Lotta Skogström

Ohjelmistotuotannon kehittäminen elementtikirjaston avulla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

6.9.2016

Metropolia

Tekijä Otsikko Sivumäärä Aika	Lotta Skogström Ohjelmistotuotannon kehittäminen elementtikirjaston avulla 57 sivua + 3 liitettä 6.9.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Mediatekniikka
Suuntautumisvaihtoehto	Digitaalinen media
Ohjaajat	Managing partner Jesse Jokinen Yliopettaja Kari Aaltonen

Insinöörityön tarkoituksena oli kehittää elementtikirjasto ohjelmistoalan yritykselle. Tilaajayritys toteuttaa elementeistä eli komponenteista koostuvia verkkosovelluksia muiden yritysten perehdytys- ja koulutustarpeisiin. Elementtien uudelleenkäyttöä pyrittiin helpottamaan kehittämällä kirjasto, johon tallennetut elementit ovat kaikkien työntekijöiden saatavilla helposti selattavassa muodossa. Tavoitteena oli kehittää yrityksen tiedonhallintaa ja parantaa tuotantoprosessia. Työhön kuului järjestelmän suunnittelu, toteutus ja käyttöönotto.

Kirjastoon haluttiin ensisijaisesti tallentaa elementtien koodit. Lisäksi tarvittiin käyttöliittymä elementtien selaamiseen. Elementteihin haluttiin myös liittää erilaisia metatietoja. Tarkempien vaatimusten selvittämiseksi toteutettiin kysely, jonka avulla kartoitettiin kirjaston mahdollisia käyttötapoja ja työntekijöiden toiveita kirjaston ominaisuuksien suhteen. Lisäksi suunnitteluvaiheessa perehdyttiin yrityksen tuotantoprosessiin. Projektin lopussa kartoitettiin, miten uusi järjestelmä vaikutti prosessiin.

Elementtikirjaston vaatimukset listattiin ja käsiteltiin käyttötapauskaavion avulla. Sisältömallin suunnittelua varten selvitettiin, millaista sisältöä kirjastoon halutaan lisätä. Lisäksi suunniteltiin, miten sisältöjä luokitellaan ja mitä tietoja niihin liitetään. Elementtien koodit päätettiin tallentaa ohjelmistoprojektien versionhallintapalveluun. Elementtien esikatselua varten suunniteltiin erillinen käyttöliittymä, joka toteutettiin sisällönhallintajärjestelmällä. Käyttöliittymään toteutettiin muun muassa interaktiivinen esimerkki elementtien toimintojen testaamiseen, kuvagalleria toteutuksista ja näkymä, jossa listattavia elementtejä voidaan suodattaa niille määriteltyjen luokittelujen ja muiden metatietojen avulla.

Kun käyttöliittymän tärkeimmät ominaisuudet oli toteutettu, järjestettiin käyttöönottopalaveri, jossa työntekijöitä opastettiin kirjaston käyttöön. Lisäksi kirjastolle laadittiin käyttöohje. Käyttöönoton jälkeen työntekijöiltä kerättiin palautetta. Vastausten perusteella uuteen järjestelmään oltiin tyytyväisiä: toteutustapaa pidettiin toimivana ja ohjeistusta hyödyllisenä. Konkreettiset hyödyt tulevat näkyviin vasta myöhemmin, kun kirjastoon on lisätty useita elementtejä, joita voidaan jatkokehittää ja uudelleenkäyttää. Järjestelmän hyödyllisyyden kannalta on tärkeää saada työntekijät käyttämään sitä aktiivisesti. Myös selkeän ohjeistuksen ja sen noudattamisen merkitys on suuri. Elementit tulee nimetä ja luokitella johdonmukaisesti, jotta kirjaston käyttö on selkeää ja elementtien löytäminen helppoa.

Avainsanat	tiedonhallinta, elementtikirjasto, sisällönhallintajärjestelmä, tuotantoprosessi

Author Title Number of Pages Date	Lotta Skogström Improving software engineering with a component library 57 pages + 3 appendices 6 September 2016
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructors	Jesse Jokinen, Managing Partner Kari Aaltonen, Principal Lecturer

The purpose of the thesis was to develop a component library for a software company that is specialized in creating component-based web applications for different organizations' educational needs. To ease the reuse of the components, they wanted to be made easily accessible for all employees through a library. The goal was to improve the company's data management and production process. Designing, developing and deploying the system were all part of the project.

The library needed to have a place for saving the components' code as well as a user interface for browsing the components. The components also needed to have metadata assigned to them. More detailed requirements were defined with the help of a survey, where the employees were asked to describe what kind of features the component library should have. The company's production process was also analysed at the beginning of the project. When the project was finished, the library's effects on the process were defined.

The requirements of the component library were listed and also presented as a use case diagram. In order to create a content model for the library, the content types and their metadata were defined. It was decided that the components' code would be saved into a code hosting service. A separate user interface for previewing the components was built with a content management system. It included an interactive example for demonstrating the components' functionalities, a photo gallery of different implementations and a view for filtering the components based on their categories and other metadata.

When the main functionalities of the user interface were implemented, all employees were invited to a training session, where they were shown how to use the component library and how to add new elements. After the library had been in use for a while, the employees were asked to give feedback about it. Based on that they were mostly pleased with the system and found it useful. The concrete benefits cannot be seen until later, when the library has multiple elements that can be reused and improved by all employees. To ensure the usefulness of the library, it is important to motivate the employees to use it actively. The components must be named and categorized consistently to guarantee their findability. Clear instructions and following them is also essential.

Keywords	data management, component library, content management system, production process

Sisällys

Liitteet

Liite 1. Yrityksen tuotantoprosessikaavio

Liite 2. Kysely yrityksen työntekijöille

Liite 3. Ohjeet elementtikirjaston käyttöön

Lyhenteet ja määritelmät

Ajax	Asynchronous JavaScript and XML. Teknologia, joka mahdollistaa sivun sisällön päivittämisen ilman sivunlatausta.
Layout	Suunnitelma käyttöliittymän ulkoasusta.
Sass	Syntactically Awesome Stylesheets. CSS:n laajennus, joka helpottaa ja nopeuttaa tyylien luomista muun muassa muuttujien ja sisäkkäisten määrittelyjen avulla.
Wireframe	Käyttöliittymän suunnitteluvaiheessa laadittava rautalankamalli, jossa määritellään käyttöliittymän toiminnot ja eri sisältöjen sijainnit ottamatta kantaa ulkoasun yksityiskohtiin, kuten väreihin ja fontteihin.

1 Johdanto

Insinöörityön tarkoituksena on suunnitella ja toteuttaa elementtikirjasto verkkosovelluksia toteuttavalle yritykselle. Tavoitteena on parantaa tiedonhallintaa ja tuotantoprosessia saattamalla tehdyt toteutukset hyvin dokumentoituina kaikkien työntekijöiden saataville. Kirjaston avulla vanhoja toteutuksia on aiempaa helpompi hyödyntää uusissa projekteissa, mikä sujuvoittaa ja nopeuttaa yrityksen tuotantoprosessia.

Työn tilaaja Apprix Oy on vuonna 2001 perustettu pieni ohjelmistoalan yritys, joka kehittää vuorovaikutteisia verkkosovelluksia pääasiassa yritysten perehdytys- ja koulutustarpeisiin. Sovelluksien avulla voidaan myös esimerkiksi tukea yrityksen uuden strategian käyttöönottoa tai saada työntekijät osallistumaan erilaisiin hankkeisiin. Projektin alkaessa yrityksessä työskenteli kymmenen henkilöä.

Yrityksen kehittämät sovellukset koostuvat komponenteista, joita ovat muun muassa infosivu, kirjautuminen, erilaiset tehtävätyypit kuten monivalintakysymys ja oikein/väärintehtävä, palautesivu ja sitoumussivu. Yrityksessä näitä kutsutaan elementeiksi. Samat elementit toistuvat usein eri sovelluksissa, joten aiemmin toteutettuja elementtejä hyödynnetään mahdollisuuksien mukaan uusissa projekteissa. Uusi järjestelmä toimisi kirjastona, johon eri projekteissa toteutetut elementit tallennetaan. Järjestelmä nimettiin yrityksessä elementtikirjastoksi.

Elementtikirjastoon tallennetaan tiedot eri projekteissa käytetyistä elementeistä sekä niiden koodit. Kirjasto tarvitsee myös käyttöliittymän, jonka kautta elementtejä on helppo selata, lisätä ja muokata. Järjestelmään lisättyjen toteutuksien on tarkoitus helpottaa elementtien uudelleenkäyttöä ja toimia referenssinä uusia sovelluksia suunniteltaessa. Lisäksi se edistää tiedon siirtymistä työntekijältä toiselle ja ehkäisee hyödyllisen tiedon unohtumista ja katoamista työntekijöiden vaihtuessa. Se helpottaa myös uusien työntekijöiden perehdytystä.

Elementtikirjaston avulla projektien tekniseen toteuttamiseen tarvittavan työmäärän arviointi helpottuu, kun jo suunnitteluvaiheessa tiedetään, mitkä elementit löytyvät valmiina vanhoista projekteista. Työn laatu paranee kaikkien käyttäessä ja kehittäessä samoja kirjastossa olevia elementtejä, jotka on jo testattu ja todettu hyviksi. Kun hyödynnetään

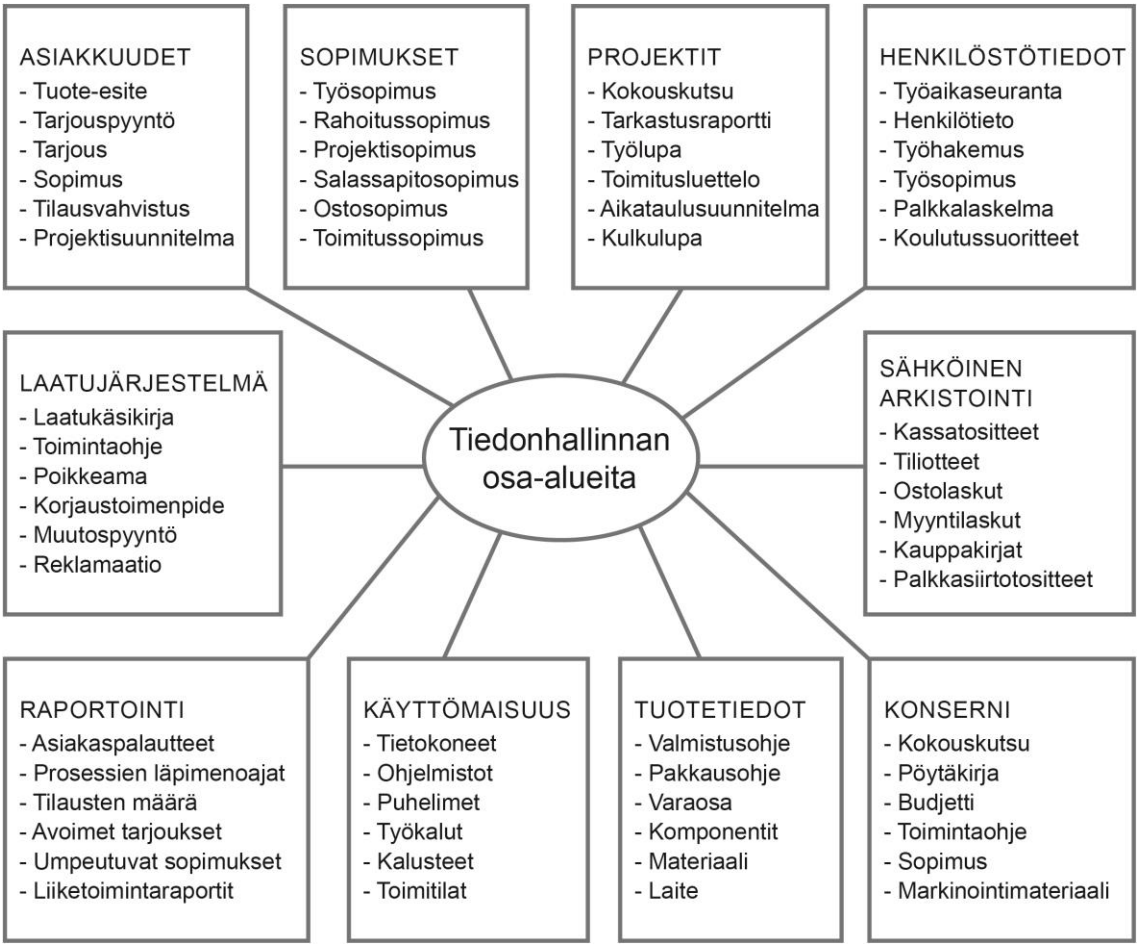
mahdollisimman hyvin olemassa olevia toteutuksia ja tiedetään, mitä kaikkea löytyy valmiina, toimitusvarmuus ja -nopeus lisääntyvät. Näiden hyötyjen seurauksena myös yrityksen kilpailukyky paranee.

Työssä perehdytään yrityksen nykyiseen tuotantoprosessiin ja selvitetään, miten elementtikirjasto voisi tukea ja sujuvoittaa prosessia. Työntekijöille suunnatun kyselyn avulla tutkitaan, millaisia ominaisuuksia kirjastossa toivotaan olevan ja mitä eri käyttötarkoituksia sillä on. Lisäksi määritellään, mitä sisältöä ja sisältöihin liitettäviä tietoja kirjastoon kerätään. Työhön kuuluu myös toteutustavan valinta ja käyttöliittymän suunnittelu ja toteutus.

2 Tiedonhallinta osana yrityksen liiketoimintaa

2.1 Tiedon ja tietojärjestelmien rooli yrityksessä

Tiedonhallinta on keskeinen osa yritysten liiketoimintaa, sillä lähes kaikki yrityksissä tehtävä työ liittyy tiedon käsittelyyn (1, s. 10, 241). Sen eri osa-alueita ovat muun muassa asiakkuuksien, sopimusten, projektien, henkilöstötietojen, yrityksen omaisuuden, tuotetietojen ja raporttien hallinta (1, s. 141). Muutamia eri osa-alueita ja niihin liittyviä tietoja ja dokumentteja on listattu kuvassa 1.



Kuva 1. Esimerkkejä tiedonhallinnan eri osa-alueista ja niihin liittyvistä dokumenteista ja tiedoista (1, s. 148, 152, 162, 164, 176, 179, 192–193, 196, 205, 209.)

Kuvan 1 mukaisesti esimerkiksi projektien tiedonhallinnassa tulee huomioida, että kokouskutsut, raportit, työluvat, aikataulusuunnitelmat ja muut tarpeelliset tiedot ja dokumentit ovat ajan tasalla ja löydettävissä. Tietojen ja dokumenttien tulee olla saatavilla sekä projektin aikana että sen loputtua. (1, s. 159, 162.)

Tiedonhallinnan ja toimivien tietojärjestelmien merkitys ymmärretään nykyään entistä paremmin. Työntekijöiltä vaaditaan enemmän kuin aiemmin: tieto pitää osata löytää itsenäisesti, mikä edellyttää kykyä hallita suuria tietomääriä ja ymmärtää tietojen välisiä suhteita. Myös tietojärjestelmiltä vaaditaan enemmän. Tieto ei saa olla piilossa tai vaikeasti hyödynnettävissä, vaan sen tulee olla helposti saatavilla paikasta, ajasta ja käytettävästä laitteesta riippumatta. Kasvaneet vaatimukset pakottavat yrityksiä parantamaan toimintatapojaan. Kehittämisen tulisi olla osa päivittäistä toimintaa, sillä tiedonhallinnan kehittäminen on samalla liiketoiminnan kehittämistä. (1, s. 9–10, 241.)

Tiedon hyödyntämisen edellytyksenä on tiedon löydettävyys, joka perustuu yhdessä sovittuihin tietojen käsittelyn käytäntöihin. Sujuvan tiedonhallinnan edellytyksenä on, että työntekijä tiedostaa omien työtapojensa vaikutukset muiden työskentelyyn. Tiedonhallinta ei ole tehokasta, jos jokainen toimii yhteisesti sovittujen käytäntöjen sijaan omien tapojensa mukaisesti ja tiedot tallennetaan paikkoihin, joista muut eivät niitä löydä. (1, s. 15, 241.) Jos tieto on olemassa, mutta sen löytäminen tai hyödyntäminen ei onnistu, tieto muuttuu hyödyttömäksi (2, s. 234). Ilman toimivaa tiedonhallintaratkaisua yrityksestä voi kadota paljon hyödyllistä tietoa esimerkiksi työntekijöiden vaihtuessa (1, s. 28).

Tiedonhallinta kaipaa kehittämistä, jos työntekijät joutuvat toistuvasti pyytämään apua tiedon etsimiseen keskeyttäen myös muiden työt. Totuttujen tapojen kyseenalaistaminen voi olla hankalaa, mutta tilanteen parantamiseksi se on välttämätöntä. Viiveet tiedon saamisessa, vanhentunut ja virheellinen sisältö, tiedon piiloutuminen ja väärin määritellyt käyttöoikeudet voivat pahimmillaan heikentää yrityksen tulosta. (1, s. 15–16.)

Tiedonhallintajärjestelmät

Yrityksen tiedot on yleensä tallennettu joko tiedosto- tai tietokantapohjaisiin järjestelmiin. Tiedostopohjaisia järjestelmiä ovat verkkolevyt, työntekijöiden omien tietokoneiden kiintolevyt sekä verkossa olevat pilvipalvelut kuten Dropbox ja Google Drive. Tiedot on tallennettu dokumentteina kansioihin, eli järjestys on sijaintiperusteinen. Tietokantapohjaisia järjestelmiä puolestaan ovat esimerkiksi erilaiset taloushallinnon, toiminnanohjauksen, myynnin, tuotannon ja työajan seurannan tietojärjestelmät, joihin tieto on tallennettu datan muodossa. Työnteon sujuvuuden takaamiseksi tietojen tulee tallennuspaikasta riippumatta olla järjestyksessä ja helposti löydettävissä. Järjestys tulee määritellä yhdessä niin, että se on selkeä kaikille työntekijöille. Yhden henkilön määrittelemä tiedostojen kansio- ja nimirakenne ei välttämättä ole selkeä muille kuin henkilölle itselleen. (1, s. 12–13, 27.)

Ihmiset ovat tottuneet tallentamaan tietoa sijantiperusteisiin kansiorakenteisiin. Tiedon ja sitä tarvitsevien ihmisten määrän kasvaessa ei kuitenkaan enää riitä, että tieto on tallessa verkkolevyn kansiossa, sähköpostissa tai yksittäisen työntekijän työpöydällä. (1, s. 27.) Yrityksissä käytössä olevat kymmeniä vuosia vanhat tietojärjestelmätkään eivät enää vasta työelämän vaatimuksia. Tilalle tulevat uudet järjestelmät ovat helpommin muokattavissa ja usein käytettävyydeltään parempia kuin vanhat. Lisäksi ne hyödyntävät

tiedonhaussa metatietoja. Metatietopohjaisessa järjestelmässä tietoa kuvataan asiasanoilla, joiden avulla tieto löytyy nopeasti suurenkin tietomäärän seasta. (1, s. 13.)

Perinteisistä sijaintipohjaisista järjestelmistä tietoa etsitään yleensä kansioista dokumentin nimen perusteella. Jos tietoa on paljon, muistettava kansiopolku voi muodostua pitkäksi. Hakutoimintokaan ei auta, jos tiedosto on nimetty eri tavalla, kuin hakija olettaa. Metatietopohjaisissa järjestelmissä dokumentteja ja dataa haetaan sisältöhaulla, joka etsii hakusanaa pelkän tiedoston nimen sijaan myös sisällöstä ja metatiedoista. Tieto on mahdollista löytää luonnollisen ajattelun avulla, ilman että pitäisi muistaa esimerkiksi tietyn dokumentin fyysinen sijainti laajassa hierarkkisessa kansiorakenteessa. Metatiedot antavat tiedolle yksilöllisen identiteetin, jonka avulla tieto löytyy helposti ja nopeasti. (1, s. 54, 83.)

Sisällön kuvailu metatietojen avulla

Metatietoa hyödyntävä järjestelmä perustuu tietojen välisiin suhteisiin, joiden avulla tietoa on helpompi järjestää ja etsiä. Yksittäinen tieto ei yleensä ole hyödyllinen, ellei tiedetä, mihin se liittyy. Yrityksen dokumentit ja data liitetään yrityksen toimintaan usein mastertietojen kautta, joita ovat esimerkiksi asiakkaat, tuotteet, projektit ja työntekijät. (1, s. 54.) Mastertietojen käyttö tiedonhallintajärjestelmien metatietoina on yleistä, koska tiedot ovat yrityksen toiminnan kannalta tärkeitä ja luonteeltaan pysyviä. Tieto löytyy helposti, jos järjestelmässä voidaan esimerkiksi yhden haun avulla saada näkyviin kaikki tiettyyn projektiin liittyvät tiedostot ja muu data. (1, s. 143.)

Metatieto on dataa, jonka avulla kuvaillaan verkossa olevia resursseja. Sen avulla voidaan kuvata tietoa järjestelmällisesti ja jäsennellysti, mikä parantaa tiedon löydettävyyttä. Metadataa voidaan käyttää esimerkiksi dokumentin otsikon, tekijän ja luontipäivämäärän määrittelyyn. Jäsennelty kuvailutieto mahdollistaa tehokkaan haun, jossa haku voidaan kohdistaa tavallista tarkemmin. (3, s. 113; 4, s. 10.)

Metatietoja käytetään myös laajemmin kuin ainoastaan yritysten suljetuissa tietojärjestelmissä. Niiden avulla verkossa oleva sisältö on mahdollista saada koneiden ymmärtämään muotoon. Tähän pyrkii esimerkiksi Semantic Web -hanke, jonka tavoitteena on, että koneet pystyisivät analysoimaan ja ymmärtämään tietoa pelkän tiedon esittämisen sijaan. Jotta koneet osaisivat käsitellä metatietoja automaattisesti, niiden lisäämiseen tarvitaan standardoitu tapa. Käytössä olevia standardeja ovat muun muassa Dublin Core

ja Googlen, Microsoftin, Yahoon ja Yandexin sponsoroima Schema.org. Metatietojen käyttö on hyödyllistä esimerkiksi hakukoneoptimoinnin kannalta, sillä se parantaa hakurobottien kykyä tulkita sivujen sisältöä, jolloin sivu on hakukonetta käytettäessä helpommin löydettävissä. (3, s. 117; 4, s. 8–11; 5; 6.)

Vaatimukset tiedolle ja tietojärjestelmille

Työn liikkuvuuden lisääntyessä myös tiedolle asetetut vaatimukset ovat kasvaneet. Jotta tieto on hyödyllistä, sen tulee olla ajantasaista ja saatavilla oikeille henkilöille oikeaan aikaan. Työnteon sujuvuuden takaamiseksi tiedon tulee olla saatavilla riippumatta ajasta, sijainnista ja käytössä olevasta päätelaitteesta. Työtä pitäisi pystyä tekemään työpaikan ulkopuolella samaan tapaan kuin työpaikalla, jotta esimerkiksi etätöitä tehdessä ylimääräistä aikaa ei kulu tavallisesta poikkeavalla tavalla tehtävään tiedon etsimiseen, tallentamiseen, muokkaamiseen ja lähettämiseen. (1, s. 39, 73.)

Muita tärkeitä tiedon ja tietojärjestelmien ominaisuuksia ovat muun muassa virheettömyys, yhdenmukaisuus ja helppokäyttöisyys. Yhdenmukaisuuden takaamiseksi tiedolle tulee määritellä muoto, jossa se esitetään. Tietojärjestelmien tulee olla helppokäyttöisiä, jotta työntekijät voivat itsenäisesti hyödyntää niihin tallennettuja tietoja. Järjestelmien käyttöä ja tiedon hyödyntämistä voidaan helpottaa ohjeistuksen avulla. Lisäksi tietojärjestelmien tulee olla riittävän joustavia, jotta niitä on mahdollista muunnella tarpeiden lisääntyessä ja muuttuessa. Järjestelmiä suunnitellessa tulee huomioida myös tietoturva ja vikatilanteet. Ongelman tai vian sattuessa tiedot eivät saa kadota lopullisesti, vaan ne pitää pystyä palauttamaan. Tietoturva huomioidaan esimerkiksi käyttöoikeuksien ja käyttäjätunnusten määritellyllä. (7, s. 256–257.)

2.2 Tiedonhallinnan kehittäminen

Suurin osa yrityksissä tehtävästä työstä liittyy tavalla tai toisella tiedon hyödyntämiseen. Jatkuvasti kehittyvistä tiedonhallintajärjestelmistä huolimatta hyödyllisen tiedon seulonta ja tiedon arkistointi on kuitenkin monessa yrityksessä toteutettu heikosti. Jos tiedon luominen, etsiminen, muokkaaminen, jakaminen ja tallentaminen tuntuvat työläältä ja vaikealta, työnteko hidastuu ja tietyssä ajassa tehtyjen työsuoritteiden määrä vähenee. Työsuoritteet vaikuttavat suoraan tai välillisesti yrityksen tulokseen, joka heikkenee huonon tiedonhallinnan myötä. Tiedonhallinnan kehittämisellä pyritään tehostamaan toimintaa ja

parantamaan tulosta: toimiva tiedonhallinta tekee työskentelystä tehokasta, sujuvaa ja mielekästä. Kun työsuoritteisiin liittyvät tiedot löytyvät helposti, niitä saadaan tehtyä aiempaa enemmän. Seurauksena tuottavuus kasvaa ja yrityksen tulos paranee. Kehittämisen taloudelliset vaikutukset voi olla vaikea määritellä tarkasti, sillä tiedonhallinta vaikuttaa useisiin eri prosesseihin. Tämän vuoksi tiedonhallintaa tulee kehittää pitkäjänteisesti ja hyötyjä arvioida kokonaisuutena. (1, s. 18, 20–21; 2, s. 234.)

Tiedonhallinta on yleensä helpompaa pienessä yrityksessä kuin suurissa, useisiin eri toimipisteisiin laajentuneissa yrityksissä. Kun yritys kasvaa, kasvaa myös tiedon ja toimintojen määrä, mikä edellyttää järjestelmällistä tiedonhallintaa. Jos tiedonhallinnan kehittämiseen ei yrityksen alkuvaiheessa kiinnitetä huomiota, yrityksen kasvaessa ongelmat moninkertaistuvat. (1, s. 104, 141.)

Tiedonhallinnan kehittämisen lähtökohtana on parantamista vaativien osa-alueiden tunnistaminen. Aluksi tulee selvittää, mitä tietoa yrityksessä luodaan ja kuka tietoja tarvitsee. Lisäksi kartoitetaan tiedon muokkaamisen tavat ja tallennuspaikat. Kattavan kokonaiskuvan muodostamiseksi tietoja kerätään eri puolilta yritystä. Yhdessä keskustelemalla pyritään luomaan yhtenäinen näkemys kehittämisen syistä, suunnasta ja tavoitelluista vaikutuksista. Kokonaiskuvan muodostamisen jälkeen päätetään kehitystapa: suunnitellaanko kokonaan uusi järjestelmä vai parannetaanko käytössä olevia toimintatapoja. (1, s. 104, 107–108.)

Kehittämisen onnistumiseksi kehitystyössä tulee olla mukana mahdollisimman kattavasti työntekijöitä eri työtehtävistä (1, s. 132). Uuden järjestelmän käyttöönoton yhteydessä tulee järjestää koulutusta ja kerätä käyttäjiltä palautetta järjestelmästä. Palautetta voidaan käyttää heti toteutettavassa järjestelmän parantelussa ja tulevan kehittämisen pohjana. (1, s. 123–124.)

Uuden tiedonhallintajärjestelmän käyttöönoton yhteydessä tulee myös huomioida, että järjestelmään ei kannata suoraan siirtää kaikkia vanhoja tietoja ja dokumentteja. Tiedon ajantasaisuuden, käyttökelpoisuuden ja eheyden varmistamiseksi on parasta lähteä liikkeelle tyhjästä ja siirtää järjestelmään vanhoja tietoja harkiten. (1, s. 114–115.)

2.3 Sisällönhallintajärjestelmä tiedonhallinnan apuna

Sisällönhallintajärjestelmä on ohjelmisto, joka helpottaa verkkosivujen sisällön lisäämistä ja muokkaamista. Yleensä se tarjoaa verkkopohjaisen käyttöliittymän, jonka kautta sivun hallinnoijat voivat päivittää verkkosivujen sisältöä ilman sen suurempaa teknistä osaamista. Suosituimpia verkkopohjaisia sisällönhallintajärjestelmiä ovat muun muassa WordPress, Blogger, Joomla ja Drupal. (8.) Sivun ylläpidon helpottamisen lisäksi sisällönhallintajärjestelmä tarjoaa verkkosivuille käyttöliittymän ja erilaisia toimintoja. Järjestelmässä valmiina olevat toiminnot helpottavat verkkosivujen kehittämistä, kun kaikkea ei tarvitse toteuttaa itse tyhjästä aloittaen. Yleensä kuitenkin osa ominaisuuksista täytyy toteuttaa itse tai järjestelmän toimintoja muokkaamalla. (9, s. 5.)

Sisältömalli

Sisältömalli (content model) määrittelee järjestelmässä olevat sisältötyypit ja niiden väliset suhteet (9, s. 180). Mallin suunnittelu aloitetaan järjestelmän vaatimusten analyysillä, jonka pohjalta listataan kaikki tarvittavat sisältötyypit ja niihin liitettävät tiedot (9, s. 35). Malli voidaan esittää esimerkiksi kaaviona tai taulukkona. Yksityiskohtaisessa mallissa sisältötyypit on jaettu pienempiin osiin: sisältötyyppi voi koostua esimerkiksi tekstistä, kuvista, multimediasisällöstä ja linkeistä. Osien listaamisen lisäksi voi olla tarpeellista määritellä, missä muodossa tiedot tulee antaa. (9, s. 34; 10.) Sisältötyyppiin liitettäviä tietoja voivat olla esimerkiksi avainsanat, kategoriat, viittaukset muihin sisältöelementteihin ja muut metatiedot, kuten sisällön luojan nimi ja luontipäivämäärä, jotka sisällönhallintajärjestelmä yleensä lisää automaattisesti. (9, s. 34–35.) Taulukossa 1 on esimerkki musiikkiaiheisen verkkosivun artistiprofiilin yksityiskohtaisesta sisältömallista.

Taulukko 1. Esimerkki artistiprofiilin sisältömallista (10).

Sisällön osa	**Osan kuvaus**	**Tyyppi**	**Pakollinen**
Artistin nimi	Artistin nimi	Tekstikenttä	x
Tiivistelmä	Artikkelin alaotsikko	Tekstikenttä	x
Tekijä	Profiilin tekijä	Viittaus tekijään	
Kuva	Artistin kuva	Kuva	x

Kuvaus	Kuvausteksti	Tekstikenttä	x
Video	Yksi tai useampi video	Viittaus videoon	

Taulukossa 1 on listattu artistiprofiiliin liittyvät osat, osien kuvaukset ja niiden tyypit. Viimeisessä sarakkeessa on merkitty, onko osan pakko olla mukana. Taulukossa voitaisiin määritellä myös esimerkiksi osien tarkat nimet sisällönhallintajärjestelmää varten sekä mahdollisia lisätietoja. Artistiprofiilin lisäksi musiikkiaiheisen sivuston sisältömalliin voisivat kuulua omina sisältötyyppeinään esimerkiksi albumi ja kappale, jotka linkittyvät artistin profiilisivuun. (10.)

Sisällönhallintajärjestelmä tarjoaa sisällön luomisen ja muokkaamisen lisäksi työkalut sisältötyyppien määrittelyyn, metatietojen liittämiseen ja sisältöjen välisten suhteiden merkitsemiseen (9, s. 5). Metatietojen ja kategorioihin ja avainsanoihin perustuvien luokittelujen avulla tieto on järjestelemässä helpommin löydettävissä. (9, s. 32.)

Sisällön jäsentelyn tarpeen määrittelee sivun käyttötarkoitus. Esimerkiksi blogiteksti voi sisältää kuvia, tekstiä, videoita ja linkkejä täysin vapaassa järjestyksessä. Jos taas halutaan rakentaa tuotesivu, tulee tuotteen nimen, kuvan, hinnan, tuotekategorian, kuvaustekstin ja muiden tietojen olla erotettavissa toisistaan. Tuotteita pitää todennäköisesti pystyä lajittelemaan eri tietojen, kuten hinnan, lisäyspäivämäärän ja kategorian, perusteella. Tiedot pitää pystyä esittämään eri tarvoin eri sivuilla: esimerkiksi listanäkymässä näytetään vain nimi, hinta ja kuva, kun taas tuotesivulla listataan kaikki tiedot. Jotta lajittelu, eri tietojen poiminen eri näkymiin ja ulkoasun muuttaminen onnistuu helposti, tietojen tulee olla sisältöä luotaessa omina kenttinään. Lajittelun toimimiseksi tulee määritellä sisällön formaatti, eli esimerkiksi hintakenttään voidaan syöttää vain numeroita pilkulla tai pisteellä erotettuna. (10.)

Sisältömallin määrittelystä on hyötyä etenkin isoissa projekteissa, joissa järjestelmän vaatimuksia, ulkoasua, toteutusta ja sisällön päivitystä on tekemässä useita eri henkilöitä. Sen avulla sisällöntuottajat tietävät, mitä sisältöä sivustolle voidaan lisätä, missä muodossa sisällön tulee olla ja miten sisältö lisätään. Sisällöntuottajien tarpeet tulee pitää mielessä sisältömallia suunnitellessa, jotta sisällön lisääminen on helppoa ja selkeää. (9, s. 33–34; 10.) Sisältömallin tulee myös vastata suunniteltua ulkoasua ja sivuston vaatimuksia. Jos kuville on ulkoasusuunnitelmassa merkitty kuvatekstit, niiden tulee

löytyä myös sisältömallista. Jos sisältöä pitää pystyä lajittelemaan päivämäärän perusteella, päivämäärän tulee olla mukana sisältömallissa. Lisäksi päivämäärän formaatin pitää olla tarkasti määritelty, jotta lajittelu onnistuu. (10.)

Sisällönhallintajärjestelmä tarjoaa yleensä jonkinlaisen sisältömallin pohjan, jota voi tarvittaessa muokata. Eri sisältötyyppejä ja niihin liitettäviä tietoja on mahdollista lisätä sisällönhallintajärjestelmän tarjoamien työkalujen tai lisäosien avulla. Selkeästi määritelty sisältömalli varmistaa, että sivuston rakentava kehittäjä osaa määritellä sisällöt halutulla tavalla. (9, s. 35; 10.)

Hakutoiminnot

Hyvin suunnitellun ja johdonmukaisen sisältömallin avulla sisällönhallintaohjelmalla toteutetulle verkkosivulle voidaan kehittää tehokas haku, joka helpottaa sisällön etsimistä. Haun apuna käytetään usein metatietoja. Sisältöä kuvailevat avainsanat ja tarkkaan määritellyt, sisältöä ryhmittelevät kategoriat toimivat haun perustana. Avainsanoja voi yleensä lisätä rajattomasti, kun taas kategorioita on rajattu määrä. Sisältöjen tulee kuulua ainakin yhteen kategoriaan. Toisinaan kategoriat voivat olla osittain päällekkäisiä, jolloin sisältö voi kuulua useampaan kuin yhteen kategoriaan. (9, s. 50–51.)

Tavallinen sanahaku ei yleensä ole riittävän tehokas, sillä valitun hakusanan tulisi olla joko sisältöelementin otsikossa tai tekstissä, jotta sisältö löytyy. Avainsanojen avulla hakutulosten kattavuutta saadaan parannettua. Kategoriat puolestaan rajaavat hakua ja tarkentavat hakutuloksia: niiden avulla käyttäjä voi valita, mihin kategorioihin haku kohdistuu ja mikä jää haun ulkopuolelle. (9, s. 49–50.)

Hakutulosten suodattamiseen ja lajitteluun käytetään usein Ajax-teknologiaa. Ajax (Asynchronous JavaScript and XML) mahdollistaa käyttäjän tekemien valintojen käsittelyn ilman sivunlatausta, mikä tekee toiminnosta interaktiivisemman ja nopeamman. Jos käyttäjä muuttaa hakusanaa tai rajaa hakutuloksia kategoriasuodattimen avulla, koko sivun päivittymisen sijaan ainoastaan hakutuloslista päivittyy. (9, s. 15–17, s. 81.)

2.4 Komponenttikirjastot

Komponenttikirjasto on kokoelma komponentteja, jotka on järjestetty jonkin logiikan perusteella mielekkääksi kokonaisuudeksi. Usein kirjasto tarjoaa myös tavan selata ja esikatsella komponentteja. Kirjasto voi muodoltaan olla lähes minkälainen tahansa; se voi esimerkiksi koostua pelkistä HTML-tiedostoista, tai sille on voitu rakentaa käyttöliittymä itse tai valmista palvelua käyttäen. Komponentti puolestaan on osa verkkosivun käyttöliittymää. Se voi olla esimerkiksi yksittäinen HTML-elementti, kokonainen sivu tai mitä tahansa siltä väliltä. HTML-koodin lisäksi komponentti voi sisältää ulkoasun eli tyylin, toimintoja ja dokumentaatiota. Komponenteista voidaan käyttää erilaisia nimityksiä, kuten moduuli, sivu, ulkoasu (layout), kappale tai elementti. Nimitys voi olla lähes mikä vain, kunhan se määritellään ja sitä käytetään johdonmukaisesti. (11.)

Komponenttikirjaston muotoon vaikuttaa sen käyttötarkoitus. Kirjasto voi olla esimerkiksi integroitu osaksi verkkosivua, jolla sitä käytetään. Se voi olla koodia ja dokumentaatiota sisältävä palvelu, joka tulee yrityksen sisäisen tai ulkopuolisen tahon käyttöön. Komponenttikirjasto voi olla myös yrityksen sisäinen työkalu, jonka avulla varmistetaan eri tiimien tekemien toteutusten yhtenäisyys. (11.)

Kirjastoon kerätään tavallisesti joko ulkoasun suunnittelumalleja (design pattern) tai verkkosivujen komponentteja. Suunnittelumallit määrittelevät, millä periaatteilla ulkoasuja kehitetään, ja tarjoavat ratkaisuja yleisiin suunnittelussa toistuviin tilanteisiin. Niiden kerääminen on hyödyllistä esimerkiksi isoille yrityksille, jotka suunnittelevat useita eri palveluita ja tuotteita, joiden ulkonäössä halutaan pitää yhtenäinen linja. (12.) Esimerkiksi Google ylläpitää Material Design -opasta, joka kertoo, miltä Googlen sovellusten tulisi näyttää. Googlen sisäisen käytön lisäksi se on suunnattu muun muassa kaikille, jotka kehittävät Android-sovelluksia. (13.) Verkkosivujen komponenttien kerääminen on hyödyllistä yrityksille, jotka rakentavat modulaarisia eli pienistä osista koostuvia verkkosivuja. Komponenttikirjasto helpottaa osien uudelleenkäyttöä. Erilaisia komponentteja voivat olla esimerkiksi navigaatio tai videosoitin. (12.)

Komponenttikirjaston käytön hyödyt ja haasteet

Komponenttikirjaston käyttämisen tavoitteena on tehdä työnteosta tehokkaampaa ja johdonmukaisempaa. Komponenttien uudelleenkäyttö nopeuttaa työtä, jolloin säästetään aikaa ja rahaa. Kirjasto toimii myös yrityksen muistina, jossa tieto toteutuksista säilyy,

vaikka työntekijät vaihtuisivat. Sen avulla edistetään työntekijöiden välistä yhteistyötä, yhdessä oppimista ja tiedon jakamista. Lisäksi kirjastoa varten luotu sanasto auttaa työntekijöitä ymmärtämään toisiaan, kun komponenteista puhuttaessa käytetään yhdessä sovittuja nimityksiä. (12.)

Komponenttikirjasto ei kuitenkaan synny itsestään, vaan sen suunnittelu, toteutus ja ylläpitäminen vaativat aikaa ja vaivannäköä. Yrityksen pitää miettiä tarkkaan, kannattaako kirjaston kehittäminen. Työntekijöiden tulee olla motivoituneita sen käyttämiseen, ja lisäksi yrityksen johdon tulee varata heille riittävästi aikaa kirjaston ylläpitoon. (12.) Käyttöliittymä tulee suunnitella tarkoin, jotta siitä on hyötyä kaikille kirjaston käyttäjille, kuten suunnittelijoille, kehittäjille, myyjille ja muille mahdollisille ryhmille (11). Hyvin toimiakseen komponenttikirjasto tarvitsee ”kirjastonhoitajan”, jonka tehtävänä on motivoida ihmisiä käyttämään kirjastoa, neuvoa sen käytössä ja valvoa, että kirjastoa käytetään sovitulla tavalla. (12.)

Komponenttikirjaston suunnittelu

Komponenttikirjaston käyttö muuttaa yleensä yrityksen työnkulkua. Projekteja ei aloiteta enää puhtaalta pöydältä, vaan suunnittelun ja toteutuksen apuna voidaan käyttää kirjastoa. Lisäksi se muuttaa projektien komponenttien, ohjeistuksien ja dokumentaation hallintatapaa. Kirjaston suunnittelussa on tärkeää varmistaa, että komponentteja on helppo löytää, niihin on helppo viitata suunnitelmissa esimerkiksi nimen tai linkin avulla ja että ne on helppo ottaa käyttöön uusissa projekteissa. (12.) Koodin kannalta haastavinta on suunnitella, miten komponentit saadaan integroitua projektien koodipohjiin. Tämän helpottamiseksi komponentit tulisi toteuttaa samoilla työkaluilla, ohjelmointikielillä ja -kirjastoilla, joita käytetään projektien toteutuksessa. Integrointitapa kannattaa suunnitella mahdollisimman aikaisessa vaiheessa, jotta komponentit kehitetään mahdollisimman hyvin uudelleenhyödynnettäviksi. (11.)

Komponenttien selaamiseen tarkoitettu käyttöliittymä ja kirjasto, jossa komponenttien koodi sijaitsee, tulee pitää erillään. Kirjaston kooditiedostojen tulee olla selkeässä ja ymmärrettävässä järjestyksessä. Tiedostoja ja kansioita pitää pystyä selaamaan niin, että haluamansa komponentit on helppo löytää ja tunnistaa. Lisäksi komponenttien välinen hiearkia tulee olla selkeästi nähtävissä, jos hierarkia on käytössä. (11.)

3 Liiketoimintaprosessit

3.1 Yritysten prosessit

Yrityksen toimintatavat kuvataan usein prosesseina. Prosessi on kokonaisuus, joka muodostuu toisiinsa liittyvistä toiminnoista, toimintojen suorittamiseen tarvittavista ihmisistä, järjestelmistä, työkaluista ja menetelmistä sekä prosessissa syntyvistä tuloksista. Ohjelmistokehityksessä yleisiä prosesseja ovat asiakas-, tuotekehitys- ja ylläpitoprosessi. (14, s. 137; 15, s. 19.) Prosessi on luonteeltaan pysyvä ja toistuva, joten sen kulkua on mahdollista mallintaa ja kehittää. Suunnittelun lähtökohtana ovat yrityksen sisäisen tai ulkopuolisen asiakkaan tarpeet, joiden tyydyttäminen on liiketoimintaprosessin tärkein tavoite. Asiakkaan tärkeyden korostamiseksi prosessin tulisi alkaa asiakkaasta ja loppua asiakkaaseen. (16, s. 25; 15, s. 20, 22.)

Prosessin aikana sisäiseltä tai ulkoiselta toimittajalta saadut syötteet muuttuvat ihmisten, laitteiden, rahoituksen ja muiden resurssien avulla asiakkaalle toimitettavaksi tuotokseksi. Syötteet ja tuotokset voivat koostua tiedosta tai fyysisestä materiaalista. (7, s. 124; 17, s. 47.) Liiketoimintaprosessien malli on havainnollistettu kuvassa 2.

Kuva 2. Liiketoimintaprosessin malli (7, s. 124; 17, s. 48).

Yrityksen prosessit luokitellaan tavallisesti ydin- ja tukiprosesseihin, jotka voidaan lisäksi jakaa alemman tason osa- ja aliprosesseihin. Ydinprosessit ovat yrityksen keskeisimpiä, ulkoista asiakasta palvelevia toimintoja. Niitä ovat esimerkiksi tuotteen kehittäminen, tuotanto, tuotteen toimittaminen ja asiakaspalvelu. Ydinprosesseissa on mukana yrityksen ulkopuolisia tahoja, kuten asiakas ja muut sidosryhmät, toisin kuin tukiprosesseissa, jotka tapahtuvat yrityksen sisällä. Tukiprosessit tukevat ydinprosesseja ja mahdollistavat

yrityksen toimintojen sujuvuuden ja tehokkuuden. Esimerkkejä tukiprosesseista ovat talous- ja henkilöstöhallinto, toiminnan suunnittelu, yhteistyö toimittajien kanssa ja tietojärjestelmien hallinta. (7, s. 130; 15, s. 55–56; 17, s. 15.)

Jokaiselle prosessille määritellään tavoitteet ja prosessikuvaus eli ohjeistus, jossa kuvataan prosessin tapahtumat, toimijat ja vastuut. Tavoitteiden tulee olla sellaisia, joiden toteutumista on mahdollista mitata. Prosessille on lisäksi määritelty yksi tai useampi omistaja, jonka vastuulla on kehittää prosessia sen perusteella, miten hyvin tavoitteet saavutettiin. (14, s. 137–138.)

Prosessien tulee olla joustavia ja tilanteeseen sopivia, sillä eri projektien vaatimukset, laajuus, mukana olevien osapuolten määrä ja työntekijöiden taidot voivat vaihdella paljon. Ohjelmistokehityksessä tilanteeseen vaikuttavat myös esimerkiksi ylläpitovaatimukset ja toteutettavan ohjelmiston käyttöaika. Lyhyessä, parin työntekijän ja pienen asiakkaan projektissa ei tarvita samaa määrää järjestelmällisyyttä ja muodollisia toimintatapoja kuin isossa ja pitkässä projektissa, jossa asiakkaana on esimerkiksi pörssiyritys tai valtio. Liian kevyt prosessi ei ohjaa riittävästi, kun taas liian tarkkaan määritelty prosessikuvaus ohjaa liikaa eikä jousta tarvittaessa. Sopivan kevyen ja väljän kuvauksen tekeminen vaatiikin usein kompromisseja. On myös mahdollista määritellä useampi vaihtoehtoinen prosessi, joista valitaan sopivin kuhunkin projektiin. (14, s. 150–151.)

Ohjelmistoyrityksen keskeisimmät prosessit

Ohjelmistotuotannon käytännöt -kirjassa Haikala ja Mikkonen määrittelevät ohjelmistoyrityksen ydinprosesseiksi asiakas-, tuotekehitys ja ylläpitoprosessit (14, s. 140). Ohjelmistoyrityksillä erityyppisiä projekteja ovat asiakaskohtainen eli täysin räätälöity projekti, valmisohjelmiston tuotekehitys ja näiden välimuotona paketoitava ohjelmisto, jossa aiemmin toteutettu järjestelmä muokataan pienillä muutoksilla uuden asiakkaan käyttöön. Yleensä projektit ovat luonteeltaan jotakin näiden kolmen vaihtoehdon väliltä. (14, s. 140–141.)

Asiakaskohtaisessa prosessissa toteutetaan asiakkaan tarpeiden mukainen järjestelmä. Sen vaiheita ovat määrittely, toteutus, testaus ja toimitus. Kun tehdään paketoitavia järjestelmiä, toteutuksen sijaan vaiheena on paketointi eli asiakaskohtaisten asetusten tai muutosten tekeminen valmiin järjestelmän pohjaan. Myös tuotekehityksessä vaiheina

ovat määrittely, toteutus ja testaus. Määrittelyjä ei kuitenkaan tehdä ulkopuolisen asiakkaan, vaan esimerkiksi oman yrityksen myynti- ja markkinointiosastojen toiveiden mukaan. Prosessissa voidaan toteuttaa uusi ominaisuus jo olemassa olevaan tuotteeseen, jolloin lopputuloksena on uusi versio tuotteesta. (14, s. 141–142.)

Näiden lisäksi voidaan puhua yleisemmin tuotantoprosessista, joka on keskeinen tuotteita tai palvelua valmistavalle yritykselle. Se sisältää kaikki toiminnot, joiden avulla tuote toteutetaan. Näitä ovat esimerkiksi tilaus, tuotespesifikaation määrittely, asiakaskohtainen suunnittelu ja tuotteen valmistus. Lisäksi siihen katsotaan kuuluvan jakelun suunnittelu ja toteutus sekä materiaalien hankinta ja alihankkijoiden ohjaus. (18, s. 350–351.) Viimeksi mainitut pätevät paremmin fyysisiä tuotteita valmistavien yritysten kohdalla, mutta laajemmin ajateltuna sopivat myös ohjelmistoyrityksen prosessiin.

3.2 Toimintatapojen kuvaaminen

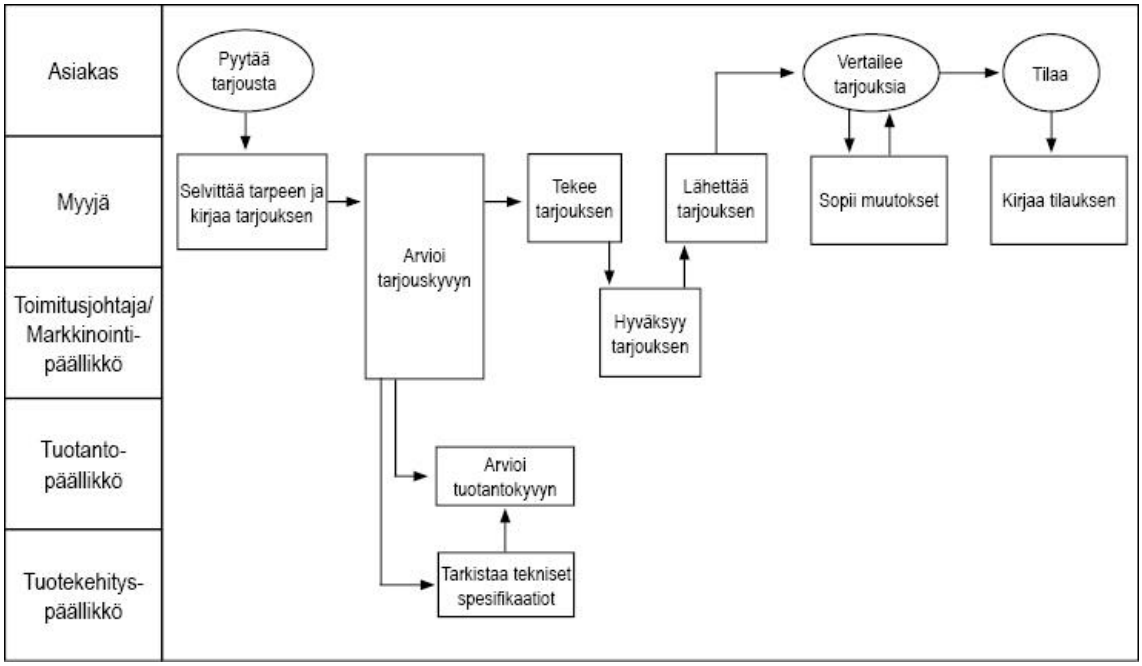
Prosessien kuvaaminen helpottaa yrityksen toiminnan ymmärtämistä (15, s. 96). Laamasen (15, s. 23) mukaan työntekijät tuntevat yleensä oman osastonsa työnkuvan, mutta eivät tiedä tarkkaan, mitä muualla yrityksessä tehdään. Prosessikuvaukset auttavat työntekijöitä hahmottamaan yrityksessä tehtävän työn kokonaisuutena, jolloin myös oma rooli osana kokonaisuutta selkiytyy. Työntekijöistä tulee aiempaan monitaitoisempia, ja arvostus muiden työntekijöiden työtä kohtaan kasvaa. (15, s. 22–23.) Hyvä prosessikuvaus myös edistää prosessissa mukana olevien työntekijöiden yhteistyötä (15, s. 76). Tarkkojen kuvausten tarve ei ole yhtä selkeä pienessä yrityksessä, jossa toimintatavat on helppo sopia yhdessä keskustellen. Uusien työntekijöiden myötä tilanne voi kuitenkin muuttua. Perehdytys on helpompaa, jos toimintatavat löytyvät myös kirjallisessa muodossa, sillä uusi työntekijä ei voi heti muistaa kaikkea, mitä hänelle parin ensimmäisen päivän aikana kerrotaan, eivätkä perehdyttäjät välttämättä edes huomaa mainita kaikkea oleellista. (14, s. 139.)

Kun työskennellään prosessikuvausten mukaisesti, yrityksen toimintaa on helpompi mitata, kehittää, ohjata ja ymmärtää. Asiat tulee tehtyä johdonmukaisesti samalla tavalla tekijästä riippumatta, jolloin toiminnan ja tuotteiden laatu on paremmin ennustettavissa. (14, s. 139–140.) Lisäksi hyvin jäsennellyt prosessit pitävät asiakkaan tyytyväisenä ja varmistavat, että toimintaa kehitetään kokonaisuutta ja asiakkaan tarpeita ajatellen (15, s. 22). Prosessikuvaukset auttavat myös hahmottamaan, mikä yrityksen toiminnassa on

kriittisintä tavoitteiden saavuttamiseksi, jolloin voidaan keskittyä oikeiden asioiden kehittämiseen (15, s. 37).

Ennen kuin prosessit voidaan kuvata, ne tulee tunnistaa ja rajata. Tässä yhteydessä määritellään prosessin alku- ja loppupiste, asiakkaat, vaiheet, tuotteet, syötteet ja toimittajat. (15, s. 52, 66.) Kuvauksesta tulisi ilmetä prosessin kannalta kriittiset asiat ja eri asioiden väliset riippuvuudet. Siihen voidaan sisällyttää esimerkiksi prosessin tehtävät ja vaiheet, tiedonkulku ja tietojenhallinta, asiakas ja asiakkaan vaatimukset, yhteistyökumppanit, tavoitteet, syötteet ja tuotokset, mukana olevat ihmiset, roolit ja vastuut, prosessiin liittyvät käsitteet ja termit sekä välineet, tilat ja muut resurssit. (15, s. 76–78.)

Prosessi kuvataan tavallisesti vuokaaviona, jossa esitetään aikajärjestyksessä yhden prosessin vaiheet ja vaiheiden suorittamisesta vastaavien osapuolten roolit (17, s. 284; 15, s. 79). Kaavion voi piirtää useilla eri tavoilla, erilaisia symboleja käyttäen. Kuvassa 3 on Laamasen (15, s. 79) suosituksen mukainen esimerkki prosessikaaviosta. Esimerkin tarkoituksena on kuvata toimintaa niin, että sitä on helpompi ymmärtää. Jos tarkoituksena olisi parantaa toimintaa tai kehittää tietojärjestelmiä, voidaan tarvita yksityiskohtaisempi kaavio. (15, s. 79.)

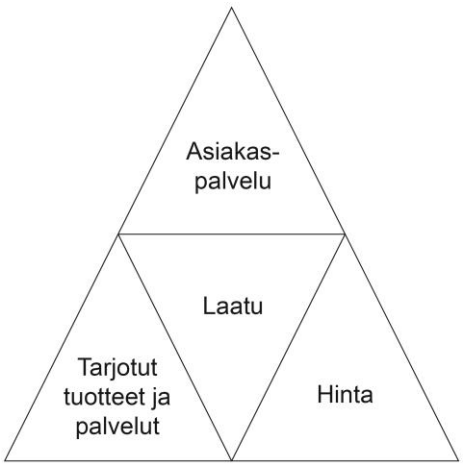


Kuva 3. Esimerkki tarjousprosessin kaaviosta (15, s. 79).

Kaavio voidaan piirtää myös niin, että roolit tai osastot on merkitty kaavion yläreunaan, mutta Laamasen mukaan niiden merkitseminen vasempaan reunaan antaa selkeämmän kuvan prosessin etenemisestä, sillä tässä tavassa vaiheet kuvataan aikajärjestyksessä vasemmalta oikealle. Hän myös suosii henkilöroolien käyttöä osastojen sijaan, jotta ihmisten on helpompi nähdä itsensä mukana osana prosessia. Asiakkaan tärkeyden korostamiseksi asiakas tulee merkitä yhdeksi rooliksi. Kaaviossa kannattaa käyttää mahdollisimman vähän eri symboleita, jotta se pysyy yksinkertaisena. Prosessin vaiheet eli tehtävät kuvataan neliöinä ja tiedonkulku nuolella. Asiakkaan toimia voidaan korostaa merkitsemällä ne neliön sijaan soikioina. Tarvittaessa kaaviossa voidaan esittää myös prosessissa liikkuva tieto, yhteys toisiin prosesseihin ja tietojärjestelemiin, aikataulutietoja tai kriittisten vaiheiden korostaminen. (15, s. 80–82.)

3.3 Prosessien kehittäminen

Prosessien parantamisen tarve voi syntyä esimerkiksi uuden tietojärjestelmän käyttöönoton yhteydessä, uusien teknologioiden kehittymisestä tai halusta parantaa yrityksen kilpailukykyä. Kehitysprojektin voi käynnistää myös reagointi tunnistettuun ongelmaan, kuten negatiiviseen asiakaspalautteeseen, kilpailutilanteen muuttumiseen tai lakimuutokseen. (15, s. 202; 17, s. 271.) Yrityksen kilpailukyky muodostuu kolmesta tekijästä, jotka ovat hinta, asiakaspalvelu ja tarjotut palvelut tai tuotteet (19). Kilpailukyky voidaan kuvata kolmiona, joka on esitetty kuvassa 4.



Kuva 4. Yrityksen kilpailukyky muodostuu hinnasta, asiakaspalvelun tasosta ja tarjottavista palveluista tai tuotteista (19).

Kuvan 4 mukaisesti kilpailukyvyn kolme kantaa muodostavat yhdessä kolmion keskiosan eli laadun. Yhtä tekijää ei voi muuttaa vaikuttamatta samalla muihin: esimerkiksi hinnan laskeminen tarkoittaa, ettei yrityksellä ole enää varaa tarjota samantasoista palvelua kuin aiemmin. (19.) Prosesseja kehitettäessä tavoitteena voi olla parantaa kilpailukykyä esimerkiksi kehittämällä laatua ja palvelua, lisäämällä joustavuutta, vähentämällä kustannuksia, nopeuttamalla prosessia tai lisäämällä tuottavuutta (7, s. 128).

Prosessien muutoksen käynnistymiseen vaikuttavat negatiiviset työntötekijät eli asiat, joiden vuoksi halutaan siirtyä pois vanhasta toimintamallista. Lisäksi siihen vaikuttavat positiiviset vetotekijät, jotka tekevät uuteen toimintatapaan siirtymisestä houkuttelevampaa. Kun näitä tekijöitä on riittävästi, muutos koetaan tarpeelliseksi. (20, s. 41.)

Kehitysprojektin onnistumisen kannalta on tärkeää, että yrityksen johto on sitoutunut hankkeeseen. Uudistamiseen tarvitaan johdolta riittävästi aikaa ja taloudellisia resursseja, jotta kehitysprojekti ei unohdu asiakasprojektikiireiden lomassa. (14, s. 150.) Onnistuminen edellyttää myös nykyisen toimintatavan ja sen ongelmien ymmärtämistä (17, s. 293). Muutossuunnitelmista ja muutoksen tarpeellisuuden syistä tulee tiedottaa koko yritykselle (17, s. 271). Kaikkien työntekijöiden tulisi olla mukana projektin eri vaiheissa, jotta koko organisaatio on motivoitunut ja sitoutunut muutoksen toteuttamiseen (20, s. 44). Mitä aikaisemmassa vaiheessa prosessiin osallistuvat työntekijät pääsevät mukaan kehitystyöhön, sitä parempia tuloksia saadaan käyttöönottovaiheessa (7, s.192).

Kehittämisen muodot

Yritykset ja niiden toimintaympäristöt muuttuvat koko ajan, joten myös toimintatavat tarvitsevat jatkuvaa kehittämistä. Muutos voi tapahtua niin hitaasti ja pienissä osissa, ettei sitä edes huomaa. (20, s. 39–40.) Toimintatapoja voidaan kehittää joko jatkuvasti pienin askelin tai radikaalisti suunnittelemalla koko prosessi uudelleen. Jatkuvassa kehityksessä pyritään parantamaan olemassa olevia toimintatapoja, kun taas uudelleensuunnittelussa aiemmat rakenteet ja toiminnot kyseenalaistetaan ja prosessin suunnittelu aloitetaan puhtaalta pöydältä. (14, s. 150; 17, s. 101.) Menetelmän valinta riippuu tilanteesta ja prosessista, eivätkä ne ole toisiaan poissulkevia vaihtoehtoja (17, s. 103). Ohjelmistokehityksessä uudelleensuunnittelu on usein liian riskialtista. Jopa pienet muutokset voivat ennen vakiintumistaan aiheuttaa hetkellistä tuottavuuden laskua. (14, s. 150.)

Jatkuva parantaminen muodostuu useista pienistä, vaiheittaisista ja ennakoitavista muutoksista, jotka yhdessä kehittävät prosessia. Kehittämiseen osallistuvat kaikki työntekijät, ja toteutettavat muutokset perustuvat usein työntekijöiden parannusehdotuksiin. Kehittäminen voi tapahtua esimerkiksi koko yrityksen kattavan laatuohjelman avulla. Menetelmän haasteena on jatkuvan muutoksen ylläpitäminen ja johdonmukaisen kehittämisen varmistaminen. (17, s. 105; 15, s. 207.) Jatkuvaa kehittämistä ovat esimerkiksi uusien tekniikoiden ja työvälineiden käyttöönotto, työohjeiden uudistaminen ja menettelytapojen sujuvoittaminen. Jos kehitetään vain yhtä työvaihetta, muutokset eivät välttämättä edes näy prosessikaaviossa. (7, s. 150.)

Uudelleensuunnittelu kohdistuu yleensä yhteen ydinprosessiin. Muutosidea on usein lähtöisin yrityksen johdolta, monesti se perustuu uusiin teknologioihin ja tietotekniikkaan. Muutoksen toteuttamiseen osallistuu pieni muutostiimi. Tavoitteena on huomattava parannus, joka toteutetaan kerralla. Riski muutoksen onnistumisessa on suuri, ja haasteisiin kuuluukin riskien tunnistaminen ja niiden poistaminen. (17, s. 105; 15, s. 207.)

Kehitysprosessin vaiheet

Kehitysprosessi voidaan jakaa lähteestä riippuen esimerkiksi 3–7 eri vaiheeseen. Vaiheet pitävät sisällään muun muassa johdon odotusten kartoittamisen, vision ja muutostavoitteiden määrittämisen, prosessin nykytilanteen analysoinnin, uuden toimintamallin suunnittelun, toteutuksen, käyttöönoton ja muutosvalmiuden hallinnan. (20, s. 37–38; 7, s. 134; 16, s. 45.) Lecklin on tiivistänyt kehittämismallin kolmeen vaiheeseen, joita ovat nykytilan kartoitus, prosessianalyysi ja prosessin parantaminen (7, s. 134).

Nykytilan kartoitus -vaiheessa laaditaan prosessikuvaus ja -kaavio nykyisestä prosessista ja arvioidaan prosessin toimivuutta (7, s. 134). Vaiheeseen osallistuu työntekijöitä kaikista organisaation eri ryhmistä. Prosessia analysoimalla ja yhdessä keskustelemalla pyritään selvittämään toimintatapojen heikkoudet ja saavuttamaan yhtenäinen näkemys nykytilanteesta ja tarvittavista muutoksista. (20, s. 43–44.) Muutoksen onnistumisen kannalta on tärkeää, että sen päämäärä on selkeä (20, s. 38). Lisäksi voidaan kartoittaa yrityksen muutosvalmiutta haastattelemalla johtoa ja työntekijöitä (20, s. 36).

Analysointivaiheessa valitaan kehittämistapa ja luodaan suunnitelma prosessin kehittämiselle (7, s. 135, 148). Ideoinnin apuna voidaan käyttää esimerkiksi aivoriihiä ja benchmarking-tekniikkaa eli oman yrityksen vertailua muihin. Vaiheen aikana selvitetään

prosessin ongelmat kartoitusvaiheessa kerättyjen tietojen pohjalta. Ongelmat voidaan yrittää ratkaista esimerkiksi erilaisia ongelmanratkaisutekniikoita ja tilastollisia menetelmiä käyttäen. Analysoinnin yhteydessä voidaan kehittää myös mittaus- ja ohjausjärjestelmä, jonka avulla prosessin toimivuutta voidaan jatkossa seurata. (7, s. 149.)

Analysoinnin jälkeen alkaa prosessin parantamisen vaihe. Aluksi laaditaan parannussuunnitelma, johon voi kuulua esimerkiksi parannustavoitteet, uusi prosessikuvaus ja toteutussuunnitelma pilottiprojektille. (7, s. 191.) Hyvä tavoite on esitetty numeroilla ja kiinnitetty aikaan, jotta sen toteutumista voidaan mitata ja arvioida (15, s. 203; 14, s. 138.) Suunnitelmaan voidaan kirjata myös tiedot vastuista, aikatauluista, mittausjärjestelmästä ja tarvittavista resursseista (7, s. 191). Suunnitelma tulisi hyväksyttää johdon lisäksi kaikilla prosessissa mukana olevilla työntekijöillä, jotta he ovat valmiita muutokseen ja käyttöönotto onnistuu mahdollisimman hyvin (7, s. 192). Ennen varsinaista käyttöönottoa uuden prosessin toimivuutta ja suorituskykyä on mahdollista testata koeprojektin avulla. Tulosten perusteella voidaan vielä parantaa prosessin ohjeistusta. (7, s. 194.)

Jos toteutettava muutos on suuri, käyttöönoton avuksi voidaan laatia erillinen suunnitelma. Dokumentissa voidaan käsitellä esimerkiksi käyttöönoton kuvaus, aikataulu, välitavoitteet, prosessin mittaus, resurssit ja koulutus. (7, s. 193–194.) Suurta muutosta ei voida toteuttaa kerrallaan, vaan sopivien välitavoitteiden avulla (15, s. 41). Ihmisillä on tapana suhtautua muutokseen skeptisesti, vaikka muutos vaikuttaisi hyvältä (20, s. 63). Muutoksen onnistumiseksi vanhoista toimintatavoista täytyy kuitenkin osata luopua (20, s. 41). Muutosvastarinnan ehkäisemiseksi työntekijät tulee ottaa mukaan kehitysprosessiin, muutoksesta tulee tiedottaa riittävästi ja henkilökuntaa täytyy kouluttaa ennen uuden prosessin käyttöönottoa (16, s. 105).

Jotta prosessin uudistus onnistuu hyvin, työntekijöille tulee antaa palautetta heti käyttöönoton jälkeen. Onnistumiset tulee huomioida positiivisesti, virheisiin ja laiminlyönteihin tulee puuttua. Hyvistä suorituksista voidaan tiedottaa myös muille työntekijöille. On hyvä kuitenkin muistaa, etteivät muutoksen vaikutukset yleensä näy heti, sillä uuden oppiminen vie aikaa ja tulokset paranevat viiveellä. Käyttöönottovaiheessa kommunikaation tulee toimia hyvin ja apua tulee olla saatavilla ongelmatilanteissa. Jos työntekijä turhautuu uudistuksiin jo alussa, hän saattaa haluta palata vanhoihin menetelmiin. Koulutuksen lisäksi hyvä dokumentaatio ja ohjeistus sujuvoittavat siirtymistä uusiin toimintata-

poihin. Prosessin toimivuutta tulee seurata myös käyttöönoton jälkeen. On hyvä selvittää, noudatetaanko uusia työtapoja, toimiiko prosessi suunnitellusti ja onko asetetut tavoitteet saavutettu. (7, s. 196.)

4 Elementtikirjaston suunnittelu

4.1 Projektin tausta

Tein insinöörityön yritykselle, jossa olin samalla itse töissä teknisenä toteuttajana. Suunnittelussa auttoi kokemus yrityksen toimintatavoista ja tieto siitä, että tulen itsekin käyttämään sovellusta. Hyvä lähtökohta oli myös se, että projektin alkaessa olin ollut yrityksessä töissä vasta muutaman kuukauden, joten en ollut tottunut vanhoihin toimintatapoihin, niin kuin yrityksessä vuosia olleet työntekijät.

Yrityksessä tehtyjä projekteja ja elementtejä ei aiemmin ole dokumentoitu keskitetysti kaikkien työntekijöiden saataville. Toteutettuihin projekteihin liittyvät tiedostot ovat hajallaan työntekijöiden omilla tietokoneilla, yhteisellä verkkolevyllä ja yrityksen verkkopalvelimella. Tämä ei ole aiemmin aiheuttanut suuria ongelmia työntekijöiden pienen määrän vuoksi. Kun yksi ihminen vastaa sisällöntuotannosta, yksi käyttöliittymän suunnittelusta ja yksi toteutuksesta, kaikki tietävät melko hyvin, mitä aiemmin on toteutettu. Työntekijöiden ja tehtyjen projektien määrän kasvaessa paremmalle dokumentoinnille on kuitenkin muodostunut selkeä tarve.

Idea elementtikirjastosta syntyi etenkin teknisten toteuttajien tarpeista. Yritykseen oli palkattu useita uusia työntekijöitä teknisiksi toteuttajiksi, mutta perehdytys oli työlästä, koska kaikki vanhat toteutukset eivät olleet helposti nähtävillä. Lisäksi projekteissa käytettävät koodipohjat olivat toisinaan melko vaikeaselkoisia ja kasvaneet niin pitkiksi tiedostoiksi, että oikea kohta oli välillä hankala löytää, jos toiminnallisuutta piti muokata. Olennainen osa elementtikirjastoa on elementtien koodi, joka tulee tallentaa paikkaan, josta se saadaan helposti käyttöön uusissa projekteissa. Koodipohjien selkeyttäminen ja pilkkominen pienempiin osiin on elementtikirjaston kannalta tärkeää, mutta varsinaiseen koodin rakenteeseen ei oteta insinöörityössä kantaa.

Koodien tallennuspaikan lisäksi toinen tärkeä elementtikirjaston osa on käyttöliittymä, jonka avulla voi selata tehtyjä elementtejä kategorioittain. Yksi kategoria voisi olla tehtävätyypit, jonka alla olisi alakategoriat eri tehtävätyypeille (esimerkiksi monivalinta ja oikein/väärin-tehtävä). Elementin yhteydessä olisi esimerkkitoteutus, kuvaus ja muuta hyödyllistä tietoa, kuten lista projekteista, joissa elementtiä on käytetty. Käyttöliittymästä olisi teknisten toteuttajien lisäksi hyötyä myös muille työntekijöille: esimerkiksi suunnittelijat voivat sen kautta helposti selata vanhoja toteutuksia saadakseen ideoita uusiin projekteihin.

Yrityksen tuotantoprosessi

Koska elementtikirjasto vaikuttaa yrityksen tuotantoprosessiin, suunnittelu aloitettiin kartoittamalla sen vaiheet. Prosessikaavio on kuvattu liitteessä 1. Tuotantoprosessi alkaa, kun asiakas on päättänyt tilata sovelluksen. Toteutettava sovellus on usein koulutus, joka on suunnattu tilaajana toimivan yrityksen työntekijöille. Myynnin ja tarjouksen tekemisen yhteydessä asiakkaan kanssa on jo käyty läpi koulutuksen kesto, tavoitteet, käyttötarkoitus, sisältö ja käyttöympäristö, eli sovelluksen ominaisuudet ovat suurimmaksi osin tiedossa. Myyntiprosessin yhteydessä sovelluksesta on usein jo tehty miellekarttana alustava luonnos sisältö- ja rakennesuunnitelmasta, jonka kehitystä tuotantotiimi ja projektipäällikkö jatkavat tuotantoprosessin aikana.

Tilauksen jälkeen saadaan yleensä asiakkaalta lisää materiaalia, jonka pohjalta koulutus suunnitellaan. Joskus asiakas on suunnitellut rakenteen ja sisällön itse, mutta yleensä Apprix suunnittelee ainakin koulutuksen pedagogisen puolen. Sisältö jaotellaan sopiviin osioihin ja samalla mietitään, millaisia elementtejä sen esittämiseen tarvitaan. Koulutukseen voidaan laittaa esimerkiksi monivalintatehtäviä, joilla käyttäjän oppimista testataan. Ideoita voidaan hakea aiemmin toteutetuista koulutuksista, mutta niin tehdään melko harvoin, koska vanhojen projektien etsiminen ja läpi selaaminen on hidasta. Suunnitteluvaiheessa asiakkaalle voidaan myös ehdottaa erilaisia elementtejä ja toteutustapoja, joista saatetaan näyttää esimerkkejä aiemmin toteutetuista sovelluksista.

Sisällön ja rakenteen lisäksi suunnitellaan toiminnallisuudet, eli esimerkiksi se, miten koulutuksessa siirrytään näkymästä toiseen. Käyttöliittymäsuunnittelija miettii, miltä elementit näyttävät ja miten ne toimivat, ja tekee ulkoasusta mockupin eli mallin joko inter-

aktiivisena tai kuvina. Suunnitelmat toimitetaan asiakkaalle kommentoitavaksi. Palautteen perusteella saatetaan vielä tehdä muutoksia, minkä jälkeen on valmiina lopullinen tuotantosuunnitelma.

Kun suunnitelmat on hyväksytty, alkaa tekninen toteutus. Samaan aikaan sisällöntuottaja kirjoittaa tarvittavia tekstejä ja käyttöliittymäsuunnittelija toteuttaa sisältöjen ulkoasumalleja (layout). Tekninen toteuttaja ottaa toteutuksen pohjaksi yleensä jonkin vanhan projektin. Usein se on toteutus, jota tekninen toteuttaja on ollut itse tekemässä, koska rakenne ja koodi ovat tuttuja ja tiedostot helposti saatavilla. Jos uudessa projektissa on jokin elementti, jota tekninen toteuttaja ei ole aiemmin toteuttanut, hän voi kysellä muilta työntekijöiltä, löytyisikö sellainen valmiina jostain toisesta projektista. Jos ei, elementti muokataan samantapaisesta elementistä tai tehdään kokonaan alusta itse.

Kun sovelluksen kokonaisrakenne on valmiina ja suurin osa sisällöstä on paikallaan, asiakkaalle toimitetaan linkki sovellukseen. Asiakkaan kommentteja odotellessa sovelluksen kehitystä ja testaamista jatketaan. Kommenttien perusteella tehdään tarvittavat muutokset ja päivitetään uusi versio asiakkaan nähtäville. Kun sovellus on valmis, se toimitetaan asiakkaalle tai otetaan käyttöön yrityksen omalla palvelimella. Projektin tiedostot jäävät talteen palvelimelle, verkkolevylle, teknisten toteuttajien omille kiintolevyille ja yrityksen käyttämään ohjelmistoprojektien versionhallintapalveluun. Asiakas julkaisee sovelluksen, minkä jälkeen se siirtyy ylläpitovaiheeseen.

Aina prosessi ei tietenkään etene näin suoraviivaisesti. Usein vaiheet menevät osittain päällekkäin, sisältöön ja ulkoasuun voi tulla muutoksia missä vaiheessa tahansa tai tekninen toteutus aloitetaan, ennen kuin kaikki sisältö on valmiina.

4.2 Kysely yrityksen työntekijöille

Elementtikirjaston ideasta keskusteltiin aluksi yrityksen johdon ja pitkäaikaisimman teknisen toteuttajan kanssa. Keskustelujen pohjalta aloin suunnitella elementtikirjaston ominaisuuksia ja toteutustapaa. Tarkoitus oli, että elementtikirjastosta olisi hyötyä kaikille yrityksen työntekijöille, joten myös muut työntekijät pyrittiin pitämään mukana suunnitteluprosessissa. Jotta työntekijöiden näkemykset tulisivat esiin jo alkuvaiheessa, suunnittelin kyselyn (liite 2), jonka avulla kartoitettiin kaikkien mielipiteitä ja toiveita elementtikirjaston suhteen. Samalla projekti esiteltiin niille, jotka eivät siitä vielä olleet kuulleet. Linkki

kyselyyn lähetettiin kaikille työntekijöille yrityksen sisäisellä keskustelukanavalla. Kyselyyn vastasi yhteensä seitsemän henkilöä.

Elementtikirjaston käyttötavat ja hyödyt

Kyselyn avulla pyrittiin selvittämään muun muassa elementtikirjaston erilaisia käyttötapoja ja hyötyjä. Vastauksissa nousi esiin myös sellaisia asioita, jotka eivät itselleni olleet aiemmin tulleet mieleen. Samalla vahvistui kuva siitä, että elementtikirjastolle on todella tarvetta. Kirjaston käyttäminen esimerkiksi parantaisi tiedon siirtymistä työntekijältä toiselle, helpottaisi kommunikointia ja estäisi hyvien ideoiden unohtumisen. Tällä hetkellä työntekijä ei yleensä tiedä, mitä tehdään niissä projekteissa, joissa itse ei ole mukana. Kirjastosta näkisi helposti myös muiden tuotoksia. Välillä hyvin toteutetut tehtävätyypit tai koodipätkät saattavat unohtua jopa niiden toteuttajilta, mutta elementtikirjastossa ne olisivat tallessa. Elementtien yhtenäinen nimeäminen parantaisi kommunikaatiota, jos kaikki käyttäisivät elementeistä kirjastossa olevia nimiä. Kommunikaatio paranisi ja väärinymmärrykset vähenisivät myös elementtien toiminnasta keskusteltaessa, kun kirjastossa olisi elementeistä esimerkki, jota klikkailemalla näkisi, miten elementti toimii.

Käyttöliittymä- ja sisältösuunnittelijoille kirjasto toimisi projektien suunnitteluvaiheen apuna. Kirjastosta näkisi suoraan kaikki aiemmin toteutetut elementit, eikä sopivaa elementtiä miettiessä tarvitsisi toimia vain muistin varassa tai etsiä vanhoja toteutuksia referenssimateriaaliksi kiintolevyjen uumenista. Sisällölle tulisi valittua sopivin mahdollinen tehtävätyyppi, kun kaikki erilaiset mahdollisuudet ovat tiedossa. Miellekarttana tehtävään sisältösuunnitelmaan voisi suoraan merkitä, mikä elementti mihinkin kohtaan on tulossa. Lisäksi sisällöntuotanto nopeutuu, kun kirjastossa olevasta esimerkistä näkee suoraan, mitä tekstejä ja ohjeita tiettyyn tehtävätyyppiin tarvitaan.

Myyjille, jotka toimivat myös projektipäällikköinä, kirjasto tarjoaisi materiaalia ja ideoita asiakasprojektien myynti- ja konseptointitilanteisiin. Kirjastosta voisi myös tarvittaessa näyttää asiakkaalle esimerkkejä yksittäisistä elementeistä. Aiemmin esimerkkejä on näytetty lähinnä vanhoista projekteista, mutta kaikkien projektien kohdalla tämä ei onnistu salassapitosopimusten vuoksi. Yleensä esittelyversion tekeminen aiheuttaa myös lisätyötä. Koska käytössä olevat sovellukset ovat usein koulutuksia, käyttäjistä kerätään tilastotietoja, kuten käyttäjän nimi, sähköpostiosoite, suoritusaika ja saadut pisteet. Esittelyä varten sovelluksesta tulee tehdä versio, jossa tietojen keräys on kytketty pois päältä, jotta tilastoihin ei ilmesty ylimääräisiä testikäyttäjiä.

Elementtikirjaston myötä teknisten toteuttajien työ nopeutuu, kun elementtien uudelleenkäyttö helpottuu. Sopivaa tehtävätyyppiä miettiessä ei tarvitsisi kysellä, muistavatko muut, onko tietynlaista tehtävää tehty aiemmin, vaan vanhoja toteutuksia voisi selata helposti itse. Jos yrityksen työntekijät vaihtuvat, kysyminen ei välttämättä ole edes mahdollista. Kirjastosta olisi paljon apua myös uusille työntekijöille. Vanhoja toteutuksia selaamalla saisi nopeasti selkeän kuvan siitä, mitä kaikkea projekteissa yleensä tehdään. Ongelmatilanteessa kirjastosta löytyisi avuksi ohjeita ja esimerkkejä. Muiden työntekijöiden voisi olla myös helpompi perehdyttää uutta työntekijää näyttämällä toteutuksien esimerkkejä kirjastosta, sen sijaan että esiteltäväksi haettaisiin vanhoja asiakasprojekteja.

Elementtikirjastossa olevan esimerkkitoteutuksen ja pieniin osiin pilkotun koodin ansiosta muiden toteuttamien elementtien toiminta olisi helppo ymmärtää, etenkin jos koodien yhteyteen on liitetty ohjeistus. Koodin pilkkominen pienempiin osiin myös vähentäisi mukana olevan turhan koodin määrää. Nykyään valmiit koodipohjat ovat pitkiä, eikä niistä aina ymmärrä, mikä vaikuttaa mihinkin, joten koodia ei välttämättä uskalla karsia. Elementtikirjasto voisi myös parantaa toteutusten laatua. Jos työntekijä kehittää elementtiä esimerkiksi lisäämällä siihen hyödyllisen ominaisuuden, hän voi päivittää muutokset elementtikirjastoon, jolloin muutkin saavat parannukset käyttöönsä tulevissa projekteissa. Lisäksi tieto siitä, että elementit menevät kaikkien käyttöön, voi motivoida teknisiä toteuttajia kirjoittamaan laadukkaampaa koodia ja välttämään huonoja ”pikaratkaisuja”.

Toteutuskieli

Kyselyssä kysyttiin mielipidettä toteutuskielestä. Olisiko se englanti, suomi vai molemmat? Suurin osa työntekijöistä oli sitä mieltä, että englanti olisi paras. Tällä hetkellä yrityksessä on vain suomenkielisiä työntekijöitä, mutta tulevaisuudessa kaikki työntekijät eivät välttämättä puhu suomea. Kaikki nykyiset työntekijät kuitenkin osaavat englantia, joten kahden kielen toteutus olisi turha. Monista elementeistä käytetään jo nyt englanninkielisiä nimiä, ja esimerkiksi kaikki koodi kirjoitetaan englanniksi. Usein myös toteutettavat projektit tehdään englanniksi tai monikielisiksi.

Kirjastoon lisättävät elementit

Kirjastoon lisättäviä elementtejä käsittelevässä kysymyksessä esimerkkeinä mainittiin tehtävätyypit, kirjautuminen ja palautesivu. Vastauksissa etenkin yleisimpiä tehtävätyyppejä ja muita usein käytettäviä elementtejä pidettiin tärkeinä. Lisäksi ehdotettiin, että joitain elementtejä voisi laittaa kirjastoon ilman koodeja, pelkkinä kuvina. Niitä olisivat esimerkiksi monimutkaiset tai vahvasti tapauskohtaiset, mutta hyvin toteutetut elementit, joiden koodi on hankala irrottaa kokonaisuudesta. Myös vanhoja, jo käytöstä poistuneilla teknologioilla tehtyjä mutta hyvin toteutettuja elementtejä voisi tarvittaessa lisätä kirjastoon kuvina, jos niitä ei ole vielä toteutettu nykyisin käytössä olevilla teknologioilla.

Lisäksi piti pohtia, millä perusteella lisättävät elementit valitaan. Miten esimerkiksi määritellään, milloin muokattu elementti on riittävän erilainen lisättäväksi elementtikirjastoon omana versionaan? Vastauksissa todettiin, että luultavasti ainakin alussa tätä pitää pohtia yhdessä tapauskohtaisesti. Kokemuksen myötä varmasti syntyy jonkinlainen käytäntö, jota voidaan noudattaa. Vastauksissa toivottiin myös, ettei kirjaston elementtilistaus täyttyisi yhden elementin useista eri versioista vaikeuttaen selausta. Uusien elementtiversioiden olisi hyvä pohjautua koodin ja toiminnallisuuden eroihin, ei visuaalisiin muutoksiin.

Elementteihin liitettävät tiedot ja elementtien selaustavat

Yksi kysymyksistä käsitteli elementtien yhteyteen liitettäviä tietoja. Esimerkkitietoina kysymyksen kuvauksessa lueteltiin elementin nimi, kategoria, elementin luonut henkilö, kuvaus, ohje, julkaisupäivämäärä, päivitykset ja lista projekteista, joissa elementtiä on käytetty. Monet pitivät näitä tietoja hyödyllisinä. Lisäksi toivottiin, että elementin yhteydessä näkyisivät projektien aiheet (esimerkiksi turvallisuusperehdytys, kilpailuoikeuskoulutus), elementin aihe eri projekteissa, koodissa käytetyt teknologiat ja kirjastot (esimerkiksi jQuery, Angular, Bootstrap), viimeisin muokkaaja, elementin versiohistoria, interaktiivinen esimerkki ja kuvia esimerkkitoteutuksista eri projekteissa. Yksi ehdotus oli arviointiasteikko, jonka avulla elementin toteuttaja voisi ilmaista, kuinka laadukasta koodi hänen mielestään on. Lisäksi kysyttiin mielipidettä mahdollisista hakutavoista. Vastauksissa toivottiin, että elementtejä voisi selata ainakin kategorian, asiakkaan, projektin aiheen ja elementin aiheen perusteella. Myös sanahaku olisi hyödyllinen.

Elementtien lisäämiseen tarvittava aika

Jotta elementtikirjastosta on hyötyä, työntekijöiden tulee käyttää sitä aktiivisesti. Kirjaston tarkoitus on säästää aikaa, mutta sen käyttö tulee välillä myös viemään sitä. Miten siis varmistetaan, että elementtien lisäämiseen varataan riittävästi aikaa? Vastauksissa oli aiheeseen liittyen paljon hyvää pohdintaa. Moni ehdotti, että elementtien lisääminen tulisi lisätä prosessin osaksi. Tehtävätyypit ja muut tarvittavat elementit ovat yleensä melko hyvin tiedossa siinä vaiheessa, kun projekti siirtyy käyttöliittymäsuunnittelijalta tekniseen toteutukseen. Ennen teknisen toteutuksen alkua voisi katsoa, mitä kirjastosta löytyy valmiina. Uusien elementtien tekemiseen, dokumentoimiseen ja kirjastoon lisäämiseen kuluva aika voitaisiin siis huomioida jo tässä vaiheessa.

Joskus elementin muokkaamisen tai lisäominaisuuksien tarve huomataan vasta projektin kuluessa. Sellaisessa tilanteessa työntekijät voisivat yhdessä arvioida, onko elementti lisättävä kirjastoon omana versionaan. Varsinainen elementin lisäys voitaisiin tehdä sopivassa kohdassa ennen projektin päättymistä. Lisäämistä ehdotettiin tehtäväksi myös projektin päätyttyä, mutta se voi helposti jäädä tekemättä lopuksi, kun pitäisi jo aloittaa seuraavaa projektia. Usein seuraava projekti on voinut jo alkaa ennen edellisen päättymistä, kun odotellaan asiakkaan kommentteja, lopullista hyväksyntää ja julkaisua.

4.3 Järjestelmän vaatimukset ja ominaisuudet

Ohjelmistojen kehittämisen lähtökohtana ovat sovellukselle asetetut vaatimukset, jotka kirjataan yleensä vaatimusmäärittelydokumenttiin (14, s. 21, 68). Vaatimukset kertovat, mitä sovelluksella voi tehdä, tai kuvaavat sen ominaisuuksia (14, s. 61). Hyvä vaatimus on tarkka, ymmärrettävä ja mitattava, eli sen toteutuminen tulee pystyä testaamaan tai todentamaan (14, s. 64). Vaatimukset on lähes mahdotonta kirjoittaa täysin tarkasti projektin alussa, sillä ne muuttuvat ja tarkentuvat aina projektin edetessä. Muutoksiin tulee siis varautua. (14, s. 22.)

Toteutettavan sovelluksen luonne vaikuttaa tarvittavan vaatimusdokumentaation laajuuteen (14, s. 64). Tämän projektin yhteydessä perusteelliselle vaatimustenmäärittelydokumentille ei ollut tarvetta. Sen sijaan sovelluksen ominaisuudet koottiin kyselyn ja yhteisten keskustelujen perusteella taulukkoon (taulukko 2). Ominaisuudet jaoteltiin heti toteu-

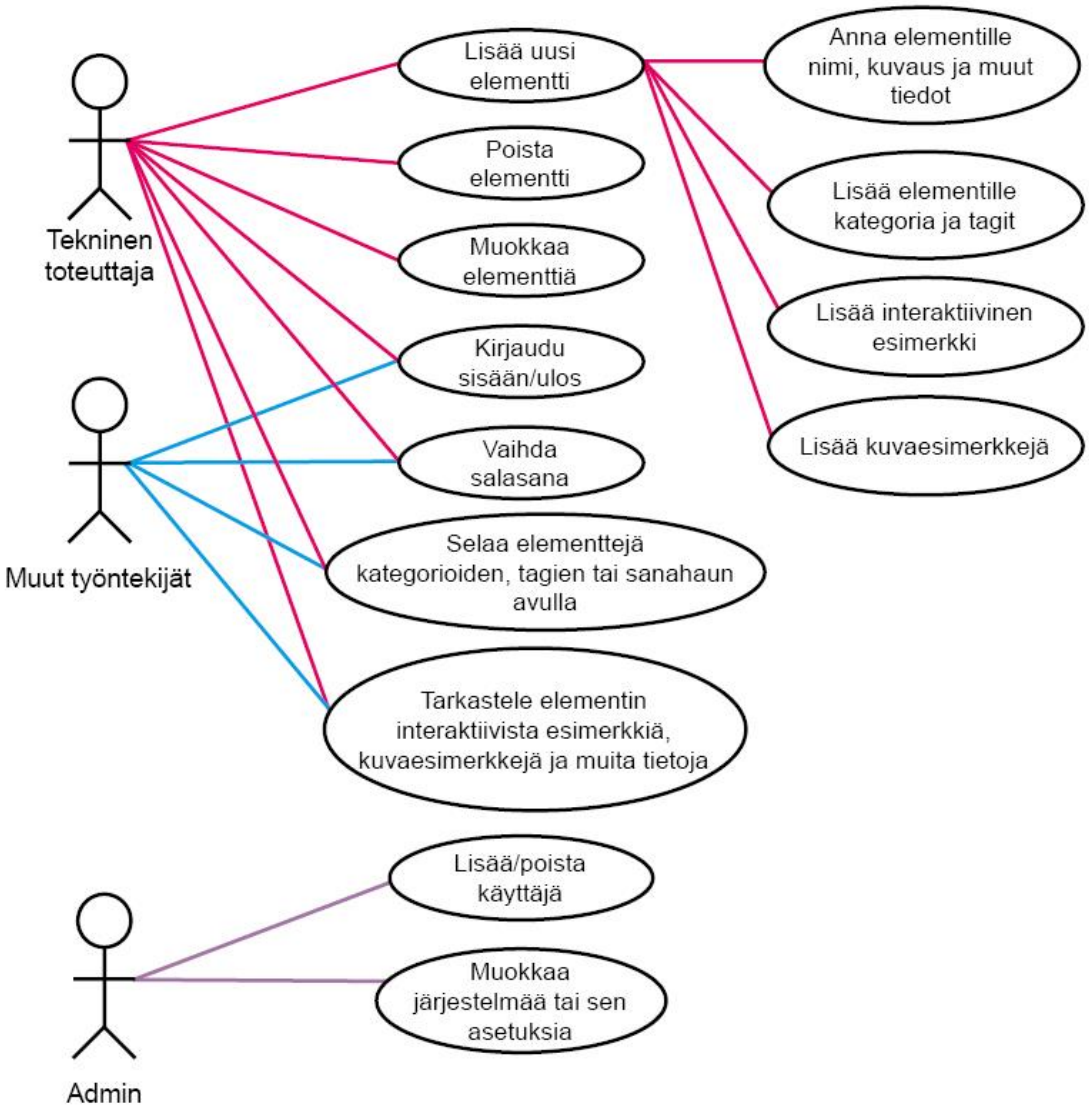
tettaviin ja myöhemmin toteutettaviin ominaisuuksiin. Muut ehdotetut ominaisuudet lisättiin toivomuslista-nimiseen ryhmään. Siihen kuuluvat ominaisuudet saatetaan toteuttaa myöhemmin, jos on aikaa tai jos niille on selkeä tarve.

Taulukko 2. Elementtikirjaston ominaisuudet.

Heti toteutettavat ominaisuudet	Elementtien lisääminen, muokkaaminen ja poistaminen Elementtien tietojen tallentaminen (nimi, tekijä, kuvaus, ohje, julkaisupäivämäärä, päivitykset, elementin tyyppi) Elementin koodien tallentaminen, muokkaaminen ja tarkastelu Interaktiivinen esimerkki elementeistä Lista projekteista, joissa elementtiä on käytetty, projektien aiheet, elementin aihe, käytetyt teknologiat Englanninkielinen käyttöliittymä Elementtien esimerkkitoteutusten lisääminen kuvina Elementtien selaaminen Elementtien hakeminen sanahaulla Elementtien selaaminen ja jaottelu elementin tyypin, asiakkaan, projektin aiheen, elementtien aiheen ja käytettyjen teknologioiden perusteella Sisään- ja uloskirjautuminen Käyttäjän luominen ja poistaminen, käyttäjätietojen muokkaaminen
Myöhemmin toteutettavat ominaisuudet	Kuvagalleria kaikista esimerkkitoteuteuskuvista
Toivomuslista	Elementtien arviointi Suomenkielinen käyttöliittymä

Ominaisuuksien listauksen lisäksi vaatimukset kuvattiin käyttötapausten avulla. Käyttötapaus kertoo, mitä sovelluksella voi tehdä ja kuka sitä käyttää. Yksi käyttötapaus sisältää yleensä useita sovelluksen toimintoja. Käyttötapaukset esitetään tavallisesti yksinkertaisena kaaviona, jossa käyttäjäryhmät on kuvattu tikku-ukkoina ja käyttötapaukset soikioina. Käyttäjät yhdistetään viivalla niihin käyttötapauksiin, joihin kyseinen käyttäjäryhmä osallistuu. (14, s. 77–78, 81.) Kaavion lisäksi jokaisesta käyttötapauksesta voidaan kirjoittaa laajempi kuvaus, jossa käsitellään tarkemmin esimerkiksi käyttötapauksen poikkeustilanteet ja tapahtuman alkutilanne ja lopputulos. Käyttötapauksia kirjoittaessa tulisi käyttää sellaista kieltä ja terminologiaa, jonka asiakas ymmärtää. (14, s. 80.) Käyttötapauskaavio toimii visuaalisena apuna, kun järjestelmän ominaisuuksista keskustellaan asiakkaan kanssa, ja varmistaa, että molemmilla osapuolilla on yhtenäinen käsitys sovelluksen vaatimuksista (14, s. 81, 83). Lisäksi se auttaa sovelluksen hahmottamisessa ja rajaamisessa sekä järjestelmän toiminnallisuuksien jakamisessa pienempiin osiin toteutusvaiheeseen siirtymistä varten (14, s. 83).

Elementtikirjaston käyttäjät jaettiin kolmeen ryhmään: admin (ylläpitäjä), tekniset toteuttajat ja muut työntekijät (myyjät, sisällöntuottajat ja käyttöliittymäsuunnittelijat). Muiden työntekijöiden jakamista omiin ryhmiinsä ei koettu tarpeelliseksi, sillä käyttäjien tarvitsemat toiminnot eivät eroa toisistaan. Käyttötapauskaavio on esitetty kuvassa 5.



Kuva 5. Elementtikirjaston käyttötapauskaavio.

Koska sivusto tulee yrityksen sisäiseen käyttöön, kaikilla käyttäjillä tulee todennäköisesti olemaan täydet oikeudet järjestelmään, eli mahdollisuus käyttää kaikkia toimintoja. Käyttötapaukset kuitenkin jaoteltiin sillä perustella, kuka luultavammin tulee mitäkin toimintoa tarvitsemaan. Esimerkiksi uusia elementtejä voi luoda kuka tahansa, mutta käytännössä tekniset toteuttajat ovat niitä, jotka toimintoa tulevat käyttämään. Tekniset toteuttajat hoitavat myös ylläpitoa eli tekevät admin-käyttäjälle määriteltyjä toimintoja.

4.4 Elementtikirjaston sisältömalli

Elementtikirjaston vaatimusten perusteella suunniteltiin järjestelmän sisältömalli. Eri sisältötyyppejä on vain yksi, elementti, joten sisältötyyppien suhdetta toisiinsa ei tarvinnut esittää kaaviona. Yksittäisen elementin osat on listattu taulukossa 3.

Taulukko 3. Elementtikirjaston sisältömalli.

Sisällön osa	**Osan kuvaus**	**Tyyppi**
Nimi	Elementille annettava kuvaava ja johdonmukainen nimi	Tekstikenttä
Kategoria	Yhdessä sovitut ylä- ja alakategoriat	Valintalaatikko
Avainsana: asiakas	Asiakkaan nimi, uusien avainsanojen lisääminen mahdollista	Pudotusvalikko/tekstikenttä
Avainsana: projektin aihe	Projektin aihe, ennalta määritelty lista	Pudotusvalikko
Avainsana: elementin aihe	Elementin aihe, uusien lisääminen mahdollista	Pudotusvalikko/tekstikenttä
Avainsana: teknologia	Käytetyt teknologiat, uusien lisääminen mahdollista	Pudotusvalikko/tekstikenttä
Kuvaus	Elementin kuvausteksti	Tekstikenttä
Ohje	Ohje elementin käyttöön	Tekstikenttä
Koodilinkki	Linkki elementin koodeihin	Tekstikenttä
Tekijä	Elementin lisännyt henkilö, määritellään automaattisesti kirjautuneen käyttäjän perusteella	Viittaus tekijään
Julkaisupäivämäärä	Julkaisupäivämäärä, lisätään automaattisesti	Päivämäärä
Kuvaesimerkki	Yksi tai useampi kuvaesimerkki elementin toteutuksesta projektissa	Kuva

Interaktiivinen esimerkki	Elementin yksinkertaistettu versio, jota klikkailemalla voi testata elementin toimintoja	Upotettu koodi

Sisältömallissa (taulukko 3) listattiin elementin osat ja osien kuvaukset. Lisäksi määriteltiin, missä muodossa sisällöt syötetään. Elementin nimi, kuvaus, ohje ja linkki elementin koodeihin kirjoitetaan tekstikenttään. Kategoriat määritellään valintalaatikoilla, joista voi valita yhden tai useamman. Avainsanat valitaan pudotusvalikosta tai kirjoitetaan tekstikenttään, jos haluttua avainsanaa ei ole ennestään käytetty. Elementin tekijän ja luontipäivämäärän on tarkoitus tulla järjestelmästä automaattisesti, eli niitä ei tarvitse erikseen kirjoittaa tai valita. Esimerkkikuvat lisätään kuvatiedostoina ja interaktiivinen esimerkki sivulle upotettuna koodina.

Elementtien luokittelu

Sopivien kategorioiden ja muiden metatietojen tarkemman suunnittelun avuksi koottiin yhdessä muiden työntekijöiden kanssa lista erilaisista elementeistä, joita sovelluksiin on toteutettu. Elementit päätettiin alustavasti jakaa kolmeen kategoriaan, joita ovat tehtävät (task element), peruselementit (basic elements) ja muut elementit (other elements).

Tehtävät ovat elementtejä, joissa käyttäjää pyydetään esimerkiksi valitsemaan tai liikuttamaan jotakin. Niiden avulla voidaan testata osaamista, kysyä mielipidettä tai kerätä tietoa. Tehtäväelementteihin kuuluvat muun muassa monivalinta (valitse yksi tai useampi annetuista vaihtoehdoista), oikein/väärin-tehtävä (valitse, onko väittämä oikein vai väärin), nelikenttä (raahaa asioita sopiviin kohtiin nelikentässä) ja liukusäädin (liu'uta säädin sopivaan kohtaan esimerkiksi asteikolla 0–100).

Peruselementit toistuvat lähes jokaisessa toteutettavassa projektissa. Niitä ovat esimerkiksi kirjautuminen, palautesivu ja sitoumussivu. Palautesivulla käyttäjää pyydetään antamaan mielipiteensä eri väittämiin asteikolla yhdestä viiteen. Sitoumussivulla käyttäjää pyydetään yleensä vahvistamaan, että hän on ymmärtänyt koulutuksen sisällön ja sitoutuu noudattamaan sovelluksessa käsiteltyjä ohjeistuksia.

Muihin elementteihin kuuluvat toteutukset, jotka eivät sovi kahteen ensimmäiseen kategoriaan. Näiden lisäksi voidaan tarvittaessa luoda uusia kategorioita. Alakategorioina käytetään eri elementtejä niin, että esimerkiksi monivalinta on alakategoria, jonka alle

kootaan elementistä eri versioita. Versiot määritellään toiminnallisuuden eroilla. Tavallisesti elementti kuuluu vain yhteen kategoriaan ja alakategoriaan.

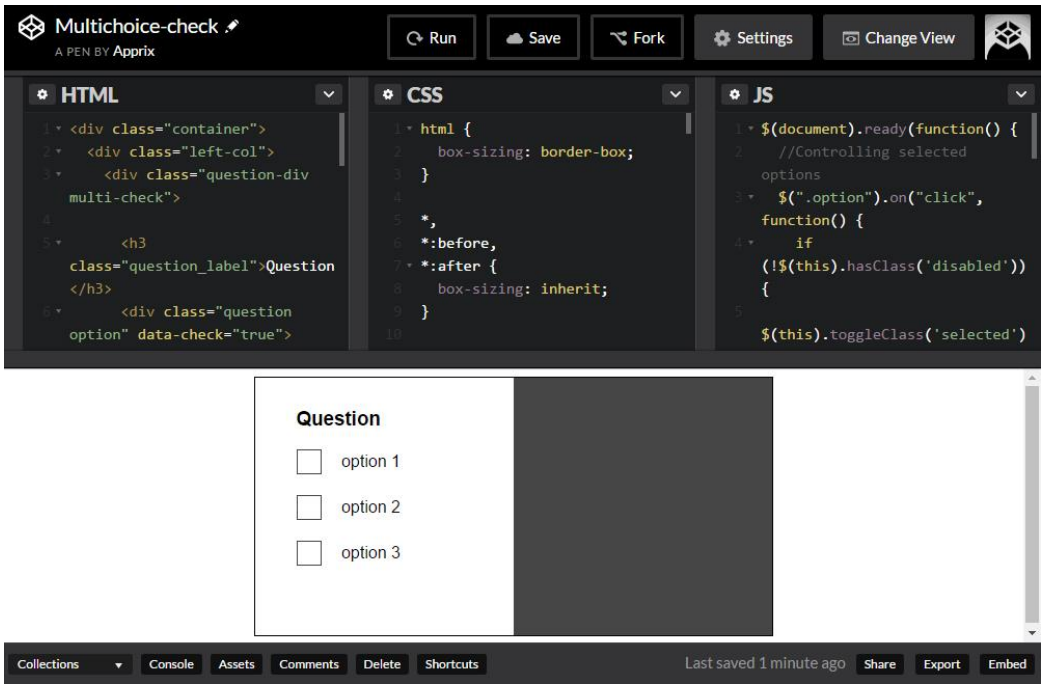
Kategorian lisäksi elementin tietoihin kerätään tageja eli avainsanoja. Avainsanat jaoteltiin eri ryhmiin, joita ovat asiakas, projektin aihe (esimerkiksi legal, communication, HR ja finance), elementin aihe (esimerkiksi turvallisuusperehdytyksessä yhden elementin aihe voi olla järjestys ja siisteys) ja käytetyt teknologiat (esimerkiksi jQuery ja Angular). Yhdellä elementillä voi olla useita avainsanoja kussakin ryhmässä: esimerkiksi monivalintaelementtiä on voitu käyttää 20 eri asiakkaan projekteissa.

5 Järjestelmän toteutus

5.1 Testielementtien luominen

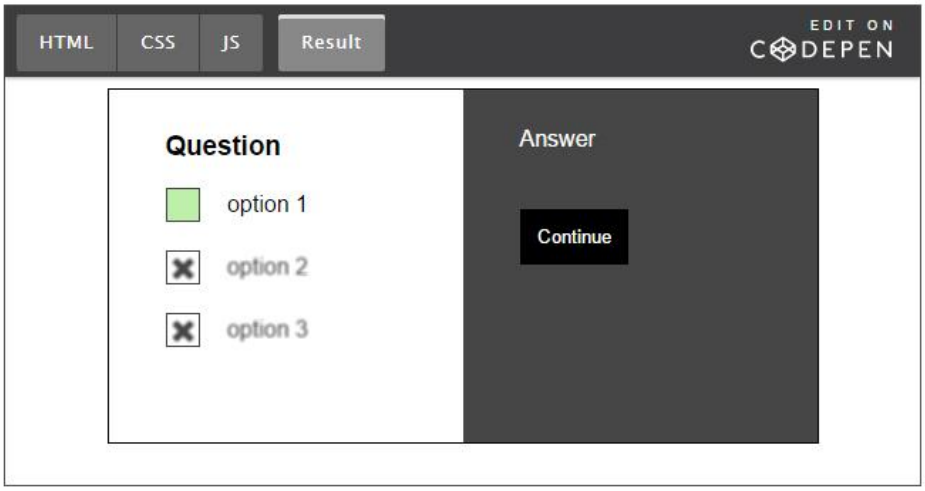
Ennen elementtikirjaston varsinaista toteutusta elementtien luomista testattiin tekemällä kaksi esimerkkielementtiä, joista kerättiin sisältömallin mukaiset tiedot taulukkoon. Näin päästiin kokeilemaan, miten elementin tekeminen ja tietojen kerääminen käytännössä toimisi. Testatut elementit olivat monivalintatehtävä ja palautesivu. Molemmista kirjattiin taulukkoon nimi, kategoria, alakategoria, kuvaus, ohje, tekijän nimi ja julkaisupäivämäärä sekä avainsanat asiakkaille, projektien aiheille, elementin aiheille ja käytetyille teknologioille. Lisäksi tallennettiin kuvat esimerkkitoteutuksista, eli kuvakaappaus siitä, miltä elementin toteutus on näyttänyt eri projekteissa.

Testielementeistä tehtiin myös interaktiivinen esimerkki, jota klikkailemalla voi kokeilla, miten elementti toimii. Esimerkkiä varten elementin koodit irrotettiin projektikokonaisuudesta ja muutettiin mahdollisimman yksinkertaiseen muotoon. Ulkoasusta poistettiin projektikohtaiset värit ja muut ylimääräiset tyylit, jotta elementti myös näyttäisi selkeältä ja yksinkertaiselta. Esimerkin esittelyä varten koodi laitettiin CodePen.io-palveluun, joka tarjoaa selaimessa toimivan editorin HTML-, CSS- ja JavaScript-koodille (21). Tulos näkyy editorissa ja päivittyy reaaliaikaisesti, kun koodia muokkaa. Kuvakaappaus editorinäkymästä monivalintatehtävää muokatessa näkyy kuvassa 6.



Kuva 6. CodePen-palvelun editorinäkymä. HTML-, CSS- ja JavaScript-koodit syötetään omiin kenttiinsä, tulos näkyy alla.

Palvelu sopii hyvin omien tuotosten esittelyyn, uusien asioiden testaamiseen, nopeiden esimerkkien rakenteluun esimerkiksi ongelmaa ratkaistaessa ja erilaisten uudelleen käytettävien komponenttien tallentamiseen. Palvelun avulla tuotokset on helppo jakaa muille tai upottaa mille tahansa verkkosivulle. Kuvassa 7 on esimerkki upotetusta CodePen-toteutuksesta. Myös upotetun version koodeja on mahdollista tarkastella ja muokata. (21.) Ajatuksena oli, että interaktiiviset esimerkit voisi laittaa näkyviin lopulliseen elementtikirjastoon CodePenin avulla.



Kuva 7. Verkkosivulle upotettu CodePen-toteutus. Koodeja voi tarkastella vasemman yläkulman valikon kautta. Oikeaa yläkulmaa klikkaamalla toteutusta pääsee muokkaamaan CodePen-sivuston editorinäkymässä.

Elementin varsinaiset koodit saattavat olla monimutkaisempia kuin yksinkertaiset esimerkit. Niiden on tarkoitus sisältää myös dokumentaatiota koodin toiminnasta ja ohjeita sen käyttöön. Nämä koodit päätettiin tallentaa GitHubiin, joka on maailman suurin koodin säilyttämiseen, jakamiseen ja ohjelmointiprojektien ylläpitoon tarkoitettu palvelu. GitHubin avulla useampi henkilö voi samanaikaisesti työskennellä saman projektin ja jopa samassa tiedostossa olevan koodin parissa. Lisäksi se tarjoaa versionhallinnan ja monia muita hyödyllisiä toimintoja ja työkaluja. Palvelussa kehitetään paljon avoimen lähdekoodin projekteja, mutta myös yritykset voivat ottaa sen käyttöönsä. (22; 23.) GitHub oli yrityksessä käytössä jo ennestään, joten se oli luonteva paikka myös elementtien koodille.

Testivaiheen perusteella elementeistä kerättäviin tietoihin ei tullut muutoksia. Kerättävät tiedot tuntuivat hyödyllisiltä, vaikka joitakin niistä oli hankala täydentää. Esimerkiksi ohjetekstin sisältö vaati hiukan miettimistä, mutta varmasti kokemuksen myötä tietää paremmin, minkälainen ohjeistus on hyödyllinen. Työläintä oli muokata koodit CodePen-esimerkkiin ja GitHubiin sopiviksi, mutta sekin muuttuu luultavasti helpommaksi ajan myötä. Kun vanhoja, usein käytettäviä elementtejä alkaa kertyä kirjastoon, uusia tarvitsee lisätä yhä harvemmin. Täysin uudet toteutukset tulee varmasti myös toteutettua pienempinä ja helpommin hallittavina kokonaisuuksina, kun tietää, että elementti tulee lisätä myös elementtikirjastoon.

5.2 Käyttöliittymän suunnittelu ja toteutustavan valinta

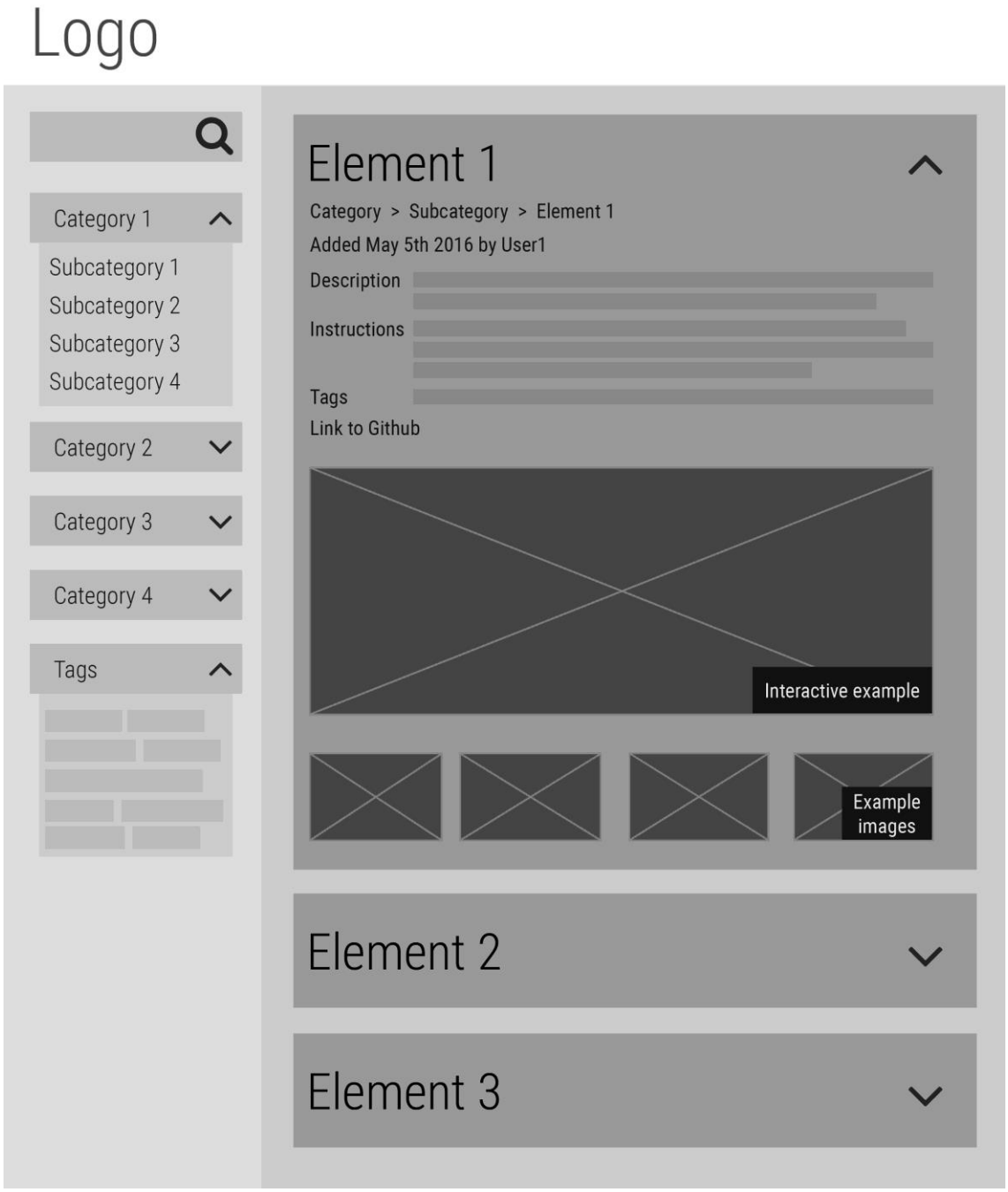
Käyttöliittymän toteutustapa

Elementtikirjaston käyttöliittymä päätettiin toteuttaa sisällönhallintajärjestelmällä. Käytettäväksi ohjelmistoksi valittiin WordPress, joka on yleisin käytössä oleva sisällönhallintajärjestelmä (24). WordPressin mukaan sitä käyttää yli 24 % kaikista maailman verkkosivuista (25). Valintaan vaikutti se, että järjestelmä on itselleni jo ennestään tuttu. WordPressin yleisyyden takia on todennäköistä, että myös tulevat työntekijät osaavat tarvittaessa kehittää ja muokata sillä toteutettua elementtikirjastoa. Lisäksi se on dokumentoitu hyvin ja verkosta löytyy paljon ohjeita WordPress-sivujen kehittämiseen.

WordPressin hyviä ominaisuuksia ovat ohjelmiston mukana tulevien perustoimintojen lisäksi muokattavuus, joustavuus ja laajennettavuus. Se on ilmainen avoimen lähdekoodin ohjelma, jonka voi muokata täysin omiin tarpeisiinsa sopivaksi. Elementtikirjaston kannalta hyödyllisiä perustoimintoja ovat muun muassa valmiit tietokantarakenteet, käyttäjien hallinta, sisällön piilottaminen rekisteröitymättömiltä käyttäjiltä, mahdollisuus kuvien lisäämiseen ja hallinnoimiseen, hakutoiminnot ja muokattavat sisältötyypit, joihin voi liittää erilaisia luokitteluja ja metatietoa. Perustoimintojen lisäksi tarjolla on tuhansia ilmaisia ja maksullisia lisäosia, joiden avulla saa käyttöönsä uusia ominaisuuksia. WordPress-sivun ulkoasua hallitaan teemoilla, joita on myös saatavilla suuri valikoima. Jos käyttää valmista teemaa ja oletustoimintoja, verkkosivun saa toimintaan muutamissa minuuteissa. Jos ulkoasun tekee itse ja haluaa muokata myös toimintoja, aikaa saa kulumaan paljon enemmän. (25; 26.)

Käyttöliittymän suunnittelu

Käyttöliittymän suunnittelu aloitettiin piirtämällä rautalankamalli (wireframe) eli käyttöliittymän ”luuranko”. Piirroksesta käyvät ilmi käyttöliittymän toiminnot, eri sisältöjen tarvitseman tilan määrä sekä sisältöjen sijainti ja tärkeysjärjestys. Rautalankamalli on yksinkertainen suunnitelma, jossa ei vielä oteta kantaa käyttöliittymän ulkoasuun eli esimerkiksi väreihin, fontteihin ja kuviin. (27.) Käyttöliittymästä tehtiin aluksi käsin pari luonnosta, joiden avulla suunnitelmista keskusteltiin yrityksen toimitusjohtajan ja käyttöliittymäsuunnittelijan kanssa. Keskustelun pohjalta valittiin toteutettava versio ja tehtiin siihen tarvittavat muutokset. Kuvassa 8 on tietokoneella tehty versio rautalankamallista.



Kuva 8. Rautalankamalli elementtikirjaston käyttöliittymästä suunnitteluvaiheen alussa.

Kuvan 8 mukaisesti elementtikirjaston käyttöliittymä koostuu yläpalkista, sivupalkista ja pääsisältöalueesta. Yläpalkissa on logo, sivupalkissa haku ja valikko sisällön suodattamiseen, sisältöalueella listataan elementit. Sivupalkissa eri kategoriat ja tagit eli avainsanat on jaoteltu omiin osioihinsa, joiden sisällön saa näkyviin ja pois näkyvistä otsikkoa klikkaamalla. Myöhemmin avainsanoja ja alakategorioita tulee olemaan paljon, jolloin mahdollisuus sisältöjen piilottamiseen säästää tilaa ja tekee valikon käytöstä helpompaa.

Valikosta voi valita esimerkiksi yhden alakategorian, jolloin pääsisältöalueella listataan kaikki kyseiseen kategoriaan kuuluvat elementit. Jos valitsee kategorian ja sen lisäksi avainsanan, näytetään vain ne kategoriaan kuuluvat elementit, joihin valittu avainsana on liitetty.

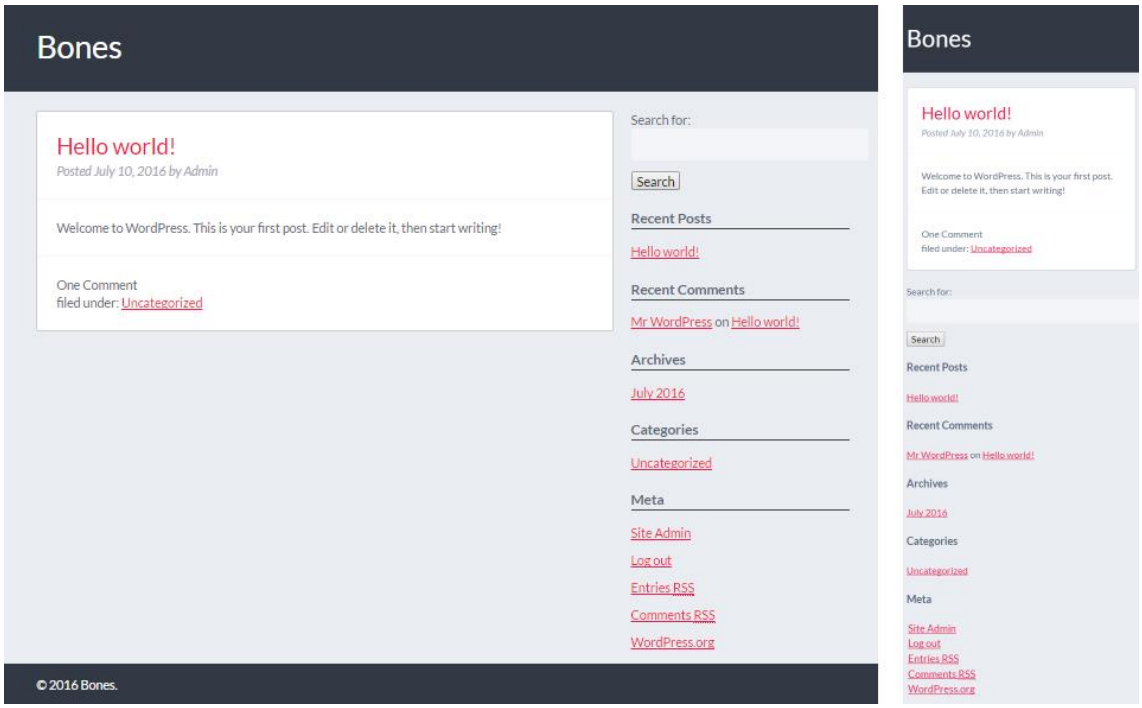
Kategoria- ja avainsanavalikoiden tapaan myös elementtien sisällön saa otsikkoa klikkaamalla näkyviin ja pois näkyvistä. Jokaisesta elementistä näytetään elementin nimi, kategoriat, elementin tekijä, luontipäivämäärä, kuvaus, ohje, avainsanat, linkki elementin koodeihin GitHubissa, interaktiivinen esimerkki ja esimerkkikuvat. Kuvassa 8 interaktiivinen esimerkki ja kuvat on merkitty neliöillä, joiden päällä on rasti. Se on suositeltu tapa kuva-alueiden merkitsemiseen rautalankamallissa (27). Harmaan eri sävyt viittaavat elementtien tärkeysjärjestykseen: mitä tummempi sävy, sitä tärkeämpi elementti (28).

Suunnitteluprosessi etenee eri tavoin tilanteesta ja suunnittelijasta riippuen. Joskus rautalankamallista tehdään toinen, vielä tarkempi versio tai interaktiivinen prototyyppi, minkä jälkeen toteutetaan ulkoasun visuaalinen suunnitelma. Joskus siirrytään suoraan paperille luonnostelusta koodin kirjoittamiseen, eli prototyyppi ja lopulta myös visuaalinen ilme toteutetaan suoraan HTML:n ja CSS:n avulla. (28.) Elementtikirjaston tapauksessa tekninen toteutus aloitettiin rautalankamallin pohjalta. Visuaalinen ilme ei yrityksen sisäisellä verkkosivulla ole ensisijaisen tärkeä asia, joten siihen päätettiin keskittyä tarkemmin myöhemmin.

5.3 Käyttöliittymän toteutus

Teema

Toteutus aloitettiin asentamalla WordPress yrityksen palvelimelle. Sivuston pohjana käytettiin verkkokehittäjille tarkoitettua Bones-teemaa. Kuvassa 9 näkyy, miltä sivu näyttää asennuksen jälkeen, kun teema on aktivoitu. Bones on teemapohja (started theme), jota käytetään oman teeman tekemisen apuna. Teemapohjan käyttö nopeuttaa ja helpottaa sivuston rakentamista paljon verrattuna täysin alusta itse rakennettuun teemaan. Usein valmiita WordPress-teemoja muokataan erillisen lapsiteeman (child theme) avulla, mutta teemapohjia käytettäessä on tarkoitus muokata itse teemaa. Bonesin mukana tulee paljon hyödyllisiä ominaisuuksia ja esimerkkejä, joita voi ottaa käyttöön tai poistaa käytöstä omien tarpeidensa mukaan. Ulkoasultaan se on hyvin yksinkertainen. (29.)



Kuva 9. Bones-teemaa käyttävä WordPress-sivu. Oikealla näkyy sivun ulkoasu mobiilikoossa, vasemmalla suuremmassa koossa eli esimerkiksi tietokoneen näytöllä tarkasteltuna.

Bones-teeman ulkoasu on rakennettu responsiiviseksi mobile-first-mallia noudattaen, eli tyylimäärittelyt asetetaan ensisijaisesti pienille näytöille, kuten mobiilipuhelimille. Responsiivisuuden ansiosta sisältö mukautuu sopivaksi kaikille erikokoisille laitteille ja näytöille. (29; 30.) Kuvassa 9 ulkoasu on esitetty kahdessa eri koossa. Vasemmanpuoleinen kuva on suurelta näytöltä, oikeanpuoleisessa kuvassa sama sisältö on mukautunut pienempään tilaan.

Tyylimäärittelyt tehdään Bonesissa Sass-kielellä. Sass on CSS:n laajennus, joka helpottaa ja nopeuttaa tyylien luomista muun muassa muuttujien ja sisäkkäisten määrittelyjen avulla. Tyylimäärittelyt kirjoitetaan lähes samalla tavalla kuin CSS:ssä, ja lopuksi tiedosto käännetään CSS-muotoon, jotta selain osaa tulkita sitä. (29; 31.)

Elementtien tiedot

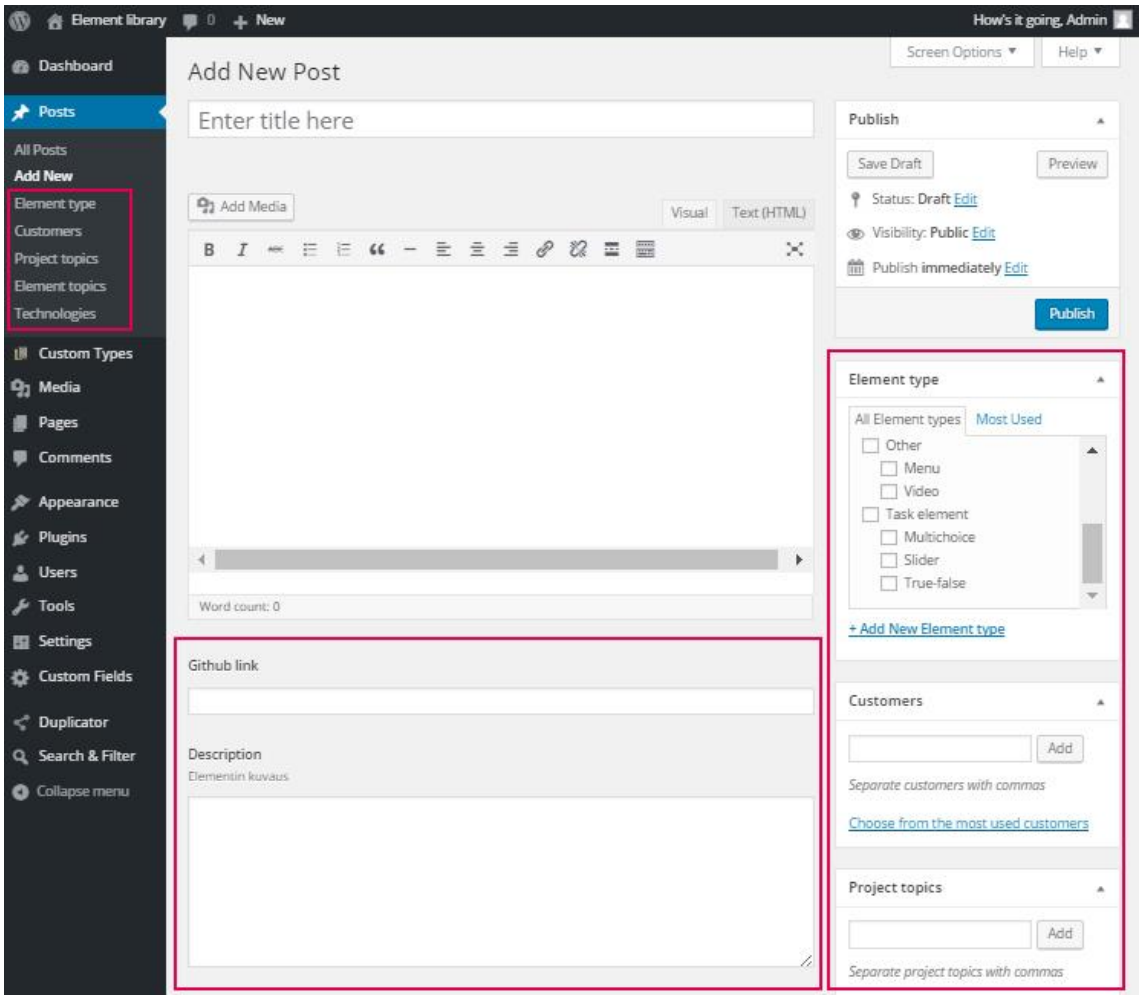
WordPressin sisältöjä kutsutaan nimellä post types. Sisältöjä hallitaan admin-näkymästä. Erityyppisiä sisältöjä ovat post, page, attachment, revision ja navigation menu. Post on yleensä käytössä blogi-sivustoilla: jokainen julkaisu on post, jonka voisi suomentaa vaikka artikkeliksi. Tavalliset verkkosivut on yleensä rakennettu page-tyypin sisällöillä; esimerkiksi yrityksen sivuilla yhteystiedot voi olla yksi page eli sivu. Jos sivuilla

lisäksi julkaistaan uutisia, ne ovat artikkeleita. Artikkeli voi käytännössä olla mikä tahansa sisältö; esimerkiksi verkkokaupassa jokainen tuote on artikkeli. Artikkeleihin voi liittää erilaisia luokitteluja, joita oletuksena ovat kategoriat ja avainsanat. Lisäksi kaikki artikkelit voi helposti listata sivuilla esimerkiksi aikajärjestykseen. (32.) Elementtikirjastossa elementit oli kätevintä toteuttaa artikkeleina.

Tavallisesti WordPressin artikkeleihin voidaan liittää esimerkiksi otsikko, sisältö, kategoria, avainsanat ja featured image eli artikkelin ”pääkuva”. Myös sisältöosiossa voi olla kuvia. Lisäksi järjestelmä tallentaa julkaisuajan ja artikkelin luoneen käyttäjän nimen. (33.) Jotta elementeistä saatiin kerättyä kaikki tarpeelliset tiedot, oletussisältötyyppiä piti muokata. Jokaisen elementin yhteyteen haluttiin kuvaus, ohje ja linkki GitHubiin. Jotta näitä ei kirjoitettaisi sekavassa järjestyksessä sisältökenttään, jokaiselle asialle lisättiin oma tekstikenttä. Kenttien lisääminen minkä tahansa sisältötyypin muokkausnäkymään onnistuu helposti Advanced Custom Fields -lisäosalla (34). Lisäkenttien avulla lisättyjen tietojen sijaintia ja ulkoasua käyttöliittymässä on myös helpompi hallita kuin sisältökenttään kirjoitettuja tietoja. Lisätyt kentät näkyvät kuvassa 10.

WordPressissä sisältöjen luokittelujärjestelmää kutsutaan taxonomy-nimellä. Oletusluokituksia ovat muun muassa category ja tag. Kategoriat ovat hierarkkisia, eli yksi kategoria voi sisältää alakategorioita. Tag eli avainsana ei ole hierarkkinen. Yhteen artikkeliin voi liittyä useita avainsanoja, kun taas kategorioita on yleensä vain yksi. (35.) Elementin kategorian ja avainsanojen listaamiseen oli aluksi tarkoitus käyttää oletusluokitteluja. Avainsanoja haluttiin kuitenkin lisätä useissa eri ryhmissä, joita olivat muun muassa asiakas, projektin aihe ja käytetyt teknologiat. Avainsanojen ryhmittelyominaisuutta testattiin aluksi parin eri lisäosan avulla, mutta ne eivät toimineet halutulla tavalla.

Tarvittavien avainsanryhmien luomisessa päätettiin lopulta hyödyntää mahdollisuutta omien luokittelujen (custom taxonomy) luomiseen. Luokittelut lisättiin teeman functions.php-tiedostoon laitettavan register_taxonomy -funktion avulla. Funktio lisäsi uudet luokittelut automaattisesti myös admin-näkymään. (35.) Kaikki avainsanaryhmät lisättiin omina ei-hierarkkisina luokitteluinaan. Myös kategoria-luokittelu päätettiin korvata uudella hierarkkisella luokittelulla, jonka nimeksi määriteltiin element type. Alkuperäisiä category- ja tag-luokitteluja ei enää tarvittu, joten ne piilotettiin admin-näkymästä Adminimize-lisäosan avulla. Kuvassa 10 on korostettu admin-näkymän artikkelisivun muuttuneita kohtia.



Kuva 10. Elementtikirjaston admin-näkymän artikkelisivu. Muokatut kohdat on merkitty kehyksillä.

Kuvassa 10 vasemmalla olevan kehyksen kohdalla on valikko, johon itse lisätyt luokittelut ovat ilmestyneet. Niiden kautta kategorioita ja avainsanoja voi lisätä, poistaa ja muokata. Artikkelinäkymän oikeasta reunasta määritellään yksittäisen elementin luokittelut. Tavallisesti oikealla olevan kehyksen kohdalla on category- ja tag-laatikot, jotka on nyt poistettu näkyvistä ja korvattu omilla luokitteluilla. Myös näiden laatikoiden kautta voi lisätä uusia kategorioita, eli elementtityyppejä, sekä avainsanoja, eli esimerkiksi asiakkaita tai projektien aiheita. Keskellä olevan kehyksen sisällä näkyy osa itse lisätyistä sisältökentistä.

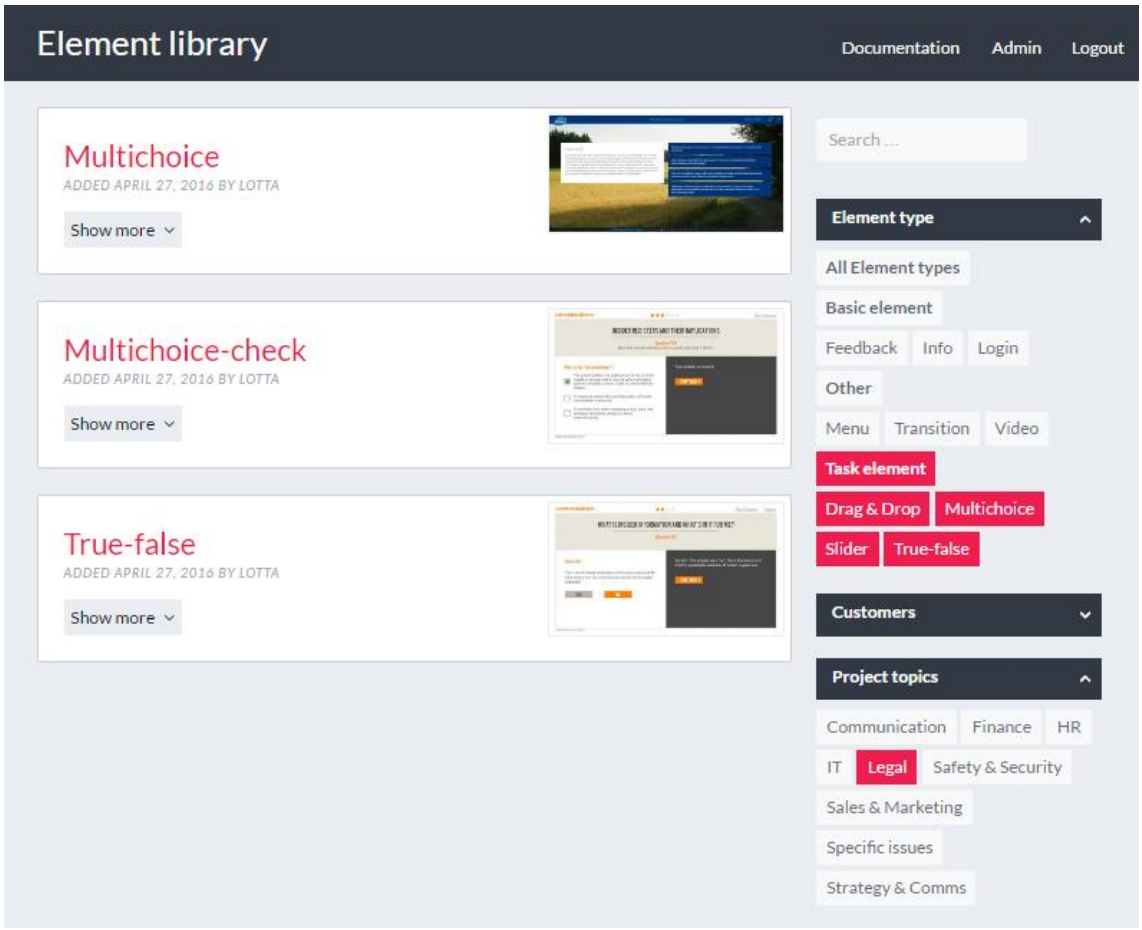
Hakutoiminto ja elementtien selaaminen luokittelujen avulla

Wordpressissä on oletustoimintoja, jotka listaavat kaikki kategoriat ja avainsanat. Yhtä kategorian nimeä tai avainsanaa klikatessa näytetään kaikki siihen kuuluvat artikkelit.

Elementtikirjastoon haluttiin lisäksi mahdollisuus suodattaa sisältöä luokittelujen avulla eli esimerkiksi näyttää elementtejä, jotka kuuluvat tiettyyn kategoriaan ja sen lisäksi sisältävät tietyn avainsanan. Kun WordPress-sivustolle tarvitsee mukautetun toiminnallisuuden, kannattaa ensimmäiseksi selvittää, onko joku jo toteuttanut sen. Järjestelmän yleisyyden vuoksi on hyvin todennäköistä, että joku muukin on joskus kaivannut kyseistä ominaisuutta. Tuhansien ilmaisten lisäosien seasta löytyy usein ratkaisu ainakin yleisimpiin ongelmiin. (36.)

Elementtikirjastoon sopivat haku- ja lajitteluominaisuudet saatiin toteutettua Search and Filter -nimisellä lisäosalla. Lisäosa oli ilmainen, mutta siitä oli tarjolla myös maksullinen versio Search & Filter Pro, joka tarjosi ilmaiseen verrattuna useita lisäominaisuuksia. Yksi ominaisuuksista oli hakutulosten päivittyminen Ajaxin avulla eli ilman sivunlatausta. (37.) Koska Ajax tekee elementtien selaamisesta ja hakemisesta paljon sulavampaa ja nopeampaa, lisäosasta päätettiin ostaa Pro-versio.

Kuvassa 11 on esimerkki elementtien selausnäkymästä, jossa selataan kaikkia task element -kategoriaan kuuluvia elementtejä, joihin on liitetty project topics -ryhmän avainsana legal. Alkuperäisestä suunnitelmasta poiketen elementtien haku- ja suodatustoiminnot ovat oikealla ja elementit listataan vasemmalla. Suunnitelmassa eri kategoriat oli jaettu omien otsikoidensa alle ja avainsanat olivat yhtenä ryhmänä, mutta käyttöliittymän toteutuksessa tehtiin toisin päin. Kategoriat ovat yhden otsikon alla ja avainsana erillisinä ryhminä, sillä avainsanaryhmiä ja avainsanoja tulee olemaan paljon enemmän kuin kategorioita.



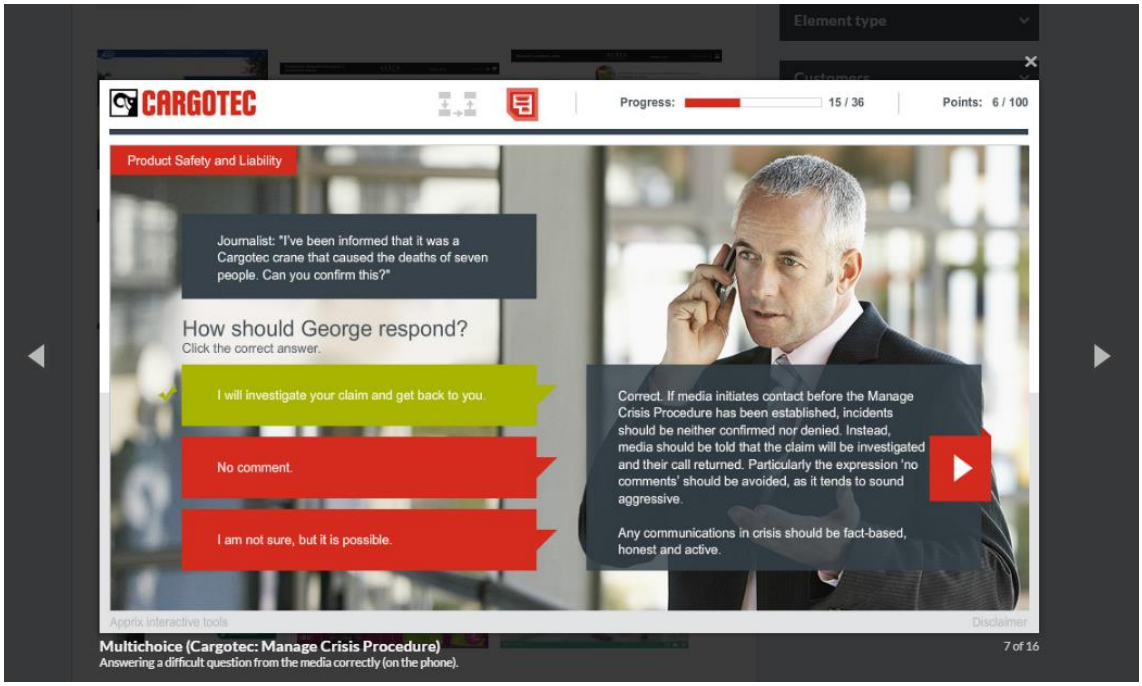
Kuva 11. Elementtien selausnäkymä. Oikealla on sivupalkki, jonka toimintojen avulla voidaan määritellä, mitä elementtejä vasemmalla näytetään.

Kategoria- ja avainsanalistojen lisäksi sivupalkissa on hakukenttä, johon voi kirjoittaa haluamansa hakusanan tai hakusanoja. Hakukenttää voi käyttää myös yhdessä suodatusvalikoiden kanssa. Jos valitsee kategorian, hakukenttään kirjoitettua sisältöä etsitään vain siihen kategoriaan kuuluvista elementeistä. WordPressin oletushaku etsii hakusanoja ainoastaan artikkelien otsikoista ja sisältöteksteistä (38). Jos haluaa haun kattavan myös esimerkiksi artikkelien kategoriat, avainsanat, itse määriteltyihin lisäkenttiin syötetyt tiedot ja artikkelin tekijän nimen, hakua pitää muokata. Elementtikirjastossa haun parantamiseen käytettiin Relevanssi-nimistä lisäosaa. Se sisältää myös useita muita hyödyllisiä toimintoja, mutta elementtikirjaston kannalta haettavan sisällön kattavuus oli oleellisinta. (39.)

Kuvagalleria

WordPress-sivulle on mahdollista lisätä kuvia, joita hallitaan mediakirjaston (media library) kautta (40). Elementtikirjastossa jokaisen artikkelin eli elementin yhteyteen haluttiin oma kuvagalleria, johon olisi helppo lisätä kuvia. Sen lisäksi kuvien selaamisen käyttöliittymässä tulisi toimia sujuvasti. Gallerian toteutusta kokeiltiin ensin Easy Image Gallery -lisäosan avulla. Galleria toimi hyvin muuten, mutta se oli suunniteltu toimimaan ainoastaan yksittäisen artikkelin näkymässä eikä sivulla, jossa listataan useita artikkeleita. Muutkaan lisäosat eivät toimineet halutulla tavalla, joten lopulta kuvagalleria toteutettiin ilman lisäosaa, WordPressin oman galleriatoiminnon avulla. Galleria lisätään artikkelin muokkausnäkymässä sisältökenttään. (41.) Elementtikirjastossa sisältökenttään ei kirjoiteta elementin tietoja, koska tiedoille on lisätty omat tekstikenttänsä. Kuvagallerian lisäksi sisältökenttään tulee vain CodePen-upotuksen koodi.

Käyttöliittymän puolella kuvagalleria luotiin Magnific Popup -nimisen JavaScript-lisäosan avulla. Magnific Popup ei ole WordPress-lisäosa, vaan sitä voi käyttää millä tahansa verkkosivulla. (42.) Kuvassa 12 on esimerkki gallerian ulkonäöstä.



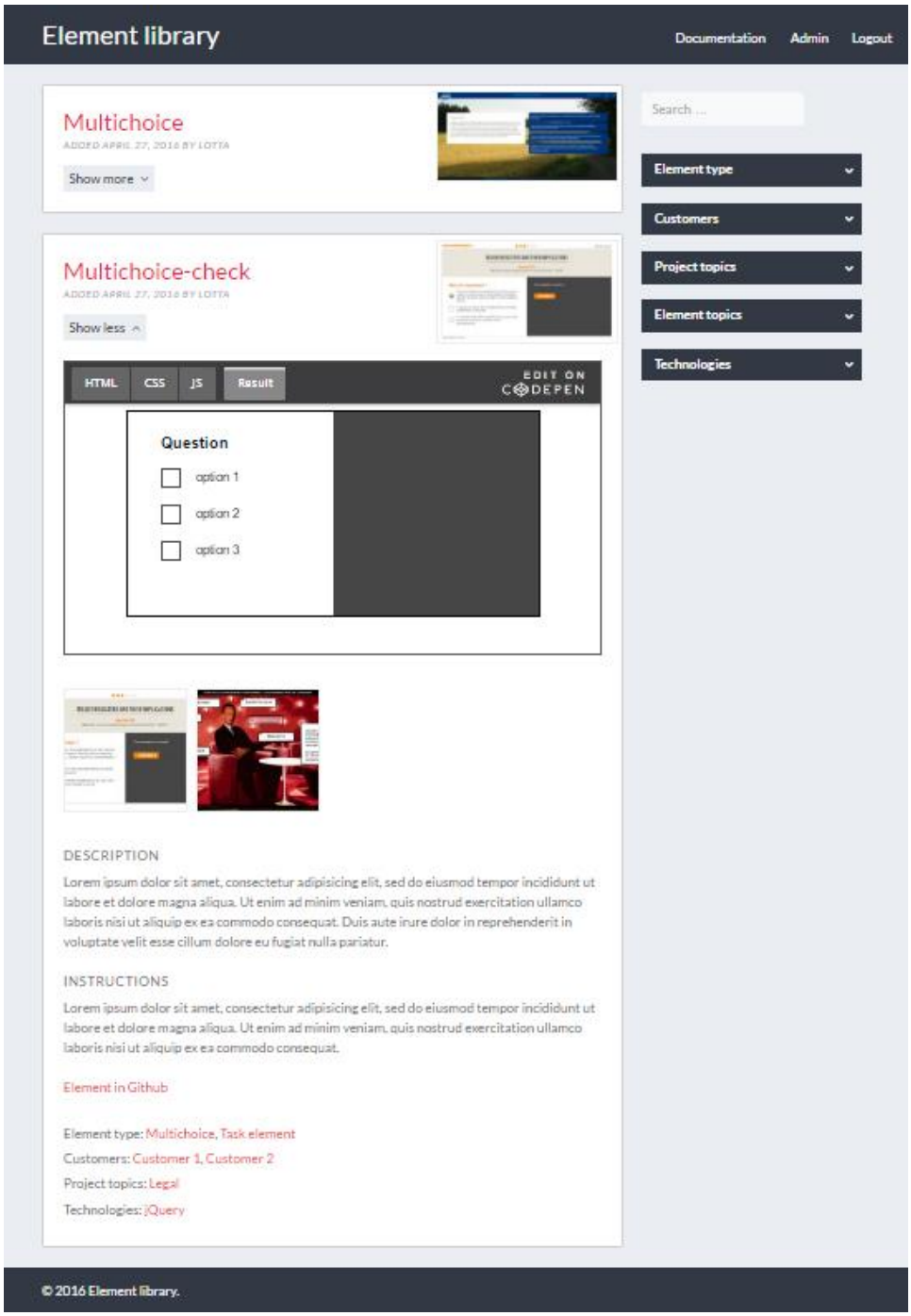
Kuva 12. Esimerkki näkymästä, jossa elementtikirjastossa olevaa kuvaa tarkastellaan Magnific Popup -kuvagallerian avulla.

Magnific Popup avaa kuvan verkkosivun päälle lightbox-nimiseen näkymään (kuva 12). Kuvan voi sulkea oikean yläkulman ruksista tai klikkaamalla mihin tahansa kuvan ulkopuolella näkyvälle taustalle. Reunoissa olevista nuolista voi liikkua edelliseen ja seuraavaan kuvaan. Kuvan alle on lisätty kuvan nimi ja kuvateksti. (42.)

Käyttöliittymän toiminnot

Elementtikirjasto on yrityksen sisäinen verkkosivu, joten ulkopuolisten pääsy sivustolle haluttiin estää. Registered users only -lisäosan avulla sivusto saatiin piilotettua käyttäjiltä, jotka eivät ole kirjautuneet sivulle. Jokaiselle yrityksen työntekijälle tehdään omat tunnukset, joiden avulla sivustolle pääsee kirjautumaan. Elementtikirjaston käyttöliittymän oikeaan yläkulmaan lisättiin linkkejä, joiden avulla voi siirtyä admin-näkymään tai kirjautua ulos (kuva 13). Lisäksi yhdestä linkistä pääsee dokumentaatio-sivulle, jonne kirjoitettiin ohjeita elementtikirjaston käyttöön.

Kirjaston etusivulla listataan kaikki elementit aikajärjestyksessä uusimmasta vanhimpaan. Lista päivittyy dynaamisesti sivupalkin hakua ja luokitteluvalikkoja käyttämällä. Elementeistä näytetään listassa nimi, featured image -toiminnolla lisätty esimerkkikuva sekä elementin lisäyspäivämäärä ja lisääjän nimi. Jokaisen elementin yhteydessä on Show more -nappi, jota klikkaamalla elementistä ilmestyy näkyviin enemmän tietoja. Kuvassa 13 on esimerkki elementistä, josta on avattu näkyviin kaikki tiedot. Tiedot aukeavat samaan näkymään ilman sivun vaihtumista. Jos elementin haluaa avata omalle sivulleen, sen voi tehdä elementin otsikkoa klikkaamalla. Yksittäisen elementin sivulla elementistä näytetään samat tiedot kuin Show more -nappia painamalla. Ulkoasullisesti ainoa ero listanäkymään on se, että näkymästä on poistettu sivupalkki, jolloin sisältöalueesta on saatu leveämpi. Yksittäisen elementin sivu on tarpeellinen esimerkiksi silloin, kun tarvitaan linkki tiettyyn elementtiin.



Kuva 13. Elementtikirjaston käyttöliittymä ja esimerkki yhden elementin tiedoista.

Otsikon ja esimerkkikuvan jälkeen elementin tiedoissa on CodePen-upotus, jonka avulla näkee suoraan, miten elementti toimii. Sen jälkeen tulee kuvagalleria, elementin kuvaus, ohjeet ja linkki elementin koodeihin GitHubiin. Alimpana on listattu elementin kategoriat sekä avainsanat eli asiakkaat, projektien aiheet ja käytetyt teknologiat. Kategoriat ja avainsanat ovat linkkejä, joita klikkaamalla pääsee näkymään, jossa on listattu kaikki kyseiseen ryhmään kuuluvat elementit.

6 Elementtikirjaston käyttöönotto

6.1 Kehitysprojekti ja käyttöönottopalaveri

Käyttöliittymän toteutuksen aikana pidettiin muutamia palavereja, joissa keskusteltiin elementtikirjaston ominaisuuksista yrityksen johdon ja käyttöliittymäsuunnittelijan kanssa. Välillä paikalla oli myös muita teknisiä toteuttajia. Projektin etenemisestä raportoitiin muille työntekijöille yrityksen sisäisellä keskustelukanavalla. Samalla kysyttiin mielipidettä suunnitelmista ja kirjastoon haluttavista ominaisuuksista ja niiden toteutustavoista. Kun käyttöliittymän tärkeimmät ominaisuudet oli rakennettu ja kirjastoon oli lisätty muutamia esimerkkielementtejä, kaikille jaettiin testitunnukset. Työntekijöitä oli kuitenkin melko hankala saada oma-aloitteisesti kommentoimaan suunnitelmia ja toteutusta. Jos halusi palautetta kaikilta, paras tapa oli tehdä kysely, joka kaikkien oli pakko täyttää tiettyyn määräaikaan mennessä, järjestää palaveri, jossa kaikki olivat paikalla, tai kysyä ihmisiltä henkilökohtaisesti.

Kun elementtikirjaston tärkeimmät ominaisuudet oli toteutettu ja käyttöliittymään oltiin tyytyväisiä, järjestettiin käyttöönottopalaveri. Siihen osallistuivat lähes kaikki yrityksen työntekijät. Palaverissa kerrattiin aluksi lyhyesti elementtikirjaston tarkoitus ja tavoitteet, minkä jälkeen esiteltiin käyttöliittymä ja sen ominaisuudet. Lisäksi kerrottiin, miten elementin koodit lisätään GitHubiin, miten CodePen-esimerkki luodaan ja miten elementti ja sen tiedot lisätään kirjastoon. Lopuksi työntekijöille ohjeistettiin, miten kirjastoon luodaan omat tunnukset. Palaverissa myös sovittiin, miten kirjasto otetaan käyttöön. Juuri päättyneet ja päättymässä olevat projektit ja niistä lisättävät elementit listattiin, ja lisäksi sovittiin, kenen vastuulla minkäkin elementin lisääminen on. Käyttäjien avuksi elementtikirjastoon lisättiin dokumentaatio-sivu, jolle kirjoitettiin ohjeet elementtikirjaston käyttämiseen. Ohjeistuksen sisältö on luettavissa liitteessä 3.

Muutokset ja kehitysehdotukset

Kehitysprojektin aikana ja käyttöönoton yhteydessä käydyissä keskusteluissa päätettiin useista pienistä ja suurista elementtikirjastoon tulevista muutoksista ja lisäyksistä. Kehitysvaiheessa kirjastoon päätettiin lisätä avainsanaryhmä project name (projektin nimi), jotta elementtejä voisi selata myös yksittäisen projektin perusteella eikä pelkästään asi-

akkaan nimen tai projektin aiheen perusteella. Ryhmään kuuluvat avainsanat ovat muotoa ”Asiakas: Projektin nimi”. Jo suunnitteluvaiheessa oli myös mietitty mahdollisuutta kerätä selkeiden elementtien lisäksi talteen pienempiä hyödyllisiä koodipätkiä. Käyttöönottopalaverissa päätettiin, että kirjastoon lisätään uusi kategoria code, jolla on kaksi alakategoriaa: base ja snippet. Snippet-kategoriaan lisätään pieniä koodipätkiä, base-kategoriaan isompia projektien pohjatiedostoja. Code-kategoriaan lisättävien elementtien luonteesta riippuen CodePen-esimerkkiä ja kuvagalleriaa ei välttämättä aina tarvita.

Käyttöönottopalaverissa päätettiin myös lisätä uusi, vielä nimeämätön avainsanaryhmä, johon kuuluvia avainsanoja olisivat muun muassa signal gathering (signaalien keräys) ja rich engagement (osallistava elementti). Ryhmään kuuluvat avainsanat kertovat, minkätyyppiseen tarkoitukseen elementtiä voidaan käyttää. Lisäksi palaverissa keskusteltiin monista erilaisista kirjastoon lisättävistä elementeistä ja niille sopivista kategorioista.

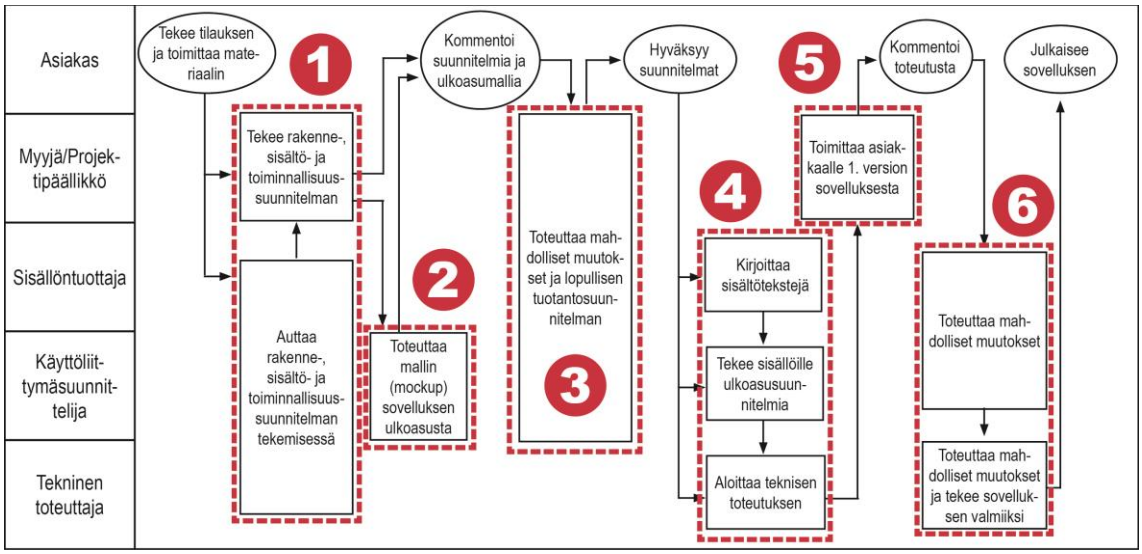
Käyttöliittymään myöhemmin toteutettaviin ominaisuuksiin kuuluu muun muassa kuvagallerianäkymä, jossa voi selata ainoastaan elementtien kuvia. Ominaisuus on hyödyllinen suunnittelijoille, jotka etsivät vanhoista toteutuksista inspiraatiota uusiin projekteihin. Myös käyttöliittymän ulkoasu todennäköisesti muuttuu sitten, kun yrityksen käyttöliittymäsuunnittelijalla on aikaa miettiä ulkoasun yksityiskohtia. Lisäksi elementtien esimerkkitoteutusten kuvien lisäämiseen tuli käyttöönoton jälkeen hyvä ehdotus eräältä tekniseltä toteuttajalta. Hän kokeili lisätä kuvan GIF-animaationa ja totesi, että ne toimivat kuvagalleriassa. Animaatio pelkän kuvan sijaan on hyödyllinen, jos elementti sisältää paljon interaktiivisuutta, efektejä ja animaatioita.

Kehitysvaiheen aikana yrityksen johdon kanssa keskustellessa esiin nousi ajatus tallentaa elementtien lisäksi myös projektien tiedot kaikkien työntekijöiden saataville. Projekteista kerrottaisiin asiakkaan nimi, projektin nimi ja aihe, projektissa mukana olleiden työntekijöiden nimet, julkaisupaikka, toteutusajankohta, linkki testiversioon ja mahdollisesti lista projektissa käytetyistä elementeistä ja muita oleellisia projektikohtaisia tietoja. Tiedoista olisi paljon hyötyä, jos pitää esimerkiksi tehdä tietylle asiakkaalle jotakin samalla tavalla kuin aiemmassa samalle asiakkaalle tehdyssä projektissa, jossa ei itse ole ollut mukana. Linkki testiversioon olisi myös erittäin hyödyllinen, jos haluaa tarkastella tarkemmin jonkin elementin toteutusta ja toimintaa tietyssä projektissa. Projektiarkiston voisi mahdollisesti toteuttaa elementtikirjaston yhteyteen.

Käyttöönottopalaverissa yrityksen käyttöliittymäsuunnittelija ehdotti, että elementtikirjastoon voisi tallentaa myös tietoja asiakaskohtaisista projektivaatimuksista. Ne päätettiin lisätä dokumentaatiosivulle. Samalla mietittiin, että dokumentaatioon voisi liittää muitakin hyödyllisiä ohjeita ja tietoja yrityksen eri työkalujen käyttöön liittyen. Dokumentaation ja tulevan projektiarkiston myötä elementtikirjasto alkaa laajentua alkuperäistä tarkoitustaan suuremmaksi. Jos ominaisuuksia tulee yhä enemmän, sivustosta tulee ikään kuin yrityksen intranet, jonka yksi osa elementtikirjasto on.

6.2 Vaikutus tuotantoprosessiin

Elementtikirjasto vaikuttaa paljon yrityksen toimintatapoihin ja prosesseihin. Suurimmat vaikutukset näkyvät aliprosesseissa, kuten esimerkiksi projektin teknisessä toteutuksessa. Tuotantoprosessi on ylemmän tason prosessi, joten itse prosessikaavioon ei tullut elementtikirjaston myötä muutoksia. Elementtikirjasto kuitenkin vaikuttaa kaikkiin tuotantoprosessin vaiheisiin. Vaikutukset eri vaiheisiin käsitellään seuraavaksi kuvan 14 numeroinnin mukaisesti. Suurempi kuva prosessikaaviosta on liitteessä 1.



Kuva 14. Tuotantoprosessin vaiheet numeroituina.

Tuotantoprosessin vaihe 1 (kuva 14) kattaa alustavan rakenne-, sisältö- ja toiminnallisuussuunnitelman laatimisen asiakkaan toimittamien materiaalien ja asiakkaan kanssa käytyjen keskustelujen pohjalta. Vaiheessa suunnitellaan, mitä elementtejä projektissa käytetään. Elementtikirjasto toimii suunnittelun apuna: kirjastosta on helppo katsoa, minkälaisia elementtejä on käytetty aiemmin saman asiakkaan, saman toimialan yrityksen

tai saman aihepiirin projekteissa. Kun esimerkkejä on nähtävillä kaikista erityyppisistä elementeistä, sisällölle voidaan valita sopivin mahdollinen esitystapa.

Kirjaston kautta voidaan myös esitellä erilaisia elementtivaihtoehtoja asiakkaalle. Kirjaston interaktiivista esimerkeistä on poistettu asiakaskohtaiset sisällöt ja tyylit, jolloin elementtiä voi esitellä, vaikka se olisi alun perin tehty salassa pidettävään projektiin. Yksittäisen elementin esittely kirjaston kautta on muutenkin helpompaa kuin oikean projektin käyttäminen esimerkkinä. Käytössä olevat projektit esimerkiksi keräävät käyttäjistä tilastotietoja, joita ei haluta sotkea ylimääräisillä testikirjautumisilla. Lisäksi projektit ovat kaikki omien linkkiensä ja tunnuksiensa takana, kun taas elementtikirjastossa kaikki on selattavissa samassa paikassa.

Tuotantoprosessikaavioon merkityssä kohdassa 2 (kuva 14) toteutetaan ensimmäisen vaiheen suunnitelmien perusteella mockup eli malli sovelluksen ulkoasusta. Käyttöliittymäsuunnittelija voi hakea suunnitteluun inspiraatiota aiemmin toteutetuista elementeistä. Kirjasto toimii myös projektipäällikön ja käyttöliittymäsuunnittelijan kommunikaation apuna. Kumpikin voi kirjastossa olevien esimerkkien avulla selittää toiselle, millaista elementtiä, toiminnallisuutta, animaatiota tai ulkoasua on ajateltu, jolloin väärinymmärryksien todennäköisyys pienenee.

Kolmas vaihe tuotantoprosessikaaviossa on asiakkaan kommenttien antamisen jälkeen tehtävä lopullinen tuotantosuunnitelma. Suunnitelmaan merkitään, mitä elementtejä missäkin kohdassa käytetään. Jos elementti on jo aiemmin toteutettu, voidaan käyttää elementtikirjastossa olevaa nimeä tai jopa lisätä linkki elementtikirjastossa olevaan toteutukseen. Näin kaikille suunnitelmaa lukeville on selvää, mitä elementtiä tarkoitetaan. Tässä vaiheessa voidaan myös sopia, mitä uusia elementtejä projektista lisätään kirjastoon myöhemmin. Tieto olemassa olevista elementeistä ja uusista kirjastoon lisättävistä elementeistä myös helpottaa projektin toteutukseen tarvittavan työmäärän arviointia.

Neljännessä kohdassa aloitetaan sovelluksen toteutusvaihe. Elementtikirjastosta on eniten hyötyä teknisille toteuttajille, jotka ideaalitilanteessa saavat sieltä valmiina elementtien koodit. Aika säästyy, kun kaikki vanhat toteutukset ovat helposti saatavilla eikä niitä tarvitse pyytää muilta tai kaivaa omista vanhoista projekteista. Pahimmassa tapauksessa tekninen toteuttaja on voinut tehdä jo olemassa olevan elementin alusta asti uudelleen, kun tietoa vanhasta toteutuksesta ei ole löytynyt. Työn laatu paranee, kun käy-

tetään jo aiemmin toimiviksi toteutettuja elementtejä ja kehitetään niitä paremmiksi. Lisäksi elementtikirjastosta löytyy ohjeita, jotka helpottavat jonkun muun kirjottaman koodin ymmärtämistä, käyttämistä ja muokkaamista.

Toteutusvaiheessa projektia tekevät myös sisällöntuottaja ja käyttöliittymäsuunnittelija. Sisällöntuottajan on helpompi miettiä elementteihin tarvittavat tekstit elementtikirjaston avulla. Kirjaston esimerkeistä näkee, miten elementti toimii ja millaisia ohjeita ja muita tekstejä sen yhteydessä yleensä käytetään. Myös käyttöliittymäsuunnittelija voi käyttää ulkoasusuunnitelmien ideoinnissa apuna vanhoja toteutuksia. Tässäkin vaiheessa elementtikirjasto parantaa kommunikaatiota: esimerkiksi käyttöliittymäsuunnittelijan on helppo näyttää teknisille toteuttajalle, miten jonkin elementin on tarkoitus toimia, jos se tai jokin samantapainen toteutus löytyy kirjastosta.

Viidennessä kohdassa sovellus lähetetään asiakkaalle kommentoitavaksi. Suurin osa sovelluksesta on tässä vaiheessa jo valmiina, joten projektiin toteutettujen uusien elementtien lisääminen kirjastoon voidaan aloittaa. Samalla lisätään kuvia ja avainsanoja elementteihin, jotka ovat jo kirjastossa ja päivitetään elementtejä, joiden koodeihin on tullut parannuksia. Elementtien lisäämistä ei jätetä projektin loppuun, jotta se ei jää tekemättä uusien projektien alkaessa. Jos asiakkaan kommenttien myötä elementteihin tulee oleellisia muutoksia, ne voidaan päivittää kuudennen vaiheen aikana eli samalla, kun viimeistellään sovellus julkaisuvalmiiksi.

6.3 Palaute

Noin kuukausi käyttöönoton jälkeen työntekijöitä pyydettiin antamaan palautetta elementtikirjastosta vastaamalla verkossa olevaan kyselyyn. Tässä vaiheessa kirjastoon oli saatu lisättyä muutamia uusia elementtejä. Palaute oli vapaamuotoinen, mutta kirjoittamisen avuksi annettiin muutamia kysymyksiä, joita sai halutessaan pohtia. Kysymyksissä kysyttiin mielipidettä elementtikirjaston toteutustavasta, käyttöliittymästä, käyttöönottopalaverista ja kirjaston käyttökokemuksista. Lisäksi kysyttiin kirjaston käytön, toimivuuden ja hyödyllisyyden haasteista ja esteistä.

Vastausten perusteella työntekijät olivat tyytyväisiä elementtikirjastoon ja uskoivat siitä olevan hyötyä työnteossa. Toteutustapaa pidettiin toimivana ja käyttöliittymää sopivan yksinkertaisena. Myös mahdollisuutta uusien toiminnallisuuksien lisäämiseen pidettiin

tärkeänä. Elementtien selaaminen todettiin käteväksi etenkin suodatusominaisuuksien, interaktiivisen esimerkin ja animoitujen esimerkkikuvien ansiosta, joiden avulla elementtien toiminnoista saa nopeasti selkeän kuvan. Elementtien selaamiseen toivottiin lisäksi vaihtoehtoista, vielä visuaalisempaa selaustapaa, jonka toteutusta oli jo aiemmin suunniteltu. Se päätettiin toteuttaa heti kun mahdollista.

Käyttöönottopalaveria ja kirjaston käyttöön laadittua ohjeistusta pidettiin hyödyllisinä. Osan mielestä palaverissa tuli sopivasti tietoa, jonkun mielestä asiat käytiin läpi liiankin yksityiskohtaisesti. Ne työntekijät, jotka olivat jo ehtineet lisätä elementtejä kirjastoon, pitivät prosessia helppona, mutta työläänä. Suurin työ on irrottaa elementtien koodi projektista, karsia ylimääräiset asiat pois ja muokata koodi mahdollisimman helposti uudelleenkäytettävään muotoon. Myös dokumentaation kirjoittaminen vie aikaa. Selkeiden ohjeiden todettiin kuitenkin helpottavan elementtien lisäämistä. Standardoitua lisäämisprosessia pidettiin hyvänä ominaisuutena, sillä sen ansioista jokainen elementti lisätään kirjastoon samassa muodossa, jolloin kaikki elementin tiedot löytyvät aina samasta paikasta. Työntekijät totesivat ohjeiden noudattamisen ja elementtien tarkan kategorisoinnin olevan tärkeää elementtien löytymisen ja kirjaston hyödyllisyyden kannalta.

Yhtenä haasteena elementtien tallentamisessa mainittiin teknologian nopea kehitys. Elementtejä tulee ylläpitää ja kehittää, jotta ne eivät vanhene. Eri teknologioita käytettäessä voi aiheutua myös yhteensopivuusongelmia. Suurimpana haasteena kirjaston käytössä nähtiin elementtien luomiseen kuluva aika. Jos elementit halutaan toteuttaa hyvin, tarvitaan aikaa ja motivaatiota, joiden löytäminen voi projektikiireiden keskellä olla vaikeaa. Tarkoituksena kuitenkin on, että kulutettu aika saadaan takaisin elementtien uudelleenkäytön helpottuessa. Palautteissa pohdittiin, että kirjaston käyttö tulee saada osaksi työntekijöiden rutiineja ja yrityksen toimintakulttuuria, jotta sen hyödyt saadaan konkretisoitumaan. Kokemuksen myötä nähdään, miten hyvin kirjasto toimii käytännössä.

7 Yhteenveto

Insinöörityön tavoitteena oli parantaa tilaajayrityksen tiedonhallintaa ja tuotantoprosessia kehittämällä elementtikirjasto, jossa eri projekteihin tehtyjen elementtien toteutukset olisivat helposti selattavissa ja uudelleenkäytettävissä. Projektissa kartoitettiin toteutettavan järjestelmän vaatimukset, valittiin järjestelmän toteutustapa ja suunniteltiin sen käyttöliittymä. Kirjasto toteutettiin hyödyntäen GitHubia ja sisällönhallintajärjestelmä

WordPressiä. Toteutus onnistui hyvin: lopputuloksena oli työntekijöiden palautteen perusteella selkeä ja toimiva järjestelmä. WordPressin valmiina tarjoamat toiminnallisuudet ja mahdollisuus niiden muokkaamiseen tekivät kirjaston kehittämisestä helppoa. Sisällönhallintajärjestelmän joustavuuden ansiosta kirjaston ominaisuuksia on helppo muuttaa ja lisätä tarvittaessa. Järjestelmän muokattavuus on tärkeä ominaisuus, sillä jo projektin aikana tuli ilmi erilaisia lisätoiveita, joiden perusteella sivustoa tulevaisuudessa laajennetaan ja kehitetään.

Elementtikirjaston hyödyt eivät konkretisoidu heti. Ennen kuin kirjastoa voidaan kunnolla käyttää, sinne tulee lisätä elementtejä, mikä vaatii aikaa ja työntekijöiden aktiivisuutta. Uusien elementtien luomisen lisäksi vanhoja elementtejä tulee kehittää ja päivittää. Jos elementtikirjastosta saadaan rakennettua hyödyllinen kokonaisuus, se toimii hyvänä tukena tuotantoprosessille. Elementtien tehokkaamman uudelleenkäytön myötä prosessia saadaan nopeutettua ja sujuvoitettua. Lisäksi elementtien tallentaminen, dokumentointi ja tiedon jakaminen kehittävät yrityksen tiedonhallintaa.

Kirjaston käyttöönoton yhteydessä huomattiin tarkkaan määritellyn elementtien lisäystavan ja kattavan ohjeistuksen tärkeys. Elementtien yhtenäinen nimeämiskäytäntö ja johdonmukainen kategorisointi varmistavat, että kirjastossa olevia tietoja on helppo etsiä ja hyödyntää. Hyvä ohjeistus puolestaan helpottaa elementin lisääjän työtä. Kun prosessi on mahdollisimman tarkkaan määritelty, elementtien luominen on selkeää ja suoraviivaista. Käyttöönoton jälkeen ohjeistusta päivitettiin tarvittaessa ja muun muassa nimeämisen ja elementtien ohjeistuksien kirjoittamisen käytäntöjä tarkennettiin.

Työntekijöiltä kerätty palaute osoitti, että toteutukseen oltiin tyytyväisiä. Elementtikirjastolla todettiin olevan paljon potentiaalia, joka saadaan hyödynnettyä, jos järjestelmää käytetään ja kehitetään aktiivisesti.

Lähteet

1 Lindén, Jukka. 2015. Tiedonhallinta & yrityksen menestys. Lempäälä: Netera Consulting.

2 Grönroos, Mauri. 2003. Mahdollisuuksien aika – kohti virtuaalista organisaatiota. Tampere: Transatlanta.

3 Haasio, Ari & Piukkula, Juha (toim.). 2002. Tietoverkot ja kirjastot. Helsinki: BTJ Kirjastopalvelu.

4 Yu, Liyang. 2007. Introduction to the Semantic Web and Semantic Web Services. Boca Raton: Taylor & Francis Group.

5 About Schema.org. Verkkodokumentti. Schema.org. <http://schema.org/docs/about.html/>. Luettu 14.8.2016.

6 Shewan, Dan. 2015. How to Use Schema Markup for SEO: Making Your Site Easier to Find for Stupid Machines. Verkkodokumentti. WordStream. <http://www.wordstream.com/blog/ws/2014/03/20/schema-seo>. Last updated: Oct 8, 2015. Luettu 14.8.2016.

7 Lecklin, Olli. 2006. Laatu yrityksen menestystekijänä. Helsinki: Talentum Media.

8 CMS. Verkkodokumentti. Sharpened Productions. <http://techterms.com/definition/cms>. Updated: March 28, 2013. Luettu 9.7.2016.

9 Rüping, Andreas. 2009. Where Code and Content Meet – Design Patterns for Web Content Management and Delivery, Personalisation and User Participation. United Kingdom: John Wiley & Sons.

10 Lovinger, Rachel. 2012. Content Modelling: A Master Skill. Verkkodokumentti. A List Apart. <http://alistapart.com/article/content-modelling-a-master-skill>. 24.4.2012. Luettu 4.8.2016.

11 Perkins, Mark. 2016. On Building Component Libraries. Verkkodokumentti. Clearleft. <http://clearleft.com/thinks/382>. 28.4.2016. Luettu 31.7.2016.

12 Curtis, Nathan. 2010. So You Wanna Build a Library, Eh? Verkkodokumentti. Boxes and Arrows. <http://boxesandarrows.com/so-you-wanna-build-a-library-eh/>. 13.9.2010. Luettu 31.7.2016.

13 Material Design for Android. Verkkodokumentti. Google. <https://developer.android.com/design/material/index.html>. Luettu 1.8.2016.

14 Haikala, Ilkka & Mikkonen, Tommi. 2011. Ohjelmistotuotannon käytännöt. Helsinki: Talentum Media.

15 Laamanen, Kai. 2001. Johda liiketoimintaa prosessien verkkona – ideasta käytäntöön. Helsinki: Suomen Laatukeskus Koulutuspalvelut.

16 Martola, Ulla & Santola, Riku. 1997. Liiketoimintaprosessit – BRP-muutoksen johtaminen. Helsinki: WSOY.

17 Hannus, Jouko. 1994. Prosessijohtaminen: ydinprosessien uudistaminen ja yrityksen suorituskyky. Espoo: HM&V Research.

18 Haverila, Matti J,: Uusi-Rauva, Erkki; Kouri, Ilkka; Miettinen, Asko. 2009. Teollisuustalous. Tampere: Infacts.

19 Pedersen, JT. 2014. How Your Three Competitive Levers Define You. Verkkodokumentti. Ignite, LLC & JT Pedersen. <http://www.jtpedersen.net/2014/01/03/how-your-three-competitive-levers-define-you/>. 3.1.2014. Luettu 25.8.2016.

20 Kiiskinen, Satu; Linkoaho, Anssi & Santala, Riku. 2002. Prosessien johtaminen ja ulkoistaminen. Helsinki: WSOY.

21 CodePen is a place to write and share front-end code. Verkkodokumentti. CodePen. <https://codepen.io/hello/>. Luettu 4.7.2016.

22 GitHub. Verkkodokumentti. CrunchBase. <https://www.crunchbase.com/organization/github#/>. Last updated June 8, 2016. Luettu 3.7.2016.

23 GitHub. Verkkodokumentti. GitHub, Inc. 2016. <https://github.com>. Luettu 3.7.2016.

24 CMS Usage Statistics. Verkkodokumentti. BuiltWith® Pty Ltd. <http://trends.builtwith.com/cms>. Luettu 9.7.2016.

25 Features. Verkkodokumentti. WordPress. <https://wordpress.org/about/features>. Luettu 9.7.2016.

26 WordPress Features. Verkkodokumentti. WordPress. <https://codex.wordpress.org/WordPress_Features>. Luettu 9.7.2016.

27 Wireframing. Verkkodokumentti. U.S. Department of Health & Human Services. <https://www.usability.gov/how-to-and-tools/methods/wireframing.html>. Luettu 9.7.2016.

28 Lim, Winnie. 2012. A Beginner's Guide to Wireframing. Verkkodokumentti. Envato Pty Ltd. <http://webdesign.tutsplus.com/articles/a-beginners-guide-to-wireframing--webdesign-7399>. 18.6.2012. Luettu 10.7.2016.

29 Machado, Eddie. Sophisticated WordPress Development. Verkkodokumentti. Themble. <http://themble.com/bones/>. Luettu 10.7.2016.

30 Gremillion, Ben. A Hands-On Guide to Mobile-First Responsive Design. Verkkodokumentti. UXPin. <https://www.uxpin.com/studio/blog/a-hands-on-guide-to-mobile-first-design/>. Luettu 10.7.2016.

31 Coron, Tammy. 2015. What is Sass? Guide to CSS with superpowers. Verkkodokumentti. Future plc. <http://www.creativebloq.com/web-design/what-is-sass-111517618>. 5.11.2015. Luettu 10.7.2016.

32 Post Types. Verkkodokumentti. WordPress. <https://codex.wordpress.org/Post_Types>. Luettu 10.7.2016.

33 Posts. Verkkodokumentti. WordPress. <https://codex.wordpress.org/Posts>. Luettu 10.7.2016.

34 Elliot Condon. Edit smarter with Advanced Custom Fields for WordPress Developers. Verkkodokumentti. <https://www.advancedcustomfields.com/>. Luettu 10.7.2016.

35 Taxonomies. Verkkodokumentti. WordPress. <https://codex.wordpress.org/Taxonomies>. Luettu 10.7.2016.

36 Plugins. Verkkodokumentti. WordPress. <https://codex.wordpress.org/Plugins>. Luettu 13.7.2016.

37 Search & Filter Pro – The Ultimate WordPress Filter Plugin. Verkkodokumentti. Designs & Code. <http://www.designsandcode.com/wordpress-plugins/search-filter-pro/>. Luettu 13.7.2016.

38 Knowles, Chris. 2014. Build Your Own Custom WordPress Search. Verkkodokumentti. Incsub. <https://premium.wpmudev.org/blog/build-your-own-custom-wordpress-search/>. 13.3.2014. Luettu 15.7.2016.

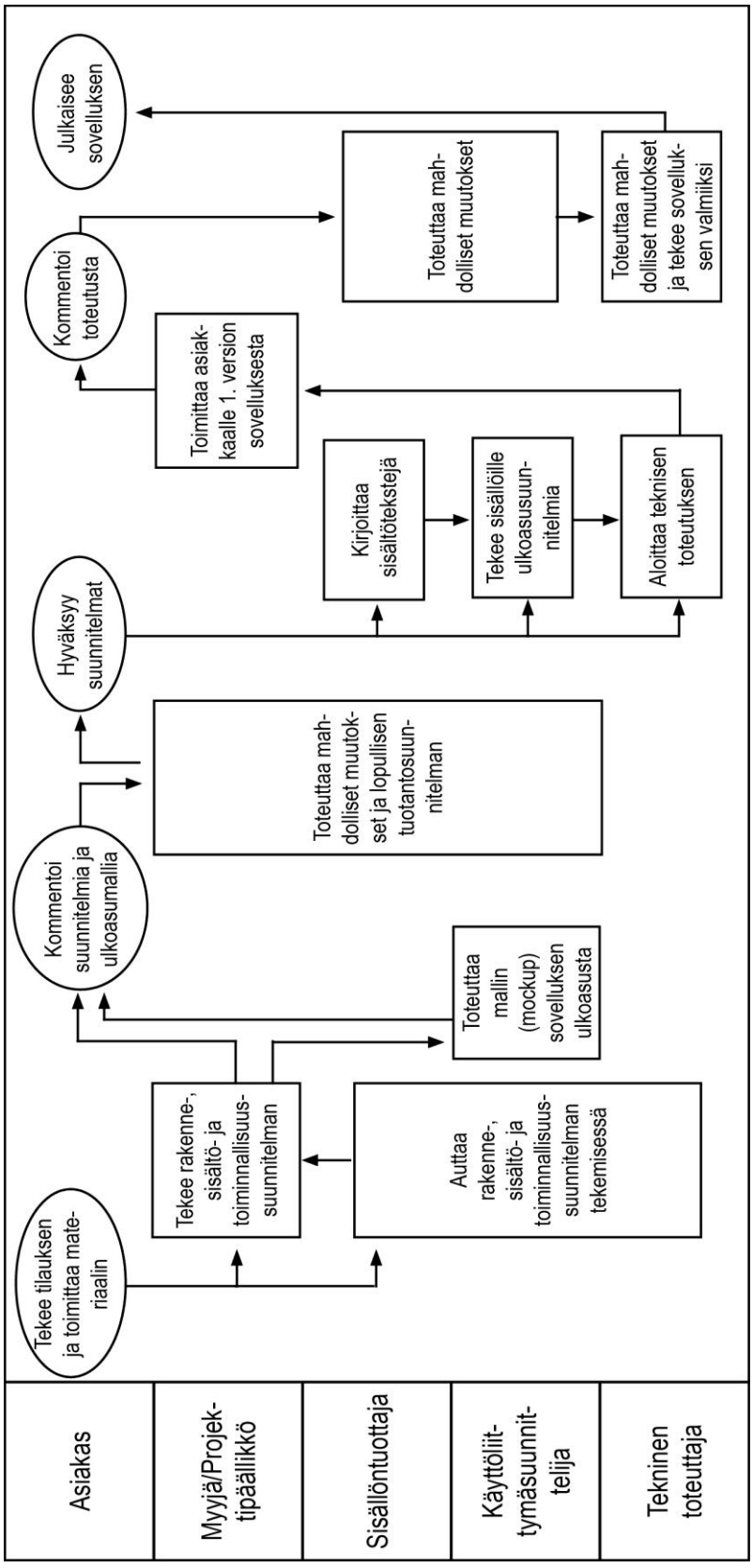
39 Saari, Mikko. Relevanssi - A Better Search. Verkkodokumentti. WordPress. <https://wordpress.org/plugins/relevanssi/>. Luettu 15.7.2016.

40 Media Library Screen. Verkkodokumentti. WordPress. <https://codex.wordpress.org/Media_Library_Screen>. Luettu 15.7.2016.

41 The WordPress Gallery. Verkkodokumentti. WordPress. <https://codex.word-press.org/The_WordPress_Gallery>. Luettu 17.7.2016.

42 Magnific Popup. Verkkodokumentti. Dmitry Semenov. <http://dimsemenov.com/plugins/magnific-popup/>. Luettu 17.7.2016.

Yrityksen tuotantoprosessikaavio



Kysely yrityksen työntekijöille

Elementtikirjastostosta on tarkoitus tulla verkkosivu, joka helpottaa, sujuvoittaa ja nopeuttaa projektien toteuttamista. Erilaisia elementtejä (esim. monivalintatehtävä) voi selata sivulla eri kategorioihin (esim. tehtävätyypit) jaoteltuina. Elementeistä on sivulla esimerkki ja lisäksi mahdollisesti linkkejä tai mainінta siitä, missä projektissa elementtiä on käytetty.

Tavoitteena on kehittää projektien tuotantoprosessia niin, että projektin alussa tekninen toteuttaja voi käydä katsomassa, mitä elementtejä voisi hyödyntää ja ladata ne käyttöönsä. Projektin lopussa katsotaan, onko syntynyt uusia elementtejä, ja lisätään ne kirjastoon.

Sivu on kuitenkin tarkoitettu myös muiden kuin teknisten toteuttajien käyttöön. Sen avulla esimerkiksi projektipäälliköt näkevät, millaisia elementtejä on toteutettu niissä projekteissa, missä ei itse ole ollut mukana. Kirjastossa olevia elementtejä voi esitellä asiakkaille ja käyttöliittymäsuunnittelijalle voi ohjeistaa, että tähän kohtaan tulisi kirjastosta löytyvä elementti X. Kirjasto myös helpottaisi uuden työntekijän perehdytystä, kun kaikki löytyy yhdestä paikasta dokumentoituna.

Nimi

Työnimike/työnkuva

Miten voisit hyödyntää elementtikirjastoa työssäsi?

Millaista hyötyä uskot elementtikirjastosta olevan muille työntekijöille?

Tuleeko mieleesi jotain tilanteita, joissa elementtikirjasto olisi voinut auttaa tekemään jonkin asian nopeammin ja paremmin, jos sellainen olisi ollut käytössä?

Pitäisikö sivun olla englanniksi, suomeksi vai sekä että?

Mitä elementtejä kirjastosta tulisi löytyä? (Esim. tehtävätyypit eli monivalinta, nelikenttä ym., login, palautesivu…)

Mitä kaikkea tietoa olisi hyvä liittää jokaiseen elementtiin? (Esim. nimi, kategoria, elementin luonut henkilö, kuvaus, ohje, julkaisupäivämäärä, päivitykset ja lista projekteista, joissa elementtiä on käytetty...)

Millaisia hakutapoja haluaisit käyttää ja millä tavoin elementtejä tulisi pystyä selaamaan? (Aluksi elementtejä on vai muutama, mutta hiljalleen niiden määrä kasvaa.)

Miten määritellään, mitä kaikkea kirjastoon lisätään, onko esimerkiksi joku muokattu elementti riittävän erilainen kuin alkuperäinen elementti?

Miten varmistetaan, että projekteissa syntyneet uudet elementit varmasti lisätään kirjastoon? Voisiko työnantaja jotenkin kannustaa siihen ja varmistaa, että elementtien lisäämiseen jää aikaa projektien lomassa?

Mitä muita ominaisuuksia toivoisit elementtikirjastossa olevan?

Muuta kommentoitavaa tai kysyttävää?

Ohjeet elementtikirjaston käyttöön

Tekninen toteuttaja lisää projektin elementit kirjastoon projektien loppuvaiheessa, esimerkiksi silloin, kun odotetaan kommentteja asiakkaalta. Jos jo olemassa olevaa elementtiä käytetään toisessa projektissa, riittää uuden kuvan ja tagien lisäys. Kaikista ei välttämättä lisätä edes kuvaa.

1 Elementin lisääminen GitHubiin

Kategorisointi ja nimeäminen

Elementit tulee aluksi nimetä ja kategorisoida. Yläkategorioita ovat basic (esimerkiksi feedback, kirjautuminen ja muut lähes joka koulutuksessa toistuvat elementit), task (erilaiset tehtävät, esim. monivalinta, slider, nelikenttä), code (hyödylliset koodipätkät ja -pohjat) ja other.

Alakategoriaksi tulee elementin nimi, esimerkiksi multichoice. Multichoice-kategoriaan kuuluvat kaikki eri versiot elementistä. Elementtien nimet alkavat alakategorialla, jonka jälkeen tulee jokin täsmentävä sana, esimerkiksi multichoice-check. Sopivia nimiä ja nimeämiskäytäntöjä voidaan pohtia yhdessä aina tarvittaessa. Kategorioitakin voidaan muuttaa tai lisätä.

Elementin eri versiot

Yhdessä voidaan pohtia myös sitä, milloin jokin elementti lisätään kirjastoon omana versionaan. Lähtökohta on, että elementti toimii eri tavalla kuin aiemmat versiot. Esimerkiksi monivalinnan eri versioita ovat elementti, joka hyväksyy vain yhden vaihtoehdon ja näyttää vastauksen heti ja elementti, jossa voi valita useamman ja vastaus näytetään check-nappia painamalla. Eri teknologioilla (esim. jQuery, React, Builder) toteutetut elementit ovat GitHubissa omina versioinaan, mutta elementtikirjastossa yhden version alla, jos elementit toimivat samalla tavalla. Joitakin pieniä toiminnallisuuksien eroja voidaan lisätä samaan elementtiin ja ottaa käyttöön tai kommentoida pois tarpeen mukaan.

Koodien lisääminen ja dokumentointi

Elementin sijaintina on GitHubin element-library repository -> kategoria-kansio -> alakategoria-kansio -> elementti-kansio eli esimerkiksi tasks/multichoice/multichoice-check. Koodeista lisätään mahdollisimman riisuttu versio. Mukana voi hiukan elementistä riippuen olla HTML, CSS ja JavaScript (tai esim. PHP). Koodeja on hyvä kommentoida. Lisäksi kansioon tulee readme-tiedosto, johon kirjoitetaan tarkempi ohjeistus elementin käytöstä. Ohjeistus on tärkeä etenkin, jos kaikkea oleellista koodia ei ole mukana esimerkissä (joissakin tilanteissa kaikkia koodeja on hankala irrottaa kokonaisuudesta). Ohjeistusta kannattaa päivittää tarvittaessa. Myös elementtejä on tarkoitus kehittää, eli jos tekee koodiin jonkin parannuksen tai korjauksen, se tulee päivittää kirjastoon kaikkien saataville.

2 Codepen-esimerkki

Elementeistä lisätään kirjastoon interaktiivinen esimerkki CodePen-palvelun avulla. [Linkki ja tunnukset.] Aiemmin luodut Penit löytää klikkaamalla profiilikuvaa oikeassa yläkulmassa ja esiin tulevasta valikosta Pens. Koodit on tarkoitus tallentaa yksityisinä, joten Penit löytyvät Private-välilehden alta.

Uuden Penin luominen

Luo uusi Pen klikkaamalla New Pen -nappia oikeassa yläreunassa. Syötä kenttiin tarvittavat HTML-, CSS- ja JavaScript-koodit. Esimerkin tulisi esittää elementin toiminnallisuus mahdollisimman yksinkertaisesti, myös tyylit on hyvä pitää minimissä (mallia voi katsoa aiemmista toteutuksista). Esimerkistä on hyvä poistaa projektikohtaiset tyylit ja sisällöt.

HTML-kenttään ei tarvita html- ja head-tageja. Asetuksista (Settings-nappi oikeassa yläreunassa) voi muun muassa lisätä luokkia html-elementtiin, ottaa käyttöön CSS Normalizen ja Autoprefixerin sekä lisätä ulkoisia CSS- ja JavaScript-kirjastoja (esim. Fontawesome, jQuery).

Penin tallentaminen ja upotus

Tallenna klikkaamalla yläreunasta Save as Private. Klikkaa lopuksi Embed oikeasta alakulmasta. Valitse oletuksena auki näkyväksi välilehdeksi pelkkä Result ja säädä alueen korkeus sopivaksi niin, että koko toteutus on näkyvissä. Kopioi alareunasta HTML-koodi. Se lisätään kirjastossa elementin kuvauskenttään, siitä lisää seuraavassa luvussa.

3 Elementin lisääminen ja muokkaaminen

Uuden elementin luominen ja nimeäminen

- Siirry admin-näkymään etusivun Admin-linkistä tai osoitteesta [linkki]. Valitse valikosta Posts -> Add new.

- Kirjoita elementille nimi ylimpään tekstikenttään.

CodePen-esimerkki ja kuvat

- Sisältökenttään (iso tekstikenttä) tulee ensimmäiseksi CodePen-upotus. Se kopioidaan editorin Text-näkymään (ei Visual-näkymään).

- Sisältökenttään tulee myös kuvagalleria, joka lisätään Add media -napista. Jos elementissä on paljon interaktiivisuutta ja animaatioita, esimerkkikuvat kannattaa lisätä GIF-animaationa. Animaation voi tallentaa ruudulta esimerkiksi Screen to Gif -ohjelmalla.

- Klikkaa Create gallery ja lisää kuvat. Valitse haluamasi kuvat Media library -välilehdellä.

- Kirjoita Caption-kohtaan elementin kuvaus ja Alt text -riville elementin otsikko. Elementin otsikko on muotoa Elementin nimi (Asiakas: Projektin nimi) eli esimerkiksi True-false (Lemminkainen: Insider information). Nämä näkyvät, kun kuvan

avaa kirjastossa lightboxiin. (Otsikko tulee kuvagalleriaan näkyviin nimenomaan alt-attribuutista, ei titlestä.)

- Klikkaa Create a new gallery. Gallery settings -valinnoista Link to -kohtaan valitaan Media file, Columns maksimiin. Jos kuva ei avaudu kirjastossa lightboxissa, Link to -vaihtoehdoista on luultavasti valittuna jokin muu kuin Media file (joten muista vaihtaa se).

- Lopuksi klikkaa Insert gallery. Galleria ilmestyy tekstikenttään, siihen kohtaan missä kohdistin on. Galleria laitetaan CodePen-esimerkin jälkeen, sen voi tarvittaessa helposti siirtää editorin Text (html) -näkymässä.

- Kuvien otsikoita ja kuvauksia voi päivittää klikkaamalla Add Media, etsimällä oikean kuvan ja muuttamalla tietoja oikealla olevasta sarakkeesta. Muutokset tallentuvat automaattisesti. Tietoja voi päivittää myös menun kautta Media-valikosta.

- Olemassa olevaan galleriaan pääsee lisäämään kuvia tekstieditorin Visual-näkymän kautta. Klikkaa kuvia ja yläpuolelle ilmestyvää kynäkuvaketta (Edit). Myös tätä kautta voi muokata kuvan tietoja. Kuvia voi lisätä vasemman reunan Add to Gallery -linkin kautta. Tee haluamasi muutokset ja klikkaa Update gallery.

Muut tiedot

- GitHub-link: linkki elementin sijaintiin GitHubissa.

- Description: kuvaus elementistä. Miten se toimii, miten se eroaa muista elementin versioista jne. Kirjoita englanniksi.

- Instructions: ohjeet elementin käyttöön.

Kategoriat ja tagit

- Element type: elementin kategoria, valitse sekä ylä- että alakategoria. Uusia kategorioita voi lisätä tarvittaessa laatikon alareunasta. Kategoriat eivät välttämättä näy oikein kategorialaatikossa; esimerkiksi jollakin yläkategorialla voi näyttää olevan on vain yksi alakategoria, vaikka todellisuudessa niitä on enemmän. Admin-näkymän valikosta Posts -> Element type voi tarvittaessa tarkistaa kaikki yläkategoriat ja niihin kuuluvat alakategoriat. Siellä listan pitäisi näkyä oikein.

- Customers: lisää projektin asiakas.

- Project topics: lisää projektin aihe. Lista kaikista aiheista löytyy valikosta Posts -> Project topics. Jos et ole varma aiheesta, tarkista projektipäälliköltä.

- Project names: lisää projektin nimi. Nimi on muotoa ”Asiakas: Projektin nimi”. Jos et ole varma nimestä, tarkista se projektipäälliköiltä.

- Element topics: lisää elementin aihe. Tätä ei tarvita aina; tarpeellisuuden voi tarkistaa projektipäälliköltä tai käyttöliittymäsuunnittelijalta.

- Technologies: lisää käytetyt teknologiat, esimerkiksi Angular, React, jQuery jne.

Featured image

- Oikeasta alakulmasta löytyy Featured image, eli kuva, joka näkyy elementin yhteydessä pienenä oikeassa yläkulmassa elementtien listausnäkymässä.

Elementin julkaiseminen/muutosten tallentaminen

- Klikkaa lopuksi Publish (tai Update, jos muokkaat elementtiä).

4 Uuden käyttäjän luominen

- Siirry Admin-näkymään yläreunan admin-linkistä tai osoitteesta [linkki].

- Sivuvalikosta Users -> Add new.

- Kirjoita Username (mielellään etunimi, näkyy elementin tiedoissa lisääjän nimenä).

- Kirjoita sähköpostiosoite (kirjautua voi usernamen tai emailin avulla).

- WordPress luo automaattisesti salasanan, ja sen voi vaihtaa klikkaamalla Show password.

- Jos haluat tiedot tunnuksesta sähköpostiin, jätä ruksi Send user notification –kohtaan.

- Rooliksi Administrator, jotta pääsee muokkaamaan kaikkea.

- Klikkaa Add new user.

- WordPress näyttää oletuksena admin-näkymän yläpalkin myös silloin, kun ei ole admin-näkymässä. Sen saa piilotettua muokkaamalla oman profiilin tietoja. Mene Users-sivulle, klikkaa omaa tunnustasi, ota ruksi pois Show Toolbar when viewing site -kohdasta ja klikkaa alhaalta Update user.

5 Salasanan vaihtaminen

- Siirry Admin-näkymään yläreunan admin-linkistä tai osoitteesta [linkki].

- Valitse sivuvalikosta Users ja etsi listalta tunnus, jota haluat muokata.

- Vieritä alas ja klikkaa Generate password. Kirjoita kenttään haluamasi salasana.

- Klikkaa Update Profile.