Verkkopalvelun suunnittelu ja kehitys

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulututkinnon opinnäytetyö

Tietojenkäsittelyn koulutus, Hämeenlinnan korkeakoulukeskus
kevät, 2021

Eero Salminen

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tietojenkäsittelyn koulutus Tiivistelmä

Hämeenlinnan korkeakoulukeskus

Tekijä Eero Salminen Vuosi 2021

Työn nimi Verkkopalvelun suunnittelu ja toteutus

Ohjaajat Mirlinda Kosova-Alija

TIIVISTELMÄ

Tämän opinnäytetyön tavoitteena oli suunnitella ja toteuttaa Angular-ohjelmistokehykseen pohjautuva yhden sivun verkkopalvelu. Tavoitteena oli tutustua aluksi yleisesti verkko-ohjelmointiin sekä Angular-ohjelmistokehykseen. Opinnäytetyössä tarkastellaan nettipalvelun suunnittelua sekä käsitellään palveluiden saavutettavuusohjeistuksia. Työssä käydään läpi myös opinnäytetyöprojektin aikana toteutettu Angular-verkkopalvelu ja sitä varten toteutetut taustajärjestelmät. Opinnäytetyöllä ei ole toimeksiantajaa, mutta kyseessä on toiminnallinen opinnäytetyöprojekti.

Opinnäytetyön teoriaosuudessa käsitellään työn kannalta keskeisimmät käsitteet. Aluksi käsitellään verkkopalveluiden ohjelmointia, jonka jälkeen tarkastellaan yleisiä verkkopalveluiden suunnitteluperiaatteita. Teoriaosuudessa tutustutaan lisäksi ohjelmoinnissa käytettäviin työkaluihin sekä ohjelmointikieliin.

Opinnäytetyön tuloksena saatiin toimiva Angular-ohjelmistokehykseen perustuva verkkopalvelun prototyyppiversio sekä sitä varten toteutettu käyttöliittymäsuunnitelma ja taustajärjestelmät. Verkkopalvelun kehitystyön perusteella Angular todettiin toimivaksi ohjelmistokehykseksi yksinkertaisille verkkopalveluille sen järjestelmällisen sekä ymmärrettävän kokonaisuuden osalta. Verkkopalvelun prototyyppiversiossa on toteutettuna kolme palvelun kannalta keskeisintä toimintoa, joiden pohjalta palvelua on helppo jatkokehittää tulevaisuudessa.

Avainsanat Ohjelmointi, Verkkopalvelu, Angular

Sivut 32 sivua ja liitteitä 1 sivua

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology
Hämeenlinna University Centre

Abstract

Author Eero Salminen
Year 2021
Subject Webservice design and development
Supervisors Mirlinda Kosova-Alija

ABSTRACT

The purpose of the thesis was to design and implement a single page web service based on the Angular framework. The aim was to get acquainted with web programming and the Angular framework in general. The thesis examines the design of the web service and explores the accessibility guidelines for web services. The thesis also explores the Angular web service and its background systems implemented during the thesis project. The thesis does not have a commissioner, but it is a practical thesis project.

The theoretical part of the thesis includes the most important related concepts for the project. At the beginning of theses go through the web service programming and general web service design principles. The theoretical part also introduces the tools used for programming and some programming languages.

The result of the thesis was a working prototype version of the web service based on the Angular software framework, as well as the user interface plan and background systems for the web service. Based on the development process of the web service, Angular was found to be a functional software framework for simple web services because it was systematic and easy to understand. The prototype version of the web service contains three of the most important functions for the service. In the future, the prototype version will be easy to develop further.

Keywords Programming, Webservice, Angular
Pages 32 pages and appendices 1 pages

Sanasto

Figma	Figma on pilvipohjainen verkkopalveluiden käyttöliittymien suunnittelutyökalu.
MySQL	MySQL on avoimen lähdekoodin relaatiotietokantojen hallintajärjestelmä.
Angular	Angular on TypeScript-pohjainen avoimen lähdekoodin ohjelmistokehys (framework).
Angular CLI	Angular CLI on komentorivikäyttöliittymä, jota käytetään Angular-projektien ylläpidossa.
HTML	Lyhenne sanoista Hypertext Markup Language, jolla kuvataan WWW-sivujen rakenne.
CSS	CSS on lyhenne sanoista Cascading Style Sheet ja sitä hyödynnetään nettisivun tyylittelyssä.
JavaScript	JavaScript on skriptauskieli, jota hyödynnetään monesti front-end-ohjelmoinnissa HTML:n kaverina.
TypeScript	TypeScript on avoimen lähdekoodin kieli, joka perustuu JavaScriptiin, ja TypeScriptiin on lisätty staattisia vahvoja tyypityksiä.
PHP	PHP on yleisesti käytetty ohjelmointikieli palvelinympäristöissä dynaamisia web-palveluita kehitettäessä.
Front-end	Front-end-ohjelmoinnissa kehitetään verkkopalvelun tai sovelluksen käyttöliittymää.
Back-end	Back-end-ohjelmoinnissa kehitetään taustapalveluita, kuten palvelinpuolta.
Full-stack	Full-stack-ohjelmoinnissa kehittäjä työstää niin käyttöliittymää kuin myös taustapalveluita.

Sisällys

Kuvat, ohjelmakoodit ja taulukot

Liitteet

1 Johdanto

Opinnäytetyön ideana on suunnitella ja ohjelmoida palvelun prototyyppiversio, jossa loppukäyttäjä pystyy tarkastelemaan tuotteiden ominaisuuksia sekä vertailemaan eri tuotteita keskenään. Sivuston tavoitteena on olla mahdollisimman käyttäjäystävällinen, tavoitettava sekä käytettävissä erilaisilla laitteilla, kuten tietokoneella tai mobiililaitteilla. Loppukäyttäjälle näkyvän sivuston ohella opinnäytetyössä on tavoitteena kehittää taustapalvelut sekä palvelu, jossa ylläpitäjä voi lisätä uusia tuotteita tietokantaan.

Sivuston front-end-ohjelmointi tapahtui hyödyntämällä Angular-ohjelmistokehystä ja back-end-ohjelmoinnissa hyödynnettiin PHP-ohjelmointikieltä. Ohjelmointiprojektin lopullisena tavoitteena oli tuottaa palvelu, joka helpottaa käyttäjiä löytämään paremmin tietoja tuotteista sekä vertaamaan niitä ennen ostopäätöstä. Opinnäytetyössä käsitellään projektissa käytettäviä ohjelmointitekniikoita sekä käydään läpi lopullinen toteutettu palvelu sekä sen käyttötoimintaperiaate.

Opinnäytetyössä vastataan neljään tutkimuskysymykseen, jotka ovat esiteltynä alapuolella. Tavoitteena oli opinnäytetyön aikana saada parempi käsitys Angularin toimivuudesta tuotevertailusivuston luomisessa sekä saada toteutettua suunnitelma, miten kyseinen sivusto kannattaisi suunnitella käyttöliittymän kannalta.

- Mikä palvelu sopisi käyttöliittymäsuunnitelman toteuttamiseen?
- Kuinka toimiva Angular on front-end-ohjelmoinnissa?
- Millaiset taustapalvelut vaaditaan palvelun toteuttamisiksi?
- Mitkä ohjelmointikielet ja ohjelmistokehykset sopivat palvelun kehittämiseen?

2 Web-ohjelmointi

Ohjelmointi on laaja käsite, joka pitää sisällänsä kaikenlaiset ohjelmointikielet sekä rajapinnat. Ohjelmointi on tapa, jolla annetaan tietokoneelle käsky suorittaa jokin tehtävä. Ohjelmointikieliä on olemassa monia, muun muassa C#, PHP, Python ja Java. Ohjelmointi voidaan jakaa moniin eri alalajeihinsa ja tässä opinnäytetyössä on keskitytty web-ohjelmointiin, mikä tarkoittaa nettisivujen ohjelmointia. Nettisivut voivat olla kaikkien nähtävillä internetissä tai esimerkiksi organisaation työntekijöiden sisäisessä käytössä oleva osa palvelua. (Host Shopper, n.d.)

2.1 Front-end-, back-end- ja full-stack-ohjelmoinnit

Web-ohjelmoinnissa on kaksi laajaa osa-aluetta, joita kutsutaan nimillä front-end- ja back-end-ohjelmointi. Front-end-ohjelmoinnissa kehitetään verkkopalvelun tai sovelluksen käyttöliittymää ja back-end-ohjelmoinnissa kehitetään taustapalveluita, kuten palvelinpuolta. Lisäksi kolmas tärkeä osa-alue on full-stack-kehitys, jossa kehittäjä työstää niin käyttöliittymää kuin myös taustapalveluita. (Laine Tuulikki, 2015)

Front-end-ohjelmointia tekevä henkilö vastaa käyttöliittymän suunnittelemisesta sekä kehittämisestä. Kyseessä on verkkopalvelun tai sivun ulkoasu, joka näkyy käyttäjälle. Front-end-ohjelmointiin keskittyvällä henkilöllä on hyvä olla perustiedot ja ymmärrys kolmesta osa-alueesta, jotka ovat HTML, CSS ja JavaScript. HTML on merkintäkieli, jolla kuvataan www-sivujen rakenne. HTML:llä ei voida määrittää sivujen ulkoasua tarkasti, vaan sillä saadaan aikaiseksi rakenne, joka koostuu esimerkiksi otsikoista tai kappaleiden rajoista. CSS on tyyliohjeiden laji, jolla pystytään määrittelemään tarkemmin ulkoasun rakenne. JavaScript on ohjelmointikieli, joka lisää web-palveluiden ja nettisivujen dynaamista toimintaa (Laine Tuulikki, 2015).

Back-end-ohjelmoinnissa hallitaan, mitä tapahtuu verkkopalvelun taustalla, eikä käyttäjät näe taustalla tehtyä työtä. Back-end-ohjelmoijan tehtävänä on toteuttaa koodit, jotka kommunikoivat tietokannan tietojen kanssa ja näyttää niitä selaimessa. Yleisesti back-end-ohjelmointia suoritetaan lukuisilla ohjelmointikielillä, joita ovat muun muassa PHP sekä Python. PHP on ohjelmointikieli, joka on hyvin suosittu palvelinympäristöissä web-

ohjelmoinnissa. PHP-ohjelmointikoodi suoritetaan jo palvelimen puolella, ja sen avulla määritetään mitä tietoja lähetetään nettisivulle. Python on yksinkertainen, helposti ymmärrettävä ohjelmointikieli. (GeeksforGeeks, 2021)

Full-stack-ohjelmoinnissa täytyy olla perehtynyt molempiin aikaisemmin mainittuihin ohjelmoinnin osa-alueisiin. Full-stack-ohjelmoija osaa toteuttaa back-end-palvelinpuolen ohjelmointia sekä käyttöliittymän toteutetusta front-end-ohjelmoinnissa. Full-stack-ohjelmoija pystyy kattavalla ammattitaidollaan toteuttamaan kokonaisen verkkosivun alusta loppuun. Tällaiset ohjelmoijat ovat työskennelleet vuosia erilaisissa rooleissa, ja he perehtyvät verkkosivuihin kokonaisuutena ja ovat myös kiinnostuneita ohjelmoinnista. (Eric, 2021)

Koska full-stack-ohjelmoijalla on usein kattava osaaminen niin back-end- kuin front-end-ohjelmoinnista, niin kyseessä on erittäin arvokas henkilö yritykselle tai tiimille. Full-stack-ohjelmoija ymmärtää molemmat puolet kehityksestä sekä pystyy keskustelemaan ja yhdistämään prosessit. Samalla tällainen ohjelmoija osaa arvioida ja kommunikoida helposti muiden henkilöiden kanssa siitä, miltä verkkosivun pitäisi näyttää ja kuinka sen pitäisi toimia, sillä hän tietää back-end-ohjelmoinnin tekniset rajoitukset, jotka saattavat vaikuttaa front-end-ohjelmointiin. (Eric, 2021)

3 Verkkopalvelun käyttöliittymäsuunnittelu ja käytetyt työkalut

Verkkopalveluiden sekä sivustojen suunnitteluun on tarjolla monenlaisia työkaluja sekä suuntaa antavia ohjeistuksia. Palveluiden käyttöliittymää suunnitellessa pyritään ottamaan huomioon eri tahojen suosituksia, joiden perusteella on tavoitteena tuottaa mahdollisimman positiivinen sekä saavutettava käyttökokemus. Ohjeistuksia on hyvä käyttää hyödyksi muun muassa arvioitaessa käyttöliittymän intuitiivisuutta, opettavuutta, tehokkuutta sekä johdonmukaisuutta. (Interaction Design Foundation, n.d.)

Yleisesti käyttöliittymän suunnittelu jaetaan kuuteen kokonaisuuteen, jotka ovat tyyli, asettelu, käyttöliittymän komponentit, tekstit, esteettömyys sekä suunnittelumallit. Tyyli kattaa tuotemerkin logot sekä yleisen värimaailman. Asettelussa otetaan huomioon käyttöliittymän rakenne, onko toteutus ruudukkomainen vai luettelomainen. Käyttöliittymän komponenteissa otetaan huomioon esimerkiksi valikot ja painikkeet, jotta ne olivat yhdenmukaisia koko sivustolla. Tekstien fontti ja sävyt pitäisivät olla myös yhdenmukaiset muun käyttöliittymän toteutuksen kanssa. Esteettömyys on yksi käyttöliittymän kulmakivistä, jotta käyttöliittymä olisi suunniteltu erilaisille ihmisille käytettäväksi. Suunnittelumallit pitävät sisällänsä muun muassa lomakkeet, joiden pitäisi olla myös yhdenmukaiset käyttöliittymän kanssa. (Interaction Design Foundation, n.d.)

3.1 Verkkopalveluiden saavutettavuus

Verkkopalveluita varten on olemassa yksi tärkeä ohjeistus, joka kantaa nimeä Web Content Accessibility Guidelines tai lyhyemmin WCAG. Suomeksi kyseessä on Verkkosisällön saavutettavuusohjeet, ja kyseessä on kansainvälinen ohjeistus verkkosisältöjen saavutettavuudesta. WCAG on luotu varmistamaan, että vammaiset tai toimintarajoitteiset ihmiset pystyisivät hyödyntämään verkkopalveluita. (W3, n.d.-b)

Verkkosisällön saavutettavuusohjeet julkaistiin alun perin vuonna 1999, mutta teknologioiden kehittymisen takia sitä päivitetään tasaisin väliajoin. Vuonna 2008 ohjeet saivat WCAG 2.0 -version ja vuonna 2018 hyväksyttiin WCAG 2.1 -versio. Verkkosisällön saavutettavuusohjeiden kehitys jatkuu edelleen ja uusi versio on kehitteillä. Ohjeita kehittää

kansainvälinen World Wide Web -yhteenliittymä, mikä tunnetaan lyhyemmin myös nimellä W3C. (W3, n.d.-b)

Verkkosisällön saavutettavuusohjeet ovat rakennettu eri tasoille, ja kaikista korkeimmalla ovat periaatteet kuten havaittavuus, hallittavuus, ymmärrettävyys ja toimintavarmuus. Periaatteiden alaisuudessa on 13 ohjetta, joissa määritellään tarkemmin mitä ne sisältävät ja mitkä ovat niiden tavoitteet. Lisäksi ohjeissa on kriteereitä, jotka ovat myös jaettu kolmeen eri tasoon, jotka ovat A-, AA- ja AAA-tasot. Tiukimmat kriteerit ovat AAA-tasolla, jossa verkkopalvelun täytyy olla saavutettavissa laajalle yleisölle, ja huomioon on otettu vammaiset ja toimintarajoitteiset ihmiset. Suomessa on myös digitaalisten palveluiden laki, jonka mukaan julkisien toimijoiden pitää tarjota verkkopalvelut niin, että ne täyttävät ainakin A- ja AA-tason kriteerit. (Aluehallintovirasto, n.d.)

WCAG:ssa on määritetty muun muassa kuinka verkkopalvelussa kuuluisi tekstit esitettävän, tarjota äänitiedostot myös tekstiversiona sekä esimerkiksi, miten värejä käytetään palvelussa. Ohjeistuksen tavoitteena on tehdä verkkosivuista mahdollisimman saavutettavat ottaen huomioon erilaiset ihmiset ja heidän tarpeensa. (Aluehallintovirasto, n.d.)

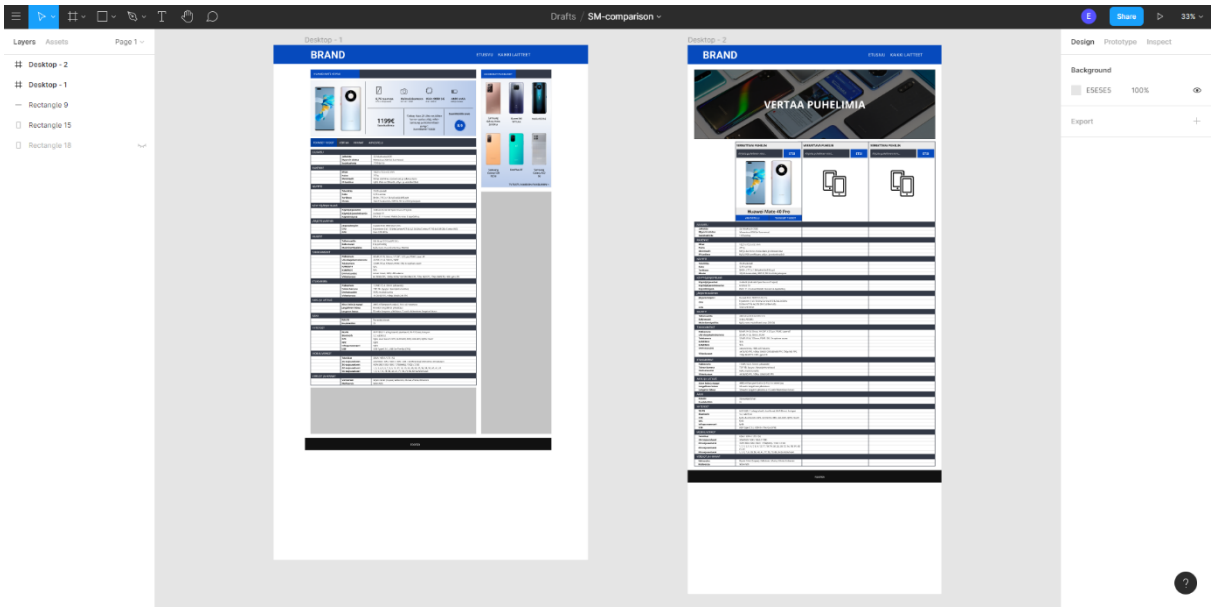
3.2 Figma

Figma on käyttöliittymäsuunnitteluun kehitetty pilvipohjainen työkalu, jota voidaan käyttää selaimella, mutta työkalusta on olemassa myös työpöytäsovellus. Pilvipohjaisuus tarkoittaa, että kyseessä on palvelu, jossa tiedot pysyvät pilvessä, ja niihin päästään käsiksi kaikkien laitteiden kautta. Figma perustettiin vuonna 2015, ja työkalua on kehitetty vahvasti yhteistyössä suunnittelijoiden kanssa. (Figma, n.d.-a)

Figmalla on mahdollista suunnitella erilaisia käyttöliittymiä muun muassa verkkosivuista mobiilisovelluksiin. Figman yksi ominaisuuksista on tiimityöskentely, eli monet käyttäjät voivat tarkastella ja muokata käyttöliittymäsuunnitelmaa samanaikaisesti. Lisäksi tiimityöskentely mahdollistaa reaaliaikaisen palautteen antamisen muiden suunnittelijoiden kanssa. Figma antaa yksinkertaisen työskentely-ympäristön kattavien sekä yksityiskohtaisten käyttöliittymäsuunnitelmien toteuttamiseen. Figma-työkalun selkeä työympäristön käyttöliittymä on esillä kuvassa 1. (Figma, n.d.-a)

Hinnoittelultaan Figma on oletusarvoisesti ilmainen tiettyyn pisteeseen asti, joten käyttöliittymien suunnittelu pienessä tiimissä onnistuu ilmaisversiolla. Figmasta on tarjolla myös kaksi maksullista versiota, joiden hinnat ovat 12 dollaria (professional) sekä 45 dollaria (organization). Maksullisen versiot tarjoavat lisäominaisuuksia, kuten rajoittamattomat projektit, rajoittamattoman versiohistorian sekä muita kattavampia ominaisuuksia. (Figma, n.d.-b)

Kuva 1 Figman käyttöliittymä.



4 Ohjelmoinnissa käytetyt työkalut

Ohjelmointityökalut ovat avainasemassa verkkopalveluiden kehittämisessä, ja tarvittaessa käytössä täytyy olla myös työkaluja tietylle ohjelmointikielelle. Opinnäytetyössä hyödynnettiin pääasiallisesti työkaluna Microsoftin kehittämiä palveluita, kuten Visual Studiota sekä Visual Studio Codea. Lisäksi olemassa on myös kattava valikoima muita työkaluja, joissa koodia voidaan kirjoittaa sekä muokata. Tässä opinnäytetyössä nämä työkalut valikoituivat käytettäväksi niiden yleisyyden sekä helppokäyttöisyyden ansiosta.

4.1 Visual Studio ja Visual Studio Code

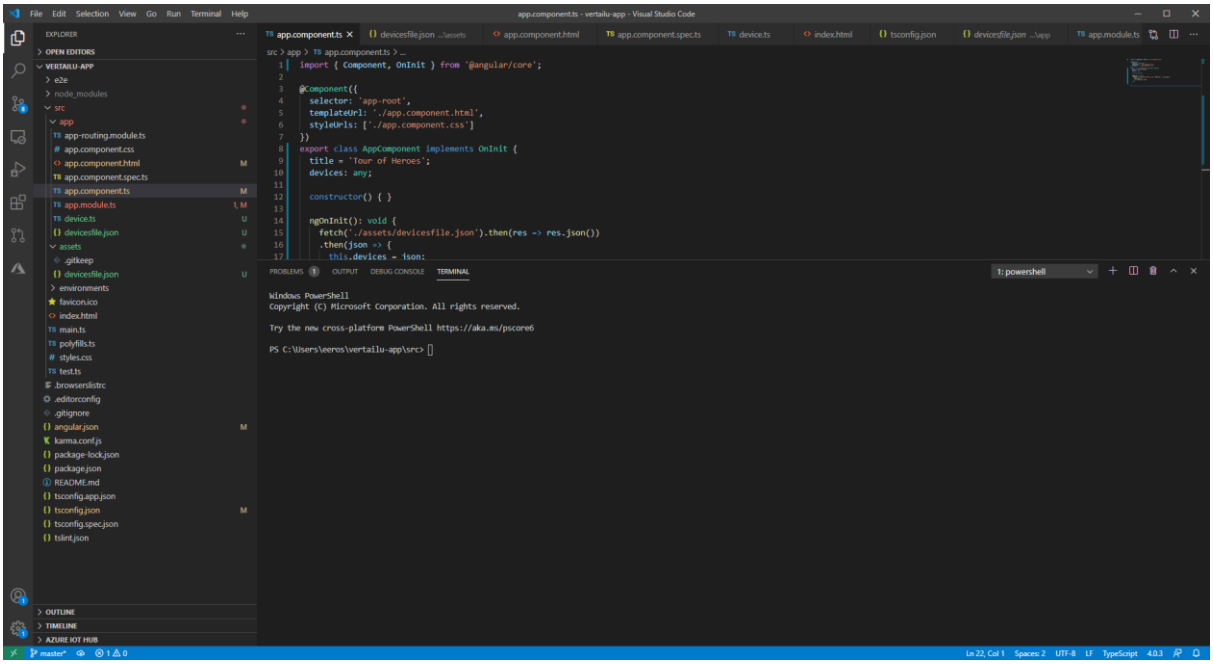
Visual Studio on yhdysvaltalaisen ohjelmisto- ja laitteistoyritys Microsoftin ohjelmointityökalu. Visual Studio tarjoaa monipuoliset ominaisuudet lukuisien ohjelmointikielien ohjelmointiin, ja sen avustuksella pystytään kirjoittamaan koodia nopeasti sekä diagnosoimaan koodia helposti. Visual Studio on saatavilla useassa versiossa sekä monille käyttöjärjestelmille, joten ohjelmointia voidaan tehdä useilla laitteilla. (GeeksforGeeks, 2019)

Microsoft julkaisi Visual Studio -työkalun ensimmäisen kerran vuonna 1997. Visual Studio on helpottanut kehittäjiä ohjelmoinnissa, ja työkalusta on julkaisu muutamien vuosien välein uusia versioita, joihin Microsoft on lisännyt aina uusia ominaisuuksia sekä tukia uusille ohjelmointikielille. Tuorein versio on Visual Studio 2019, joka julkaistiin vuonna 2019, ja kyseistä versiota hyödynnettiin tämän opinnäytetyöprojektin toteuttamisessa. (GeeksforGeeks, 2019)

Visual Studio Code on Microsoftin kehittämä ilmainen ja avoimen koodin koodieditori, joka on saatavilla monille käyttöjärjestelmille, kuten Windowsille, Linuxille ja macOS:lle. Kyseessä on erittäin suosittu ohjelmointiympäristö, sillä kyseessä on kevytkäyttöinen sovellus, mutta siinä on silti monipuolisia sekä hyödyllisiä toiminnallisuuksia. Visual Studio Code tukee kattavasti ohjelmointikieliä, joita ovat muun muassa Java, C++, Python, CSS sekä monet muut. (Anusheh, n.d.)

Käyttöliittymältään Visual Studio Code on hyvin yksinkertainen ja se on jaettu viiteen pääosaan, joita ovat toimintapalkki, sivupalkki, editointiryhmät, paneeli sekä tilarivi. Ohjelman yksinkertainen käyttöliittymä on esillä alapuolella olevassa kuvassa 2.

Kuva 2 Visual Studio Coden helppokäyttöinen käyttöliittymä.



5 Hyödynnetyt ohjelmointi- ja merkintäkielet

Opinnäytetyöprojektin aikana hyödynnettiin muutamia keskeisiä tekniikoita ja ohjelmointikieliä, joista päällimmäisenä on avoimen lähdekoodin Typescrip-pohjainen Angular-ohjelmistokehys. Lisäksi verkkopalvelun toteutukseen käytettiin muita tekniikoita, joita ovat muun muassa HTML, CSS ja PHP. Tässä kappaleessa tutustumme syvällisemmin Angulariin sekä muihin keskeisiin ohjelmointikieliin ja kehyksiin.

5.1 Angular

Angular on TypeScript-pohjainen ohjelmistokehys, joka pohjautuu avoimeen lähdekoodiin ja sen avulla pystytään toteuttamaan yhden sivun sovelluksia (Single Page Applications). Angular on yhdysvaltalaisen ohjelmistoyritys Googlen ylläpitämä selainpohjaisiin sovelluksiin suunnattu ohjelmistokehys. Angular julkaistiin jo vuonna 2009, ja vuonna 2019 kyseessä oli Stackoverflow-kyselyssä peräti kolmanneksi eniten käytetty framework. (Stack Overflow, n.d.)

Alun perin Angularia kutsuttiin nimellä AngularJS, mutta nykyisin Angular 2 -versiosta ylöspäin kyseessä on vain Angular. Angularilla pystytään rakentamaan sovelluksia hyödyntäen tekniikoita, kuten HTML, CSS, JavaScript ja TypeScript. Angularin valttikortteja ovat muun muassa sovelluksien puhdas, helposti ylläpidettävä sekä ymmärrettävä rakenne. (Ankit, 2019)

5.1.1 Single Page Application (SPA)

Angularin arkkitehtuuri on toteutettu Single Page Application (SPA) -toteutuksella, mikä tarkoittaa kyseessä olevan yhden sivun sovellus. Angularin kohdalla pystytään toteuttamaan sovelluksia, joissa selaimen ei tarvitse ladata sivua uudestaan sitä käytettäessä. Yhden sivun sovellukset renderöivät palvelun osia käyttäjän toiminnan mukaisesti. Sivua ei ladata joka kerralla uudestaan, vaan ainoastaan sivun sisältö haetaan uudelleen. (Angular University Blog, n.d.)

Yhden sivun sovelluksissa on ainoastaan yksi sivu, joka on pohjana sovelluksen sisällöille. Sisällöt korvataan dynaamisesti yhdellä sivulla, jonka ansiosta yhden sivun sovellukset eivät ole yhtä riippuvaisia palvelimesta kuin monet perinteisemmät monen sivun sovellukset. (Angular University Blog, n.d.)

5.2 Angular CLI

Vahvasti Angulariin kiinnitetty ominaisuus on Angular CLI (Angular Command Line Interface), mikä on komentorivin käyttöliittymätyökalu. Angular CLI -ominaisuuden avulla voidaan luoda, kehittää sekä ylläpitää Angular-projekteja suoraan komentoriviltä käsin. Angular-projektia varten on hyvä tietää muutamia peruskomentoja, joiden avulla pystytään hallinnoimaan projekteja sekä luomaan uusia komponentteja. (Anaxinet, n.d.)

5.2.1 Angular CLI:n muutamia peruskomentoja

Ennen aloittamista Angular CLI on asennettava hyödyntämällä tietokoneella olevaa komentoriviä (Angular, n.d.), joka tapahtuu yksinkertaisella komennolla:

```
npm install -g @angular/cli
```

Angular CLI:n asentamisen jälkeen pystytään luomaan uusi Angular-projekti lyhyellä komennolla. Komennossa määritetään Angular-projektille nimi, jonka jälkeen Angular CLI luo kansion, jonne Angular-projektin vaatimat komponentit, tiedostot sekä projektin rakenne asetetaan. Komento, jolla tämä kaikki tapahtuu:

```
ng new AppName
```

Angular-projekti saadaan käynnistettyä paikallisesti alapuolella olevalla komennolla. Tämän jälkeen projekti käynnistyy oletusarvoisesti localhost:4200-portilla, jonka kirjoittamalla tietokoneen selaimen hakukenttään saadaan nähtyä reaaliaikaisessa ympäristössä, miltä projekti näyttää. Komento, jolla projekti saadaan käynnistettyä:

```
ng serve
```

Projektin sisälle on myös mahdollista luoda Angular CLI:n kautta komponentteja, palveluita sekä muita keskeisiä sovelluksen alueita. Komennon loppuun lisätään vielä erikseen, mitä halutaan luoda, kuten ”ng generate component” tai ”ng generate service”:

```
ng generate
```

Angular CLI on yksinkertaisesti helppokäyttöinen komentorivin käyttöliittymä, jolla voidaan hallita projekteja muutamalla yksinkertaisella komennolla. Tämän opinnäytetyön aikana Angular CLI oli vahvasti käytössä, sillä sen avustuksella luotiin itse sovellus sekä esimerkiksi tarvittavia projektin komponentteja. (Anaxinet, n.d.)

5.3 PHP

PHP on yleisesti käytetty ohjelmointikieli palvelinympäristöissä dynaamisia web-palveluita kehitettäessä. PHP on lyhenne sanoista ”PHP: Hypertext Preprosessor”. Tässä työssä PHP-ohjelmointikieltä hyödynnettiin palvelinympäristöissä tietokantojen hallintaa ja päivittämistä varten, joten se on avainasemassa back-end-ohjelmoinnissa. PHP:ta on nähty nimitettävän usein myös komentokieleksi tai skriptauskieleksi, jolla tarkoitetaan koodikokonaisuutta, jolla pyritään automatisoimaan prosessit, jotka muuten olisi suoritettava vaiheittain yksitellen aina sivuston koodissa. (Brian, 2021)

PHP:n historia ylettyy aina vuoteen 1994, jolloin Rasmus Lerforf hyödynsi julkaisematonta versiota kotisivulta käyttäjien seuraamista varten. Ensimmäisen kerran PHP nähtiin muiden käytössä vuoden 1995 alussa, ja se tunnettiin nimellä ”Personal Home Page Tools”. PHP:n suosio kasvoi nopeasti, sillä vuonna 1996 sitä käytettiin jo vähintään 15 000 nettisivulla ympäri maailmaa (ifj.edu.pl, n.d.). Tuoreimpien W3Techs.com-sivuston (W3Techs, n.d.) tutkimuksien mukaan PHP-ohjelmointikielen suosio on edelleen vahva, sillä peräti 79,1 prosenttia nettisivustoista käyttää PHP:ta palvelinpuolen ohjelmoinnissa vielä helmikuussa 2021. (Brian, 2021)

Koska kyseessä on palvelinpuolen ohjelmointikieli, niin se ei ole aina välittömästi näkyvissä suoraan verkkopalvelun käyttäjälle. PHP-ohjelmointikielellä pystytään noutamaan tietoja suoraan palvelimelta, jotka näytetään verkkopalvelussa käyttäjälle. Tämä on hyödyllinen

toteutus, sillä kaikkea tietoa ei tarvitse tallentaa paikallisesti verkkosivulle. Lisäksi tietojen löytyminen erillisestä tietokannasta mahdollistaa tehokkaampien ja nopeampien verkkopalveluiden toteuttamisen. (Scott, n.d.-b)

5.4 JavaScript

JavaScript on skriptauskieli, jota hyödynnetään PHP:sta poiketen enemmän front-end-ohjelmoinnissa HTML:n kaverina. JavaScriptin on kuvattu usein saavan nettisivut elämään, sillä sen avustuksella sivuista saadaan interaktiivisempia ja näyttävämpiä. JavaScriptin ohjelmia kutsutaan skripteiksi, ja ne voidaan ohjelmoida suoraan verkkosivun HTML-koodiin ja suorittaa sivun latautumisen yhteydessä automaattisesti. (JavaScript.info, n.d.)

Alun perin JavaScript oli yhdysvaltaisen Netscapen kehittämä skriptauskieli ja sitä kutsuttiin nimellä LiveScript. Myöhemmin nimi muutettiin JavaScriptiksi, sillä Java oli tuolloin erittäin suosittu, joten Javasta pyrittiin hakemaan nimessä tunnettavuutta. JavaScriptillä ei ollut kuitenkaan mitään tekemistä Javan kanssa, sillä kyseessä oli täysin oma ohjelmointikielensä, eikä sillä ole lainkaan yhteyttä Javaan. (JavaScript.info, n.d.)

Nykyisin JavaScript ei ole enää selainriippuvainen, vaan sitä voidaan hyödyntää myös palvelinpuolella ja oikeastaan myös millä tahansa laitteella, jossa on JavaScript-moottori. JavaScriptin ominaisuudet riippuvat paljolti ympäristöstä, jossa sitä suoritutetaan. Palvelinpuolella JavaScriptillä voidaan lukea tai kirjoittaa tiedostoja tai vaikka suorittaa verkkopyyntöjä. Selaimissa JavaScript pystyy tekemään asioita, mitkä liittyvät verkkosivujen manipulointiin, verkkopalvelimeen tai esimerkiksi vuorovaikutukseen käyttäjän kanssa. (JavaScript.info, n.d.)

5.5 TypeScript

TypeScript on lyhyesti ja ytimekkäästi kaikki mitä JavaScript tarjoaa ja hieman enemmän. TypeScript on avoimen lähdekoodin kieli, joka perustuu JavaScriptiin, joka on yksi maailman eniten käytetty skriptauskieli, mutta TypeScriptiin on lisätty staattisia vahvoja tyypityksiä. Alun perin TypeScript on ollut yhdysvaltalaisen Microsoftin kehittämä ja se julkaistiin vuonna 2012. (Jani, 2016)

Vahvat tyypitykset TypeScriptissa tarkoittavat, että muuttujille voidaan asettaa tyypit sekä esimerkiksi rajapintaluokat. Hyvä esimerkki tyypityksestä on esimerkiksi se, että muuttujat ”valtio” ja ”väkiluku” ovat määritettynä merkkijonoksi sekä numeroksi, niin tällöin tietokone ymmärtää muuttujan ”väkiluku” olevan numero (esim. 6 000 000), eikä merkkijono (kuusimiljoonaa). TypeScriptissä tyypitystä jäljitellään ohjelmointiympäristössä ja lopputulos käännetään normaaliksi JavaScriptiksi, joten sitä tukevat lähes kaikki selaimet ja ympäristöt. Tyypityksen edut tulevat esille laajojen ohjelmistojen kehityksessä. (Jani, 2016)

5.6 HTML

HTML on lyhenne sanoista Hypertext Markup Language ja sitä käytetään luodessa nettisivun runkoa. Ohjelmoija voi luoda ja jäsentää osoita, kappaleita, otsikoita, linkkejä ja muita perusasioita nettisivulle. HTML ei ole ohjelmointikieli, sillä sen avulla ei ole mahdollista luoda dynaamisia toimintoja. HTML on merkintäkieli (*markup language*), joka on erittäin yksinkertainen ja sitä on helppoa oppia käyttämään. (Domantas, 2019)

Alun perin HTML:n keksi Tim Berners-Lee Sveitsissä, joka keksi idean internet-pohjaisesta hypertekstijärjestelmästä vuonna 1991. HTML:n suosio kasvoi nopeasti vuosien varrella ja nyt sitä pidetään virallisena verkkostandardina. Nykyisin HTML-merkintäkieltä ylläpitää ja kehittää World Wide Web Consortium (W3C), joka julkaisi suuren HTML5-päivityksen vuonna 2014. Tuorein päivitys on vuonna 2017 julkaistu HTML 5.2 -versio. (Domantas, 2019)

5.7 CSS

CSS on lyhenne sanoista Cascading Style Sheet ja sitä hyödynnetään nettisivun tyylittelyssä. Siinä missä HTML:ää käytetään nettisivun rungon rakentamiseen, niin CSS:ää hyödynnetään sivun tyylin toteuttamiseen. CSS:ssa voidaan määrittää esimerkiksi sivun asettelu, värit, fontit ja muut viimeistelyt. HTML toimii yksinkertaisena pohjana nettisivulle ja CSS:n kautta pystytään tyylittelemään sivun käyttäjälle näkyvä osuus. CSS toimii käsikädessä HTML:n kanssa. (Scott, n.d.-a)

CSS voidaan toteuttaa samaan tiedostoon HTML:n kanssa <style>-tagin sisällä tai sitä varten voi olla oma tiedostonsa, jota kutsutaan tarvittaessa HTML-tiedostossa. Lisäksi CSS-koodia voidaan kirjoittaa myös suoraan elementtiin. HTML-koodissa voi olla esimerkiksi kappale ” Tervetuloa! ”, ja jos siihen halutaan tyylittelyä, niin sitä varten voidaan toteuttaa CSS-koodi, missä teksti muutetaan punaiseksi sekä lihavoiduksi. Tämä onnistuu koodinpätkällä ”p { color:red; font-weight:bold; }”. (Scott, n.d.-a)

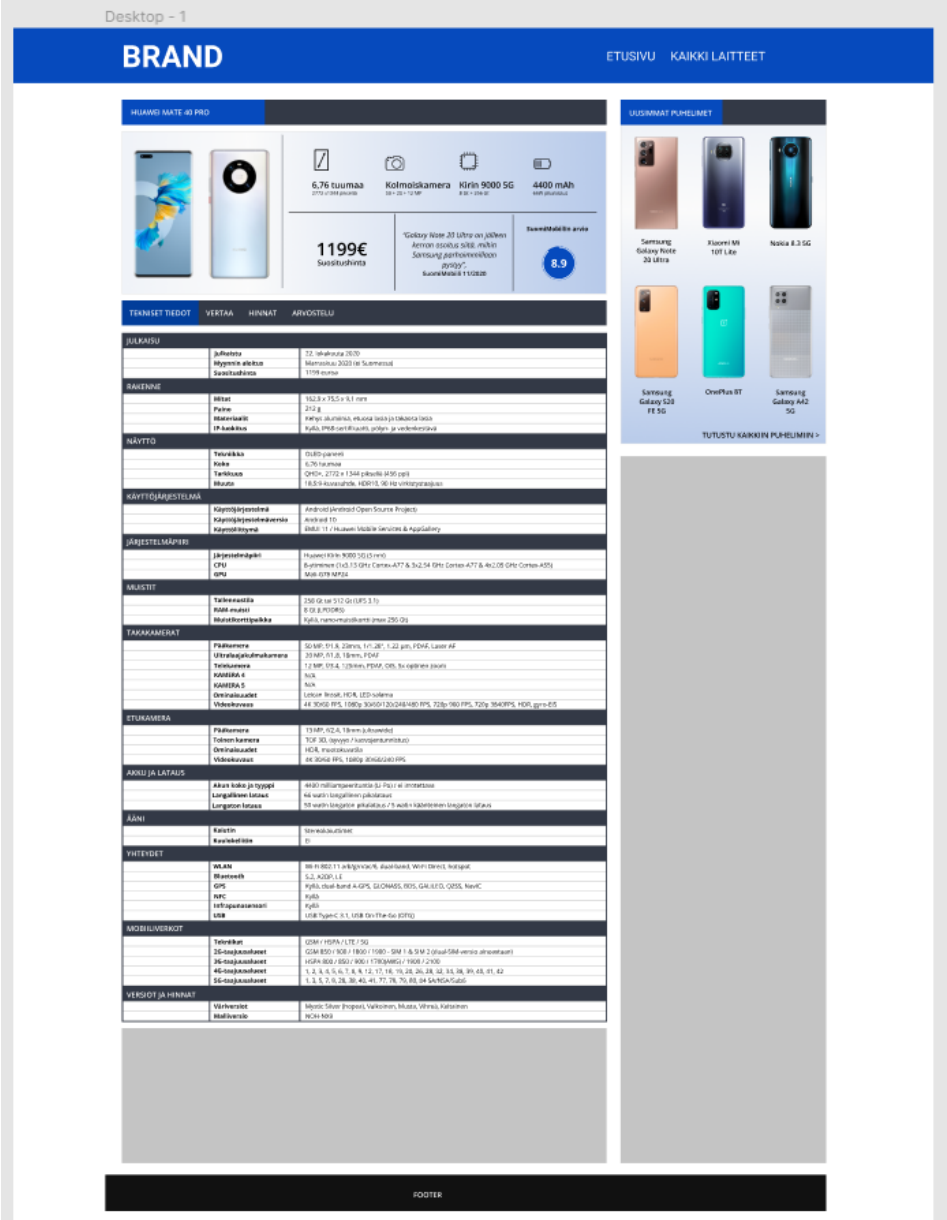
Edellä mainitussa koodinpätkässä ”p” on viittaus HTML-koodissa olevaan kappaleeseen, jonka jälkeen aaltosulkeissa olevat CSS-koodit ovat ilmoituksia, joissa määritetään fontin väri sekä paino. ”color” ja ”font-weight” ovat molemmat ominaisuuksia ja ”red” ja ”bold” vastaavasti arvoja, joiden mukaisesti HTML-kappaleen välissä oleva teksti muokkaantuu halutulla tavalla (Scott, n.d.-a). CSS-koodissa pystytään toteuttamaan myös monipuolisempia määrityksiä, mutta käytännön toimivuus ilmenee esimerkistä selkeästi. (W3, n.d.-a)

6 Kehitysprojektin toteuttaminen

Ohjelmointia ennen sivuston ulkoasu suunniteltiin Figma-työkalulla niin, että toteutuksessa olisi otettu huomioon monet käyttöliittymäsuunnittelun periaatteet. Sivusto toteutettiin kahdessa näkymässä, jotka ovat tuotesivu sekä vertailusivu. Palvelussa pystyy katsomaan joko yhden tuotteen tietoja kerrallaan tai vertaamaan tietoja kahden tai kolmen tuotteen kesken.

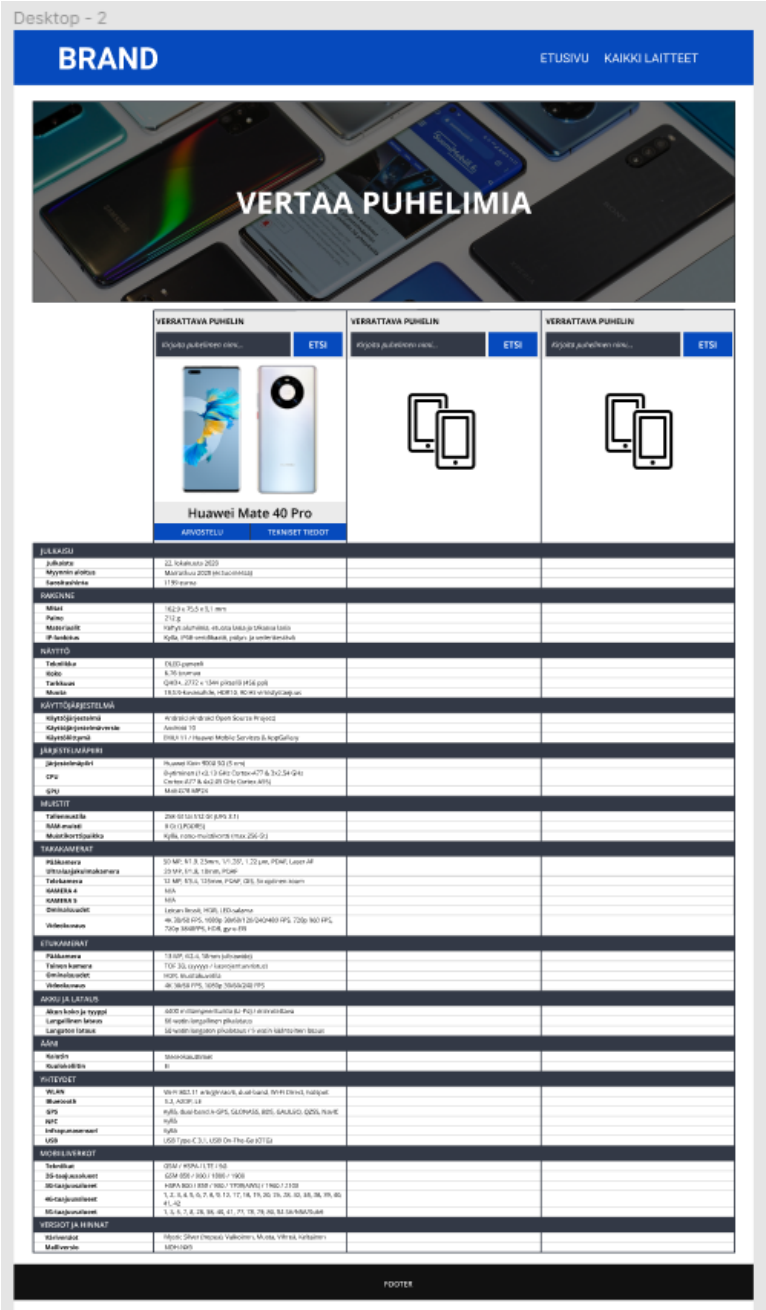
Käyttöliittymä on suunniteltu mahdollisimman yhteneväksi niin, että se on helppokäyttöinen ja tarjoaa kaiken tarvittavan. Yhden tuotteen sivulla on listattuna kaikki keskeisimmät tuotteen tekniset tiedot yksityiskohtaisesti, ja samalla käyttäjä näkee kuvan tuotteesta ja myös laitteen hinnan. Suunnitelmassa on otettuna huomioon mahdolliset lisätoiminnallisuudet, jotka saatetaan jatkokehittää palveluun myöhemmin. Tällaisia ominaisuuksia ovat muun muassa hintojen vertailu tai linkki tuotteen arvosteluihin. Yksittäisen tuotteen sivulla hyödynnetään sivupalkillista toteutusta. Tämä mahdollistaisi sivupalkin hyödyntämisen erilaisien tietojen näyttämisessä ja siihen voidaan kehittää erilaisia toimintoja tai asettaa linkit sosiaalisen median kanaviin. Käyttöliittymäsuunnitelma on esillä kuvassa 3 alapuolella.

Kuva 3 Yksinkertainen suunnitelma siitä, miltä yhden tuotteen tietonäkymä voisi näyttää.



Kuvassa 4 on esillä suunnitelma tuotteiden teknisien ominaisuuksien vertailusivusta, jossa on mahdollista verrata kahta tai maksimissaan kolmea tuotetta keskenään. Käyttäjä näkee yhdellä silmäyksellä kaikki haluamansa tuotteet kerralla, jonka lisäksi tuotteita voi vaihtaa yläpuolella olevasta tuotevalitsimesta.

Kuva 4 Yksinkertainen suunnitelma siitä, miltä maksimissaan kolmen tuotteen tietonäkymä voisi näyttää.

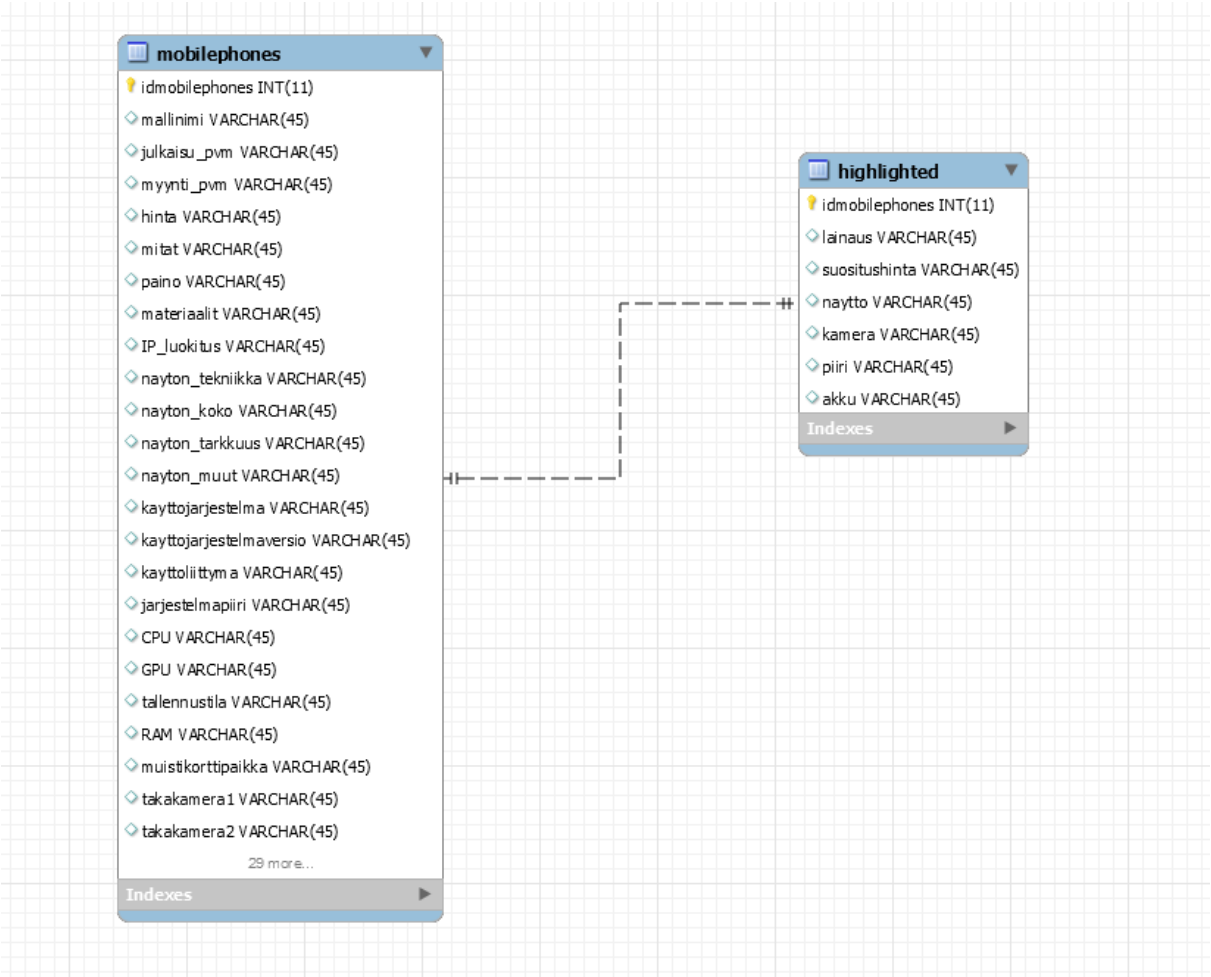


7 Verkkopalvelun tietokanta ja hallintajärjestelmä

Verkkopalvelua varten toteutettiin tietokantasuunnitelma. Palvelussa tietokantaan tallennetaan tuotteiden tiedot yhteen tauluun sekä toisessa taulussa on tallennettuna tuotteen korostetut ominaisuudet. Suunnitelman tavoitteena oli toteuttaa tietokanta mahdollisimman yksinkertaiseksi, jotta sen hallinta olisi helppoa. Tietokannan nimeksi

valikoidu ”productdb” ja sen sisältämät taulut ovat ”mobilephones” sekä ”highlighted”. Kuvassa 5 on nähtävillä tietokantamallinnus.

Kuva 5 Productdb-tietokannan taulut



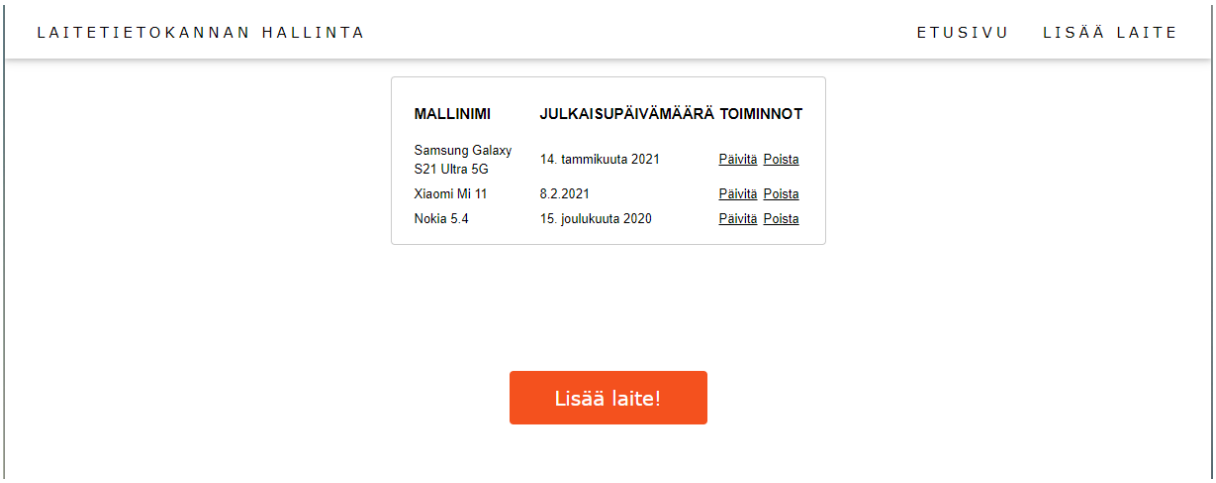
Opinnäytetyön aikana MYSQL-tietokanta toteutettiin paikallisesti omalla tietokoneella, sillä koko projekti toteutettiin yhdellä tietokoneella. Lisäksi tietokanta on kehitetty niin, että sitä on mahdollista jälkikäteen muokata helposti. Projektin ensimmäisessä vaiheessa on käytetty ainoastaan ”mobilephones” -taulua, mutta ”highlighted” -taulu on toteutettu palvelun jatkokehittämistä varten jo valmiiksi.

7.1 Tietokannan hallintajärjestelmä

Opinnäytetyön tuloksena toteutettiin verkkopalvelun prototyyppiversio, jossa käyttäjä pystyy tarkastelemaan tuotteen tietoja sekä vertailemaan tuotteita keskenään. Verkkopalvelu koostuu kahdesta kokonaisuudesta, jotka ovat käyttäjälle näkyvä pääsivu sekä

hallintaa varten toteutettu tuotteiden hallintanäkymä, jossa on mahdollista lisätä, muokata sekä poistaa tuotteita. On hyvä huomata, että kyseessä on myös hallintanäkymän täysin toimiva raakaversio, jossa on päätarkoituksena demonstroida erilaisia toiminnallisuuksia. Tietokannan hallintanäkymässä on listattuna kaikki tietokannassa olevat tuotteet, jonka lisäksi on toteutettu kaksi erillistä HTML-sivua, joissa pystyy lisäämään uuden tuotteen sekä päivittämään tietokannassa olevan tuotteen tietoja. Kuvassa 6 on esillä hallintanäkymän pääsivun yksinkertainen toteutus.

Kuva 6 Tietokannan hallintanäkymän etusivu.



Tietokannan hallintanäkymä toteutettiin hyödyntämällä PHP-ohjelmointikieltä. Hallintanäkymässä on mahdollista nähdä kaikki MySQL-tietokannassa olevat tuotteet taulukossa, jonka lisäksi hallintapaneelin käyttäjä pystyy päivittämään ja poistamaan tuotteita tietokannasta. Hallintapaneelissa käyttäjä pystyy myös lisäämään uusia tuotteita tietokantaan yksinkertaisen lomakkeen avulla, jossa syötetään tuotteen vaadittavat tiedot.

Toteutus on tehty demonstroimaan helppoa tapaa lisätä, muokata sekä poistaa tietoja MySQL-tietokannasta paikallisesti. Kyseessä ei ole koskaan laajalle yleisölle tarkoitettu palvelu, vaan hallintanäkymä on suunnattu ainoastaan tiimin tai yhteisön sisäiseen käyttöön osana paikallisia intrapalveluita.

7.2 Kohteen lisääminen tietokantaan

Kohteita lisätään MySQL-tietokantaan SQL INSERT INTO -käskyllä, jossa jokaiseen taulun sarakkeeseen tieto lisätään käyttäjän lomakkeeseen syöttämien tietojen mukaisesti. Alapuolella olevassa kuvassa 7 on esillä sql-käsky, jolla tiedot lisätään tietokantaa. Toteutuksessa luodaan tietokantaan aina uusi rivi uudelle laitteelle, jolloin kaikki käyttäjän syöttämät tiedot siirtyvät tauluun. Tietokannassa uudelle riville luodaan automaattisesti ID-tunnus, jolla se erottuu muista taulussa olevista riveistä.

Kuva 7 Lomakkeessa olevien tietojen lisäys paikalliseen tietokantaan.

```
$sql = "INSERT INTO mobilephones (mallinimi, julkaisu_pvm, myynti_pvm, hinta, mitat,
if(mysqli_query($link, $sql)){
    echo "Records added successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}

// Yhteyden sulkeminen
mysqli_close($link);
?>
<tr>
<A HREF="index.php">Palaa päänäkymään.</A>
</tr>
```

7.3 Kohteiden päivittäminen

Tietokannassa olevien kohteiden päivittämiseen käytettiin MYSQL UPDATE -käskyä, jolla voidaan päivittää tietyn kohteen yksittäisiä tietoja tai vaihtoehtoisesti päivittää kohteen kaikkia sarakkeita. Päivittämistä varten on toteutettu lomake, johon haetaan tietokannasta ID:n perusteella nykyiset tiedot, ja niitä on mahdollista muokata lomakkeen kautta. Kuvassa 8 on esillä sql-komento, jolla tietoja päivitetään paikallisessa MySQL-tietokannassa.

Kuva 8 Tietojen päivittäminen tietokannassa.

```php
<?php
include_once 'database.php';
if(count($_POST)>0) {
mysqli_query($conn,"UPDATE mobilephones set idmobilephones='" . $_POST['idmobilephones'] . "', mallinimi='" .
$message = "Record Modified Successfully";
}
$result = mysqli_query($conn,"SELECT * FROM mobilephones WHERE idmobilephones='" . $_GET['idmobilephones'] . "
$row= mysqli_fetch_array($result);
?>
```

7.4 Kohteiden poistaminen tietokannasta

Tietokannassa olevia tietoja voidaan myös poistaa hallintanäkymässä yksinkertaisesti DELETE FROM -komennolla, jolloin kohteen kaikki tiedot voidaan pysyvästi poistaa tietokannasta. Kohde poistetaan tietokannasta ID:n perusteella, jolloin taulusta poistetaan valitun ID:n kaikki tiedot. Kuvassa 9 on esillä komento, jolla tiedot poistetaan riveittäin tietokannasta.

Kuva 9 Tietojen poistaminen tietokannasta ID:n perusteella.

```php
<?php
include_once 'database.php';
$sql = "DELETE FROM mobilephones WHERE idmobilephones='" . $_GET["idmobilephones"] . "'";
if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

8 Verkkopalvelun front-end-prototyyppiversio Angularilla

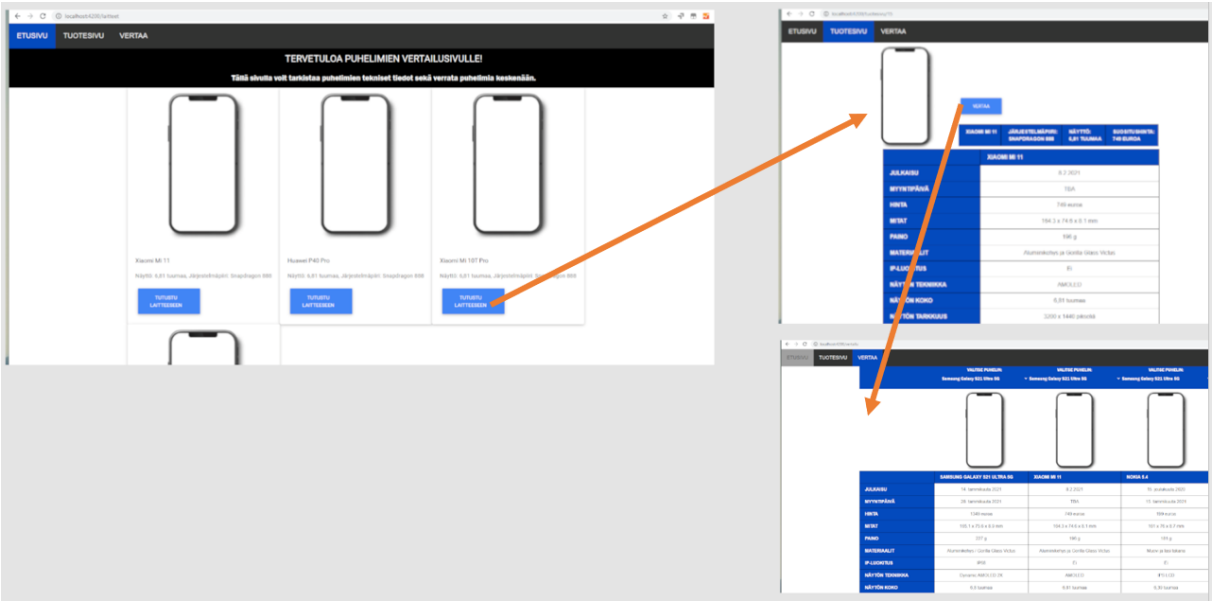
Projektin aikana oli tarkoitus tutustua Angulariin toteuttamalla suunnitellusta palvelusta prototyyppiversio, jota on mahdollista jälkikäteen jalostaa eteenpäin. Tavoitteena oli saada toteutettua sivu, jossa pystytään katsomaan yksittäisen tuotteen teknisiä ominaisuuksia sekä verrata niitä toiseen tuotteeseen. Prototyyppiversiossa ideana ei ole toteuttaa suunnitelmassa toteutettua käyttöliittymää, eikä hyödyntää tietokannan sisältöä kokonaisuudessaan osana palvelua. Ideana on tarkistaa mallinnuksen avulla, olisiko Angularista vaihtoehtoinen toteutustapa vertailusivuston front-end-toteuttamisessa.

Verkkopalvelua lähdettiin toteuttamaan tutustumalla Angularin periaatteisiin sekä hyödynnettiin Angularin tarjoamaa yhden sivun toteutusta. Palvelusta tehtiin mahdollisimman helposti hallittava, sillä se koostuu kolmesta pääkomponentista. Tässä luvussa käydään läpi palvelun toteutus sekä opinnäytetyöprojektin tärkein osa-alue.

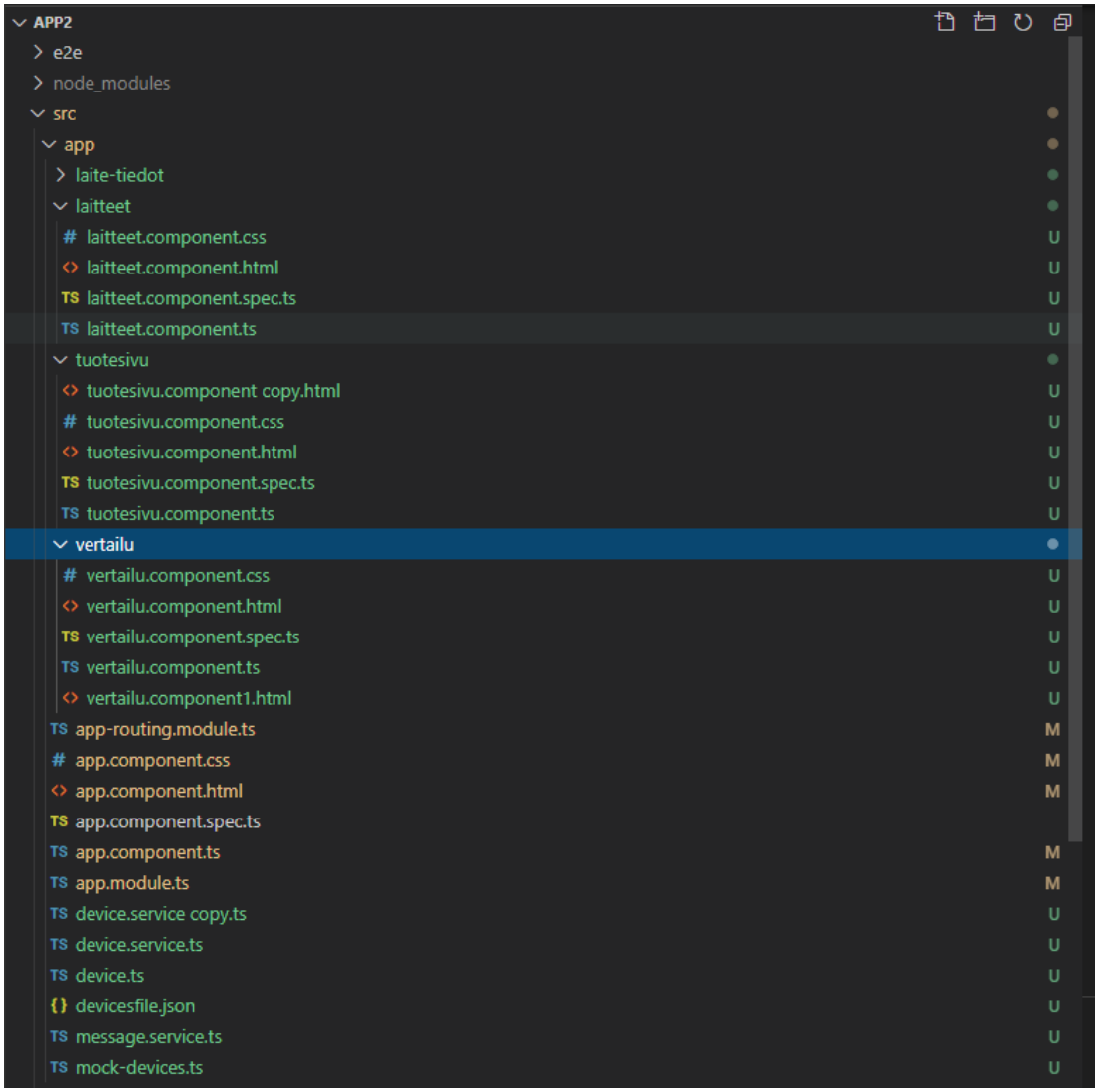
8.1 Sivustorakenne ja komponentit

Sivustorakenne koostuu Angular-projektissa päänäkymästä, tuotenäkymästä sekä vertailunäkymästä, joihin tämän opinnäytetyön aikana keskityttiin. Sivuston rakenne on esitettynä kuvassa 10. Nämä kolme näkymää valikoituivat sivuston prototyyppiversioon niiden tärkeyden takia, sillä ne ovat palvelun kokonaisuuden kannalta avainasemassa. Itse Angular-projekti koostuu useista komponenteista, jotka pitävät sisällänsä aina html-, typescript- ja css-tiedostot. Jokainen komponentti on jaettu helpoiksi kokonaisuuksiksi, joita on helppo hallita sekä kehittää itsenäisesti. Angular-projekteissa on aina myös juurikomponentti, johon voidaan toteuttaa palvelun keskeisiä asioita, joita halutaan näyttää koko palvelussa. Muilla komponenteilla voidaan toteuttaa erilaisia toimintoja palveluun, jotka voidaan näyttää osana palvelun päänäkymää tai vaihtoehtoisesti omana sivunaan.

Kuva 10 Esitettynä sivustorakenne; päänäkymä, tuotenäkymä sekä vertailunäkymä.



Kuva 11 Projektin rakenne ja komponentit.



Opinnäytetyön aikana toteutetun Angular-projektin rakenne ja komponentit ovat nähtävissä kuvassa 11. Kuvassa nähdään projektissa olevan laitteet-, tuotesivu- ja vertailu-komponentit, joissa kaikissa on html-tiedosto, css-tyylitiedosto sekä typescript-tiedosto. Projektin avainasemassa on myös juurikomponentti, joka koostuu tiedostoista ”app.component.html”, ”app.component.css” sekä ”app.component.ts”. Näiden tiedostojen varassa on projektin runko. Projektissa toteutettiin juurikomponentissa navigointipalkki sekä alapalkki, jotka näytetään kaikilla sivuilla samanlaisena. Oikeastaan toisille sivuille ei koskaan siirrytä, vaan luodut komponentit ladataan juurikomponentin sisälle niitä tarvittaessa. Projektissa toteutettiin Angularin perinteinen yhden sivun toteutus, eikä palvelua käytettäessä tarvitse koskaan päivittää selainta uudelleen, jotta jokin sivun osa latautuisi.

Sovelluksen sisällä navigointiin hyödynnetään Angularin reititys-toimintoa (router), jolloin komponentit saadaan näkyviin juurikomponentin sisällä. Reitittimelle määritetään projektissa reitit, jotta juuri oikea komponentti osataan ladata. Reitit määritetään app-routing-module.ts-tiedostossa, joka on esillä kuvassa 12. Reitit määritetään Ruote-olioina, jonka avulla URL-polku saadaan yhdistettyä komponenttiin. Kuvassa 12 nähdään erilaisia path-arvoja ja esimerkiksi kolmannessa on polun lopussa lisäys /:id, jolla merkitään, että sivulla voidaan lukea tietoja parametrista. Lisäksi reititykseen on otettu käyttöön ohjaus, joka on kuvassa 12 ensimmäisenä näkyvä path-arvo. Palvelu siirtyy aina määritettyyn ohjaussivustoon, jos polku on jokin muu kuin mitä on ennalta määritetty.

Kuva 12 Angular-projektin reititys.

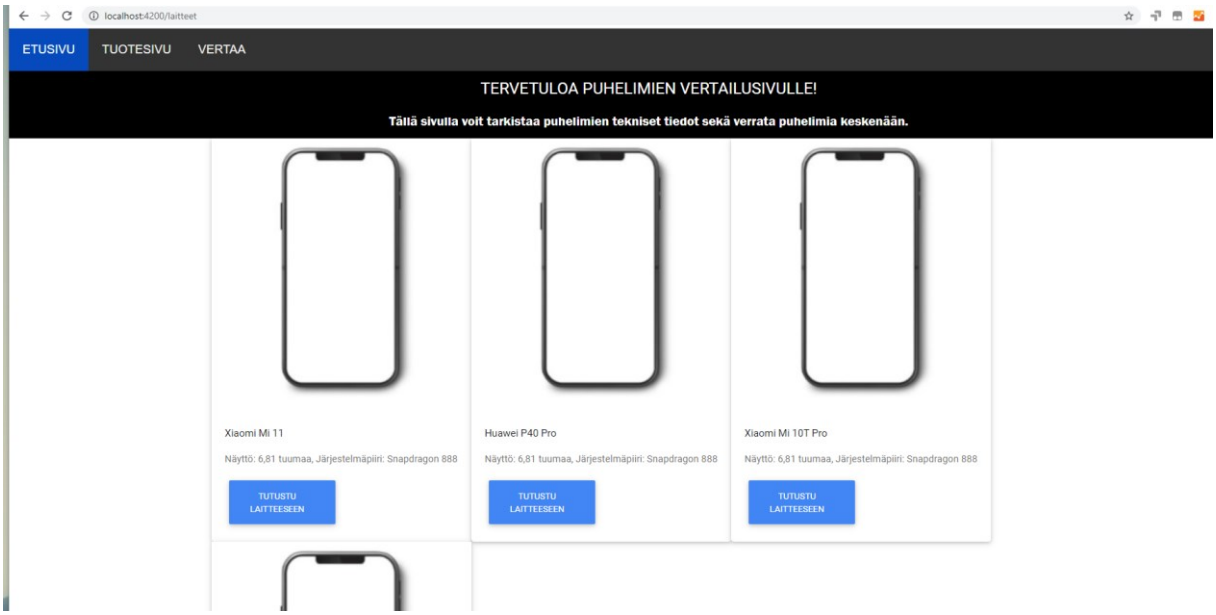
```
src > app > TS app-routing.module.ts > ...
1   import { NgModule } from '@angular/core';
2   import { RouterModule, Routes } from '@angular/router';
3
4   import { LaitteetComponent } from './laitteet/laitteet.component';
5   import { TuotesivuComponent } from './tuotesivu/tuotesivu.component';
6   import { VertailuComponent } from './vertailu/vertailu.component';
7
8   const routes: Routes = [
9     { path: '', redirectTo: '/laitteet', pathMatch: 'full' },
10    { path: 'laitteet', component: LaitteetComponent },
11    { path: 'tuotesivu/:id', component: TuotesivuComponent },
12    { path: 'vertailu/:id', component: VertailuComponent },
13    { path: 'vertailu', component: VertailuComponent }
14
15  ];
16
17  @NgModule({
18    imports: [RouterModule.forRoot(routes)],
19    exports: [RouterModule]
20  })
21  export class AppRoutingModule { }
22
```

8.1.1 Päänäkymä

Palvelun päänäkymästä on käyttöliittymän prototyyppiversiossa tehty hyvin yksinkertainen ja se on nähtävillä kuvassa 13. Päänäkymässä on esillä navigointipalkki, josta pystytään siirtymään palvelun eri osioihin. Navigointipalkin alapuolelle on toteutettuna käyttäjää informoiva tekstialue, jossa kerrotaan lyhyesti palvelun idea. Päänäkymän sisältöosassa on listattuna tietokannasta haettuja tuotteita sekä visuaalisuutta on tuotu hakemalla tuotteiden kuvat, mallinimi sekä muutamia teknisiä tietoja, joista käyttäjä saa heti nopealla tarkistamisella käsityksen millaisesta tuotteesta on kysymys. Jokaisen tuotteen kohdalla on

myös painike, jota painamalla käyttäjä pystyy siirtymään kyseisen tuotteen yksittäiselle tietosivulle.

Kuva 13 Palvelun prototyyppiversion päänäkymä.



Päänäkymän tuotteet ovat asetettuna korttinäkymään, joka on toteutettuna Angular Bootsrap -komponenteilla. Toteutus valikoitui päänäkymään korttinäkymän yksinkertaisuuden sekä skaalautuvuuden mukaan. Kortit ovat rakenteeltaan yksinkertaisia sekä sisältö on selvästi esitettynä. Korteissa hyödynnetään lisäksi niin sanottua materiaalitoteutusta (Material Design), mikä on internetissä sekä mobiilisovelluksissa varsin yleinen standardi.

Kaikki tuotteet ovat listattuna päänäkymään ngFor-direktiivin avustuksella ennalta määritetystä laitetietokannasta. Jokaiselle tuotteelle tehdään automaattisesti oma korttinsa

päänäkymään, kun direktiivi on asetettu kyseisen komponentin tagin sisälle kuvan 14 mukaisesti.

Kuva 14 Päänäkymän korttien toteutus koodissa.

```
<div class="card-group">
  <b *ngFor="let device of devices"
      routerLink="/tuotesivu/{{device.id}}">

  <!-- KORTTINÄKYMÄ -->
  <mdb-card>
    <div class="view rgba-white-slight waves-light" mdbWavesEffect>
      <!-- KORTIN KUVA  -->
      <mdb-card-img src="https://uploads.suomimobiili.fi/2021/03/phone-kuva.jpeg" alt="Card image cap"></mdb-card-img>
      <a>
        <div class="mask"></div>
      </a>
    </div>
    <!--KORTIN SISÄLTÖ-->
    <mdb-card-body>
      <!--KORTIN OTSIKKO-->
      <mdb-card-title>
        <h4>{{device.mallinimi}}</h4>
      </mdb-card-title>

      <!--KORTIN TEKSTI-->
      <mdb-card-text> Näyttö: {{device.nayton_koko}}, Järjestelmäpiiri: {{device.jarjestelmapiiri}}
      </mdb-card-text>

      <a href="tuotesivu/{{device.id}}" mdbBtn color="primary" mdbWavesEffect>Tutustu laitteeseen</a>
    </mdb-card-body>
  </mdb-card>
</b>
```

8.1.2 Tuotenäkymä

Palvelun prototyyppiversiossa päänäkymästä siirrytään halutun tuotteen tuotenäkymään, josta on mahdollista tarkistaa yksittäisen tuotteen tarkat ominaisuudet sekä tiedot. Näkymässä on nähtävillä tuotteen kuva, tiedot yhdessä taulussa sekä vertaa-painike, jota painamalla kyseistä tuotetta voidaan verrata muihin laitteisiin. Verrattavat tuotteet voidaan valita jälkikäteen erillisessä vertailunäkymässä.

Näkymässä olevaan tauluun tuotekohtaiset tiedot ovat haettu tietokannasta käyttäjän valitseman tuotteen id-merkinnän mukaisesti. Koodissa on hyödynnetty NgIf-direktiiviä, jolla pystytään hakemaan tietokannasta vain kyseisen tuotteen yksityiskohtaiset tiedot. Kuva toteutuksesta on esillä kuvassa 15. Device-luokassa on myös määritetty, miten getDevice-metodilla saadaan tietokannasta haettua id-tunnuksen avulla vain tietyn tuotteen tiedot.

Kuva 15 Yksittäisen tuotteen tietojen näyttämiseen hyödynnetään Nglf-direktiivia.

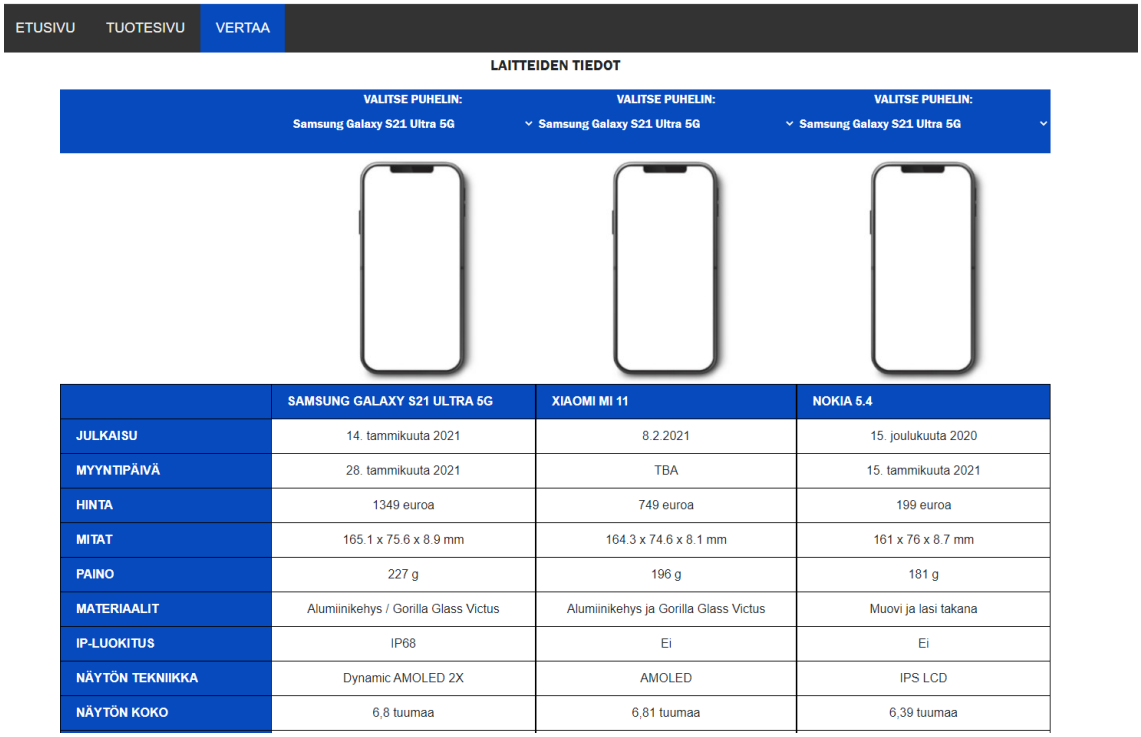
```
34  <div class="column" >
35
36      <table *ngIf="device">
37        <tr>
38        <th> </th>
39        <th>{{ device.mallinimi }}</th>
40        <tr>
41        <th>Julkaisu</th>
42        <td>{{ device.julkaisu_pvm }}</td>
43        <tr>
44          <th>Myyntipäivä</th>
45          <td>{{ device.myynti_pvm }}</td>
46              <tr>
47          <th>Hinta</th>
48          <td>{{ device.hinta }}</td>
49          <tr>
50          <th>Mitat</th>
51          <td>{{ device.mitat }}</td>
52          <tr>
53          <th>Paino</th>
54          <td>{{ device.paino }}</td>
55          <tr>
```

8.1.3 Vertailunäkymä

Vertailunäkymään päästään palvelussa joko yksittäisen tuotteen tuotesivulta tai vaihtoehtoisesti palvelussa on mahdollista siirtyä suoraan vertailunäkymään, jossa voidaan valita halutut verrattavat tuotteet jälkikäteen. Vertailunäkymässä on mahdollista tarkistaa kolmen tuotteen tiedot rinnakkain, jolloin käyttäjä pystyy vertailemaan tietoja keskenään.

Vertailtavien tuotteiden tiedot haetaan tietokannasta sivulla näkyvään tauluun samalla tavalla id-merkinnän mukaisesti kuin muillakin palvelun sivuilla. Sivulla on kolme erillistä taulua, jotka ovat asetettuna rinnakkain ja vertailtavaksi halutun tuotteen, käyttäjä voi valita taulun yläpuolella olevasta valitsimesta. Vertailunäkymän toteutus on nähtävillä kuvassa 16.

Kuva 16 Vertailunäkymän toteutus.



	SAMSUNG GALAXY S21 ULTRA 5G	XIAOMI MI 11	NOKIA 5.4
JULKAISU	14. tammikuuta 2021	8.2.2021	15. joulukuuta 2020
MYYNTIPÄIVÄ	28. tammikuuta 2021	TBA	15. tammikuuta 2021
HINTA	1349 euroa	749 euroa	199 euroa
MITAT	165.1 x 75.6 x 8.9 mm	164.3 x 74.6 x 8.1 mm	161 x 76 x 8.7 mm
PAINO	227 g	196 g	181 g
MATERIAALIT	Alumiinikehys / Gorilla Glass Victus	Alumiinikehys ja Gorilla Glass Victus	Muovi ja lasi takana
IP-LUOKITUS	IP68	Ei	Ei
NÄYTÖN TEKNIIKKA	Dynamic AMOLED 2X	AMOLED	IPS LCD
NÄYTÖN KOKO	6,8 tuumaa	6,81 tuumaa	6,39 tuumaa

8.2 Angular font-end-toteutuksessa

Yksi opinnäytetyön tavoitteista oli vastata siihen, onko Angular hyvä vaihtoehto verkkopalvelun front-end-toteuttamiseen. Tähän kysymykseen pyrin hakemaan vastausta Angular-projektin kautta ilman aikaisempaa kokemusta Angularista. Verkkopalveluiden ja sivujen ohjelmointi Angularilla tuntui aluksi hyvin haastavalta, mutta perehdyttäessä Angularin ideaan ja toimintaperiaatteeseen ohjelmointi tuntui loogiselta ja luontevalta. Yhdessä päivässä Angularin ymmärtäminen ja kokonaisuuden hallitseminen ei onnistunut, mutta verkosta löytyvien kattavien ohjeistuksien sekä yhteisön vinkkien avustuksella Angularin toiminnan ymmärtää helposti. Lisäksi tekemällä virheitä toimintaperiaatteen ymmärtää nopeasti.

Angular sopii front-end-ohjelmointiin erityisesti yksinkertaisen hallittavuuden sekä selkeän kokonaisuuden ansiosta. Komponentteihin perustuva rakenne helpottaa kokonaisuuden ylläpitämistä ja yhteen komponenttiin voidaan tehdä muutoksia niin, että ne eivät vaikuta muihin komponentteihin tai niiden rakenteeseen. Opinnäytetyötä varten tehdyn projektin

aikana havaitsin komponenttien muokkaamisen ja kehityksen olevan helppoa ja yksittäisen tiedostot eivät kasva selkeän rakenteen ansiosta liian pitkiksi ja hankaliksi hallita.

Yhden sivun idea tekee Angular-ohjelmistokehyksestä houkuttelevan tekniikan front-end-ohjelmoinnissa, sillä valttikorttina ovat nopeat palvelukokonaisuudet. Yhden sivun toteutuksen ansiosta selainta ei päivitetä palvelua käytettäessä uudelleen juuri koskaan ja ylimääräistä palvelimeen yhdistämistä ei tarvitse toteuttaa. Erityisesti Angular on hyvä pieniä palvelukokonaisuuksia toteuttaessa, eikä palvelun tarvitse olla edes kokonainen nettisivu. Angularilla voidaan toteuttaa komponentteja, joita voidaan käyttää hyödyksi muissa palvelukokonaisuuksissa.

Opinnäytetyöprojekti on toteutukseltaan vielä sen verran yksinkertainen ja hallittavissa oleva kokonaisuus, joka edesauttoi Angularin logiikan ja ohjelmoinnin oppimisessa. Toteutettu palvelu osoitti hyvin myös Angularin yhden sivun idean ja palvelussa päästiin hyödyntämään myös monia Angularin sisältämiä direktiivejä, joten osa niistäkin tuli tutuksi.

9 Johtopäätökset ja pohdinta

Tämän opinnäytetyön tavoitteena oli suunnitella verkkopalvelukokonaisuus sekä toteuttaa prototyyppiversio palvelusta. Projektimuotoisesti opinnäytetyön aikana saatiin toteutettua käyttöliittymäsuunnitelma, tietokanta ja sen hallintapalvelu sekä prototyyppiversio tuotteiden vertailupalvelusta, joka on ohjelmoitu hyödyntämällä Angularia. Lopputulos oli kokonaisuudessaan sellainen mitä pitikin, vaikka opinnäytetyön hyvin rajallinen työstöaika tuotti haasteita. Tiukan aikarajan takia kattavan toiminnallisen opinnäytetyön tekeminen on haastavaa, joka näkyy myös lopputuloksessa. Opinnäytetyössä esitellyn palvelun prototyyppiversiota ei pystytty toteuttamaan aivan niin pitkälle, mitä oli alun perin suunniteltu. Opinnäytetyössä pystyttiin silti vastaamaan tutkimuskysymyksiin.

Koska projektin tuloksena syntyi prototyyppiversio verkkopalvelusta, jossa pystytään vertailemaan laitteita keskenään, niin tulevaisuudessa palvelua voidaan kehittää eteenpäin. Palvelua voisi jatkokehittää selvästi eteenpäin tuomalla lisää toimintoja sekä tarjoamalla käyttäjälle aikaisempaa enemmän mahdollisuuksia personoida tuotteiden vertailuluetteloa. Esimerkiksi vertailutaulukossa voisi korostaa ominaisuuksia, jotka eroavat tuotteiden kesken tai esimerkiksi joistakin ominaisuuksista voisi antaa enemmän tietoa. Palveluun voisi integroida myös kolmannen osapuolen näkemyksiä tuotteista tai esimerkiksi ilmoittaa reaaliaikaisesti tuotteiden hinnat ja jälleenmyyjät. Palvelun jatkokehitysmahdollisuudet ovat lähes loputtomat ja aina palvelua voi viimeistellä paremmaksi.

Erityisesti Suomessa erilaisten tuotteiden vertailupalveluiden tarjonta ei ole kovin laaja, joten myös kysyntää vastaavalle palvelulle voisi olla. Palvelua voisi laajentaa esimerkiksi useampiin tuotekategorioihin, jotta loppukäyttäjä pystyisi vertailemaan nykyistä kattavammin erilaisien tuotekategorioiden tuotteita keskenään. Lisäksi jos palvelua kehittää eteenpäin, niin ehkä tulevaisuudessa sen pystyisi kaupallistamaan oikeaksi tuotteeksi tai ideaa voisi hyödyntää hieman muunneltuna tulevissa projekteissa. Projekti oli sen verran kattava, että kyseessä on hyvä pohja lähteä kehittämään kokonaisuutta eteenpäin, jos inspiraatiota löytyy.

10 Yhteenveto

Opinnäytetyön lopputuloksena syntyi palvelu, jossa käyttäjä pystyy tarkastelemaan tuotteen ominaisuuksia sekä vertailemaan useampia tuotteita keskenään. Pääasiallinen tavoite oli tutustua itselle ennalta täysin tuntemattomaan Angulariin ja sillä tehtävään front-end-ohjelmointiin. Lisäksi tarkoituksena oli saada parempi kuva siitä, onko Angular hyvä ohjelmistokehys erilaisten nettipalveluiden kehittämiseen. Opinnäytetyössä vastattiin kaikkiin tutkimuskysymyksiin ja erityisesti kyseessä on kattava katsaus Angulariin ja sovelluksien kehittämiseen yhden sivun toteutuksella.

Suurimmat hankaluudet työn aikana olivat Angulariin tutustuminen ja sen opiskelu, joka ei ollut helpoimmasta päästä. Lisäksi opinnäytetyön hyvin rajallinen aika hankaloitti työn edistämistä siihen suuntaan, mitä olisi voinut lopputuloksen kuvitella olevan. Ennen opinnäytetyön aloittamista otin työn oppimisprojektina, jotta pystyisin laajentamaan omaa osaamistani, mutta opinnäytetyön aikataulutus ei tukenut tätä ajatusmaailmaa. Opinnäytetyön aikana opin hyödyntämään Angularia varsin hyvin front-end-ohjelmoinnissa ja samalla tutustuin ensimmäistä kertaa yhden sivun verkkopalveluihin.

Verkkopalvelua voidaan kehittää jatkossa nykyisen prototyyppiversion pohjalta eteenpäin. Palveluun voidaan lisätä uusia ominaisuuksia tai laajempia tuotekategorioita. Nykyisen toteutuksen päälle on helppo kehittää uusia lisäyksiä. Tulevaisuudessa palvelu voidaan myös kaupallistaa tai kehittää osaksi jonkinlaisia nykyisiä palvelukokonaisuuksia.

Lähteet

Aluehallintovirasto. (n.d.). *WCAG 2.1: lain vaatimukset - Saavutettavuusvaatimukset*. Retrieved February 17, 2021, from https://www.saavutettavuusvaatimukset.fi/digipalvelulain-vaatimukset/wcag-2-1/

Anaxinet. (n.d.). *What is Angular CLI and how to do I use it?* Anexinet. Retrieved March 20, 2021, from https://anexinet.com/blog/angular-cli-use/

Angular. (n.d.). *Angular CLI*. Angular. Retrieved March 20, 2021, from https://cli.angular.io/

Angular University Blog. (n.d.). *Why Single Page Applications?* Angular University Blog. Retrieved February 18, 2021, from https://blog.angular-university.io/why-a-single-page-application-what-are-the-benefits-what-is-a-spa/

Ankit, M. (2019, November 3). *What is Angular?* Medium. https://medium.com/javascript-in-plain-english/what-angular-is-5d27bffb1fb1

Anusheh, Z. (n.d.). *What is Visual Studio Code?* Retrieved February 17, 2021, from https://www.educative.io/edpresso/what-is-visual-studio-code

Brian, J. (2021, February 2). *Is PHP Dead? No! At Least Not According to PHP Usage Statistics*. Kinsta. https://kinsta.com/blog/is-php-dead/

Domantas, G. (2019, November 25). *What is HTML?* . Hostinger Tutorials. https://www.hostinger.com/tutorials/what-is-html

Eric, A. (2021, January 11). *What Exactly Is A Full-Stack Developer? [2021 Guide]*. CareerFoundry. https://careerfoundry.com/en/blog/web-development/what-is-a-full-stack-web-developer/#4-what-is-full-stack-web-development

Figma. (n.d.-a). *About Figma, the collaborative interface design tool.* Retrieved February 17, 2021, from https://www.figma.com/about/

Figma. (n.d.-b). *Pricing for Figma*. Retrieved February 17, 2021, from https://www.figma.com/pricing/

GeeksforGeeks. (2019, September 4). *Introduction to Visual Studio*. GeeksforGeeks. https://www.geeksforgeeks.org/introduction-to-visual-studio/

GeeksforGeeks. (2021, March 9). *Frontend vs Backend*. GeeksforGeeks. https://www.geeksforgeeks.org/frontend-vs-backend/

Host Shopper. (n.d.). *What is Web Programming?* Host Shopper. Retrieved March 20, 2021, from https://www.host-shopper.com/what-is-web-programming.html

ifj.edu.pl. (n.d.). *A brief history of PHP*. Retrieved February 17, 2021, from

https://www.ifj.edu.pl/private/krawczyk/php/intro-history.html

Interaction Design Foundation. (n.d.). *What are Design Guidelines?* Interaction Design Foundation. Retrieved February 17, 2021, from https://www.interaction-design.org/literature/topics/design-guidelines

Jani, T. (2016, June 30). *Mikä on TypeScript?* Symfony. https://symfony.fi/artikkeli/mika-on-typescript

JavaScript.info. (n.d.). *An Introduction to JavaScript*. Retrieved February 17, 2021, from https://javascript.info/intro

Laine Tuulikki. (2015, September 23). *Mitä markkinoijan tulee ymmärtää web-ohjelmoinnista*. Dagmar. https://www.dagmar.fi/verkkopalvelukehitys/mita-markkinoijan-tulee-ymmartaa-web-ohjelmoinnista/

Scott, M. (n.d.-a). *What is CSS*. Skillcrush. Retrieved February 17, 2021, from https://skillcrush.com/blog/css/

Scott, M. (n.d.-b). *What Is PHP?* Skillcrush. Retrieved February 17, 2021, from https://skillcrush.com/blog/php/

Stack Overflow. (n.d.). *Stack Overflow Developer Survey 2019*. Stackoverflow. Retrieved February 17, 2021, from https://insights.stackoverflow.com/survey/2019

W3. (n.d.-a). *HTML & CSS*. W3C. Retrieved February 17, 2021, from https://www.w3.org/standards/webdesign/htmlcss.html

W3. (n.d.-b). *Verkkosisällön saavutettavuusohjeet (WCAG) 2.1*. Retrieved February 17, 2021, from https://www.w3.org/Translations/WCAG21-fi/

W3Techs. (n.d.). *Usage Statistics and Market Share of PHP for Websites*. W3Techs. Retrieved February 17, 2021, from https://w3techs.com/technologies/details/pl-php

Liite 1: Aineistonhallintasuunnitelma

Kehitysprojektin aikana pidetään päiväkirjaa, johon kerätään teknistä tietoa projektista. Tämä tieto analysoidaan opinnäytetyötä varten. Päiväkirjaa säilytetään tekijän tietokoneen C-asemalla, ja siitä tehdään säännöllisesti varmuuskopioita pilveen. Päiväkirjaa säilytetään C-asemalla ainakin vuoden verran opinnäytetyön valmistumisesta. Opinnäytetyöprojekti ei sisältänyt luottamuksellista tai arkaluonteista materiaalia.