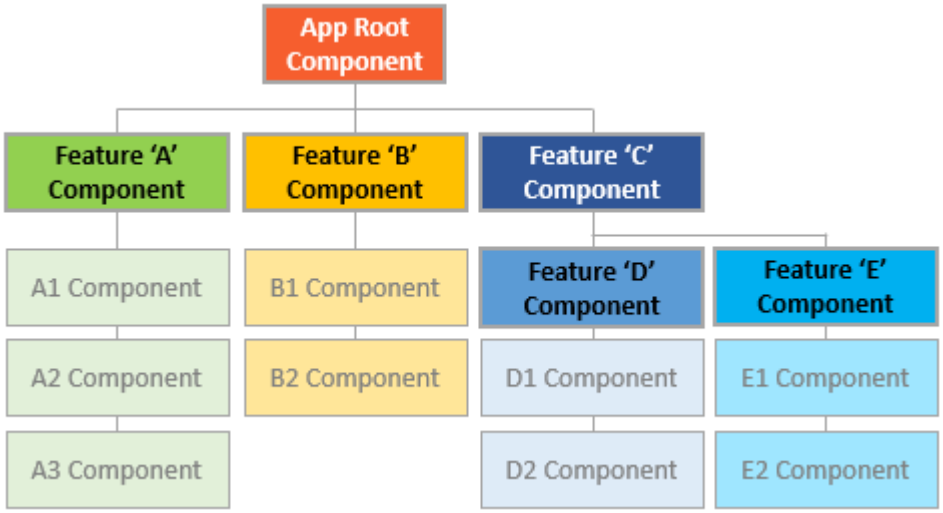
TypeScriptin merkittävimpiin ominaisuuksiin kuuluu muuttujien tyypitys, joka vähentää virheiden syntymistä koodissa ja helpottaa sen ymmärrettävyyttä. TypeScript lisää myös tuen luokkien, rajapintojen, enumeraattoreiden ja periytymisen käytölle.

TypeScript muutetaan JavaScript-koodiksi ohjelman rakennusvaiheessa.

4.3.2 Angular 5

Angular 5 on Googlen kehittämä web-sovelluskehys yksisivuisten web-sovellusten rakentamiseksi, ja se on saavuttanut laajan käyttäjäkunnan. Angularia kehitetään lähinnä käyttämällä TypeScriptia kehityskielenä. Angulariin kuuluu myös kokoelma komentoriviohjelmia, nimeltä Angular CLI, jotka auttavat kehittäjiä alustamaan uusia projekteja, kasaamaan projektin, validoimaan projektin koodia (linting) ja ajamaan automaatio- ja

yksikkötestejä [6]. Angular web-sovellukset rakentuvat komponenteista ja voidaan esittää komponenttipuuna, joka alkaa juurikomponentista. Komponentteja voidaan rakentaa toisista komponenteista, joista muodostuu puun oksat (ks. kuva 10).



Kuva 10. Angularin komponenttipuu [7].

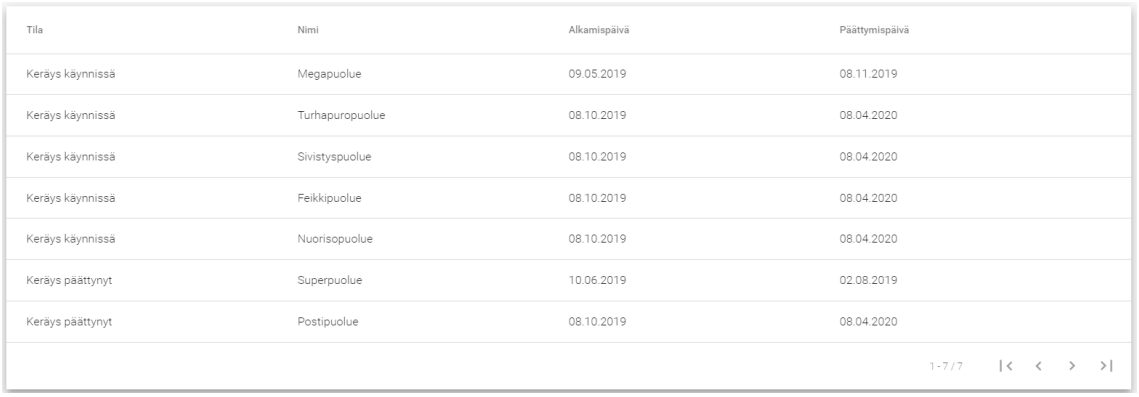
Angularin tärkeimpiin ominaisuuksiin kuuluu myös kaksisuuntainen datasidonta. Kaksisuuntainen datasidonta välittää dataa komponentin TypeScript osalta html-pohjalle, eli käyttöliittymälle, sekä päivittää käyttöliittymässä tapahtuvan muutoksen takaisin komponentin dataan. Dataa liikkuu siis molempiin suuntiin komponentin ja käyttöliittymän välillä.

Angular sisältää myös reitittimen, joka mahdollistaa navigoinnin Angular-sovelluksen sisällä. Reitittimen avulla voidaan määritellä URL-osoitteita komponenttikohtaisesti, jolloin navigointi yksisivuisessa web-sovelluksessa onnistuu samaan tyyliin kuin perinteisellä web-sivustolla.

4.3.3 Material Angular

Material Angular on Angular-tiimin kehittämä komponenttikirjasto. Sen mukana tulee kokoelma komponentteja, jotka ovat hyvin skaalautuvia ja optimoituja Angularille. Material Angular -komponentit on toteutettu noudattamaan Material Design -muotokieltä, joka on

alun perin Googlen kehittämä muotokieli Android-puhelinten käyttöliittymää varten. Material Angular soveltuu siis hyvin projekteihin, joissa halutaan nopeasti kehittää yhteneväinen käyttöliittymä, joka noudattaa moderneja standardeja ja on mobiiliyhteensopiva.



Tila	Nimi	Alkamispäivä	Päättymispäivä
Keräys käynnissä	Megapuolue	09.05.2019	08.11.2019
Keräys käynnissä	Turhapuropuolue	08.10.2019	08.04.2020
Keräys käynnissä	Sivistyspuolue	08.10.2019	08.04.2020
Keräys käynnissä	Feikkipuolue	08.10.2019	08.04.2020
Keräys käynnissä	Nuorisopuolue	08.10.2019	08.04.2020
Keräys päättynyt	Superpuolue	10.06.2019	02.08.2019
Keräys päättynyt	Postipuolue	08.10.2019	08.04.2020

1 - 7 / 7

Kuva 11. Esimerkki Material Angularin taulukkokomponentista.

Toisaalta Material Angularin kaltaisen komponenttikirjaston käyttöön voi myös liittyä varjopuolia. Valmiit komponentit ovat usein huonosti muokattavissa, eli ongelmatilanteita voi tulla vastaan, jos valmiskomponentin toiminnallisuus ei suoraan vastaa projektissa vaadittua toiminnallisuutta. Näissä tilanteissa komponenttia joudutaan räätälöimään omaan tarkoitukseen.

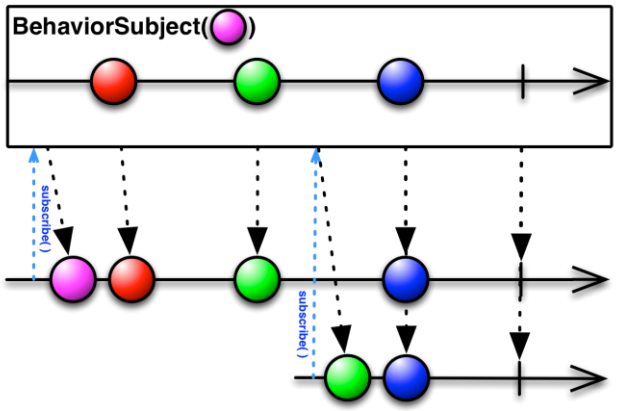
4.3.4 RxJS

RxJS on JavaScript-implementaatio suositusta ReactiveX (Reactive Extensions) -kirjastosta. ReactiveX on kirjasto reaktiiviselle ohjelmoinnille käyttäen tietovirtoja (data streams). ReactiveX luo tietovirran mistä tahansa luettavasta tiedonlähteestä, oli se esimerkiksi HTTP-kutsu tai hiiren nappulan klikkaus-tapahtuma. Tietovirtojen ansiosta RxJS-kirjaston avulla voidaan käsitellä monesta lähteestä saatua dataa samalla tavalla. Tietovirjoten lähteitä kutsutaan tuottajiksi, ja RxJS tarjoaa oman rajapinnan tuottajille nimeltä Observablet [8]. Observablet toimivat kulmakivenä asynkronisen ja takaisinkutsufunktioihin (callback) perustuvan koodin kirjoittamisessa.

Observablet kiinnitetään datavirtaan, ja ne ovat vastuussa datan muokkaamisesta ja sen lähettämisestä Observerille. Observablet ovat verrattavissa JavaScriptin natiiviin Promise olioon, mutta toimivat samalla myös datavirtana ja ovat usealla tavalla monipuolisempia.

ReactiveX tarjoaa myös käsitteen Subject. Subjectit voivat toimia Observablena, mutta ne voivat myös välittää dataa usealle Observerille samanaikaisesti (subscribe). RxJS-kirjasto tarjoaa neljää eri Subject-mallia [9].

- Subject
  - Lähettää vastaanottajille uudet arvot, mutta uudet vastaanottajat saavat vain tilauksen jälkeiset arvot.
- AsyncSubject
  - Lähettää vain viimeisimmän arvon lähde Observablelta vastaanottajille.
- BehaviorSubject
  - Lähettää viimeisimmän arvon kaikille uusille vastaanottajille, jonka jälkeen se jatkaa uusien arvojen lähetystä kaikille vastaanottajille. BehaviorSubjectille voidaan myös antaa ensimmäiseksi arvoksi vakioarvo, jos sillä ei ole vielä vastaanottajia (ks. kuva 12).
- ReplaySubject
  - Toimii kuten BehaviorSubject, paitsi että se lähettää kaikki edelliset arvot uusille vastaanottajille.



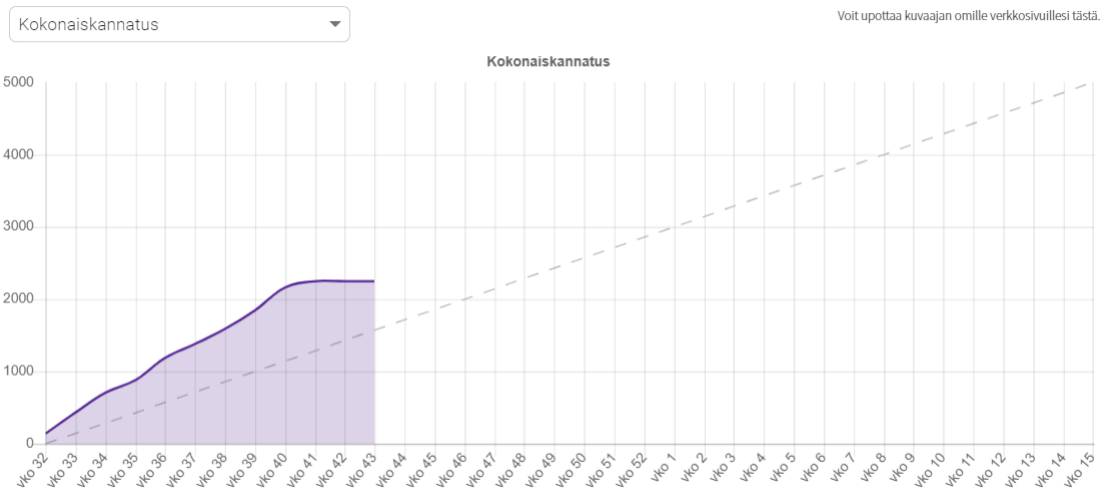
Kuva 12. BehaviorSubject-olion toimintoa kuvaava diagrammi [9].

Subject-olioiden käyttö oli keskeisessä asemassa puoluerekisteri.fi-projektissa, sillä kaikki web-sovelluksen kutsut ja tapahtumat käsiteltiin asynkronisesti Subject-olioita

hyödyntäen. Projektissa käytettiin kaikkia eri Subject-malleja, riippuen siitä, millaista toimintoa vaadittiin.

4.3.5 Chart.js

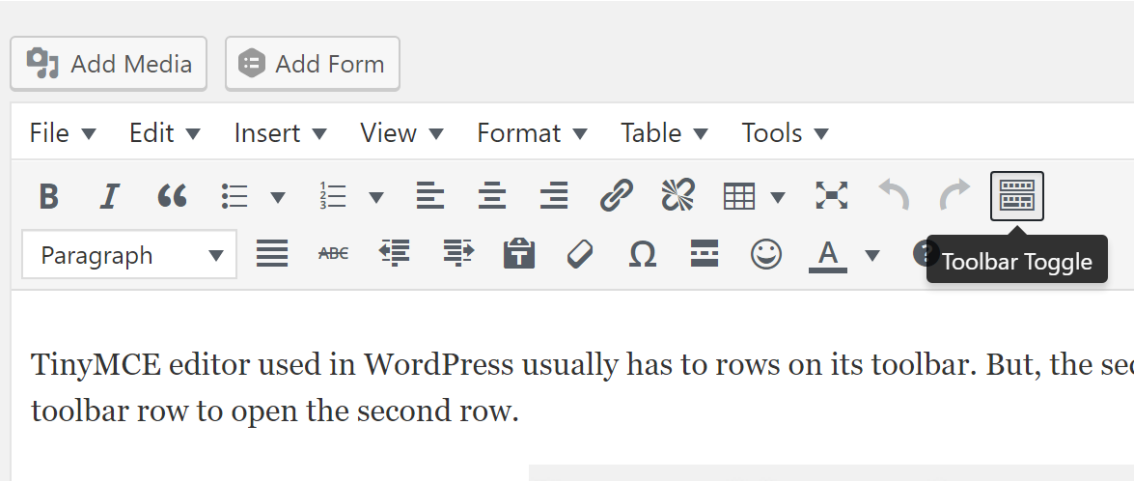
Chart.js on avoimen lähdekoodin JavaScript-kirjasto datan visualisointia varten. Kirjaston avulla on yksinkertaista luoda erilaisia dynaamisia HTML5-taulukoita ja kuvaajia. Projektissa käytettiin Chart.js-kirjastoa luomaan erilaisia kaavioita puoluehakemuksen kannattajamääristä eri aikaväleillä (ks. kuva 13).



Kuva 13. Puoluehakemuksen sivulla näkyvä kokonaiskannatusmäärää kuvaava kaavio.

4.3.6 TinyMCE

TinyMCE on avoimen lähdekoodin, WYSIWYG -periaatteeseen perustuva HTML-editori. TinyMCE:n vahvuuksia ovat sen helppo muokattavuus ja laajennettavuus, sekä editorin helppokäyttöisyys. Oletusasetuksilla TinyMCE-editori on ominaisuusrikas tekstieditori, jolla voidaan käsitellä ja luoda monipuolista HTML-sisältöä piittaamatta taustalla olevasta koodista.



Kuva 14. Esimerkkikokoonpano TinyMCE-editorista, johon on lisätty monta lisäosaa [10].

Editoria voidaan myös räätälöidä asiakkaan tarpeiden mukaan lisäämällä ominaisuuksia, joko liittämällä editoriin valmiita lisäosia tai käyttämällä TinyMCE:n tarjoamaa rajapintaa.

4.4 Kehitysympäristön asennus

Jokainen kehittäjä asentaa vaadittavat työkalut omalle Windows-kehitystietokoneelle. Työkalujen asennusvaiheita ei tarkemmin kuvata, mikäli ohjelma ei vaatinut erillistä konfigurointia.

4.4.1 JDK, Maven

Kehityskoneelle asennetaan JDK 1.8, sillä haluamme JDK:n, joka tukee uusimpia Java 8 ominaisuuksia. JDK saa olla maksimissaan revisio 161/162, sillä uudempi vaatisi lisenssin. Asennuspaketti voidaan hakea Oraclen sivuilta. Java 8 hyödyllisiin ominaisuuksiin kuulu optionalit, streamit ja päivitykset Date-rajapintaan, mitä tullaan projektissa käyttämään runsaasti.

Asennuksen jälkeen konfiguroidaan uusi ympäristömuuttuja `JAVA_HOME`, joka viittaa suoraan JDK-asennukseen (ks. esimerkkikoodi 5).

```
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_xxx.
```

Esimerkkikoodi 5. `JAVA_HOME`-ympäristömuuttujan asetus.

Tämän jälkeen muokataan olemassa olevaa `PATH`-muuttujaa lisäämällä sen perään uusi arvo: `%JAVA_HOME%\bin`. Järjestelmä tietää nyt mistä löytää oikean JDK:n. Asennuksen onnistuminen voidaan tarkastaa komennolla `java -version`, jonka pitäisi tulostaa konsoliin käytetyn Java-version tiedot.

Projektissa käytetään Maven 3.5.0 -versiota, jonka voi hakea Mavenin sivulta. Puretaan haettu zip-kansio Windows-käyttäjäkansioon esim. `C:\Users\username\Progs\apache-maven-3.5.0\bin\`. Muokataan PATH-ympäristömuuttujaa osoittamaan bin-kansion sijaintiin. Huomioidaan, että polun kansiosta täytyy löytyä mvn-komento/bat-tiedosto. Jos piilotettua .m2-kansiota ei ole, voidaan se luoda komennolla (ks. Esimerkkikoodi 6).

```
md c:\Users\username\.m2
```

Esimerkkikoodi 6. Luodaan maven-kansio, mikäli sitä ei ole.

Seuraavaksi luodaan uusi Maven-asetustiedosto/settings.xml, sijaintiin `C:\Users\username\.m2\settings.xml`. Tiedosto sisältää tietolähteiden osoitteita, jotta Maven osaa hakea riippuvuudet oikeasta paikasta.

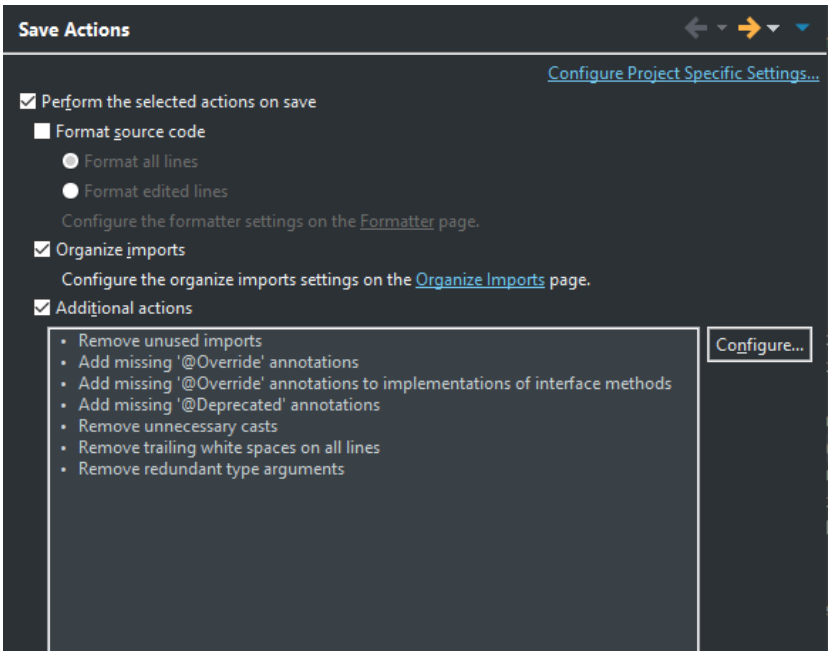
4.4.2 Node.js, Angular ja Yarn

Node.js on monialustainen ajoaikainen kirjasto, jolla voidaan ajaa JavaScript-koodia selaimen ulkopuolella. Node.js hyödyntää ei lukitsevaa I/O-arkkitehtuuria, mikä mahdollistaa monien kutsujen tekemisen, vaikka Node.js itse käyttää vain yhtä säiettä. Node.js-alustaan kuuluu npm-niminen pakettienhallintatyökalu, joka mahdollistaa avoimeen lähdekoodiin perustuvien JavaScript-kirjastojen hyödyntämisen. Projektissa käytimme kuitenkin Yarnia, joka on Facebookin vastaava työkalu npm:n tilalla nopeuden ja riippuvuuksien lukituksen tuen vuoksi.

4.4.3 Eclipse

Eclipse on avoimeen lähdekoodiin perustuva ohjelmointiympäristö. Se on yksi käytetyimpiä ohjelmointiympäristöjä Javalle, ja se on ollut kehityksessä yli 15 vuotta. Tämä tekee Eclipsestä varteenotettavan ja luotettavan vaihtoehdon esimerkiksi IntelliJ:n Idealle [11]. Eclipse tukee uusimpia Java-versioita, rikasta liitännäiskirjastoa ja kehittyneen koodin kirjoittamisavustajaa. Projektin kannalta yksi tärkeimmistä työkaluista Spring Tool Suite on saatavilla Eclipseen, joka mahdollistaa monien Spring Boot -pohjaisten sovellusten samanaikaisen ajon. Tämä on ensisijaisen tärkeää, jotta voimme ajaa SAMPO-projektia Puoluerekisterin rinnalla.

Eclipse ei suoraan asennettuna täyttänyt haluttaja vaatimuksia, mistä syystä näimme tarpeelliseksi muuttaa sen asetuksia. Aloitamme ensiksi lataamalla version Eclipse IDE for Java Developers, jonka jälkeen tuomme Eclipseen vaadittavat projektit. Haluamme päivittää kaikkien projektien Maven-riippuvuudet seuraavaksi, jotta saamme kaikki tarvittavat kirjastot käyttöömme. Vaatimuksena oli konfiguroida Eclipse käyttämään uljas_code_format.xml-tiedostoa formatointiin. Saimme haettua tiedoston yleisistä resursseista, sillä samaa formatointia käytetään muissakin yksikömme projekteissa. Save actions Eclipsessä määrittää, mitä tapahtuu sen jälkeen, kun käyttäjä tallentaa. Lisäämme tänne muutaman ehdon, jotka poistavat turhat importit, välilyönnit ja tyyppimuunnokset ja lisäävät puuttuvat `override/deprecated` annotaatiot (ks. kuva 15).



Kuva 15. Eclipse Save Actionsin konfiguraatio.

4.4.4 Visual Studio Code

Visual Studio Code on Microsoftin kehittämä ohjelmointiin soveltuva tekstieditori. Se on avoimeen lähdekoodiin perustuva ohjelma, johon käyttäjien on mahdollista luoda vapaasti liitännäisiä. VSCode on tällä hetkellä suosituin kehittämistyökalu, joka takaa sen, että Microsoft tulee tukemaan ja kehittämään sovellusta [12].

Käytimme Visual Studio Code editoria ja monia liitännäisiä pääasiassa selainpuolen koodaamisessa ja versionhallintatyökaluna Git-integraation kautta. Erityisesti Angular-pohjaisessa ratkaisussa VSCode tarjoaa erinomaisia liitännäisiä, kuten TSLint, joka huomauttaa reaaliajassa virheistä Typescript-kooditiedostoissa.

Yksi suosituimpia liitännäisiä on Angular Language Service, joka helpottaa editointikokemusta. Tähän kuuluvat automaattinen täydennys, diagnostiikkaviestit, nopea informaatio komponenteista ja mahdollisuus siirtyä funktioiden deklaraatioihin.

4.4.5 MySQL ja HeidiSQL

Jokaisella Puoluerekisterin kehittäjällä on oma MySQL 5.7 -tietokanta asennettuna tietokoneellaan. Asennuksen jälkeen luodaan uusi tietokanta nimellä ”pure”. Jotta yhteys palvelimelta tietokantaan toimii, ohjataan Hibernaten datasource osoittamaan lokaalin kannan osoitteeseen.

Tietokantaoperaatioita varten halusimme kevyen ja avoimeen lähdekoodiin perustuvan ohjelman, jota on helppo oppia käyttämään, mistä syystä päädyimme käyttämään HeidiSQL-kantatyökalua. HeidiSQL tarjoaa mahdollisuuden luoda ja muokata tauluja, sekä selata ja muuttaa dataa. Ohjelma on ladattavissa ilmaiseksi Microsoft-kaupasta.

4.4.6 Sovellusten ajaminen

Java projektit: hallintapaneeli, sähköinen asiointi ja SAMPO, käynnistetään Eclipsellä Spring Tool Suite -lisäosan kautta. Seurataan lokeja ja varmistetaan, että projektit käynnistyivät ilman virheitä. Käynnistetään MySQL-tietokanta ja tarkastetaan, että yhteys siihen toimii. Seuraavaksi avataan komentorivi projektien sähköinen asiointi ja hallintapaneeli juuressa ja ajetaan `yarn install`-komento. Tämä hakee ja päivittää selainpuolen projektien riippuvuudet. Jotta kirjautuminen lokaalissa ympäristössä saadaan toimimaan, vaihdetaan hallintapaneeli projektin login.html-tiedostoon sisäänkirjautumisskriptistä HTTPS HTTP:ksi. Lopuksi navigoidaan hallintopaneelin osoitteeseen `http://localhost:9000` ja yritetään kirjautua sisään testitunnuksilla `admin/admin`. Tehdään sama sähköisen asioinnin puolella kirjautumalla generoidulla henkilötunnuksella osoitteessa `http://localhost:3400`. Sen jälkeen, kun on varmistettu, että yhteydet sivuille toimivat ja kirjautuminen ei anna virheitä, voidaan vahvistaa lokaalin ympäristön asennuksen onnistuminen.

5 Jatkuva integraatio, toimitus

Ennen kuin jatkuvasta integraatiosta ja ketterästä kehityksestä tuli hallitsevia kehitysmenetelmiä suoritettiin integraatiot isoissa palasissa, jolloin myös testaamien tapahtui pal-

jon toteutuksen jälkeen. Moderneista yrityksistä moni on siirtynyt tämänkaltaisesta vesiputousmallista ketterimpiin ratkaisuihin [13]. Alkuperäisessä vesiputousmallissa kehittäminen on jaettu kuuteen eri vaiheeseen, jotka ovat systeemi, analyysi, suunnittelu, koodaus, testaus ja operaatiot [14]. Mallissa edetään ylhäältä alas vaiheissa, jotka muistuttavat ryöppyävää vesiputousta.

Jatkuvassa integraatiossa kehittäjät yhdistävät oman haaransa muutokset master-haaraan, joka vastaa tuotantovalmista sovellusta mahdollisimman usein. Uudet muutokset Gitissä laukaisevat webhookin avulla kasauksen ja automaattitestien validoinnin. Mikäli kasauksessa tai testeissä ilmenee ongelmia, lähtee tästä kehittäjille viesti. Nopeasti saatava palaute mahdollistaa yksittäisten ongelmien korjaamisen ja estää integraatiovaikeudet tilanteessa, missä moni henkilö yhdistäisi pitkän aikavälin tuotokset samanaikaisesti.

Jatkuvaa toimitusta voidaan pitää jatkuvan integraation lisänä. Koodi käännetään, automaatio testataan, minkä jälkeen se pudotetaan testiympäristöön. Projektia ei vielä toteutusvaiheessa oltu julkaistu, mistä syystä puoluerekisterin jatkuva toimitus ei koske tuotantoa. Asiakkaalla on kuitenkin käytössä HYTE-ympäristö (hyväksymistestaus-ympäristö), jossa asiakas suorittaa toiminnallisuuksien hyväksymistestauksen. Kun testiympäristöön on kertynyt ja tarpeeksi muutoksia ja asiakas on hyväksynyt paikon, voidaan HYTE-ympäristö päivittää testiympäristön tasolle.

5.1 Docker

Projektia aloittaessamme, meillä oli valittavana, hyödynnämmekö virtuaalikoneita vai Dockeria kehityksessä. Docker on työkalu, joka antaa kehittäjille mahdollisuudet luoda, hallita ja toimittaa sovelluksia ympäristöstä riippumatta. [15.] Docker-saattaa ensisilmäyksellä muistuttaa virtuaalikoneita, kuten VirtualBoxia tai VMWarea, mutta Docker eroaa näistä sillä, että se ei itse luo virtuaalikonetta. Docker-teknologian pohjalla ovat kontit, jotka sisältävät valmiiksi koodin, kirjastot, käyttöjärjestelmän ja järjestelmäasetukset. Jokainen kontti pyörii lähes täydellisessä eristyksessä. Eri konttien sisäiset riippuvuudet eivät ole vaikutuksessa keskenään, mikä vähentää palvelimella konfliktien syntyä.

Suorituskyvyltään kontit pärjäävät vertailussa hyvin virtuaalikoneita vastaan. Erityisesti kontit loistavat käynnistysnopeudessa; virtuaalikoneella voi keskimäärin mennä kymmeniä sekunteja käynnistyä, kun taas konttiohjelma suoriutuu vastaavasta sekunneissa (ks. kuva 16). Virtuaalikoneiden heikkouksia on aina ollut resurssien allokoiminen. Silloin kun uusi virtuaalikone luodaan, sille annetaan käyttöön osa isäntä koneen resursseista, ja nämä resurssit ovat varattuja, vaikka virtuaalikone ei niitä käyttäisi. Kontit varaavat itselleen resursseja dynaamisesti käyttäen vain sen verran, kuin sillä hetkellä tarvitsevat. Tällainen optimaalinen resurssien jako vähentää isäntäkoneen kuormaa ja mahdollistaa suurenkin konttimäärän ajamisen. [16; 17.]

Teknologia	Käynnistysaika	Sammutusaika
Docker-kontit	< 50 ms	< 50 ms
Virtuaalikoneet	30-45 sec	5-10 sec

Kuva 16. Verrataan virtuaalikoneiden ja Dockerin käynnistymisnopeuksia [18].

Dockerin etuna myös on, että jokainen kehittäjä voi lokaalissa ympäristössään ajaa omaa versiota sovelluksesta kutsumalla konttien käynnistyskomentosarjaa. Tällä voidaan varmistaa, että jokaisella kehittäjällä on käytössä oma ympäristö, joka on sama kaikille kehittäjille ja on mahdollista palauttaa alkuasetuksiin nopeasti hakemalla kannan Docker-image uudestaan.

5.1.1 Docker-image

Docker-maailmassa on konttien lisäksi imageja. Docker-imaget ovat muuttumattomia mallineita, jotka sisältävät ohjeet, miten ja millä asetuksilla luoda kontteja. Kontti on siis imagesta rakennettu instanssi, joka on käynnistetty `docker run` -komennolla. Docker-imaget koostuvat monista kerroksista (ks. kuva 17). Docker-imagen voi tehdä alusta alkaen, mutta yleisintä on käyttää pohja-imagea. Imagen käyttäjä valitsee imagen, joka vastaa eniten käyttötarkoitusta, minkä jälkeen lisätään kustomointi. Se voi olla esimerkiksi web-palvelimen, kuten Nginx, lisäys tai muutos käynnistysskriptin toimintaan. Tällä säästetään aikaa, kun jokaisen kehittäjän ei tarvitse tehdä samoja konfiguraatioita.

Jokaisella Docker-imagella on kerros, jota voi muokata ja lukea. Tähän kerrokseen tehdään omat muutokset [19]. Kerroksia voi selata polusta `/var/lib/docker/aufs/diff` tai kirjoittamalla komennon `docker history` komentoriville [20]. Hyödyntämällä liitostiedostojärjestelmää, jokaista luotua kerrosta voi käyttää rajoittamaton määrä Docker-imageja. Esimerkiksi jos Ubuntu on yhdellä imagella pohjana, voivat muut imaget käyttää samaa Ubuntu-imagea. Näin voidaan vähentää huomattavasti levytilan tarvetta.

Kuva 17. Dockerin kerrosrakenteinen tiedostojärjestelmä [21].

5.1.2 Docker-maven-plugin

Docker-maven-plugin on liitännäinen Mavenille, jolla voidaan koota ja siirtää Docker imageja. Spotify kehitti liitännäisen vuonna 2014 käytettäväksi sisäisesti auttamaan Java imagejen koonnissa. Spotifyn kehittäjät tulivat itse lopulta siihen tulokseen, että on yksinkertaisempaa käyttää Dockerfile-tiedostoja, jotka vähentävät abstraktioiden ja konfiguraation määrää. Liitännäinen kuitenkin tukee Dockerfile-tiedostoja, joita me myös hyödynsimme.

Saadaksemme liitännäisen toimimaan haluamallamme tavalla täytyi meidän lisätä pom.xml-tiedostoon uuden liitännäisen asetukset. `ImageTagilla` pystymme hallinnoi-

maan imagen versioita. Käyttämällä projektin versiomuuttujaa pystymme yksittäistä arvoa muuttamalla nostaa imagen ja projektin versiota, jolloin kummatkin pysyvät ajan tasalla. Imagen nimi muodostuu `docker.image.prefix`-arvosta (10.188.13.135:5000/ork/services/pepu) ja artifact id:stä (pure). Koska käytämme Docker imagea, täytyy meidän myös määrittää polku `dockerDirectory`, jossa image sijaitsee. Haluamme myös, että image sisältää projektin JAR-tiedoston, niin sisällytämme sen `include`-tagilla (ks. kuva 18).

```
<plugin>
    <groupId>com.spotify</groupId>
    <artifactId>docker-maven-plugin</artifactId>
    <version>${docker-maven-plugin.version}</version>
    <configuration>
        <imageName>${docker.image.prefix}/${project.artifactId}</imageName>
        <dockerDirectory>src/main/docker</dockerDirectory>
        <resources>
            <resource>
                <targetPath>/</targetPath>
                <directory>${project.build.directory}</directory>
                <include>${project.build.finalName}.jar</include>
            </resource>
        </resources>
        <imageTags>
            <imageTag>${project.version}</imageTag>
        </imageTags>
    </configuration>
</plugin>
```

Kuva 18. Docker-maven liitännäisen konfiguraatio.

Liitännäinen ajaa kasausvaiheessa Dockerfilen, joka sisältää tarkempaa tietoa imagestamme. Tiedostoon määritetään, mitä käytämme pohjaimagenamme. `VOLUME`-komento kertoo Dockerille, mitkä kansiot säilyvät ennallaan uudelleenkäynnistyksen jälkeenkin ja mahdollistaa pääsyn kansiossa oleviin tiedostoihin kontin ulkopuolelta. `EXPOSE`-komennolla voimme kertoa Dockerille, mitä porttia kannattaa kuunnella ajoaikana. `EXPOSE` ei paljasta porttia itse, mutta se toimii dokumentaationa muille henkilöille, jotka saattavat imagea hyödyntää [22]. `ADD`– ja `RUN`-komennot siirtävät luodun JAR-tiedoston imageen, jotta se voidaan ajaa `ENTRYPOINT`-vaiheessa. `ENTRYPOINT`-komennolla määritetään se, mitä Docker ajaa silloin kun ajetaan `docker run` -komento. Puoluerekisterin tapauksessa haluamme, että `ENTRYPOINT` käynnistää projektin JAR-tiedoston (ks. kuva 19).

```
FROM openjdk:8u191-jre-alpine3.9
VOLUME /tmp
EXPOSE 8081
ADD *.jar /app.jar
RUN sh -c 'touch /app.jar'
ENV JAVA_OPTS=""
ENTRYPOINT [ "sh", "-c", "java $JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar /app.jar" ]
```

Kuva 19. Hallintapaneelin Dockerfile-konfiguraatio

5.1.3 Docker-compose

Puoluerekisteri koostuu monista konteista, mistä syystä niiden hallinnointi yksitellen olisi vaikeaa. Multikonttiapplikaatioita varten Dockerille on kuitenkin kehitetty docker-compose, jolla voidaan yhdellä komennolla luoda ja käynnistää monia kontteja samanaikaisesti [23]. Puoluerekisterin tapauksessa compose käynnistää samalla myös SAMPO-alustaan liittyvät kontit ja asettaa oikeat ympäristömuuttujat. Olennainen osa composea on myös porttien paljastaminen muiden konttien käyttöön ja niihin ulkomaailmasta yhdistäminen. Jokaisen yksittäisen kontin asetukseen määritetään, mitä kontin sisäisen portin halutaan vastaavan kontin ulkopuolisessa ympäristössä. Kehitysympäristön composessa asetamme Gateway-kontin sisäisen portin 443 vastaamaan porttia 16080. Gateway-kontti tarjoaa sähköisen asioinnin selainpuolen, mikä tarkoittaa, että yhdistämällä osoitteeseen `https://10.188.xx.xx:16080/` pääsemme applikaation etusivulle.

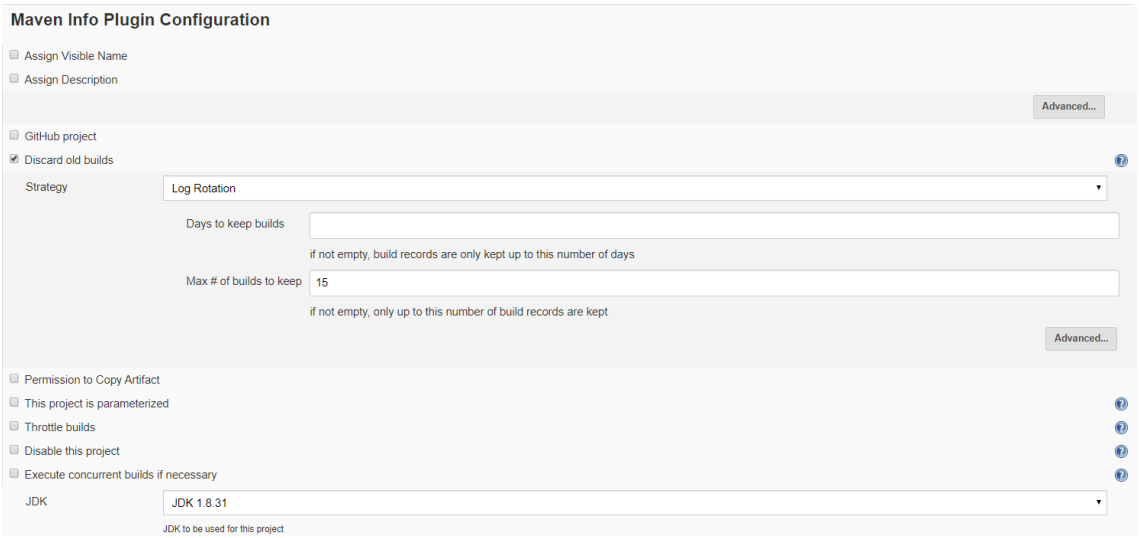
5.2 Jenkins

Jatkuvassa integraatiossa jokaisen masteriin menevän commitin jälkeen koodi kootaan ja automaatiotestataan. Kaikkien vaiheiden onnistumisen jälkeen voidaan ohjelma päivittää testiympäristöön. Tämä prosessi on jatkuvaa ja automaattista, mikä varmistaa, ettei toimimaton koodi pääse hajottamaan testiympäristöä. Tämänlainen toiminnallisuus vaatii kuitenkin työkalun, mikäli haluamme selvitä mahdollisimman vähällä konfiguroinnilla.

Päädyimme projektissa käyttämään avoimeen lähdekoodiin perustuvaa Jenkinsiä. Jenkinsiä pidetään toiminnallisuuksien, käyttäjämäärien ja luotettavuuden takia ykkösvaihtoehtona monille kehittäjille. Jenkins sopii moniin projekteihin tuhansien liitännäisten ja

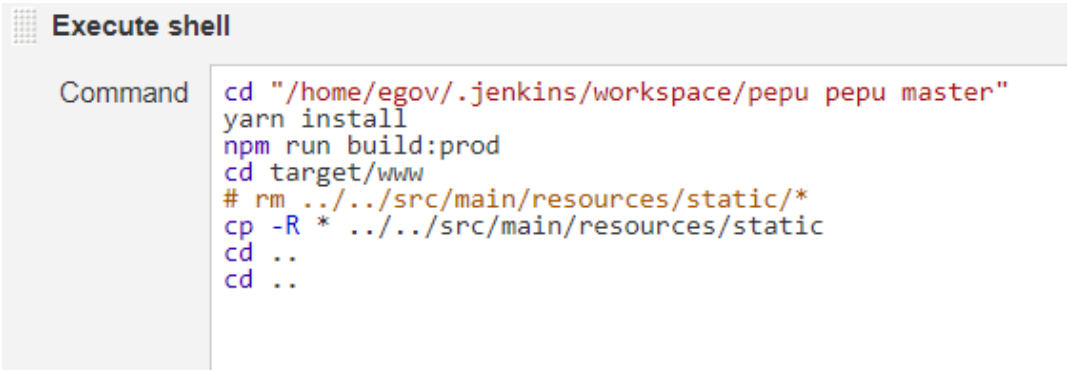
hyvän kustomoinnin ansiosta. [24.] Yksikössämme on myös muita projekteja, jotka hyödyntävät samaa Jenkins-instanssia, mikä tarkoitti, ettei meidän tarvinnut huolehtia Jenkinsin asennuksesta.

Jenkins-pipelinemme koostuu viidestä kasauksesta. Kolme kasausta pitää huolen siitä, että projekti kootaan joka kerta, kun repositoreihin tulee muutoksia. Projekti koostuu kolmesta Git-repositorysta: hallintopaneelista, sähköisestä asioinnista ja yhteisistä resursseista. Kun kehittäjä tekee muutoksia joihinkin näistä, aloittaa Jenkins toiminnan. Muutoksen tunnistamiseen käytämme Jenkins-liitännäistä Build Triggeriä, johon on mahdollista määrittää, kuinka usein Jenkins tarkistaa uudet muutokset. Olemme asettaneet sen jatkuvaksi nopeuttaaksemme prosessia. Toinen liitännäinen, jota käytämme, on Maven Info Plugin Configuration, johon voimme määrittää, milloin haluamme vanhojen kasauksien poistuvan (ks. kuva 20).



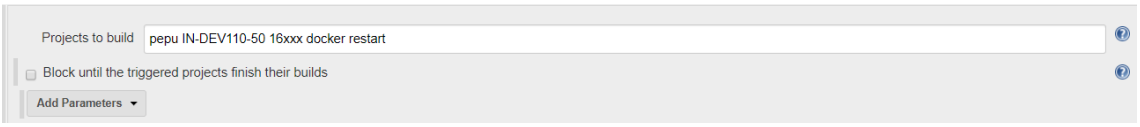
Kuva 20. Asetetaan kasausten maksimimäärä.

Kasauksen ensimmäisessä vaiheessa Jenkins siirtyy projektin polkuun ja ajaa komennon `yarn install`. Onnistuneen asennuksen jälkeen Jenkins kokoaa selainpuolen projektin tuotantomuotoon ja siirtää tiedostot staattiseen kansioon, mistä palvelinpuoli voi sen jakaa käyttäjille (ks. kuva 21).



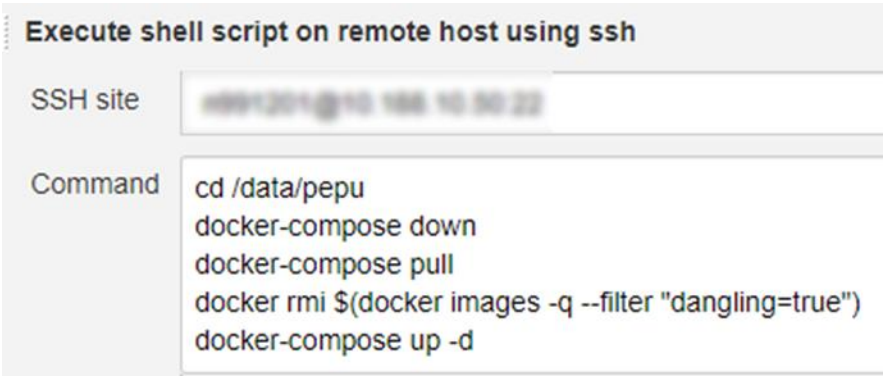
Kuva 21. Suoritettavat komennot kun Jenkins ajaa kokoamisen.

Mikäli koonnissa ei havaita ongelmia, muodostaa Jenkins JAR-tiedoston ja tallentaa sen yksikön artifactory-säilöön, mistä se voidaan hakea käyttöönottovaiheessa. Post Steps -liitännäinen tarjoaa eri mahdollisuuksia kasauksen jälkeisille toiminnoille. Onnistunut kasaus käynnistää seuraavan vaiheen, jossa uusi kasaus otetaan käyttöön testiympäristössä (ks. kuva 22).



Kuva 22. Määrittämällä arvo ”Projects to build”, voidaan asettaa seuraava kasauksen kohde.

Dockerin uudelleenkäynnistyskasausskripti ottaa aluksi SSH-yhteyden palvelimelle ja siirtyy oikeaan polkuun. Seuraavaksi `docker-compose down` ajaa alas kaikki kontit, jotta ne voidaan päivittää. `Docker-compose pull` hakee palvelimelta uudet versiot docker-imageista, mikäli versionumeroissa on ollut muutoksia. Välttyäksemme palvelimen kovalevytilan loppumiselta on hyvä ajaa `rmi`-komento parametrilla `dangling=true`, joka poistaa imaget, jotka eivät ole käytössä. Lopuksi ajamme komennon `docker-compose up`, joka uudelleenkäynnistää pysähtyneet kontit uusilla imageilla (ks. kuva 23).



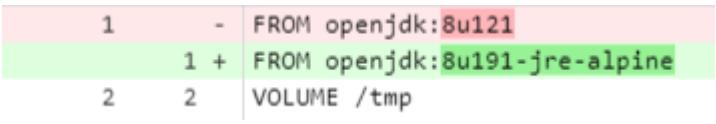
Kuva 23. Dockerin uudelleenkäynnistysskripti.

5.3 Alpine Linux

Kun aloitimme projektin, valitsimme openjdk:8u191-jre:n pohjaimageksi. 8u191-imagea on suositeltu käytettäväksi, jos ei ole varma siitä, mitä haluaa. 8u191-image on Debian-pohjainen Java 8 JRE image, joka sisältää monia käytettyjä kirjastoja, mistä syystä sen pitäisi soveltua monen eri applikaation ajoympäristöksi. Halusimme kuitenkin optimoida imagen kokoa ja poistaa tarpeettomia toiminnallisuuksia.

Alpine Linux on yksinkertainen ja yleiskäyttöinen tehokäyttäjille suunniteltu distribuutio. Suuri ero Alpinessa Debianiin verrattuna on musl-kirjaston käyttö glibc:n sijaan. Molemmat ovat implementaatioita standardoidusta libc:stä, joka on C-kirjasto. Linux-maailmassa libc tarjoaa mm. erilaisia funktioita, muistinhallintaa ja merkkijonojen käsittelyä. Glib on kahdesta käytetympi ja suorituskyvyltään tehokkaampi. Musl vie vähemmän tilaa ja on turvallisuusorientoituneempi, mistä syystä se sopii meidän tarkoituksiimme paremmin. [25.]

Päivittääksemme sovelluksemme käyttämään uutta imagea täytyy meidän aluksi muuttaa sähköisen asioinnin ja hallintapaanelin Dockerfilen pohjaimagea (ks. kuva 24).



Kuva 24. Muutos Dockefile-tiedostoon Gitissä

Käytämme operatiivisella puolella kirjautumiseen LDAP-verkkoprotokollaa. Sitä pääasiallisesti käytetään käyttöoikeuksien tarkistamiseen ja käyttäjien autentikoitiin. Kun siirryimme uusiin Alpine-pohjaisiin imageihin, päivitimme samalla uudempaan Java-versioon, mikä aiheutti ongelman käyttäjien autentikoinnissa. Uusi Java-versio toi mukanaan pakollisen päätepisteen tunnistamisen, mikä aiheuttaa SSL-virheen. Korjataksemme tämän lisäsimme meidän projektien docker compose -tiedostojen Java-muuttujiin arvon, joka poisti sen käytöstä (ks. esimerkkikoodi 7).

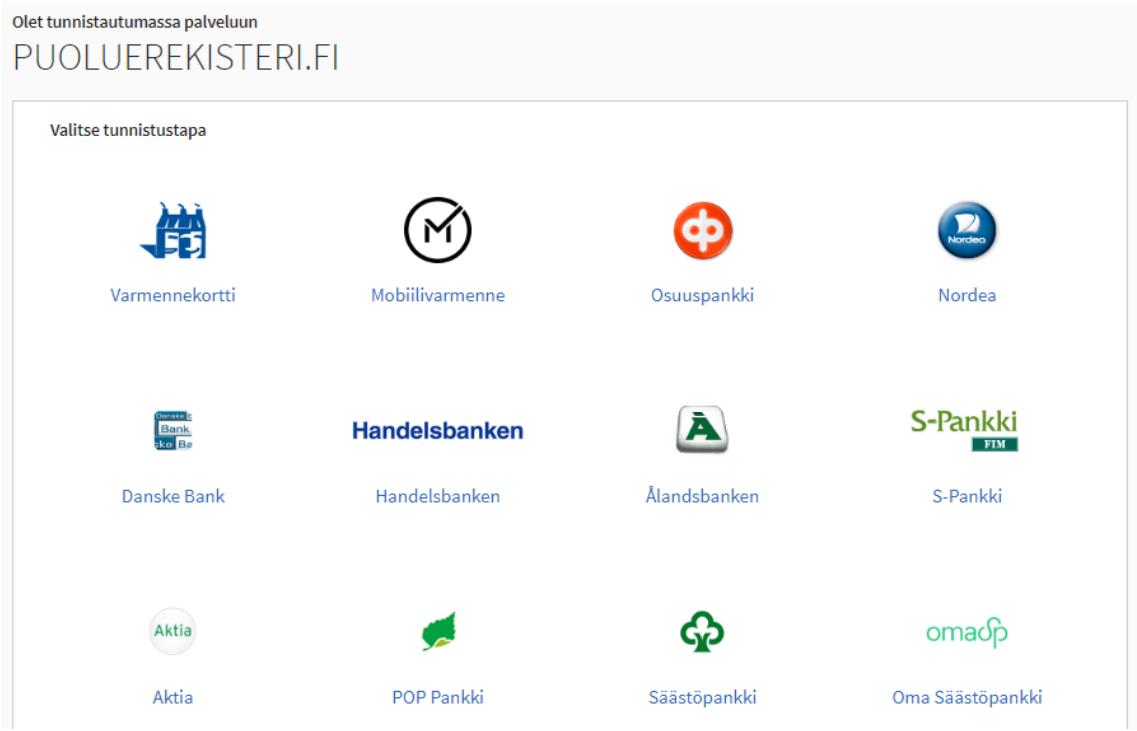
```
JAVA_OPTS=-Dcom.sun.jndi.ldap.object.disableEndpointIdentification=true
```

Esimerkkikoodi 7. Java-ympäristömuuttujan päivitys.

6 Integraatiot

6.1 Suomi.fi-tunnistautuminen

Sähköiseen asiointiin kirjaudutaan vahvasti tunnistautumalla SAMPO-alustan implementoimaa Suomi.fi -tunnistautumista käyttäen.



Kuva 25. Suomi.fi-palvelun tunnistautumissivu

Suomi.fi-palvelun sivustolta löytyy seuraavanlaista tietoa tunnistautumisesta [26]:

- Suomi.fi-tunnistuksen avulla suomalaisissa palveluissa voivat asioida Suomen ja muiden EU-maiden kansalaiset.
- Voit kirjautua kaikkiin julkishallinnon sähköisiin palveluihin, joiden käyttäminen edellyttää vahvaa tunnistamista.
- Kertakirjautumisen avulla voit käyttää kaikkia tunnistukseen liitettyjä palveluja ilman erillistä kirjautumista. Kirjautuminen on kerrallaan voimassa 32 minuuttia. Myös uloskirjautuminen tapahtuu samanaikaisesti kaikista asiointipalveluista. Kertakirjautuminen ei ole mahdollista eIDAS-tunnistuksessa.
- Tunnistus toimii tietokoneella ja mobiililaitteissa. Sitä voi käyttää eri selainohjelmilla ja niiden oletusasetuksilla.

6.2 LDAP

Käyttäjätunnukset ovat OM:n käyttäjähakemistossa (LDAP), josta saadaan myös käyttäjäroolit. Operatiiviseen perusjärjestelmään kirjautuminen tapahtuu oikeusministeriön omaa käyttäjähakemistoa (LDAP Pakka) käyttäen. LDAP eli Lightweight Directory Access Protocol on ohjelmistoprotokolla, jolla mahdollistetaan ihmisten ja organisaatioiden haun niin Internetin välityksellä kuin sisäverkossakin. LDAP-hakemisto on puumallinen, joka koostuu monista tasoista. [27.] Juurihakemisto haarautuu alueisiin, joka taas haarautuu organisaatioihin (Oikeusministeriö) ja yksiköihin (Puoluerekisteri) organisaation sisällä.

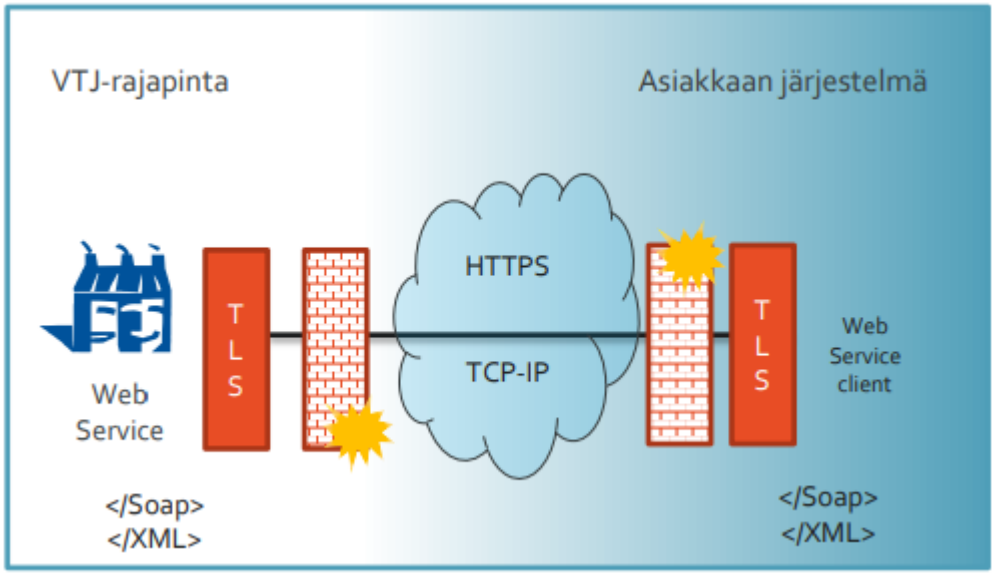
LDAP-tietojen hakuun käytetään JNDI-kirjastoa (Java Naming and Directory Interface). Koska kirjasto on Javan sisäinen, ei meidän tarvitse käyttää ulkoisia kirjastoja. Yhteys LDAP-palvelimelle konfiguroidaan hajautustaulun avulla. Hajautustauluun asetetaan ympäristömuuttujia, kuten käyttäjänimi, salasana ja LDAP-palvelimen osoite. Koska LDAP on puumallinen ja monta eri OM:n järjestelmää hyödyntää samaa käyttäjähakemistoa, täytyy meidän määritellä hakuehdot, joiden mukaan haemme vain oikean haaran tiedot. `Peoplepostfix`-arvo sisältää tiedon siitä, että haemme OM:n organisaatiosta henkilöitä, OM:n domainista Suomesta. Teemme näillä tiedoilla esihaun, joka palauttaa listan attribuutteja. Henkilön roolit haetaan erillisenä hakuna `getRoles()`-metodilla. Henkilön roolien haun yhteydessä määritämme järjestelmäksi Puoluerekisterin ja sen, että etsimme admin-roolisia käyttäjiä. Palautettavaan LdapResult-olioon haetaan attribuuteista käyttäjätiedot ja asetetaan aikaisemmin haetut roolitiedot (ks. kuva 26). Jos käyttäjää ei näillä ehdoilla löydy, lähettää järjestelmä virheviestin ja estää kirjautumisen.

```
public LdapResult hae(String ldapUsername, String ldapPassword) throws NamingException {
    LdapResult res = new LdapResult();
    try {
        Hashtable<String, Object> env = new Hashtable<>();
        env.put(Context.SECURITY_AUTHENTICATION, "simple");
        env.put(Context.SECURITY_PRINCIPAL, "uid=" + ldapUsername + "," + peoplepostfix);
        env.put(Context.SECURITY_CREDENTIALS, ldapPassword);
        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL, ldapAdServer);
        LdapContext ctx = new InitialLdapContext(env, null);

        Attributes attrs = getAttributes("uid", ldapUsername, ctx);
        res.setRoles(getRoles(ldapUsername, ctx));
        res.setId(getAttribute(attrs, idAttrId));
        res.setFirstName(getAttribute(attrs, "givenName"));
        res.setLastName(getAttribute(attrs, "sn"));
        res.setEmail(getAttribute(attrs, "mail"));
        res.setLoc(getAttribute(attrs, "omHomeOffice"));

        if (res.getRoles().isEmpty()) {
            throw new NamingException("LDAP_ERROR_NO_USER_NOT_FOUND");
        }
    } catch (NamingException ex) {
        ex.printStackTrace();
        throw ex;
    }
    return res;
```

Kuva 26. LDAP-hakuehtojen asetus ja vastauksen käsittely.

6.3 VTJ-kyselyrajapinta

VTJ-rajapinta on Väestörekisterikeskuksen ylläpitämä ja hallinnoima järjestelmä, joka mahdollistaa väestötietojen hakemisen XML-sanomina. Rajapinnan käyttäminen vaati asiakassovellukselta luotetun tahon myöntämän palvelinvarmenteen, sekä valmiuden ottaa vastaan tietoja HTTPS-protokollaa hyödyntäen TLS-protokollalla salattuna (ks. kuva 27). [28.]



Kuva 27. Kutsu saapuu VTJ-rajapintaan HTTPS-protokollaa käyttäen, minkä jälkeen se palauttaa asiakassovellukseen vastauksen XML-muodossa [28].

6.3.1 SOAP

VTJ-rajapinnan palvelukuvauksessa kerrotaan, että asiakassovelluksen tulee olla sovelluskyselyjä varten kykeneväinen tehdä HTTP(S) kutsuja WebService-rajapintaan käyttäen WebService-clienttia. Koska käytössämme oli jo Spring WebService, näimme eduksemme käyttää Springin tapaa tehdä SOAP-kutsuja. SOAP on XML:ään perustuva protokolla, jolla voi välittää dataa palvelimelta palvelimelle. REST-rajapintaan verrattuna SOAP on raskaampi ja on haastavampi ottaa käyttöön. SOAP on kuitenkin alusta- ja kieliriippumaton, mikä tarkoittaa, että asiakassovelluksen ei tarvitse huolehtia yhteensopivuusongelmista.

SOAP viesti muistuttaa tavallista XML-dokumenttia, joka koostuu neljästä osasta.

- Kirjekuoresta, joka kertoo, että viesti on SOAP-viesti.
- Valinnaisista otsikkotiedoista, jotka kertovat lisätietoa järjestelmälle.
- Viestin ohjelmakoodista, joka sisältää tiedon siitä, mitä haetaan.

- Tila- ja virheviesteistä, jotka kertovat viestin käsittelyn aikana tapahtuvista ongelmista.

SOAP-viesti, jonka PURE lähettää, on Java-olio, joka on muutettu XML:ksi. Vastaus, joka palaa VTJ-rajapinnasta, muutetaan myös XML:stä Java-olioksi. Luokkia varten VRK tarjoaa WSDL-tiedoston, joka kuvaa, minkälainen Web Service on. WSDL:stä voidaan myös generoida Java-luokkia, hyödyntäen maven-jaxb2-pluginia. Liitännäiselle määritellään polku, johon Java-luokat halutaan luoda, ja WSDL, joka käytetään generoinnin pohjana (ks. kuva 28). Generoinnin tuloksena saamme kaksi luokkaa `TeeHenkilonTunnusKysely`, jonka lähetämme VTJ-rajapintaan ja `TeeHenkilonTunnusKyselyResponse`, jonka rajapinta palauttaa.

```
<plugin>
  <groupId>org.jvnet.jaxb2.maven2</groupId>
  <artifactId>maven-jaxb2-plugin</artifactId>
  <version>0.13.3</version>
  <executions>
    <execution>
      <goals>
        <goal>generate</goal>
      </goals>
      <configuration>
        <schemaLanguage>WSDL</schemaLanguage>
        <generatePackage>vtj</generatePackage>
        <generateDirectory>C:\Users\himel.rahman\pe-work\pepu\pepu/target/generated/src/main/java</generateDirectory>
        <schemas>
          <schema>
            <url>src/main/resources/PERAVIA.wsdl</url>
          </schema>
        </schemas>
      </configuration>
    </execution>
  </executions>
  <configuration>
    <schemaLanguage>WSDL</schemaLanguage>
    <generatePackage>vtj</generatePackage>
    <generateDirectory>C:\Users\himel.rahman\pe-work\pepu\pepu/target/generated/src/main/java</generateDirectory>
    <schemas>
      <schema>
        <url>src/main/resources/PERAVIA.wsdl</url>
      </schema>
    </schemas>
  </configuration>
</plugin>
<plugin>
```

Kuva 28. Maven liitännäisen konfiguraatio, joka tekee Java luokat WSDL-tiedostosta.

6.3.2 Äänioikeus

Suurin syy hyödyntää VTJ-rajapintaa Puoluerekisterissä on äänioikeuden tarkistaminen. VTJ-rajapinta ei tarjoa suoraan tietoa, onko henkilö äänioikeutettu vai ei, vaan asiakassovelluksen tulee päätellä tämä itse rajapinnasta saaduista tiedoista (ks. kuva 29).

```
private static boolean hasVotingRight(Henkilo voter) {
    if (!stringToDate(voter.getSyntymaaika()).plusYears(18).isAfter(LocalDate.now())) {
        if (voter.getKansalaisuus().stream()
                .anyMatch(kansalaisuus -> kansalaisuus.getKansalaisuusE().equals("Finland"))) {
            return true;
        } else if (!notInFinland.contains(voter.getKotikunta().getKuntaS())
                && voter.getKansalaisuus().stream()
                        .anyMatch(kansalaisuus -> countries.contains(kansalaisuus.getKansalaisuusE()))) {
            return true;
        } else if (!notInFinland.contains(voter.getKotikunta().getKuntaS())
                && !voter.getKotikunta().getKuntasuhdeAlkupvm().equals("")
                && !stringToDate(voter.getKotikunta().getKuntasuhdeAlkupvm()).plusYears(2)
                        .isAfter(LocalDate.now())) {
            return true;
        } else if (!notInFinland.contains(voter.getKotikunta().getKuntaS())
                && voter.getAanioikeus().stream().anyMatch(aanioikeus -> electionType.contains(aanioikeus.getVaalilajiS())
                && (aanioikeus.getAanioikeusperustelaji().equals("30") || aanioikeus.getAanioikeusperustelaji().equals("40")))) {
            return true;
        }
    }
    return false;
}
```

Kuva 29. Java-metodi, joka tarkistaa henkilön äänioikeuden eri ehtojen kautta.

6.3.3 Henkilötiedot

Äänioikeuden lisäksi VTJ:stä saadaan henkilötiedot. Kun uusi puoluehakemus luodaan, niin hakemuksen tekijä lisää toiset asiamiehet, jotka hallinnoivat hakemusta. Mikäli hakemuksen tekijä syöttää virheellistä tietoa asiamiehistä luontivaiheessa, korjaa VTJ kyseisen henkilön tiedot, kun asiamies kirjautuu palveluun ja hyväksyy roolinsa.

6.4 VIA

Valtion integraatioalusta VIA on Valtorin kehittämä palvelu, jonka tarkoituksena on auttaa viestien välittämistä eri sovellusten välillä vähällä vaivalla ja valvotusti. VIA tukee suosituimpia rajapintoja kuten SOAP:ia ja REST:iä, mikä mahdollistaa erilaisten sanomamuotojen kommunikoinnin. Kutsujen epäonnistuessa, johtuen esimerkiksi väärästä polusta tai taustapalvelun ongelmista, osaa VIA palauttaa virhekoodin ja viestin, jota voi hyödyntää virheiden etsimiseen.

VIA:n käyttö vaatii minimissään kahden sovelluksen olemassaolon. Puoluerekisterin kohdalla kaikki kutsut, jotka lähtevät sähköisen asioinnista, menevät OM-hallintapaneeliin VIA:n kautta. Sähköisen asioinnin palvelinpuolelle on määritetty, mihin osoitteeseen kutsut lähetetään. Päätepisteenosoite muodostuu VIA-etuliitteestä, joka Puoluerekisterin kohdalla on ”puoluerekisteri/taustapalvelu”, ja itse resurssin osoitteesta, johon kutsu lopulta päätyy taustapalvelussa.

Kutsun saapuessa hallintapaneelin RESTful-rajapintaan reititetään kutsu oikealle kontrollerille, mikä suorittaa tarvittavat operaatiot ja palauttaa onnistuessaan HTTP-koodin 200 ja vastausviestin. Viesti palaa takaisin samaa reittiä VIA:n kautta sähköisen asioinnin palvelinpuolelle, jonka jälkeen data näytetään käyttäjälle selainpuolella.

VIA sisältää kolme eri ympäristöä: IT1-integraatioympäristön, QAT-hyväksymistestausympäristön ja PROD-tuotantoympäristön, jotka otetaan käyttöön järjestyksessä. Edellinen ympäristö tulee olla otettu käyttöön, ennen kuin voidaan siirtyä seuraavaan. Yhteyksien toiminnan kannalta on tärkeää, että Valtori ottaa ympäristöt käyttöön taustapalvelukohtaisesti ongelmien välttämiseksi. Palveluiden käyttöönottoon liittyy myös Valtorin sivulta ympäristökohtaisten varmenteiden lataaminen ja niiden lisääminen Javan truststoreen, joka mahdollistaa kommunikaation HTTPS:n avulla. VIA:n verkkopalvelin, johon kutsuja tehdään, lähettää takaisin kopion SSL/TLS-sertifikaatistaan. Tämän saatuaan Java tarkistaa, löytyykö kyseistä varmennetta truststoresta, jos sertifikaattia ei löydy, menee kutsu virheeseen. Kutsuihin tulee myös lisätä neljä headeria, jotka sisältävät kohdejärjestelmän ja lähdejärjestelmän tiedot.

Ensimmäinen käyttöönotettava ympäristö on IT1-integraatiotestaus. IT1-ympäristöä käytetään kehityksen aikana, sillä se ei vaadi SSH-tunnelia toimiakseen ja sillä varmistetaan, että järjestelmät voivat kommunikoida keskenään. Järjestelmän siirtyessä hyväksymistestaukseen, vaihdetaan QAT-ympäristöön. QAT vaatii SSH-tunnelin toimiakseen ja on toiminnoiltaan lähellä tuotannossa käytettävää ympäristöä. Näiden kahden ympäristön toimivuuden jälkeen ja tuotantoasennuksen aikana voidaan ottaa käyttöön tuotantoympäristö PROD. PROD on tarkoitettu vain tuotantoon, mikä viivyttää yhteyden kokeilemista tuotantoasennukseen asti. Ongelmien välttämiseksi on ensiarvoisen tärkeää varmistaa QAT-ympäristön toimivuus. [29.]

6.5 Sähköpostiviestit

Puoluerekisteri.fi-palvelu lähettää sähköposti-ilmoituksia monesta eri tapahtumasta liittyen puolueen perustamisprosessiin. Sähköposti-ilmoituksia lähetetään muun muassa puoluehakemuksessa listattujen yhdistyksen asiamiehille aina, kun hakemukseen tehdään muutoksia tai se etenee prosessin seuraavaan vaiheeseen. Sähköposti-ilmoituksia lähetetään myös Oikeusministeriön yleisiin sähköpostiosoitteisiin, joita hallinnoidaan

puoluerekisterin hallintapaneelissa. Näitä ilmoituksia lähetetään aina, kun puoluehakemus on edennyt vaiheeseen, jossa vaaditaan Oikeusministeriöltä toimia.

Sähköposti-ilmoitukset ovat oleellinen osa palvelun toiminnallisuuksia, sillä ne ovat ainoa tapa seurata hakemuksen edistymistä ja saada tietoa vaadittavista toimenpiteistä kirjautumatta palveluun.

Sähköpostiviestien sisällöt on tallennettu suomeksi ja ruotsiksi properties-tiedostoihin, joten niitä voidaan ylläpitää koskematta palvelun koodiin. Viestien dynaamiset osiot täydennetään sähköposteista vastaavassa Service-luokassa, joka on myös vastuussa sähköpostiviestien välityksestä valtion yhteiselle integraatioalustalle.

Sähköpostien lähetys on toteutettu VIA:n Messaging-rajapintaa hyödyntäen. Rajapinta käyttää SOAP-protokolaa, joten lähetysvaiheessa jokaisen sähköpostiviestin sisällöstä ja vastaanottajasta muodostetaan SOAP-kutsu, joka lähetetään rajapinnalle. Rajapinta palauttaa vastauksena viestin onnistuneesta tai epäonnistuneesta lähetyksestä, jonka järjestelmä kirjoittaa lokille.

6.6 Rajapinta

Palvelun rajapinta on toteutettu RESTful -rajapintana, joten se perustuu HTTP-kutsuihin ja on tilaton, eli ei vaadi palvelimella istunnon ylläpitämistä. Puoluerekisteri.fi-palvelun käyttöliittymä lähettää sähköisen asioinnin palvelimelle kutsuja JSON-muodossa, jotka edelleen lähetetään VIA-alustan kautta palvelun hallintapanelin palvelimelle, joka käsittelee itse ohjelmalogiikan ja muodostaa paluuvastauksen. Paluuvastaus välitetään vastaavasti VIA:n kautta takaisin sähköisen asioinnin palvelimen kautta takaisin käyttöliittymälle.

Sähköisen asioinnin palvelin liittää käyttöliittymän kutsuihin myös VIA:n vaatimat HTTP-otsikkotiedot (headers), sekä mahdollisen JWT-muotoisen autentikointitunnisteen, jonka se hakee evästeestä, mikäli käyttäjä on kirjautunut sisään palveluun.

Palvelun sähköisen asioinnin REST-rajapinta on jaettu kahteen osaan, julkiseen rajapintaan ja suojattuun rajapintaan. Julkinen rajapinta ei vaadi kirjautumista ja on kaikille

avoin. Sen kautta haetaan esimerkiksi rekisteröityneiden puolueiden tiedot, tai puoluehakemuksen kannatusmäärät.

Suojattu rajapinta vaatii, että käyttäjä on kirjautunut sisään, eli hänellä on voimassa oleva, SAMPO-kirjautumisalustan kautta muodostettu autentikaatiotunniste. Tunniste sisältää käyttäjän henkilötiedot, jotka saadaan Suomi.fi-kirjautumisen yhteydessä, sekä varmisteen ja tunnisteen vanhenemisajan. Palvelin validoi tunnisteen ja hakee sen sisältämät tiedot jokaisen suojatun kutsun yhteydessä. Jos validointi epäonnistuu, palvelin palauttaa HTTP-vastauksen 401 (Unauthorized). Suojattu rajapinta on myös rajoitettu käyttäjäroolin mukaan. Esimerkiksi jotain kutsuja voi suorittaa vain Väestörekisterikeskuksen henkilökunta.

Projektin vaatimuksiin kuului, että julkinen rajapinta tukisi paluuvastauksia JSON- ja XML-muodossa. Vaikka käyttöliittymä käyttää vain JSON-vastauksia, XML-vastaukset voivat olla hyödyllisiä esimerkiksi muille palveluntarjoajille tai kehittäjille, jotka haluavat käyttää puoluerekisteri.fi-palvelun rajapintaa omissa projekteissaan. Rajapinnan paluuvastaus voidaan määritellä joko lisäämällä osoitteen perään `format`-parametri halutulla formaattiarvolla, taikka käyttämällä `Content-Type` HTTP-otsikkotietoa.

Spring Boot -kirjastolla tehdyt rajapinnat palauttavat oletuskonfiguraatiolla vain JSON-dataa, joten XML-yhteensopivuus toteutettiin lisäämällä koodikantaan uusia datasiirto-olioluokkia (DTO) XML-annotaatioilla, joita käytetään vain, jos kutsu pyytää XML-dataa (ks. esimerkkikoodi 8). RestController-luokassa on myös erikseen metodit, jotka käyttävät oliomuunnosluokkaa (ObjectMapper), muuttamaan datasiirto-oliot XML-tiedoksi.

```
@XmlRootElement(name="parties")
@XmlAccessorType(XmlAccessType.FIELD)
public class PartyRestDTOListXml implements Serializable{

     private static final long serialVersionUID = 150486667455379602L;
     @XmlElement(name="party")
     protected List<PartyRestDTO> parties;
```

Esimerkkikoodi 8. Datasiirto-olioluokkia XML-annotaatioilla.

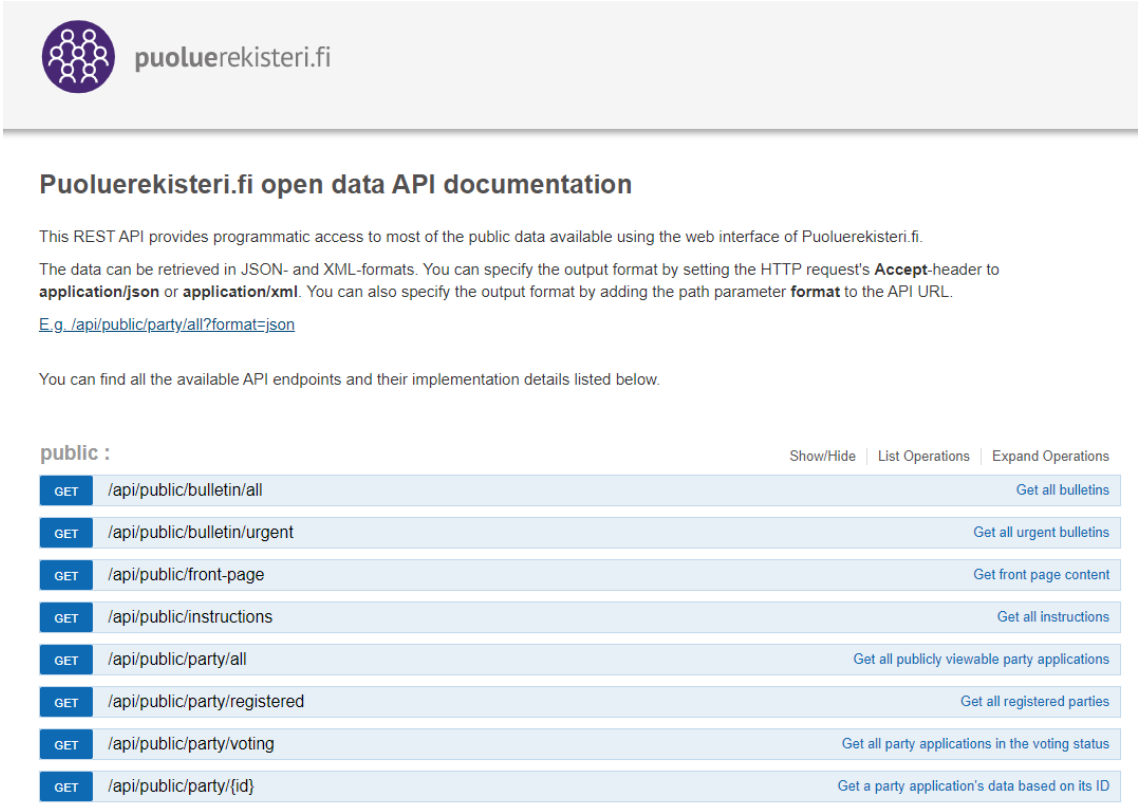
6.6.1 Swagger

Palvelun julkisen rajapinnan dokumentaatio on toteutettu Swagger ja Swagger UI-kirjastojen avulla. Swagger on avoimen lähdekoodin kirjasto, joka sisältää työkaluja auttamaan kehittäjiä suunnittelemaan, rakentamaan ja dokumentoimaan REST-rajapintoja [30]. Swaggerin Java-kirjasto toimii lisäämällä Spring Bootin RestController-luokkiin ja metodeihin Java-annotaatioita, joissa kuvataan kutsun parametreja ja paluuvastauksia, sekä kerrotaan kutsun tarkoituksesta (ks. esimerkkikoodi 9). Swaggerin konfiguraatioon voidaan myös lisätä yleinen kuvaus palvelun rajapinnasta. Swagger muodostaa oman rajapinnan, johon se kokoaa tietoa isäntärajapinnasta annotaatioiden ja konfiguraation perusteella.

```
@Api(tags = "public", description=" ", produces="application/json, applica-
tion/xml")

@ApiOperation(value = "Get all publicly viewable party applications", response
= PartyRestDTO.class, responseContainer = "List")
```

Esimerkkikoodi 9. RestController luokka Swagger-annotaatioilla

Swaggerin käyttöliittymä Swagger UI, kutsuu Swaggerin muodostamaa rajapintaa ja muodostaa käyttäjäystävällisen ja monipuolisen rajapintadokumentaation. Swagger UI on saatavilla myös valmiina Java-kirjastona, mutta koska puoluerekisteri.fi-palvelulle vaadittiin räätälöityä käyttöliittymää rajapintakuvaukselle, jouduimme muokkaamaan Swagger UI:n koodia ja lisäämään muokatut komponentit suoraan oman käyttöliittymän koodikantaan (ks. kuva 30).



Kuva 30. Swagger UI:llä toteutettu puoluerekisteri.fi-palvelun julkisen rajapinnan dokumentaatiosivu.

7 Toteutus

7.1 Puolueen perustamisprosessi

Puolueen perustamisprosessi kuvaa puoluerekisteri.fi-palvelun perustoiminnallisuuden ja sivuston pääkäyttötarkoituksen. Prosessi alkaa puoluehakemuksen täyttämisestä ja päättyy Oikeusministeriön lopulliseen päätökseen puolueen rekisteröimisestä. Prosessi kuvaa myös peruskansalaisen roolin puolueen perustamisessa (ks. liite 2).

Prosessi perustuu tulevaan lakimuutokseen, joten se oli asiakaan kannalta projektin tärkeimpiä seikkoja palvelun kehitysvaiheessa. Sen läpikäyntiin ja toiminnallisuuden testaamiseen käytettiin eniten aikaa asiakaspalavereissa sekä hyväksymistestausvaiheessa.

7.1.1 Hakemuksen täyttäminen

Puolueen perustamisprosessi alkaa, kun yhdistyksen asiamies kirjautuu palveluun käyttäen Suomi.fi-vahvaa sähköistä tunnistautumista. Kirjautuneena asiamies voi joko palvelun etusivulta tai valikosta siirtyä puoluehakemus sivulle.

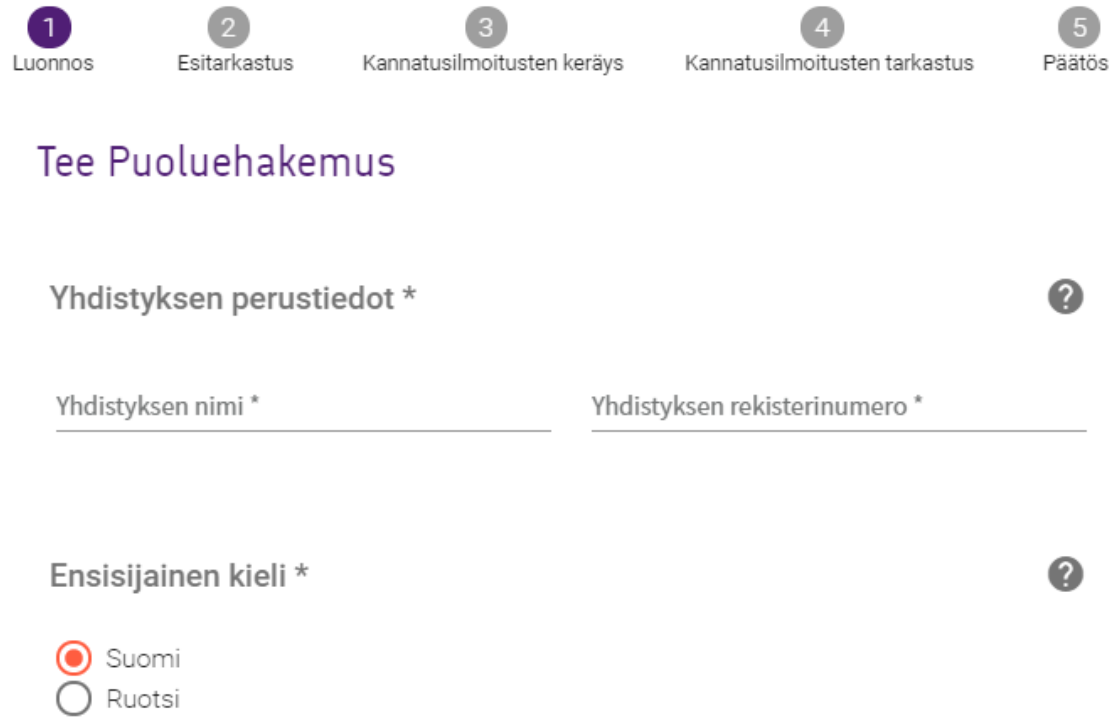
Asiamies täyttää puoluehakemukseen seuraavat tiedot:

- yhdistyksen nimi (pakollinen)
- yhdistyksen rekisterinumero (pakollinen)
- ensisijainen kieli (pakollinen)
- kieliversion lisääminen (kyllä/ei) (pakollinen)
- yhdistyksen esittely (jos kieliversio lisätty, niin esittely annetaan myös toisella kielellä)
- yhdistysrekisteriote (liitetiedosto) (pakollinen)
- yhdistyksen säännöt (liitetiedosto) (jos kieliversio lisätty, niin säännöt annetaan myös toisella kielellä) (pakollinen)
- yleisohjelma (liitetiedosto) (jos kieliversio lisätty, niin yleisohjelma voidaan antaa myös toisella kielellä) (pakollinen)
- yhdistyksen logo (liitetiedosto)
- yhteystiedot
    - osoite (pakollinen), (jos kieliversio lisätty, niin osoite ja toimipaikka voidaan antaa myös toisella kielellä)
    - postinumero (pakollinen)
    - toimipaikka (pakollinen), (jos kieliversio lisätty, niin osoite ja toimipaikka voidaan antaa myös toisella kielellä)
    - puhelinnumero (pakollinen)

- sähköpostiosoite (pakollinen)
- verkkosivu

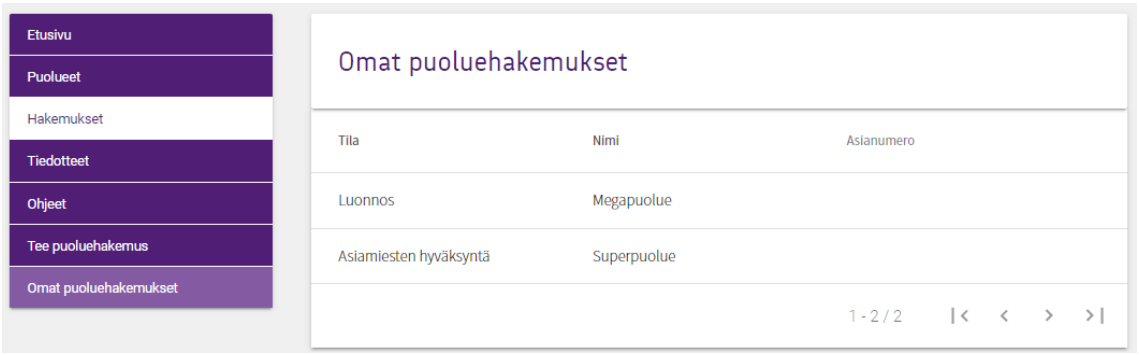
- jokaiselta asiamieheltä
  - nimi (pakollinen)
  - henkilötunnus (pakollinen)
  - sähköpostiosoite (pakollinen)
  - kotikunta (pakollinen)
  - valtuutus (nimenkirjoittaja tai valtuutettu asiamies)
  - asiamiehen valtakirja (pakollinen liitetiedosto, vain jos valtuutettu asiamies)

- paperisten kannatusilmoitusten keräys (kyllä/ei)



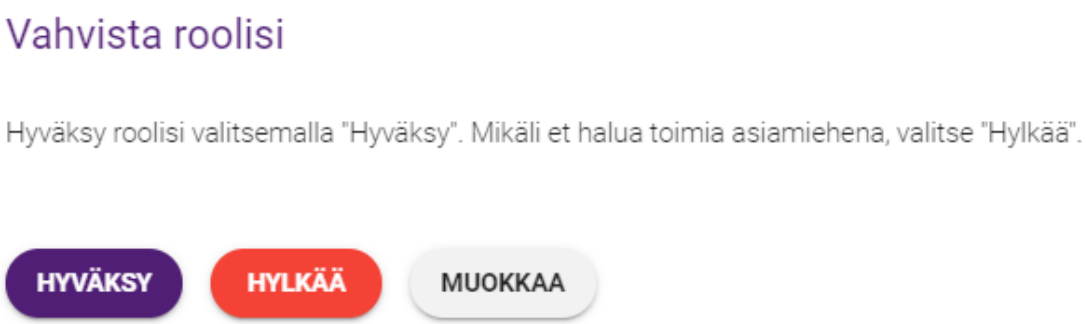
Kuva 31. Osittainen kuva tyhjästä puoluehakemuksesta

Apua hakemuksen täyttämiseen saa viemällä hiiren kursorin jokaisen kentän vieressä olevan kysymysmerkin päälle. Hakemuksen voi tallentaa luonnoksena missä vaiheessa tahansa, vaikka kaikkia kenttiä ei olisi vielä täytetty, jolloin asiamies voi esimerkiksi poistua palvelusta ja jatkaa hakemuksen täyttämistä toisella kertaa. Kun hakemus on täytetty ja siihen on lisätty vaadittava määrä muita asiamiehiä, hän voi lähettää hakemuksen. Hakemuksen lähettäminen lähettää samalla sähköposti-ilmoituksen hakemuksessa listatuille, yhdistyksen muille asiamiehille. Ilmoituksessa pyydetään heitä hyväksymään roolinsa yhdistyksen asiamiehenä tai muokkaamaan hakemusta.



Kuva 32. Listaus omista puoluehakemuksista

Samoin kuin alkuperäinen asiamies, yhdistyksen muut potentiaaliset asiamiehet kirjautuvat palveluun ja näkevät sivuvalikosta omat hakemuksensa, josta he pääsevät selaamaan kyseistä hakemusta (ks. kuva 32). Tässä vaiheessa muut asiamiehet voivat joko hyväksyä tai hylätä roolinsa, mistä lähtee sähköposti-ilmoitus muille roolinsa jo hyväksyneille asiamiehille, tai muokata hakemuksen tietoja (ks. kuva 33).



Kuva 33. Asiamiehen roolinvahvistusikkuna.

Jos tietoja päätetään muokata, niin kaikille asiamiehille lähetetään tästäkin asiasta sähköposti-ilmoitus, jossa pyydetään asiamiesroolin uudelleenhyväksyntää, myös alkuperäiseltä asiamieheltä.

Kuva 34. Roolinsa hyväksyneen asiamiehen näkymä hakemussivulla, kun hakemus odottaa toisen asiamiehen hyväksyntää.

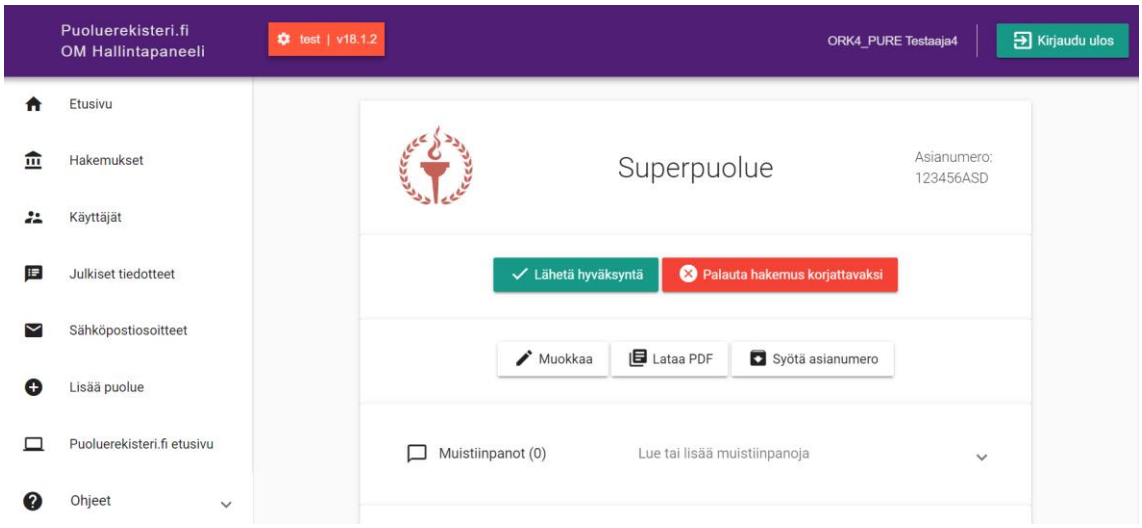
Prosessi jatkuu, kunnes vaadittava määrä asiamiehiä ovat hyväksyneet hakemuksen, jolloin kuka vaan heistä voi edelleen lähettää hakemuksen oikeusministeriön hyväksyntään. Tässä vaiheessa hakemuksen kentät ovat lukittuja ja asiamiehet eivät voi enää muokata sen tietoja, poissulkien uusien asiamiesten lisäyksen ja yhteystietojen muokkauksen.

7.1.2 Hakemuksen esitarkastus

Prosessin seuraava vaihe toteutetaan Oikeusministeriön puolesta palvelun hallintapaneelissa. Oikeusministeriön työntekijä kirjautuu hallintapaneeliin OM:n käyttäjähakemiston (LDAP) tunnuksilla ja näkee heti etusivulla, jos on uusia hakemuksia odottamassa esitarkastusta. Hän etenee puoluehakemusten listaussivulle selaamaan hakemuksia, ja suodattamaan listalta vain hyväksyntää odottavat hakemukset.

Kuva 35. Asiamiehen näkymä hakemussivulla, kun hakemus odottaa esitarkastusta.

Avattuaan hakemuksen OM-käyttäjä tarkastaa hakemuksen tietojen asiallisuuden ja oikeanmukaisuuden (ks. kuva 36). Jos hakemuksessa ilmenee puutteita tai muita ongelmia, niin OM-käyttäjä hylkää hakemuksen. Hakemusta hylätessä hän kirjaa järjestelmään hylkäyksen syyn ja vaadittavat muutokset, jotka tallennetaan hakemussivun muistiinpanoihin ja lähetetään hakemuksen jättäneen yhdistyksen asiamiehille sähköpostitse. Hylätty hakemus siirtyy tilaan `VAATII_MUUTOKSIA`, joka vastaa luonnostilassa olevaa hakemusta ja prosessi siirtyy takaisin edelliseen vaiheeseen, asiamiesten käsittelyyn.



Kuva 36. Puoluehakemuksen sivu OM-Hallintapaneelissa.

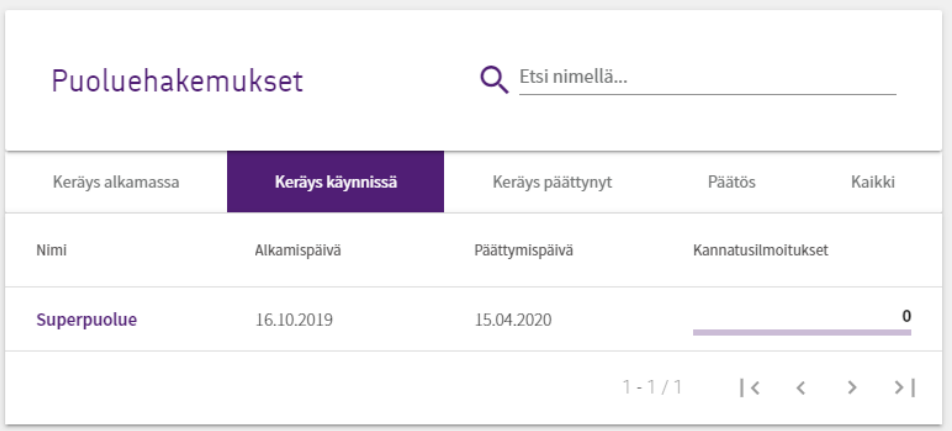
Jos OM-käyttäjä toteaa, että hakemuksen tiedot ja liitteet täyttävät kaikki vaatimukset, hän hyväksyy hakemuksen, jolloin asiamiehille lähetetään sähköposti-ilmoitus ja he voivat siirtää hakemuksen seuraavaan prosessin vaiheeseen.

7.1.3 Kannatusten keräys

Puoluehakemuksen edettyä Oikeusministeriön esitarkastuksen läpi yhdistyksen asiamiehet voivat käydä hakemuksen sivulta valitsemassa kannatusilmoitusten keräyksen alkamispäivämäärän.

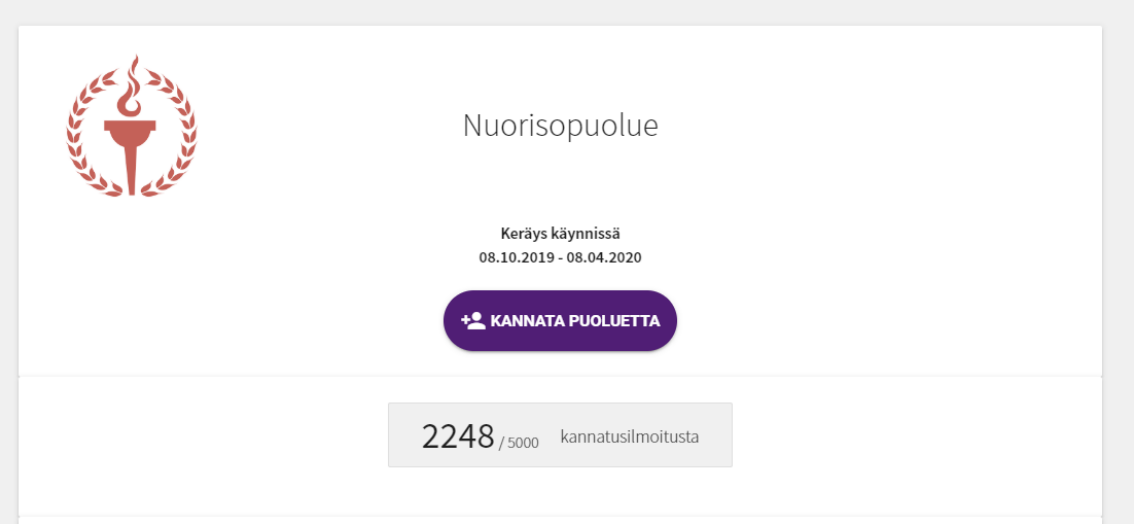
Kuva 37. Asiamiehen näkymä puoluehakemuksen sivulla, kun hakemukselle voi syöttää kannatusten keräämisen alkamispäivän.

Asiamiehet valitsevat päivämäärän, jolloin puoluehakemuksen sivu julkaistaan ja kannatusten keräys aloitetaan. Jos yhdistys on jo aloittanut paperisten kannatuskorttien keräyksen ennen sähköisten kannatusten keräystä, asetetaan päivämäärä menneisyyteen, siihen päivään, jolloin kannatusten keräys aloitettiin. Tällöin sähköisten kannatusten keräys aloitetaan saman tien ja puoluehakemus näkyy julkisesti kaikille sivuston vierailijoille.



Kuva 38. Osittainen kuvakaappaus palvelun puoluehakemuslistaussivusta.

Tavallisen kansalaisen ainoa syy kirjautua Puoluerekisteri.fi-palveluun on toisen puoluehakemuksen kannattaminen. Kannattaminen onnistuu menemällä kannatuksia keräävän puoluehakemuksen sivulle ja painamalla “Kannata”-nappulaa, joka edelleen ohjaa käyttäjän kannatussivulle (ks. kuva 39).



Kuva 39. Puoluehakemuksen julkinen sivu.

Kannattamisen edellytyksenä on, että käyttäjä on äänioikeutettu europarlamenttivaaleissa. Suomessa toimitettavissa europarlamenttivaaleissa äänioikeutettu on jokainen viimeistään vaalipäivänä 18 vuotta täyttänyt [31]:

a) Suomen kansalainen asuinpaikkaan katsomatta

b) muun EU:n jäsenvaltion kansalainen, joka ei ole menettänyt vaalikelpoisuuttaan kotivaltiossaan ja on viimeistään 80. päivänä ennen vaalipäivää ilmoittautunut maistraatille otettavaksi Suomen äänioikeusrekisteriin

   b1) jolla on kotikunta Suomessa 51. päivänä ennen vaalipäivää

   b2) joka on EU:n tai Suomessa toimivan kansainvälisen järjestön palveluksessa tai tällaisen henkilön perheenjäsen, joka asuu Suomessa.

Lisäksi lain mukaan kannattajakortissa, jonka kaavan oikeusministeriö on päätöksellään (479/2012) vahvistanut, tulee olla [32]:

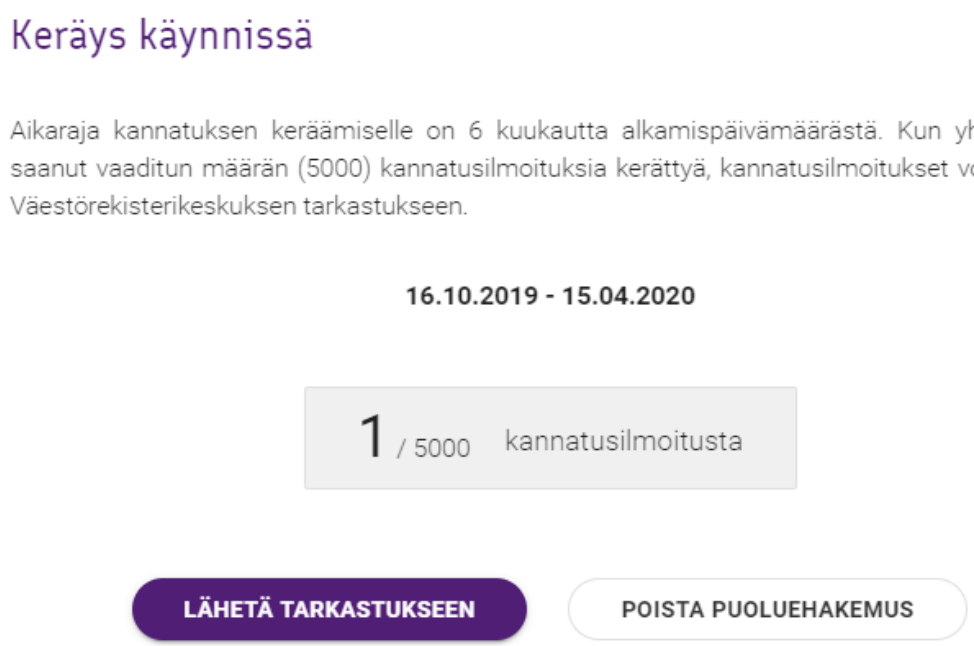
- kannattajan henkilötiedot
- kannattajan vakuutus siitä, että hän on äänioikeutettu eduskuntavaaleissa, kuntavaaleissa tai europarlamenttivaaleissa
- päivämäärä, joka ei saa olla vuotta vanhempi
- kannattajan omakätinen allekirjoitus.

Äänioikeus ja kirjautuneen käyttäjän tiedot haetaan VTJ-suorahakurajapinnan kautta ja käyttäjän henkilötiedot ja päivämäärä täydentyvät automaattisesti ilmoitukselle (ks. kuva 40). Järjestelmä ilmoittaa käyttäjälle, jos hänellä ei ole äänioikeutta. Kun käyttäjä on vahvistanut kannatuksensa, järjestelmä palauttaa hänet puoluehakemuksen sivulle ja päivittää sivun kannatuslaskurin ja taulukon.

Kuva 40. Sähköinen kannatusilmoitus.

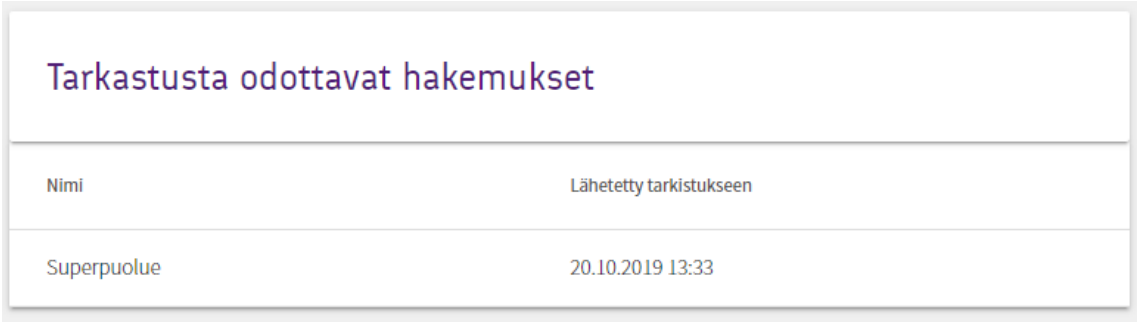
7.1.4 Kannatusilmoitusten tarkastus

Yhdistyksen saatua riittävästi kannatuksia, joko sähköisesti tai yhdessä paperisten kannattajakorttien kanssa (vaadittava määrä on kirjoitushetkellä 5000), se voi keskeyttää keräyksen ja lähettää kannatukset tarkastukseen (ks. kuva 41). Keräys keskeytetään ja kannatukset siirretään tarkastukseen myös automaattisesti, jos maksimikeräysaika on ylittynyt (kirjoitushetkellä 6 kk).



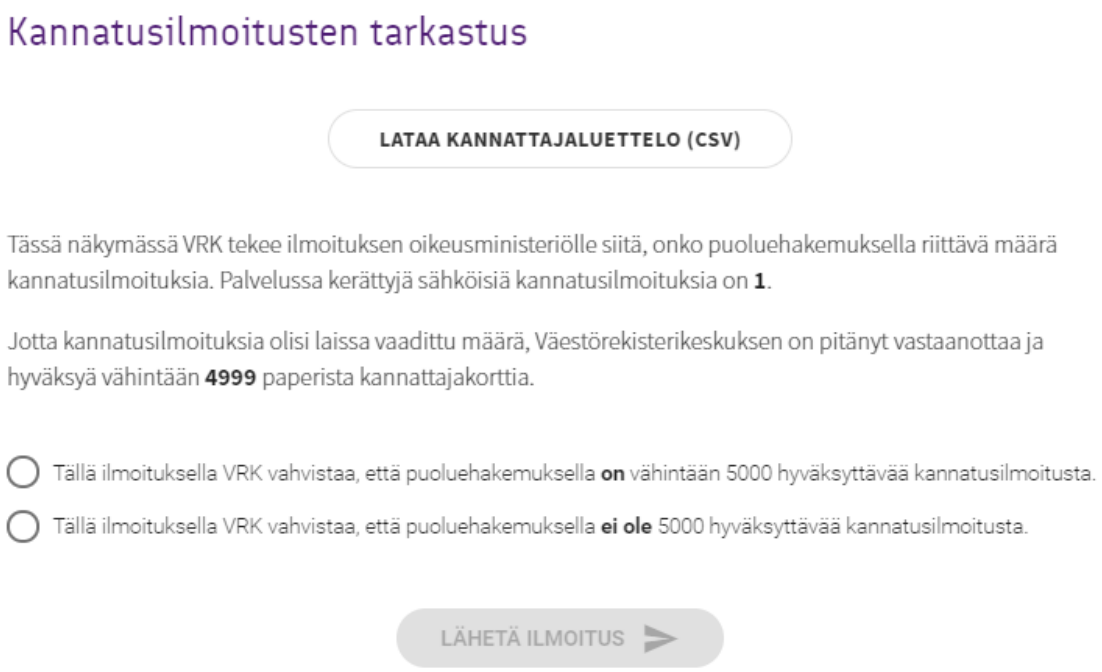
Kuva 41. Asiamiehen näkymä puoluehakemuksen sivulla, kun hakemus on kannatusten keräysvaiheessa.

Kannatusten tarkastus tapahtuu Väestörekisterikeskuksen toimesta. VRK-käyttäjä kirjautuu palveluun samalla tavalla kuin tavallinen kansalainen, mutta hänelle näkyy ylimääräinen sivu, johon on listattu kannatusten tarkastusta odottavat puoluehakemukset (ks. kuva 42). Hakemussivulla VRK-käyttäjällä on myös ylimääräinen valikko, josta hän voi ladata CVS-tiedoston. CSV-tiedostoon on listattu tietoa kaikista yhdistystä kannattaneista henkilöistä, kuten heidän kotikuntansa, nimet ja syntymäajat.



Kuva 42. VRK-käyttäjän näkemä lista tarkastusta odottavista hakemuksista.

VRK-käyttäjä validoi manuaalisesti jokaisen kannattajan tiedot ja laskee niiden yhteismäärän, mukaan lukien mahdolliset paperikannatuskortit, joita on toimitettu VRK:lle. Seuraavaksi hän vahvistaa kannatuksen määrän käyttöliittymällä ja järjestelmä lähettää vahvistusilmoituksen Oikeusministeriön hallintapaneelille ja Oikeusministeriön yhteiseen sähköpostiositteeseen (ks. kuva 43). VRK:n vahvistus näkyy myös yhdistyksen asiamiehille hakemussivulla.



Kuva 43. VRK-käyttäjän näkemä kannatusilmoitusten tarkastuslomake.

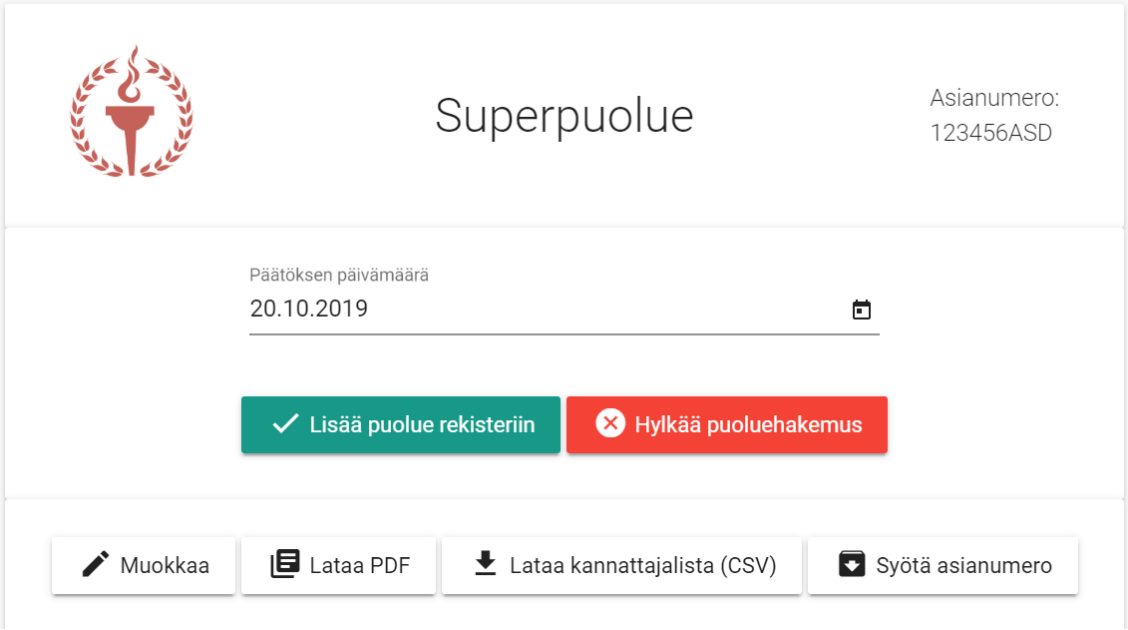
Tilanteessa, jossa vahvistettuja kannatuksia ei ole tarpeeksi, mutta keräysaikaa on vielä jäljellä, siirtyy hakemus takaisin keräykseen. Toisin, jos keräysaika on umpeutunut, puoluehakemus siirtyy takaisin Oikeusministeriön jatkokäsittelyyn VRK:n kielteisellä vahvistuksella.

7.1.5 Puolueen rekisteröinti

Prosessin viimeinen vaihe on yhdistyksen lisääminen puoluerekisteriin. OM-käyttäjät näkevät hallintapaneelin etusivulla ja puoluehakemuslistalla mahdolliset rekisteripäätöstä

odottavat hakemukset. OM-käyttäjät käyvät puoluehakemuksen sivulla ja käyvät vahvistamassa hakemuksen tiedot vielä viimeisen kerran. Hallintapaneelin kautta on myös mahdollista ladata kannattajalista CSV-tiedostona sekä PDF-tiedosto puoluehakemuksesta arkistointia varten.

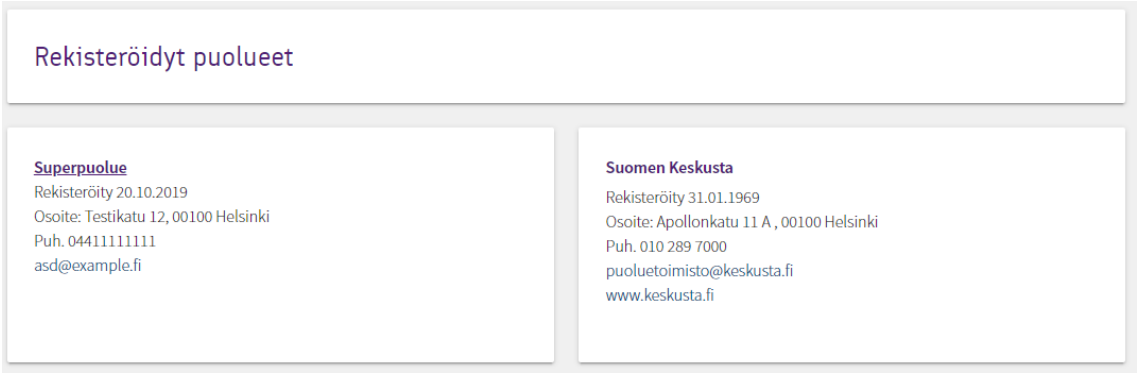
Koska varsinainen päätös puolueen rekisteröinnistä on voitu tehdä ennen varsinaista rekisteröintiä, Oikeusministeriön sisäisien prosessien takia, OM-käyttäjällä voi myös kirjata järjestelmään päätöksentekopäivän.



Kuva 44. Hallintapaneelissa näkyvä puolueen rekisteröintipäätöksen ikkuna.

Tilanteessa, jossa VRK on vahvistanut, että kannattajia on ollut liian vähän, tai hakemuksessa ilmenee tässä vaiheessa ongelmia, OM-käyttäjä hylkää puoluehakemuksen, eikä yhdistystä lisätä rekisteriin.

Toisin, jos VRK on vahvistanut kannattajamäärän riittävänä ja hakemuksen muut tiedot ovat kunnossa, niin yhdistys lisätään virallisesti puoluerekisteriin. Rekisteröidyt puolueet näkyvät palvelussa omalla sivullaan (ks. kuva 45).



Kuva 45. Puoluerekisteri.fi-palvelun “Rekisteröidyt puolueet”-sivu.

7.2 Hallinnolliset toiminnot

Puoluerekisteri.fi koostuu kahdesta osasta, jotka ovat sähköinen asiointi ja operatiivinen hallinnollinen puoli. Operatiivista puolta käyttää pääasiassa vain Oikeusministeriön työntekijät, jotka ovat puolueen rekisteröinnistä vastuussa. Heidän tehtäviinsä kuuluu sivuston ylläpitoa ja hakemusprosessin toiminnan varmistaminen.

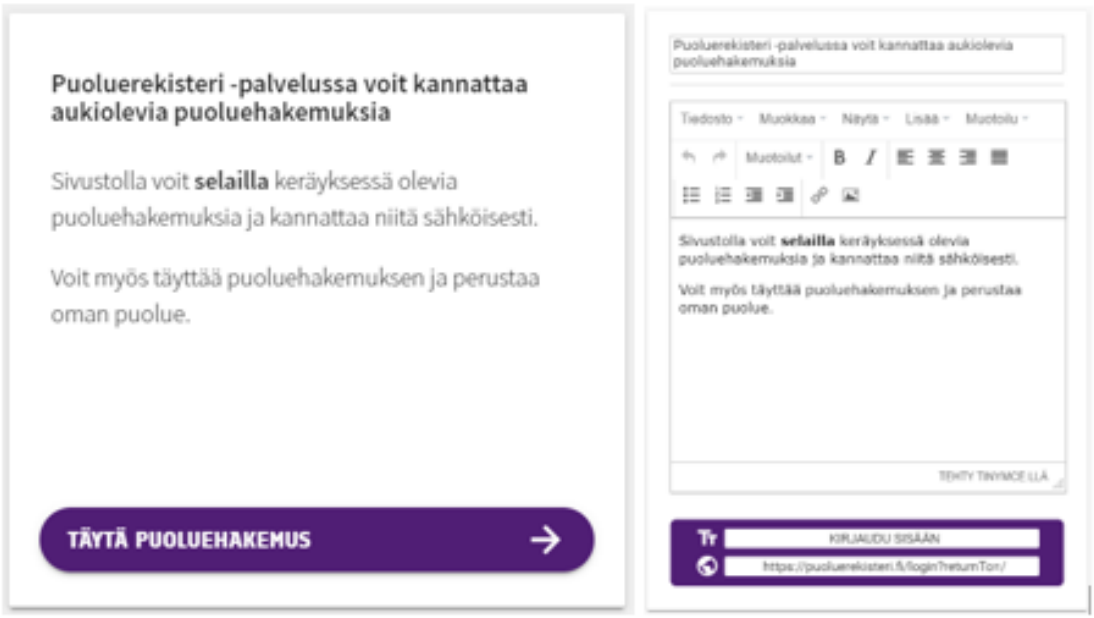
Niin kuin sähköistä asiointia operatiivista puolta käytetään myös web-käyttöliittymältä. Sivuvalikosta voi valita erilaisia toimenpiteitä tarpeen mukaan. Jokaisella käyttäjällä on oikeudet kaikkiin toimintoihin, mikä tarkoittaa, että turvallisuussyistä operatiiviseen puoleen pääsee vain sisäverkosta käsiksi.

7.2.1 Sivuston tekstien ylläpito

Pystyäkseen pitämään sivun tiedot ajan tasalla, OM-käyttäjillä tulee olla mahdollisuus päivittää sisältöä reaaliajassa. Rakensimme tietojen päivittämistä varten erikoisvalmisteisia komponentteja, joilla voidaan päivittää haluttuja tietoja.

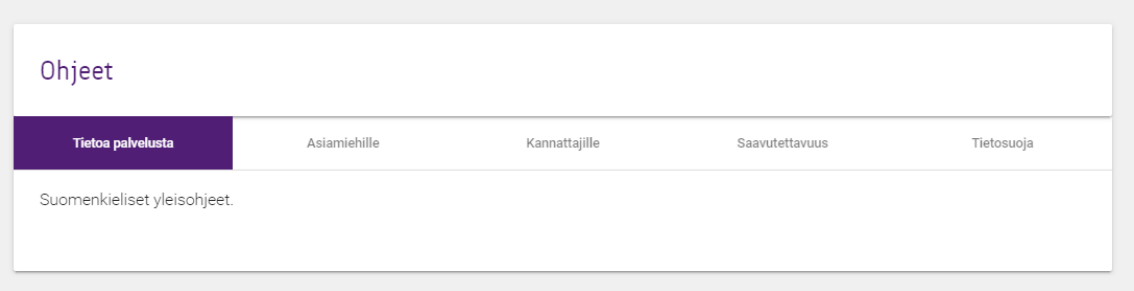
Etusivu ja ohjeet

Puoluerekisteri.fi palvelun etusivun suomen- ja ruotsinkieliset tekstit ja linkit ovat vapaasti muokattavissa OM-hallintapaneelissa. Tehdyt muutokset tallennetaan tietokantaan ja päivittyvät sivulle reaaliajassa. Etusivun sisällön päivittämiseen käytetään TinyMCE HTML -editoria.



Kuva 46. Vasemmalla on palveluun kirjautuneen käyttäjän näkemä, etusivun infolaatikko. Oikealla on hallintapaneelin etusivun infolaatikon sisällön editori.

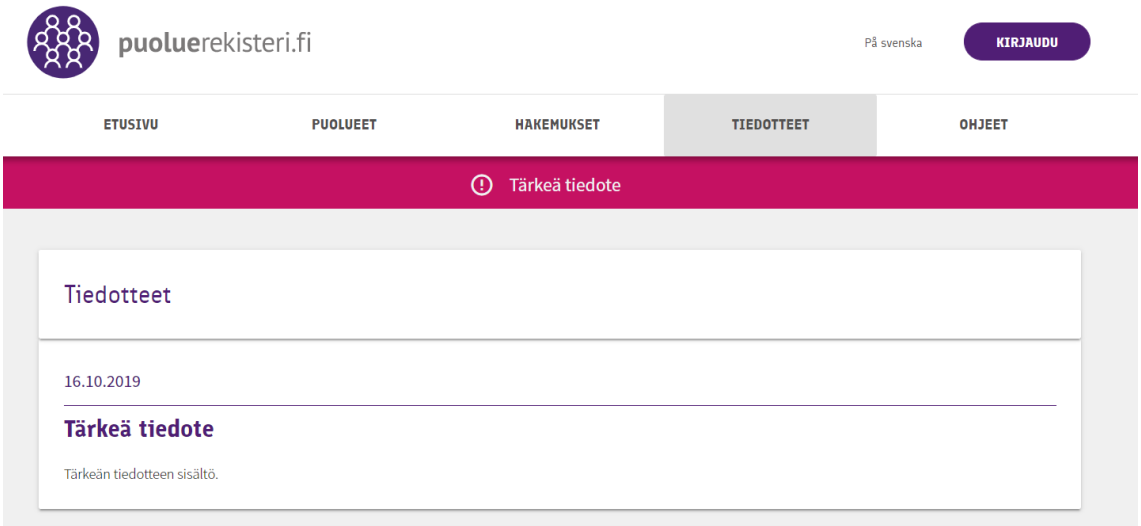
Palvelun ohjetekstejä voidaan myös muokata OM-hallintapaneelilta samalla tapaa kuin etusivun tekstejä. Ohjeisiin on mahdollista liittää myös esimerkiksi kuvatiedostoja TinyMCE-editorin ansiosta.



Kuva 47. Puoluerekisteri.fi -palvelun “Ohjeet”-sivu.

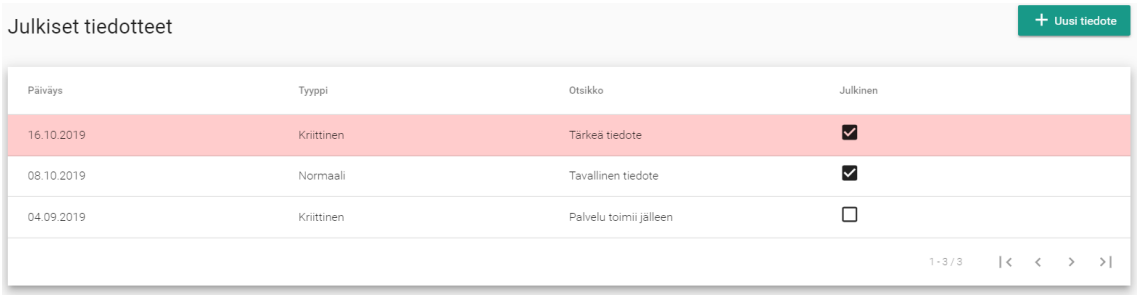
Tiedotteet

Tiedotteet toimivat Oikeusministeriön yleisenä viestintätapana Puoluerekisteri.fi-palvelun käyttäjille. Niitä on palvelussa kahdenlaisia: kriittisiä ja normaaleja. Normaalit tiedotteet näkyvät palvelun tiedotteet sivulla ja niiden tarkoitus on tiedottaa käyttäjille esimerkiksi järjestelmäpäivityksistä ja tulevista lakimuutoksista, jotka vaikuttavat palveluun.



Kuva 48. Puoluerekisteri.fi-palvelun “Tiedotteet”-sivu. Punaisella taustalla näkyy kriittinen tiedote.

Kriittisen tiedotteen tarkoitus on huomauttaa kaikille käyttäjille tärkeistä asioista, kuten palvelukatkoista. Ne näkyvät palvelun kaikilla sivulla, helposti huomattavana, punataustaisena bannerina. Painamalla banneria käyttäjä ohjataan tiedote sivulle, missä näkyy tiedotteen varsinainen sisältö (ks. kuva 48).

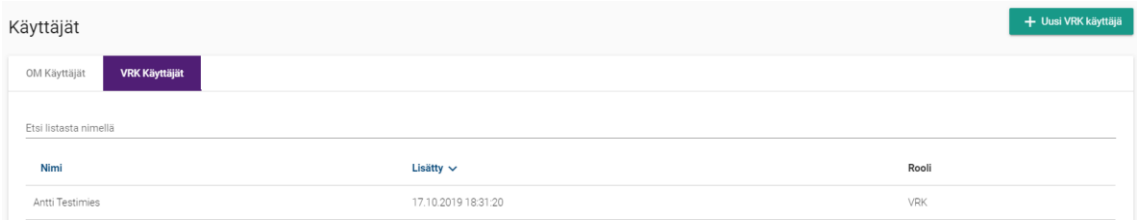


Kuva 49. Hallintapaneelilla näkyvä listaus tiedotteista.

Oikeusministeriön käyttäjä voi hallintapaneelin kautta poistaa, muokata tai luoda uusia tiedotteita. Uudelle tiedotteelle annetaan suomen- ja ruotsinkieliset otsikot ja sisällöt, sekä tiedotteen tyyppi ja näkyvyys. Tiedotelistauksessa näkyy ensin julkiset kriittiset tiedotteet punaisella taustalla, jotta OM-käyttäjät huomaisivat ne helposti, sitten muut julkiset tiedotteet (ks. kuva 49). Tiedotteita on myös mahdollista muokata, joten ne voidaan esimerkiksi piilottaa palvelusta, jos ne eivät ole enää relevantteja.

7.2.2 Käyttäjähallinta

Puoluerekisteri.fi-palvelua käyttävät ihmiset, joilla on eri rooleja. Operatiivisessa käyttäjienhallintapaneelissa on mahdollista muokata näistä kahta. VRK-käyttäjät ovat henkilöitä, jotka kirjautuvat sähköiseen asiointiin ja vahvistavat kannattajakorttien lukumäärän. Jotta järjestelmä pystyy tunnistamaan, ketkä ovat VRK:n edustajia, täytyy heidät lisätä järjestelmään ensiksi. Uudesta käyttäjästä syötetään hänen nimensä, henkilötunnus ja rooli (VRK). Järjestelmä tarkastaa, löytyykö syötetyllä henkilötunnuksella aikaisempaa käyttäjää.

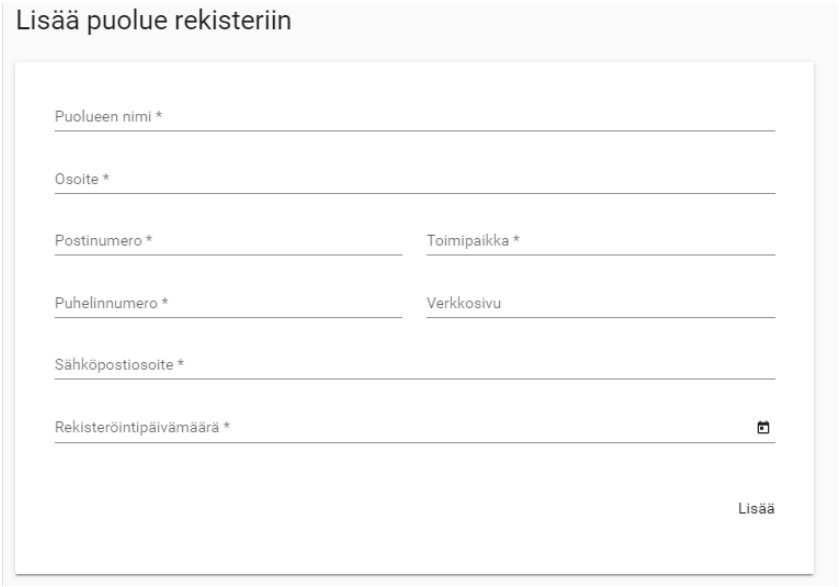


Kuva 50. Hallintapaneelin käyttäjähallintasivu VRK-käyttäjätvälilehdellä.

Oikeusministeriön käyttäjät on tallennettu heidän LDAP-palvelimellensa, mutta kun he ensimmäisen kerran kirjautuvat operatiiviselle puolelle heidän tietonsa kopioidaan myös Puoluerekisterin tietokantaan. OM-käyttäjiä ei manuaalisesti voi lisätä, mutta heidän tietojansa voi päivittää. Päivitettäviin tietoihin kuuluu käyttäjätunnus, salasana, etunimi, sukunimi, sähköpostiosoite ja käyttöoikeuden voimassaolo. Tärkeimpänä ominaisuutena voidaan pitää sähköpostin päivitystä, sillä järjestelmä lähettää ilmoitusluontaisia viestejä käyttäjille, jotka on tärkeä saada oikeaan sähköpostiosoitteeseen.

7.2.3 Puolueen lisäys

Puoluerekisterin käyttöönoton jälkeen sivustolle halutaan myös lisätä jo olemassa olevat puolueet ja ne puolueet, jotka keräävät kannattajakortteja ei-sähköisesti. Tätä varten käyttäjillä on mahdollisuus lisätä järjestelmään uusia puolueita lomakkeen kautta.



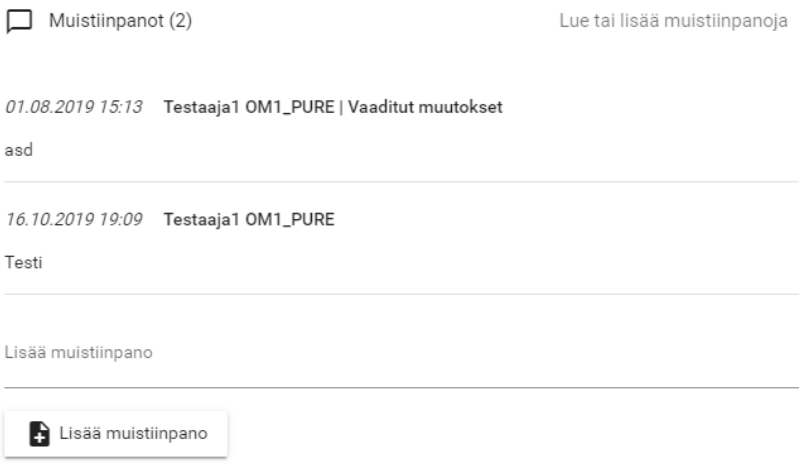
Kuva 51. Puolueen lisäys rekisteriin -näkymä.

Lomakkeessa tähdellä merkityt kentät ovat pakollisia tietoja. Kun puolue lisätään operatiiviselta puolelta käsin, on syötettyjä tietoja vähemmän tavalliseen hakemukseen verrattuna.

7.2.4 Muistiinpanojen lisäys

Hakemuksille on mahdollista lisätä muistiinpanoja puoluesivulla. Muistiinpanoista on hyötyä tilanteessa, missä eri henkilöt käsittelevät hakemusta ja haluavat helpottaa seuraavan käsittelijän työtä tiedon välityksellä. Esitarkastusvaiheen epäonnistumisen muistiinpanot lisätään automaattisesti tunnisteella `VAADITUT_MUUTOKSET`.

Muistiinpanot tallennetaan tietokantaa JSON-merkkijonona, johon lisätään loppuun uusi muistiinpano. Selainpuolella JSON puretaan objektiksi ja esitetään käyttäjälle. Muistiinpanosta tallennetaan sisällön lisäksi myös aikaleima ja henkilö, joka muistiinpanon on lisännyt.



Kuva 52. Muistiinpanon lisäys -näkymä. Näytetään aikaisemmin lisätyt muistiinpanot ja kenttä, johon voidaan syöttää uuden muistiinpanon teksti.

7.3 Yhteensopivuus

Tärkeä vaatimus palvelun suhteen oli sen yhteensopivuus erilaisten päätelaitteiden ja selainten suhteen. Selainyhteensopivuusvaatimus tuotti ongelmia lähinnä Internet Explorer 11 -selaimen suhteen, sillä se ei tue esimerkiksi kaikkia moderneja CSS- ja JavaScript-standardeja. JavaScript-ongelmat jouduttiin korjaamaan käyttämällä erilaisia polyfilleja, eli JavaScript-funktioita, jotka kääntävät yhteensopimattomat funktiot toimivan muotoisiksi. Esimerkiksi IE ei tue JavaScript-listojen `contains()`-funktiota, jolle anne-

taan parametrina arvo ja joka palauttaa totuusarvon riippuen siitä, löytyykö annettua arvoa listasta. Jouduimme lisäämään polyfillin, joka kääntsi `contains()`-funktion funktioksi, joka palauttaa totuusarvon, jos annetun arvon indeksi on yhtä kuin -1.

CSS-koodin yhteensopivuus oli myös hyvin vaikea toteuttaa Internet Explorer -selaimelle ja usein tyydyttiin käyttämään IE-spesifiä tyylitystä ongelman ratkaisemiseksi (ks. Esimerkkikoodi 10).

```
@media screen and (-ms-high-contrast: active), (-ms-high-contrast: none) {
      .home-info-box, .logged-home-box {
            display: block !important;
            button {
                  width: calc(1024px / 3 - 60px);
                  position: absolute;
                  bottom: 40px;
                  margin: 0 !important;
            }
      }
}
```

Esimerkkikoodi 10. CSS-tyylisääntöjä, jotka ovat voimassa vain Internet Explorer -selaimessa.

Meidän kehitystietokoneemme käyttivät Windows-käyttöjärjestelmää, joten Applen Safari-selaimen yhteensopivuuden testaamiseksi hyödynnettiin BrowserStack-web-ohjelmistoa, jonka kautta pystyimme testaamaan Puoluerekisteri.fi-sivustoa Safarilla virtuaali-macOS- ja iOS -alustoilla.

Päätelaiteyhteensopivuudella tarkoitetaan tässä kontekstissa sivuston responsiivisuutta, eli miten hyvin sivusto skaalautuu pienemmille näytöille ilman että käyttökokemus kärsisi. Tähän ominaisuuteen sijoitettiin paljon aikaa, sillä yli 50 % internetin käytöstä tapahtui mobiililaitteilla vuonna 2019 [33]. Osa skaalautuvuudesta toimi suoraan, sillä käytimme Materialize CSS-kirjaston taulukkojärjestelmää sivuston pohjana. Myös suurin osa käyttämistämme Material Angularin komponenteista skaalautuvat hyvin automaattisesti.

Kuva 53. Puoluerekisteri.fi-palvelun etusivu mobiililaitteella.

Oli kuitenkin monia tapauksia, joissa jokin komponentti ei toiminut hyvin pienillä näytöillä, joten se jouduttiin dynaamisesti vaihtamaan riippuen näytön koosta. Näytön koosta riippuvia CSS-tyylejä käytettiin myös runsaasti, ja sivuston ulkoasu onkin hyvin erilainen esimerkiksi mobiililaitteella, vaikka toiminnallisuudet ja käytettävyys pysyvät samalla tasolla. Esimerkiksi navigaatio ja yläpalkki on toteutettu eri tavalla usealla eri näytön koolla.

7.4 Asennus tuotantoon

Vaikka Puoluerekisteri ei ole menossa tuotantoon kuin vasta aikaisintaan kesällä 2020, voidaan asennus tuotantoympäristöön suorittaa silti aikaisemmin. Palvelimet ovat Tiedon puolelta, mistä syystä Tieto asentaa sovelluksen kehittäjien opastuksella. Kyseessä on moniosainen sovellus, joka vaatii paljon tuotantospesifejä konfiguraatiota, jonka takia kirjoitetut asennusohjeet ja kehittäjien saapuminen paikan päälle on tarpeen.

Asentajalle annettavat ohjeet sisältävät valmiiksi konfiguroidut docker-compose-, nginx- ja modsecurity-tiedostot, sekä asennuksen tarkat vaiheet sisältävän ohjeen. Asennusohjeiden pohjana käytettiin aikaisemman OM-projektin vastaavaa. Ohjeessa sähköinen asiointi on eriytetty operatiivisesta selkeyden vuoksi, sekä sähköisen asioinnin ohjeet koskevat molempia palvelimia.

Operatiivisen puolen asennusohje alkaa uuden tietokannan luonnilla. Asentaja luo uuden tietokannan nimellä `pure` ja uuden käyttäjän tietokantaan, mille annetaan kaikki oikeudet. Tämän jälkeen varmistetaan, että yhteydet docker-kontin IP-osoitteesta tietokantaan ovat sallittuja. Luodaan seuraavaksi hakemistorakenne (ks. kuva 54).

```
/data/pure
        | docker-compose.yml
        | --/security
                | pure_keystore.jks
                | pure_truststore.jks
```

Kuva 54. Hallintapaneelin hakemistorakenne.

Docker-compose.yml haetaan asennusohjeiden mukana tulleesta resurssikansiosta ja siirretään polkuun. Truststore ja keystore kopioidaan toisesta projektista ja nimetään uudelleen. Docker-compose.yml sisältää monia ympäristömuuttujia, joita tulee muuttaa asennuksen aikana. Näitä ovat:

- SERVER_SSL_KEY-STORE-PASSWORD
    - Kopioidun keystoren salasana.

- SERVER_SSL_TRUST-STORE-PASSWORD
    - Kopioidun trustoren salasana.
- HTTP_CLIENT_SSL_KEY-STORE-PASSWORD
    - Kopioidun keystoren salasana.
- HTTP_CLIENT_SSL_TRUST-STORE-PASSWORD
    - Kopioidun trustoren salasana.
- APPLICATION_FRONTEND_JWT-PUBLIC-KEY
    - Julkista avainta käytetään viestien allekirjoittamiseen ja lähettäjän tunnistamiseen. Avainpari generoidaan JWT-skriptillä, joka luo sekä julkisen että salaisen avaimen. [34.]
- SPRING_DATASOURCE_PASSWORD
    - Aikaisemmin luodun tietokantakäyttäjän salasana.
- APPLICATION_VTJ_SOSONIMI
    - VTJ-suorahakurajapinnan hakuparametri muutetaan, jos ei enää paikkaansa pitävä.
- APPLICATION_VTJ_USERNAME
    - VTJ-suorahakurajapinnan käyttäjätunnus.
- APPLICATION_VTJ_PASSWORD
    - VTJ-suorahakurajapinnan salasana.

Seuraavaksi muutetaan docker-compose.yml verkkokonfiguroinnin ip-osoitteiden X arvot (tarkistettava palvelimella seuraava vapaa X -> 192.168.X.0). Siirrytään samaan polkuun, jossa docker-compose.yml sijaitsee ja ajetaan `docker-compose up`. Seurataan lokeja ja tarkistetaan, ettei tule virheitä.

Seuraavat tehdään molemmille sähköisen asioinnin palvelimille DMZ-PR140-17 ja DMZ-PR140-18. Aloitetaan luomalla uusi keystore komentorivikäskyllä (ks. esimerkkikoodi 12).

```
keytool -genkey -alias server-alias -keyalg RSA -keystore pure_sa_keystore.jks
```

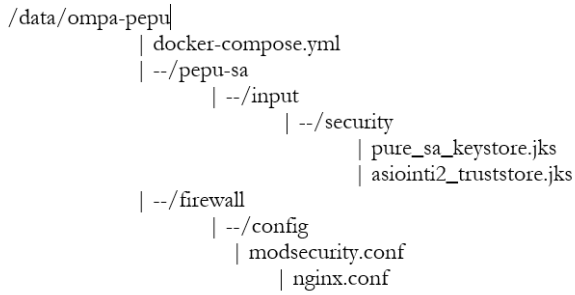
Esimerkkikoodi 11. Keystoren generoimiskomento.

Lisätään juuri generoituun keystoreen sertifikaatti komentorivikäskyllä (ks. esimerkkikoodi 13).

```
keytool -export -alias server-alias -storepass CHANGEIT -file CHANGEIT.cer -
keystore pure_sa_keystore.jks
```

Esimerkkikoodi 12. Sertifikaatin lisäys keystoreen komennolla.

Storepass `CHANGEIT`-arvo vaihdetaan edellisessä vaiheessa luotuun salasanaan ja CHANGEIT.cer-tiedoston nimi oikean sertifikaatin nimeksi. Tämän jälkeen luodaan seuraava hakemistorakenne (ks. kuva 55).



Kuva 55. Sähköisen asioinnin hakemistorakenne.

Haetaan kopioitavat tiedostot palvelimesta riippuen, joko ompa-pepu-17 tai ompa-pepu-18-kansiosta. Muutetaan docker-compose.yml-tiedostoista seuraavat ympäristömuuttujat.

- HTTP_CLIENT_SSL_KEY-STORE-PASSWORD
  - Aikaisemmin luodun keystoren salasanan.
- HTTP_CLIENT_SSL_TRUST-STORE-PASSWORD
  - Palvelimelta kopioidun truststoren salasana.

Seuraavaksi tarkistetaan SAMPO adminin portti ja sen perusteella muutetaan docker-compose.yml-tiedostoon admin kontin portti arvosta `changeit` oikeaan porttiin. Ajetaan `docker-compose up -d` ja seuraan lokeja virheiden varalta.

8 Yhteenveto

Insinöörityön tavoitteena oli kehittää verkkopalvelu sähköisten kannatusilmoitusten keräystä ja Suomen puoluerekisterin ylläpitoa varten. Puolurekisteri.fi tehdään osana demokratiapoliittista toimintaohjelmaa, jonka tavoitteena on antaa kansalaisille uusia osallistumis- ja vaikuttamismahdollisuuksia. Puoluerekisteri.fi on yksi 37 mukana olevasta hankkeesta. [35.]

Asiakkaan kanssa aluksi määritellyt tekniikat pysyivät lähes muuttumattomina koko projektin ajan. Projektin loppuvaiheilla suoritimme Linux-jakelupaketin vaihdon ja päivitimme Spring Bootin versioon kaksi. Tarpeen vaatiessa lisäsimme erinäisiä riippuvuuksia, kuten Swaggerin ja TinyMCE:n. Lisätyt kirjastot eivät vaatineet muutoksia järjestelmän arkkitehtuuriin, vaan ne mahdollistivat toimintojen lisäämisen vaivattomasti.

Projektin edetessä kasvatimme omaa substanssiosaamistamme, joka mahdollisti muutosten tekemisen nopeasti. Noudatimme Agile-toimintaperiaatteita ja pyrimme olemaan joustavia sovelluskehityksessä. Onnistuimme tässä mielestämme erinomaisesti, sillä

projekti valmistui kaksi kuukautta etuajassa odottamaan asennusta tuotantoympäristöön.

Saavutimme itsellemme asetetut ja asiakkaan meille antamat tavoitteet. Tämä oli meidän ensimmäinen työtehtävämme ja samaan aikaan oppimiskokemus. Opimme tärkeitä ohjelmointitaitoja käytetyistä tekniikoista ja taitoja soveltaa tietämystä muihinkin tehtäviin. Samalla tutuistamme julkishallinnon projektin kehittämiseen ja asiakkaan kanssa kommunikoimiseen.

Verkkosovelluksen kehittäminen oli meille tuttua aikaisemmista omista projekteista. Projektissa oli kuitenkin käytössä asiakkaan puolelta kehitysympäristöt, jotka toivat omat haasteensa yhteyksien muodostamisen osalta. Tuotantoasennuksessa oli myös mukana monia kolmansia osapuolia, joiden kanssa ripeä kommunikointi oli välttämätöntä. Tämä aiheutti asennuksen yhteydessä VIA-yhteyden konfigurointiongelmia, kun emme olleet perillä vaatimuksista.

Puoluerekisteri.fi-hankkeeseen liittyi kaksi optiota jatkokehityksen kannalta kaksi. Optio yksi sisälsi palvelun ylläpidon ja mahdollisen yleisen jatkokehityksen. Optio kaksi oli kokonaan uusi puoluerekisteri.fi-palveluun perustuva palvelu, jossa puolueen rekisteröinnin ja kannattamisen sijaan voitaisiin kannattaa ja perustaa valitsijayhdistyksiä, joiden tarkoituksina on asettaa ehdokkaita vaaleihin. Asiakas ei ole tehnyt vielä lopullista päätöstä optioiden suhteen, mutta olemme optimistisia, että saamme jatkaa palvelun kehitystä.

Lähteet

[1] 2 § (30.12.2015/1688) Puolueen rekisteröinti. (2015). Puoluelaki 10/1969. [verkkodokumentti] Saatavilla: https://www.finlex.fi/fi/laki/ajantasa/1969/19690010?search%5Btype%5D=pika&search%5Bpika%5D=puolue [Luettu 30.09.2019].

[2] Vänskä, O. (2011). Vain joka neljäs kriittinen palvelin on kahdennettu. [verkkodokumentti] Tivi. Saatavilla: https://www.tivi.fi/uutiset/vain-joka-neljas-kriittinen-palvelin-on-kahdennettu/a34bd5c3-78ba-3043-8dea-602e8cfd7434 [Luettu 23.10.2019].

[3] Gonigberg, A. (2018). Netflix/zuul Wiki. [verkkodokumentti] GitHub. Saatavilla: https://github.com/Netflix/zuul/wiki [Luettu 22.10.2019].

[4] Tyson, M. (2019). What is JPA? Introduction to the Java Persistence API. [verkkodokumentti] JavaWorld. Saatavilla: https://www.javaworld.com/article/3379043/what-is-jpa-introduction-to-the-java-persistence-api.html [Luettu 09.09.2019].

[5] Flyway by Redgate. (n.d.). *Database Migrations Made Easy*. [verkkodokumentti] Saatavilla: https://flywaydb.org/getstarted/how [Luettu 12.09.2019].

[6] Wilken, J. (2019). Angular in Action. Manning Publication Co., p.6.

[7] Angular.io. (n.d.). Routing & Navigation. [verkkodokumentti] Saatavilla: https://angular.io/guide/router [Luettu 11.09.2019].

[8] Daniels, P. and Atencio, L. (2017). RxJS in Action. Manning Publications Co., s.23.

[9] Reactivex.io. (n.d.). ReactiveX - Subject. [verkkodokumentti] Saatavilla: http://reactivex.io/documentation/subject.html [Luettu 11.09.2019].

[10] Milla, N. (2016). Open TinyMCE editor second row by default. [verkkodokumentti] Dev4Press. Saatavilla: https://www.dev4press.com/blog/tutorials/2016/open-tinymce-editor-second-row-by-default/ [Luettu 13.09.2019].

[11] Mindfire Solutions. (2018). Best Java IDEs 2018 | Most Popular Java IDE. [verkkodokumentti] Saatavilla: http://www.mindfiresolutions.com/blog/2018/05/best-java-ides-2018/ [Luettu 08.09.2019].

[12] Stack Overflow. (2019). Stack Overflow Developer Survey 2019. [verkkodokumentti] Saatavilla: https://insights.stackoverflow.com/survey/2019 [Luettu 09.09.2019].

[13] Sealights. (2019). 6 CI/CD Tools to Build Your Next-Gen Delivery Pipeline. [verkkodokumentti] Saatavilla: https://www.sealights.io/software-development-metrics/6-ci-cd-tools-to-build-your-next-gen-delivery-pipeline/ [Luettu 16.10.2019].

[14] Melonfire, C. (2006). Understanding the pros and cons of the Waterfall Model of software development. [verkkodokumentti] TechRepublic. Saatavilla: https://www.techrepublic.com/article/understanding-the-pros-and-cons-of-the-waterfall-model-of-software-development/ [Luettu 15.10.2019].

[15] Kobos, J. (2018). When and Why to Use Docker. [verkkodokumentti] Linode Guides & Tutorials. Saatavilla: https://www.linode.com/docs/applications/containers/when-and-why-to-use-docker/ [Luettu 15.10.2019].

[16] Schlosser, H. (2018). Docker vs. Virtual Machine: Where are the differences?. [verkkodokumentti] DevOps Conference. Saatavilla: https://devopsconference.de/blog/docker/docker-vs-virtual-machine-where-are-the-differences/ [Luettu 15.10.2019].

[17] Tozzi, C. (2019). Docker vs. Virtual Machines: Understanding the Performance Differences. [verkkodokumentti] Channel Futures. Saatavilla: https://www.channelfutures.com/technologies/docker-vs-virtual-machines-understanding-the-performance-differences [Luettu 15.10.2019].

[18] Runnable Docker Guides. (n.d.). Why use Docker?. [verkkodokumentti] Saatavilla: https://runnable.com/docker/why-use-docker [Luettu 16.09.2019].

[19] G, J. (2016). Digging into Docker layers. [verkkodokumentti] Medium. Saatavilla: https://medium.com/@jessgreb01/digging-into-docker-layers-c22f948ed612 [Luettu 16.10.2019].

[20] Rouse, M. (2019). What is Docker image?. [verkkodokumentti] SearchITOperations. Saatavilla: https://searchitoperations.techtarget.com/definition/Docker-image [Luettu 23.10.2019].

[21] Agarwal, N. (2017). Docker Container’s Filesystem Demystified. [verkkodokumentti] Medium. Saatavilla: https://medium.com/@nagarwal/docker-containers-filesystem-demystified-b6ed8112a04a [Luettu 17.09.2019].

[22] Docker Documentation. (2019). Dockerfile reference. [verkkodokumentti] Saatavilla: https://docs.docker.com/engine/reference/builder/ [Luettu 16.10.2019].

[23] Docker Documentation. (2019). Overview of Docker Compose. [verkkodokumentti] Saatavilla: https://docs.docker.com/compose/ [Luettu 16.10.2019].

[24] Stackify. (2017). Top Continuous Integration Tools: The 50 Best CI & Continuous Delivery Tools. [verkkodokumentti] Saatavilla: https://stackify.com/top-continuous-integration-tools/ [Luettu 23.10.2019].

[25] Etalabs.net. (n.d.). Comparison of C/POSIX standard library implementations for Linux. [verkkodokumentti] Saatavilla: http://www.etalabs.net/compare_libcs.html [Luettu 15.10.2019].

[26] Suomi.fi. (n.d.). Näin tunnistautuminen toimii. [verkkodokumentti] Saatavilla: https://www.suomi.fi/ohjeet-ja-tuki/tietoa-tunnistuksesta/nain-tunnistautuminen-toimii [Luettu 10.09.2019].

[27] Rouse, M. (2019). What is the Lightweight Directory Access Protocol (LDAP)?. [verkkodokumentti] SearchMobileComputing. Saatavilla: https://searchmobilecomputing.techtarget.com/definition/LDAP [Luettu 01.10.2019].

[28] Väestörekisterikeskus. (2019). VTJ-palvelukuvaus. [verkkodokumentti] Saatavilla: https://vrk.fi/documents/2252790/13496803/VTJ-rajapinta+palvelukuvaus+15.2.2019.pdf/d148c69c-6f45-7478-5185-ebcd7b9c1df5/VTJ-rajapinta+palvelukuvaus+15.2.2019.pdf [Luettu 18.09.2019].

[29] Valtori. (n.d.). Integraatiopalvelut valtionhallinnolle. [verkkodokumentti] Saatavilla: https://valtori.fi/integraatiopalvelut [Luettu 23.10.2019].

[30] Swagger.io. (2019). What is Swagger. [verkkodokumentti] Saatavilla: https://swagger.io/docs/specification/2-0/what-is-swagger/ [Luettu 18.10.2019].

[31] 2§ Äänioikeus. (1998). Vaalilaki 714/1998. [verkkodokumentti] Saatavilla: https://www.finlex.fi/fi/laki/ajantasa/1998/19980714#a714-1998 [Luettu 30.09.2019].

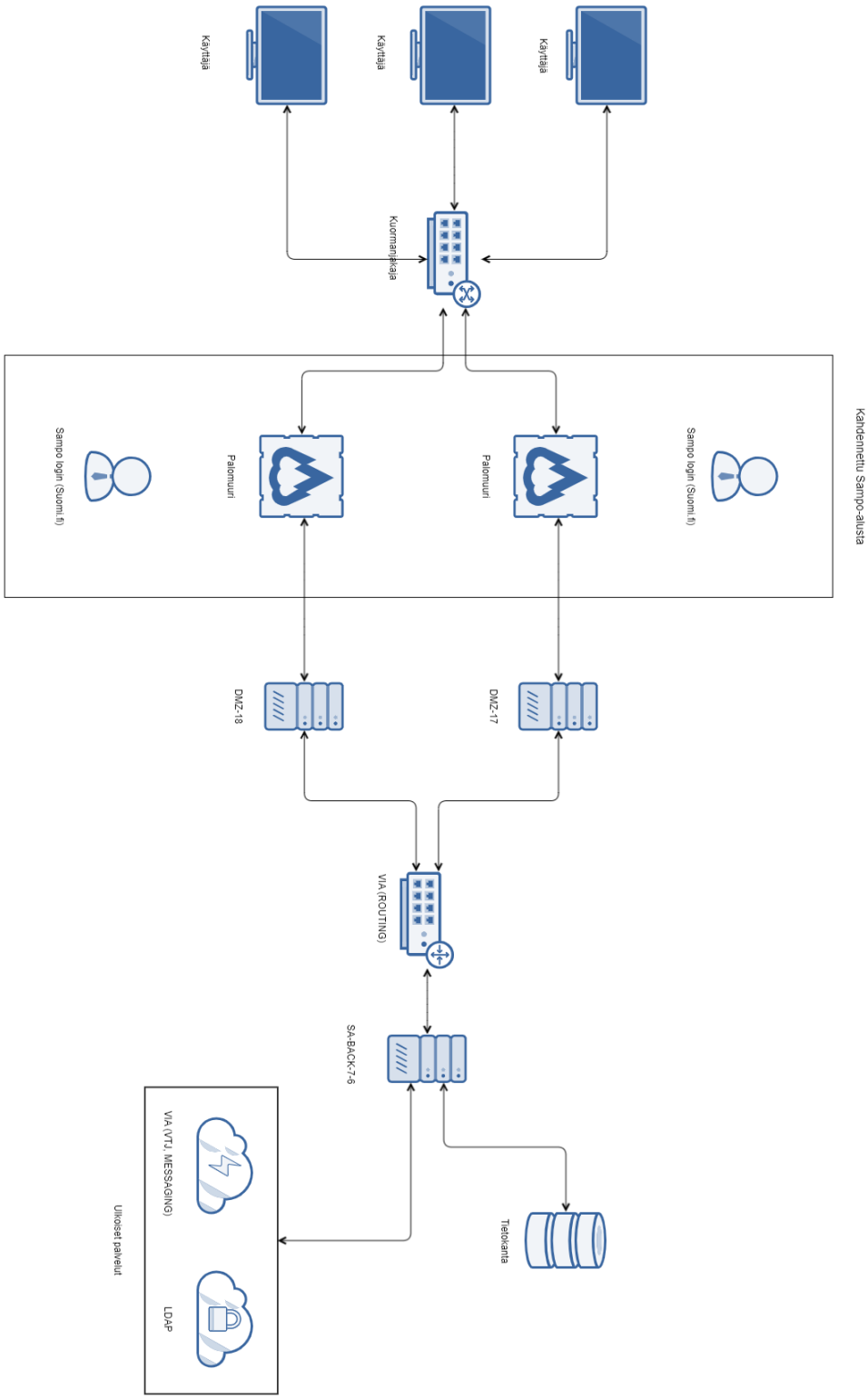
[32] Vaalit. (n.d.). Puolueen perustaminen. [verkkodokumentti] Saatavilla: https://vaalit.fi/puolueen-perustaminen [Luettu 14.10.2019].

[33] Enge, E. (2019). Mobile vs Desktop Traffic in 2019. [verkkodokumentti] Saatavilla: https://www.perficientdigital.com/insights/our-research/mobile-vs-desktop-usage-study [Luettu 14.10.2019].

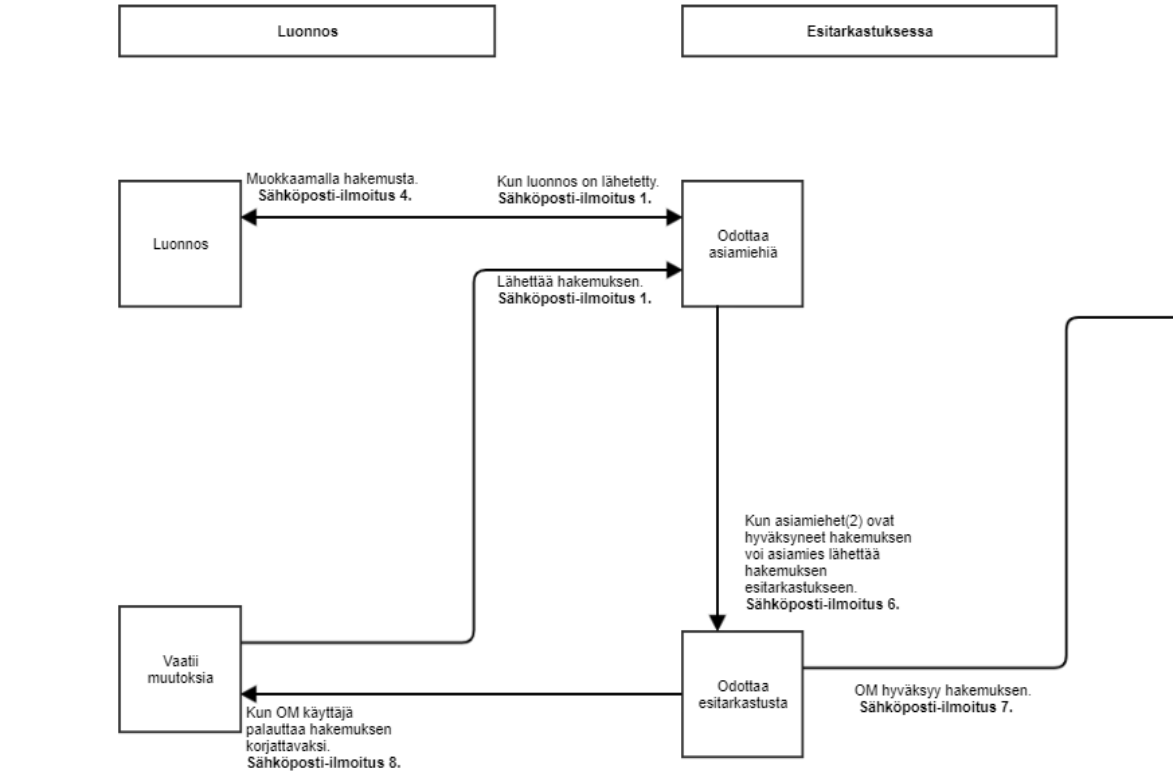
[34] Peyrott, S. (2018). The JWT Handbook. Auth0 Inc., pp.33-34.

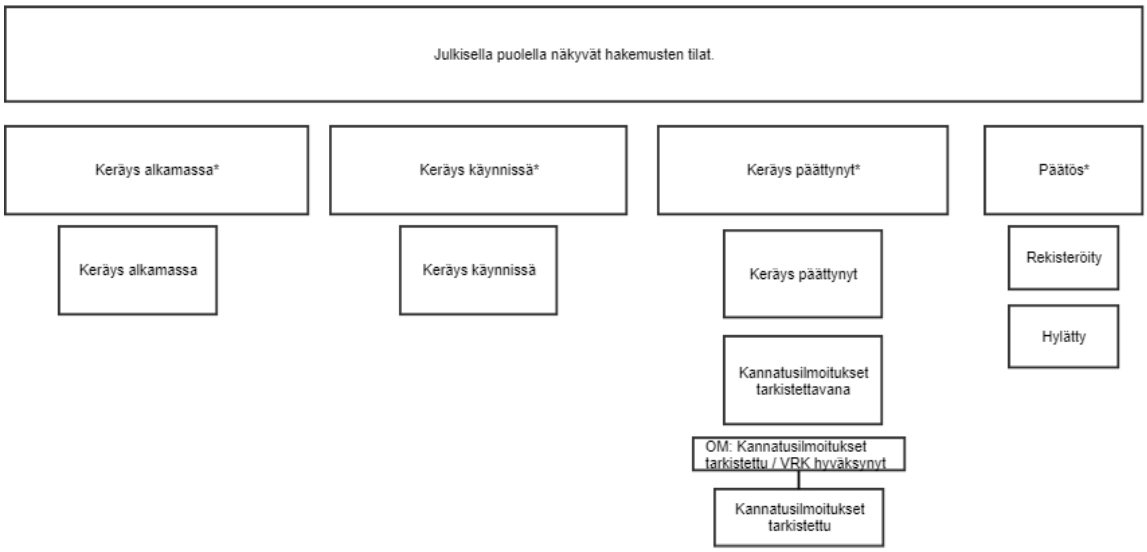
[35] Oikeusministeriö. (2017). Demokratiapoliittinen toimintaohjelma painottaa yhdenvertaisia osallistumismahdollisuuksia. [verkkodokumentti] Saatavilla: https://oikeusministerio.fi/artikkeli/-/asset_publisher/demokratiapoliittinen-toimintaohjelma-painottaa-yhdenvertaisia-osallistumismahdollisuuksia [Luettu 24.10.2019].

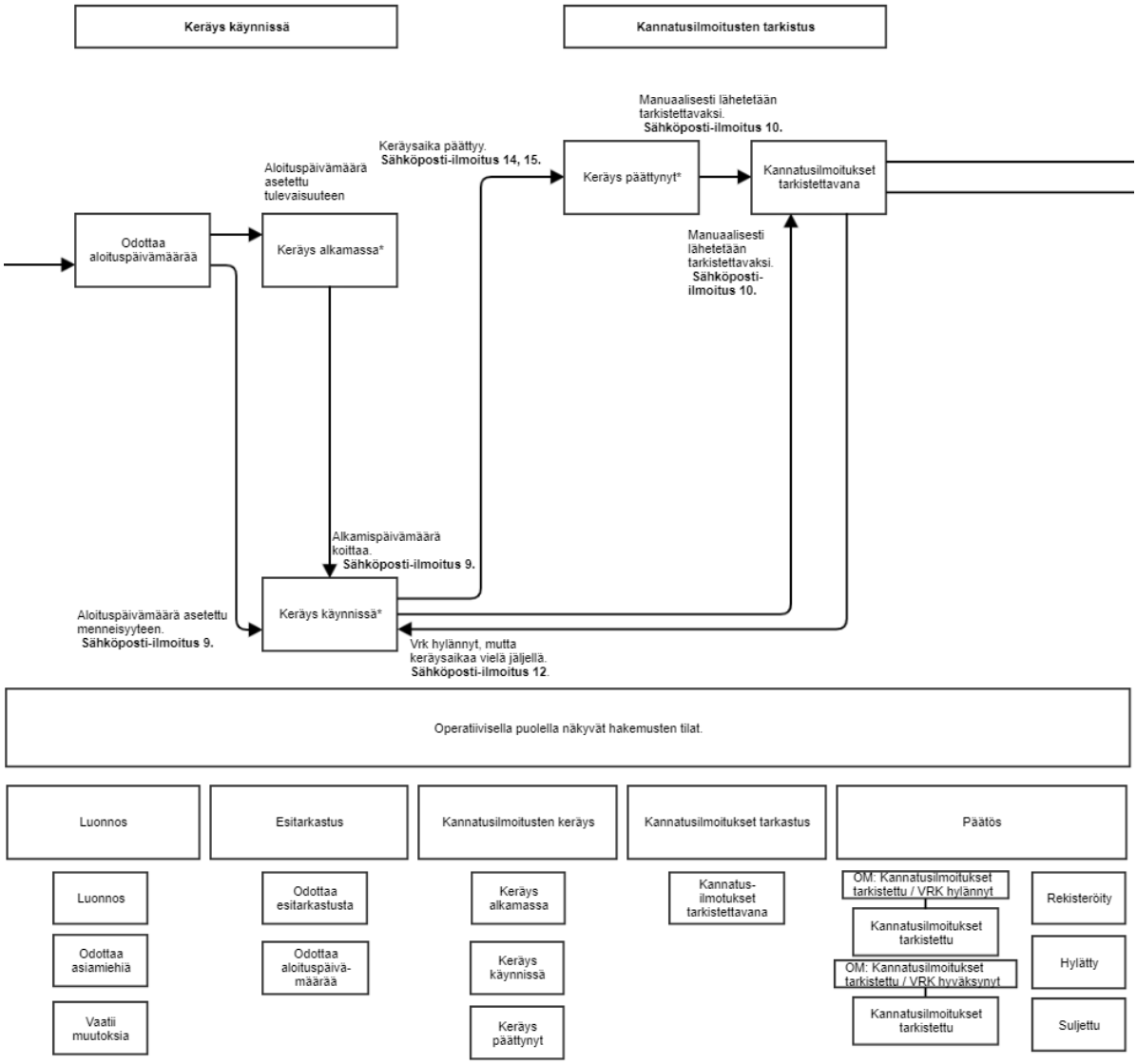
Puoluerekisteri.fi-palvelun arkkitehtuurikaavio

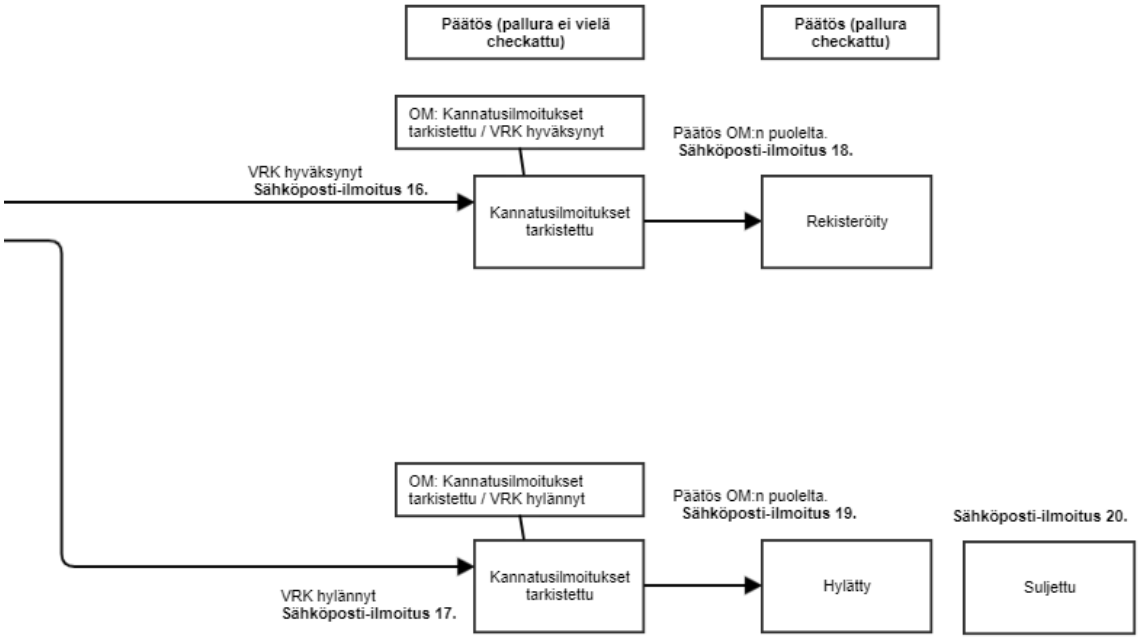


Puolueen rekisteröimisprosessin tilakaavio






Keräys käynnissä
Kannatusilmoitusten tarkistus
Manuaalisesti lähetetään tarkistettavaksi. Sähköposti-ilmoitus 10.
Keräysaika päättyy. Sähköposti-ilmoitus 14, 15.
Aloituspäivämäärä asetettu tulevaisuuteen
Keräys päättynyt*
Kannatusilmoitukset tarkistettavana
Odottaa aloituspäivämäärää
Keräys alkamassa*
Manuaalisesti lähetetään tarkistettavaksi. Sähköposti-ilmoitus 10.
Alkamispäivämäärä koittaa. Sähköposti-ilmoitus 9.
Aloituspäivämäärä asetettu menneisyyteen. Sähköposti-ilmoitus 9.
Keräys käynnissä*
Vrk hylännyt, mutta keräysaikaa vielä jäljellä. Sähköposti-ilmoitus 12.
Operatiivisella puolella näkyvät hakemusten tilat.
Luonnos
Esitarkastus
Kannatusilmoitusten keräys
Kannatusilmoitukset tarkastus
Päätös
Luonnos
Odottaa asiamiehiä
Vaatii muutoksia
Odottaa esitarkastusta
Odottaa aloituspäivä-määrää
Keräys alkamassa
Keräys käynnissä
Keräys päättynyt
Kannatus-ilmotukset tarkistettavana
OM: Kannatusilmoitukset tarkistettu / VRK hylännyt
Kannatusilmoitukset tarkistettu
OM: Kannatusilmoitukset tarkistettu / VRK hyväksynyt
Kannatusilmoitukset tarkistettu
Rekisteröity
Hylätty
Suljettu



Huom!

1. Tähdellä* merkityt näkyvät julkisella puolella Hakemus kohdassa.

2. Ylhäällä näkyvät tilat luonnos, esitarkastuksessa, keräys käynnissä yms. näkyvät asiamiehille hakemusta selatessa (stepperi).

3. Hakemuksen voi sulkea kaikkissa vaiheissa(ei rekisteröitynä).

4. Sähköposti-ilmoitukset