Tampereen ammattikorkeakoulu

Sovelluskehysten Angular, React ja Vue.js vertailua

Patricia Paavilainen

OPINNÄYTETYÖ
Toukokuu 2019

Tieto- ja viestintätekniikan koulutus
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikan koulutus
Ohjelmistotekniikka

PAAVILAINEN, PATRICIA:
Sovelluskehysten Angular, React ja Vue.js vertailua

Opinnäytetyö 40 sivua
Toukokuu 2019

Työssä tarkastellaan JavaScriptiä ja sen johdannaisia Angular, React ja Vue.js ja miten ne eroavat toisistaan. Vertailun tarkoituksena on tuoda esille kielien välisiä eroja ja yhtäläisyyksiä ja antaa lukijalle hyvä mielikuva siitä, mitä sovelluskehykset pystyvät tekemään.

JavaScript on 1990-luvulla kehitetty ohjelmointikieli sen aikaisia selaimia varten. Se on vuosien varrella kehittynyt ja päässyt myös serverin puolella käytettäväksi kieleksi. Siitä löytyy johdannaisia kieliä kuten TypeScript, ajoympäristöjä kuten Node.js ja front- ja backend frameworkkeja.

Tunnetuimpia frontend-sovelluskehyksiä ovat Angular, React ja Vue.js. Angular on Googlen kehittämä framework, joka pohjautuu vanhaan AngularJS-frameworkkiin. React on Facebookin kehittämä framework, joka on käytössä Facebookin uutissyötteessä. Vue.js on kolmikon uusin tulokas, jonka suosio on noussut hurjalla vauhdilla vuoden 2016 Vue 2 -päivityksen jälkeen. Kaikki näistä ovat komponenttipohjaisia.

Suurimmat kehysten väliset erot ovat syntakseissa ja projektirakenteissa. Angularissa jokainen komponentti on pilkottu kolmeen eri tiedostoon: logiikka, käyttöliittymä ja tyylit. Reactissa komponentin logiikka ja käyttöliittymä ovat yhdessä tiedostossa, johon voi halutessaan lisätä tyylit. Vuessa luonnostaan kaikki ovat samassa tiedostossa erillisissä lohkoissa. Eroihin vaikuttaa myös se, että Angular on TypeScript-pohjainen kieli, kun React ja Vue ovat JavaScript-pohjaisia.

Asiasanat: angular, javascript, react, vue

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree in ICT Engineering
Software Engineering

PAAVILAINEN, PATRICIA:
Comparision of Angular, React and Vue.js Frameworks

Bachelor's thesis 40 pages
May 2019

This thesis explores the concept of JavaScript and its standardization, what JavaScript-based frameworks there are. Out of these frameworks Angular, React and Vue.js will get a closer look to get a basic idea of all three frameworks. The comparisons point is not to put the frameworks in order from best to worst or vice versa.

JavaScript is a scripting language that was developed throughout the 1990s. Originally developed by NetScape for their NetScape Navigator 2.0 browser, to bring life to the dull websites. Over the years JavaScript has evolved tremendously, from the website scripting langue to server-side code and to runtime environments. As well as to dozens of frameworks for both front- and backend use. And new languages like TypeScript that are supersets of JavaScript.

The most notable frontend frameworks are Angular, React and Vue.js. Angular is the successor of Googles AngularJS framework, it’s even developed by the same team. React is Facebooks own framework that it uses in Facebooks news-feed. Vue.js is the most recent framework, after its Vue 2 update in 2016 its popularity has skyrocketed and recently passed React in Github-stars.

The differences between the frameworks are most notable when it comes to syntax and project architecture. Angular splits all of its individual components into three pieces, logic, UI and styles, while Vue keeps all of the component related code in one file, although in different segments. React on the other hand has both logic and UI in the same file, while styling is in a separate file. Although styling can also be done in the same file. A contributing factor to this is that Angular is TypeScript-based while React and Vue are JavaScript-based.

Key words: angular, javascript, react, vue

SISÄLLYS

1 JOHDANTO 7
2 JAVASCRIPT 8
2.1 Ominaisuudet 8
2.2 Standardit 8
2.3 Johdannaiset kielet 9
2.4 TypeScript 10
2.5 Bootstrap 11
3 FRAMEWORKIT 13
3.1 React 13
3.1.1 Ominaisuudet 13
3.1.2 CLI 14
3.1.3 Syntaksi 15
3.1.4 DevTools 16
3.1.5 Router 18
3.1.6 Tilanhallinta 18
3.1.7 Bootstrap 18
3.2 Vue.js 18
3.2.1 Ominaisuudet 19
3.2.2 CLI 19
3.2.3 Syntaksi 22
3.2.4 DevTools 24
3.2.5 Router 25
3.2.6 Tilanhallinta 25
3.2.7 Bootstrap 26
3.3 Angular 26
3.3.1 Ominaisuudet 26
3.3.2 CLI 27
3.3.3 Syntaksi 29
3.3.4 DevTools 31
3.3.5 Router 31
3.3.6 Tilanhallinta 31
3.3.7 Bootstrap 32
4 VERTAILU 33
4.1 Ominaisuudet 33
4.2 CLI 34

4.3 Syntaksi 34
4.4 DevTools 35
4.5 Router 36
4.6 Tilanhallinta 36
4.7 Bootstrap 37
5 POHDINTA 38
LÄHTEET 40

LYHENTEET JA TERMIT

Framework	Ohjelmistokehys tai sovelluskehys, on ohjelmistotuote, joka muodostaa rungon sen ympärille rakennettavelle ohjelmalle. Kehykset sisältävät valmiita ohjelman osia.
Superset	Ohjelmointikieli, joka sisältää täysin toisen kielen ominaisuudet, TypeScript sisältää kaikki JavaScriptin ominaisuudet.
CLI	*Command Line Interface,* komentoriviltä käytettävä apuohjelma, joka sujuvoittaa kyseisien toimintojen suorittamista.
Tilanhallintajärjestelmä	*State management system*, on tapa hallita ohjelmassa kulkevaa dataa.
Store	Tilanhallintajärjestelmän sisäinen osa, joka pitää sisällään ohjelman tilan.
npm	*Node package manager*, pakettimanageri, on maailman suurin ohjelmistorekisteri, joka sisältää yli 800 000 koodipakettia.
npx	Npm-pakettimanagerin lisäosa, jonka tarkoituksena on tehdä CLI-työkalujen käytöstä helpompaa.
yarn	Vaihtoehtoinen pakettimanageri npm:lle, joka on npm-rekisterin kanssa yhteensopiva.
Router	Sivuston sisäisen reitityksen hallinta, toisin sanoen linkit sivulta toiselle.
SPA	*Single Page Application*, yksisivuinen aplikaatio, aplikaation sisällä liikkuminen ei aiheuta selaimen uudelleenpäivittymistä.

1 JOHDANTO

Opinnäytetyössä tutustutaan JavaScriptiin ja muutamaan siitä johdannaiseen frontend-sovelluskehykseen eli -frameworkkiin. Työhön valittiin kolme frameworkkia, joita vertaillaan keskenään eri osa-alueilta, valitut kielet ovat Angular, React ja Vue.js. Kyseiset kielet valittiin, koska ne ovat monella tavoin hyvin samankaltaisia, Vue.js-frameworkkia voi ajatella Reactin ja Angularin välimuotona.

Aluksi työssä esitellään JavaScriptiä yleisesti, sen standardeja, lyhyesti mitä suosituimpia framework johdannaisia (front- ja backend) on olemassa, sekä ominaisuuksia ja syntaxia. Lisäksi esitetään pieni katsaus TypeScriptiin ja mikä sen yhteys JavaScriptiin on.

Työn pääaiheena ovat Angular, React ja Vue.js, aluksi työssä kerrotaan kustakin kielestä erilaisia asioita. Frameworkeista tarkastellaan mm. ominaisuuksia, syntaxia, CLI-työkaluja, routereita ja tilanhallintajärjestelmiä. Tämän jälkeen kyseisiä aiheita vertaillaan keskenään.

Työn tarkoituksena ei ole sijoittaa kyseisiä frameworkkeja parhaimmasta huonoimpaan, vaan ottaa selvää, miten kielten eri ominaisuudet eroavat toisistaan.

2 JAVASCRIPT

JavaScript on NetScapen 1990-luvulla luoma ohjelmointikieli. Alkujaan kielen nimi oli LiveScript, joka myöhemmin muutettiin JavaScriptiksi, koska Java oli tunnettu kieli ja uuden nimen toivottiin auttavan markkinoinnissa. JavaScript tuli ensimmäisen kerran käyttöön NetScape Navigator 2.0 -selaimessa maaliskuussa 1996. (Peyrott, S. 2017) JavaScript alun perin kehitettiin tuomaan verkkosivut “eloon” (JavaScript.info. n.d).

2.1 Ominaisuudet

JavaScriptillä tehtyjä ohjelmia kutsutaan skripteiksi, jotka voi kirjoittaa suoraan verkkosivun HTML-rakenteeseen, josta ne toteutuvat automaattisesti sivun latautuessa. Skriptit toimitetaan ja toteutetaan tavallisena tekstinä, ne eivät tarvitse erillistä valmistelua tai kääntämistä toimiakseen. (JavaScript.info. n.d)

JavaScript on prototyyppipohjainen, dynaaminen kieli, joka tukee olio-ohjelmointia ja toiminnallisia ohjelmointityylejä. Se toimii webin käyttäjän puolella, jota voidaan hyödyntää, kun verkkosivun käyttäytyminen suunnitellaan/ohjelmoidaan erilaisten tapahtumien kohdalla. (MDN Web Docs. n.d.)

Nykyään JavaScriptiä voi toteuttaa muuallakin kuin selaimessa, huomattavimpana ovat serverit, lisäksi millä tahansa alustalla, jolla on JavaScript engine -niminen ohjelma. Selaimissa on sulautettu JavaScript engine, jota välillä kutsutaan JavaScript-virtuaalikoneeksi. Moottoreilla on erilaisia “koodinimiä” riippuen selaimesta, esimerkiksi Chromessa ja Operassa sitä kutsutaan V8-nimellä ja Firefoxissa SpiderMonkeyksi. (JavaScript.info. n.d)

2.2 Standardit

NetScape toimitti JavaScriptin ECMA Internationalille standardoitavaksi. Tästä syntyi uusi kielistandardi, joka tunnetaan nimellä ECMAScript. JavaScript on suosituin toteutus ECMAScript-standardista, muita toteutuksia ovat SpiderMonkey- ja V8 JavaScript -moottorit. (Morelli, B. 2017)

ECMAScript usein lyhennetään nimikkeeksi ES. ES:n perässä on useimmiten jokin numero (esim. ES5 tai ES2015), joka ilmaisee painoksen version. Ensimmäiset neljä versiota ilmestyivät vuosien 1997-99 välisenä aikana, jotka tunnettiin nimillä ES1, ES2, ES3 ja ES4. Seuraava versio, ES5, tuli vasta kymmenen vuotta myöhemmin. Seuraava julkaisu oli ES6 / ES2015, vastuussa oleva komitea päätti, että julkaisusta eteenpäin uudistukset tapahtuvat vuosittain. Tämän takia nimeämistyyli muuttui ES6-tyylistä ES2015-tyyliin. Tämän jälkeen tulleet versiot on nimetty julkaisuvuoden mukaan. Lisäksi on olemassa nimike ES.Next, joka aina viittaa seuraavana vuonna tulevaan standardiin. (Morelli, B. 2017)

ECMAScript-standardien muutoksien tarkoituksena on kehittää kieltä tuomalla siihen uusia ominaisuuksia sekä päivittämällä tai poistamalla vanhoja.

2.3 Johdannaiset kielet

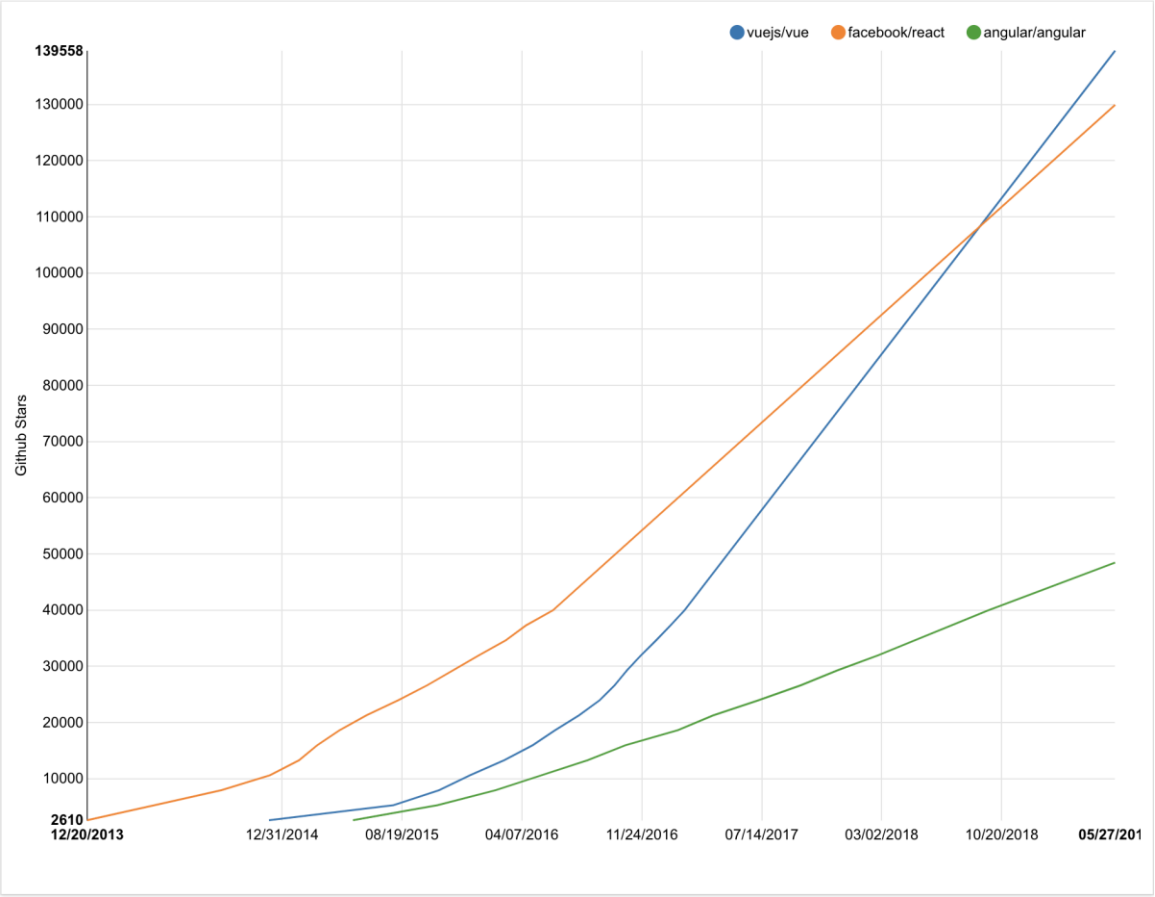
JavaScriptiin pohjautuvia ohjelmointikieliä, frameworkkeja, on nykyään monia kymmeniä. Frameworkkeja on sekä backend- ja frontend-kehitykseen. Lisäksi on JavaScript-ajoympäristö Node.js. JavaScript backend -frameworkkeja usein kutsutaan myös Node.js-frameworkeiksi, sillä ne vaativat Node:n toimiakseen.

Taulukossa 1 luetellaan vuoden 2018 suosituimmat frontend- ja backend-frameworkit. Taulukon tiedot perustuvat StateofJS:n tekemään kyselyyn, johon otti osaa yli 20 000 ohjelmistokehittäjää ympäri maailmaa. (StateofJS. 2018)

TAULUKKO 1 Suosituimmat JavaScript-frameworkit vuodelta 2018 (StateofJS, 2018)

Frontend	**Backend**
React	Express
Vue.js	Next.js
Angular	Koa
Preact	Meteor
Ember	Sails
Polymer	FeatherJS

Tunnetuimmat frontend-frameworkit ovat React, Vue.js ja Angular, joista React on kaikista käytetyin. Vue.js:n suosio on kasvanut räjähdysmäisesti vuoden 2016 julkaisun jälkeen (ensimmäinen versio julkaistiin 2014) ja on mennyt GitHubissa Reactin ohi tähtien määrässä (kuva 1). Kuvassa näkyy kyseisten frameworkkien Github-tähtien määrä. Kuva on generoitu Star History -ohjelmalla (Star History), jonka on kehittänyt t9t.io. Tunnetuimmat backend-frameworkit olivat Express, Meteor ja Next.js.



KUVA 1 Github-tähtien määrä kyseisissä repositorioissa, graafi on luotu Star History -ohjelmalla (Star History. n.d.)

2.4 TypeScript

TypeScript on JavaScriptin superset, eli se sisältää kaikki JavaScriptin toiminnot ja ominaisuudet, mutta on vielä niiden päälle lisännyt omia toiminnallisuuksiaan.

TypeScript-koodi käännetään takaisin JavaScript-koodiksi, kun ohjelmaa ajetaan.

TypeScript on Microsoftin julkaisema ja ylläpitämä kieli, jonka ensimmäinen julkaisu oli 2012.

Jotkin JavaScript-frameworkit tukevat TypeScriptiä ja mahdollistavat sen käytön JavaScriptin tilalla. Toiset frameworkit puolestaan toimivat vain TypeScriptillä.

2.5 Bootstrap

Bootstrap on ilmainen open source CSS framework, se sisältää CSS- ja JavaScript-pohjaisia templateja. Se on mobile-first-tyylinen framework, eli kaikki komponentit on suunniteltu siten, että ne skaalautuvat hyvin myös mobiililaitteilla.

Sen tarkoituksena on säästää koodaajien aikaa tuomalla valmiiksi tyyliteltyjä komponentteja. Bootstrappia voi hyödyntää ilman, että lataa sitä koneelle, tämä tapahtuu hyödyntämällä seuraavia koodin pätkiä (ohjelma 1) HTML-koodissa. *Link*-osa tulee laittaa HTML headeriin ja *Script*-osat bodyyn. (Bootstarp. n.d.)

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.
min.css" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper
.min.js" integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.mi
n.js" integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
```

OHJELMA 1 Bootstrap CSS- ja JavaScript-lisäykset koodiin (Bootstrap. n.d.)

Script-lohkot kattavat Bootstrap-komponentit, jotka vaativat JavaScriptiä toimiakseen, joten ne voi jättää pois koodista, jos niitä ei käytetä. (Bootstrap. n.d.)

3 FRAMEWORKIT

Seuraavaksi tutustutaan Angular-, React- ja Vue.js -frameworkkeihin. Jokaisesta kielestä tarkastellaan samat asiat, jotta niitä voidaan myöhemmin vertailla näiltä osin toisiinsa.

Aluksi kerrotaan hieman taustaa ja yleistä tietoa kielistä, jonka jälkeen tarkastellaan, eri kielten tärkeimpiä ominaisuuksia. Tämän jälkeen on hieman tietoa CLI- ja DevTools-työkaluista, näistä lähinnä kerrotaan, mitä on tarjolla ja mitä ominaisuuksia niillä on. Lopuksi eritetään, millaisia toteutustapoja ja lisäosia kielillä on router- ja tilanhallintatoiminnoille, sekä mitä mahdollisia Bootstrap-versioita kieli voi käyttää.

3.1 React

React on yksi tämän hetken suosituimmista JavaScript-frameworkeista, jonka on kehittänyt Facebook, ja se on osa Facebook Open Sourcea. Reactin ensimmäinen julkaisu oli 2013 toukokuussa. Koska React on osa Facebook Open Sourcea, on myös luontaista, että Facebook ylläpitää sitä. Tunnetuimpia sivuja, jotka käyttävät Reactia, ovat Facebook, Instagram, Netflix ja WhatsApp (Warcholinski, M. n.d). Reactin lisäksi Facebook on kehittänyt React Native frameworkin, joka on tarkoitettu natiivien mobiiliohjelmien tekemiseen.

3.1.1 Ominaisuudet

React on komponenttipohjainen kieli, joka mahdollistaa käyttöliittymän jakamisen pieniin osiin. Komponenttiensa takia React muistuttaa HTML-kieltä. (Lindley, C. 2018.) Tämä johtuu Reactin JSX-syntaksilaajennuksesta, joka mahdollistaa XML:n kaltaisen merkintätavan (React. n.d). Komponenttien käyttäminen parantaa koodin uudelleenkäytettävyyttä, sillä yhtä komponenttia voidaan käyttää useassa eri osassa sovellusta. Tämän takia komponenteista kannattaa tehdä toisistaan riippumattomia.

Yksi Reactin tärkeistä ominaisuuksista on virtuaalinen dokumenttiobjektimalli (VDOM), joka on JavaScript-pohjainen abstraktio web-selaimen sisällä

toimivasta dokumenttiobjektimallista (DOM). Erona DOMin ja VDOMin välillä on se, miten selaimen näkymä päivitetään, jos tehdään pieni muutos näkymään, DOM-mutaation seurauksena selain joutuu uudelleen piirtämään koko sivun. VDOMia hyödyntäessä puolestaan koko käyttöliittymä piirretään uudelleen virtuaalisena mallina, minkä jälkeen uutta ja vanhaa VDOMia verrataan keskenään ja DOMiin päivitetään vain ne asiat, jotka ovat muuttuneet. Ohjelmoijan näkökulmasta VDOM yksinkertaistaa komponenttien päivittämisen, sillä React päivittää DOMin käyttäjän puolesta. (Minnick, C. 2016.)

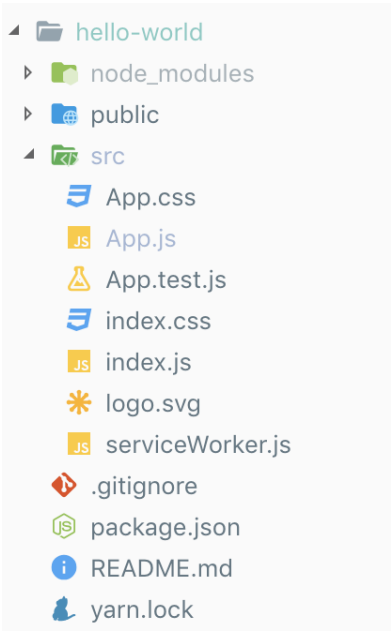
3.1.2 CLI

Reactin CLI-työkalu (Command Line Interface) on Create React App, jonka avulla on helppoa luoda uusi react-projekti. Työkalua ei tarvitse etukäteen asentaa, vaan projektin voi luoda suoraan alla olevilla komennoilla. (Create React App. n.d)

- npx create-react-app my-app
- npm init react-app my-app
- yarn create react-app my-app

Vaikka komennoissa on hieman eroja riippuen siitä, minkä pakettimanagerin avulla sen suorittaa, lopputulos on sama. Create React Appsilla on mahdollista myös pystyttää TypeScript-pohjainen React-sovellus, tätä varten komentoon on lisättävä –typescript-lippu (Create React App. n.d). Kuvassa 3 näkyy luodun projektin kansiorakenne, projekti on luotu kuvan 2 mukaisella komennolla.

KUVA 2 Create-react-app-komento



KUVA 3 React-projektin kansiorakenne

Projektissa on node_modules- ja public-kansioiden lisäksi src-kansio, jonka sisällä ovat kaikki muokattavat koodit.

3.1.3 Syntaksi

Seuraavassa koodiotteessa (ohjelma 2) näkyy luodun React-projektin App.js-tiedosto, joka pitää sisällään verkkosivulle tulostuvan sisällön. Koodin alussa ovat *import*-komennot, jotka lisäävät tarvittavat paketit ja tiedostot kuten React ja App.css, joka pitää sisällään ohjelman ulkoasun.

Ohjelmasta näkee React-komponentin rungon, joka on *constructor*, mahdolliset funktiot ja *render*-lohko, jonka sisällä on XML-tyylisesti verkkosivun sisältö. *Render*-lohkon sisällä ei suoraan voi hyödyntää JavaScript-koodia, vaan se pitää ympäröidä yksillä aaltosuluilla { }. Aaltosulkuja voi myös hyödyntää, kun verkkosivulle halutaan näyttää muuttujan arvo, muuttujan nimi laitetaan sulkuihin mahdollisien avainsanojen kera (esim. *this*. etuliite).

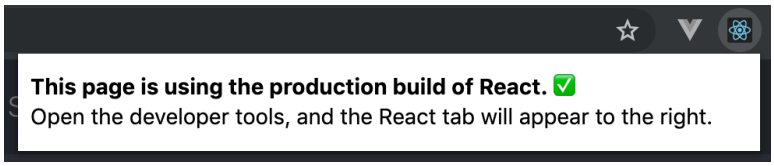
```
import React from 'react';
import logo from './logo.svg';
import './App.css';

class App extends React.Component {

  constructor(props){
    super(props);
    this.state = {
      hello: ""
    }
    this.sayHello = this.sayHello.bind(this)
  }


  sayHello(){
    this.setState({hello: "HELLO WORLD"})
  }

  render(){
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <p>
            {this.state.hello}
          </p>
          <button onClick={this.sayHello}>Press here</button>
        </header>
      </div>
    );
  }
}

export default App;
```

OHJELMA 2 React-projektin App.js-tiedosto

3.1.4 DevTools

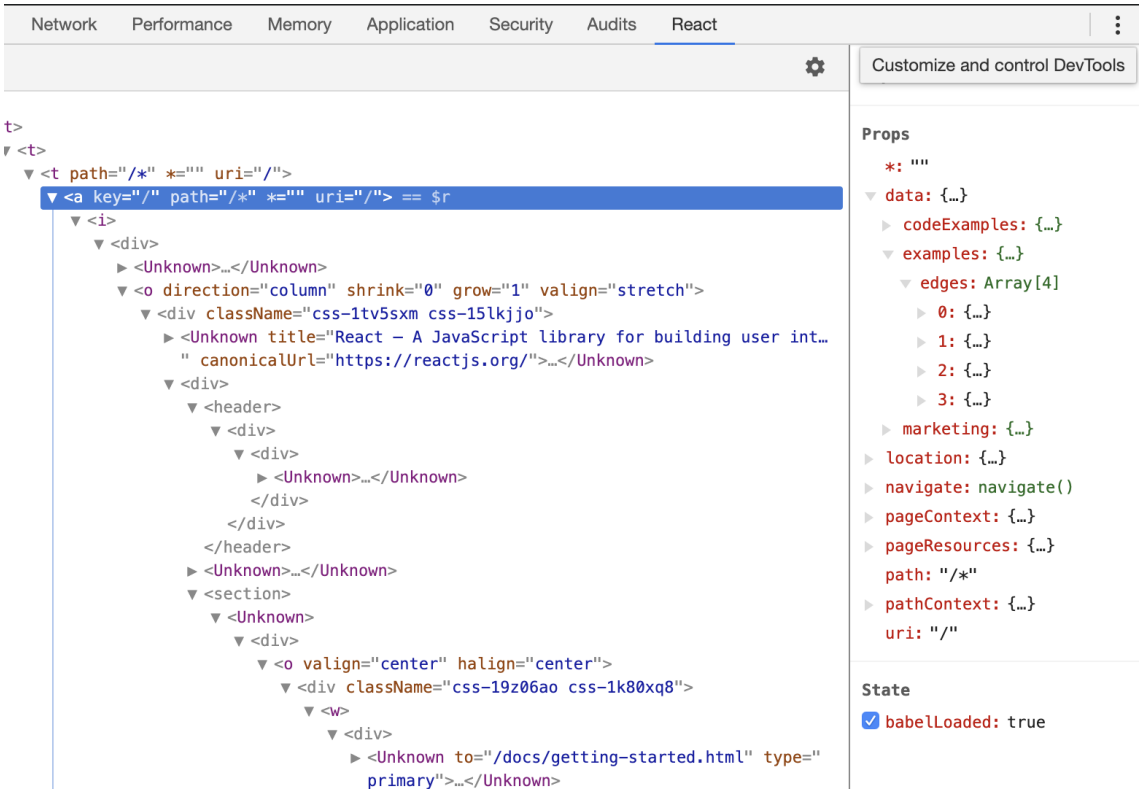
Facebook on kehittänyt React Developer Tools -työkalun, jonka voi ladata lisäosana Chromelle ja Firefoxille. Työkalun saa myös erillisenä ohjelmana, jos sitä haluaa käyttää esimerkiksi Safarin tai React Nativen kanssa. (React Devtools. n.d.)

Selaimeen ilmestyy hakupalkin vierelle React-ikoni, sitä painettaessa aukeaa pieni laatikko, joka kertoo Reactin olevan käytössä (kuva 4).



KUVA 4 React-ikoni selaimessa

Itse työkaluun pääsee käsiksi, kun menee selaimen Inspect-näkymään ja valitsee välilehden React. Välilehdellä näkyy sivun lähdekoodi, josta voi valita tietyn kohdan, jota haluaa tarkastella. Oikealla puolella näkyy, mitä dataa kyseinen komponentti saa (kuva 5).



KUVA 5 React Developer Tools -näkymä

Muita ominaisuuksia ovat, että oikeasta sivupalkista voi tallentaa globaaleja muuttujia sekä päivittää komponenteissa olevia tekstejä. Lisäksi on hakutoiminto, jolla voi etsiä tekstiä tai komponentteja nimeltä.

3.1.5 Router

Reactin suosituin router-paketti on react-router, joka tarjoaa routerin ydintoiminnot. Paketista on myös muita versioita, jotka on erikseen tarkoitettu React-verkkosovelluksille ja React Native -sovelluksille. Verkkosovelluksille tulee käyttää react-router-dom-pakettia ja puolestaan React Native -sovellusten kanssa react-router-native-pakettia. (Npm. n.d)

React-router mahdollistaa helpon tavan hallita sivuston sisäistä liikkumista linkkien välityksellä. Router-komponentti mahdollistaa sivun vaihtamisen ilman selaimen päivitystä.

3.1.6 Tilanhallinta

React-Redux-kirjasto on tilanhallintajärjestelmä, joka hyödyntää Redux-kirjastoa nimensä mukaan. React-Redux on suoraan yhteensopiva Reactin kanssa, muutoin se toimii samalla tavalla kuin Redux.

3.1.7 Bootstrap

Reactin kanssa voi käyttää perus-Bootsrappia, mutta on helpompaa hyödyntää Reactille suunniteltuja Bootstrap-versioita, kuten React Bootsrap tai reactstrap. Näistä kahdesta React Bootstrap on suositumpi, se pyrkii yhdenvertaisuuteen tavallisen Bootsrapin kanssa. Reactstrap puolestaan on hieman kompaktimpi ja sopii pienempiin projekteihin hyvin, kompaktiutensa takia siinä ei ole aivan kaikkia tavallisen Bootstrapin ominaisuuksia. (Create React App. n.d)

3.2 Vue.js

Vue.js on yksi suosituimmista frontend-frameworkeista, joita on tällä hetkellä saatavilla. Sen ensimmäinen julkaisu oli 2014. Vuen suosio alkoi kasvaa vuoden 2016 päivityksen jälkeen.

Vue on progressiivinen framework, jota käytetään rakennettaessa käyttöliittymiä. Toisin kuin muut massiiviset frameworkit, Vue on rakennettu alhaalta ylöspäin, jotta se olisi mahdollisimman helppo ottaa vähitellen käyttöön. Sen ydinkirjasto

keskittyy vain view-kerrokseen (layer), ja Vuen voi helposti lisätä jo olemassa olevaan projektiin. (Vue.js. n.d.)

3.2.1 Ominaisuudet

Vue.js tarjoaa HTML-pohjaisia *template*ja, jotka sitovat DOM:n Vue dataan. Vue tuottaa templateista VDOM-renderöintifunktioita. (Tutorialspoint, n.d.)

Yksi Vuen tärkeimmistä ominaisuuksista on lasketut ominaisuudet (computed properties). Se auttaa kuuntelemaan UI-elementeissä tapahtuvia muutoksia ja tekemään tarvittavia laskelmia, jotta komponentteja voidaan päivittää. (Tutorialspoint, n.d.)

Vuessa on sisäänrakennettuja direktiivejä: v-on, v-if, v-else, v-show, v-bind ja v-model. Kyseisillä direktiiveillä on mahdollista kontrolloida frontendiä eri tavoin, esimerkiksi ehdollisesti renderöidä elementtejä. V-bind-direktiivillä voi sitoa HTML-elementteihin arvoja ja muuttaa niiden tyylejä. V-on-direktiivillä voi käsitellä DOM-elementtien taphtumia. (Tutorialspoint, n.d.)

Komponenttien tyylin kannalta tärkeä ominaisuus on *style*-tunnisteeseen lisättävä *scoped*-lippu, joka ansiosta määritelty css-tyyli pätee vain kyseisiin komponentteihin, eikä sen lapsikomponentteihin. (Tutorialspoint, n.d.)

3.2.2 CLI

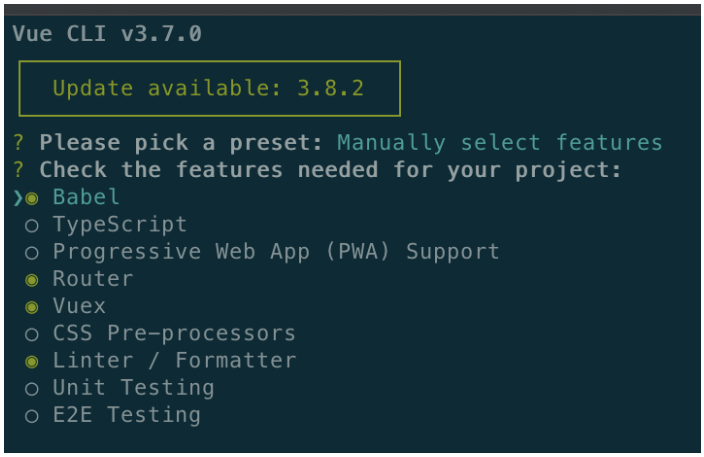
Vuen CLI-työkalu mahdollistaa työskentelyn komentoriviltä tai selaimesta Vue UI:n avulla. Molemmat työkalut mahdollistavat samat toiminnot, erona tietenkin on, että Vue UI on graafinen käyttöliittymä. UI käynnistetään komentoriviltä komennolla `vue ui`, joka avaa selaimessa loacalhostin portissa 8000.

Komentorivillä projekti luodaan komennolla *vue create hello-world* (kuva 6), jossa *v*ue on CLI-työkalu, *create* on komento, joka annetaan työkalulle, ja *hello-world* on luotavan projektin nimi.

```
/CLI-examples  vue create hello-world
```

KUVA 6 Vue-projektin luomiskomento

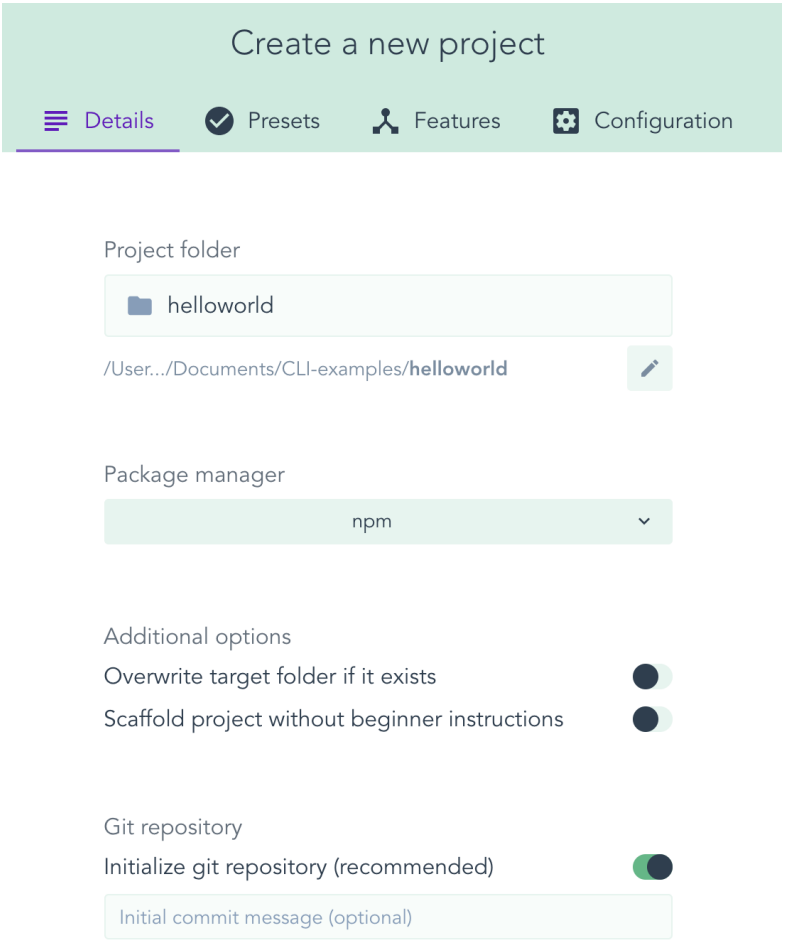
Seuraavaksi CLI pyytää käyttäjää valitsemaan projektin ominaisuudet. Ensikäynnistyksellä on olemassa default-profiili, joka sisältää babel- ja eslint-paketit. Default-profiilin lisäksi on mahdollista valita asennettavat paketit manuaalisesti, kuvassa 7 näkyvät mahdolliset paketit. Tämän jälkeen konsoliin tulee erilaisia asetuskysymyksiä liittyen valittuihin paketteihin, millä managerilla ne asennetaan ja tallennetaanko tämä uudeksi profiiliksi.



KUVA 7 Manuaalinen pakettien valinta

Komentoriviltä projektin voi käynnistää selaimeen komennolla `yarn serve´, kun ollaan projektikansiossa.

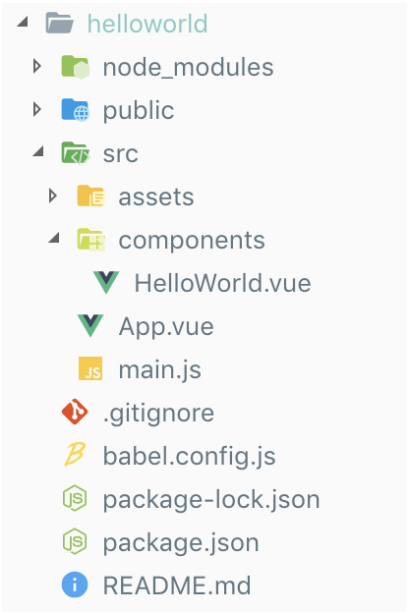
Komennolla `vue ui` käynnistetään selaimeen avautuva Vue Project Manager. Managerista näkee nykyiset Vue-projektit, lisäksi voi luoda uuden projektin tai tuoda jo olemassa olevan projektin. Projektin luomisen ensimmäinen välilehti on kuvassa 8.



KUVA 8 Projektin luonti Vue UI:n avulla

Kun kaikki projektin asetukset on tehty, manageri aloittaa projektin luomisen, jonka jälkeen aukeaa dashboard-näkymä. Näkymästä näkee mm. projektiin asennetut lisäosat, riippuvuudet ja mahdolliset konfiguraatiotiedostot, jotka on mahdollista avata ja muokata graafisessa näkymässä.

Kuvassa 9 näkyy projektin rakenne, joka on syntynyt CLI-komennolla. Projektille tärkeät muokattavat tiedostot sijaitsevat src-kansiossa, jonka juuressa ovat App.vue ja main.js, jotka ovat ohjelman pohja, App.vue kutsuu components-kansiossa olevia komponentteja ja siten komponentit näkyvät ohjelmassa.



KUVA 9 Vue.js-projektin kansiorakenne

3.2.3 Syntaksi

Vue-tiedostoissa on kolme lohkoa *template*, *script* ja *style*. Ensimmäisessä lohkossa määritellään ulkoasukomponentit, toisessa logiikka, joka liittyy kyseisiin komponentteihin, ja kolmannessa komponenttien tyyli.

App.vue-tiedostossa (ohjelma 3) templatessa asetetaan Vue-logo sivulle ja kutsutaan HelloWorld-komponenttia, jonka sisältö tulee näkymään logon alapuolella. Script-lohkossa ensin ajetaan import-komennot ja määritellään käytettävät komponentit. Viimeisessä lohkossa määritellään ohjelman yleinen tyyli, koska se ei ole *scoped*, tyylit pätevät myös lapsikomponentteihin.

```
<template>
  <div id="app">
    <img alt="Vue logo" src="./assets/logo.png">
    <HelloWorld msg="Welcome to Your Vue.js App"/>
  </div>
</template>

<script>
import HelloWorld from './components/HelloWorld.vue'

export default {
  name: 'app',
  components: {
    HelloWorld
  }
}
</script>

<style>
#app {
  font-family: 'Avenir', Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

OHJELMA 3 App.vue-tiedoston sisältö

Komponentin HelloWorld.vue rakenne on samanlainen kuin App.vue, seuraavassa koodiotteessa (ohjelmassa 4) *style*-lohko on pienennetty.

Template-lohkossa on *heading*- ja *paragraph*-komponentit, joiden sisällä ovat *msg*- ja *hello*-muuttujat, muuttujia on kutsuttu aaltusulkeiden sisällä. Lisäksi on *button*-komponentti, johon on liitetty *sayHello*-funktio, joka toteutetaan nappia painettaessa.

Script-lohkossa näkyy komponentin nimi, mahdolliset prop-muuttujat, jotka komponentti saa App.vue-tiedosta, kun se hyödyntää *HelloWorld*-komponenttia. Tämän jälkeen on *data*-lohko, jossa ovat kaikki komponentin sisäiset muuttujat. Lisäksi on *methods*-lohko, jonka sisälle toteutetaan kaikki funktiot.

```
<template>
  <div class="hello">
    <h1>{{ msg }}</h1>
    <p>{{hello}}</p>
    <button v-on:click="sayHello">Press here</button>
  </div>
</template>

<script>
export default {
  name: 'HelloWorld',
  props: {
    msg: String
  },
  data() {
    return {
      hello: ''
    }
  },
  methods: {
    sayHello() {
      this.hello = "Hello"
      return this.hello
    }
  }
}
</script>


<!-- Add "scoped" attribute to limit CSS to this component only -->
<style scoped> …
```

OHJELMA 4 *HelloWorld.vue*-tiedosto

3.2.4 DevTools

Vue Devtoosl -kehittäjätyökalut on mahdollista saada Chrome- ja Firefox-lisäosina sekä itsenäisenä Electron-sovelluksena, joka toimii missä tahansa ympäristössä (mobiiliselain, safari, natiivi scripti) (Vue Devtools. n.d.). Vue DevTools mahdollistaa Vue komponenttien tarkastelun selaimessa. DevToolsista voi katsoa, mitä dataa kyseinen komponentti saa, tapahtuneita tapahtumia tai reititystä. Lisäksi DevTools integroituu Vuex-tilanhallinnan kanssa saumattomasti.

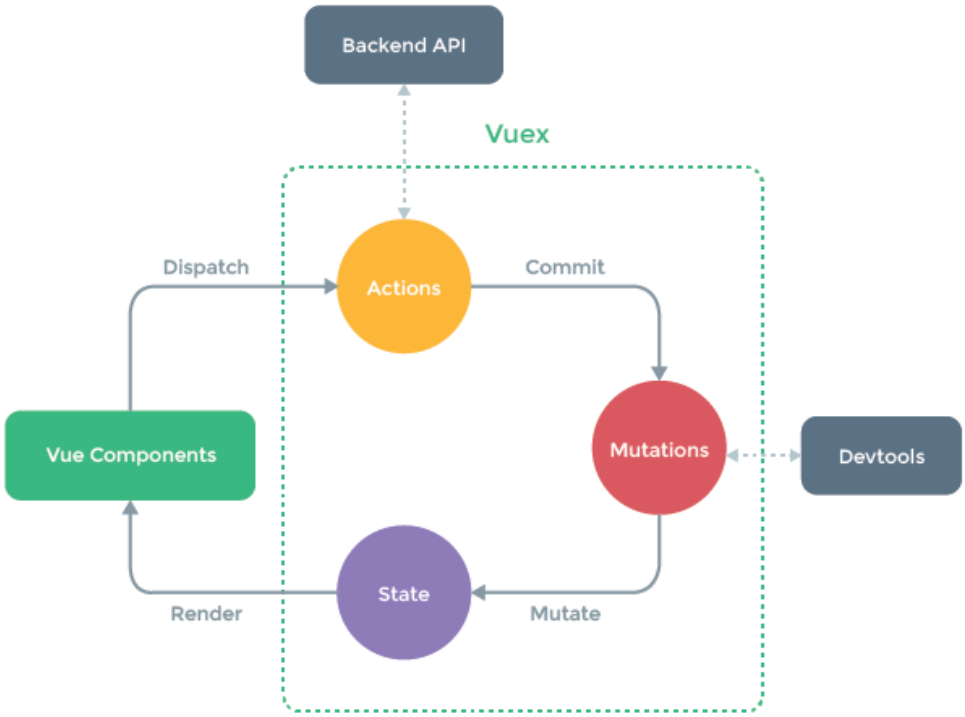
3.2.5 Router

Vuen virallinen reititin on Vue Router, joka integroituu hyvin Vuen kanssa. Routerin avulla on erittäin helppoa luoda SPA-sivuja (Single Page Application). (Vue Router. n.d.)

Router mahdollistaa dynaamiset URL:it, eli osoitteeseen voi asettaa dynaamisesti esimerkiksi käyttäjän id:n, jolloin sivusto osaa avata käyttäjän profiilin. (Vue Router. n.d.)

3.2.6 Tilanhallinta

Vue hyödyntää tilanhallintaan Vuex-kirjastoa, joka perustuu Reduxin kaltaisesti Flux-arkkitehtuuriin. Vuex on Vuen virallinen tilanhallintakirjasto. Se toimii keskeisenä tietovarastona (store) ohjelman kaikille komponenteille sellaisten sääntöjen kera, että tilaa voi muuttaa vain ennalta arvattavalla tavalla. Se integroituu myös Vue DevToolsin kanssa. Kuvasta 10 näkee, kuinka Vue-komponentit kommunikoivat Vuex-tietovaraston kanssa, joka puolestaan kommunikoi backendin ja DevToolsien kanssa. (Vue.js. n.d.)



KUVA 10 Vue + Vuex-järjestelmä (Vue.js. n.d)

3.2.7 Bootstrap

Bootstrapista on tehty Vue-yhteensopiva versio, BootstrapVue. Se sisältää kaikki Bootsrapin komponentit. Se vaatii Portal-Vue-paketin toimiakseen. (BootsrapVue. n.d.)

3.3 Angular

Angular on Googlen tekemä ja ylläpitämä TypeScript-pohjainen framework, joka on avointa lähdekoodia. Angular on AngularJS uudelleen keksittynä. Ensimmäinen julkaisu oli 2016, nimen Angular 2+ alla, uusien julkaisujen mukana versionumero vaihtuu. Viimeisin versio on Angular 7.

3.3.1 Ominaisuudet

Angular sisältää ng-direktiivejä, joiden avulla käyttöliittymäkomponenteissa voidaan toteuttaa eräänlaista logiikkaa: esimerkiksi *if*-lauseita tai *for*-looppeja, joiden avulla voi piirtää taulun kaikki elementit listana.

Angular hyödyntää reaktiivista RxJS-kirjastoa (Reactive Extensions for JavaScript), joka helpottaa asynkronisia tai call-back-pohjaisia ohjelmia. (Angular. n.d.)

3.3.2 CLI

Angular CLI -työkalu mahdollistaa projektin luomisen, kehittämisen ja ylläpitämisen komentoriviltä. Lisäksi on Angular Console -applikaatio, jonka voi asentaa koneelle erillisenä ohjelmana tai lisätä VSCodeen lisäosana (Angular Console. n.d.).

Projektin luominen Angular CLI:n avulla tapahtuu kuvan 11 mukaisella komennolla.

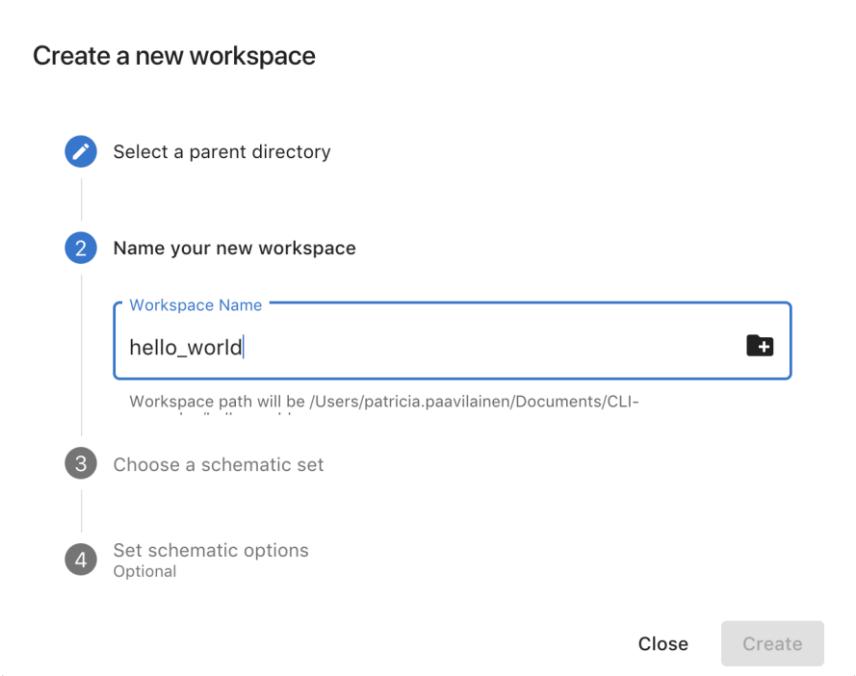
KUVA 11 Angular CLI, projektin luomiskomento

Komennossa *ng* on Angular CLI -työkalun nimike komentorivillä, *new* on komento, joka annetaan työkalulle, ja *hello-world* on uuden projektin nimi. Työkalu tämän jälkeen kysyy, haluaako käyttäjä asentaa Angular Routerin samalla ja mitä tyylitiedostoja käytetään (CSS, SASS, SCSS yms.). Komento luo nykyiseen kansioon hello-world-nimisen kansion, jossa juuri luotu projekti sijaitsee.

Projektikansiossa voi käyttää komentoa *ng serve*, joka pystyttää projektin lokaalisti selaimessa nähtäväksi sovellukseksi. Sovellus on oletuksena portissa 4200, eli sen löytää osoitteesta *http://localhost:4200/*. Niin kauan kuin *ng serve* on komentorivillä päällä, työkalu automaattisesti päivittää sivun, kun koodiin tehdään muutoksia.

Erillinen aplikaatio Angular Console, avaa työpöydälle ikkunan, josta voi luoda uuden Angular-projektin tai lisätä jo ennestään olemassa olevan. Kun luo uuden projektin, aplikaatioon aukeaa kuvan 12 mukainen dialogi. Siinä voi valita, mihin

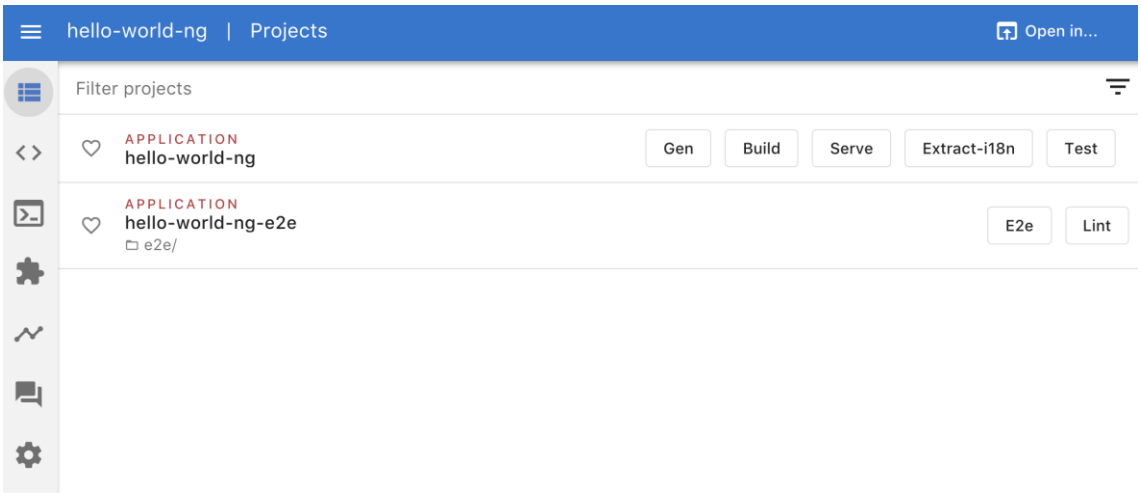
kansioon projekti luodaan, mikä projektin nimi on ja millaista pohjaa se käyttää: *default* vai *enterprise*.



KUVA 12 Angular Console, uuden projektin lisääminen

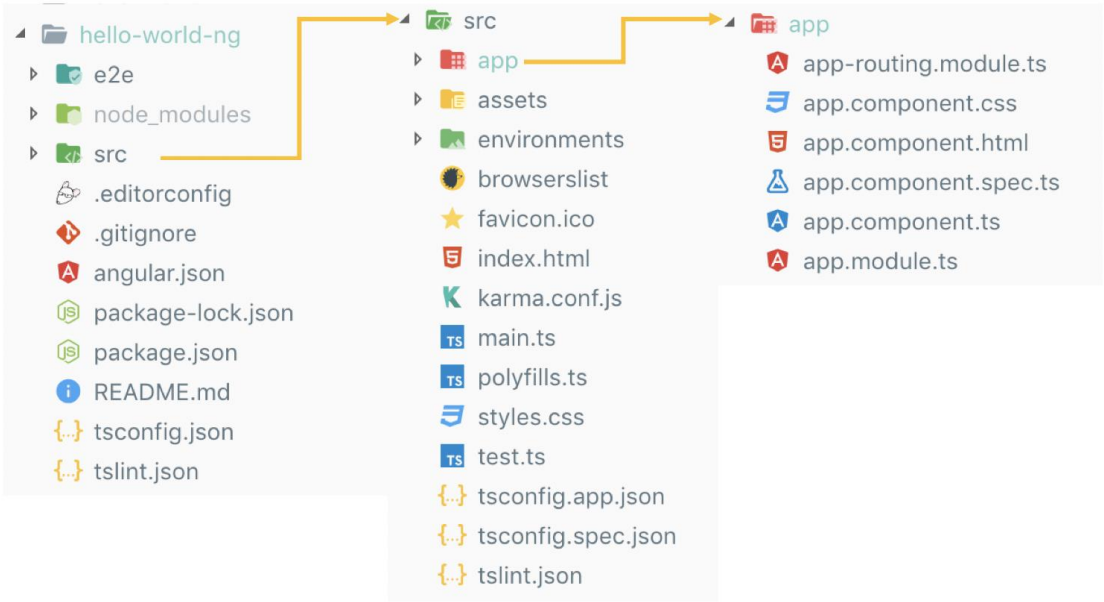
Tämän jälkeen ikkunaan aukeaa konsolinäkymä, jossa näkyvät komennot, jotka ohjelma ajaa pystyttäessään projektia.

Tämän jälkeen projekti näkyy dashboard-tyyppisessä näkymässä, josta projektia voi hallita (kuva 13).



KUVA 13 Angular Console, projektin hallintanäkymä

Angular-projektin kansiorakenne on kuvassa 14, projektin sisällä on paljon tiedostoja ja kansioita.



KUVA 14 Angular-projektin rakenne

3.3.3 Syntaksi

Angular-projektissa komponenttien logiikka, käyttöliittymä ja tyylit on jaettu eri tiedostoihin, app.component.ts/html/css. Kaikki alkaa index.html-tiedostosta,

jossa kutstuaan app-root-komponenttia, joka puolestaan kutsuu muita komponentteja.

Alla on tiedoston app.component.ts sisältö (ohjelma 5). Tiedostossa alustetaan komponentti ja luokan sisällä luodaan *hello*-muuttuja, joka alustetaan *constructorissa*. *Constructorin* jälkeen on *sayHello*-funktio, joka päivittää *hello*-muuttujan arvon.

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  public hello: string;

  constructor() {
    this.hello = ""
  }

  public sayHello(){
    this.hello = "Hello"
  }
}
```

OHJELMA 5 Angular-projektin tiedosto app.component.ts

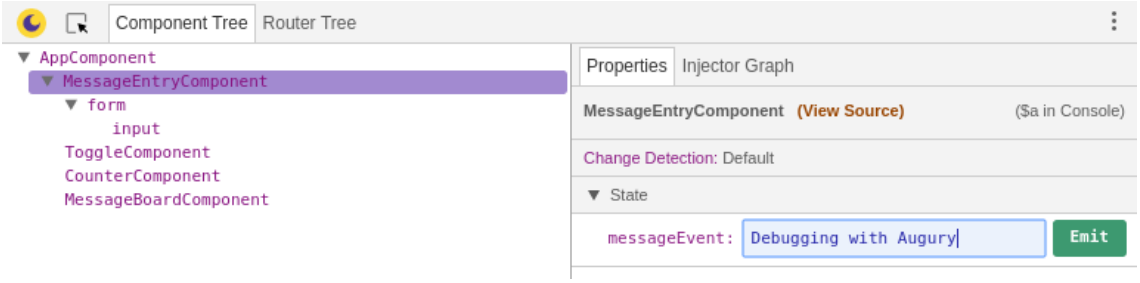
Funktiota *sayHello* kutsutaan tiedostosta app.component.html, kun painiketta on painettu (ohjelma 5). Komponentissa ei ole muuta kuin *heading*-komponentti, jossa näytetään *hello*-muuttujan arvo, sekä painike, jolla kutsutaan *sayHell*-funktiota. *Router-outlet*-komponentti on tiedostossa automaattisesti, jos projektia luotaessa lisäsi Angular Routerin projektiin.

```
<div style="text-align:center">
  <h2> {{ hello }} </h2>
  <button (click)="sayHello()">Press here</button>
</div>

<router-outlet></router-outlet>
```

OHJELMA 6 Angular-projektin tiedosto app.component.html

3.3.4 DevTools

Angularin DevTools on Angular Augury, joka on saatavilla Chromelle ja Firefoxille lisäosana. Työkalussa voi puunäkymässä tarkastella Angular-komponenttien saamaa dataa. Halutessaan työkalun kautta voi syöttää komponenteille dataa testatakseen komponentin toimintaa. (Augury. n.d.)



KUVA 15 Angular Augury -työkalu selaimessa (Augury. n.d.)

3.3.5 Router

Angular Router on Angularin virallinen reititysominaisuus, joka on Angular-tiimin kehittämä. Angular Routerin tarvitsee, jos haluaa liikkua näkymästä toiseen ohjelman sisällä.

3.3.6 Tilanhallinta

Angularilla on kaksi eri tilanhallintajärjestelmää: NgRx ja NGXS. NgRx on RxJS-pohjainen framework, jonka mukana tulee myös muita ominaisuuksia kuten Angular Routerin kiinnittäminen tietovarastoon. NGXS on tilanhallintakirjasto, jonka arkkitehtuuri pitkälti muistuttaa NgRx- ja Redux-järjestelmiä. Lisäksi

Angular voi hyödyntää RxJS-kirjastoa. Angularilla on siis monta erilaista tilanhallintajärjestelmää, jota se voi hyödyntää, näistä kaikista suosituin on NgRx (De Semet, O. 2018).

RxJS-kirjasto on tarkoitettu reaktiivista ohjelmointia varten, se sisältää *Obervable*-tyypin, joka helpottaa asynkronisen tai call-back-pohjaisen koodin toteuttamista. (Angular. n.d.)

NgRx on RxJS-pohjainen tilanhallintajärjestelmä, joka on ottanut inspiraatiota Reduxista. Se tarjoaa tilanhallintajärjestelmän lisäksi sivuvaikutusten eristyksen, router-kiinnityksen ja paljon muuta. (NgRx. n.d.)

3.3.7 Bootstrap

Angularilla on kaksi Angular-pohjaista Bootstrappia, jotka on luonut eri Angular-yhteisön tiimit, ng-Bootstrap ja ngx-Bootstrap. Molemmissa ovat samat Bootstrap-komponentit saatavilla, mutta eri tyyleillä. Näiden kahden välillä ng-Bootstrap on suositumpi.

4 VERTAILU

Huomioitavaa on, että vertailun tarkoituksena ei ole valita parasta tai huonointa kieltä, sillä kielen parhaus tai huonous on subjektiivista, johon vaikuttaa, mitä projektia ollaan tekemässä ja myös kunkin henkilökohtainen preferenssi. Tarkoituksena on selvittää, miten frameworkkien ominaisuudet on toteutettu ja miten ne eroavat toisistaan.

4.1 Ominaisuudet

Taulukossa 2 näkyy muutamia tärkeitä ominaisuuksia, joita joillakin näistä frameworkeista on. Kaikki kolme frameworkkia ovat komponenttipohjaisia, tosin ne toimivat hieman eri tavoin. Tämä nostaa koodien uudelleenkäytettävyyttä, sillä samaa komponenttia voidaan hyödyntää useissa paikoissa.

Tästä frameworkkien erot alkavat, React ja Vue molemmat hyödyntävät VDOMia, jotta vain tarvittavat elementit päivitetään. Angular puolestaan käyttää vain tavallista DOMia.

Angular on täysin TypeScript-pohjainen ja Reactilla on osittain TypeScript tuki, eli React on normaalisti JavaScript-pohjainen, mutta projektit voi halutessaan tehdä TypeScriptillä. Vue ei tue TypeScriptiä ollenkaan.

Angular ja Vue omaavat kasan direktiivejä kuten v-on/ng-on ja v-if/ng-if, joilla voi toteuttaa logiikkaa käyttöliittymäkomponenteissa. Lisäksi Angular ja Vue voivat hyödyntää kaksisuuntaista tiedon sitomista. React puolestaan käyttää vain yksisuuntaista, josta on tietyissä tilanteissa hyötyä.

TAULUKKO 2 Angularin, Reactin ja Vuen ominaisuudet

Ominaisuus	**Angular**	**React**	**Vue.js**
Komponenttipohjainen	x	x	x
VDOM		x	x
TypeScript-tuki	x	x	
Direktiivit	x		x
Kaksi suuntainen tiedon sitominen	x		x

4.2 CLI

Angularilla ja Vuella on kunnon CLI-työkalut, joiden avulla komentorivillä voi säätää projektin asetuksia jo luomisvaiheessa. Lisäksi molemmilla on graafinen käyttöliittymä, jossa projekteja voi luoda, hallita tai lisätä jo olemassa olevia projekteja. Reactilla ei ole varsinaista CLI-työkalua, vaikka se hyödyntä Create React Appia. Siinä ei voi antaa kuin yhden komennon, jolla projekti luodaan. Projektin luomisvaiheessa ei voi tehdä mitään asetuksia siihen liittyen.

4.3 Syntaksi

Kaikkien kolmen syntaksi ja projektirakenteet ovat hyvin erilaiset, etenkin Angularin. Tähän toki vaikuttaa, että Angular on TypeScript-pohjainen ja React ja Vue JavaScript-pohjaisia.

Angularissa yksi komponentti on jaettu kolmeen eri tiedostoon: logiikka, käyttöliittymä ja tyyli. Tämä tekee komponenttien tiedostoista lyhyempiä ja selkeämpiä, mutta projektin kansiorakenne saattaa kärsiä siitä, että jokaista komponenttia varten on useita tiedostoja.

React ja Vue pitävät logiikan ja käyttöliittymän samassa tiedostossa. Reactissa tämä tapahtuu kaikki luokan sisällä, jossa on JSX-tyylillä käyttöliittymä kirjoitettu *render*-funktion sisälle. Vuessa tiedoston alussa on *template*-lohko, jonka sisällä on HTML-tyylisiä komponentteja, ja tämän jälkeen *script*-lohko, jossa kaikki komponentin logiikka suoritetaan. Tyylien lisäys on eri lailla hoidettu Reactin ja Vuen välillä, Reactissa kutakin komponenttia varten voi luoda oman CSS-

tiedoston, siinä voi myös lisätä CSS-lohkon samaan tiedostoon. Vuessa CSS on *style*-lohkossa tiedoston lopussa, lohkon voi myös merkata *scoped*-lipulla, jonka seurauksena kyseinen CSS ei vaikuta komponentin lapsikomponentteihin.

Kaikkien kolmen syntaksi käyttöliittymäelementtien kohdalla on samanlaista, koska ne hyödyntävät HTML-pohjaisia elementtejä, kuten *div*, *button* ja *input.* Reactin JSX:n sisällä muuttujia kutsuttaessa tarvitaan *this*-liite ja muuttujan tulee olla yksien aaltosulkujen sisällä. Vuessa ja Angularissa ei tarvita mitään etuliitteitä, mutta puolestaan käytetään kahta aaltosulkuparia muuttujien ympärillä.

Logiikassa kohdalla ovat suurimmat erot: Reactin ja Angularin logiikka on luokkapohjaista, luokissa on *constructor*-elementti, jossa alustetaan arvoja, ja funktiot määritetään tämän jälkeen suoraan luokan sisään. Vuessa voidaan myös hyödyntää luokkia, vaikka sitä ei tehdä kovin usein. Vuen *script*-lohkon sisälle lisätään alalohkoja, kuten *props*, *data*, *computed* ja *methods*. *Computed*-lohkossa tehdään erilaisia laskennallisia toimenpiteitä, *methods*-lohkossa on puolestaan funktiot.

4.4 DevTools

Jokaisella frameworkilla on hyvin saman tyylinen DevTools, josta pääsee tarkemmin tutkimaan komponentteja ja niissä olevaa dataa. Huomattavimpana erona on, että Angular Augury -konsolin voi avata millä tahansa sivulla, vaikkei sivua olisi tehtyä Angularilla. React ja Vue puolestaan eivät anna vaihtoehtoa avata konsolia, ellei lisäosa havaitse sivulla kyseistä frameworkkia.

Reactin työkalulla pystyy katselemaan tietoja, myös julkaistuilla sivuilla, ellei sitä sivun tekijä ole erikseen kieltänyt. Vuen työkalu ei näytä mitään komponentteja julkaistuilla sivuilla, mutta se ei ilmoita asiasta. Augury myöskään ei suostu näyttämään julkaistujen sivujen sisältöä, mutta se antaa ilmoituksen käyttäjälle asiasta.

4.5 Router

Frameworkkien reititinkomponentit ovat hyvin samankaltaisia toiminnoiltaan, niitä hyödynnetään, kun halutaan luoda SPA-sovullus. Kaikissa on mahdollista käyttää dynaamista reititystä.

Erona on, että Angularin ja Vuen router-komponentit ovat niin sanotusti virallisia, eli Angularin ja Vuen kehittäjät ovat tehneet ne. Reactin router on React Training -yhteisön tekemä, eikä Facebookin.

4.6 Tilanhallinta

Frameworkkien tilanhallintajärjestelmät kaikki perustuvat tai ovat ottaneet inspiraatiota jollakin asteella Redux-tilanhallintajärjestelmästä. Seurauksena on, että kaikki järjestelmät toimivat hyvin samankaltaisesti pienillä eroilla. Taulukossa 3 on muutama tilanhallintajärjestelmä ja tieto siitä, mille frameworkille se on tehty.

TAULUKKO 3 Tilanhallintajärjestelmät

Järjestelmä	**Angular**	**React**	**Vue.js**
Akita	x	x	x
NgRx	x		
NGXS	x		
React-Redux		x	
Redux	x	x	x
RxJS	x	x	x
Vuex			x

Redux on yleinen JavaScript-tilanhallintajärjestelmä, jota pystyy hyödyntämään kaikilla frameworkeillä. Toinen yleiskäyttöön sopiva järjestelmä on RxJS, jonka pystyy yhdistämään Reactiin ja Vuen Vuex-järjestelmään. Angular RxJS ei ole perinteinen tilanhallintajärjestelmä itsessään, vaan reaktiivinen JavaScript-kirjasto, joka mahdollistaa asynkronisen ja call-back-pohjaisen koodauksen.

Akita on RxJS:ää pohjanaan hyödyntävä tilanhallintajärjestelmä, joka on yrittänyt ottaa Reduxin ja Fluxin parhaimmat puolet, lopputuloksena on Observable Data Stores Model. Akitaa voi hyödyntää mikä tahansa framework tai vaikka perus-JavaScript-ohjelma. (Akita. n.d.)

Reactilla ja Vuella on vain yksi niille tarkoitettu tilanhallintajärjestelmä, toisin kuin Angularilla, jolla on NgRx ja NGXS.

4.7 Bootstrap

Jokaiselle frameworkille on oma Bootstrap-versio, Angularille ja Reactille jopa useita, jos ajatellaan vain täysin ilmaisia versioita. Vuella on material design -Bootstrap, josta on ilmainen mutta hyvin rajoitettu versio.

Angularin kahden Bootstrap-kirjaston välillä on eroa vain ulkoasussa, ja nekin ovat hyvin saman tyylisiä. Reactin versioiden ulkoasuissa ei vaikuta olevan muita eroja kuin väritys. React-Bootstrapissa on perusvärien lisäksi myös light- ja dark-värit. Suurin ero on komponenteissa, sillä reactstrap on karsinut joitain vähemmän käytettyjä komponentteja, jotta se olisi mahdollisimman kevyt. Vuen Bootstrap-versio sisältää kaikki samat komponentit kuin alkuperäinen Bootsrap, mutta vain Vue-yhteensopivina.

5 POHDINTA

Työssä esiteltiin JavaScriptia ja sen johdannaiskieliä vain pintaa raapaisten. Työhön valittuja Angular-, React- ja Vue.js-sovelluskehyksiä tarkasteltiin hieman tarkemmin. Tarkasteltavat osa-alueet valittiin antamaan tietoa usein käytetyistä toiminnoista, kuten CLI-työkalut, DevTools-työkalut, router ja tilanhallinta, sillä nämä vaikuttavat sovelluskehyksen valintaan. Sovelluskehys pitäisi pystyä valitsemaan projektin tarpeiden mukaan, eikä projektia sovelluskehyksen mukaan.

Eniten tarkasteltuja ominaisuuksia oli Angularilla ja Vuella, mikä ei tee Reactista yhtään huonompaa kieltä. Sen sijaan se vahvistaa sitä, että kaikilla on omat käyttökohteensa.

CLI-työkalujen osalta Angularilla ja Vuella on paljon toimintorikkaammat työkalut. Vuen työkalu itsessään sisälsi mahdollisuuden käyttää graafista käyttöliittymää, Angularille se pitää erikseen ladata, jonka takia se on hieman epäkäytännöllisempää. Reactilla ei varsinaisesti ole CLI-työkalua, joka helpottaisi projektin asetusten määrittelyä heti projektia luotaessa.

Syntaksiltaan kaikki ovat hyvin erilaisia. React hyödyntää JSX-elementtejä ja toteuttaa logiikan JavaScript-luokan sisällä. Angular paloittelee komponentin eri tiedostoihin, joka saa projektikansion paisumaan, toisaalta koodia ei pääse kertymään tiedostoihin satoja tai tuhansia rivejä. Vue tekee asian aivan toisin päin ja kaikki komponenttiin liittyvä koodi on samassa tiedostossa helposti erotettavissa lohkoissa. Näistä aloittelijaystävällisin syntaksin kannalta on Vue, sillä koodit ovat samassa tiedostossa ja erittäin loogisesti eroteltuna, muuttujiin pääsee käsiksi ilman *this*-etuliitettä.

DevTools-työkalut ovat kaikilla hyvin samankaltaisia ja helppokäyttöisiä. Kaikki ajavat asiansa. Routereiden toiminnallisuuksissa ei myöskään ole suuria eroja.

Sovelluskehysten tilanhallintatyökaluissa ja niiden määrissä on jonkin verran eroja ja lisäksi on järjestelmiä, jotka toimivat kaikilla. Kaikista yksinkertaisimmin toimiva järjestelmä on Vuex, joka on Vue-tiimin itsensä tekemä, joten se sopii

Vuen kanssa yhteen täydellisesti. Angularilla on monta erilaista tilanhallintajärjestelmää, joiden välillä on eroja. Tämä tekee tilanhallintajärjestelmän valinnasta vaikeaa, etenkin, jos järjestelmät eivät ole entuudestaan tuttuja.

Jokaiselle sovelluskehykselle on Bootstrap-versioita, jotka on tehty mahdollisimman yhteensopiviksi kyseiselle kielelle. Reactin Bootstrap-versiot ovat näiden kesken parhaimmat, pelkästään siksi, että toinen versioista on tarkoituksellisesti pienempi ja soveltuu pieniin tarpeisiin hyvin. Angularilla on kaksi Bootstrap-versiota, joka vaikuttaa hieman turhalta, koska molemmissa on sama toiminnallisuus, komponentit ja lähes samat tyylit.

Kaiken kaikkiaan jokaisella frameworkilla on omia hyviä ja huonoja puolia, joiden perusteella kannattaa valita projektiin sopiva framework. Aloittelijaystävällisin on Vue. Angular ja React ovat hyviä valintoja, jos haluaa työskennellä TypeScriptin parissa tai hyödyntää olio-ohjelmointi-tyylistä logiikkaa.

LÄHTEET

Akita. n.d. Akita Github-dokumentaatio. Luettu 28.5.2019. https://github.com/datorama/akita

Angular. n.d. Angular-dokumentaatio. Luettu 24.5.2019. https://angular.io/guide/rx-library

Augury. n.d. Augury-dokumentaatio. Luettu 27.5.2019. https://augury.rangle.io/

Bootstrap. n.d. Bootstrap-dokumentaatio. Luettu 27.5.2019. https://getbootstrap.com/

BootstrapVue. n.d. BootstrapVue-dokumentaatio. Luettu 27.5.2019. https://bootstrap-vue.js.org/

Create React App. n.d. Create React App-dokumentaatio. Luettu 26.5.2019. https://facebook.github.io/create-react-app/

De Semet, O. 2018. Blogi-kirjoitus. NGRX VS. NGXS VS. AKITA VS. RXJS: FIGHT! Luettu 24.5.2019. https://ordina-jworks.github.io/angular/2018/10/08/angular-state-management-comparison.html

Morelli, B. 2017. Blogi-kirjoitus. JavaScript – WTF is ES6, ES8, ES2015, ECMAScript? Luettu 25.5.2019 https://codeburst.io/javascript-wtf-is-es6-es8-es-2017-ecmascript-dca859e4821c

JavaScript.info. n.d. JavaScript tutoriaali. https://javascript.info/intro

Lindley, C. 2018. React Enlightenment https://www.reactenlightenment.com/

MDN Web Docs. n.d. What is JavaScript. Luettu 26.5.2019. https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

Minnick, C. 2016. The Real Benefits of the Virtual DOM in React.js. Blogikirjoitus. Luettu 1.4.2019. https://www.accelebrate.com/blog/the-real-benefits-of-the-virtual-dom-in-react-js/

NgRx. n.d. NgRx-dokumentaatio. Luettu 27.5.2019. https://ngrx.io/

Node.js. n.d. Node.js-dokumentaatio. Luettu 1.4.2019. https://nodejs.org/en/

Npm. n.d. Npm pakettimanageri. Luettu 26.5.2019. https://www.npmjs.com/package/react-router

Peyrott, S. 2017. Blogi-kirjoitus. A Brief History of JavaScript. Luettu 26.5.2019. https://auth0.com/blog/a-brief-history-of-javascript/

React. n.d. React-dokumentaatio. Luettu 1.4.2019. https://reactjs.org/docs/

React Devtools. n.d. React Devtools Github-dokumentaatio. Luettu 25.5.2019. https://github.com/facebook/react-devtools

Redux. n.d. Redux-dokumentaatio. Luettu 1.4.2019. https://redux.js.org/

Star History. n.d. Github Star History. https://star-history.t9t.io/

StateofJS. 2018. State of JS survey results 2018. Luettu 26.6.2019. https://2018.stateofjs.com

Tutorialspoint. n.d. VueJS – Overview. Luettu 26.5.2019. https://www.tutorialspoint.com/vuejs/vuejs_overview.htm

Vue.js. n.d. Vue.js-dokumentaatio. Luettu 27.5.2019 https:// vuejs.org/

Vue Devtools. n.d. Vue Devtools Github-dokumentaatio. Luettu 25.5.2019. https://github.com/vuejs/vue-devtools/tree/master

Vue Router. n.d. Vue Router-dokumentaatio. Luettu 27.5.2019 https://router.vuejs.org/

Warcholinski, M. n.d. 10 Famous Apps using ReactJS Nowadays. Blogikirjoitus. Luettu 26.5.2019. https://brainhub.eu/blog/10-famous-apps-using-reactjs-nowadays/