Paavo Helin

Kuvien jakopalvelun luominen

Metropolia Ammattikorkeakoulu
Insinööri (AMK)
Tieto- ja viestintätekniikka
Insinöörityö
8.5.2023

Tiivistelmä

Tekijä:	Paavo Helin
Otsikko:	Kuvien jakopalvelun luominen
Sivumäärä:	33 sivua
Aika:	8.5.2023
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikka
Ammatillinen pääaine:	Ohjelmistotuotanto
Ohjaajat:	Lehtori Simo Silander

Tässä opinnäytetyössä luotiin kuvien jako- ja selaussivusto, käyttäen JavaScriptin sovelluskehystä, Next.js:ää ja Firebasen pilvipalveluita. Tarkoituksena oli esittää kuinka monet suositut nettisivut toimivat samoilla periaatteilla ja kuinka yksinkertaista samantyylisen nettisivun luominen on.

Projektin ideaksi valittiin kuvien jako- ja selaussivusto, koska se teknisesti sisältää kaikki tärkeimmät osat, joiden pohjalta voi tehdä monia muita erilaisia nettisivuja esim. blogit, tiedostojen säilytys- ja jakopalvelut, Instagram tai Pinterestin tapaiset sivut. Edellä mainitut kaikki esimerkit sisältävät ja enimmäkseen perustuvat käyttäjätietojen hallintaan ja kuvien sekä tekstin tallentamiseen ja esittämiseen.

Sovelluksen teknologiat valittiin niiden suuren suosion vuoksi. Ohjelmoinnissa sovelluskehyksenä käytettiin Next.js:ää, joka on tämän hetken suosituin React-pohjainen sovelluskehys. Firebase on myös yksi suosituimmista verkkosovellusten kehysalustoista. Vaikka Firebase tunnetaan tietokantapalveluna, se myös tarjoaa monia muita palveluita, joista tässä työssä käytettiin Realtime Database, Cloud Storage ja Authentication. Lopussa sovellus isännöitiin Next.js:n kehittäjäyrityksen, Vercelin, palvelimilla.

Avainsanat: Firebase, Next.js, Vercel

Abstract

Author:	Paavo Helin
Title:	Creating Image Sharing Site
Number of Pages:	33 pages
Date:	8 May 2023
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communications Technology
Professional Major:	Software Engineering
Supervisors:	Simo Silander, Senior Lecturer

The goal of the study was to create an image-sharing site, using the JavaScript framework, Next.js, and Firebase cloud services. The purpose was to show how many popular websites work on the same basis and how easy it is to create one with similar features.

An image sharing site was chosen as the idea of the project, because it technically contains most of the same key features as several other websites, such as blogs, file-storage and management sites, Instagram, or Pinterest. The above-mentioned sites all contain and are mostly composed of features such as user data management and storage as well as presentation of images and text.

The technologies for the project were chosen because of their popularity. Next.js was chosen as the framework for the frontend of the project. Next.js is currently the most popular React-based framework. Firebase is also one of the most popular web application development platforms. Although Firebase is known as a database service, it also offers many other services, of which Realtime Database, Cloud Storage, and Authentication were used in the current project. Once the project was finished, it was hosted on the servers of Vercels, the developer of Next.js.

Keywords: Firebase, Next.js, Vercel

Sisällys

Lyhenteet

BaaS: Backend as a Service. Yritysten tarjoamat backend-pilvipalvelut.

CSS: Cascading Style Sheets. CSS on erityisesti verkkosivuille kehitetty tyylisivu.

DOM: Document Object Model. DOM määrittelee asiakirjojen loogisen rakenteen ja tavan, jolla asiakirjaa käsitellään.

HTML: Hyper Text Markup Language. HTML on Web-sivujen standardi merkintäkieli.

JSON: JavaScript Object Notation. JSON on kevyt tiedonsiirtomuoto, joka on ihmisille helppo lukea ja kirjoittaa sekä koneiden jäsentää ja luoda.

JSX: JavaScript XML. JSX mahdollistaa HTML-koodin kirjoittamisen suoraan JavaScript-koodiin.

MVC: Model View Controller. Ohjelmistoarkkitehtuuri, jonka tarkoituksena on käyttöliittymän erottaminen kolmeen eri komponenttiin eli malli, näkymä ja käsittelijä.

NoSQL: Not only SQL tai non-SQL. Perinteisestä relaationmallista poikkeava tietokanta. Nämä tietokannat eivät seuraa mitään kiinteästi määrättyä taulukkoskeemaa.

npm: Node Packet Manager. npm on Node.js:än paketinhallintaohjelmisto.

npx: Node Package eXecute. npx on npm:än niin sanottu paketinkuljettaja.

SPA: Single Page Application. SPA on verkkosovellustoteutus, joka lataa vain yhden verkkoasiakirjan ja päivittää sitten kyseisen asiakirjan runkosisällön.

SSR: Server-Side Rendering. SSR mahdollistaa JavaScriptin renderöinnin HTML:äksi serverillä.

XML: Extensible Markup Language. XML on datan säilytykseen ja siirtämiseen käytetty merkintäkieli.

1 Johdanto

Tämän opinnäytetyön tavoitteena on selvittää ja esittää, miten saa yksinkertaisesti ja kätevästi luotua mahdollisimman kattavan esimerkin Reactiin pohjautuvasta Next.js:än mahdollisuuksista, sekä esittää kuinka monet nettisivut ja sovellukset perustuvat samoista perusaisoista. Opinnäytetyön tarkoituksena oli myös tutustua Next.js:ään sekä Googlen Firebase-tietokantaan ja sen tuomiin uusiin mahdollisuuksiin.

Aiheeksi on valittu kuvien jako- ja selaussivuston, koska se teknisesti sisältää kaikki tärkeimmät osat, joiden pohjalta voi tehdä monia muita erilaisia nettisivuja esim. blogit, tiedostojen säilytys- ja jakopalvelut, Instagram tai Pinterest tapaiset sivut. Edellä mainitut esimerkit sisältävät ja enimmikseen perustuvat käyttäjätietojen hallinnasta ja kuvien sekä tekstin tallentamisesta ja esittämisestä.

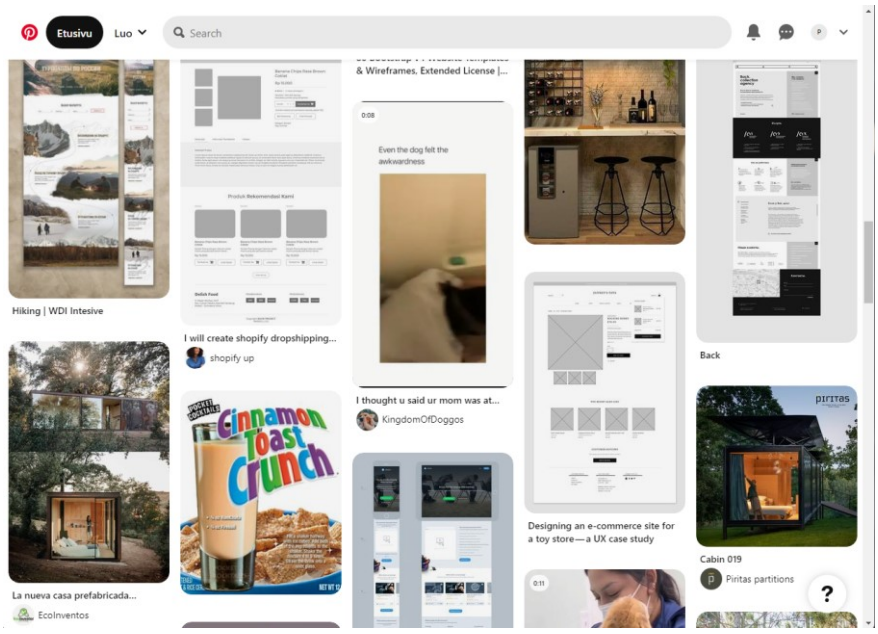
2 Sovelluksen suunnittelu

2.1 Sovelluksen perusidea

Opinnäytetyön tavoitteena on luoda kuvien jako- ja selaussivusto, joka sisältää yleisimmät näkökohdat, joita monet erilaiset nettisivut omaavat.

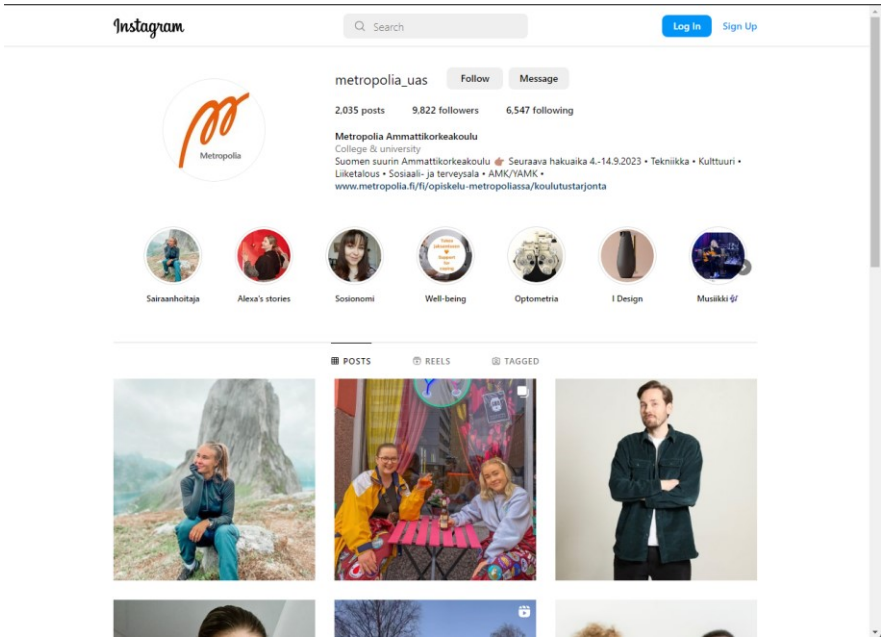
Käytän kolmea tunnettua nettisivua esimerkkeinä perustelussani ja oman sivuston suunnittelussani.

Ensimmäinen esimerkki on Pinterest. Se on ilmoitustaulutyyppinen, sosiaalinen linkkien ja kuvien jakopalvelu. Pinterest on maaliskuussa 2023, maailman 28:nneksi vierailluin nettisivu [1].



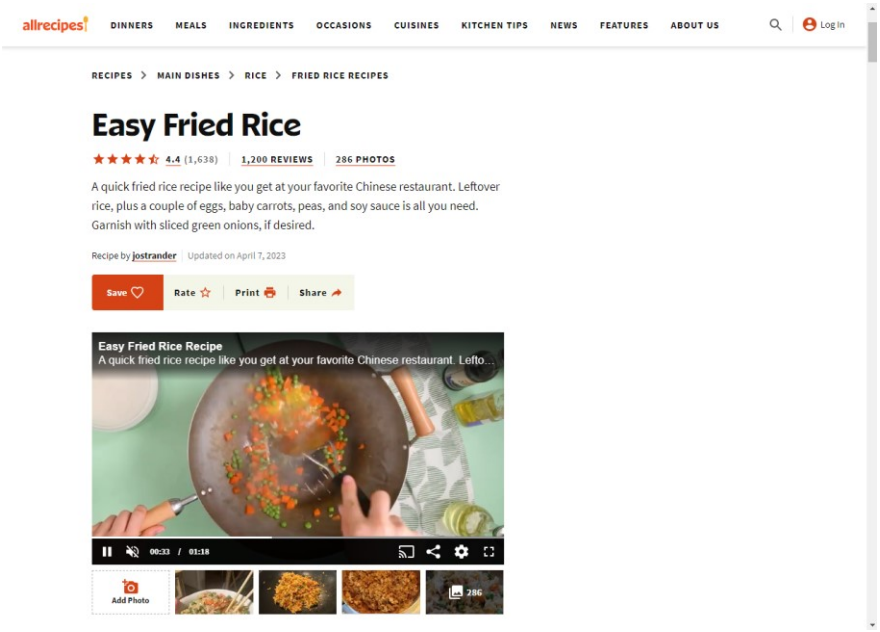
Kuva 1. Pinterestin kotisivu.

Seuraava esimerkki on Instagram. Instagram on Meta Inc. (aiemmin Facebook) omistama sosiaalinen media palvelu, kuvien ja videoiden jakamiseksi käyttäjiensä kesken. Instagram on maaliskuussa 2023, maailman viidenneksi vierailluin nettisivu [1].



Kuva 2. Metropolian Instagram-sivu.

Kolmas esimerkki on allrecipes.com. allrecipes.com on blogimainen, ruuanlaittokeskeinen sosiaalinen media. allrecipes.com on maaliskuussa 2023 maailman toisiksi vierailluin ruuanlaittoaiheinen nettisivu. [2]



Kuva 3. allrecipes.com ”Easy Fried Rice” julkaisu.

Jokaista näitä kolmea esimerkkisivustoa yhdistää se, että ne esittävät kuvaa tai/ja videota sekä niihin liittyvät tekstit ja käyttäjätiedot. Käyttäjä myös pystyy luoda omat käyttäjätunnukset palveluun ja sitä mukaan luoda omia ilmoituksiaan sekä hakea ja selata muiden käyttäjien ilmoituksia.

2.2 Käyttäjävaatimukset

Tässä luvussa määritellään mitä käyttäjän pitää pystyä tehdä sovelluksessa.

Käyttäjä voi luoda sovellukseen oman käyttäjätilin, johon kirjaudutaan salasanalla tai käyttäjä voi kirjautua omilla Google tilin tunnuksilla.

Käyttäjätilin omaava käyttäjä voi julkaista kuvia ja siihen liittyvää tietoa.

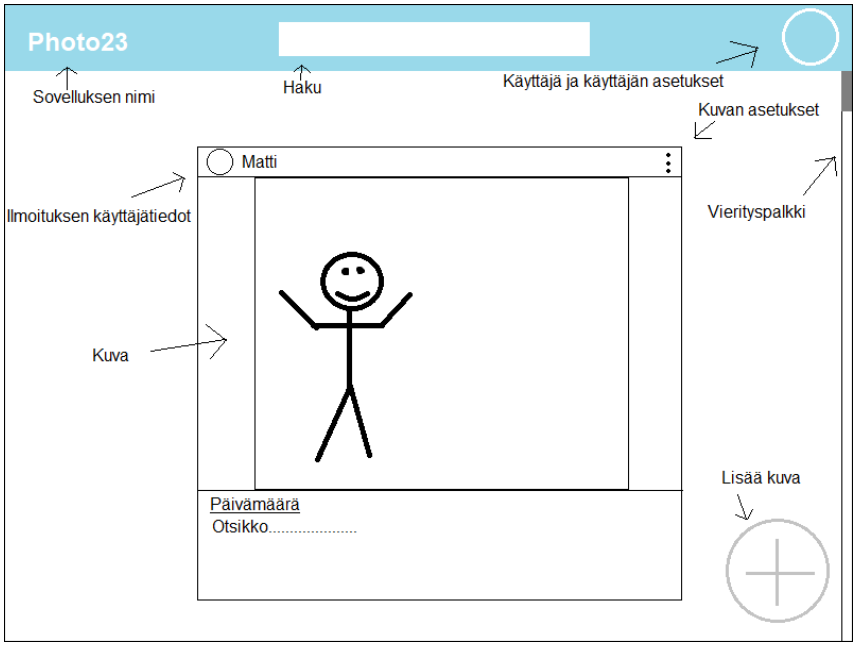
Käyttäjä, joka on julkaissut kuvan/kuvia voi muokata tai poistaa omia julkaisujaan.

Käyttäjä voi selata muiden käyttäjien julkaisemia kuvia ja lukea niihin liittyvää tietoa.

2.3 Käyttöliittymä

Sovellus avautuu sisäänkirjautumissivulle, josta löytyy myös linkki käyttäjätunnusten luontisivulle.

Sisään kirjauduttua, siirtyy kotisivulle, jossa on listattuna kaikki sovellukseen julkaistut julkaisut.



Kuva 4. Sovelluksen kotisivun luonnos.

Kuvassa 4. näkyy sovelluksen kotisivujen luonnos. Luonnoksessa on ilmaistu, mitä missäkin kohtaa näkymää halutaan saada aikaiseksi.

Yläpalkissa on vasemmalla sovelluksen nimi, keskellä hakukenttä ja oikeassa reunassa käyttäjän avatar, josta painamalla tulee esille valikko, jossa on oma

profiili sekä uloskirjautuminen. Uloskirjatumista painamalla siirtyy sisäänkirjautumisnäkymään.

Julkaisut ovat keskellä näkymää ja niissä tulee ilmi kuva, kuvaan liittyvät tekstit, julkaisun päivämäärä sekä julkaisija. Julkaisun oikeassa yläkulmassa on valikkopainike, josta painamalla tulee esille valikko, jossa on julkaisun poisto ja muokkaus. Poistoa painamalla julkaisu poistuu näkymästä ja tietokannasta. Muokkausta painamalla käyttäjä siirtyy julkaisun muokkausnäkymään.

Oikean alakulman ”+” nappia painamalla pääsee uuden julkaisun teko-osioon.

2.4 Tietokannan rakenne

Tietokantana tässä työssä käytetään Firebase Realtime Databasea, joka on JSON-pohjainen reaaliaikainen tietokanta. Rakenteena käytetään yksinkertaista ratkaisua, jossa ”posts” on kaikki julkaisut, julkaisut erotellaan käyttäen ”Id”:tä ja jokaisella julkaisulla on luokat: Id, käyttäjä, kuva, luonti ajankohta ja otsikko.

```
{
    "posts": {
      "Id": {
        "Id": "Paavotestjpg",
        "User": "Paavo",
        "coverImage": "test.jpg",
        "dateCreated": 1609718400000,
        "title": "Tämä on testi julkaisu"
      }
    }
  }
```

Esimerkkikoodi 1. Testijulkaisun luonnin JSON-koodi.

Kotisivun julkaisusyöte hakee kaikki tietokannassa olevat julkaisut, muokkaus ja poisto hakee Id:tä vastaavat julkaisut ja korvaa tai poistaa haetun julkaisun

arvot, hakupalkki hakee kaikista julkaisuista, käyttäjät ja onnistuneen haun jälkeen näyttää kyseisen käyttäjän julkaisut, kotisivun julkaisusyötteessä.

3 Teknologioiden valitseminen

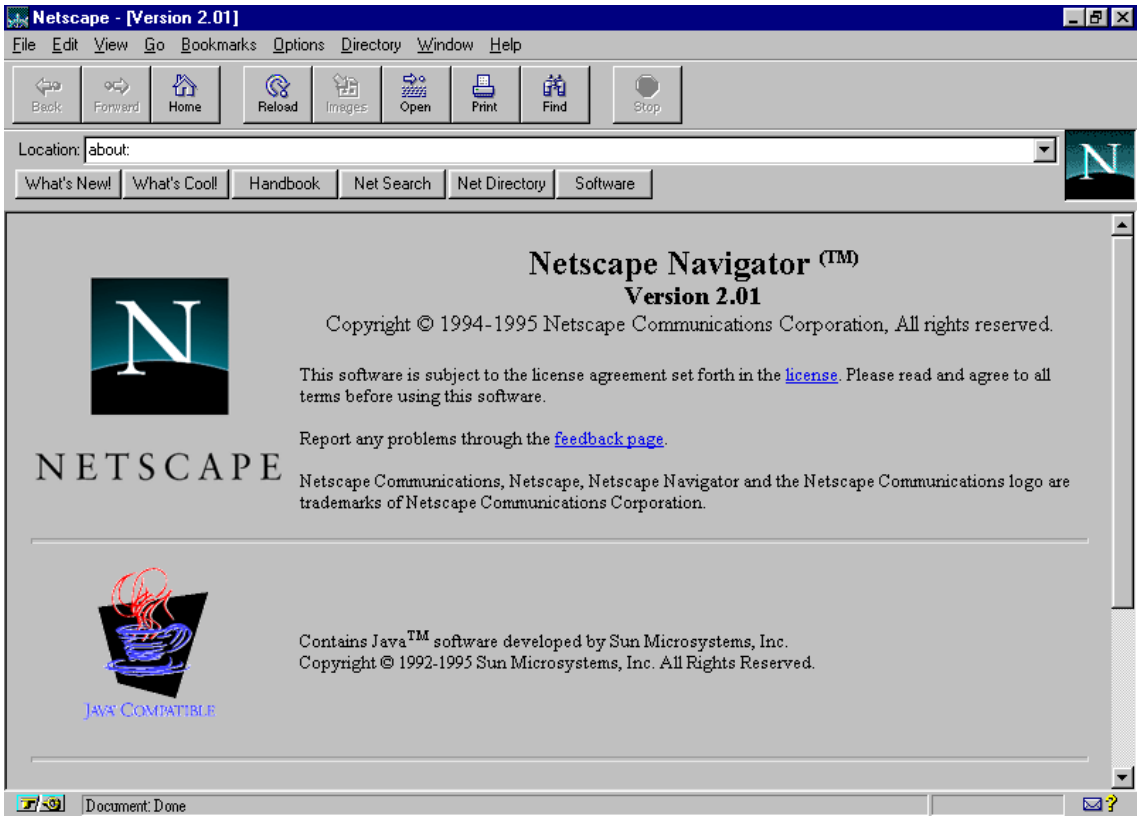
3.1 JavaScript

JavaScript (usein lyhennettynä ”JS”) on moniparadigmainen komentosarja- eli skriptikieli. JavaScript on yksi käytetyimmistä skriptikielistä ja se on parhaiten tunnettu verkkosivujen käytössä, mutta sitä käytetään myös monissa selaimen ulkopuolisissa tarkoituksissa esimerkiksi Node.js, Apache CouchDB ja Adobe Acrobat [3]. JavaScriptin paradigmoja ovat prototyyppipohjainen, imperatiivinen, funktionaalinen, olio-ohjelmointi, yksisäikeinen, dynaaminen kieli [4].

NCSA Mosaic loi vuonna 1993 maailman ensimmäiseksi suosituksi selaimeksi kutsutun selaimen ”Mosaic”.Mosaic:illa oli innovatiivinen graafinen käyttöliittymä, joka teki World Wide Web:in käytöstä helppoa ja helpommin saatavilla olevaa keskivertohenkilölle. Tämän myötä alkoi 90-luvun IT-kupla. Mosaic:in kehitystiimin johtaja Marc Andreessen aloitti oman yrityksensä ”Netscape”. Marc ja moni entinen Mosaic ohjelmoija loi Netscape:lla oman Mosaicin inspiroiman selaimen ”Netscape Navigator” vuonna 1994. Navigatorista tuli nopeasti maailman suosituin selain. Vuoden 1994 lopussa, Navigatorin osuus selainmaailmasta oli melkein 60 % [5].

Tähän aikaan kaikki sivun toiminnallisuus piti mennä serverin kautta ja niinkin yksinkertainen toiminto kuin lomakkeen täytön oikeellisuustarkastus tehtiin palvelimella. Tämä loi paljon ongelmia, joten idea luoda webille oma skriptikieli, syntyi. Vuonna 1995 Netscape palkkasi Brendan Eich lisäämään Scheme-tuen (Lispin inspiroima ohjelmointikieli) Netscapen Navigator-selaimeen. Ennen kuin Brendan Eich aloitti työnsä, Netscape ja Sun Microsystems kävivät keskusteluja Java-tuen lisäyksestä Netscapen Navigator-selaimeen. Tämä johti kiistelyyn, miksi tarvitaan kaksi ohjelmointikieltä, Java ja jokin muu skriptikieli. Vastauksena oli, että on tarve palvella kahta enimmäkseen hajallaan olevaa yleisöä,

komponenttien käsikirjottajat, jotka käyttivät C++:aa tai Javaa sekä ”skriptaajat”, jotka halusivat kirjoittaa koodia suoraan HTML:ään. Päädyttiin ideaan, että halutaan luoda uusi oma skriptikieli, joka muistuttaa Javaa. Netscape ohjelmoija Brendan Eich, loi JavaScriptin prototyypin 10 päivässä, toukokuussa 1995 [6]. JavaScript luotiin alkuperin nimellä ”Mocha”, joka muutettiin lokakuussa 1995 ”LiveScript” sopimaan paremmin Netscapen nimeämisrakenteen mukaan, lopulta ohjelmointi kielen ”Java” innoittamana, joulukuussa 1995 JavaScript sai lopullisen nimensä. JavaScript julkaistiin yleisöille vuonna 1995, integroituna Netscape Navigator 2.0-selaimeen [7] (kuva 5).



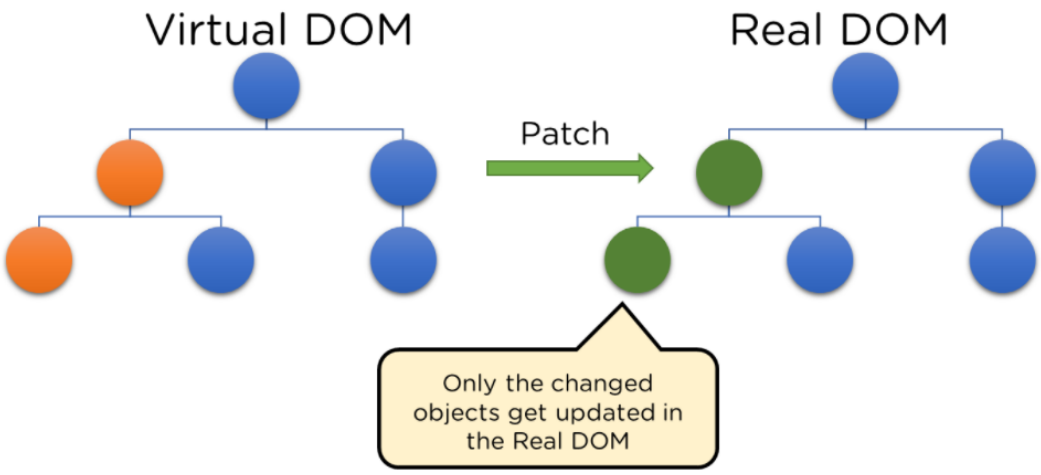
Kuva 5. Netscape Navigator versio 2.01:n kotisivu [8]

3.2 React

React, joka tunnetaan myös nimellä ”React.js”, on tämän hetken suosituin JavaScriptin frontend-sovelluskehys. React on Facebookin 2013 luoma sovelluskehys. Facebook alkoi kehittämään Reactia 2011, Jordan Walken johtamalla

tiimillä. Alun perin Reactin prototyypin nimi oli ”FaxJS” ja sitä käytettiin Facebookin uutissyötteessä.

Yksi Reactin keskeinen ominaisuus on JSX, joka mahdollistaa HTML:n kirjoittamisen suoraan JavaScript koodiin. React toi myös mukanaan virtuaalisen DOM:n, joka renderöi vain muuttuneen komponentin, koko sivun renderöinnin sijasta [9]. (kuva 6)



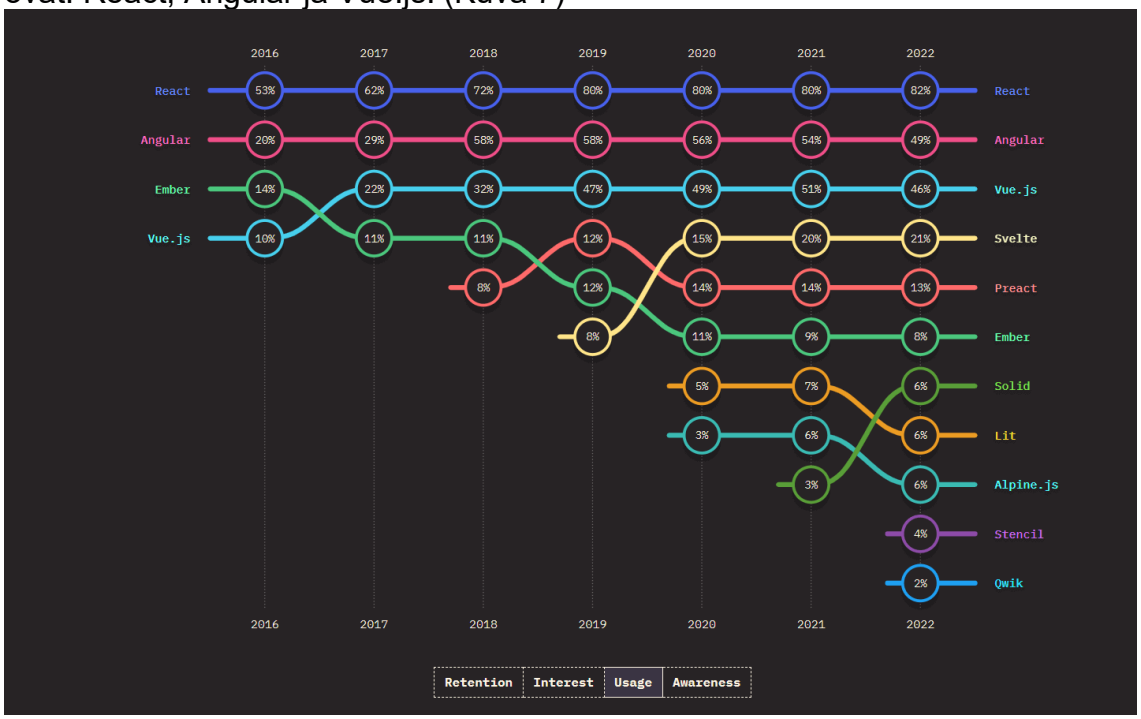
Kuva 6. Virtual DOM:in toiminta kuvattuna [10].

Jo yli puoli vuosikymmentä käytetyimpänä sovelluskehyksenä ollut React on kevyt, nopea ja yksinkertainen. Sillä on myös GitHubissa isoin yhteisön kannatus ja Facebookin iso panostus. Reactista on myös mobiilisovelluksien luontiin tarkoitettu sovelluskehys ”React Native”.

Reactin isoimpana haittapuolena on se, että se ei ole täysimittainen sovelluskehys. Tämä tarkoittaa, että monimutkaisen käyttöliittymän luomiseen todennäköisesti tarvitset lisäksi muita kirjastoja kuten esimerkiksi Redux.

3.2.1 Vertailu muihin JavaScriptin sovelluskehyksiin

Tällä hetkellä kolme eniten käytettyä JavaScriptin frontend-sovelluskehystä ovat: React, Angular ja Vue.js. (Kuva 7)



Kuva 7. 2022 kyselyn tulokset käytetyimmistä JavaScript-sovelluskehyksistä [11].

3.2.2 Angular

Angular on Googlen kehittämä, TypeScriptiin perustuva, ilmainen ja avoimeen lähdekoodiin perustuva, täysimittainen verkkosovelluskehys. Sitä käytetään laajalti yhdensivun sovellusten luomiseen (SPA) Angular julkaistiin 2016, ja se on jatkoa 2010 julkaistulle AngularJS:lle [12].

Angularin markkinoille tulemisen alkuvaiheissa se kiinnitti monien kehittäjien huomion erinomaisten ominaisuuksiensa ansiosta ja sen ollessa teknologiajätin Googlen tukema. Yleisenä erehdyksenä monet sekoittavat Angularin Googlen vanhempaan sovelluskehykseen AngularJS:ään. Angularin ja AngularJS:n pääero on, että AngularJS käyttää JavaScriptiä, kun taas Angular käyttää TypeScriptiä [13]. TypeScript on JavaScriptin syntaktinen superjoukko, joka lisää

staattista ohjelmoimista. Tämä periaatteessa tarkoittaa sitä, että TypeScript lisää syntaksin JavaScriptin päälle, jolloin kehittäjät voivat käyttää tyyppejä [14].

Angularin vahvuuksiin kuuluu se, että se sisältää kaiken tarpeellinen yritystason sovelluksen kehittämiseen ja toisiksi suurin yhteisön panos GitHubissa tarkoittaa kehyksen varmaa ylläpitoa ja kehitystä.

Huonoina puolina Angularissa on, että se vaatii TypeScriptin osaamista. Angular on myös paljon isompi kooltaan verrattuina kevyempiin vaihtoehtoihin kuten React ja Vue.

3.2.3 Vue.js

Vue.js on avoimeen lähdekoodiin perustuva progressiivinen sovelluskehys. Vuea kutsutaan progressiiviseksi, koska Vuen kanssa voit kytkeä Vue-koodia olemassa olevan sovelluksen yhteen osaan ja laajentaa siitä halun ja tarpeen mukaan, eli Vue ei ole monoliittinen sovelluskehys. Vue on alun perin entisen Googlen työntekijän, Evan Youn kehittämä sivuprojekti vuonna 2014. Nykyään Vuea ylläpitää aktiivisesti tiimi kokopäiväisiä työntekijöitä, sekä joukko vapaaehtoisia jäseniä ympäri maailmaa [15]. Vuella ei ole samanlaista ison organisaation tukea, kuten Reactilla on Facebook ja Angularilla on Google, vaan Vue on sponsoreiden rahoittama ja yhteisön avustama.

Vuen hyvät puolet ovat sen alhainen oppimiskynnys ja se, että se tuo Angularin ja Reactin parhaat puolet saman katon alle, samalla myös koittamalla parantaa niitä. Vuea yleisesti pidetään näistä kolmesta helpoimpana oppia koska, koska se perustuu pitkälti Angularin ja Reactin parhaimpiin puoliin, ilman että tarvitsee esim. osata TypeScriptiä niin kuin Angularia käyttäessä tai JSX:ää niin kuin Reactissa.

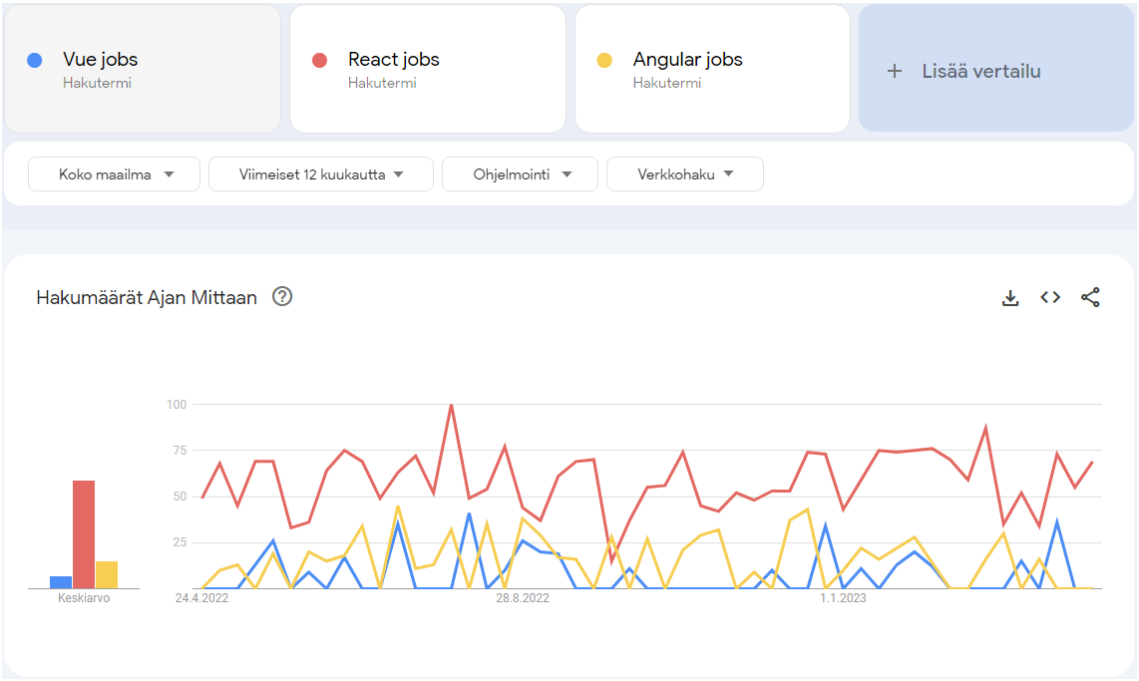
Vuen isoimpana huonona puolena tulee sen kannatus, Googlen ja Facebookin ansiosta, Angularilla ja Reactilla on parempi rahoitus, mikä tarkoittaa, että niitä kehitettään tasaisemmin ja varmemmin. Vuella ei ole suoranaisesti mitään isoa

yritystä rahoittamassa kehitystä, vain Vuen kehitys on pitkälti avoimen lähdekoodi yhteisön avustama, ja tämäkin yhteisö on alle kolmas osa siitä, mitä se on Reactissa ja Angularissa.

3.2.4 Miksi React

Opinnäytetyön tavoitteena on selvittää ja esittää, miten saa yksinkertaisesti ja kätevästi luotua mahdollisimman kattavan esimerkin mitä monet nettisivut ja sovellukset tarvitsevat toimiakseen. Reactilla on kattavat dokumentoinnit ja ”Getting Started guide” jotka auttavat sen oppimisessa, Reactilla on myös isoin määrä käyttäjiä, joiden kautta saada neuvoa ongelmissa. React ei myöskään vaadi TypeScriptin tai MVC:n käyttöä, vaikkakin kyllä sallii ne, toisinkuin Angular.

React on ollut nyt pitkään suosituin JavaScriptin frontend-sovelluskehys, eikä hidastumista näy. Yli 11 908 000 käynnissä olevaa verkkosivustoa käyttää Reactia [15] ja kuvassa 8 näkyy Google Trends-haku viimeiseltä 12kk, React työtehtäviä on haettu huomattavasti enemmän, keskiarvot per viikko Vue: 7, Angular: 15 ja React: 59 (”Numerot esittävät haun suosiota valitulla ajanjaksolla ja alueella suhteutettuna kaavion suurimpaan arvoon. Asteikon arvo 100 on alue, jolla termi oli suosituin. 50 on alue, jolla termillä tehtiin hakuja puolet vähemmän kuin 100 pisteen alueella, ja 0 on alue, jolla termistä ei ole saatavilla riittävästi tietoja.”) [16].



Kuva 8. Google Trends/tulokset Vue/React/Angular työtehtävistä 24.4.2022 – 15.4.2023 [16].

Eli valitsin Reactin koska se on suhteellisen helppo oppia, nopea, stabiili ja työtehtävien suhteen kysytyin.

3.3 Next.js

Next.js on Vercelin 2016 julkaisema, tämän hetken suosituin Reactiin perustuva sovelluskehys [17]. Koska Next.js perustuu Reactiin, siinä toimii kaikki Reactista tutut hyvät puolet ja tuo mukanaan parannuksia kuten staattiset ja palvelimen renderöimät sovellukset, jotka parantavat hakukoneoptimointia, automaattinen sivujen reititys, parannettuja CSS-tyyli vaihtoehtoja ja automaattinen kuvan optimointi. Next.js parannukset yleisesti yrittävät nopeuttaa sivujen latausta sekä vähentää kirjoitettavan koodin määrää.

3.4 Vercel (hosting)

Vercel (entinen Zeit huhtikuu 2020 asti) on 2015, Guillermo Raunchin perustama BaaS-yritys. Vercel on parhaiten tunnettu Next.js- kehittäjinä, mutta yritys

tarjoaa myös hosting-mahdollisuuksia, niin ilmaisena harrastaja-, kuin maksullisina pro- ja yritystasoina [18].

Koska Vercel on Next.js-kehittäjiä, on myös niiden hosting optimoitu Next.js-koodille, vaikkakin toimii kaikilla moderneilla sovelluskehyksillä.

Vaikkakin 3.6 luvussa kerrotulla Firebasella on myös oma hosting-palvelu ”Firebase Hosting”, se ei natiivisti tue Next.js:ässä käytettyä getServerSidePropsia (SSR). Koska tämän opinnäytetyön tarkoitus on esittää kuinka kätevästi luoda kuvien jako- ja selauspalvelu, käytän siksi Vercelin hosting-palvelua.

3.5 MUI

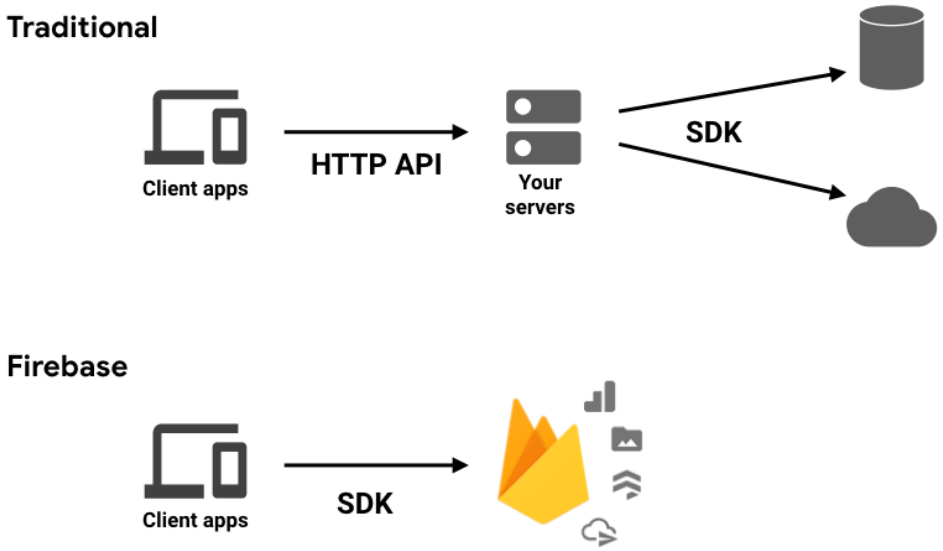
MUI (entinen Material UI, syyskuuhun 2021 asti), on vuonna 2014 perustettu avoimen lähdekoodin React-kirjasto erilaisia käyttöliittymäkomponentteja, jotka noudattavat Googlen Material Design-standardeja, sekä periaatteita. MUI:lla on yli neljä miljoonaa viikoittaista npm-latausta [19].

3.6 Firebase

Firebase sai alkunsa 2011, Envolve nimisenä yrityksenä, jonka perustivat Andrew Lee ja James Tamplin. Envolve tarjosi pilviviestintäalustaa, jota kehittäjät käyttivät reaaliaikaisiin Netcat-viesteihin. Myöhemmin samana vuonna viestintäalustan reaaliaikainen arkkitehtuuri päätettiin erottaa ja niin syntyi Firebase. Firebase lanseerattiin julkisesti huhtikuussa 2012. Firebasesta tuli osa Googlea, lokakuussa 2014 [20].

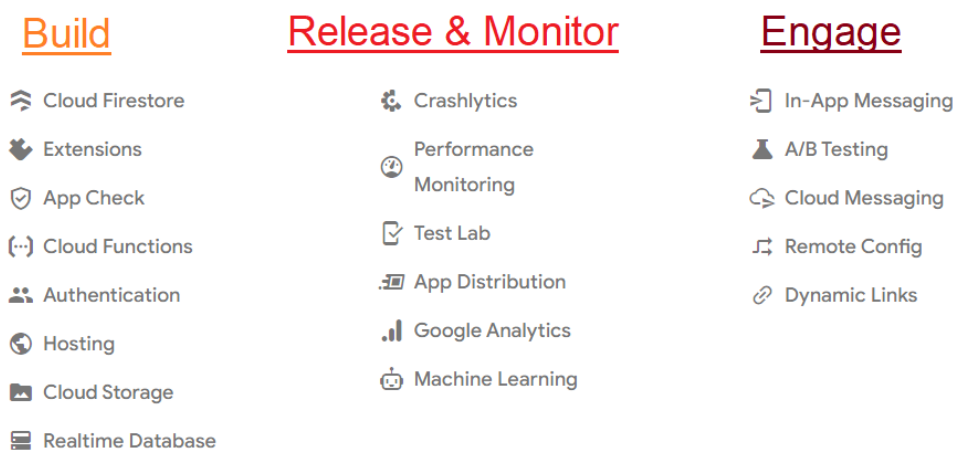
Firebase on Googlen tarjoama mobiili- ja verkkosovellusten kehysalusta. Se tarjoaa työkalut, jotka kehittäjien normaalisti pitäisi rakentaa itse (kuva 9). Firebase on reaaliaikainen pilvipalveluna toimiva NoSQL-tietokanta, joka tuo mukanaan paljon muutakin, kun vain tiedon tallennusta. Firebase korvaa

useimmissa tapauksissa sovelluksen normaalin backendin, sen sijaan tietokantakutsut tehdään suoraan asiakassovelluksesta Firebaseen.



Kuva 9. Miten Firebase korvaa perinteisen backendin ja tietokannat.

Firebase tarjoaa 18 eri tuotetta, joiden tarkoituksena on auttaa kehittäjiä ”luomaan, parantamaan ja kasvattamaan” sovelluksiaan. Firebasen-pilvipalvelut on suunnattu iOS-, Android- ja web-laitteille, mutta sillä on myös kasvava tuki Unity, C++ ja Flutterin kanssa. Firebase on luokitellut tuotteensa ”Build”, ”Release & Monitor” ja ”Engage”. Kuvassa 10 näkyy kaikki Firebasen tarjoamat tuotteet. Tässä opinnäytetyössä käytetään neljää näistä, eli Realtime Database, Cloud Storage, Authentication ja Hosting.



Kuva 10. Firebasen tarjoamat palvelut.

3.6.1 Realtime Database ja Cloud Firestore

Firebase tarjoaa kaksi eri tietokantaa, Realtime Database ja Cloud Firestore.

Realtime Database käyttää JSON-dokumentteja avainarvo-parien tallentamiseen. Se sisältää ominaisuuksia tietojen synkronointiin WebSocket-protokolaa käyttämällä ja asynkronista synkronointia offline-laitteiden tukemiseksi. Realtime Database ei rajoita datatyyppejä, vaan rajoitukset pitää itse asettaa.

Cloud Firestore on uudempi tietokantaversio, joka käyttää dokumenttikokoelmia tietojen tallentamiseen. Jokainen kokoelman dokumentti voi sisältää alikokoelmia tai tietokenttiä. Tämän rakenteen avulla voidaan tallentaa taulukoiden ja rivien kaltaisia tietoja, mikä laajentaa tietokannan yhteensopivuutta. Firestoressa sallitut datatyypit ovat boolean, oliot, taulukot, numerot, merkkijonot, tyhjä, geopisteet, aikaleimat ja matalat viittaukset. Dokumenttien määrityksessä voi käyttää viitteitä tietojen denormalisoinnin välttämiseksi vähentämällä tarvittavien kopioiden määrää. Lisäksi, vaikka kokoelmien välillä ei voi tehdä kyselyjä, voi sen sijaan käyttää viittauksia paikallisten kopioiden hakemiseen [21].

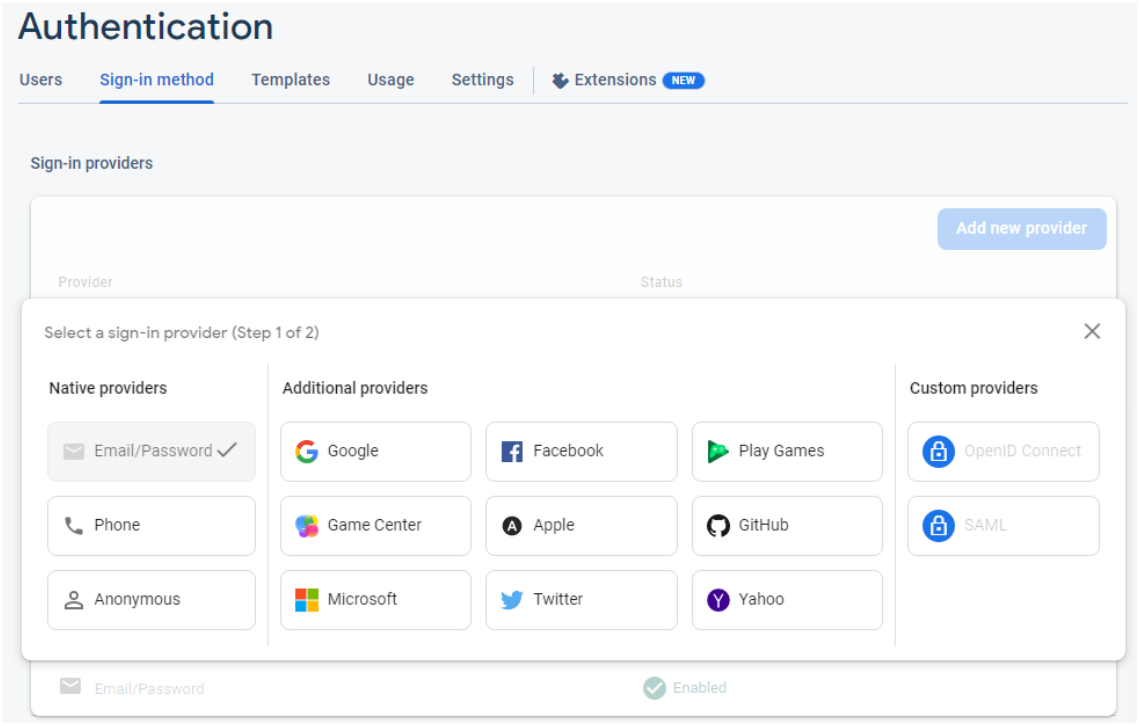
Vaikka kumpikin tietokantavaihtoehto olisi ollut sopiva tähän työhön, Realtime Databasen ominaisuudet kuitenkin vastasivat tarpeita hieman paremmin.

3.6.2 Cloud Storage

Cloud Storage on Firebasen tietokanta kuva-, video- ja äänitiedostoille sekä muulle käyttäjän sisällölle. Firebase Cloud Storage on Google Cloud Storagen varmuuskopioima. Tämä turvaa lataukset Firebasen välillä, verkkoyhteyden laadusta huolimatta [22]. Esimerkiksi jos verkkoyhteys katkeaa kesken latauksen, lataus jatkuu, kun verkkoyhteys palautuu. Cloud Storagen automaattisen skaalautuvuuden ansiosta voi käyttää yhtä samaa pilvipalvelua, vaikka olisi yksi lataus kuukaudessa tai miljardi.

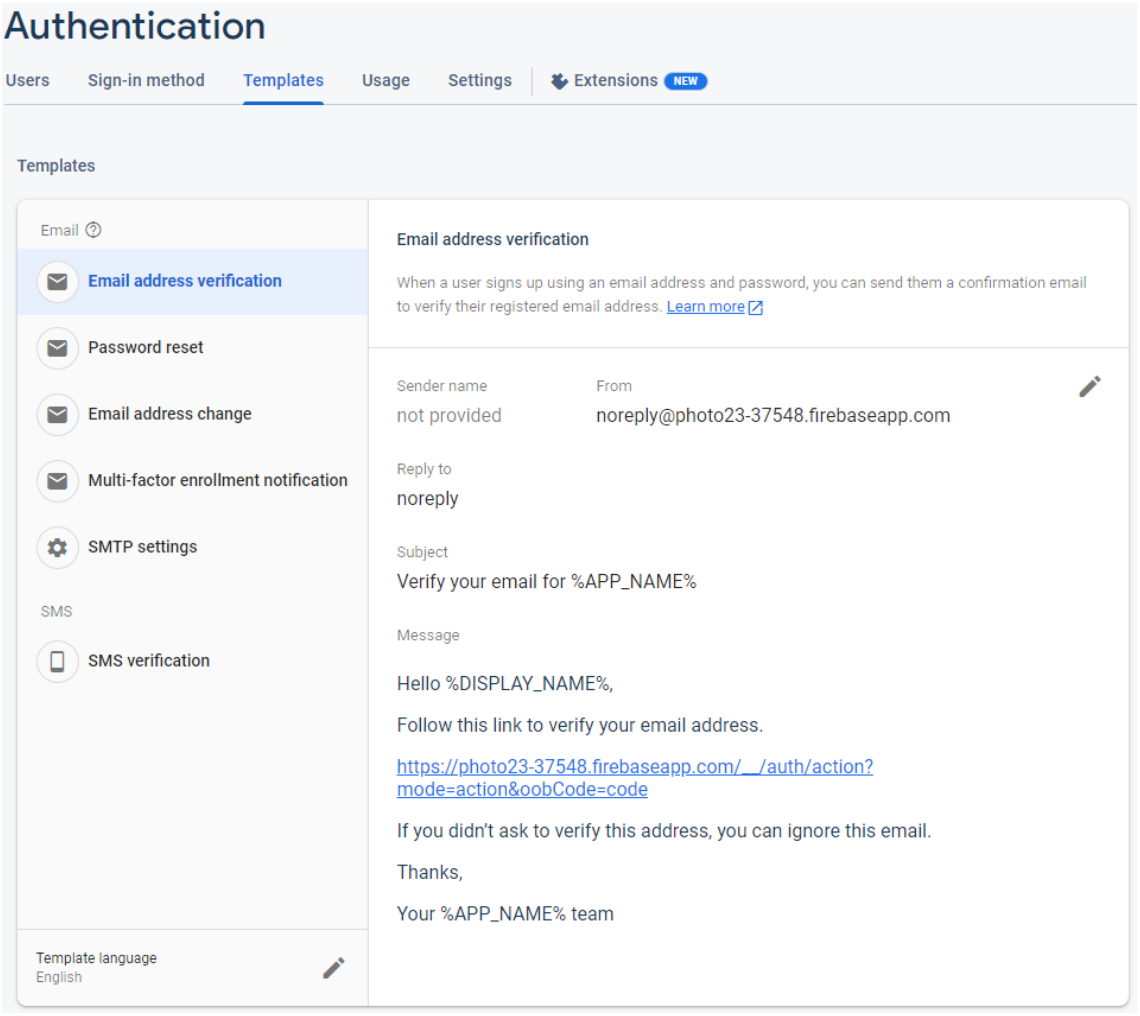
3.6.3 Authentication

Authentication on Firebasen luoma käyttäjänhallintajärjestelmä, jonka avulla kehittäjät voivat hallita käyttäjän todennusta Firebasen kautta. Authentication myös tarjoaa valmiita avoimen lähdekoodin kirjautumiskäyttöliittymiä. Authenticationin käyttöoikeuksia voi myös käyttää Firestoren, Realtime Databasen ja Cloud Storagen kanssa. Authentication kanssa on mahdollista kirjautua monella eri tavalla ja sen mukaan asettaa erilaisia käyttöoikeuksia esimerkiksi, että Anonymous voi lukea mutta GitHub-tunnuksilla kirjautunut voi lukea ja kirjoittaa. Kuvassa 11 näkyvät mahdolliset sisäänkirjautumisvaihtoehdot, joista ”Native providers” ovat vaihtoehtoja, jotka täytyy erikseen luoda, kun esimerkiksi jos Facebookilla kirjautuminen on päällä, voi kirjautu sisään omilla Facebook-tunnuksillasi.



Kuva 11. Authenticationin tarjoamat kirjautumisvaihtoehdot

Authenticationilla on myös mahdollista lähettää automaattisia sähköposti tai sms vahvistus tai tietojen muutos viestejä. (kuva 12)



Kuva 12. Authenticatorin automaattinen sähköpostin vahvistus.

4 Sovelluksen luominen

Uuden Next.js-projektin saa luotua npx:än avulla, suorittamalla komennon:

```
npx create-next-app photo23
```

Tämän jälkeen kehitysserverin saa päälle juuriluodun hakemiston juuresta komennolla:

```
npm run dev
```

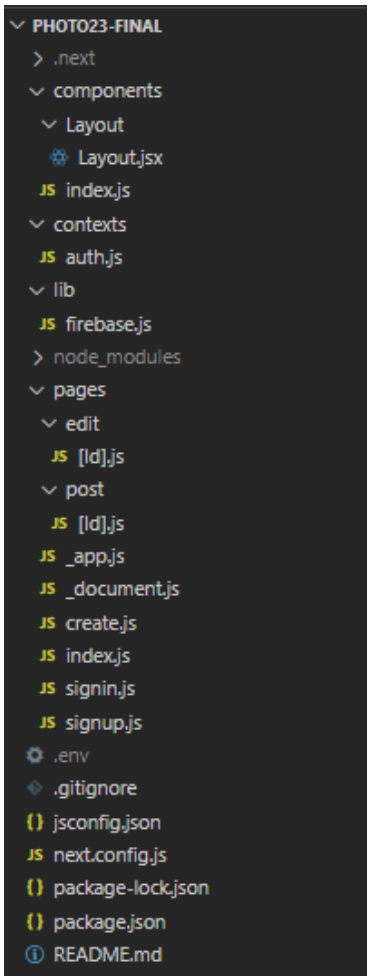
Komento käynnistää HTTP-palvelimen, jonka jälkeen sovellus on nähtävillä selaimessa, oletuksena osoitteessa localhost:3000.

Projekti vaatii myös Firebasen ja viiden eri MUI-kirjaston lataamista npm:ästä, komennolla:

```
npm install firebase @mui/material @mui/joy @mui/icons-material @emo-
tion/react @emotion/styled
```

4.1 Käyttöliittymä

Sovelluksessa on kuusi eri näkymää: kotisivu, sisään kirjautuminen, käyttäjän rekisteröinti, uuden julkaisun luominen, julkaisun muokkaus ja pelkkä yksittäinen julkaisu. Näillä kaikilla kuudella on omat JS-tiedostonsa, jotka käyttävät Next.js:n tiedostojärjestelmäpohjaista reititystä, eli Next.js automaattisesti luo reititykset JS-tiedostoille, jotka ovat ”pages”-kansion sisällä. Julkaisun muokkaus- ja yksittäinen julkaisusivu, käyttää myös Next.js:n dynaamista reititystä. Tämä reititystapa mahdollistaa [Id].js nimetyn tiedoston luomaan useita erilaisia sivuja, eri ”Id”-parametria käyttäen. Eli näin pystytään luomaan oman sivu jokaiselle julkaisulle, käyttäen julkaisun omaa tunnistetta. Kotisivu, uuden julkaisun luominen, julkaisun muokkaus ja pelkkä yksittäinen julkaisu, jakaa työkalurivin, jolla on oma JSX-tiedosto (Kuva 13).



Kuva 13. Sovelluksen kansiorakenne.

4.2 Tietokanta

Tietokantana käytetään Firebase Realtime Database. Jotta tietokannan saa yhdistettyä käyttöliittymään, täytyy aluksi alustaa Firebase sovelluksessa. Alla oleva koodi alustaa Firebase yhteyden, käyttämällä Firebasessa oman projektin ohjauspanelista löytyviä apiKey, DatabaseURL, ProjectId, storageBucket ja authDomain.

```
const initFirebase = async () => {
  if (!firebase.apps.length) {
    firebase.initializeApp({
      apiKey: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
      databaseURL: process.env.NEXT_PUBLIC_FIREBASE_DATABASE_URL,
      projectId: process.env.NEXT_PUBLIC_FIREBASE_PROJECT_ID,
      storageBucket: process.env.NEXT_PUBLIC_FIREBASE_STORAGEBUCKET,
      authDomain: process.env.NEXT_PUBLIC_FIREBASE_AUTHDOMAIN,
    })
  }
}
```

Esimerkkikoodi 2. Firebasen alustaminen.

Yhteyden luotua täytyy luoda functiot joilla haetaan ja luodaan julkaisut.

```
export const getPosts = async () => {

  initFirebase()

 const posts = await firebase
    .database()
    .ref('/posts')
    .orderByChild('dateCreated')
    .once('value')
    .then((snapshot) => {
      const snapshotVal = snapshot.val()

      const result = []
      for (var Id in snapshotVal) {
        const post = snapshotVal[Id]
        result.push(post)
      }

      return result.reverse()
    })

  return posts
}
export const createPost = async (post) => {
  initFirebase()

  post.dateCreated = firebase.database.ServerValue.TIMESTAMP

  console.log(post)

  return firebase.database().ref(`/posts/${post.Id}`).set(post)
}
```

Esimerkkikoodi 3. Julkaisujen hakeminen ja lisääminen.

Esimerkkikoodissa 3 haetaan kaikki julkaisut ja palautetaan ne käännetyssä järjestyksestä, jotta uusin, luotu julkaisu on ensimmäisenä. Uusi julkaisu

määritellään sovelluksessa ”create”-näkymässä (Kuva 20) ja lähetetään tietokantaan ”createPost”-functiolla.

```
{
  "rules": {
    ".read": true,
    ".write": "auth != null"
  }
}
```

Esimerkkikoodi 4. Realtime Databasen turvasäännöt.

Tietokannan turvasäännöt on asetettu niin että, kuka vaan voi lukea tietoa mutta vain sisään kirjautunut käyttäjä voi muokata.

4.3 Kuvien tallennus

Edellisessä luvussa alustettiin Firebase, joten sitä ei tarvitse enää luoda, mutta koska Storage on eri tietokanta kuin Realtime Database, tarvitsee se myös omat haku ja lisäys function.

```
export const uploadPic = async (pic, user) => {
  initFirebase()

  await firebase.storage().ref(`/${user.email}/${pic.name}`).put(pic)
  const url = await firebase. stor-
age().ref(`${user.email}`).child(pic.name).getDownloadURL()
  return url
}
```

Esimerkkikoodi 5. Kuvan lisääminen ja URL:n hakeminen.

Storageen lisättävän kuva määritellään myös ”create”-näkymässä (Kuva 20) ja lisätään Storageen esimerkkikoodi 5 olevalla uploadPic functiolla. Koska Realtime Database ei pysty tallentamaan kuvaa, täytyy ne tallentaa Storageen. Kuva saadaan lisättyä muuhun julkaisu dataan, hakemalla sen URL-osoite ja lisäämällä se julkaisuun. Siksi uploadPic functio palauttaa kuvan URL-osoitteen, Storageen lisäyksen jälkeen.

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Esimerkkikoodi 3. Storagen turvasäännöt.

Storagen turvasäännöt on myös asetettu niin että, kuka vaan voi lukea tietoa mutta vain sisään kirjautunut käyttäjä voi muokata.

4.4 Käyttäjien hallinta

Käyttäjän hallinta on hoidettu Firebasen Authenticationilla ja käyttäjä voi rekisteröityä sovellukseen ”Signup”-näkymän kautta, jolloin tiedot menevät Authenticationiin talteen. Käyttäjä pystyy myös kirjautumaan suoraan sisään sovellukseen käyttämällä olemassa olevia Google-tunnuksia. Authentication salaa käyttäjän salasanan koko ajan tietojen lähetyksestä asti, eikä edes sovelluksen kehittäjä pääse niihin käsiksi.

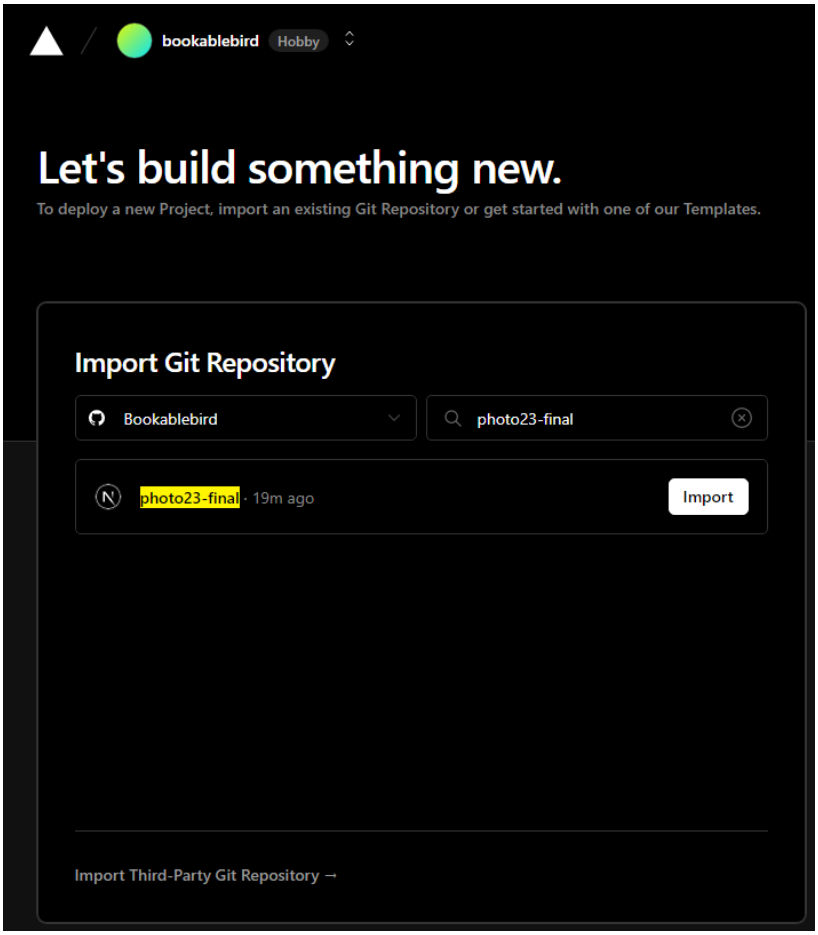
Realtime Databasen ja Storagen tietoturvasäännöt ovat asetettu yksinkertaisesti niin että kaikki pystyy lukemaan tietokannassa olevaa tietoa, mutta vain sisään kirjautunut käyttäjä pystyy muokata sitä.

4.5 Hosting

Hosting-palveluna tässä projektissa käytetään Vercelin tarjoamaa ilmaista Hobby-tason hosting-palvelua.

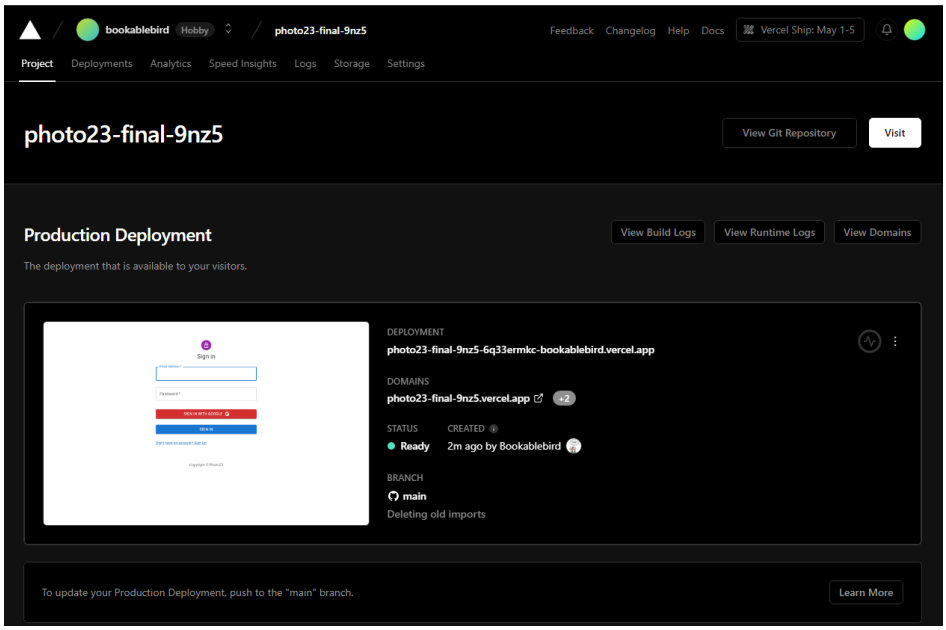
Projektin lataaminen Verceliin onnistuu helposti GitHubin kautta. Vercelin sivuille käyttäjätunnusten luotua, painamalla ”Add New…” -> ”Project”, tulee mahdollisuus kirjautua sisään GitHubiin ja lisätä sitä kautta olemassa olevan projektin. Vaihtoehtoisesti voi myös käyttää Vercelin tarjoamia malleja. GitHub

tunnuksilla kirjautumisen jälkeen tulee valikko kaikista käyttäjän GitHub arkistoista (Kuva 14).



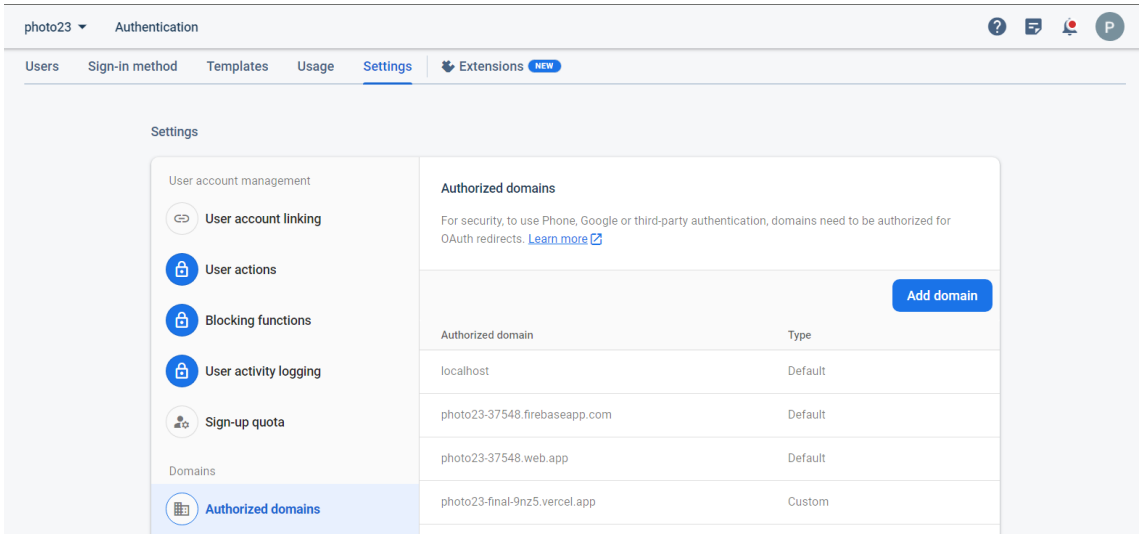
Kuva 14. GitHub projektin lisäys Verceliin.

Projektin lisäyksen jälkeen, Vercelin ohjauspaneli ilmoittaa mahdollisista virheistä tai luo projektille oman URL-osoitteen (Kuva 15), joka toimii ympäri maailmaa.



Kuva 15. Vercel projektin ohjauspaneli.

Viimeisenä vaiheena täytyy käydä Firebasen Authentication asetuksissa lisäämässä uusi Vercelin verkkotunnus, sallittujen listalle (Kuva 16).



Kuva 16. Firebase Authenticationin sallitut verkkotunnukset.

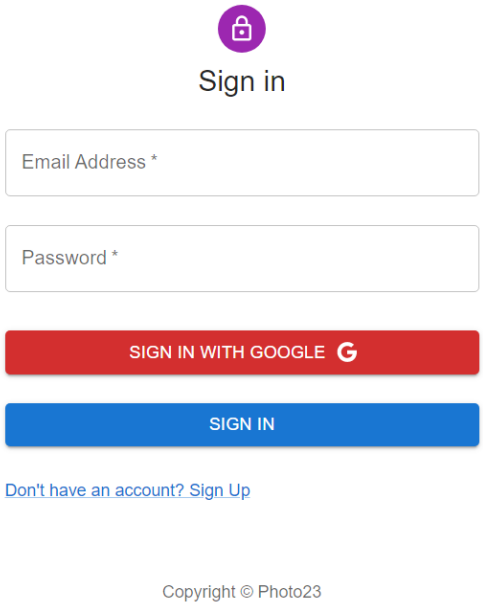
Tässä vaiheessa projekti toimii ja on aktiivisena verkossa, kaikille URL-tunnuksen omaaville.

5 Tulokset

5.1 Saavutetut tavoitteet

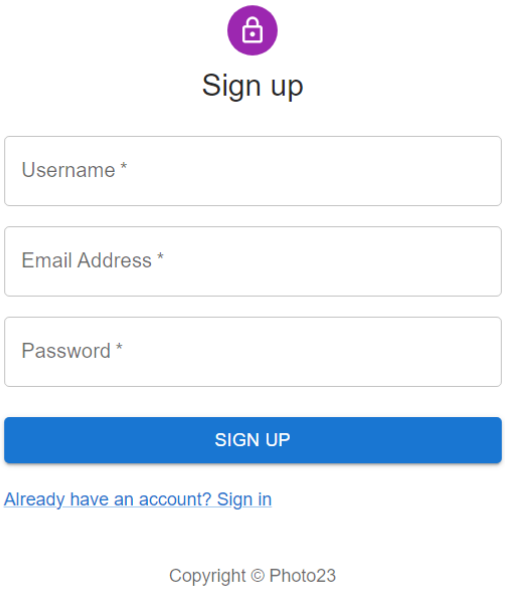
Opinnäytetyön tavoitteena oli luoda kuvien jako- ja selaussivusto, joka sisältää yleisimmät näkökohdat, joita monet erilaiset nettisivut omaavat ja samalla opetella käyttämään uusia teknologiota. Näihin tavoitteisiin päästiin tyydyttävästi, vaikkakin parannuksen varaa löytyy.

Luvussa 2.3 suunniteltiin mitä käyttöliittymän osia tulisi olla ja mistä niihin pääsee. Tähän tavoitteeseen päästiin suunnitellusti. Käyttöliittymä alkaa ”Sign in” näkymällä (Kuva 17).



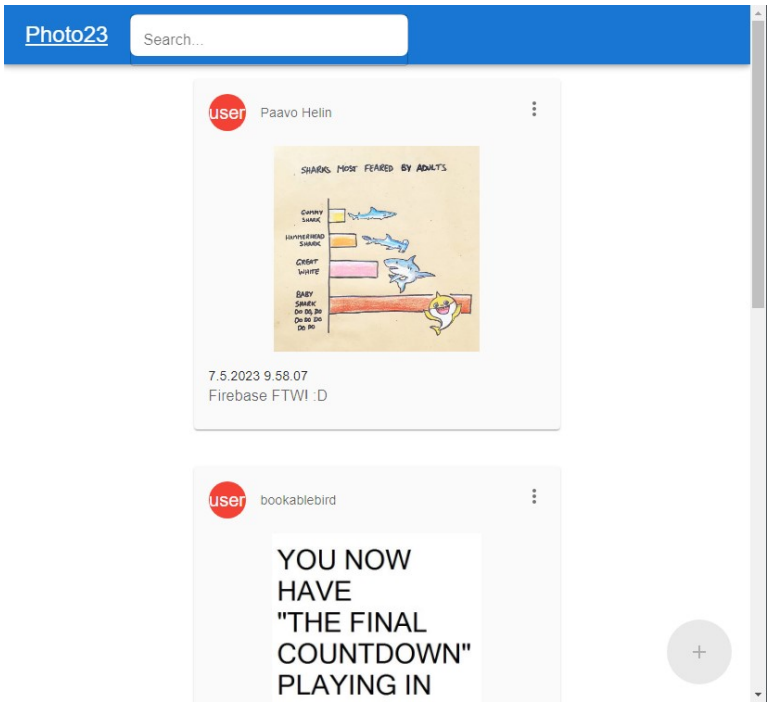
Kuva 17. "Sign in"-näkymä.

”Sign in”-näkymästä pääsee oikeilla tunnuksilla kirjautumalla kotisivulle ja jos ei ole vielä rekisteröitynyt käyttäjäksi pääsee ”Don’t have an account? Sign Up” linkki painamalla ”Sign up”-näkymään, jossa käyttäjä voi rekisteröityä (Kuva 18).



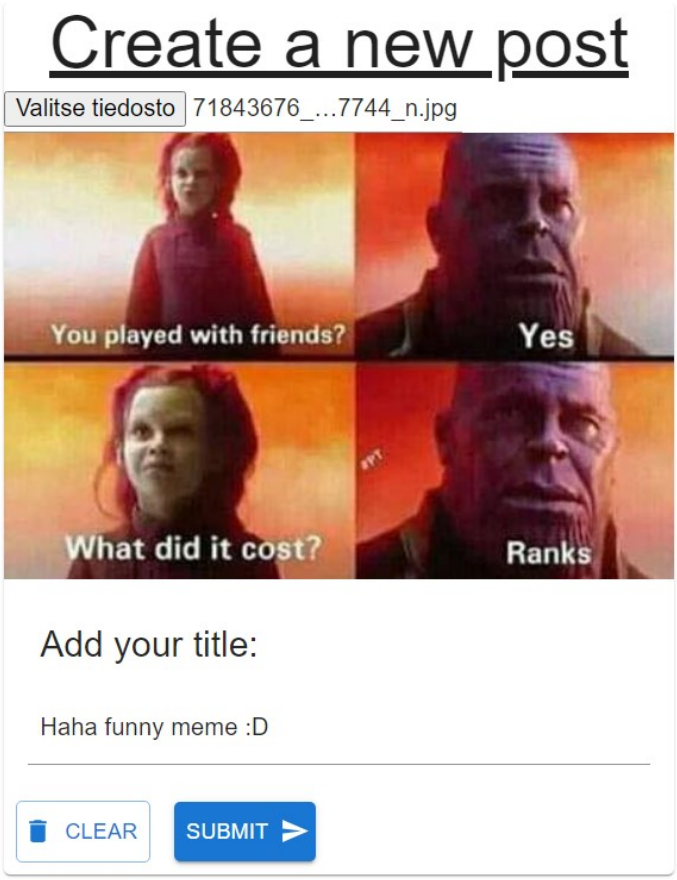
Kuva 18. ”Sign up”-näkymä.

Kuvassa 19, näkyy että lopullinen kotisivu näyttää melkein samalta kuin Kuva 4. esitetyssä kotisivun luonnoksessa.



Kuva 19. Valmiin sovelluksen kotisivu.

Myös muihin luvun 2.3 käyttöliittymän suunnitelmiin on päästy. Kuvassa 20 näkyy julkaisun luonti, jossa käyttäjä on lisännyt kuvan ja siihen liittyvän kommentin otsikoksi.



Kuva 20. Uuden julkaisun luomisnäkymä.

Julkaisun muokkaus näkymä on periaatteessa sama kuin uuden julkaisun luonti näkymä, ainoana erona näkymän otsikko ja että muokkaus näkymässä on valmiiksi ladattuna vanhan julkaisun tiedot (Kuva 21).

Kuva 21. Julkaisun muokkausnäkymä.

Next.js oli todella mukava parannus normaaliin Reactiin verrattuna ja siihen pääsi hyvin käsiksi keskivertoisten React kokemuksen pohjalta. Next.js:än kanssa oli myös omat ongelmansa lähinnä dokumentaation ja stackoverflow-keskusteluiden vähyyden takia, mutta koska Next.js on rakennettu Reactin päälle, pystyi monet ongelmat ratkaisemaan etsimällä Reactiin vastauksia. Yksi ylitsepääsemätön ongelma Next.js:än kanssa oli sen yhteensopivuus Firebase Hostingin kanssa. Sillä Firebase Hosting ei natiivisti tue Next.js:ässä käytettyä getServerSidePropsia (SSR), jouduin käyttämään muuta hosting-palvelua. Onneksi Next.js:n kehittäjäyritys Vercel tarjoaa hosting-palvelua, joka on optimoitu Next.js:n käyttöön. Oppikokemuksena Firebasen-palvelut olivat mielenkiintoisia ja kun pääsi alkuun, myös helppo käyttöisiä. Tulevaisuuden projekteissa aion käyttää kaikkia käyttämiäni teknologioita ja kokeilla lisää Firebasen-palveluita, joita en tässä työssä tarvinnut tai päässyt käyttämään.

5.2 Kehitysmahdollisuudet ja epäonnistumiset

Epäonnistumia tässä työssä ei varsinaisesti ollut muuta kuin Firebase Hostingin epäsopivuus. Kehitysmahdollisuuksia löytyy lukuisia. Koska tämän opinnäytetyön tarkoitus oli esittää, kuinka luoda yksinkertainen, mutta mahdollisimman kattava kuvien jako ja -selaussivusto, jäi myös sovelluksen sisältö hieman vähäiseksi.

Ensisijaisia kehityskohteita olisi julkaisuihin reagoimismahdollisuudet. Reagoimismahdollisuuksilla tarkoitan esimerkiksi kommentoiminen, jakaminen, tykkäys tai emojilla reagoiminen. Myös ulkoasua voisi parantaa ja tehdä sivusta persoonallisemman. MUI on kehitetty sitä varten, että sovellusten suunnittelu pysyisi yhdenmukaisena ja antaa siihen hyvät työkalut, joilla luoda teemat sovelluksille. Komponenttien vähäisyyden takia en luonut tähän työhön omaa teemaa, vaan käytin MUI:n oletusteemaa.

Muita kehitysmahdollisuuksia löytyy ihan sitä mukaa mihin suuntaan halutaan sovelluksen kanssa mennä. Jos halutaan luoda kilpailija instagramille, niin Firebaselta löytyy muun muassa In-App Messaging, jolla voi lisätä keskustelutoiminnon ja Analytics, jonka avulla voi ryhmittää käyttäjiä, jotta voi näyttää heille kiinnostavampaa sisältöä. Jos taas sovelluksesta halutaan enemmän blogimainen, niin sovelluksen kotisivun julkaisusyötteen voi muuttaa näyttämään yhtä isoa julkaisua. Niinpä uskon päässeeni tavoitteeseeni, luomalla yksinkertaisen ja kätevän sovelluksen, jolla on suuri kasvuvara, käyttäen esitettyjä teknologioita.

6 Yhteenveto

Tämä opinnäytetyö pyrki esittämään, millaisia vaiheita sisältyy yleisien sovelluksien kehittämiseen ja julkaisemiseen käyttäen suosittuja alustoja. Työn frontend tehtiin käyttäen Next.js:ää, joka on tämän hetken suosituin React.js:ään pohjautuva sovelluskehys. Frontendissä käytettiin myös avuksi MUI-kirjaston käyttöliittymäkomponentteja, jotka seuraavat Googlen Material Design-standardeja, sekä periaatteita. Backendissä käytettiin Firebasen pilvipalveluita, jonka ansiosta ei täytynyt luoda normaalia backendiä vain tietokantakutsut tehtiin asiakaspuolen sovelluksesta Firebasen tietokantoihin.

Valmiissa työssä käyttäjällä on mahdollisuus luoda omat käyttäjätunnukset tai kirjautua sisään käyttäen valmiita Google-käyttäjän tunnuksiaan. Sisään kirjauduttua käyttäjä pystyy luoda, muokata ja poistaa julkaisuja. Julkaisut koostuvat kuvasta ja siihen liittyvistä teksteistä sekä julkaisussa tulee myös ilmi, kuka on luonut julkaisun ja milloin. Käyttäjä pystyy myös selata kaikkien sovelluksen käyttäjien julkaisemia julkaisuja tai hakemaan yksittäisten käyttäjien nimen perusteella.

Projektiin lähdettiin keskinkertaisella React- ja MUI-kirjastojen osaamisella, mutta Vercelin Next.js sekä hosting ja Firebasen-palveluiden suhteen ei ollut aiempaa kokemusta. Lopussa päästiin tyydyttävään tulokseen ja lopputuloksena saatiin toimiva sovellus, vaikkakin parannuksen varaa löytyy. Opinnäytetyön ohjelmointiosuus antoi arvokasta kokemusta ja toi uutta intoa käyttämään lisää varsinkin Next.js:n ja eri Firebase-palveluita. Dokumenttiosuutta tehdessäni, käytin useita tunteja tehdessäni tutkimusta eri teknologioista ja niiden hyvistä ja huonoista puolista. Olen tyytyväin valitsemiini teknologioihin ja uskon tehneeni oikeat valinnat, mutta pidän tulevaisuudessa myös silmällä esimerkiksi hylkäämiäni JavaScript-sovelluskehyksiä Vue ja Angular.

Lähteet

1 Top Website Ranking. 2023. Verkkoaineisto. Similarweb. <https://www.similarweb.com/top-websites/>. Luettu 5.5.2023.

2 Top Website Ranking – Cooking and Recpes. 2023. Verkkoaineisto. Similarweb. <https://www.similarweb.com/top-websites/food-and-drink/cooking-and-recipes/>. Luettu 5.5.2023.

3 JavaScript. Verkkoaineisto. MDN Web Docs. <https://developer.mozilla.org/en-US/docs/Web/JavaScript> Luettu 22.4.2022.

4 JavaScript. Verkkoaineisto. Wikipedia. <https://fi.wikipedia.org/wiki/JavaScript> Luettu 22.4.2022.

5 Samuel E. Bodily & Sanjay Vakharia, Browser Wars: Microsoft Versus Netscape, 2009. Verkkoaineisto. SSRN. <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1422954> Luettu 22.4.2022.

6 Brendan Eich. Popularity. 2008. Verkkoaineisto. Brendan Eich. <https://brendaneich.com/2008/04/popularity/> Luettu 22.4.2022.

7 History and evolution of JavaScript. Verkkoaineisto. Exploringjs. <https://exploringjs.com/impatient-js/ch_history.html> Luettu 22.4.2022.

8 Netscape Navigator 2.0. Verkkoaineisto. Web Design Museum <https://www.webdesignmuseum.org/old-software/web-browsers/netscape-navigator-2-0> Luettu 22.4.2022.

9 Mehul Gadhiya. 2023. 20 Best JavaScript Frameworks for 2023. Verkkoaineisto. Lambdatest. <https://www.lambdatest.com/blog/best-javascript-frameworks/> Luettu 15.4.2023.

10 Mohit Joshi. 2022. Angular vs React vs Vue: Core Differences. Verkkoaineisto. BrowserStack. <https://www.browserstack.com/guide/angular-vs-react-vs-vue> Luettu 15.4.2023.

11 Chinmayee Deshpande. 2023. The Best Guide to Know What Is React. Verkkoaineisto. simplilearn. <https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs> Luettu 15.4.2023.

12 TypeScript Introduction. Verkkoaineisto. W3School. <https://www.w3schools.com/typescript/typescript_intro.php> Luettu 15.4.2023.

13 Mehul Gadhiya. 2023. 20 Best JavaScript Frameworks For 2023. Verkkoaineisto. Lambdatest. <https://www.lambdatest.com/blog/best-javascript-frameworks/> Luettu 15.4.2023.

14 Faq. Verkkoaineisto. Vue. <https://vuejs.org/about/faq.html> Luettu 15.4.2023.

15 React Usage Statics. 2023. Verkkoaineistoa. Verkkoaineisto. built with. <https://trends.builtwith.com/javascript/React>. Luettu 18.4.2023

16 Google Trends. Verkkoaineisto. <https://trends.google.com/trends/explore?cat=31&q=Vue%20jobs,React%20jobs,Angular%20jobs> Luettu 18.4.2023.

17 Maab Saleem. 2022. What is Next.js? Verkkoaineisto. ButterCMS. <https://buttercms.com/blog/what-is-nextjs/> Luettu 3.5.2023.

18 Julian Wallis. 2023. What Is VERCEL? Is It The Right Platform For Front-End Developers?, Verkkoaineisto. WEBO Digital. <https://webo.digital/blog/what-is-vercel-is-it-the-right-platform-for-front-end-developers/> Luettu 4.5.2023.

19 Manik Sharma. 2022. What is MUI and what do you need to know about it? Verkkoaineisto. TALENT500. <https://talent500.co/blog/what-is-mui-and-what-do-you-need-to-know-about-it/> Luettu 3.5.2023.

20 George Batschinski. What is Firebase? All the secrets unlocked. Verkkoaineisto. back4app. <https://blog.back4app.com/firebase/#Firebase_History> Luettu 2.5.2023.

21 Kobi Bohbot. Firebase Database: Should You Choose Realtime DB or Cloud Firestore? Verkkoaineisto. back4app. <https://blog.back4app.com/firebase-realtime-database-vs-cloud-firestore/> Luettu 2.5.2023.

22 Firebase, Verkkoaineisto. <https://firebase.google.com/> Luettu 2.5.2023