Yrityksen verkkosivujen suunnittelu ja toteutus – Case Unsettling Company

Anni Mankki

Haaga-Helia ammattikorkeakoulu

Tradenomi

Opinnäytetyö

2024

Tiivistelmä

Tekijä(t)
Anni Mankki

Tutkinto
Tradenomi

Raportin/Opinnäytetyön nimi
Yrityksen verkkosivujen suunnittelu ja toteutus – Case Unsettling Company

Sivu- ja liitesivumäärä
40 + 1

Opinnäytetyön tavoitteena oli suunnitella ja toteuttaa verkkosivut aloittelevalle taidetta tuottavalle yritykselle. Toimeksiantajana toimi Helsingissä vuonna 2023 perustettu Unsettling Company. Työn aloituksen aikaan Unsettling Company toimi pelkästään eri sosiaalisen median kanavilla, joten tarve omille verkkosivuille oli selvä. Työn tarkoituksena oli tarjota yritykselle vakaa ja persoonallinen alusta sisällön esittelemiselle.

Opinnäytetyön aihe rajattiin käsittelemään verkkosivujen suunnittelua ja toteutusta. Työstä jätettiin pois verkkosivujen julkaisu ja testaus. Näiden lisäksi sivuton visuaalinen ilme rajattiin aiheen ulkopuolelle ja suunnitteluosio kattoi sivuston sisällön ja ulkoasun rakenteen. Työn tietoperustassa keskitytään toteutusvaiheessa käytössä oleviin teknologioihin, joihin lukeutuvat muun muassa React ja Firebase. Niiden lisäksi työssä käydään läpi tietoperustaa käyttöliittymäsuunnittelun taustalla.

Tietoperustan ohella työssä esitellään kattavasti verkkosivujen suunnitteluprosessia. Suunnittelussa hyödynnettiin käyttöliittymäsuunnittelun periaatteita ja työpajaa. Työpajassa yrityksen jäsenet ideoivat eri työkalujen avulla verkkosivujen sisältöä ja ulkoasua. Lopulta verkkosivujen tärkeimmiksi ominaisuuksiksi valittiin listaus, jossa esitellään taidetta sekä yhteydenottolomake, jota kautta asiakas voi lähettää yritykselle sähköpostia. Niiden lisäksi sivustolla on kotisivu ja sivu, joka antaa lisätietoa yrityksestä. Verkkosivut toteutettiin React-ohjelmakirjaston avulla ja sivuston toiminnallisuuksissa hyödynnettiin Firebasen ja EmailJS:n palveluita.

Verkkosivujen koodi on valmiina seuraavaan vaiheeseen, joka on testauksen suunnittelu ja toteutus sekä sivuston julkaisu. Verkkosivut vastaavat tällä hetkellä hyvin toimeksiantajan toiveisiin ja ne sopivat hyvin brändille luotuun ilmeeseen. Tulevaisuudessa verkkosivut tukevat brändin näkyvyyttä ja sitä kautta yrityksen liiketoimintaa.

Asiasanat
React, Firebase, Verkkosivut, Käyttöliittymäsuunnittelu

Sisällys

1 Johdanto

Nykyaikana liike-elämässä digitaalisen läsnäolon merkitys on kiistaton. Ilman selkeää, saavutettavaa ja brändille sopivaa alustaa on yrityksen hankala löytää jalansijaa ja tavoittaa asiakkaitaan (Webflow, Inc. 2023). Tämän vuoksi opinnäytetyön aiheeksi valittiin projekti, jossa suunnitellaan ja toteutetaan verkkosivut taidetta tuottavalle yritykselle Unsettling Company. Opinnäytetyön toimeksiantaja, Unsettling Company, on aloitteleva taide- ja viihdealan yritys, joka tuottaa sekä digitaalista että perinteistä taidetta eri muodoissa. Yritys on perustettu Nelli Pöyryn ja Anni Mankin toimesta kesäkuussa 2023.

Tässä opinnäytetyössä syvennytään verkkosivujen suunnittelun ja toteutuksen eri vaiheisiin, hyödyntäen nykyaikaisia teknologioita ja suunnitteluperiaatteita. Aihe on ajankohtainen sekä yhteiskunnallisesti, että kohdeyrityksen kannalta. Aiheessa nivoutuu yhteen taide, teknologia ja liiketoiminta. Sen lisäksi projekti tarjoaa merkittävää hyötyä yrityksen liiketoimintaan. Yritys toimii tällä hetkellä pelkästään Instagramissa, joten verkkosivuille on selkeä tarve brändin näkyvyyden kasvattamiseksi.

Opinnäytetyöraportti koostuu tietoperustasta, kehittämisprojektin suunnitteluvaiheesta ja verkkosivujen toteutuksesta. Tekstin tukena työssä on kuvia suunnitteluprosessin eri vaiheista sekä verkkosivuilta toteutusvaiheesta. Niiden lisäksi teoriatietoa ja verkkosivujen rakentamista havainnollistetaan koodipätkillä, jotka on työssä nimetty Koodi 1., Koodi 2. jne.

Verkkosivujen suunnittelun apuna käytettään työpajoja, joissa yrityksen jäsenet suunnittelevat yhdessä verkkosivujen sisältöä ja ulkoasua. Työpajoissa ideoidaan ja hyödynnetään benchmarking-menetelmää, jossa tutkitaan muita vastaavia verkkosivuja ja niiden ulkoasua sekä toiminnallisuutta. Verkkosivujen rakennetta suunnitellaan myös piirroksilla, joiden avulla voi nopeasti vertailla erilaisia vaihtoehtoja. Yrityksen brändi-identiteetti suunnitellaan opinnäytetyönä toisen perustajajäsenen, Nelli Pöyryn toimesta. Brändi-identiteetti sisältää muun muassa brändin arvot, logon, fontit ja väripaletin.

1.1 Työn tarkoitus ja tavoite

Opinnäytetyön tarkoituksena on tarjota yritykselle vakaa ja persoonallinen alusta sisällön esittelemiselle. Kuten aiemmin mainittiin, on nykypäivänä liike-elämässä digitaalisen läsnäolon merkitys valtava ja ilman selkeää ja miellyttävää digitaalista alustaa on yrityksen hankala menestyä. Omien verkkosivujen kautta pystyy myös muita sosiaalisen median alustoja paremmin tuomaan esille omaa brändiä ja yritysilmettä.

Opinnäytetyön tavoitteena on saada konkreettinen lopputuotos – koodi verkkosivuille. Sivusto rakennetaan React-ohjelmakirjaston ja Visual Studio Code -ohjelman avulla. Ohjelman palvelinpuoli hallinnoidaan Firebasen tietokannan ja EmailJS-palvelun avulla. Näiden lisäksi verkkosivujen ulkoasun määrittelyssä hyödynnetään Material UI:n elementtejä.

Verkkosivujen on tarkoitus olla käyttäjäystävälliset, sekä brändiin sopivat. Yrityksen tavoitteena onkin saada enemmän oman näköinen väylä brändin esittelyä varten. Eri sosiaalisen median alustat ovat toki tehokas ja tärkeä keino seuraajakunnan kasvattamiseksi, mutta omien verkkosivujen kustomointimahdollisuudet tuovat kokonaan uuden ulottuvuuden oman brändin ja sanoman viestimiseen. Tulevaisuudessa verkkosivut tukevat brändin näkyvyyttä ja yrityksen liiketoimintaa.

1.2 Aiheen rajaus

Työn toiminnallinen osuus rajataan verkkosivujen suunnitteluun ja toteutukseen. Suunnitteluvaihe tulee perustumaan yrityksen jäsenten ideointiin ja tarpeiden kartoitukseen, sekä käyttöliittymäsuunnittelun teoriaan. Suunnittelussa ei huomioida käyttäjäkokemusta eikä siihen liittyen tehdä tutkimusta.

Toteutusvaiheessa luodaan verkkosivut, joiden kautta on mahdollista saada tietoa yrityksestä, tarkastella yrityksen tuottamaa taidetta ja ottaa yritykseen yhteyttä. Toteutusvaiheessa on tarkoituksena saada aikaan julkaisuvalmiit verkkosivut. Sivuston julkaisu ja testaus rajataan työn ulkopuolelle.

1.3 Keskeiset termit

DOM:	DOM eli Document Object Model kuvaa sitä, miten sivuston rakenne esitetään selkeästi. DOM esittää dokumentin eli koodin osasia hierarkkisesti ja sitä voidaan kuvailla puurakenteiseksi. (MDN 2024a.)
Komponentti:	Komponentti on yksittäinen ja uudelleenkäytettävä palanen koodia (W3Schools 2024a).
Elementti:	Elementti on koodin pienin rakennuspalanen. Elementtien avulla määritellään mitä näytöllä on näkyvissä. (React 2024b.)
Renderöinti:	Renderöinti on ikään kuin sovelluksen päivittyminen. Siinä koodi ja komponentit muuntuvat verkkosisällöksi. (React 2024c.)
Responsiivisuus:	Responsiivisuudella tarkoitetaan verkkosivun kykyä arvioida näytön koko laitteella, jolla verkkosivua katsellaan. Tämän perusteella

verkkosivu voi muuttaa ulkoasuaan ja asetteluaan toimiakseen hyvin esimerkiksi myös mobiilinäytöllä. (GeeksforGeeks 2024.)

Front-end: Front-end on ohjelmoinnissa se osa-alue, jonka kanssa käyttäjät ovat vuorovaikutuksessa. Se sisältää sovelluksen visuaalisen ja interaktiivisen puolen. (Simmons 2023.)

Back-end: Back-end on ohjelmoinnissa se osa-alue, joka luo näkymättömät rakenteet Front-endin takana. Back-end ohjelmointi keskittyy palvelimen puolelle. (Simmons 2023.)

API: API on ohjelmarajapinta, joka mahdollistaa eri ohjelmistojen tai palveluiden välisen tiedon siirron ja kommunikoinnin (Contentful 2023).

API-avain: API-avain on yksilöllinen tunniste, jonka avulla APIin kohdistuvat pyynnöt autentikoidaan (Contentful 2023).

Ympäristomuuttuja: Ympäristömuuttuja on muuttuja, jonka arvo nimetään ohjelman ulkopuolella. Ympäristömuuttuja muodostuu avain-arvo-pareista. (Medlock 2018.)

2 Tietoperusta

Seuraavissa kappaleissa esitellään verkkosivujen tärkeyttä yritystoiminnan kannalta, sekä opinnäytetyössä käytettyjen teknologioiden ja menetelmien tietoperustaa. Siihen sisältyy verkkosivujen rakentamiseen käytettävä kirjasto, React, sekä Reactin materiaalikirjasto, MUI ja muutama muu käytössä oleva työkalu. Joidenkin teoriaosuuksien yhteydessä aihetta esitellään myös käytännön esimerkkien avulla, yksinkertaisilla koodipätkillä. Lopuksi käydään vielä läpi teoriatietoa verkkosivujen suunnitteluprosessin taustalla.

2.1 Verkkosivujen merkitys

Nykyaikana digitaalisen läsnäolon merkitys yritystoiminnassa on valtava. Verkkosivustot ovat merkittävässä roolissa asiakkaiden tavoittamisessa ja yrityksen oman viestin toimittamisessa. Sivusto voi toimia sekä näyteikkunana että yhteydenottoväylänä yrityksen ja asiakkaan välillä. (Webflow, Inc. 2023.)

Verkkosivujen tärkeimpiä etuja ovat ammattimaisuuden esilletuonti, asiakasvuorovaikutuksen tehostaminen ja brändin näkyvyyden lisääminen. Näitä tukevat selkeät ja omaan brändiin sopivat verkkosivut. Niiden lisäksi sivustolla voidaan hyödyntää monenlaista datan analysointia ja esimerkiksi hakukoneoptimointia, jotta yrityksen näkyvyys verkossa kasvaa ja potentiaaliset asiakkaat löytävät sisällön paremmin. (Webflow, Inc. 2023.)

2.2 React

React on Facebookin kehittämä avoimen lähdekoodin ohjelmakirjasto, jonka avulla voidaan luoda yksinkertaisesti monipuolisia verkkosivuja eri tarkoituksiin. Reactin pääasiallinen tavoite on tarjota selkeä, tehokas ja joustava pohja verkkosivujen rakentamiselle. Näillä keinoin se pyrkii parantamaan ja suoraviivaistamaan ohjelmointikokemusta. Reactin joustavuus ja uudelleenkäytettävyys ovat olennaisessa roolissa sen suosion taustalla. Reactin ketterää toimintaa tukee sen kyky integroida monenlaisia kirjastoja ja työkaluja sujuvasti osaksi projekteja. Näin voidaan helposti moninkertaistaa kehitysmahdollisuuksia ja parantaa perusominaisuuksia. (A M & Sonpatki 2016: 2; Stefanov 2022; Thinkwink 2017.)

Reactilla luodaan yksisivuisia sovelluksia, joiden sisältöä hallitaan erillisten komponenttien kautta. Sivun vaihto tapahtuu näytettävää komponenttia vaihtamalla, eikä sivun verkko-osoite siis muutu. Reactin toiminta sekä suosio ohjelmoijien keskuudessa perustuu osittain nimenomaan näihin koodin yksittäisiin komponentteihin, joita hallitaan erikseen. Käyttäjät voivat työstää koodia pienemmissä osissa sen sijaan, että pitäisi jatkuvasti ottaa huomioon sivuston jokainen ominaisuus. Näin mahdollistetaan selkeä ja helposti muokattava koodi. Siinä missä muilla menetelmillä

yksinkertaisen muokkauksen tekeminen vaatii suuriakin toimenpiteitä, mahdollistaa React täsmällisten korjausten tekemisen ja koodin helpon uudelleenkäytettävyyden. (A M & Sonpatki 2016: 2; Thinkwink 2017.)

A M & Sonpatkin (2016: 148) mukaan Reactin käyttäjäystävällisyyteen vaikuttaa myös merkittävästi sen nopea reaktioaika, joka perustuu virtuaalisen DOMin käyttöön, ja kykyyn reagoida tehtyihin muutoksiin reaaliajassa. React luo tämän virtuaalisen DOMin, joka kuvaa varsinaista DOMia. Sovellusta ajettaessa muutokset mallinnetaan virtuaaliseen DOMiin, jota React sitten vertaa varsinaiseen malliin. Vertailun perusteella React suorittaa pienimmän mahdollisen määrän toimia, joilla päästään haluttuun lopputulokseen. React pystyy siis havaitsemaan koodissa tapahtuneet muutokset ja päivittämään ainoastaan elementit, joita nämä muutokset koskevat.

React hyödyntää JSX Syntaksia (JavaScript XML), joka mahdollistaa HTML-tyyppisen koodin kirjoittamisen JavaScriptin sisällä. JSX:n läheisyys HTML:n kanssa tekee Reactista hyvin helppokäyttöisen monelle, joille HTML on jo ennestään tuttua. Erilaiset elementit, tagit ja attribuutit ovat samalla tavalla käytettävissä molemmissa. JSX:n avulla on myös mahdollista upottaa JavaScript-lauseita hyödyntämällä aaltosulkeita. Näin sisältöä voidaan renderöidä dynaamisesti. Usein JSX liitetään vain Reactiin, mutta sitä voidaan käyttää myös muiden ohjelmistojen tai kirjastojen kanssa. (React 2024a.)

2.2.1 Tilanhallinta Reactilla

Tilanhallinnalla tarkoitetaan komponenttien tilojen seurantaa ja päivittämistä, kun sovelluksen tila muuttuu. Tilanhallinnan avulla voidaan säilyttää tietoa komponenttien muutoksista. Näihin muutoksiin voi lukeutua muun muassa käyttäjän syöte ja valinnat, tai sovelluksen sisällön tila kuten tietokannasta ladatut tiedot. Tilanhallinta on välttämätön osa sovelluksia, joissa on vuorovaikutteisia osia. Tilanhallinnan tarve sekä hallinnan työkalujen vaatimukset vaihtelevat sen mukaan, kuinka monimutkainen sovellus on. Esimerkiksi yksinkertaisissa staattisissa verkkosivuissa tilanhallinnalle ei ole välttämättä lainkaan tarvetta. (Banks & Porcello 2020: 97–99; React 2024c.)

Banks & Porcellon (2020: 99) mukaan Reactissa tilanhallinta voidaan toteuttaa useammalla eri tavalla. Yksinkertaisessa sovelluksessa oiva vaihtoehto on Reactin oma sisäänrakennettu tilanhallintamekanismi. Se perustuu komponentin tilaan ja sen päivitykseen. Tila määritetään ja päivitetään setState ja useState -funktioilla. Näille annetaan haluttu tila, joka voidaan sitten linkittää komponentin toimintaan. Aina kun komponentin tilaa muutetaan setState-funktiota käyttäen, React uudelleenrenderöi komponentin ja uusi päivitetty tieto tulee näkyviin. (React 2024c.)

Koodissa 1 käytetään Reactin omaa tilanhallintajärjestelmää. Siinä luodaan funktio Laskuri, jossa nappia painamalla muuttujan arvoa voidaan lisätä aina yhdellä. Funktion määrittämisen jälkeen

nimetään tämä muuttuja (kpl) ja luodaan tila hyödyntämällä setState ja useState -funktioita. Tämän jälkeen luodaan lisäys-funktio, jossa setState-funktion avulla tila päivitetään vastaamaan uutta arvoa (kpl + 1). Ohjelmassa renderöidään eli palautetaan elementteinä kappale, joka näyttää laskurin nykyisen luvun, sekä painike, jota painamalla lisäys-funktiota voidaan kutsua.

```
import React, { useState } from 'react';

function Laskuri() {
  // Määrittää tilan ja tilan päivitysfunktion
  const [kpl, setKpl] = useState(0);

  // Funktio lisää tilan arvoa yhdellä
  const lisäys = () => {
    setKpl(kpl + 1);
  };

  // Renderöi komponentin ja näyttää tilan
  return (
    <div>
      <p>Määrä: {kpl}</p>
      <button onClick={lisäys}>Lisäys</button>
    </div>
  );
}

export default Laskuri;
```

Koodi 1. Tilanhallinta Reactilla

Monimutkaisemmissa sovelluksissa Reactin sisäänrakennettu tilanhallintajärjestelmä ei ole riittävä ja silloin tarvitaan todennäköisesti erillisiä työkaluja. Tällaisia tilanhallintajärjestelmiä ovat esimerkiksi Redux ja MobX. Redux on johdonmukainen ja laaja-alaisesti toimiva järjestelmä, joka tarjoaa keskitetyn ja vakaan tavan hallita sovelluksen tilaa. Reduxin avulla sovellukset toimivat monenlaisissa ympäristöissä ja niiden testaaminen on vaivatonta. (Redux 2024.) Toinen tilanhallinnan suosittu työkalu on MobX. MobX on yksinkertainen ja joustava tilanhallintajärjestelmä, jolla on vankka suorituskyky. (MobX s.a.)

2.2.2 Reititys Reactissa

Jotta React-sovelluksessa voidaan siirtyä komponentista toiseen ja vaihtaa sivua, tarvitaan reititystä. React ei sisällä valmiiksi sivujen reititystä, vaan tulee sitä varten asentaa erillinen ratkaisu. Suosittu reititykseen käytettävä kirjasto React-sovelluksia rakennettaessa on React Router. React Router tarjoaa tehokkaan ja joustavan tavan hallita reittejä sovelluksessa, joka sisältää useita eri sivuja eli komponentteja. Routerin avulla voidaan määrittää, mihin eri reitteihin navigoidaan, ja

mitkä komponentit renderöidään. Näiden lisäksi React Router tarjoaa muita ominaisuuksia, kuten dynaamiset reitit ja historiankäsittelyn, joiden ansiosta reittien käsittely on entistä joustavampaa. (W3Schools 2024b.)

Koodissa 2 näkyy App.js komponentti, jota kutsutaan, kun sovellus käynnistyy. Komponentin funktio palauttaa BrowserRouter-elementin, johon on käärityt tiedot sovelluksessa käytettävistä reiteistä. Esimerkki voisi kuulua sovellukseen, jossa on vähintään kolme muuta komponenttia: TabsMenu, Home ja Blog. Nämä komponentit tuodaan App.js komponenttiin, jotta niihin voidaan viitata reitityksessä. Näin mahdollistetaan eri komponenttien välillä liikkuminen sovellusta käytettäessä. (W3Schools 2024b.)

```
import React from 'react';
import { BrowserRouter, Routes, Route } from 'react-router-dom';
import TabsMenu from './pages/TabsMenu';
import Home from './pages/Home';
import Blog from './pages/Blog';

function App() {
    return (
        <BrowserRouter>
            <Routes>
                <Route path="/" element={<TabsMenu />}>
                    <Route index element={<Home />} />
                    <Route path="Blog" element={<Blog />} />
                </Route>
            </Routes>
        </BrowserRouter>
    );
}

export default App;
```

Koodi 2. Reititys Reactissa

Reitit esitetään Routes-elementin sisällä muutamalla eri tavalla. Ensin esimerkissä määritellään juuripolku, jonka sisään muut polut asetetaan. Juuripolku on näkyvissä jokaisella sivulla, ja koodissa 2 kyseessä on valikko. Valikon alle tulee seuraavana index-polku, joka määrittää oletussivun, johon sovellus laskeutuu, kun se käynnistetään. Tässä tapauksessa oletusaloitussivu on Home-komponentti. Viimeisenä määritellään normaali polku, path, jonka avulla määritellään reitti toiseen komponenttiin. Näitä path-reittejä voi olla useampia, riippuen siitä kuinka monen komponentin välillä sovelluksessa vaihdellaan. (W3Schools 2024b.)

Näihin luotuihin reitteihin viitataan myöhemmin koodissa käyttämällä Link-elementtiä. Link-elementissä nimetään reitti viittaamalla Route-elementissä annettuun Path-nimeen. (W3Schools 2024b.) Koodissa 3 linkit on luotu Tab-elementtien sisään, jotta navigointi onnistuu Tabs-valikon kautta kotisivulle ja blogiin.

```
<Tab label='Home' component={Link} to='/Home' />
<Tab label='Blog' component={Link} to='/Blog' />
```

Koodi 3. Tab-elementit

2.3 Palvelinratkaisut

Verkkosivujen palvelinpuolen koodissa hyödynnetään ulkoisia palveluntarjoajia ja niiden tarjoamia ohjelmarajapintoja. Sivustolle tarvitaan tietokanta taiteen esittämistä varten sekä lomake, jonka kautta käyttäjä voi lähettää sähköpostia ennalta määrättyyn osoitteeseen. Tietokanta rakennetaan Firebasen työkaluilla ja sähköpostilomake EmailJS-palvelun avulla.

Firebase on Googlen tarjoama sovelluskehityksen alusta, joka kattaa monipuolisen joukon pilvipohjaisia työkaluja. Niiden avulla sovelluskehittäjä voi rakentaa, parantaa ja kasvattaa verkko- ja mobiilisovelluksiaan. Firebase tarjoaa monipuolisen korin ratkaisuja eri tarpeisiin, oli ne sitten datan hallintaa ja tietojen varmennusta, laadun seurantaa ja ylläpitämistä, tai yleisön aktivointia ja kasvattamista. Kaikki palvelut ovat pilvessä ja niitä voidaan skaalata monipuolisesti tarpeiden mukaan. (Stevenson 2018.)

Stevenson (2018) mainitsee, että Firebasea voi käyttää monenlaisten sovellusten kehittämiseen ja parantamiseen. Firebasen tarjoamiin tuotteisiin lukeutuu muun muassa käyttäjän tunnistautuminen, reaaliaikainen tietokanta, pilvitallennustila sekä koneoppiminen. Näitä työkaluja voidaan hyödyntää monipuolisesti erilaisissa sovelluksissa sekä eri alustoilla. Firebase vastaa siis kattavasti monenlaisiin tarpeisiin ja se on myös hyvin helppokäyttöinen ratkaisu erilaisille kehittäjille ja projekteille.

Firebasen ajatuksena on tarjota valmiiksi käytettäviä työkaluja ohjelmiston backendin rakentamiseen. Sen avulla voidaan nopeuttaa ohjelman kirjoittamista ja siirtää backendin hallinta muualle. Firebasen tarjoamiin ja ohjelmaan integroituihin ominaisuuksiin pääsee käsiksi Firebase-konsolin kautta. (Stevenson 2018.)

EmailJS on palvelu, jonka avulla sovellukseen voi implementoida sähköpostilomakkeen. Tätä hyödyntäen sovelluksesta voi lähettää sähköposteja suoraan, ilman palvelinpuolen koodia. EmailJS-tilille käyttäjä voi yhdistää haluamansa sähköpostipalvelun ja luoda omia sähköpostimalleja. Palvelun käyttöönottoa varten tulee koodiin lisätä EmailJS SDK ja tarvittavat komponentit. EmailJS on

suosittu palvelu, joka tukee kattavasti monia sähköpostialustoja ja tarjoaa työkaluja roskapostia vastaan, sekä tietoturvan parantamiseksi. (EmailJS 2024.)

2.4 Material UI

Material UI (MUI) on avoimen lähdekoodin komponenttikirjasto, jota voidaan hyödyntää React-sovellusten kehittämisessä. Se sisältää kattavan valikoiman valmiiksi rakennettuja komponentteja, joiden avulla voidaan helposti luoda ketteriä ja responsiivisia käyttöliittymiä. Näihin komponentteihin sisältyy muun muassa ruudukoita, painikkeita ja kaavioita. MUI:n sisältö perustuu Googlen Material Design-ohjenuoriin. Material Design on suunnittelukieli, joka tarjoaa käyttäjälleen mukavan ja intuitiivisen käyttökokemuksen. Sen avulla voidaan varmistaa, että kirjasto tarjoaa vakaan pohjan modernille ja käyttäjäystävälliselle käyttöliittymälle. (MUI 2024a; GeeksforGeeks 2024.)

MUI on suunniteltu helppokäyttöiseksi ja käyttöönottoa varten tulee sovellukseen vain asentaa MUI-kirjasto, eikä muita toimenpiteitä vaadita. MUI:n dokumentaatio tarjoaa komponenteille valmiit koodit, jotka voi sellaisenaan lisätä omaan sovellukseen ja kirjaston asentamisen jälkeen komponentteja voi suoraan hyödyntää koodissa. Ne ovat sellaisenaan käyttövalmiita myös julkaistavassa sovelluksessa. Material UI-komponentit on suunniteltu tukemaan responsiivisuutta ja näin ollen ne toimivat saumattomasti eri kokoisilla laitteilla ja näytöillä. (MUI 2024a; GeeksforGeeks 2024.)

MUI voi tehostaa sovellusta kaikin puolin sen elinkaaren eri vaiheissa. MUI:n tarjoamat valmiit komponentit nopeuttavat ohjelman kirjoittamista ja vähentävät aikaa, jota kuluisi peruskomponenttien toistuvaan rakentamiseen. Tämän lisäksi MUI:n komponentit ovat hyvin kevyitä, mikä taas nopeuttaa niiden lataamista. Näin ollen sovelluksen latausajat lyhenevät ja suorituskyky paranee. Se taas tarjoaa käyttäjille nopean pääsyn sovelluksen sisältöön ja parantaa siten käyttäjäkokemusta. (GeeksforGeeks 2024.)

2.4.1 Material UI -Teema

Material UI tarjoaa myös mahdollisuuden käyttää Teemaa, jonka avulla voidaan vaivattomasti määritellä yhtenäinen tyyli koko sovellukselle. Teeman avulla voidaan muokata yleistä värimaailmaa, fontteja, tekstien värejä ja paljon muuta. Kun yleiseen Teemaan määritellään esimerkiksi brändin värit, pysyy sivuston ulkoasu kätevästi johdonmukaisena. (MUI 2024a.)

Teema voidaan luoda sovellukseen ThemeProvider-komponentin avulla. Käyttäjä voi kätevästi muokata sitä halutunlaiseksi, nimeämällä teeman ja antamalla sille haluamansa määrittelyt. (MUI 2024a.) Koodissa 4 näytetään Teeman alustus sovelluksessa.

```
const theme = createTheme({

  palette: {
    primary: { main: '#DAD2BC', contrastText: '#252323' },
    secondary: { main: '#70798C', contrastText: '#252323' },
    text: { primary: '#252323', secondary: '#70798C', contrastText: '#70798C' },
    background: { default: '#DAD2BC' }
  },

  typography: {
    fontFamily: "'Koulen', sans-serif",
    fontWeight: '400',
    fontStyle: 'regular',
  },
});
```

Koodi 4. Teeman luominen

2.4.2 Material UI Styled

MUI Styled on MUI:n Reactille tarjoama työkalu, joka mahdollistaa komponenttien tehokkaan ja yksinkertaisen tyylittelyn. Sen avulla kehittäjät voivat määrittää mukautettuja tyylejä komponenteille ja näin muokata sivuston ulkoasua tarkemmin. Eri komponenteille annettavat tyylit nimetään ja niille annetaan arvoja. Tämän lisäksi tyyleihin voidaan yhdistää MUI Teema, josta voidaan hakea haluttuja ominaisuuksia kustomoitaviin tyyleihin. (MUI 2024b.)

Koodissa 5 näytetään, miten Styled()-työkalulla luodaan ja käytetään tyylejä yhdistäen ne MUI Teemaan, joka on luotu erillisessä tiedostossa. Tiedostoon tuodaan styled ja useTheme MUI:n kirjastosta. Tämän jälkeen tyylit nimetään ja niiden sisältö muokataan halutun kaltaiseksi eri parametrejä hyödyntäen. Manuaalisen kirjoittamisen lisäksi tyylissä voidaan suoraan viitata kustomoituun teemaan. Myöhemmin koodissa elementit esitellään niille tyylissä annetulla nimellä. Näin määrätään mitä tyyliä missäkin elementissä noudatetaan. (MUI 2024b.)

```
import React from 'react';
import { Box, Typography, styled, useTheme } from '@mui/material';

function Styled() {
    const theme = useTheme();

    const MainBox = styled(Box)({
        display: "flex",
        flexDirection: "column",
        marginBottom: '100px',
    });
```

```
    const Text = styled(Typography) ({
        color: theme.palette.secondary.main,
    });

    return (
        <MainBox>
            <Text>Teksti</Text>
        </MainBox>
    );
}

export default Styled;
```

Koodi 5. Styled()-komponentit

2.5 UI-suunnittelu

UI-suunnittelu, eli käyttöliittymäsuunnittelu on keskeinen osa jokaisen verkkosivun ja sovelluksen suunnitteluprosessia. Käyttöliittymäsuunnittelun avulla pyritään luomaan selkeä ja helppokäyttöinen sivusto tai sovellus, joka tarjoaa käyttäjälleen miellyttävän ja halutun kokemuksen. Käyttöliittymäsuunnittelu liittyy myös vahvasti UX-suunnitteluun, eli käyttökokemussuunnitteluun. Nämä kaksi eroavat toisistaan siinä, että siinä missä UI-suunnittelussa luodaan sivustolle ulkokuori, keskitytään UX-suunnittelussa nimenomaan käyttäjän kokemukseen ja käyttäjäpolkuun verkkosivuja käytettäessä. (Stone, Jarrett, Woodroffe & Minocha 2005, 3; Haltu 2023.) Seuraavissa kappaleissa esitellään tarkemmin käyttöliittymäsuunnittelua.

Haltun (2023) mukaan käyttöliittymäsuunnittelussa keskitytään erityisesti siihen, miten sivusto tai sovellus toimii ja miltä se näyttää käyttäjän näkökulmasta. Siihen kuuluu käytännössä kaikki näkyvät ja interaktiiviset osat, jotka näyttäytyvät käyttäjälle. Näihin sisältyy muun muassa komponenttien rakenne, elementit ja niiden väliset suhteet, sekä sovelluksen ja käyttäjän välinen vuorovaikutus. Näiden lisäksi UI-suunnittelun osana on värien, fonttien ja muiden visuaalisten elementtien käyttö. Kaikilla näillä keinoilla pyritään luomaan houkutteleva ja helppokäyttöinen sivusto. (Stone ym. 2005, 6.)

Hyvä käyttöliittymäsuunnittelu kannustaa helppoon ja luonnolliseen vuorovaikutukseen käyttäjän ja sovelluksen välillä. Tämän periaatteen avulla käyttäjät voivat suorittaa tarvittavat tehtävät vaivattomasti ja intuitiivisesti. Käyttäjän tulisi voida jopa ikään kuin unohtaa, että on vuorovaikutuksessa tietokoneen kanssa. Kuten aiemmin mainittiin, kuuluu UI-suunnitteluun merkittävästi sivuston ulkoasu. Sen esteettisyys on kuitenkin hyvin subjektiivista ja sen mittaaminen taas hankalaa. UI-suunnittelun merkitys voidaankin kiteyttää yhteen sanaan: *käytettävyys*. (Stone ym. 2005, 6.)

Käytettävyys kiteyttää sanana sen, miten käyttäjät voivat saavuttaa tavoitteensa tehokkaasti vuorovaikutuksessa tuotteen tai palvelun kanssa. Oleellisena osana käytettävyyttä tehokkuuden rinnalla on myös käyttäjän tyytyväisyys kokemukseen. Hyvän käytettävyyden käyttöliittymän suunnitteleminen ei kuitenkaan ole aivan suoraviivaista. Käytettävyys on kontekstista riippuvaista ja siinä missä järjestelmä voi olla käyttökelpoinen yhdessä ympäristössä, voi se osoittautua käyttökelvottomaksi toisessa. (Stone 2005, 6–7.)

Näin ollen onkin merkityksellistä tunnistaa projektin tarpeet ja arvioida niitä monipuolisesti eri tilanteiden kannalta. Verkkosivuja suunnitellessa on tärkeää, että sivusto on laaja-alaisesti toimiva ja saavutettava. Tämän mahdollistaa kattava suunnittelu ja kokonaiskuvan arviointi. Seuraavassa kappaleessa esitellään verkkosivujen suunnitteluprosessi sekä teoriatietoa valittujen suunnittelumenetelmien taustalla.

3 Kehittämisprojektin suunnittelu

Unsettling Companyn verkkosivujen suunnittelu aloitettiin nykytilan selvittämisellä ja kattavalla tarpeiden kartoittamisella. Projektin alussa Unsettling Company toimi ainoastaan eri sosiaalisen median alustoilla. Yrityksellä on Instagram-, TikTok- ja LinkedIn-sivut, joiden kautta se voi tavoittaa seuraajia ja mahdollisia asiakkaita. Nämä sosiaalisen median sivut toimivat projektin alkuvaiheessa ikään kuin portfoliona yrityksen tuotoksille. Yrityksen jäsenillä on toiveena saada ammattimaisempi ja persoonallisempi alusta sisällön esittelyä ja mahdollista myyntiä varten. Tämän pohjalta Unsettling Companyn verkkosivuja alettiin suunnittelemaan eri menetelmiä hyödyntäen.

3.1 Tarpeiden kartoitus

Vastatakseen yrityksen tarpeisiin, tulee verkkosivulla olla erilaisia ominaisuuksia ja yrityksen jäsenet ovat ajan mittaan rajanneet tärkeimmät ominaisuudet, joita verkkosivuille kaivataan. Ensinnäkin verkkosivujen yleisilme tulee kehitellä vastaamaan yrityksen brändiä ja tulee sen tuoda esille haluttu ilme ja tunnelma. Yritys haluaa erottua joukosta omalla taiteellaan mutta samalla on hyvä, että verkkosivut tarjoavat kävijälle katsauksen yrityksen persoonaan. Sivusto on ikään kuin maailma, jossa pääsee uppoutumaan brändin mieleen.

Taidetta tuottavan yrityksen verkkosivujen keskiössä on toki sisällön esittely. On äärimmäisen tärkeää luoda houkutteleva tapa esitellä yrityksen tuottamaa taidetta. Näin kävijöiden on helppo tutustua tarjontaan ja löytää mahdollisesti itseään kiinnostavia teoksia. Taidetta esittelevän sivun tuleekin ilmentää yrityksen henkeä ja antaa tilaa itse taiteelle. On myös hyvä huomioida taidesivun informatiivinen puoli; mitä kustakin taideteoksesta halutaan tuoda ilmi ja kuinka rajata tiedot vain oleellisiin. Näillä keinoin voidaan saavuttaa mahdollisimman käyttäjäystävällinen ja selkeä sivu, joka houkuttelee katselemaan teoksia pidemmän aikaa.

Viimeisimpänä verkkosivujen tulee tarjota selkeä tapa ottaa yhteyttä yritykseen mahdollisten tiedusteluiden merkeissä. Yhteydenotto voidaan toteuttaa esimerkiksi sähköpostin, yhteydenottolomakkeen tai puhelinnumeron avulla. On tärkeää huolehtia siitä, että yhteystiedot ja mahdollinen lomake on helposti saavutettavissa ja yhteyden ottamiseen olisi mahdollisimman matala kynnys.

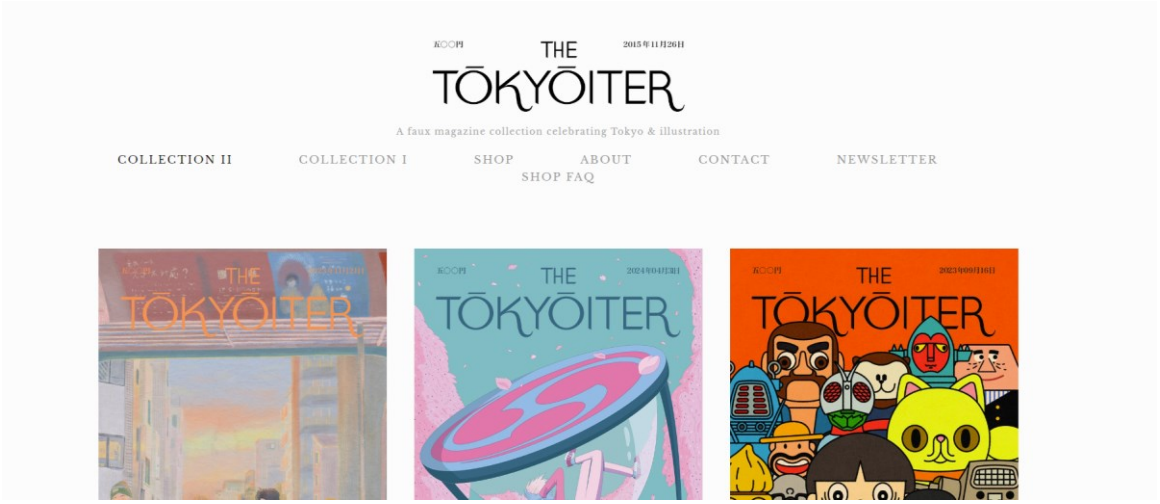
Tarpeiden kartoituksen yhteydessä todettiin, että verkkosivujen tulisi olla saavutettavat myös erilaisilla laitteilla ja eri kokoisilla näytöillä. Tämä tarkoittaa sitä, että verkkosivujen tulee olla responsiiviset. Tällöin niiden ulkoasu mukautuu näytön koon mukaan ja sisältö on selkeästi nähtävissä, on käyttäjä sitten tietokoneella tai mobiililaitteella. Kartoituksen jälkeen oli hieman selvempää, mihin suuntaa verkkosivujen kanssa edetään. Seuraavissa kappaleissa esitellään tarkemmin suunnitteluprosessia ja menetelmiä sen taustalla.

3.2 Benchmarking

Impiö (2022) mainitsee tekstissään, että kilpailijoilla voi olla hyvin merkittävä rooli yrityksen menestyksessä. Tämän vuoksi kilpailijoiden vertailu on tärkeässä osassa yrityksen toiminnan suunnittelussa. Tämä kilpailijoiden vertailu, tai benchmarking voi antaa hyvin arvokasta näkemystä siihen, mihin suuntaan yritystä tai sen palveluita tulisi viedä. Yksi tapa toteuttaa benchmarkingia voi olla esimerkiksi kilpailijoiden verkkosivujen sisällön ja ulkoasun arviointi.

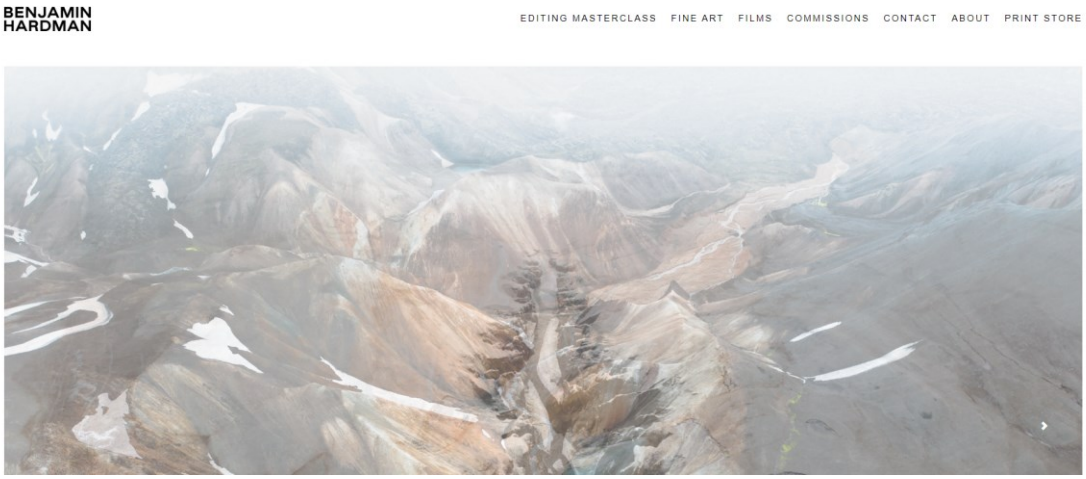
Opinnäytetyön verkkosivujen ulkoasun ideointi aloitettiin vertailemalla muita vastaavia sivustoja. Vertailun ajatuksena on kartoittaa alan nykytilannetta ja kerätä ajatuksia omaa sivustoa varten. Vertaillessa voi löytää hyviä ideoita, joita hyödyntää sivustoa rakentaessa. Tämän lisäksi voi kuitenkin myös kohdata ratkaisuja, jotka eivät ole omaan tarkoitukseen sopivia. Näistä ammentamalla saa selkeämmän kuvan omista ajatuksista ja mieltymyksistä. Projektia varten vertailuun valittiin kolme erilaista taideverkkosivustoa.

Ensimmäisenä tutkailtiin verkkosivua, joka toimii galleriana ja kauppana monen taiteilijan kuvituksille. Kuvassa 1 esitellään sivuston kotisvua. Sivuston jokainen sivu on rakenteen ja ulkoasun puolesta yksinkertainen. Sivujen pohjavärimaailma on vaalea ja hyvin neutraali. Sivujen ylälaidan keskiössä on persoonallisella fontilla kirjoitettu logo, joka korostuu selkeästi ensivilkaisulla. Pääosassa lähes jokaisella sivulla on kuitenkin yrityksen taide, joka tarjoaa räväkkää ja voimakasta sisältöä kautta sivuston. Verkkosivuilla on erikseen muutama sivu eri kokoelmien esittelylle. Näiden lisäksi siellä on erikseen kauppa, josta voi ostaa tiettyjä teoksia. Sivuilla on myös yleinen esittelysivu, UKK-osio ja yhteydenottolomake, jonka yhteydessä taiteilijat esitellään erikseen. (The Tokyoiter s.a.)



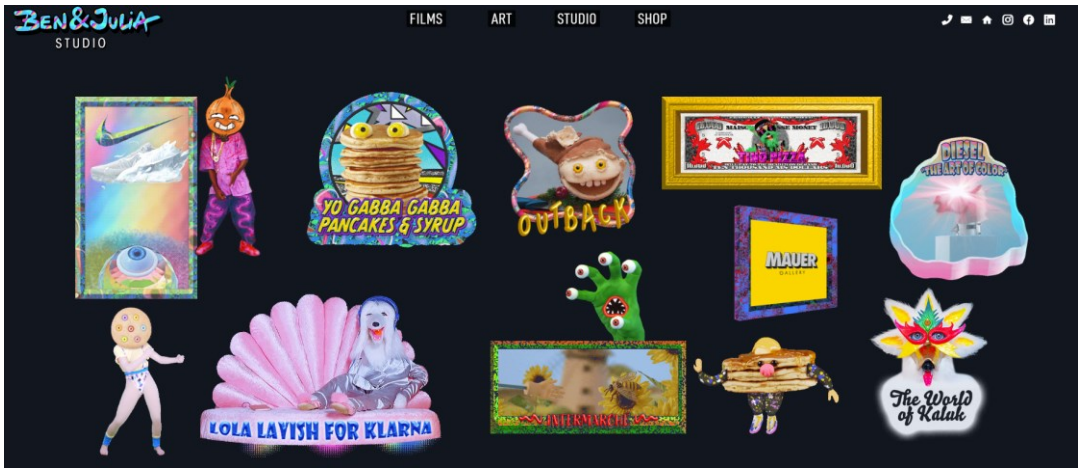
Kuva 1. The Tokyoiterin verkkosivujen kotisivu (The Tokyoiter s.a.)

Toinen vertailtava sivusto on niin ikään galleria ja kauppa taiteelle. Hän esittelee ja myy omia valokuvateoksiaan ja palveluita sivustolla, joka tarjoaa alustan kattavalle toiminnalle. Kuvassa 2 näkyy, kuinka sivusto itsessään on hyvin riisuttu ja brändin ilme tulee esiin nimenomaan teosten kautta. Sivuston yläpalkki ja linkit eri sivuille ovat hyvin yksinkertaiset ja eivät erityisesti korostu sisällön joukosta. Sivuilla voi tarkastella taiteilijan teoksia, lukea lisää hänen töistään, sekä ottaa yhteyttä lomakkeen kautta. Näiden lisäksi sivuilla on linkit ulkoisiin palveluihin, kuten printtikauppaan ja taiteilijan myymään kurssiin. (Benjamin Hardman s.a.)



Kuva 2. Benjamin Hardmanin verkkosivujen kotisivu (Benjamin Hardman s.a.)

Kolmantena vertailuun valittiin berliiniläisen luovan studion räväkät nettisivut. Kuten kuvasta 3 voidaan huomata, on sivusto heti ensivilkaisulla hyvin värikäs ja vilkas. Kuitenkin kuten aiemmissakin vertailtavissa sivustoissa, itse sivuston toiminnalliset elementit ovat hyvin yksinkertaisia ja jäävät taka-alalle. Sen sijaan yrityksen tuottama sisältö on nostettu aggressiivisesti esille. Sivujen sisältö on hyvin eläväistä ja suhteellisen sekavaa ja jopa interaktiivista. Sivuilla pääsee tarkastelemaan yrityksen erilaisia teoksia, tutustumaan taiteilijoihin ja siirtymään ulkoiseen verkkokauppaan, josta teoksia voi ostaa. (Ben and Julia s.a.)



Kuva 3. Ben and Julian verkkosivujen kotisivu (Ben and Julia s.a.)

Selkeimpänä yhteisenä tekijänä näitä kolmea sivua arvioitaessa vaikuttaa olevan verkkosivujen elementtien yksinkertaisuus. Verkkosivujen yleinen ulkoasu eroaa paljon, sillä tuotettu sisältö on hyvin eri tyylistä. Sivustot sisältävät kuitenkin hyvin samankaltaisia sivuja ja rakenteita. Jokainen taiteilija antaa myös omasta taiteestaan selkeän kuvan esittelemällä sitä suuresti heti etusivulla.

Yhteistä verkkosivuille on myös valikkojen yksinkertaisuus ja se, että ne ovat heti näkyvillä. Jokaisella sivulla on yläreunassa tekstimuotoinen logo ja sen läheisyydessä linkit eri sivuille. Varsinkin kahdella viimeisellä vertailtavalla yläpalkki on varsin vaatimaton ja vie vai vähän tilaa koko sivusta. Ensimmäinen vertailtava sivusto on keskittänyt logon ja valikon päällekkäin yläreunaan, joten ne kiinnittävät hieman enemmän huomiota.

Näiden vertailujen perusteella tultiin siihen lopputulokseen, että parhaiten omaa brändiä ja sisältöä saa esiin luomalla verrattain yksinkertaisen sivuston, joka antaa tilaa taiteelle. Sivuston tulee kuitenkin olla myös informatiivinen ja persoonallinen, joten esimerkiksi yläpalkille ja valikolle annetaan tilaa ja näkyvyyttä. Myös yrityksen logo saa olla selkeästi erottuvan kokoinen. Sivustolla tulisi myös olla esittelysivu, josta kävijä voi oppia lisää yrityksestä ja sen taustoista. Tärkeänä osiona varsinkin siinä vaiheessa, kun yrityksellä on myytäviä tuotteita, tulee olemaan väylä hoitaa kauppaa. Alkuvaiheessa se on yksinkertaisesti yhteydenottolomake, jonka kautta asiakkaat voivat tiedustella teoksista. Jatkossa kun myytävää sisältöä tulee lisää, voidaan harkita varsinaiseen verkkokauppaan siirtymistä.

3.3 Työpaja

Seuraava vaihe suunnitteluprojektia oli sivuston ulkoasun ja rakenteen tarkempi suunnittelu. Huolella tehdyn suunnitelman avulla voidaan suoraviivaistaa projektin etenemistä ja varmistaa halutun kaltainen lopputulos. Työpaja yhtenä suunnittelun menetelmänä on luova ja tehokas keino. Siinä

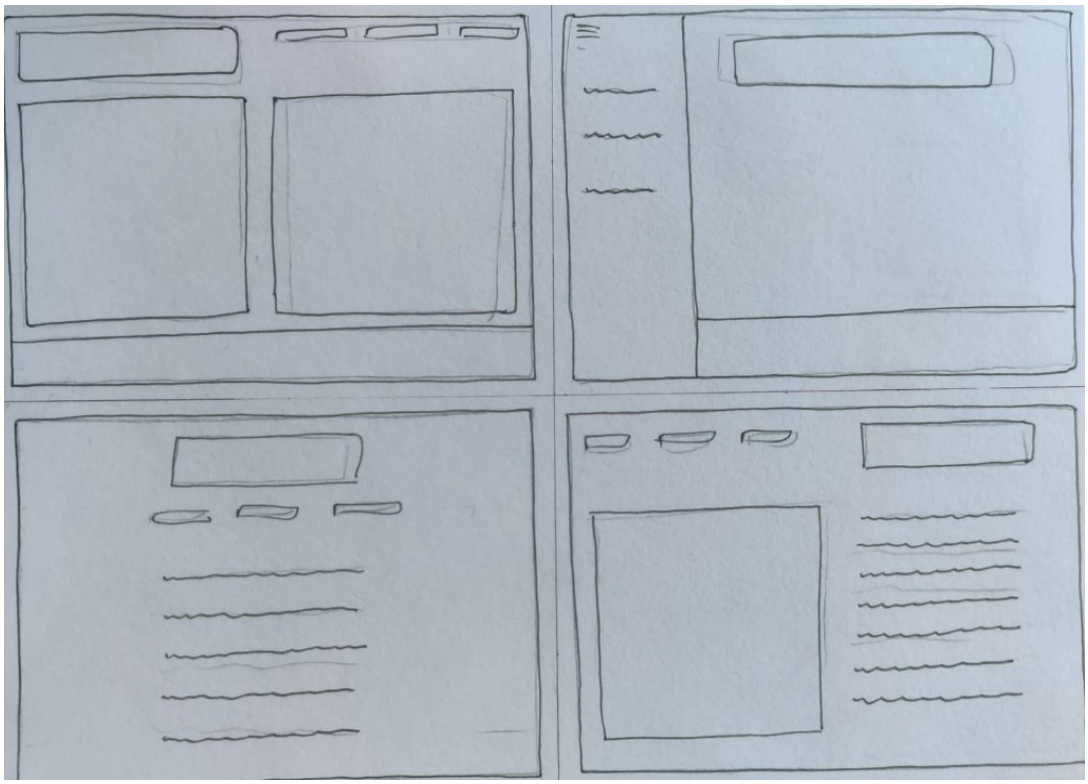
ryhmä osanottajia kokoontuu vaihtamaan ajatuksiaan ja ideoitaan keskenään. Näin ajatuksia pallottelemalla voidaan parantaa luovuutta ja kehittää toimivampia ratkaisuja haluttuun ongelmaan. Työpaja voidaan rakentaa monella eri tavalla riippuen projektin luonteesta ja tavoitteesta. Työpaja sisältää kuitenkin lähtökohtaisesti aina aivoriihen, ongelmanratkaisuharjoitteita, sekä ryhmäkeskustelua. (Bulykina 2023.)

Aivoriihen aikana on tarkoitus heitellä ilmoille erilaisia ajatuksia ja tutkailla miltä ne vaikuttavat ääneen lausuttuna. Ongelmanratkaisuharjoituksina voi olla monenlaisia tehtäviä, joissa on tarkoitus luoda nopeasti ratkaisua tutkailtavaan ongelmaan. Näitä voi olla esimerkiksi Crazy 8's, jossa jokainen osanottaja piirtää kahdeksan minuutin aikana kahdeksan luonnosta verkkosivun tai sovelluksen ulkoasusta. Toinen tehokas menetelmä on Prototyypin teko. Siinä lopputuotoksesta luodaan yksinkertainen prototyyppi, jonka avulla voidaan kokeilla erilaisia ideoita ja saada saman tien konkreettinen vastine ajatukselle. Jokaisen harjoituksen yhteydessä on tärkeää keskustella ajatuksista yhdessä ja purkaa ideat ulos. (Bulykina 2023.)

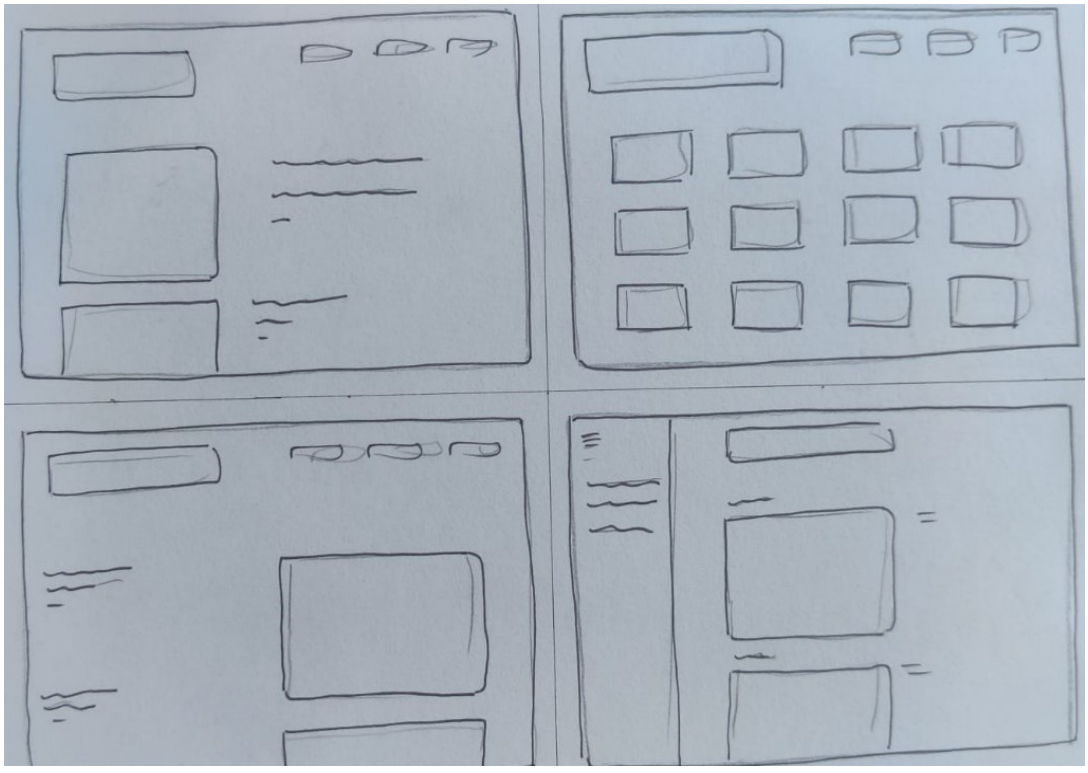
3.4 Työpaja verkkosivujen ulkoasun suunnittelulle

Työpaja järjestettiin 22.2.2024 ja siihen osallistuivat yrityksen kaksi perustajajäsentä. Työpajan aikataulu ja sisältö on nähtävissä liitteessä 1. Työpaja alkoi tilanteen kartoittamisella ja yleisellä keskustelulla. Keskustelun aikana määriteltiin käsiteltävä ongelma ja luotiin suuntaviivat sille, mitä verkkosivujen tulisi vähintään sisältää. Tämän keskustelun pohjana hyödynnettiin aiemmin määriteltyjä tarpeita ja Benchmarking-prosessin pohdintoja. Verkkosivujen sisältö on tarkoitus pitää suhteellisen yksinkertaisena ja sivuiksi valikoitui *Home, About, Art* ja *Contact*. Nämä sivut tarjoavat kaiken tarpeellisen sivuston vierailijalle. Kotisivu on kuin käyntikortti, joka antaa ensimakua yleisestä tunnelmasta ja yrityksen luonteesta. About-sivu tarjoaa lisätietoa yrityksestä ja sen taustoista. Art-sivu esittelee yrityksen tuottamaa taidetta. Contact-sivulla vierailijan on mahdollista ottaa yhteyttä yritykseen sähköpostitse.

Suuntaviivojen luomisen jälkeen siirryttiin luomaan pikaisia suunnitelmia verkkosivujen ulkoasulle. Suunnitelmissa mukailtiin Crazy 8's menetelmää ja työpajassa piirrettiin nopeasti muutamia erilaisia versioita sivuston rakenteesta. Sekä kotisivun että taidetta esittelevän sivun rakenteista piirrettiin neljä ideaa, jotka esitellään kuvissa 4 ja 5.



Kuva 4. Ideointi Home-sivun rakenteelle

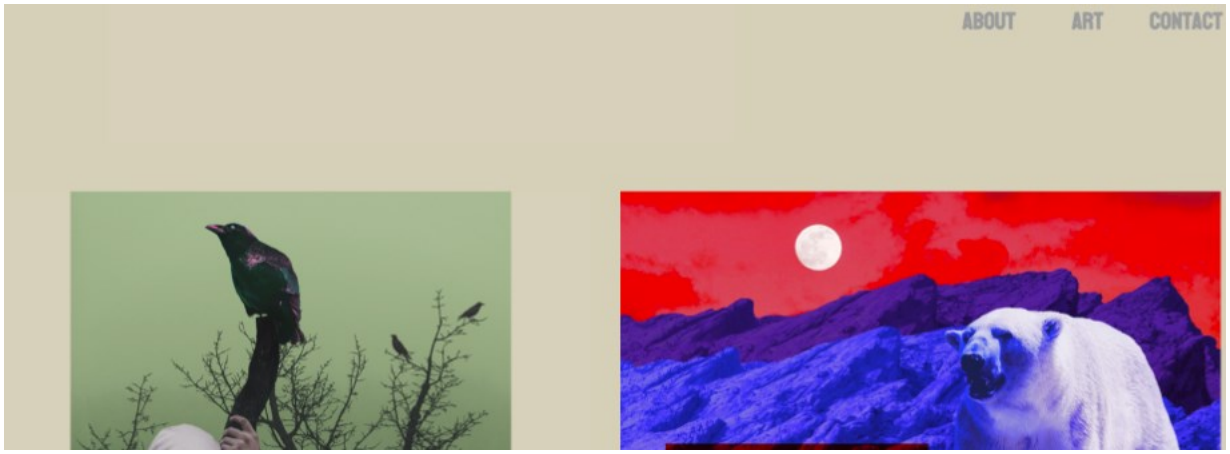


Kuva 5. Ideointi Art-sivun rakenteelle

Kun piirustukset olivat valmiina, niitä pysähdyttiin tarkastelemaan ja arvioimaan. Piirustuksista poimittiin keskusteluun seikkoja, jotka vaikuttivat toimivilta ja kommentoitiin osioita, jotka eivät viehättäneet. Erityisesti kotisivulle kiinnostavalta vaihtoehdolta vaikutti näkymä, jossa on kaksi suurta kuvaa, eikä lainkaan tekstiä (Kuva 4). Kotisivujen suunnitelmista poimittiin myös idea About-sivulle, johon houkuttelevimmalta vaikutti vaihtoehto, jossa on vaakasuunnassa puolet sivusta tekstiä ja puolet kuvaa. Art-sivulla oli selkeästi kaksi hyvin eri tyylistä rakennetta: ruudukko ja lista. Näistä kahdesta heti ensi alkuun listamainen näkymä vaikutti toimivammalta (Kuva 5).

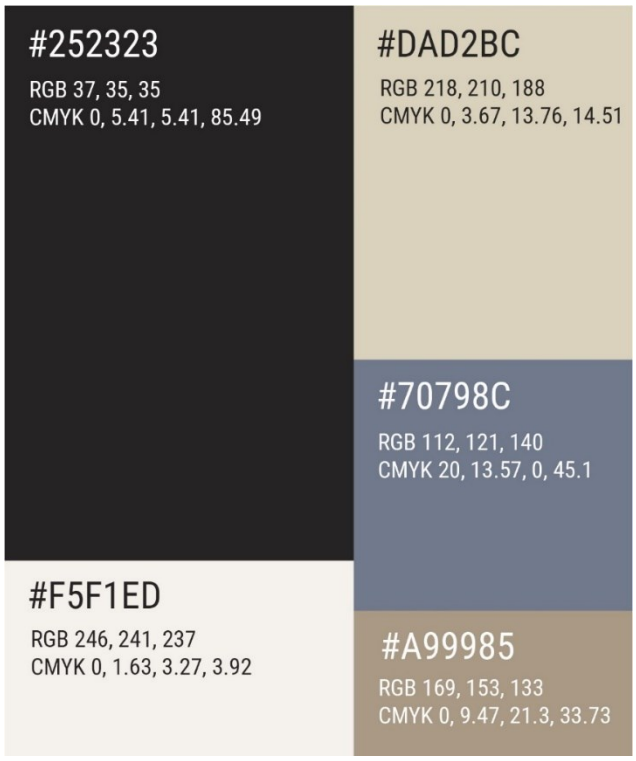
Keskustelun jälkeen siirryttiin työpajan seuraavaan vaiheeseen, jossa rakenteita kokeiltiin käytännössä. Tätä varten oli valmiina yksinkertainen React-sovellus ja tarvittavat komponentit, joiden avulla oli helppo tarkastella eri vaihtoehtoja käytännössä. Samalla päästiin myös ensimmäisen kerran kokeilemaan käytännössä brändivärejä ja fontteja. Eri versioiden kokeilun päätteeksi tultiin verkkosivujen ulkoasun rakenteesta selkeään lopputulemaan. Home-sivulle tulisi Logo, valikko ja kaksi suurta kuvaa, About-sivulle vierekkäin teksti ja suuri kuva, Art-sivulle listamaisesti kuvia ja niiden vasemmalle puolelle tekstiä ja Contact-sivulle yksinkertaisesti teksti ja sen viereen yhteydenottolomake. Jokaiselle sivulle tulee myös alatunniste, johon tulee yrityksen nimi, sähköpostiosoite, sekä linkit eri sosiaalisen median kanaviin.

Kaiken kaikkiaan piirustusten ja käytännön kokeilun jälkeen selkeäksi suosikiksi nousivat sivut, joissa on vahvasti esillä yrityksen tuottama sisältö. Vaikka brändin yleisilme ja värit ovat hyvin vähäpuheiset, tuo kuvat taiteesta sivuille hyvin dramaattisen olemuksen. Juuri tämä on se, mitä yritys haluaa verkkosivuillaan viestiä. Kuvissa 6 ja 7 näkyy vaihe testisovelluksesta ja käytännön kokeilusta muutaman sivun osalta. Kuvassa 8 esitellään yrityksen brändivärit.



Kuva 6. Prototyyppi Home-sivulta

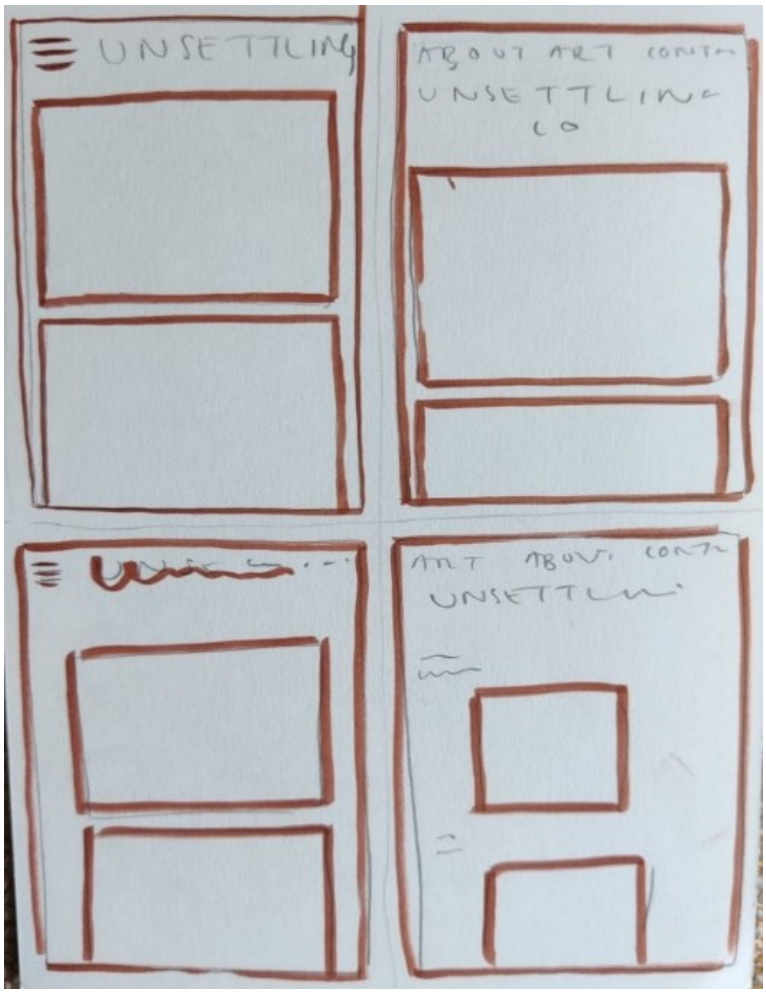
Kuva 7. Prototyyppi Art-sivulta



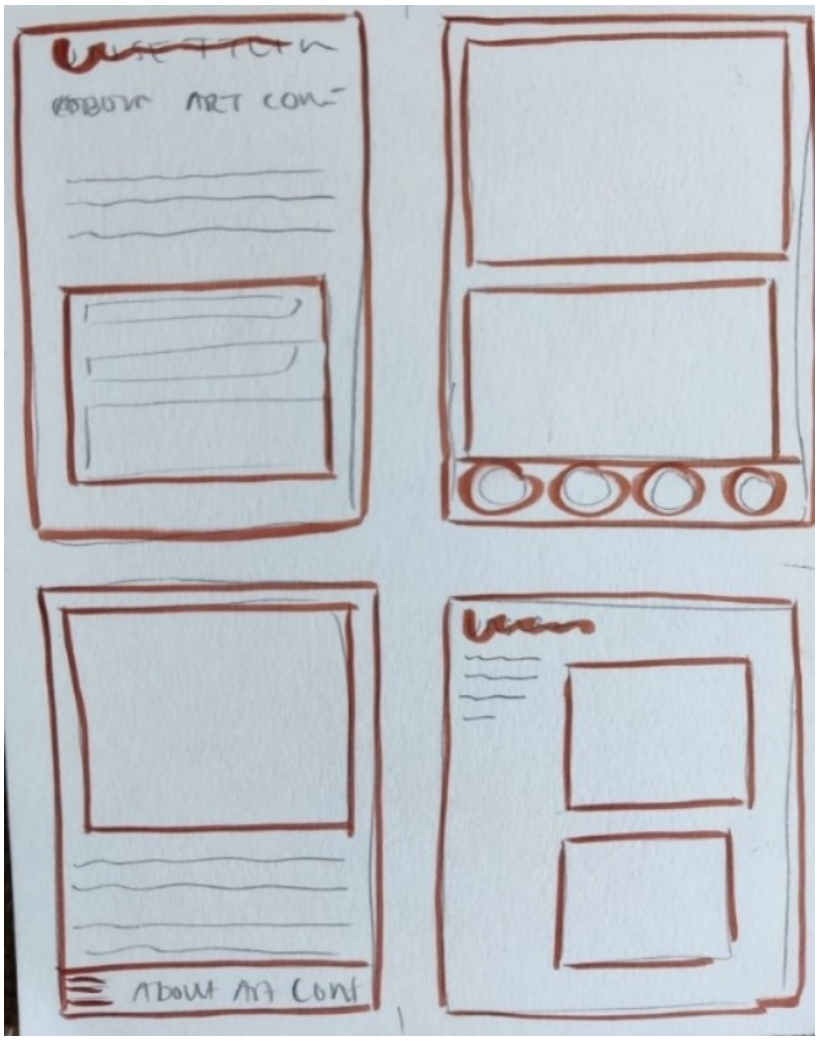
Kuva 8. Yrityksen Brändivärit (Pöyry 2024)

Lopuksi työpajassa tehtiin toinen tuokio pikaisia piirustuksia, kun ideoitiin mobiilinäkymiä. Seuraavissa kuvissa (Kuva 9; Kuva 10) näkyy mobiilinäkymien suunnitelmat, joiden avulla näkymät myöhemmin toteutetaan. Mobiilinäkymien osalta työpajassa ei tehty erillistä käytännön testausta sovelluksen kanssa vaan se tullaan toteuttamaan tilanteen mukaan varsinaisessa toteutusvaiheessa.

Mobiilinäkymän suunnittelussa suurimmat erot tietokoneversioon tulivat valikon tyylin ja sijainnin suhteen. Vaihtoehtoina ilmeni tietokonesivuston kaltainen yläreunan Tab-valikko, sekä sivusta erikseen avattava valikko. Näiden lisäksi ideoinnissa tuli esille yleisesti mobiilisovelluksissa esiintyvä alareunan Tab-valikko. Huomattavaa on myös, että oletettavasti logo ei mahdu Tab-valikkoon vierekkäin samaan tapaan kuin verkkosivuilla. Näin ollen suunnitelmissa näkyy vaihtoehtoja, joissa logo on joko Tabsien ylä- tai alapuolella.



Kuva 9. Ideointi mobiilinäkymää varten 1: Home, Home, Art, Art



Kuva 10. Ideointi mobiilinäkymää varten 2: Contact, Home, About, Art

Näiden suunnitelmien ja aikaisempien päätösten perusteella vaikuttaa kuitenkin selvältä, mihin suuntaan mobiilinäkymän kanssa mennään. Myös mobiiliin halutaan samankaltainen vahva ja dramaattinen olemus kuin tietokonenäkymässä ja se tiettävästi saadaan tuomalla esiin yrityksen teoksia. Näin ollen todennäköisimmin mobiilinäkymään tulee valituksi suuret kuvat, jotka vievät isomman osan näytöstä. Samalla on kuitenkin huomioitava, että tarvittava informaatio on edelleen selkeästi saatavilla ja sivusto ei ole epäselvä.

4 Verkkosivujen toteutus

Teoreettisen pohjan keräämisen ja huolellisen suunnittelun jälkeen siirryttiin varsinaiseen toteuttamisprosessiin. Seuraavissa osioissa esitellään syitä käytettyjen teknologien valitsemisen taustalla. Sen lisäksi keskitytään tarkemmin eri toteuttamisprosessin vaiheisiin kuvaillen myös haasteita, joita matkan varrella kohdattiin. Keskeisimmät termit ja työkalut, joihin seuraavissa kappaleissa keskitytään tarkemmin ovat React, MUI, Firebase ja EmailJS. Yhteistä näille kaikille valituille teknologioille on niiden merkitys yksinkertaisen ja helposti hallittavan koodin kirjoittamisessa. Jokaisella niistä on merkittävä rooli ohjelman luomisessa ja kaikki ne tukevat koodin kirjoittamista tarjoamalla valmiiksi rakennettuja työkaluja. Näiden työkalujen avulla rakennetaan toki ohjelman Front-end-puolta, mutta minimoidaan samalla Back-end-puolen ohjelmointia. Tällaisten työkalujen avulla verkkosivujen ohjelmointi on saavutettavaa ja niiden ylläpitäminen on myös omistajalle yksinkertaisempaa.

4.1 Työkalujen valinnat

React valittiin työssä käytettäväksi ohjelmakirjastoksi, sillä se tarjoaa joustavan ja kevyen tavan luoda monipuolisia sovelluksia. Reactin suosio oli myös päätöksen takana, sillä sen laaja käyttöaste tuo luotettavuutta ja varmistaa kattavan dokumentaation ja verkosta saatavan tuen saavutettavuuden. Näiden lisäksi React on myös opinnäytetyön tekijälle ennestään opinnoista tuttu työkalu, joten sen valinta verkkosivujen luomiseen tuntui luontevalta.

Material UI valittiin niin ikää sen vuoksi, että sen käyttöönotto ja käyttö ohjelman luomisessa on helppoa ja intuitiivista. Samaan tapaan kuin React, on Material UI suosittu ja tarjoaa se kattavan dokumentaation, jonka avulla sen käyttö ja soveltaminen on helppoa. Näin sovelluksen ulkoasua ja toiminnallisuuksia voi muokata lähes rajattomasti oman mielikuvituksen ja yrityksen tarpeiden mukaan.

Alusta asti oli selvää, että verkkosivuille tulisi esille yrityksen tuottamia taideteoksia listauksena, jossa näkyisi kuva teoksesta, sekä lisätietoa muun muassa sen tekoajankohdasta ja toteutustavasta. Vaikka yrityksellä on alkuvaiheessa vain muutamia teoksia, päätettiin niiden hallitsemiseen ja sivustolla näyttämiseen käyttää Firebase-palvelua, ja sen tarjoamaa reaaliaikaista tietokantaa. Firebase-tietokantaan ei kuitenkaan pysty suoraan tallentamaan kuvia, joten sitä varten tarvittiin vielä erillinen ratkaisu. Tätä varten työhön otettiin käyttöön Firebase-pilvitallennuspalvelu, johon kuvat teoksista tallennetaan. Jokainen teos saa oman linkin, jonka avulla kuva haetaan tietokantaan ja näin saadaan näkyviin myös verkkosivuilla.

Firebase valittiin työhön, sillä se tarjoaa monipuolisesti palveluita kaikenlaisiin tarpeisiin. Vaikka alussa yritys hyödyntää ainoastaan tietokantaa ja pilvitallennustilaa, voi tulevaisuudessa tarpeet

kuitenkin muuttua ja Firebasen kattavat palvelut saattavat tulla suurempaan merkitykseen. Tämän lisäksi tietokannan käyttöönotto ja hallitseminen on selkeää ja luotettavaa.

4.2 Projektin luominen Reactilla

Yrityksen kotisivut luotiin käyttäen React-ohjelmakirjastoa ja Visual Studio Code -tekstieditoria. Kattavan suunnitteluvaiheen jälkeen ohjelmaa alettiin rakentamaan komponentti kerrallaan ideointivaiheen piirustusten perusteella. Ensimmäinen askel React-verkkosivun luomisessa on toki itse React-projektin luominen. React on monella tapaa käyttäjäystävällinen ja näin ollen myös projektin alustus on äärimmäisen helppoa. React-projektin luominen aloitetaan siirtymällä komentotulkissa kansioon, johon projekti halutaan luoda. Tämän jälkeen komentotulkissa ajetaan seuraava komento:

```
npx create-react-app projektin_nimi
```

Komennon ajon päätteeksi sovellus on jo käyttökunnossa ja se voidaan käynnistää esimerkiksi komentotulkin tai editorin kautta. Projektin alustuksen jälkeen valittuun kansioon on luotu uusi projektikansio, joka sisältää tarvittavat alikansiot ja tiedostot sovelluksen luomiseen. Käytännössä yksinkertaisimmillaan ohjelman kirjoittajan tarvitsee vain luoda haluamansa komponentit ja varmistaa, että niitä voidaan kutsua sovelluksen sisällä. Sovelluksen käynnistäminen tapahtuu siirtymällä komentotulkissa projektin kansioon (projektin_nimi) ja antamalla käynnistyskomennon.

```
cd projektin_nimi
npm start
```

4.2.1 Komponenttien kirjoittaminen

Kuten aikaisemmin todettiin, voidaan Reactilla luoda yksisivuisia sovelluksia, jossa verkkosivulla siirrytään sisällöstä toiseen vaihtamalla näkyvissä olevaa komponenttia. Komponentit kirjoitetaan yleensä paremman hallittavuuden vuoksi erillisiin tiedostoihin ja näytettävien komponenttien välillä liikutaan reitityksen ja linkkien avulla. Suunnitteluvaiheessa yrityksen tarpeita kartoitettiin ja havaittiin, että tarvittavat sivut yrityksen verkkosivuille olisivat Home, About, Art ja Contact. Näin ollen projektiin tulisi luoda vähintään neljä komponenttia. Näiden lisäksi tarvitaan kuitenkin myös omat komponentit sivuston valikkoa (Menu) ja alaviitettä (Footer) varten, joten projektiin luotaisiin kuusi eri komponenttia sivuston perusrakennetta varten.

Ensimmäisenä projektiin luotiin neljä peruskomponenttia eri sivuja varten hyvin yksinkertaisessa muodossa. Näin luotiin selkeyttä projektin rakenteelle ja varmistettiin, että kaikki oleellinen tulisi

sisältymään projektiin. Peruskomponenttien luomisen yhteydessä rakennettiin sovelluksen App.js-tiedostoon reititys teoriaesimerkin kaltaisesti.

Kun tarvittavat komponentit eri sivuille ja reititys niiden välillä oli alustettu, voitiin siirtyä navigoinnin näkyviin osiin. Suunnitteluvaiheen perusteella verkkosivuille valittiin valikko, joka asettuu sivun yläreunaan ja jossa jokainen linkki on heti näkyvissä. Tätä varten rakennettiin Menu-komponentti hyödyntäen Material UI:n AppBar-, Tab- ja Tabs -elementtejä. Koodissa 6 näkyy yksinkertaistettu ja rajattu versio sovelluksen Menu-komponentista. Siinä valikon rakenne muodostuu AppBar-elementin sisään, johon asetetaan Tabs-elementti. Tabsin sisään taas laitetaan jokainen haluttu valikon osa omana Tab-elementtinä. Näihin Tab-elementteihin kirjataan haluttu viittaus reittiin, joka on luotu App.js-tiedostossa. Verkkosivun TabBar-elementissä on kuva logosta, joka toimii linkkinä kotisivulle, sekä kolme muuta linkkiä, jotka johtavat eri sivuille. Menu on komponentti, joka näkyy sivuston jokaisella sivulla.

```
return (
    <Box>
      <AppBar position='static' elevation={0}>
        <Tabs>
           <Tab component={Link} to='/Home' icon={<img src="../assets/ogol.svg"
m           alt="Logo" />
           <Tab label='About' component={Link} to='/About' />
           <Tab label='Art' component={Link} to='/Art' />
           <Tab label='Contact' component={Link} to='/Contact' />
        </Tabs>
      </AppBar>
      <Outlet />
    </Box>
  )
```

Koodi 6. Valikko AppBar-elementillä

Toinen koko sivuston kattava elementti on Footer eli alaviite, joka asettuu jokaisen sivun alareunaan ja sisältää hyvin perustasoista tietoa yrityksestä. Footerin kautta pääsee myös navigoimaan yrityksen eri sosiaalisen median sivustoille. Footer rakennettiin myös omaan komponenttiin, ja siihen hyödynnettiin Material UI:n Grid-elementtejä. Grid-elementtejä voi helposti asetella vierekkäin ja päällekkäin ja näin ollen Footerin sisällön rakentaminen oli yksinkertaista.

Itse Footer-komponentin luominen ja sivustolla näyttäminen oli yksinkertaista, mutta haasteita tuotti sen oikeaoppinen asettaminen sivun alalaitaan. Aluksi Footer asettui aina sivuston näkyvän osan alareunaan ja peitti täten osan sivusta. Ongelman ratkomisessa seuraava vaihe oli, kun alaviite oli sivun alareunassa, mutta se ikään kuin ilmestyi aina vasta kun sivu vieritettiin alas asti. Tavoitteena oli, että alaviite olisi aivan sivun pohjalla koko ajan, eikä käyttäjä näkisi välähdyksenä

alaviitteen ilmestyvän aina kun saavutaan sivun alareunaan. Lopulta tyylittely onnistuttiin rakentamaan toivotun kaltaiseksi niin, että Footer on sivun pohjalla ja näkyy heti kun käyttäjä rullaa sivun alas. Koodissa 7 näkyy tyylejä, joiden avulla alaviite saatiin haluttuun paikkaan. ContainerStyle-tyylissä display: ‘flex’ määrittelee, että elementin sisältö asettuu automaattisesti täyttämään oman alueensa. FlexDirection: ’column’ täsmentää edellistä ja ohjaa sisältöä asettumaan kolumniin eli päällekkäin. ContentStyle-tyyli pitää ikään kuin paikkaa ja auttaa työntämään itse alaviitteen sivun alareunaan.

```
const ContainerStyle = styled('div')({
    minHeight: '20vh',
    display: 'flex',
    flexDirection: 'column',
  });

  const ContentStyle = styled('div')({
    flex: '1',
  });
```

Koodi 7. Alaviitteen asettelu

Seuraavaksi eri sivujen komponentteja alettiin täydentämään ennalta määrätyllä sisällöllä. Komponentit Home ja About ovat rakenteeltaan lähes identtiset. Molemmat ovat yksinkertaisia sivuja, jotka koostuvat koodiin tallennetuista kuvista ja tekstistä. Art- ja Contact -komponentit sisältävät enemmän toiminnallisuuksia ja molemmissa hyödynnetään APIa ja vaihdetaan näin tietoja ulkoisten palveluiden kanssa. Näiden toteuttamisesta kerrotaan lisää seuraavissa kappaleissa.

4.2.2 Tietokanta

Verkkosivujen tietokannan rakentamisessa hyödynnettiin Firebasen reaaliaikaista tietokantaa. Ensimmäinen askel Firebase-tietokannan käyttöönotossa on itse projektin luominen Firebase-konsoliin. Konsolin kautta voidaan hallita kaikkia Firebasen työkaluja ja tässä projektissa tietokannan käsittely tapahtuu täysin konsolin eli Firebasen verkkosivujen kautta. Projektin luominen on hyvin suoraviivaista ja ohjattua ja luomisen jälkeen Firebase tarjoaa kaiken oleellisen koodin tietokannan ja ohjelmakoodin yhdistämiselle. Firebase-projektin alustamisen jälkeen käyttäjä saa ohjeet käyttöönotolle, sekä automaattisesti luotuja API-avaimia tietokannan käyttöä varten.

Seuraava askel tietokannan käyttöönotossa oli Firebasen asentaminen projektiin komentotulkin kautta. Tämän jälkeen aikaisemmin luotuun Art-komponenttiin alettiin lisäämään tietokannan kannalta oleellisia elementtejä. Jotta tietokantaan voidaan saada yhteys, tulee projektiin tuoda useampia työkaluja (Koodi 8). Niiden lisäksi koodiin tarvitaan Firebasen luomat API-avaimet. Tässä tapauksessa avaimet määriteltiin erillisessä config-tiedostossa (Koodi 9), josta tiedot haettiin

varsinaiseen Art-komponenttiin. Avaimia ei ole tarve käsitellä erikseen alustamisen jälkeen, joten ne voidaan nimetä omassa tiedostossa. Näin koodi pysyy selkeämpänä ja helpommin hallittavana.

```
import { initializeApp } from 'firebase/app';
import { getDatabase, ref, onValue } from 'firebase/database';
const app = initializeApp(config);
const database = getDatabase(app);
```

Koodi 8. Firebase-työkalujen tuonti

```
const config = {
    apiKey: apiKey,
    authDomain: authDomain,
    projectId: projectId,
    storageBucket: storageBucket,
    messagingSenderId: messagingSenderId,
    appId: appId
  };

export default config;
```

Koodi 9. Firebasen API-avainten alustus config-tiedostossa

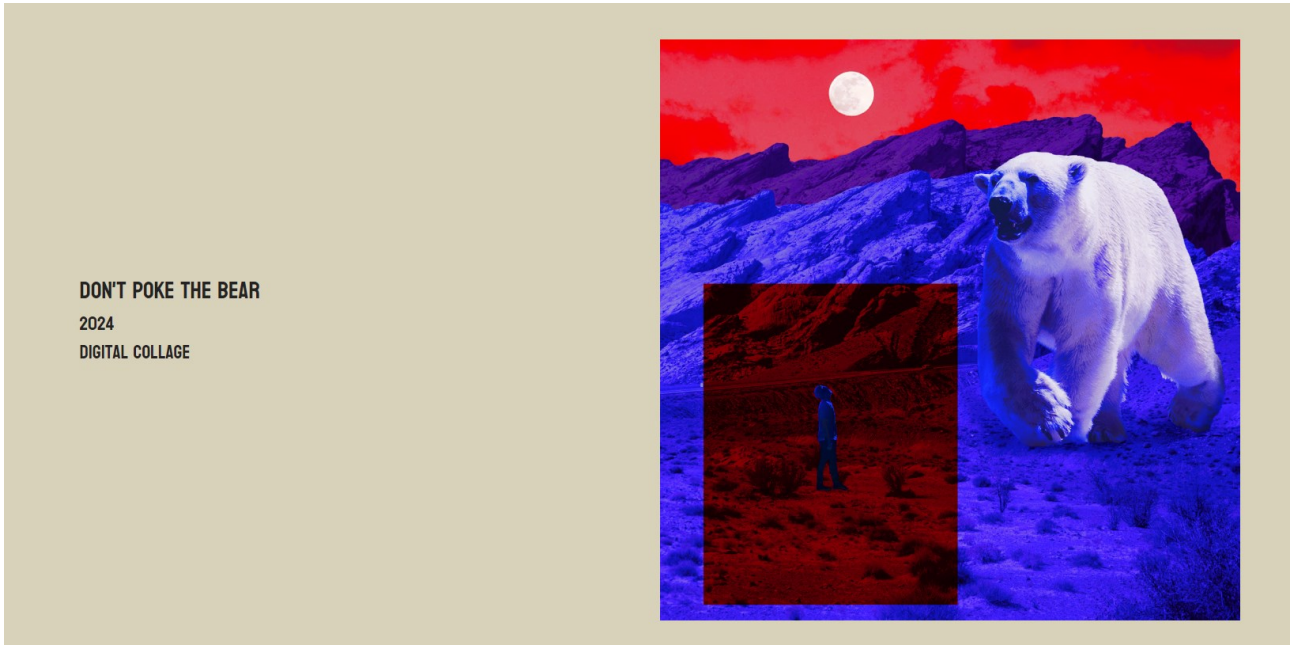
Taide-datan hakemisessa hyödynnetään Reactin omaa tilanhallintajärjestelmää ja ennen datan palauttamista järjestellään se tietueiden sisältämän päivämäärän mukaan, uusimmasta vanhimpaan. Tämän jälkeen tiedot näytetään Card-elementeissä, joihin haetaan jokaisesta tietueesta nimi, vuosi, tekotapa ja kuva (Koodi 10).

```
<ArtCard key={artwork.id} elevation={0}>
            <ArtCardContent>
              <InfoDiv>
                 <Typography variant="h6">{artwork.name}</Typography>
                 <Typography variant="body1">{artwork.date.slice(0, 4)}</Typography>
                 <Typography variant="body1">{artwork.medium}</Typography>
               </InfoDiv>
               <ImgDiv>
                  <Img src={artwork.imgURL} alt={artwork.name} />
               </ImgDiv>
             </ArtCardContent>
</ArtCard>
```

Koodi 10. Card-elementti taiteen näyttämistä varten

Firebasen reaaliaikaiseen tietokantaan ei saa lisättyä suoraan kuvaa. Kuvan lisäämistä varten luotiin projektiin Firebasen konsolissa myös pilvitallennustila, johon halutut kuvat voidaan lisätä. Tämä

jälkeen niihin voidaan viitata tietokannassa ja näin ne saadaan haettua myös verkkosivuille (Kuva 11).



Kuva 11. Card-elementti verkkosivulla

4.2.3 EmailJS

Contact-komponenttiin luotiin sähköpostilomake EmailJS-palvelun avulla ja Reactin omaa tilanhallintaa hyödyntäen. Samaan tapaan kuin Firebase, tarjoaa EmailJS backendin, rakenteen ja tarvittavat työkalut palvelun käyttöönottoa varten. Palvelun käyttöä varten tuleekin siis vain rakentaa Front-end-puoli.

Contact-komponenttiin lisättiin lomake, johon käyttäjä lisää oman nimensä, sähköpostiosoitteensa sekä viestin, jonka haluaa yritykselle lähettää. Lomakkeen alapuolella on painike, jonka painaminen kutsuu viestin lähettämisfunktiota. Napin painamisen jälkeen näytölle ilmestyy pieni ikkuna, joka ilmoittaa onnistuiko viestin lähetys. (Kuva 12.) Yritykselle saapuvassa sähköpostiviestissä näkyy kaikki käyttäjän antamat tiedot ja niiden perusteella yritys voi olla käyttäjään yhteydessä.

Kuva 12. Yhteydenottolomake verkkosivulla

4.3 Ympäristömuuttujat

Sekä Firebasen palvelut että EmailJS hyödyntävät API-avaimia, jotka ovat kuin salasanoja palveluiden saavuttamiseen. Näin ollen koodin kirjoittamisessa ja ohjelman julkaisussa tulee huomioida, ettei henkilökohtaisia tietoja julkaista kaikkien saataville. Tässä projektissa asia ratkaistiin asettamalla API-avaimet ympäristömuuttujiin. Sitä varten projektin juurikansioon luotiin .env-tiedosto, jossa API-avaimet nimettiin. React-projektissa ympäristömuuttujat nimetään esimerkin koodi 11 mukaisesti ja niihin viitataan myöhemmin koodin 12 tavalla.

```
REACT_APP_omaAvain = “omaAvain”
```

Koodi 11. API-avaimen nimeäminen ympäristömuuttujana

```
apiKey: process.env.REACT_APP_omaAvain
```

Koodi 12. Ympäristömuuttujaan viittaaminen

Näiden ympäristömuuttjien siirtyminen julkiseen versionhallintajärjestelmään voidaan estää projektikansion juuresta löytyvän .gitignore-tiedoston avulla. Tiedostoon kirjataan kaikki ne kansiot ja tiedostot, joiden ei haluta siirtyvän julkiseen säilöön. Näin arkaluontoista tietoa voidaan hallita projektin kehitysvaiheessa.

4.4 Responsiivisuus

Suunnitteluvaiheessa todettiin, että tärkeä osa verkkosivujen toimivuutta on niiden responsiivisuus. Verkkosivujen halutaan toimivan sekä tietokoneella että mobiililaitteella ja näin ollen koodin kirjoittamisessa tuli huomioida eri kokoiset näytöt. MUI:n elementit tukevat itsessään responsiivisuutta ja esimerkiksi MUI:n Grid-elementti mukautuu automaattisesti asettelultaan näytön koon mukaan.

Joitain elementtejä oli kuitenkin tarve muokata yksityiskohtaisemmin. Tämän saavuttamiseksi komponentteihin lisättiin MUI:n kautta saatava työkalu useMediaQuery. Se pystyy havaitsemaan käytettävän laitteen näytön koon ja tätä tietoa voidaan hyödyntää koodissa muuttujan avulla. Verkkosivun lähes jokaiseen komponenttiin luotiin muuttuja, joka tarkistaa näytön koon. Usein sen avulla muokattiin sivun asettelua vaakasuuntaisesta pystysuuntaiseen tai muuta vastaavaa. Suurin ero tietokonesivun ja mobiilisivun välillä on valikossa.

Kuten suunnitteluvaiheessa todettiin, ei yrityksen logo mahdu mobiilinäkymässä valikon viereen. Tätä varten tuli ulkoasulle esiin muutamia vaihtoehtoja, mutta lopulta päädyttiin ratkaisuun, jossa valikko on sivun ylälaidassa ja logo, joka toimii myös edelleen linkkinä etusivulle, on valikon alapuolella. Ohjelmapuolella tämä rakennettiin niin, että komponenttiin tuodaan Material UI:n kautta useMediaQuery, joka määrätään muuttujaan isMobile (Koodi 13).

```
import React, { useState } from 'react';
import { AppBar, Tabs, Tab, Box, styled, useTheme, useMediaQuery } from "@mui/mate-
rial";
import { Outlet, Link } from 'react-router-dom';

function Menu() {
  const theme = useTheme();
  const isMobile = useMediaQuery(theme.breakpoints.down('sm'));
```

Koodi 13. Responsiivisuus 1

Kun komponentti renderöidään, kutsutaan Tabs-elementin sisällä isMobile-muuttujaa tarkastamaan näytön koko. Koodi !isMobile … tarkistaa, että onko näyttö mobiilinäytön kokoinen ja jos näyttö ei ole mobiilinäyttö, renderöidään myös !isMobilen sisällä oleva elementti eli Logo valikossa. Jos taas !isMobile toteaa näytön olevan mobiililaitteen, jätetään logo ja sen Tab renderöimättä. Tällöin AppBar näyttää vain kolme painiketta. (Koodi 14.)

```
return (
    <Box>
      <StyledAppBar position='static' elevation={0}>
        <Tabs>
          {!isMobile && (
            <Tab component={Link} to='/Home' icon={<img src="../assets/ogol.svg"
oo.            alt="logo" />} />
           )}
          <Tab label='About' component={Link}to='/About' />
          <Tab label='Art' component={Link} to='/Art' />
          <Tab label='Contact' component={Link} to='/Contact' />
        </Tabs>
      </StyledAppBar>
      <Outlet />
```

```
    </Box>
  )
```

Koodi 14. Responsiivisuus 2

Mobiilinäytöllä logo jää siis Tabs-elementeistä pois, mutta se renderöidään toisella tavalla. Jokaisessa komponentissa kutsutaan ensimmäisenä vastaavanlaista isMobile-muuttujaa, joka arvioi näytön koon ja palauttaa kuvan logosta, jos havaitsee näytön olevan mobiililaitteen kokoinen (Koodi 15). Kuvassa 13 näkyy, millä tavalla valikko ja logo näkyvät käyttäjälle mobiilinäkymässä. Kuva logosta toimii linkkinä kotisivulle samaan tapaan, kuin tietokonenäkymässä.

```
<LogoBox>
   {isMobile && (
      <LogoLink to="/Home">
         <img src="../assets/ogol.svg" alt="Logo" />
      </LogoLink>
   )}
</LogoBox>
```

Koodi 15. Responsiivisuus 3

Kuva 13. Valikko mobiilinäkymässä

Näiden lisäksi useMediaQueryä ja sille määrättyä muuttujaa hyödynnettiin monissa eri tyyleissä lähes jokaisessa komponentissa. Koodissa 16 näkyy, kuinka tyylissä kutsutaan muuttujaa, joka taas tarkastaa näytön koon. ArtCard-tyylissä kortin suhteellinen leveys määrittyy sen mukaan, onko näyttö suuri vai pieni. Myös ArtCardContent-tyylissä hyödynnetään muuttujaa ja määritellään kortin sisällön asettelu näytön koon mukaan. Tietokonenäkymässä eli leveämmällä näytöllä kortin sisältö on rivissä ja mobiilinäkymässä sisältö on päällekkäin, kuten kuvasta 14 käy ilmi.

```
const ArtCard = styled(Card)({
    width: isMobile ? '100%' : '80%',
    marginBottom: '16px',
    backgroundColor: theme.palette.background.default,
});

const ArtCardContent = styled(CardContent)({
    display: 'flex',
    flex: 1,
    flexDirection: isMobile ? 'column' : 'row',
});
```

Koodi 16. Responsiivisuus 4

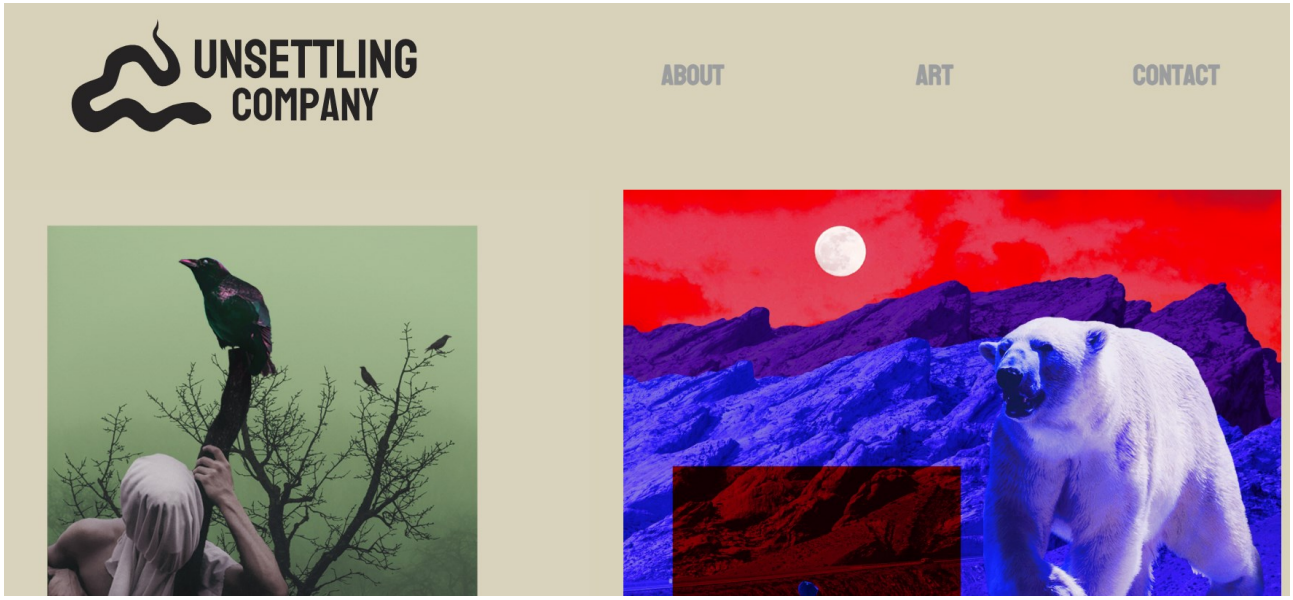
Kuva 14. Card-elementti mobiilinäkymässä

5 Yhteenveto

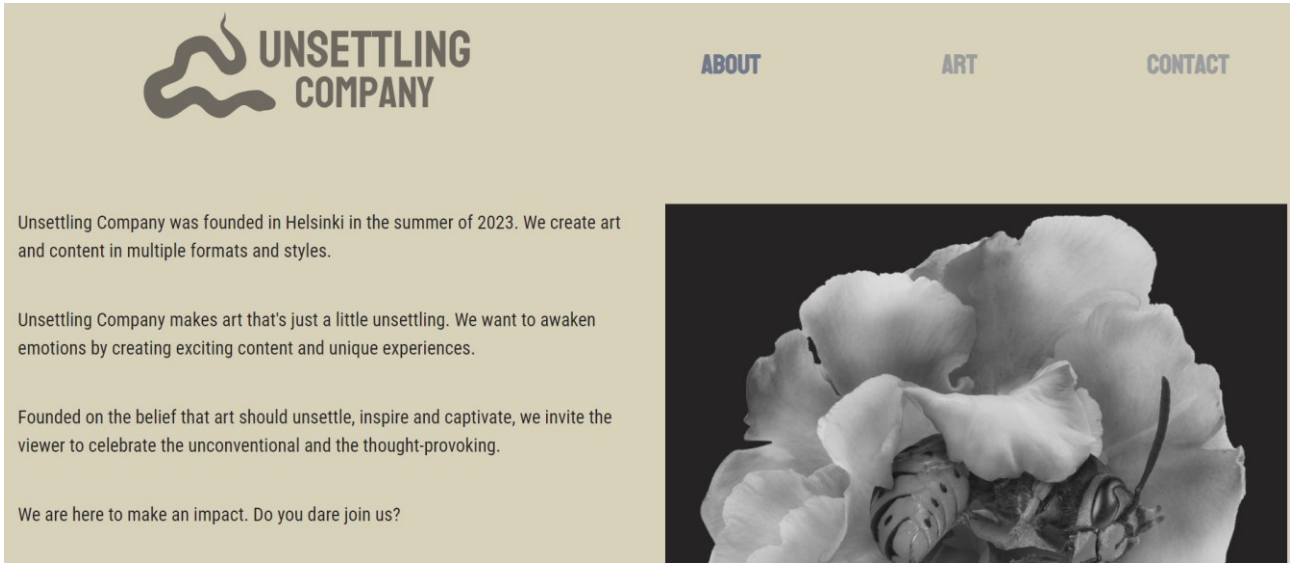
Opinnäytetyön tuloksena on syntynyt koodi, joka voidaan seuraavaksi julkaista yrityksen verkkosivuiksi. Kotisivujen toiminnallisuudet vastaavat tällä hetkellä hyvin yrityksen tarpeisiin ja lopputulos on myös lähellä suunnitteluvaiheen versiota. Sivut toimivat ennen julkaisua odotetusti erilaisilla näytöillä ja niiden eri ominaisuudet toimivat moitteettomasti. Näin ollen ohjelman tämän hetkinen tilanne vastaa hyvin odotettua ja julkaisuvaiheeseen on vaivatonta siirtyä. Koodi on nähtävissä julkisessa GitHub-repositiossa osoitteessa https://github.com/annimank/haikala.

Kuten todettiin, vastaa verkkosivujen ulkoasu ja sisältö pääsääntöisesti hyvin sitä versiota, joka suunnitteluvaiheessa rakennettiin. Koko opinnäytetyön suunnitteluvaiheessa verkkosivuille ajateltiin myös verkkokaupan rakentamista, mutta se rajattiin pois yrityksen tämänhetkisten tarpeiden ja projektin aikataulun vuoksi. Myös ulkoasullisesti lopputulokseen tuli kauttaaltaan pieniä muutoksia, enimmäkseen värien ja yksityiskohtaisemman asettelun osalta. Sivuston mobiilinäkymän ulkoasu varmentui myös vasta viimeistelyvaiheessa.

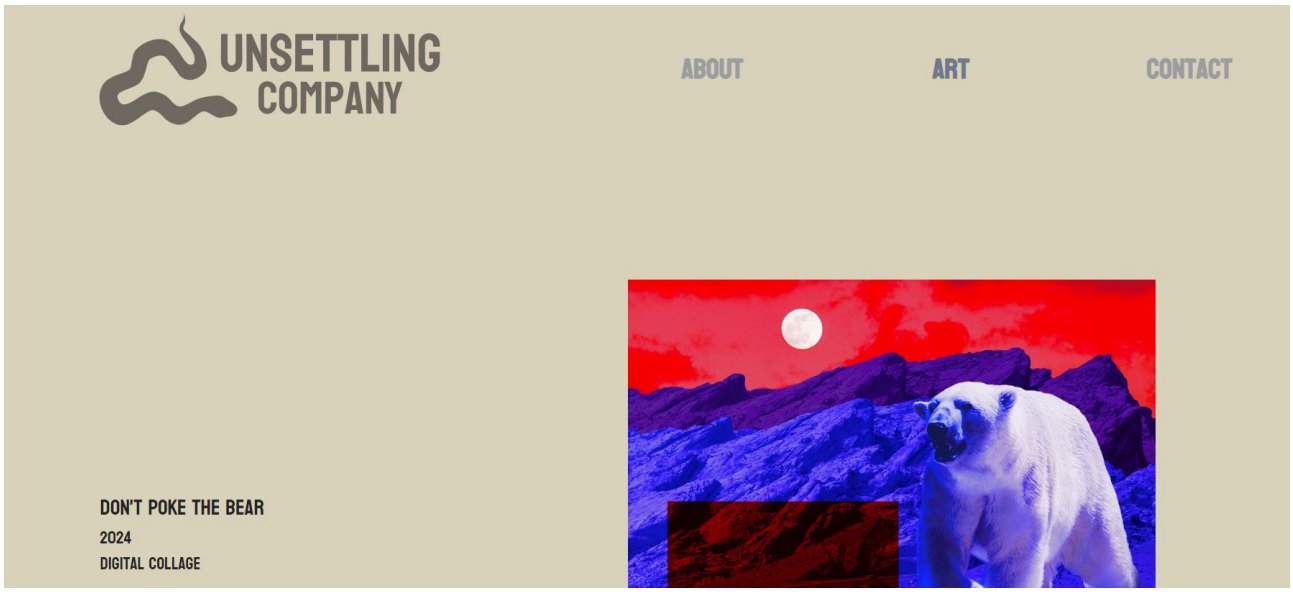
Verkkosivuilla on neljä eri sivua: Home, About, Art ja Contact ja ne sisältävät kaikki yritykselle tällä hetkellä oleelliset ominaisuudet. Yrityksen toiveena oli nimenomaan saada alusta, jota kautta omaa brändiä ja sisältöä voisi tuoda esiin. Verkkosivuilla näkyy kauttaaltaan yrityksen brändin yleisilme, joka tarjoaa neutraalin, mutta vahvan pohjan taiteen esittelyä varten. Sivuilla myös esitellään yrityksen perustietoja ja tarjotaan mahdollisuus yhteydenottoon, joko lomakkeella tai suoraan sähköpostilla. Kuvissa 15, 16, 17 ja 18 sivuston eri sivujen viimeisteltyä ulkoasua.



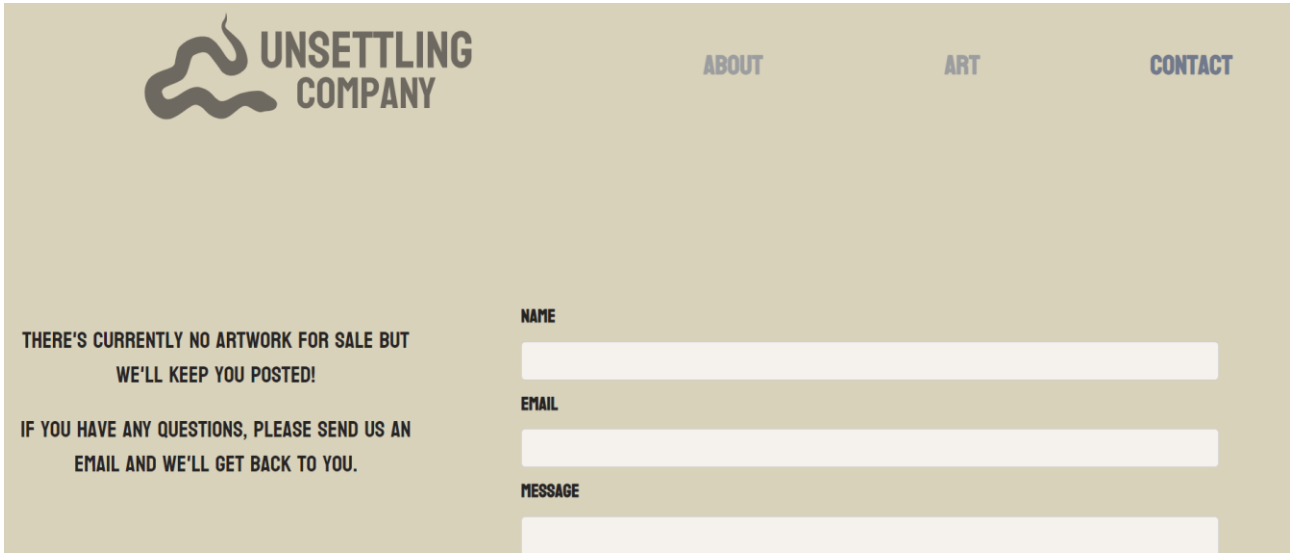
Kuva 15. Näkymä Home-sivulta



Kuva 16. Näkymä About-sivulta



Kuva 17. Näkymä Art-sivulta



Kuva 18. Näkymä Contact-sivulta

5.1 Kehityskohteet

Verkkosivut riittävät hyvin yrityksen tarpeisiin tällä hetkellä, mutta on odotettavaa, että jatkossa yritys kaipaa kattavampaa sivustoa ja näin ollen myös koodia tulee päivittää. Todennäköisesti suurimmat muutokset tulevat tapahtumaan ohjelman tilanhallinnassa, ja sivustolla taiteen myyntiin liittyvissä osioissa. Tällä hetkellä käytössä oleva Reactin oma tilanhallintajärjestelmä ei ole riittävä yhtään suuremmille ohjelmille, joten jatkossa tulee ottaa käyttöön joku ulkopuolinen työkalu. Sen lisäksi tulevaisuudessa tullaan todennäköisesti tarvitsemaan kunnollinen verkkokauppa, jota kautta taidetta ja muuta sisältöä voidaan myydä.

Edellä mainittujen lisäksi yksi kehityskohde sivustolle on responsiivisuuden parantaminen. Tällä hetkellä verkkosivut reagoivat kyllä mobiilikokoiseen näyttöön mutta laajempia tarkennuksia näyttöjen koolle ja sisällön asettelulle ei ole määritelty. Tämä tarkoittaa sitä, että maltillisen kokoisella tietokoneen näytöllä, sekä mobiililaitteella sivusto näyttäytyy suunnitellusti. Voi kuitenkin olla, että selkeästi suuremmilla näytöillä jotkin elementit ovat väärän kokoisia.

Opinnäytetyön ideointivaiheesta asti on ollut myös pohdinnassa verkkosivujen mahdollinen julkaisu. Yrityksellä on hallussaan oma verkkotunnus, johon sivusto on tarkoitus julkaista. Julkaisu jää kuitenkin pois opinnäytetyöstä yrityksen tämänhetkisten tarpeiden vuoksi.

Ennen verkkosivujen lopullista julkaisua sekä julkaisun jälkeen olisi myös hyvä suunnitella ja toteuttaa testausta, jonka avulla tarkastellaan kattavasti verkkosivujen toimivuutta erilaisissa tilanteissa. Testaus voidaan toteuttaa sekä koodin avulla että käyttäjätestauksena. Tämä vaihe olisi hyvä saada valmiiksi viimeistään ennen, kun sivustolle odotetaan suuria asiakasmääriä.

5.2 Haasteet ja onnistumiset

Työn haastavin osuus oli teoriapohjan kerääminen ja tietojen kirjoittaminen. Aiheena tämä on hyvin tekninen ja iso osa työtä on itse sivuston suunnittelu ja koodin kirjoittaminen. Laadukasta teoriatietoa aiheesta olikin haastava löytää ja sisällön jäsentely aiheutti myös päänvaivaa. Sen lisäksi sisällön rajaaminen tuntui paikoin vaikealta, sillä monessa osiossa aiheeseen voisi syventyä aina vaan enemmän ja enemmän. Työn edetessä kävi usein kuitenkin niin, että koodin kirjoittamisen lomassa tuli taas uusia ajatuksia itse teoriapohjaan ja sen jäsentelyyn.

Työkaluksi valikoitunut React osoittautui erinomaiseksi vaihtoehdoksi tämän tyyppiseen projektiin. Vaikka työkalu olikin opinnäytetyön tekijälle ennestään hieman tuttu, tarjosi työn tekeminen kuitenkin paljon uuden oppimista ja taitojen kartuttamista. Aika ajoin haastavammat vaiheet saattoivat tuntua hyvin hankalilta, mutta kattavan dokumentaation ja verkosta löytyvän tuen ansiosta haasteet saatiin kohdattua onnistuneesti. Myös muut työkalut, kuten Firebase ja EmailJS tarjosivat jonkin verran haasteita, sekä onnistumisia. Firebase oli tekijälle ennestään hieman tuttu mutta EmailJS oli aivan uusi tuttavuus, joka tuli vastaan tiedonkeruuvaiheessa.

Kaiken kaikkiaan prosessi on ollut äärimmäisen antoisa ja opettavainen. Vaikka tietojenkäsittelyn opinnoissa on tehty monia sovelluksia ja verkkosivuja erilaisilla työkaluilla ja alustoilla, on tämä opinnäytetyö ollut ehdottomasti kokonaisvaltaisin kaikista. Työn lomassa on saanut teknisten taitojen lisäksi oppia myös paljon lisää ajanhallinnasta ja itsensä johtamisesta.

Toimeksiantaja kokee työn tuotoksen erittäin hyödylliseksi ja verkkosivut vastaavat hyvin yrityksen toiveisiin. Sivusto seuraa ulkoasultaan brändille luotua ilmettä ja sisältää kaikki toivotut toiminnallisuudet. Tulevaisuudessa verkkosivujen odotetaan tukevan brändin näkyvyyttä ja sitä kautta myös yrityksen liiketoimintaa.

Lähteet

A M, V. & Sonpatki, P. 2016. ReactJS by Example - Building Modern Web Applications with React. Packt Publishing Ltd. Birmingham, UK. Luettavissa: http://bdfc.com.vn/wp-content/uploads/2018/03/ReactJS_by_Example.pdf . Luettu: 25.1.2024.

Banks, A & Porcello, E. 2020. Learning React. 2. painos. O'Reilly. Sebastopol, Kalifornia.

Ben and Julia s.a. Luettavissa: https://benandjulia.com/ . Luettu 14.3.2024.

Benjamin Hardman s.a. Luettavissa: https://benjaminhardman.com/ . Luettu 14.3.2024.

Bulykina, A. 2023. Ideation Workshop: Top 10 Practical Tools and Techniques. Right Information. Luettavissa: https://rightinformation.com/blog/ideation-workshop-top-10-practical-tools-and-techniques/ Luettu 5.3.2024.

Contentful 2023. What is an API? How APIs work, simply explained. Luettavissa: https://www.contentful.com/api/ . Luettu 1.4.2024.

EmailJS 2024. Send Email Directly From Your Code. Luettavissa: https://www.emailjs.com/ . Luettu 14.3.2024.

GeeksforGeeks 2024. React Material UI. Luettavissa: https://www.geeksforgeeks.org/react-material-ui/ . Luettu 23.3.2024.

Haltu 2023. Käyttöliittymäsuunnittelu - mitä se on ja onko siitä hyötyä? Luettavissa: https://www.haltu.fi/blogi/kayttoliittymasuunnittelu#kayttoliittymasuunnittelu-vai-ux-suunnittelu . Luettu 29.1.2024.

Impiö, A. 2022. Benchmarking eli kilpailijavertailu auttaa parantamaan pienemmänkin yrityksen tuloksellisuutta. Oulun Ammattikorkeakoulun blogi. Luettavissa: https://blogi.oamk.fi/2022/05/16/benchmarking-kilpailijavertailu-auttaa-parantamaan-yrityksen-tuloksellisuutta/ . Luettu 5.3.2024.

MDN 2024a. Introduction to the DOM. Luettavissa: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction . Luettu 7.3.2024.

MDN 2024b. What is CSS? Luettavissa: https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS . Luettu 13.3.2024.

Medlock, J. 2018. An Introduction to Environment Variables and How to Use Them. Medium. Luettavissa: https://medium.com/chingu/an-introduction-to-environment-variables-and-how-to-use-them-f602f66d15fa . Luettu 1.4.2024.

MobX s.a. MobX. Luettavissa: https://mobx.js.org/README.html . Luettu 14.3.2024.

MUI 2024a. Material UI – Overview. Luettavissa: https://mui.com/material-ui/getting-started/ . Luettu 13.3.2024.

MUI 2024b. styled(). Luettavissa: https://mui.com/system/styled/ . Luettu 6.4.2024.

Pöyry, N. 2024. Yrityksen Unsettling Company Brändiohjeistus.

React 2024a. Writing Markup with JSX. Luettavissa: https://react.dev/learn/writing-markup-with-jsx . Luettu 2.3.2024.

React 2024b. Rendering Elements. Luettavissa: https://legacy.reactjs.org/docs/rendering-elements.html . Luettu 13.3.2024.

React 2024c. State as a Snapshot. Luettavissa: https://react.dev/learn/state-as-a-snapshot . Luettu 14.3.2024.

Redux 2024. Redux Essentials, Part 1: Redux Overview and Concepts. Luettavissa: https://redux.js.org/tutorials/essentials/part-1-overview-concepts . Luettu 14.3.2024.

Simmons, L. 2023. Front-End vs. Back-End: What's the Difference? Computerscience.org. Luettavissa: https://www.computerscience.org/bootcamps/resources/frontend-vs-backend/ . Luettu 23.3.2024.

Stefanov, S. 2022. React: Up & Running. O'Reilly. Sebastopol, Kalifornia. Luettavissa: https://dl.ebooksworld.ir/books/React.Up.and.Running.2nd.Edition.Stoyan.Stefanov.OReilly.9781492051466.EBooksWorld.ir.pdf . Luettu: 25.1.2024.

Stevenson, D. 2018. What is Firebase? The complete story, abridged. Medium. Luettavissa: https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0 . Luettu 14.3.2024.

Stone, D., Jarret, C., Woodroffe, M. & Minocha, S. 2005. User Interface Design and Evaluation. The Open University. San Francisco. Luettavissa:

https://books.google.fi/books?hl=fi&lr=&id=VvSoyqPBPbMC&oi=fnd&pg=PR21&dq=ui+design&ots=d9JVO-kSPe&sig=REyroYlFjOJdX4TafVrBq0ujvxc&redir_esc=y#v=onepage&q=ui%20design&f=false . Luettu 29.1.2024.

The Tokyoiter s.a. Luettavissa: https://www.thetokyoiter.com/ . Luettu 14.3.2024.

Thinkwink 2017. Why ReactJS is gaining so much popularity these days. Luettavissa: https://medium.com/@thinkwik/why-reactjs-is-gaining-so-much-popularity-these-days-c3aa686ec0b3 . Luettu 25.1.2024.

W3Schools 2024a. React Components. Luettavissa: https://www.w3schools.com/react/react_components.asp . Luettu 7.3.2024.

W3Schools 2024b. React Router. Luettavissa: https://www.w3schools.com/react/react_router.asp . Luettu 23.3.2024.

Webflow, Inc. 2023. Importance of a website: 10 reasons why it matters. Webflow blogi. Luettavissa: https://webflow.com/blog/importance-of-website . Luettu 16.4.2024.

Liitteet

Liite 1. Wireframe Workshopin ohjelma

Wireframe Workshop

Helsingissä 22.2.2024 klo 17.30–19.00

Tarkoitus: Unsettling Companyn verkkosivujen ulkoasun ja rakenteen suunnittelu.

Paikalla kaksi perustajajäsentä.

Aikataulu	Teema
17.30–17.45	Keskustelu ja ongelman kuvaus
17.45–17.50	Ulkoasun nopea ideointi paperille Crazy 8's menetelmää mukaillen
17.50–18.00	Ideoiden tarkastelu ja kommentointi
18.00–18.40	Prototyypillä ideoiden testailu
18.40–19.00	Loppukeskustelu ja arviointi