Opinnäytetyö AMK

Tietotekniikka

2021

Tuomas Ranta

SÄHKÖINEN ASIAKASREKISTERI JA PROJEKTINHALLINTA

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietotekniikka

2021 | 28 sivua

Tuomas Ranta

SÄHKÖINEN ASIAKASREKISTERI JA PROJEKTINHALLINTA

Opinnäytetyön tarkoituksena oli tuottaa kauneudenhoitoalan yrityksille tarkoitettu sähköinen asiakasrekisteri korvaamaan paperiset asiakaskortit, joiden käsittely ja säilytys koettiin kyselyn mukaan hankalaksi kauneushoitola Lempi Spassa. Sähköinen asiakasrekisteri olisi tarkoitus tuotteistaa tulevalle ohjelmistoalan yritykselle osana ajanvarausjärjestelmää. Asiakasrekisteri toteutettiin selainpohjalle käyttäen palvelinpuolen PHP-kieltä. Ulkoasu suunnitteluun käytettiin Html-, JavaScript- ja CSS-kieliä.

Projekti toteutettiin tutkimalla samalla projektinhallinnan merkitystä projektin lopputulokseen. Projektinhallinnan avulla voidaan hallinnoida, organisoida ja suorittaa projekti onnistuneesti koko projektin elinkaaren ajan. Projektinhallinnalla tarkoitetaan selkeän tavoitteen omaavaa tehtävää, jossa täyttyvät suunnitelmallisuus ja tavoitteellinen työskentely. Opinnäytetyössä käydään läpi projektin vaiheet koko elinkaaren aikana, jotta saataisiin kokonaiskuva onnistuneen projektin läpiviemisestä.

Opinnäytetyön teoriaosuudessa käydään läpi projektissa käytetyt teknologiat. Kuinka verkkosovellus rakentuu ja mitä tarkoitetaan pilvipalveluilla? Tietokantojen merkitys yritysmaailmassa on suuri, ja ne ovat jokaisen yrityksen jokapäiväisessä käytössä. Asiakasrekisterikin käyttää tietokantaa asiakas tietojen hallintaan.

ASIASANAT:

asiakasrekisteri, projektinhallinta, WWW-ohjelmointi, projektipäällikkö, projektijohtaminen

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information technology

2021 | 28 pages

Tuomas Ranta

ELECTRONIC CUSTOMER REGISTER AND PROJECT MANAGEMENT

The purpose of the thesis was to produce an electronic customer register for beauty care companies to replace paper customer cards, which, according to the survey, were considered difficult to handle and store at the beauty salon Lempi Spa. The electronic customer register would be commercialized for a future software company as part of an appointment system. The client registry was implemented for the browser template using the server-side PHP language. Html, JavaScript and CSS were used to design the layout.

The project was implemented while studying the importance of project management in the project outcome. Project management allows you to manage, organize, and execute a project successfully throughout the project lifecycle. Project management refers to a task with a clear goal, where systematic and goal-oriented work is fulfilled. The thesis reviews the stages of the project throughout its life cycle in order to get an overall picture of the implementation of a successful project.

The theoretical part of the thesis reviews the technologies used in the project. How is the web application built and what is meant by cloud services? Databases are of great importance in the corporate world and are in the daily use of every company. The customer registry also uses the database to manage customer information.

KEYWORDS:

customer register, project management, web programming, project manager and project management

SISÄLTÖ

KÄYTETTY SANASTO

business case	sisältää perustelut projektin tai tehtävän aloittamiselle. Se sisältää hankkeen hyväksymiseen tarvittavat tiedot.
css	(engl. Cascading Style Sheets) Erityisesti verkkosivuille kehitetty tyylisivu, jolla muokataan verkkosivujen ulkoasua.
EcmaScript	JavaScript-standardi, jonka tarkoituksena on varmistaa verkkosivujen yhteentoimivuus eri selaimissa.
HTML	(engl. Hypertext Markup Language) Standardoitu hypertekstin merkintäkieli verkkosivujen rakentamiseen.
Javascript	verkkoympäristössä käytettävä dynaaminen komentosarjakieli. Javascriptin tärkein ominaisuus on lisätä Web-sivuille dynaamista toiminnallisuutta.
MySQL	tietokantapalvelu pilvipohjaisten sovellusten käyttöönottoon.
PHP	(engl. Hypertext Preprocessor) palvelimen komentosarjakieli ja tehokas työkalu dynaamisten ja interaktiivisten verkkosivujen luomiseen.
Script	Ohjelmakoodi
Tagi	Merkintä, jolla ilmennetään rakenteisen aineiston rakennetta tai ulkoasua.

1 JOHDANTO

Opinnäytetyössä tutkittiin projektinhallintaa ja sen merkitystä IT-sovelluksille. Projektinhallinnalla hallitaan kokonaisuuksia, jotka koostuvat esimerkiksi projektin kustannuksista ja työtuntien suunnittelusta. Projektipäällikkö vastaa projektista koko sen elinkaaren ajan. Projektipäällikön täytyy ymmärtää riskejä ja etsiä niihin ratkaisuja.

Asiakasrekisterin suunnittelu lähti liikkeelle ajatuksesta perustaa ohjelmointialan yritys, jonka kohderyhmänä olisivat kauneushoitolat. Tarkoituksena olisi tarjota palveluja yrityksille, kuten ajanvaraus ja asiakasrekisteri. Huomattiin asiakasrekisterin tarve kysymällä muutamista kauneudenhoito alan yrityksistä. Kaikilla oli käytössä paperiset asiakaslomakkeet, joiden täyttäminen tapahtui aina yrityksen omissa tiloissa. Usein asiakkaat tulevat vastaanotolle, vain muutamia minuutteja ennen hoidon aloitusta, jolloin lomakkeen täyttö viivästyy ja hoidon aloitus samalla. Tästä seuraa myös myöhempien hoitojen viivästyminen. Pilvipalveluna toteutettu asiakasrekisterin voi täyttää paikasta riippumatta, ja se antaa asiakkaille mahdollisuuden täyttää se rauhassa, vaikka kotona. Asiakasrekisterin suunnitteluvaiheessa päädyttiin käyttämään - PHP, Html, CSS, JavaScript ja MySQL tekniikoita.

Pilvipalvelu on joustava, automatisoitu ja ketterä vaihtoehto vanhoihin tietokoneelle asennettaviin sovelluksiin verrattuna. Selaimen kautta toimivat palvelut toimivat matkapuhelimissa, tableteissa ja tietokoneissa. Lisäksi selaimen kautta toimivat sovellukset ovat ohjelmistovapaita, eli ne eivät vaadi ylimääräisiä sovelluksia koneelle asennettavaksi, vaan toimivat suoraan selaimessa. Ketterä vaihtoehto mahdollistaa palvelun automaattisen päivittämisen säännöllisin väliajoin tehokkaasti. Ketterän menetelmän tavoite on tyydyttää asiakas toimittamalla hänelle jatkuvasti uusimpia versioita ohjelmistosta tiheään tahtiin ja säännöllisin väliajoin. Pilvipalvelu voi tuottaa jopa 40 – 90 %:n kustannussäästöt tavalliseen ohjelmistokehitykseen verrattuna. (CGI, 2021) Hyvänä esimerkkinä voidaan mainita Adobe ja sen tuoteperhe. Kymmenen vuotta taakse päin Adobe myi niitä yksikkö hinnalla fyysisinä DVD-levyinä, mikä hidasti myös päivitysten saatavuutta, mutta nykyisin maksu tapahtuu kuukausi hinnalla ja sovelluksen lataukset suoraan verkosta. Tämä helpottaa sovellusten päivittämisen säännöllisemmin ja useammin.

2 ASIAKASREKISTERI

Asiakasrekisteri on tarkoitettu asiakastietojen hallintaan. Sillä kerätään asiakkaista tarvittavia tietoja kuten nimi, lääkitys, sairaudet, puhelinnumero ja muita tärkeitä tietoja tietokantaan. Asiakasrekisteri on kuin kirjasto, josta löytyy tiedot kirjoista ja niiden julkaisu vuosista, tekijöistä ja nimistä. Tämä asiakasrekisteri toteutettiin suoraan PHP ja MySQL kielillä koodaamalla. Opetuksellisista syistä se oli hyvä toteuttaa alusta asti, mutta nykyisin se olisi helpompi toteuttaa valmiilla sisällönhallintajärjestelmillä kuten Wordpress tai Drupal, joissa käyttäjienhallinta on valmiiksi tehtynä. Asiakasrekisteri on monella yrityksellä huonosti hoidettu, vaikka se on yrityksen toiminnan ydin. Sillä nähdään potentiaalinen asiakaskunta ja ylläpidetään yrityksen toimintaa. (HOLOPAINEN, 2019)

Yritys on vastuussa asiakkaistaan ja sen vuoksi henkilörekisteri on hyvä olla olemassa. Sen avulla vaalitaan asiakkuussuhteita, tiedotetaan ja kaikkein tärkeimpänä voidaan suorittaa kohdennettua mainontaa. Tietosuoja-asetus eli GDPR (General Data Protection Regulation) astui voimaan 25.5.2018 kaikissa EU:n maissa. Sen tarkoituksena on asiakkaiden kyky hallita omia tietojaan. Asiakkaalla on siis oikeus vaatia omia tietojaan nähtäväksi tai tuhottavaksi. Tiedot ovat myös tuhottava, jos asiakas suhde päättyy, koska tieto ei ole enää hoidolle välttämätöntä. Tietosuoja edellyttää tiedon suojaamista ulkopuolisilta ja arkaluonteisia tietoja kuten puhelinnumeroita tai terveydentilan tietoja ei saa kerätä, ellei ne ole hoidon kannalta välttämättömiä. Tiedon säilyttämiseen ei ole määrätty oikeaa tapaa. Se voi olla sähköisesti tai paperilla, kuitenkin tietojen keruusta tarvitsee kirjata julkinen tietosuojaseloste, jossa määritellään tiedon sijainti ja tallennus muoto. (Toivonen, 2019)

3 PILVIPALVELUT

Pilvipalveluilla tarkoitetaan mallia, jossa tietotekniikkaresursseja tarjotaan verkon välityksellä, ilman että asiakkaan tarvitsee tietää missä ne fyysisesti sijaitsevat. Resursseilla tarkoitetaan tietoliikenneyhteyksiä, laskenta- ja tallennuskapasiteettejä, sovelluksia, sekä palveluita. Pilvipalvelut tarjoavat uudenlaista teknologiaa sovelluskehitykseen. Ennen sovellukset asennettiin fyysisesti koneelle, mutta nykyisin voidaan tarjota suoraan selaimen kautta toimivaa sovellusta. Tallaista sovelluksen laskutusmenetelmää kutsutaan palvelullistamiseksi eli maksut voidaan periä kuukausi- tai vuosimaksulla. Tämä mahdollisuus muuttaa liiketoiminnan ajattelutapoja ja auttaa yritystä kustannustehokkaampaan palvelun tuottamiseen ja mahdollistaa palveluiden jatkuvan päivittämisen ketterillä menetelmillä. (Salo, 2010)

Asiakasrekisterin on tarkoitus toimia verkon kautta, jolloin asiakas ei ole sidottu fyysiseen sijaintiin. Tämä mahdollistaa asiakkaalle miellyttävämmän lomakkeen täytön esimerkiksi kotona kaikessa rauhassa. Kauneushoitola Lempi Spassa paperiset lomakkeet täytettiin vastaanotossa hieman ennen hoidon aloittamista omistaja Heidi Rannan toimesta. Lomakkeet lisäsivät asiakkaiden stressiä, koska lomake tuli täyttää melko nopeasti hoidon aloittamisen vuoksi. Asiakkaan tarkoitus oli tulla hoitoon rentoutuakseen arjen kiireistä ei täyttämään kiireellä lomaketta. Asiakasrekisteri kauneudenhoitoalalle antaa myös turvaa yritykselle, koska asiakas tiedot on dokumentoitu, voidaan väärin hoidetun asiakkaan kohdalla tarkistaa missä virhe tapahtui. Asiakasrekisteri antaa juridista suojaa yritykselle ja turvaa asiakasta esimerkiksi estää allergioiden syntyä, jos pähkinäallergia on etukäteen ilmoitettu. Asiakasrekisteri tulee olemaan osana ajanvarausjärjestelmää. Nämä palvelut olisivat tulevaisuudessa tarkoitus lisätä uuden ohjelmistoalan yrityksen palvelukokonaisuuteen.

4 KÄYTETYT TEKNOLOGIAT

4.1 HTML

HTML on www-sivujen luomisen vakiomerkintäkieli (Hyper Text Markup Language). Ensimmäinen HTML toteutettiin vuonna 1991, sitä pidettiin epävirallisena versiona ja 1995 julkaistiin HTML 2.0 versio virallisesti. Vuonna 1999 julkaistiin neljäs versio ja viides versio näki päivänvalon vuonna 2012 ja on edelleen uusin versio. HTML:n Perustajana pidetään Tim Berners-Leeta. HTML versioiden tarkoitus on jakaa informaatiota maailmanlaajuisesti Internetin välityksellä, kuten sanomalehtien, mutta se mahdollistaa myös interaktiivisuuden. (Data 1998a)

HTML:n tarkoitus on kuvata verkkosivuston rakennetta ja se koostuu elementeistä, jotka kertovat selaimelle, kuinka sisältö näytetään. Käytännössä HTML on sivuston juuri, ja se voidaan jakaa kahteen suureen tagiin head ja body, joiden sisälle kaikki rakentuu. Head-tagin sisälle tulee kaikki metatieto. Metatieto on tärkeää tietoa, mikä ei suoraan näy näytölle, mutta se määrittää dokumentin otsikon, tyylin, tiedostojen linkit (projektissa esimerkiksi css tiedostot on linkitetty HTML:ään), scriptit eli komentokielellä kirjoitettujen peräkkäin toteutettavien komentojen muodostama kokonaisuus, ja muut oheistiedot. Metadatassa eli tietoa kuvailevassa tiedossa voidaan käyttää seuraavia tageja: title eli otsikko, style (tyyli), meta (tieto mikä kuvailee tietoa), link (määritys toiseen kohteeseen), script (komentosarja) ja base (tietojen kokoelma). HTML:n toinen keskeinen elementti on body. HTML:n viides versio kuitenkin mahdollistaa sen pois jättämisen kuten myös head elementin, mutta huomioitavaa on vanhojen selainversioiden tuen puute tähän ominaisuuteen. Body-tagin ympärille rakentuu kaikki näytöllä näkyvä rakenne, kuten taulukot, tekstit, lomakkeet, valikot, napit ja niin edelleen. Body osioon ja sen sisältöön voidaan kohdistaa siis kaikki tyylimäärittelyt, mitä tarvitaan ulkoasun rakentamiseen. (Koulutuksen tutkimuslaitos, 2016)

Asiakasrekisteri on rakennettu elementein ja elementtien ulkoasut on muokattu CSS tyylimäärittelyn avulla projektin suunnitelmien mukaiseksi (Kuva 1).

Kuva 1. HTML:n juureen laitetaan kaikki elementit, kuten tekstiruudut ja napit.

HTML-tiedosto kerrotaan selaimelle ensimmäiseksi, jotta selain osaa käsitellä dataa. Tiedoston alkuun ilmoitetaan mitä tyyppiä se on seuraavasti: <!DOCTYPE html>. Varsinainen HTML dokumentti alkaa tagilla: <html> ja se lopetetaan tagiin: </html>. Kielen valinta tulee muistaa, jotta lukijoiden ja hakukoneiden on helpompi käsitellä tekstiä. Kielen valinta ilmoitetaan seuraavasti: <html lang=”en-US”>.

Html5 mahdollistaa uusia ominaisuuksia aikaisempiin versioihin verrattuna. Opinnäytetyön asiakasrekisterissä on käytetty esimerkiksi taulukon pakollista täyttö ominaisuutta. Rakenne-elementit ovat alkujaan class-määritteistä, joita voidaan nykyisin käyttää suoraan elementteinä, kuten footer, header, nav, article sekä section. Footer tarkoittaa yleisesti sivustossa alaosaa, header on yläosa, nav on valikko ja article on tekstiosa. Huomattavaa kuitenkin on, että valikon voi sijoittaa suoraan header osioon. Nämä elementit ovatkin semanttisia eli kuvastavat merkitystä ja helpottavat niiden käyttöä. Mahdollisuus on myös nimetä div elementti edelleen vanhaan tapaan id-attribuutilla esimerkiksi header osioksi seuraavalla tavalla: . Tulevaisuudessa uudet elementit saattavat hyödyttää hakukoneiden toimintaa ja sivustojen hakukoneoptimointia. Myös kirjoitetavan kentän asettaminen pakolliseksi on

mahdollista. Projektissa on asetettu pakolliseksi esimerkiksi sähköpostin vaatiminen, poistaen mahdollisuuden olla vastaamatta oleellisiin kysymyksiin, joiden merkitys hoidossa on välttämätön tieto. Lisäksi uusia ominaisuuksia ovat automaattinen täydennys ja raahaa ja pudota toiminto eli ”Drag and drop”. (Data, w3schools, 1998)

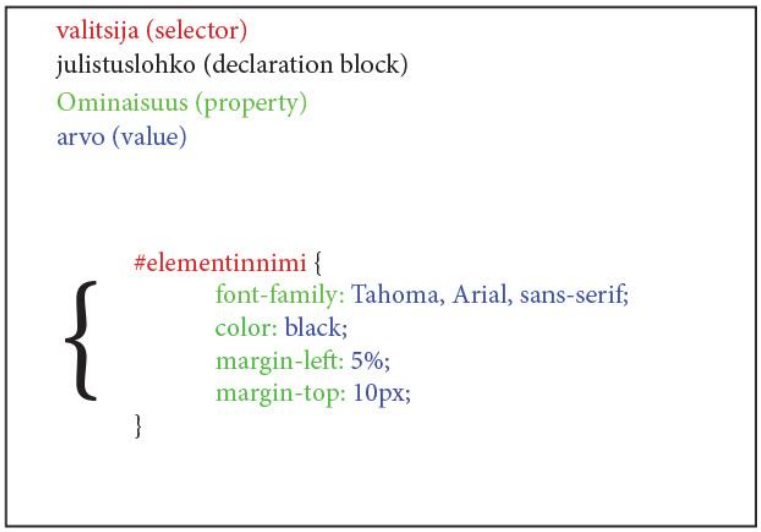
4.2 Tyylitiedosto - CSS

CSS (Cascading Style Sheets) on tyylimäärittelykieli, jonka tarkoitus on määrittää verkkojulkaisun ulkoasu. CSS on W3C:n kehittämä tekniikka, joka luotiin vuonna 1996, kun CSS:n ensimmäinen versio julkaistiin. CSS pääasiallisin tarve oli muokata verkkosivuja sanomalehtien kaltaisiksi. Aluksi CSS:n tuki selaimilla oli heikkoa, koska selaimet eivät noudattaneet standardeja. Tilanne parani vuonna 2003 Firefoxin ja Safarin julkaistujen selaimien myötä. Nykyisin on käytössä toinen versio, joka julkaistiin 2011 ja kolmas versio rinnakkain. Jälkimmäiset versiot syrjäyttivät ensimmäisen version kokonaan. (Howe 2014, s. 36-37)

Tyylitiedostojen pääte on css, ja se täytyy aina linkittää haluttuun HTML-tiedostoon head-tagien sisälle link-elementillä. CSS tyylimäärittelyt ovat myös mahdollista sisällyttää suoraan html-tiedostoon, jonka käyttöä ei suositella. Suurissa projekteissa useassa paikassa irrallaan olevat tyylimäärittelyt hukkuvat kokonaisuuteen ja heikentävät tyylien hallittavuutta verrattuna ulkoasunhallintaan saman tyylitiedoston sisällä, jolla hallitaan koko projektin ulkoasua. Tyylitiedostoja voi olla samassa projektissa useampiakin riippuen tekijän tavasta toimia. (Data 1998a)

Tyylitaulukon rakenne aloitetaan aina valitsijalla (Kuva 2). Valitsija kertoo mihin halutut ulkoasumuutokset vaikuttavat. Se voidaan suoraan antaa Tagille eli html-elementille kuten h1 tai div. Valitsija voidaan myös tarkemmin kohdentaa käyttämällä luokka määritystä lisäämällä piste eteen tai elementin id nimeä lisäämällä eteen risuaita merkki. Lisäksi on Pseudo-valitsijat, kuten hiiren kursorin meno elementin päälle tai linkkien seuranta, joka ilmoittaa onko linkissä käyty aikaisemmin. Julistuslohko (engl. declaration block) on alue, mikä sisältää kaiken ulkoasun muuttamiseen liittyvät asiat. Tämä alue on merkattuna kuvaan suurella aaltosulkeella. Ominaisuus (engl. property) määrittää mitä halutaan muuttaa, esimerkiksi taustaväri, fontti tai sijainti. Kaksoispiste ilmoittaa ominaisuuden päättymisen ja jää odottamaan sille asetettua arvoa eli value. Arvo kuten minkä värinen on, musta vai punainen. Arvo voi kertoa myös, kuinka suuri elementti on esimerkiksi kahdeksan pikseliä vai kaksikymmentä prosenttia. Jokaisen ominaisuuden ja arvon päättää puolipiste. (Linkola, 2021)

Kuva 2. CSS:n rakenne.



Tyylien arvoja voidaan antaa eritavoilla. Värin asetuksetkin voidaan ilmaista neljällä eri tavalla. Ennalta määritetty väri kertoo yleisnimen, kuten punainen tai sininen. Tämä merkintätapa on huonoin, koska esimerkiksi punaisen eri värisävyjä on yksistäänkin jo tuhansia. Lisäksi on muistettava, että värit nähdään yksilöittäin hieman erivärisinä. 8-bittinen hexadecimaaliluku on yleisin merkintätapa ja ilmaistaan esimerkiksi seuraavasti: #ff0012. Luvussa kaksi ensimmäistä numeroa tai kirjainta määrittävät punaisen värin määrän, seuraavat kaksi määrittävät vihreän värin ja viimeiset kaksi määrittävät sinisen värin. (Kuvankäsittelyohjelmalla saadaan esimerkiksi suoraan värin arvo

hexadecimaalilukuna pipettityökalua apuna käyttäen.) 0-255 desimaali voimakkuudella voidaan myös ilmaista väri esimerkiksi: rgb(255,255,0). R-kirjain tarkoittaa punaisen värin määrää, g-kirjain vihreän värin määrää ja b-kirjain sinisen värin määrää. Mitä suurempi arvo värissä on, sitä enemmän väriä käytetään. Prosentuaalinen voimakkuus on edeltäjän tapainen, mutta arvot syötetään prosentteina nollasta sataan. Väriopin mukaan kaikki värit voidaan sekoittaa kolmea pääväriä käyttäen eli punainen, sininen ja keltainen. Lisäksi on CMYK väriavaruus, joka eroaa RGB:stä. CMYK värejä käytetään painopinnan valmistuksessa kuten julisteiden tulostuksessa eli ne ovat maalia, kun taas RGB on näytöllä näkyvää valoa. (Kaukoniemi, 2000)

Tyylien etäisyydet voidaan ilmaista eri yksiköin. Yleisin verkkosovelluksessa käytettävä yksikkö on pikseli, jonka lyhenne on px. Niitä kutsutaan myös nimellä kuvapikselit. Lisäksi voidaan käyttää myös millimetriä, senttimetriä, tuumaa, suhdetta elementin kokoon eli em, prosenttia ja tulostinpistettä eli pt.

CSS on alustariippumaton ja se mahdollistaa nettisivujen ja sovelluksien käytön erikokoisilla laitteilla. Responsiivisuus eli skaalautuvuus on mahdollista tyylimäärittelyjen avulla asettamalla yksiköksi prosentit pikselien sijaan, koska prosentit muokkaantuvat näytön koon mukaan esimerkiksi kuva pienenee automaattisesti, jos selain ikkunaa pienennetään. Responsiivisuus on käytännössä projektin toimivuuden varmistamista erilaisilla laitteilla ja erikokoisilla näytöillä. (Aaltokangas, 2019) Tyylitiedostot ja Html toimivat molemmat ilman verkkoa selaimen kautta. Tämä mahdollistaa verkkosovelluksen toimivuuden suoraan tietokoneelta ilman Internet-yhteyttä. Huomioitava on kuitenkin sovellukset, mitkä käyttävät tietokantaa, koska ne vaativat palvelimen toimiakseen. CSS vähentää myös koodin määrää nopeuttaen samalla latausaikoja. Tyylimäärittelyt myös helpottavat ylläpitoa, koska ne suoritetaan yhdestä paikasta ja vaikutus on koko projektiin. Vanhoja Html-attribuutteja mitkä vaikuttavat ulkoasusuunnitteluun ei suositella käytettäväksi, vaan ne ovat pääsääntöisesti kaikki korvattavissa CSS tarjoamilla vaihtoehdoilla, kuten lihavointi. Tyylimäärittelyt suoraan Html-koodissa vaikeuttavat ohjelmoijaa löytämään oikean tyylimäärittelyn ja lisäksi saattaa aiheuttaa päällekkäisyyksiä tyyleissä. Tyylimäärittelyissä fontteja voidaan muokata monellakin tavalla. Voidaan esimerkiksi lisätä lihavuus seuraavasti: font-weight:bold; ja fontin kokoa voidaan muokata seuraavasti: font-size:12px;. Käytettävyyden kannalta marginaalien ja tyhjäntilan käyttö on ensiarvoisen tärkeää luomaan selkeän ja tyylikkään kokonaisvaikutelman. Liian tiiviisti esiintyvä teksti ”puuroutuu” yhdeksi kasaksi ja lukeminen vaikeutuu.

4.3 Javascript

Javascript on dynaamisesti tyypitetty tulkattava oliopohjainen kieli, jonka tärkein ominaisuus on dynaamisuuden eli liikkeen tuominen verkkosivuille. Se ei ole luokkapohjainen kieli kuten Java, jotka usein sekoitetaan keskenään nimen samanlaisuuden vuoksi. Kielen kehitti Netscape-yrityksessä vaikuttanut Brendan Eich vuonna 1996. Typescript nimitys muutettiin Javascriptiin lähinnä markkinoinnillisista syistä. Viimeisin versio on JavaScript 1.8.5. Se pohjautuu EcmaScript-standardiin ECMA-262 Edition 3. Standardoitua JavaScriptiä käytetään nimellä ECMAScript. JavaScript oli pitkään Front-end kieli eli asiakaspuolen ohjelmointikieli. AJAX tekniikka mahdollisti myös back-end eli serveripuolen ohjelmoinnin. Ajax rakentuu käytännössä seuraavista tekniikoista: HTML, EcmaScriptin, XML ja CSS. (Tutorials Point, 2015)

Ajaxin tarkoitus on vaihtaa taustalla pieniä määriä dataa palvelimella niin, että vain osa verkkosivusta tarvitsee ladata uudelleen. vaikka käyttäjä tekisi muutoksia. Tekniikka lisää verkkopalvelun vuorovaikutteisuutta, käytettävyyttä ja nopeutta. Javascript-kirjastoja löytyy maksullisia ja maksuttomia eli open source. Kirjastoja kannattaa käyttää projekteissa nopeuttaakseen lopputulokseen pääsyä. Kirjastot sisältävät widgettejä, eli käyttöliittymäkomponentteja useampiin erilaisiin tehtäviin, kuten interaktiivisuuden lisäämiseen tai ulkoasun yhden mukaistamiseen. Jquery ja Bootstrap ovat yleisiä kirjastoja, joita voidaan käyttää sivuston ulkoasun muokkaamiseen ja interaktiivisuuden lisäämiseen.

JavaScript-koodin voi kirjoittaa suoraan HTML-dokumentin sekaan, jolloin se merkitään seuraavasti: <script type="text/javascript"></script>. Komentojen väliin kirjoitetaan varsinainen JavaScript-koodi. Parempi tapa on sijoittaa JavaScript-koodi HTML-dokumenttiin erillisenä tiedostona, jolloin Html-tiedoston muokkaaminen helpottuu. Pääsääntönä olisi, että CSS, Html ja JavaScript pidettäisiin mahdollisimman erillään toisistaan tiedon helpomman löytämisen vuoksi. Jos JavaScript halutaan aloitettavaksi sivun latauksen yhteydessä, tulisi sen linkitys tai koodi sijoittaa head-tagien väliin. Body-tagin välissä se toteutetaan siinä kohtaa, mihin se on kirjoitettu.

Muuttujat ovat tiedon tallennuspaikkoja tietokoneen muistissa, jotka ohjelmoija itse nimeää. Muuttujien nimet saavat nykyisin sisältää myös skandinaavisia kirjaimia kuten ä ja ö. JavaScript on merkkiriippuvainen, joka on huomioitava kirjoittaessaan isoja- ja pieniä kirjaimia eli antiikva- ja gemenakirjaimia. Ne merkitsevät eri asiaa riippuen miten on kirjoitettu esimerkiksi ”Health” ja ”health” tarkoittavat koodissa eriasiaa.

JavaScriptissä ohjelmoijan ei tarvitse välittää muuttujien tyypeistä, koska ne määräytyvät automaattisesti. Tyypeillä tarkoitetaan arvoja kuten kokonaisluku eli int. Esimerkiksi jos muuttujan arvo on numeerinen, tulee muuttujaksi numero, jos taas luku on lainausmerkkien sisässä tunnistaa JavaScript sen merkkijonoksi eli stringiksi. Merkkijono koostuu merkeistä. Merkkijono sisältää yleensä tavallista tekstiä, mutta se voi olla myös erikoismerkkejä, jossa kaikki Unicode-merkit ovat sallittuja. Unicode on tietokonejärjestelmiä varten kehitetty merkistöstandardi. (Microsoft)

4.4 PHP

PHP eli Hypertext Preprocessor (suom.hypertekstin esikäsittelijä) on yleiskäyttöinen skriptikieli. PHP-kielen kehitys alkoi 1994, jolloin Rasmus Lerdorf julkaisi muutamia Perl-skriptejä. Nykyisin on käytössä PHP version viides versio. PHP:tä käytetään yleisimmin Web -sovelluskehitykseen. Sen etuna on mahdollisuus liittää se suoraan HTML koodiin. Toisin kuin HTML-sivusto PHP kuuluu palvelin puolen kieleksi ja se käsitellään PHP-ohjelman tai moduulin avulla kuten Apache sovelluksen, joka on asennettuna palvelimelle. PHP-koodilla voidaan tehdä kyselyjä tietokannasta, lukea tiedostoja, luoda kuvia, ja kirjoittaa tiedostoihin. Se voi myös keskustella etäpalvelimien kanssa. Koska PHP-koodi käsitellään heti palvelimella eikä selaimella, on se myös alttiimpi kirjoitusvirheistä johtuville virheille. HTML- koodi puolestaan toimii varmemmin koska selaimet pyrkivät korjaamaan kirjoitusvirheiden aiheuttamia ongelmia. (Tutorial Republic, 2021)

PHP-koodi muodostuu komennoista, jotka suoritetaan järjestyksessä ylhäältä alaspäin samalla tavalla kuin HTML:än kanssa. Komennot lopetetaan aina puolipisteeseen ja jokainen komento kirjoitetaan omalle rivilleen selkeyttääkseen koodin lukemista. PHP-tiedoston pääte on php. PHP- koodi upotetaan HTML-sivulle aloitus- ja lopetustagien sisään. Tagit merkitään koodissa merkein: <?php><?>. PHP:ssä muuttujaa ei tarvitse esitellä ennen käyttöä toisin kuin valtaosassa ohjelmointikielissä. Ohjelmoinnissa tärkeä osa on kommentointi. Sillä mahdollistetaan suurienkin projektien pysyminen kasassa ja selkeytetään kertomalla mitä kukin koodin osa tekee. Kommentoinnilla helpotetaan myös päivitettävyyttä. Toisen suunnittelijan on helpompi jatkaa samaa projektia, jos on tietoa jokaisesta valmiista koodista. Kommentointi suoritetaan useammallakin tavalla. Merkintä: // kommentti koskee kokonaista riviä, kun taas teksti aloitus- ja lopetustagien sisässä toimii kommenttina useammallakin rivillä. Kommentointi näin tapahtuu merkein: */* teksti tähän */.* (Koulutus- ja konsultointipalvelu KK Mediat, 2000-2021)

4.5 Tietokannat ja MySQL

MySQL-relaatiotietokannan perusti 1995 suomalainen Michael Widenius yhdessä ruotsalaisen David Axmarkin kanssa. MySQL otettiin käyttöön 1996 ja siitä rakentui suosittu web-palveluiden tietokanta. MySQL-tietokantaa käytetään mm. PHP, Python ja Perl-ohjelmointikielillä ja julkaisu tapahtuu Apache-palvelinta käyttäen Linux pohjalla. Yksi keskeinen syy MySQL suosioon on GPLv2 lisenssi eli se mahdollistaa oikeuden käyttää, muuttaa, kopioida ja jakaa edelleen ohjelmia ilmaiseksi. (TechTarget, 2021)

Tietokanta on kokoelma tietoa. Siitä voidaan hakea tietoa ja muuttaa sitä. Tietokantoja on kaikkialla esimerkiksi nettisivujen tunnukset ja salasanat sekä kaupan tuotteet ja niiden hinnastot. Tietokantojen suunnittelussa on haasteita. Suuren määrän tietoa omaavat tietokannat täytyy suunnitella tehokkaasti, jotta jatkuvasti tehtyjen hakujen toiminta olisi riittävän nopeaa. Lisäksi on huomioitava samanaikaisuus, jos tietokantaa muutetaan useammalta taholta samaan aikaan. Tilanteet saattavat aiheuttaa yllätyksiä, ja siksi tietokannan tulisi pysyä järkevänä ilman sekaannuksia. Taulun luonti tapahtuu komennolla ”CREATE TABLE”. Projektissa tietokannan nimi on asiakasrekisteri ja siinä käytetyt taulut ovat nimeltään users ja admin. (Kuva 3). Users käsittelee asiakastietoja ja admin hallintapuolen käyttäjiä. Projektissa käytetään vain toista, koska käyttäjienhallinta tarvitsee projektin aikataulutuksen vuoksi lisäaikaa toteuttamiseen ja sen tuottaminen siirtyy tulevaisuuteen. Sarakkeiden tyypeillä tarkoitetaan missä muodossa tieto on. Int tarkoittaa kokonaislukuja ja varchar tarkoittaa merkkijonoa. Suluissa olevalla luvulla ilmoitetaan montako merkkiä tieto saa enintään olla.

Kuva 3. Projektiin luodut tietokanta taulut ja sarakkeet.

Sarakkeille voidaan antaa myös pääavain, joka yksilöi jokaisen taulun rivin. Pääavaimen avulla voidaan viitata helposti mihin tahansa riviin. Tavallisesti pääavaimeksi valitaan id. Id-numerossa on yleensä juokseva numerointi ykkösestä eteenpäin, jonka avulla voidaan yksilöidä jokainen käyttäjä. Pääavaimeksi voidaan kuitenkin valita minkä tahansa sarakkeen tai sarakkeiden yhdistelmä. Select komennolla voidaan suorittaa kysely eli hakea tietoa taulusta. Jos halutaan hakea kaikki tiedot taulusta käytettään *-merkkiä. Voidaan myös hakea vain tietty tieto esimerkiksi komennolla: SELECT lname FROM users;. Tämä valitsee sukunimen taulusta käyttäjät.

Tietokannan hakuun voidaan vaikuttaa monellakin tavalla. Voidaan järjestää hakutulos pienemmästä suurempaan tai päinvastoin, karsia tuloksesta pois samat vastaukset, sekä järjestää aakkosjärjestykseen. Tietokannassa voidaan myös muokata ja poistaa tietoa komennoilla update ja delete. Suurien tietomäärien tarkastelussa myös ryhmittely on hyvä keino hakea tietoa. Komennon groupin käytöllä voitaisiin hakea esimerkiksi samoissa ammateissa olevien tupakointi tiedot yhteen. Relaatiotietokantojen keskeisin etu on, että taulut voivat viitata toisiin, joka mahdollistaa kyselyiden teon keräten tietoa useista tauluista viittausten perusteella. Viittauksena yleensä käytetään rivin id-numeroa. (Laaksonen, 2020)

5 PROJEKTINHALLINTA

5.1 Historia

Projektinhallintaa on esiintynyt jo esihistorialliselta ajalta, joskin silloin päämääränä on ollut yksinkertaisesti hengissä pysyminen. Tuolloin projektiksi saatettiin kutsua esimerkiksi tulen tekoa. Projekti on ennalta määritettyyn päämäärään tähtäävä kokonaisuus, jossa on aina aloitus ja lopetus. Se voi olla monimutkainen ja sisältää useita eri tehtäviä. Projektissa rajataan kesto, kustannukset ja laajuus. Projektinhallinta kehittyi teollisen vallankumouksen aikana, jolloin keksittiin merkittäviä keksintöjä esimerkiksi James Wattin keksimä höyrykone 1770-luvulta ja George Stephensonin höyryjuna 1814 luvulta. Kaikissa onnistuneissa suurissa projekteissa yhteistä on ollut onnistunut projektinhallinta. (Murch, 2002, s. 3-5)

Projektinhallinnan kehityksen kannalta yksi tärkeä henkilö oli Henry Gantt (1861 - 1919), joka kehitti Gantt-kaavion ensimmäisen maailmansodan aikana huomatessaan kaavioiden avulla ymmärtävänsä paremmin vaikeita rakentamisprosesseja sota-alusten rankentamisen yhteydessä. Gantt-kaavio on aikojensaatossa kehittynyt, mutta sisältö ja muoto on säilyttänyt muotonsa. Gantt-kaavio vastaa kysymyksiin projektin kestosta, kunkin tehtävän määräajoista ja vastuu henkilöistä, viivästyksien merkityksestä projektiin, miten projektin viivästyksiin voidaan vastata tai nopeuttaa toisia projekteja.

Tärkeää on myös rajata erikseen kriittiset tehtävät ja niiden aikataulutus. Kriittisillä tehtävillä tarkoitetaan projektissa välttämättömiä tehtäviä, jotka ovat sidoksissa myöhemmin tehtäviin. Ilman kriittisen tehtävän loppuunsaattamista ei ole mahdollista aloittaa seuraavaa tehtävää ja näin ollen projektin aikataulutus venyy. Kaavio seuraa myös projektiin käytettyä kokonaiskustannusta, sekä yksittäisten tehtävien kustannuksia. Rajaamalla kustannukset yksittäisiin tehtäviin saadaan kokonaiskustannuksista mahdollisimman tarkka, eikä hinta-arvion heitto kasva liian suureksi projektin toteuttamista ajatellen. (Murch, 2002, s. 6-10)

5.2 Moderni ohjelmistokehitys

Vesiputousmallia (engl. waterfall model) voidaan pitää ensimmäisenä prosessimallina. Sen suunnitteli Winston Roycea 1970-luvulla. Vesiputousmalli on suosittu

ohjelmistoyrityksissä, vaikka siinäkin on heikkoutensa. Vesiputousmallin ehdoton edellytys on hyvä dokumentointi ja huolellinen tarkastaminen.

Vesiputousmalli on vaiheittainen suunnitteluprosessi, jossa tietty asia tehdään kerrallaan valmiiksi. Karkeasti sanottuna tuotteen kiertokulku menee seuraavasti: suunnittelu, toteutus, testaus ja lopuksi toimitus. Vesiputousmalli soveltuu projekteihin, joissa vaatimuksien ei uskota muuttuvan kesken prosessin. Ohjelmistokehityksessä liian yksityiskohtaiseen suunnitteluun kulutettu aika kuluu hukkaan, jos työn vaatimukset muuttuvat matkan varrella. Vesiputousmallin etuja ovat selkeys ja kehitetty laaja teoria- ja työkalutuki. Mallille on tehty myös toimiva laadunvarmistus ja prosessin parannus. Asiakas saa jo suunnittelu vaiheessa kustannusarvion, joka usein on väärä yllättävistä tilanteista johtuen. Haittoja ovat vaatimusten muuttumiset, joka voi pahimmillaan keskeyttää projektin kulun tai viivästyttää valmistumista ja testauksia. Asiakas näkee lopputuotteen vasta sen valmistuttua, tästä syystä usein ketterät menetelmät koetaan paremmaksi, jossa muutoksia tehdään jatkuvasti ja luovasti. (Juha Taina, 2009)

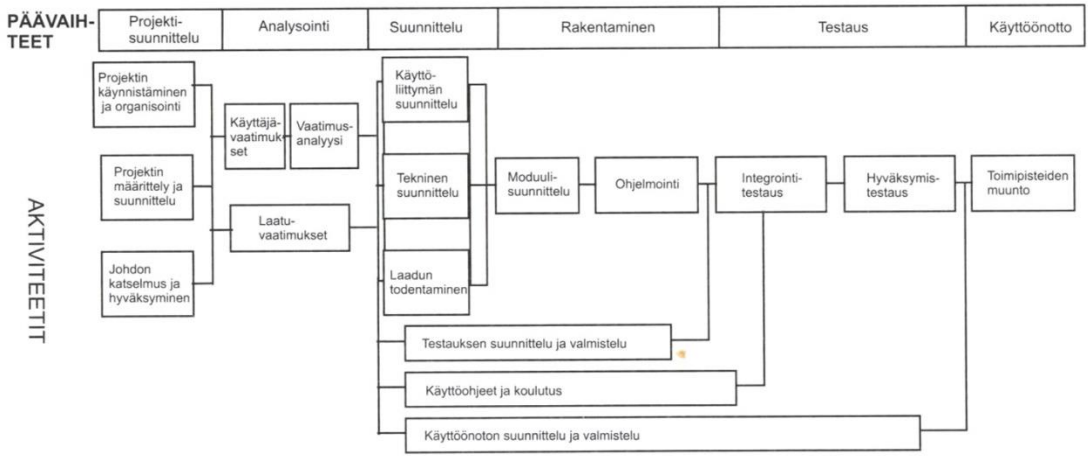
Ketteriä menetelmiä on useita, mutta pääpiirteittään ne ovat toistensa kaltaisia. Näitä ovat esimerkiksi SCRUM, Lean, Kanban ja SAFe. Ketterät menetelmät epäonnistuvat harvemmin, koska ohjelmistoja kehitetään pienemmissä osissa ja lyhyinä ajanjaksoina eli iteraatioina. Iteraatio sisältää myös suunnittelun, toteutuksen, testauksen ja toimituksen. Ketterien menetelmien etuna on, että iteraation pituus on yleensä yhdestä kahteen viikkoa, kun taas vesiputousmallin vieminen alusta loppuun saattaa viedä vuosia. Jokaisen iteraation lopuksi koko projektin tuotanto arvioidaan uudelleen, jonka jälkeen suunnitellaan seuraavia iteraatioita. (EMG – Educations Media Group, 2021)

Asiakasrekisterin työstämisessä on käytetty ketteriä menetelmiä. Projektilla oli selkeä tavoite saavuttaa toimiva asiakasrekisteri, vaikka ominaisuuksia kehitetään jatkuvasti. Sovelluskehityksessä päivityksiä tulee usein ja ne täydentävät asiakasrekisterin ominaisuuksia. Tulevissa Iteraatioissa olisi tarkoitus toteuttaa käyttäjien hallinta paremmin. Rajata kenellä on oikeuksia mihinkin tiedon katseluun ja mahdollistaa käyttäjätietojen muuttamisen rekisterinylläpitäjänä suoraan sovelluksesta. Tällä hetkellä tietojen muuttaminen ja poistaminen suoritetaan suoraan MySQL-tietokannasta PHPMyAdmin-verkkosovelluksella. Ainoastaan tietokannan katsominen onnistuu asiakasrekisteriin luodulla hallintasivustolla.

5.3 Projektin elinkaari

Elinkaari tarkoittaa projektin kokonaispituutta aina suunnittelusta käyttöönottoon. Projekti vaikuttaa sidosryhmiin, joita ovat kaikki tahot, joiden kanssa yritys on tekemisissä eli toiminta mikä vaikuttaa ja jotka vaikuttavat sen toimintaan. Erilaisia ryhmiä on yrityksen sisällä ja ulkopuolella. projektipäälliköt, työntekijät, sovelluskehityksestä vastaava johto, ylin it-johto, it-arkkitehtuurin suunnittelusta ja ohjelmiston elinkaariprosessista vastuussa olevat organisaatiot, sekä asiakkaat eli loppukäyttäjät. Myös esimerkiksi rahoittaja vaikuttaa projektin onnistumiseen ja on sitä kautta sidosryhmään kuuluva, kuten myös media mainosten välittäjänä.

Projektin kulun päävaiheet ja aktiviteetit antavat kokonaiskuvan, miltä projektin työmäärää näyttää (Kuva 4). Päävaiheiden aikataulutuksessa saattaa projektin aikana ilmentyä päällekkäisyyksiä. Testaus tulokset saattavat olla epäonnistuneita, jolloin joudutaan palaamaan takaisin suunnittelun ja rakentamisen pariin. Käytännössä projektin eteneminen vaatii jatkuvaa testausta koko projektin ajan virheiden välttämiseksi.



Kuva 4. Ohjelmistokehityksen elinkaari (Murch, 2002, s. 61).

5.4 Projektipäällikkö

Projektipäällikön tärkein tehtävä on suorittaa erilaisia projekteja onnistuneesti alusta loppuun. Projekteja voi olla menossa samaan aikaan useitakin, jolloin projektipäällikön tarvitsee hallita niitä kaikkia yhtäaikaisesti ja hahmottaa kokonaisuuksia, sekä etsiä ongelmakohtia. Projektipäällikön tärkein tavoite on välttää yllätyksiä. Yllätykset voivat

aiheuttaa projektin peruuntumisen, budjettien ylittymisen, asiakkaiden tyytymättömyyden tai projektin ulkoistamisen toiselle tekijälle. Pahimmillaan yllätykset siis lopettavat koko projektin, voivat aiheuttaa työttömyyttä ja suuria taloudellisia ongelmia yritykselle. (Murch, 2002, s. 13-14)

Projektipäällikkö laatii business casen eli liiketoimintapäätöksen. Business case on yleisesti käytössä oleva lainasana liikemaailmassa ja se voidaan käsittää liiketoiminnan kehittämiseen tähtäävänä projektiehdotuksena, liiketoimintahankkeena, projektina tai sen osana. Sillä voidaan tarkoittaa myös yksittäistä toimenpidettä, jonka vaikutukset ovat ennalta arvioitavissa lopputulokseen pääsemiseksi. Business case sanalla tarkoitetaan kaikessa yksinkertaisuudessaan tuotteen varmistamista asiakkaalle sellaisena kuin asiakas haluaa ja tuotteen sopivuus täyttää vaaditut tarpeet. Projektipäällikkö seuraa tuotteen etenemistä katselmuksilla ja tarkistuksilla läpi projektin elinkaaren. (Lehtimäki, 2006, s. 7-10)

Projektipäällikön tulee hoitaa tarvittavat sidosryhmä kontaktit kuten työntekijöiden, tilaajan, rahoittajan, johtoryhmän ja kolmannen osapuolen tuottajien neuvottelut. Hänen tulee tiedottaa korkeammalle johdolle ja tilaajalle projektin kulkua päämäärien toteutumisilla ja riskien lieventämisillä. Hänen tulee johtaa projektin henkilöstöä ja luomaan työympäristön, jossa projekti toteutuu määräajassa ja kustannustehokkaasti. Johtajalta vaaditaan ihmissuhde taitojen lisäksi teknistä ymmärrystä, joka yleensä kertyy kokemuksen kautta. On selvää, että johtamistapoja on useita, mutta tärkeintä on saada työporukka puhaltamaan yhteen hiileen ja johtajan tulee tiedostaa ongelmat ja ratkaistava ne esimerkiksi ulkoistamalla projektin osa-alueita, jos tekninen osaaminen ei riitä omassa yrityksessä tai tarjoamalla koulutus tilaisuuksia.

Projektinjohtajan apuvälineenä voi käyttää Microsoft Project-sovellusta, joka mahdollistaa kustannuksien laskemisen eri työvaiheille, tarvittavien työtuntien täyttämisen ja työvaiheiden jaksottamisen projektin eri aikoihin. Projektille voidaan asettaa useampiakin määräaikoja, kuten projektissa graafinen osuus ja ohjelmoinnin osuus saattaa olla eri aikaan valmiita. Projektinjohtaja voi joutua johtamaan erikokoisia ryhmiä, joka asettaa omat haasteensa. Ryhmän kokoamiseksi tarvitsee suorittaa rekrytointi, jonka jälkeen ryhmää täytyy johtaa henkilötasosta aina kokonaiseen ryhmään. Työryhmää täytyy ylläpitää ja motivoida, negatiivinen energia aiheuttaa uupumusta, työtehon laskua ja aikataulujen viivästymisiä.

5.5 Projektin vaiheet

Projekti jaetaan eri päävaiheisiin. Projektin elinkaari alkaa projektisuunnittelusta, myös analysointi, suunnittelu, rakentaminen, testaus ja käyttöönotto luovat projektin vaiheet. Projektissa voi olla käynnissä samaan aikaan useampikin päävaihe eri osa-alueilla esimerkiksi testaus ohjelmiston koodauksessa saattaa olla loppunut, kun graafista ulkoasua vielä rakennetaan. Aktiviteetit ovat päävaiheen tuotoksia, joiden tunnollinen seuraaminen edes auttaa projektin onnistumisessa. (Lehtimäki, 2006, s. 24-28)

5.5.1 Suunnittelu

Suunnittelun aloittamiseen tarvitaan aina taloudellinen perustelu eli business case. Sillä määritellään liiketoiminnan kannattavuus. Kun taloudellinen perustelu on hyväksytty, aloitetaan projektisuunnitelman tekeminen. Hyvässä projektisuunnitelmassa täytyy olla projektin tavoitteet ja projektinhallintaa tukevat osa-alueet tasapainossa. Projektisuunnitelmassa ei ole tarkoitus ilmoittaa kaikkia toiminnallisuuksia, vaan kertoa keskeisimmät toiminta periaatteet. Liian yksityiskohtainen suunnitelma, jokaisesta toiminnosta vaikeuttaa kokonaisuuden ymmärtämistä, jos halutaan projektisuunnitelmaan kaikki yksityiskohdat, on ne hyvä pitää erillään ja vain viitata niihin. Kaikista yksityiskohdista ei olisi edes hyötyä jokaiselle projektin osa-alueelle. Yksityiskohtaisemmat suunnitelmat voidaan lajitella esimerkiksi koodareille, graafikoille, markkinoinnille erikseen mikä koskee mitäkin tekijää. ”Puun runko pitää olla tukeva ja selkeä, mutta lehdet kaunistavat kokonaisuutta”. Hyvä projektisuunnitelma auttaa myös projektin toteutukseen, seurantaan ja arviointiin.

Kun projektin vaiheet ovat jatkuvassa seurannassa voidaan nopealla reagoimisella estää ongelmatilanteet, näin säästytään budjetin ja aikataulun ylityksiltä. Hyvässä suunnitelmassa käydään läpi projektin taustat ja tavoitteet, sekä projektin organisointi, tehtäväkokonaisuuksien aikataulutus ja niiden jakautuminen. Suunnitelmassa tulee hallita myös budjettia eri osa-alueilla, sekä hankintoja. Henkilöstöresurssit tulee ilmoittaa tarkasti, sillä työntekijöiden palkoista kertyy yleensä suurin rahallinen menoerä projektin

kokonaisbudjetista. Siksi täytyisi olla tiedossa projektiin osallistujat, heidän tehtävänsä ja varattujen työtuntien määrä kuhunkin tehtävään.

Projektinhallinnan työkaluina käytetään usein grafiikkana tiedot näyttävää sovellusta kuten Microsoft Projectia. Projektin suunnittelussa ei sovi unohtaa riskienhallintaa, eli sitä miten riskit tunnistetaan ja kuinka niitä hallitaan. Kun projektisuunnitelma hyväksytään, aloitetaan rakennusvaihe. Tulee myös tilanteita, joissa joudutaan muokkaamaan projektisuunnitelmaa myöhemmissä vaiheissa ja näihinkin tilanteisiin olisi hyvä varautua.

Suunnittelu asiakasrekisterissä alkoi mindmapilla eli ideakaaviolla, Suunnittelussa mietittiin mitä ominaisuuksia asiakasrekisterissä tarvitaan ja aikataulutuksessa mietittiin minkä verran eri osa-alueet vaativat aikaa, kuten graafinen suunnittelu ja koodaaminen. Suunnittelun vaikeutena oli kokemuksen puute, joka vaikeutti eri osa-alueiden aikataulutusta. Aikataulun arviointi oli hankalaa tietämättömyyden vuoksi.

5.5.2 Rakentaminen

Rakennusvaiheen avaintavoitteet ovat moduulisuunnittelun valmistelu ja tekeminen, komponenttien integrointi moduulien välisten yhteyksien testaus eli jonotestaaminen. Moduulisuunnittelu on projektin jakamista useampaan eri yksikköön eli erillään olevaan osa kokonaisuuteen, jotka voidaan myöhemmin integroida kokonaisprojektiin. Moduulisuunnittelun jokainen erillinen tehtävä tulisi dokumentoida ja käydä asianomaisien kanssa, kuten ohjelmoijien ja graafikoiden. Näin saadaan yhteinen kokonaiskuva projektin osasta. Rakennusvaiheessa noudatetaan tarkasti valmiita suunnitelmia, ja pyritään välttämään muutoksia, joista voi aiheutua lisäkustannuksia. Onkin ymmärrettävää, että rakennusvaiheessa toteutetaan suunniteltu kokonaisuus. Aina ei voida toteuttaa projektin kannalta parasta vaihtoehtoa, jos täytyy noudattaa suunnitelmaa. Suunnitelmasta poikkeaminen lisää välittömästi riskejä. Esimerkiksi asiakasrekisteriin haluttiin myös hallintasivulle mahdollisuus muokata käyttäjätietoja, mutta aikataulutuksellisesti se jätettiin toteuttamatta.

Asiakasrekisterin teossa opiskelija toimi suunnittelijana, ohjelmoijana, projektin johtona, testaajana ja teknisenä arkkitehtina esimerkiksi tietokanta suunnittelussa. Suuremmissa

yrityksissä näihin tehtäviin on varattu huomattavasti suurempi henkilöstö kapasiteetti, jonka ansiosta projektin aikataulutus nopeutuu merkittävästi.

5.5.3 Testaus

Testauksen suunnittelulla määritetään, mitä testejä vaaditaan laadukkaan tuotteen tuottamiseksi. Testaamista varten tarvitsee luoda tarkat testaussuunnitelmat, joita noudatetaan. Testausvaiheessa usein ilmentyy kriittisten polkujen aktiviteetit, jos suunnitelmia ei ole tarkoin noudatettu. Kriittisen polun aktiviteetilla tarkoitetaan yhtä osa-aluetta, jonka epäonnistuminen aiheuttaa ketjureaktion koko projektin kulkuun esimerkiksi, jos ohjelmoinnista löytyy suuri virhe toiminnan kannalta, joudutaan palaamaan takaisin lähtöruutuun ja pahimmillaan aloittamaan alusta. Vaikutus voi myös näkyä muissa tehtäväalueissa, kuten ulkoasusuunnittelussa tai sisällöntuotannossa. Testauksen suunnittelu- ja valmistusvaiheen rooleja ovat testisuunnittelija, asiakas, suunnittelija ja testaajat.

Integrointitestauksella varmistetaan sovelluksen sulautuminen muihin järjestelmiin ja sovelluksiin. Sillä poistetaan mahdolliset epäsopivuudet sovellusten kesken. Vaikka sovellus toimisi yksinään, voi sen käyttö haitata muita sovelluksia. Integraatiotestauksen pois jättäminen on valitettavan yleistä, koska budjetti ylittyy herkästi ja vastuu toimivuudesta voi olla toisella osapuolella esimerkiksi asiakasrekisteri ei toimi toisen yrityksen tekemän kotisivun sisällä. Tällöin vastuun ottaminen ei olekaan niin selvää ja herää kysymys: Kuka maksaa integraation suorittamisesta. Hyväksymistestaus simuloi oikeaa toiminta tilannetta ja siinä testataan lopullinen versio. Testitulosten yksityiskohtien tarkastus on myös syytä toteuttaa. Yksi testitulos ei aina kerro koko totuutta ja siksi useampaan kertaan testattu tuote löytää varmemmin ongelmakohdat.

Asikasrekisterissä testaus tapahtui läpi koko projektin. Jokaisen toiminnon valmistuessa, täytyi ne testata käytännössä ja todeta toimiviksi. Koodauksen toteutuksessa tarvittiin erityistä tarkkaavaisuutta, koska osaaminen PHP-koodista oli vähäistä ja virheiden määrä oli suuri. Virheiden etsiminen testausvaiheessa oli työlästä, mutta mitä useammin testaus tehtiin sen pienempi oli tutkittava kohde ja virheen löytyminen todennäköisempää.

5.5.4 Käyttöönotto

Käyttöönotossa varmistetaan sovelluksen valmistelutoimet loppukäyttäjille julkaistaviksi. Tarkoituksena on, että käyttöönotto sujuu ongelmitta ja suunnitellun aikataulun mukaisesti. Tuotteen korvaaminen toisella tai päivittäminen täytyy varmistaa käyttöönotto- ja muunnossuunnitelmalla, koska sovelluksen siirtäminen ja muunnokset voivat aiheuttaa odottamattomia ongelmia. Käyttöönotossa käytetyissä suunnitelmissa olisi aina hyvä nimetä vastuuhenkilöt. Suurin osa muunnossuunnitelmasta liittyy tiedostojen ja tietokantojen luomiseen, päivittämiseen ja ylläpitoon. Varmuuskopiot ja varasuunnitelmat ovat myös hyvä sisällyttää suunnitelmiin. Käyttöönottosuunnitelmassa kirjataan kaikki käyttöönottoon liittyvät aihepiirit kuten sovelluksen julkaisemisen aikataulutus, sovelluksen mahdollinen koulutus ja niin edelleen.

Käyttöönotto asiakasrekisterissä tapahtui paikallisen palvelin sovelluksen avulla ja tulevaisuudessa asiakasrekisteri tulee olemaan yhtenä osana uuden yrityksen palveluja.

6 YHTEENVETO JA POHDINTA

Projektin suunnittelu lähti ideasta perustaa oma ohjelmisto ja graafisenalan yritys. Projektissa työstettiin sähköinen asiakasrekisteri kauneudenhoito alan yrityksille, samalla haluttiin selvittää projektinhallinnan keinot suorittaa projekti onnistuneesti alusta loppuun. Pilvipalveluna toimiva asiakasrekisteri palvelee asiakkaita ja he voivat rauhassa täyttää lomakkeen, vaikka kotisohvalla.

Projektissa käytettiin selainpuolelle tuttuja teknologioita. HTML-kielellä rakennettiin lomakkeen runko ja CSS-tyylimäärittelyillä saatiin haluttu ulkoasu. Ulkoasusuunnittelussa ei ollut erityisiä vaatimuksia, ja siksi päädyttiin käyttämään vihreitä sävyjä, jotka kuvastavat luontoa ja kasvien väriä. Väripsykologiassa vihreällä on rauhoittava vaikutus ja se koetaan miellyttäväksi (AskelTerveyteen, 2018). PHP-kieltä käytettiin palvelinpuolen ohjelmointiin ja rakennettiin tietokantayhteydet MySQL-tietokantaa käyttäen.

Opinnäytetyössä käytiin läpi käytetyt teknologiat ja projektinhallinta. Projektinhallinta on tärkeä osa projektin läpiviemisessä, ja ilman sen osaamista projektin todennäköisyys onnistua on marginaalisen pieni. Projektinvaiheet kerrottiin yksityiskohtaisesti aina suunnittelusta loppukäyttäjälle, jotta voitaisiin viedä opinnäytetyö maaliin projektinhallinnan keinoin.

Opinnäytetyön aikataulutus viivästyi, ja syyksi tuli yllättävä korona pandemia, joka antoi lisähaasteita loppuun saattamiseen. Työssä oli ongelmia, joita olisi voinut tehdä toisin. Projekti sekä sen aikataulutus ja eteneminen olisi ollut hyvä suunnitella tarkemmin. Korona-virus oli toisaalta hyvä esimerkki riskienhallinnasta, jolloin riskienhallinnasta ja siitä, kuinka yllättäviinkin riskeihin tulisi varautua ja kehittää varasuunnitelma.

Projektissa oppi kuitenkin paljon ohjelmoinnista, vaikka asiakasrekisteri olisikin järkevämpi ja helpompi toteuttaa CMS-julkaisujärjestelmällä, jossa on valmiina käyttäjien hallinta esimerkiksi Joomlalla.

LÄHTEET

Aaltokangas, T. (24. 4 2019). *CMS-suunnitteluopas – Osa 7: Responsiivisuus*. Viitattu: 2.1.2021. Osoitteessa: https://www.thuledigital.fi/blogi/2019/cms-suunnitteluopas-osa-7-responsiivisuus/.

AskelTerveyteen. (7. 11 2018). *Värien psykologia*. Viitattu: 2.3.2021. Osoitteessa: https://askelterveyteen.com/varien-psykologia/.

CGI. (2021). *Pilvipalvelut*. Viitattu: 2.3.2021. Osoitteessa: https://www.cgi.com/fi/fi/pilvipalvelut.

Data, R. (1998). *w3schools*. Viitattu 14.2.2016. Osoitteessa: https://www.w3schools.com/html/html_intro.asp.

EMG – Educations Media Group. (27. 1 2021). *Mitä ovat ketterät menetelmät? – Scrum, Lean ja muut tutuksi*. Viitattu: 2.2.2021. Osoitteessa: https://www.koulutus.fi/oppaat/projektinhallinta/ketteratmenetelmat-19939.

HOLOPAINEN, S. (28. 3 2019). *almatalent.fi*. Viitattu: 3.3.2021. Osoitteessa: https://www.almatalent.fi/tietopalvelut/blogi/miten-ja-miksi-yllapitaisin-asiakasrekisteriani.

Howe, S. (2014). *Learn to code HTML & CSS: Develop & Style Websites.* Yhdysvallat: New Riders.

Juha Taina. (2009). Helsingin Yliopisto. *Ohjelmistoprosessit ja ohjelmistojen laatu.* Viitattu: 2.1.2021. Osoitteessa: https://www.cs.helsinki.fi/u/taina/opol/k-2009/pdf/luku-6_2.pdf.

Kaukoniemi, J. (2000). *RGB- ja CMYK-väriavaruudet (gamut)*. Viitattu 10.2.2021. Osoitteessa: http://www.volantis.fi/sivut/color-frame.html.

Koulutuksen tutkimuslaitos. (2016). *HTML head sivu*. Viitattu: 4.2.2021. Osoitteessa: https://peda.net/p/jamspe/html5/24-html-head-sivu.

Koulutus- ja konsultointipalvelu KK Mediat. (2000-2021). *Johdatus PHP-ohjelmointiin*. Viitattu: 20.3.2021. Osoitteessa: https://www.2kmediat.com/php/johdanto.asp.

Laaksonen, A. (2020). *Tietokantojen perusteet*. Viitattu: 1.2.2021. Osoitteessa: https://tikape-k20.mooc.fi/luku-4/1.

Lehtimäki, T. (2006). *Ohjelmistoprojektit käytännössä.* Jyväskylä: Gummers Kirjapaino Oy.

Linkola, J. (4. 7 2021). *WWW-julkaisemisen perusteet – CSS*. Viitattu: 23.3.2021. Osoitteessa: https://jml.kapsi.fi/jussi/www-julkaisemisen-perusteet/www-julkaiseminen-css/.

Microsoft. ASCII- ja Unicode Latin -pohjaisten symbolien ja merkkien lisääminen. Viitattu: 1.5.2021. Osoitteessa: https://support.microsoft.com/fi-fi/topic/ascii-ja-unicode-latin-pohjaisten-symbolien-ja-merkkien-lis%C3%A4%C3%A4minen-d13f58d3-7bcb-44a7-a4d5-972ee12e50e0.

Murch, R. (2002). *IT-projektinhallinta.* Helsinki: IT Press.

Pyry ehdonvirta, J. K. (2013). *HTML5 sovellusalustana.* Helsinki: RPS-yhtiöt.

Salo, I. (2010). *Cloud Computing.* Jyväskylä: WSOYpro Oy.

TechTarget. (2021). History of my MySQL. Viitattu: 2.2.2021. Osoitteessa: https://www.datasciencecentral.com/profiles/blogs/history-of-mysql.

Tiirikainen, V. (2010). *It ja parempi bisnes.* Helsinki: Talentum Oyj.

Toivonen, T. (30. 7 2019). *Yrittäjän tietosuojaopas.* Viitattu: 2.3.2021. Osoitteessa: https://www.yrittajat.fi/yrittajan-abc/yritystoiminnan-abc/yrittajan-tietosuojaopas-570864.

Tutorial Republic. (2021). *PHP Tutorial*. Viitattu 20.3.2021. Osoitteessa: https://www.tutorialrepublic.com/php-tutorial/.

Tutorials Point. (2015). Javascript: Javascript language.