

Bachelor's thesis

Information and Communications Technology

2020

Tung Vo

# WEB APPLICATION DEVELOPMENT WITH REACT AND GOOGLE FIREBASE

– Data visualization



Tung Vo

# WEB APPLICATION DEVELOPMENT WITH REACT AND GOOGLE FIREBASE

- Data visualization

The thesis was commissioned in the context of the project Lean 5S which belongs to the Chemical Engineering department of Turku University of Applied Sciences. The purpose of the Lean 5S project is to develop a digital solution in terms of a web application to replace an existing manual system that helps the students and teachers check the laboratory conditions. The application lets students inspect the laboratory and report the status by filling the laboratory conditions forms. The teachers can view the reports and perform administration tasks. This thesis was produced to introduce the technologies that were used by the Lean 5S development team. Also, the thesis reports the workflow and provides a relevant part of the codebase during the commission period to show the audience how the technologies were applied to achieve the tasks. Among the various range of choices in web technologies, the project team decided to choose React and Google Firebase as the core tech stack for the development.

After the first stage of the Lean 5S project, the beta version of the application was ready to be used and tested within a small group of users. Students can check the laboratory condition by filling the digital forms while the teachers can edit the forms and check the summary table. Also, a line graph will be generated to visualize the data. This graph can be filtered by date and lab number.

## KEYWORDS:

web application, React, Google Firebase, graph, visualize, summary table, data.

# CONTENTS

<b>LIST OF ABBREVIATIONS</b>	<b>6</b>
<b>1 INTRODUCTION</b>	<b>6</b>
<b>2 BACKGROUND</b>	<b>7</b>
2.1 A brief history of web and web application development	7
2.2 JavaScript history	9
<b>3 TECHNOLOGIES</b>	<b>11</b>
3.1 React	11
3.1.1 Components	12
3.1.2 Virtual DOM	17
3.1.3 JSX	21
3.1.4 Why React?	23
3.2 Google Firebase	28
3.2.1 Realtime Database	29
3.2.2 Authentication	30
3.2.3 Cloud Firestore	31
3.2.4 Cloud Storage	32
3.2.5 Hosting	33
3.2.6 Cloud Functions	33
<b>4 DEVELOPMENT</b>	<b>35</b>
4.1 Project goal	35
4.2 Methodologies	36
4.3 Tools & Software	38
4.4 Architecture	41
4.4.1 Student role	41
4.4.2 Professor role	42
4.5 Implementation	42
4.5.1 Create React app	43
4.5.2 Firebase database	43
4.5.3 Database summary page	44
4.6 Results	53

## FIGURES

Figure 1. Stack Overflow survey in 2019. Source: <a href="https://codeburst.io/javascript-trends-in-2020-b194bebc5ef8">https://codeburst.io/javascript-trends-in-2020-b194bebc5ef8</a>	11
Figure 2. Function component.	12
Figure 3. A class component.	13
Figure 4. Rendering a component.	13
Figure 5. Creating multiple instances of a component.	13
Figure 6. Declaring a component state.	14
Figure 7. Referring to state properties.	15
Figure 8. Calling setState() method.	15
Figure 9. React lifecycle. Source: <a href="http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/">http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/</a>	16
Figure 10: Example of a DOM. Source: <a href="https://www.w3schools.com/js/js_htmldom.asp">https://www.w3schools.com/js/js_htmldom.asp</a>	17
Figure 11. Real and Virtual DOMs. Source: <a href="https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/">https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/</a>	18
Figure 12. Creating elements without using JSX.	22
Figure 13. Creating elements using JSX syntax.	22
Figure 14. Writing a JavaScript expression in an HTML tag.	23
Figure 15. StackOverflow survey in 2019. Source: <a href="https://insights.stackoverflow.com/survey/2019#technology-_most-loved-dreaded-and-wanted-web-frameworks">https://insights.stackoverflow.com/survey/2019#technology-_most-loved-dreaded-and-wanted-web-frameworks</a>	25
Figure 16. Seaches summary on Google Trends. Source: <a href="https://trends.google.com/trends/explore?cat=31&amp;date=2019-01-23%202020-01-23&amp;q=Vue.js,React,Angular">https://trends.google.com/trends/explore?cat=31&amp;date=2019-01-23%202020-01-23&amp;q=Vue.js,React,Angular</a>	26
Figure 17. Npm trends. Source: <a href="https://www.npmtrends.com/react-vs-vue-vs-@angular/core">https://www.npmtrends.com/react-vs-vue-vs-@angular/core</a>	27
Figure 18. Firebase features. Source: <a href="https://www.javatpoint.com/firebase-introduction">https://www.javatpoint.com/firebase-introduction</a>	29
Figure 19. Firebase Database vs Traditional Database. Source: <a href="https://www.javatpoint.com/firebase-introduction">https://www.javatpoint.com/firebase-introduction</a>	30
Figure 20. FirebaseUI. Source: <a href="https://github.com/firebase/firebaseui-web">https://github.com/firebase/firebaseui-web</a>	31
Figure 21. An example of Cloud Firestore structure. Source: <a href="https://firebase.googleblog.com/2017/10/cloud-firestore-for-rtdb-developers.html">https://firebase.googleblog.com/2017/10/cloud-firestore-for-rtdb-developers.html</a>	32
Figure 22. Scrum lifecycle. Source: <a href="https://www.vanharen.net/blog/scrum-in-3-minutes/">https://www.vanharen.net/blog/scrum-in-3-minutes/</a>	37
Figure 24. Lean 5S workflow.	41
Figure 25. Firebase database configuration sample code.	44
Figure 26. Import statements.	45

Figure 27. Declaring state and props.	45
Figure 28. Fetching answers.	47
Figure 29. Fetching rooms and data for the graph.	48
Figure 30. Table of answers.	49
Figure 31. Filter form.	50
Figure 32. DOM references.	50
Figure 33. Filter statements.	51
Figure 34. Filter data for the graph.	52
Figure 35. State updating.	52
Figure 36. Clearing the filter.	53

## PICTURES

Picture 1. The original browser in 1993. Source: <a href="http://info.cern.ch/NextBrowser1.html">http://info.cern.ch/NextBrowser1.html</a>	7
Picture 2. Kanban board of Lean 5S.	38
Picture 3. The first UI mockup.	39
Picture 4. Log in interface.	53
Picture 5. Room search.	54
Picture 6. Room information.	54
Picture 7. Questions form.	54
Picture 8. Admin panel .	55
Picture 9. Edit questions page.	55
Picture 10. Edit labs page.	56
Picture 11. The visualized graph.	56

## LIST OF ABBREVIATIONS

AJAX	Asynchronous JavaScript and XML
BaaS	Backend as a Service
CD	Compact disc
CSS	Cascading Style Sheet
DOM	Document Object Model
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
MVVM	Model-View-ViewModel
NCSA	National Center for Supercomputing Applications
PC	Personal computer
QR	Quick Response
SDK	Software Development Kit
SSL	Secure Sockets Layer
UI	User interface
UX	User experience
W3C	World Wide Web Consortium
WWW	World Wide Web

# 1 INTRODUCTION

The Lean 5S project begins with the needs of the lecturers and students of the Chemical Engineering department of Turku University of Applied Sciences. The goal of the project Lean 5S is to develop a web application that makes the task of checking the laboratory condition more convenient and more expeditious.

Based on the research and the growth history of applications in general, this thesis was composed to provide the audience with the technical information of the modern technologies - React and Google Firebase. Also, the thesis provides the detailed workflow of the Lean 5S project and a relevant part of the codebase to demonstrate how the technologies were applied to develop the Lean 5S application.

Chapter one presents the context of the Lean 5S project and the purpose of the thesis. Also, it explains generally how the thesis was structured.

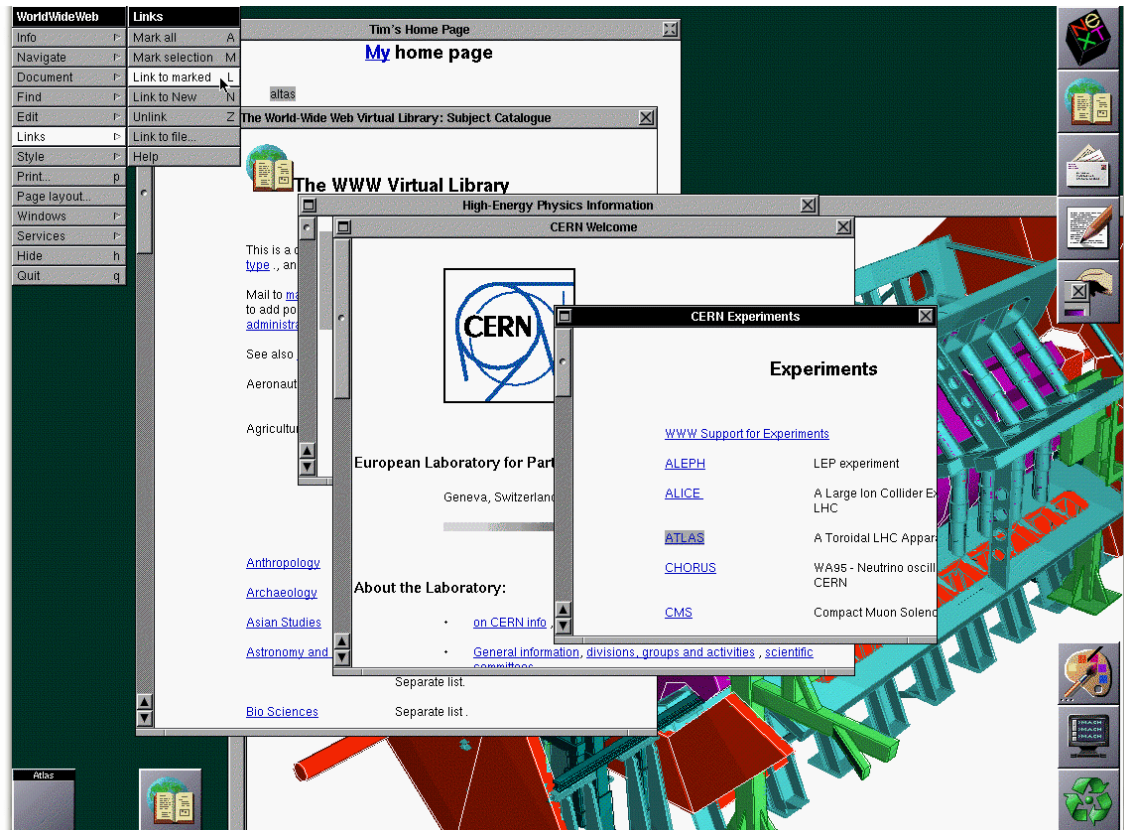
Chapter two of the thesis introduces the development history of the web in general and the web application specifically. It will provide the audience with detailed descriptions and technical aspects, also explain the reason behind the chosen technologies. Besides, it will compare multiple modern web technologies.

Chapter three demonstrates how the project was launched and processed. It includes the methodologies (e.g. brainstorming, project management tools and methods, planning, version control, communication, etc.) and the implementation of the technologies in the project study case.

Chapter four will evaluate the outcomes of the project based on the achievements and the goals set at the beginning of the development. Also, it will discuss the extra features and further improvements for the application.

## 2 BACKGROUND

### 2.1 A brief history of web and web application development



Picture 1. The original browser in 1993. Source: <http://info.cern.ch/NextBrowser1.html>

In 1989, the World Wide Web (WWW) was invented by a British scientist - Tim Berners-Lee. Next, together with Robert Cailliau - a systems engineer, Tim Berners-Lee composed a formal "management proposal" in 1990 to define the web - an interface to display "hypertext documents" on browsers. This is when he defined the Hypertext Markup Language (HTML) - a markup language used to render and create text, images, videos and other types of elements into web pages. During the same year, to actualize his idea, Tim Berners-Lee developed the first server running on a "NeXT computer" and a browser to host his first web. After that, he had modified the original browser and added more functions to grant users the ability to edit the webpage inside the browser (Picture 1) (Home.cern, n.d.). These features were the original version of some features included in the modern browsers.



However, the browser was not popular with everyone around the world because it only ran on NeXT systems. Therefore, the scientists at CERN released the "line-mode" browser in 1991. This browser was built with a "cross-platform codebase" and can be used on various kinds of machines (Line-mode.cern.ch, 2013). After that, due to the lack of human resources, Tim Berners-Lee requested more co-operation from other developers via the Internet to improve the system. As a result, the Mosaic browser was released in 1993 by NCSA. This user-friendly browser defined a route to the future of web design by allowing web pages to display text with images instead of just plain text and tables. It was a big step in popularizing the WWW around the world. By the end of 1993, the WWW hit the record at 1% usage on the Internet which is significant during that time. In 1994, the use of web escalated rapidly and reached 10 million users around the world (Home.cern, n.d.). After that, by the end of 1996, W3C introduced the styling language Cascading Style Sheet (CSS) to the world to enhance the display of hypertext documents. This language allows developers to modify the unique properties (e.g., fonts, colors, size, etc.) of the web elements without changing the HTML. This CSS and HTML duo became the irreplaceable skeleton of the web development.

Besides the web, software applications have also played an important role in the development of computing technology. For decades, developers had written and delivered their applications to customers via physical copies (e.g. CDs) or downloaded packages. These software applications were built to suit the demand of a massive amount of users or individual businesses. Even though this kind of application was considered powerful, secure, and controllable, there are certain significant disadvantages.

First of all, spending unexpected extra time to design and develop application software based on particular demands could lead to problems with developers' budgets. Secondly, a software program that was developed to fit the requirements of some specific businesses might not be compatible with other common software. This problem may put businesses in difficult situations while managing their systems. Moreover, developing application software is considered time-consuming and these software need to be completed before installed directly on the users' systems. Therefore, developers have to fix issues and maintain the software locally in business places. Besides, many users share the application software over the Internet and that may contain threats of viruses that cause bad failures on computers. Finally, developers also have to consider the customers' operating systems (e.g. Mac OS, Microsoft Windows, Linux, etc.) as it will

cost more to produce different versions of the application. Therefore, the "web application" was born and became a direct competitor of software applications.

In 1999, the idea of web application formed in the Java language. A web application (web app) is a cross-platform service that runs in a browser and does not depend on the user's operating system. One version of a web application is usable on any platform and any operating system. This feature made web apps became very popular in the 1990s and 2000s.

After the dot-com bubble chaos of 2000–2002, large tech companies revived to set the new direction for digital commerce and technology. As a result, the Internet advanced to become the main tool for telecommunications during the modern age.

## 2.2 JavaScript history

In September 1995, an employee of Netscape named Brendan Eich developed the scripting language JavaScript. With this new language, the Internet became faster and developers can produce creative interactive web components with functions and events. Even though JavaScript was considered as a client-side language, it was significantly useful and easier to learn than other existing programming languages at that time (e.g., Java, Flash, etc.). Therefore, it soon became popular among web developers and had an open supportive community. Together with HTML and CSS, JavaScript is known as the core language for developing websites.

In 1999, Microsoft improved its Outlook website by added XMLHttpRequest to the browser. This JavaScript function allows the browser to make HTTP requests in XML to retrieve data from the server behind the scenes. This means data can be updated in the client without reloading the entire page content.

In 2005, Jesse James Garrett published the article "Ajax: A New Approach to Web Application" to introduce the Asynchronous JavaScript and XML (AJAX). AJAX is a combination of techniques that enable developers to create asynchronous web apps. It allows data transfer between client and server faster without reloading the whole page. Later on, AJAX advanced to Ajax because JavaScript and XML are not compulsory and the request to a server does not need to be asynchronous anymore.

Nowadays, with the contribution of dedicated developers around the world, countless amount of new technologies were released to make web application development more approachable and more powerful. However, this thesis will target the main technologies that were used to build the project Lean 5S: React and Google Firebase.

## 3 TECHNOLOGIES

### 3.1 React

During the modern age, JavaScript became one of the most powerful and irreplaceable scripting languages in web application development.

#### Programming, Scripting, and Markup Languages

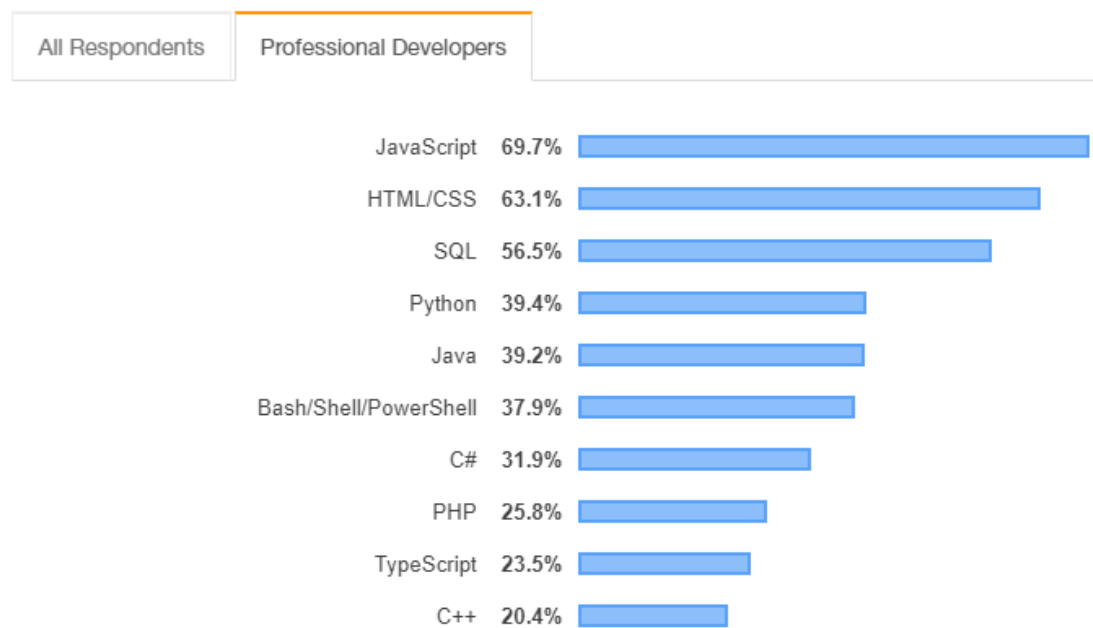


Figure 1. Stack Overflow survey in 2019. Source: <https://codeburst.io/javascript-trends-in-2020-b194bebc5ef8>

According to a survey conducted by Stack Overflow in 2019, 69.7 percent of 72,525 responses show that professional developers around the world are using JavaScript (Figure 1).

However, it is also disturbing sometimes when developers have to re-code the functions every time they want to put them in other web contents. This action is considered either time-consuming or nonsense. Therefore, JavaScript libraries were created to reduce the inconvenience. These libraries are sets of pre-written blocks of JavaScript code that can be used to perform general tasks. Strong tech companies and organizations developed their libraries to improve programming performance and React is one of them.

In 2011, React was introduced by Facebook developer Jordan Walke to improve UI development for their products. After that, it was published as a free open-source and has a big supportive community.

React is a JavaScript library or framework that is used in front-end web development to build interactive user interfaces on websites. It allows developers to create reusable UI components such as navigation bars, menus, pop up modals, buttons, tables, etc. Also, it helps in organizing the development and deliver even more advanced features. To understand and use React efficiently, developers should be familiar with the key features that make React a powerful framework: components, virtual DOM, JSX,. The following information in this section will explain in detail how these features work in React.

### 3.1.1 Components

In a web application, there are tremendous numbers of UI elements. Most of these elements will be duplicated and have the same features. If developers only use plain languages without any library or framework, it will cause consequential problems to the development team while trying to manage the DOM or maintain the system. Therefore, to make it easier for developers, React splits UI elements into independent reusable blocks of codes. These isolated pieces are called components. They contain all of the representations and behaviors of the associated UI elements. Components receive **properties** and can be declared as JavaScript functions or inherited classes of `React.Component`. For example:

- A function component:

```
function Greeting(props) {  
  return <h1>Hi {props.name}, have a nice day!</h1>;  
}
```

Figure 2. Function component.

- A class component:

```
class Greeting extends React.Component {
  render() {
    return <h1>Hi {this.props.name}, have a nice day!</h1>;
  }
}
```

Figure 3. A class component.

These components are not added to the DOM yet. They are the definitions of the UI elements which developers want to implement. To insert the component to the DOM, developers can use the `render()` function from the ReactDOM library. For example:

```
function Greeting(props) {
  return <h1>Hi {props.name}, have a nice day!</h1>;
}

const element = <Greeting name="Tung" />;
ReactDOM.render(
  element,
  document.getElementById('root')
);
```

Figure 4. Rendering a component.

In Figure 4, the `ReactDOM.render()` function takes two arguments. The first one is the new component and the second one is the parent (root) component. This means the component `<Greeting/>` with the property name "Tung" will be added to the DOM element with the id "root". Also, developers can create as many instances as they want from the components without duplicating the component declarations. For example:

```
<Greeting name="Tung" />;
<Greeting name="Vo" />;
<Greeting name="Lam" />;
<Greeting name="Thanh" />;
```

Figure 5. Creating multiple instances of a component.

Besides, developers should remember that components' behaviors must not change their props. However, because modern web apps are dynamic and almost everything should be able to modify, React has another concept called **state**. This can be considered as a storage variable that allows components to modify their data output based on users' actions or other functional executions. A component state is declared inside a constructor and has as many properties as possible. For example:

```
class Student extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = {  
      name: "Tung",  
      group: "PINFOS15",  
      major: "IT",  
    };  
  }  
  render() {  
    return (  
      <div>  
        <h1>Hi!</h1>  
      </div>  
    );  
  }  
}
```

Figure 6. Declaring a component state.

In Figure 6, the component Student has a state containing its name, group and major information. To access these pieces of information, React has an object called **this.state** that contains the component's state. Developers can refer to the state properties anywhere within the component by **this.state.targetpropertyname** syntax. For example:

```
render() {  
  return (  
    <div>  
      <h1>Hi! My name is {this.state.name}</h1>  
      <p>My major is {this.state.major}</p>  
    </div>  
  );  
}
```

Figure 7. Referring to state properties.

To modify the state, developers need to use **setState()** method inside the component. Whenever this method is called, it will trigger React to re-render the component and update the changes. For example:

```
changeName = () => {  
  this.setState({name: "Vo"});  
}
```

Figure 8. Calling setState() method.

The function **changeName()** in Figure 8 is applied to change the state property 'name' of the component Student in Figure 6 from "Tung" to "Vo". Developers need to keep in mind that states cannot be updated in the same way with variables. We have to always use the `setState()` method to update states, otherwise, the component will not be re-rendered. In addition, if a component state contains multiple objects, React will merge the modified objects and other objects will remain the same. A component can be stateless or stateful depend on how developers define their behaviors.

- A stateless component is a component that does not have any state inside. These components will be updated based on the props received from higher-level components.
- A stateful component has its state declared inside a constructor and will be updated based on both state changes and props changes.



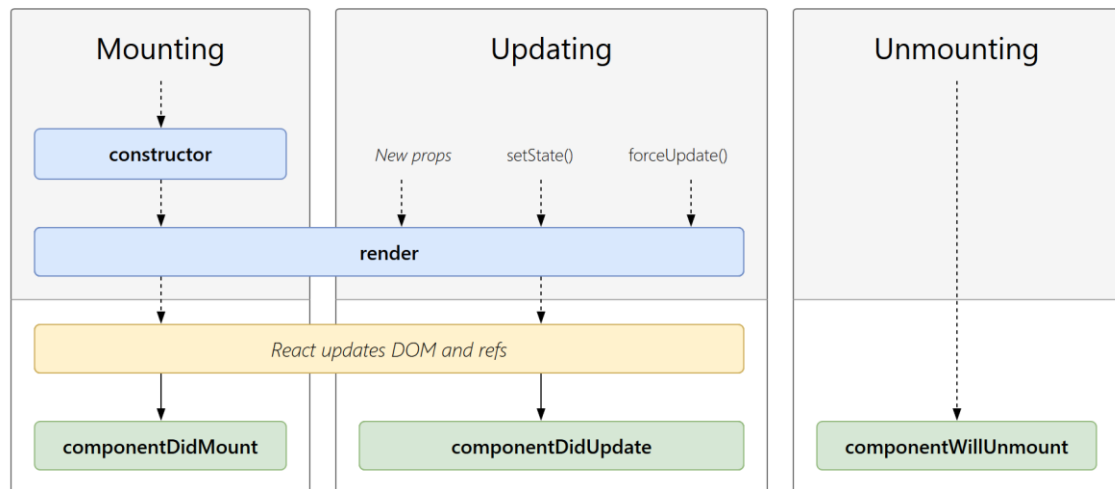


Figure 9. React lifecycle. Source: <http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/>

Besides state and props, React components also have lifecycle methods that representing their phases. There are three main phases when rendering a component: **Mounting**, **Updating** and **Unmounting**(Figure 9).

The first phase of the React lifecycle is **Mounting**. In this phase, the component will be inserted into the DOM. Two main methods are used for this phase: **constructor()** and **componentDidMount()**.

- **constructor()** is called at the beginning of the phase to initiate the component's state and other required values.
- **componentDidMount()** is called after the component was inserted into the DOM. This method is used to run the statements and actions required immediately after the component was inserted such as fetching data, update states, etc.

Next, the changes in the component based on the state and props will be processed in the **Updating** phase. Two main methods for this phase are **shouldComponentUpdate()** and **componentDidUpdate()**.

- **shouldComponentUpdate()** lets programmers declaring a Boolean variable that decides whether React should update the component or not.
- **componentDidUpdate()** allows programmers to write action statements only if the component was updated.

The last phase in the React lifecycle is **Unmounting**. The only method used in this phase is **componentWillUnmount()**. This method is called before the component's removal to do tasks such as canceling time intervals, canceling requests and unsubscribe connections declared in `componentDidMount()`.

Because not every component has the same rendering behaviors, choosing the right methods of the React lifecycle helps developers to control the component's behaviors through every phase of the rendering process efficiently and boost the application performance.

### 3.1.2 Virtual DOM

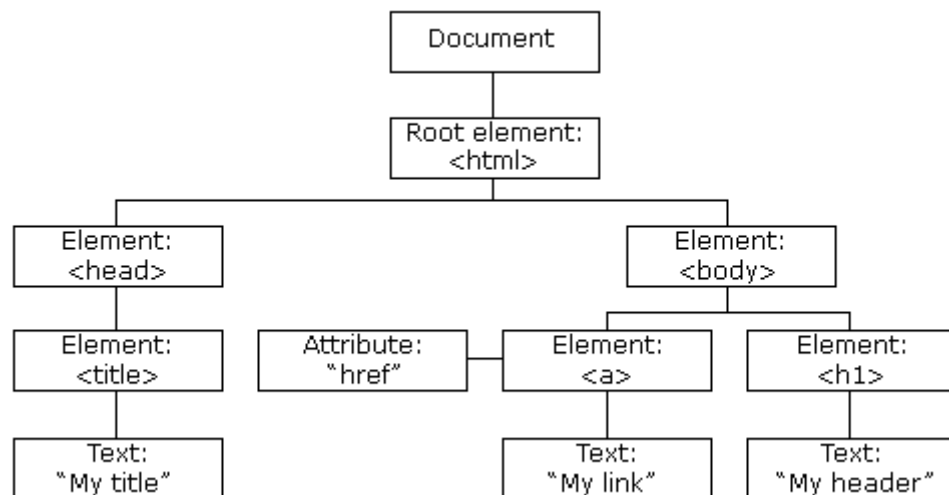


Figure 10: Example of a DOM. Source: [https://www.w3schools.com/js/js\\_htmlDOM.asp](https://www.w3schools.com/js/js_htmlDOM.asp)

In a browser, a hierarchy tree called Document Object Model (DOM) is created by layout engines after a web page was loaded. It describes the relationship of elements in an HTML document. Each node of the DOM represents an element in the web application. The child elements will inherit the characteristics of their parent elements. With this DOM, developers can manipulate the HTML document using JavaScript. There are many possibilities such as modifying (e.g. change, add, remove, etc.) HTML elements and attributes, toggling CSS classes, creating events on the page, etc. Figure 10 is an example of a DOM.

Even though the DOM helps developers to manipulate the HTML document, it has a drawback that causes bad performance to the web pages. For example, when an element of the page is modified, the browser is forced to reload the whole HTML page to update the DOM tree. This may not be a problem in simple static web pages with basic interactions. But nowadays, websites and applications are more dynamic and have large numbers of elements. Loading the whole page seems to be not an appropriate idea because it will slow down the performance of the application and it does not look smooth. To solve this problem, the development team of React integrated a feature called Virtual DOM.

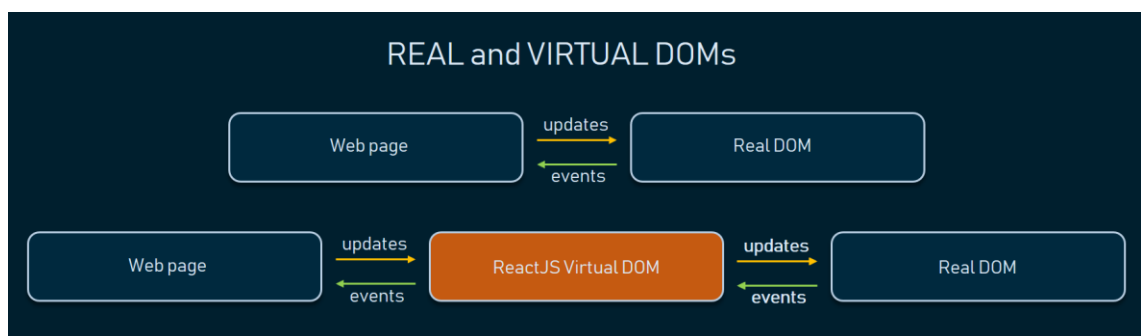


Figure 11. Real and Virtual DOMs. Source:

<https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/>

This feature allows the application to keep the copy of a UI element in memory and will be synced with the original DOM by the ReactDOM package. So, instead of modifying directly on the original DOM, the changes will be updated in the Virtual DOM which is an abstract of the real DOM and only the changed elements will be re-rendered on the page. This process is defined as the term **reconciliation**.

After created React app components, the programmer will need to use the built-in function **render()** to create a DOM elements tree. And, whenever a component's states or props are updated, the **render()** function will return a new DOM elements tree of that component. Then, React performs the  $O(n)$  algorithm to find methods to update the UI that matches the new DOM tree. To do this efficiently, React will rely on two premises that can be used practically in most cases:

- Elements of different types will provide different DOM trees.
- Unique key properties will be used to differentiate the child elements.

When differentiating between the old DOM and the new DOM, React starts by comparing two root elements from both DOMs. After that, it will act based on different situations.

If the root DOM elements are not the same type, React will destroy the old DOM node and generate the new one completely from element to element. This will add the new DOM node to the DOM and remove all the states belong to the old DOM node. The instances of the component are processed through the React component lifecycle in the following order:

1. `componentWillUnmount()`
2. `componentWillMount()`
3. `componentDidMount()`

If the root DOM elements are the same type, instead of regenerate the whole node, React will only update the DOM node based on its attributes. For example:

- Old element:

```
<div className="old" ></div>
```

- New element:

```
<div className="new"></div>
```

In this case, React will keep the target node and only change its *className* attribute from “old” to “new”. It works the same way when developers modify other attributes such as id, style, etc. After, the changes were done, React continuously iterate the same process on the child elements.

If the component elements are the same type, all changes from the component will affect its instances. The state will remain the same during renders but the props will be updated to match the new changed element. Next, the associated instances will be processed through the following functions:

- `componentWillReceiveProps()`
- `componentWillUpdate()`

Then, React executes the recursive process on the previous version and the new version to apply the changes after the `render()` function is requested.

When differentiating between old and new nodes, React executes the recursion on both lists of child elements. It will keep the identical elements and mutate only the different ones. For example:

- Old node:

```
<ul>
  <li>Turku</li>
  <li>Helsinki</li>
</ul>
```

- New node:

```
<ul>
  <li>Turku</li>
  <li>Helsinki</li>
  <li>Tampere</li>
</ul>
```

In this case, React will keep the `<li>Turku</li>` and `<li>Helsinki</li>`, only insert the `<li>Tampere</li>`. However, in a bad scenario, if the new element is added on top of the child list, React will iterate through the entire list instead of just change the different ones. For example:

- Old node:

```
<ul>
  <li>Turku</li>
  <li>Helsinki</li>
</ul>
```

- New node:

```
<ul>
  <li>Tampere</li>
  <li>Turku</li>
  <li>Helsinki</li>
</ul>
```

Because the first element in the child lists are different, React will generate an entirely new list. This will slow down the performance when rendering and opposes the benefit of Virtual DOM. Therefore, to solve this problem, React allows elements to have their **key** attributes. It helps React to recognize the elements which have the same key and

keep them. By knowing which elements to update, the rendering process will be easier and faster. For example:

- Old node:

```
<ul>
  <li key="101">Turku</li>
  <li key="102">Helsinki</li>
</ul>
```

- New node:

```
<ul>
  <li key="103">Tampere</li>
  <li key="101">Turku</li>
  <li key="102">Helsinki</li>
</ul>
```

In this case, the key "103" is new and React will only update the element that keeps the key "103". Even though this key attribute is helpful, it should be used with caution because the key must be unique for each element. Also, developers can use the element's index as a key. However, this approach works well only if the order of the child elements is not changed.

### 3.1.3 JSX

React supports transpilers (e.g. Babel, etc.) that allow preprocessors to transform HTML text written in JavaScript files into standard JavaScript. The syntax extension using for this parsing is called **JSX**. Unlike before, instead of separating the markup elements and the logical behaviors in different files, React allow developers to put HTML into JavaScript files. Therefore, it becomes easier for developers who are already familiar with the traditional syntax to create HTML elements. This approach is quite convenient for front-end developers and designers to modify the elements because they do not have to learn the new way of creating elements in React directly with the **React.createElement()** method.

```

var media = React.createElement(
  "ul",
  { id: "links" },
  React.createElement(
    "li",
    null,
    React.createElement(
      "a",
      { href: "#" },
      "Facebook"
    )
  ),
  React.createElement(
    "li",
    null,
    React.createElement(
      "a",
      { href: "#" },
      "LinkedIn"
    )
  )
);

```

Figure 12. Creating elements without using JSX.

```

var media = (
  <ul id="links">
    <li><a href="#">Facebook</a></li>
    <li><a href="#">LinkedIn</a></li>
  </ul>
);

```

Figure 13. Creating elements using JSX syntax.

Figure 12 and Figure 13 are examples of creating UI elements with and without JSX. These two methods render the same component but the JSX syntax is simpler. It shortens the work of programming and makes the code clearer to understand. It can be considered as a shortcut to call the `React.createElement()` method. React also allow developers to write expressions and variables in HTML tags using `{ }` braces.

```
const sentence = <h1>I have {5 + 5} tickets</h1>;  
// I have 10 tickets|
```

Figure 14. Writing a JavaScript expression in an HTML tag.

### 3.1.4 Why React?

Besides React, there are two other popular JavaScript frameworks: Angular and Vue. Even though React, Angular, and Vue are located in the same category as JavaScript frameworks, they have their own major features which made them different. This section will present brief comparisons between these three frameworks and give the audience an objective perspective on the differences among them. The comparisons will base on three main factors: key features, learning curve, and popularity. Also, discuss the drawbacks of each framework.

#### Key features

- React can be considered as a component-oriented framework. Almost every content and element of an application are presented as a reusable component. React also support Virtual DOM which makes the processes of rendering and modifying components become significantly faster and controllable. Besides, React allows developers to write HTML in JavaScript files by using JSX syntax. Then, transpilers such as Babel will convert them into JavaScript code. This helps developers to manage the appearances and behaviors of components effectively. In addition, React is a lightweight framework compared to other most used ones.
- Angular is a more advanced and complete JavaScript framework that is based on TypeScript. Unlike React, Angular works with directives which are connections between HTML elements and their behaviors. Also, Angular offers the Angular-language-service containing an autocomplete feature that helps the programming work more convenient. And, Angular has a layout called Model-View-ViewModel (MVVM) that lets programmers work independently on shared files and data sets. Most importantly, the structure and the content of Angular



(e.g. Components, Pipes, Modules, etc.) are suitable for building large scale and complicated applications.

- Vue is the newest framework among these three. It seems to inherited core features from the other two (e.g. Components, MVVM, etc.). Vue provides a simple structure that makes developers find it flexible and easy to work with. Also, Vue was designed in a way that helps developers to integrate its package into an existing JavaScript project promptly.

### **Learning curve**

- React has detailed guidance that helps developers to be able to set up a project in a short amount of time. And, because it is not a fully functional framework, developers will have to integrate other third-party libraries if they need more advanced features. Also, it has a wide supportive community and most of the general solutions can be found on the StackOverflow and other related pages. Therefore, React is considered easy to learn and use for in-demand projects.
- Angular's learning curve is quite steep because it is a complete framework that contains most of the essential features to build an application. It requires developers to be familiar with related techniques such as TypeScript. Hence, it requires more time to master the skills in Angular. But if developers are willing to spend their time, this powerful framework is highly recommended for building high complexity projects without installing third-party libraries.
- Vue is considered an easier framework to learn because of its customizability and flexibility. Developers who are already familiar with the concepts of React or Angular can get used to Vue quickly. But because of the simplicity of Vue, developers can have problems in debugging and manage their projects.

### **Popularity**

- React was released in 2013 and backed by Facebook. Its service is trusted and used by popular companies such as BBC, Whatsapp, Instagram, etc.
- Angular was introduced in 2010 and backed by the giant in technology Google. It is popular among big platforms such as Github, Wix, Forbes, BMW, Google, etc.

- Vue was published in 2014 by an ex-employee of Google - Evan You. Even though it is not backed by a famous company, it became well known because of its adaptability based on the other frameworks and improvements. It is also used by influential companies such as Adobe, GitLab, 9GAG, Nintendo, etc.

### Most Loved, Dreaded, and Wanted Web Frameworks

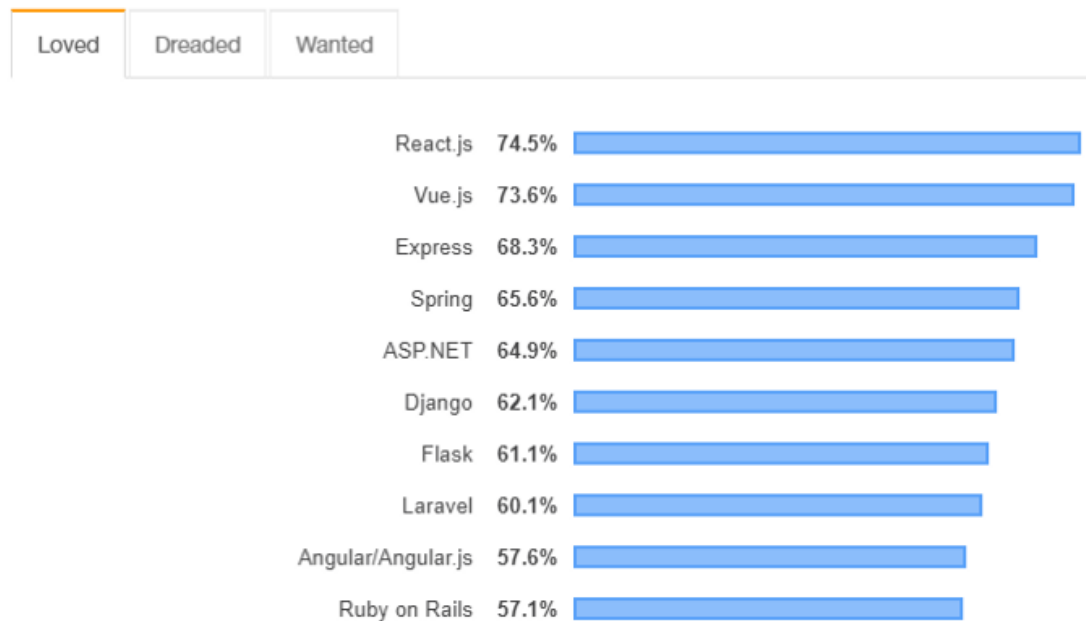


Figure 15. StackOverflow survey in 2019. Source: [https://insights.stackoverflow.com/survey/2019#technology-\\_most-loved-dreaded-and-wanted-web-frameworks](https://insights.stackoverflow.com/survey/2019#technology-_most-loved-dreaded-and-wanted-web-frameworks)

Figure 15 presents the result of a survey conducted by StackOverflow in 2019. In this survey, the numbers show that React and Vue are the most interesting frameworks with percentages of 74.5% and 73.6% respectively. Angular is far behind with 57.6%. We can assure that React and Vue are improving their effort to draw attention from developers. However, another research based on the number of keyword searches on Google Trends shows a different situation.

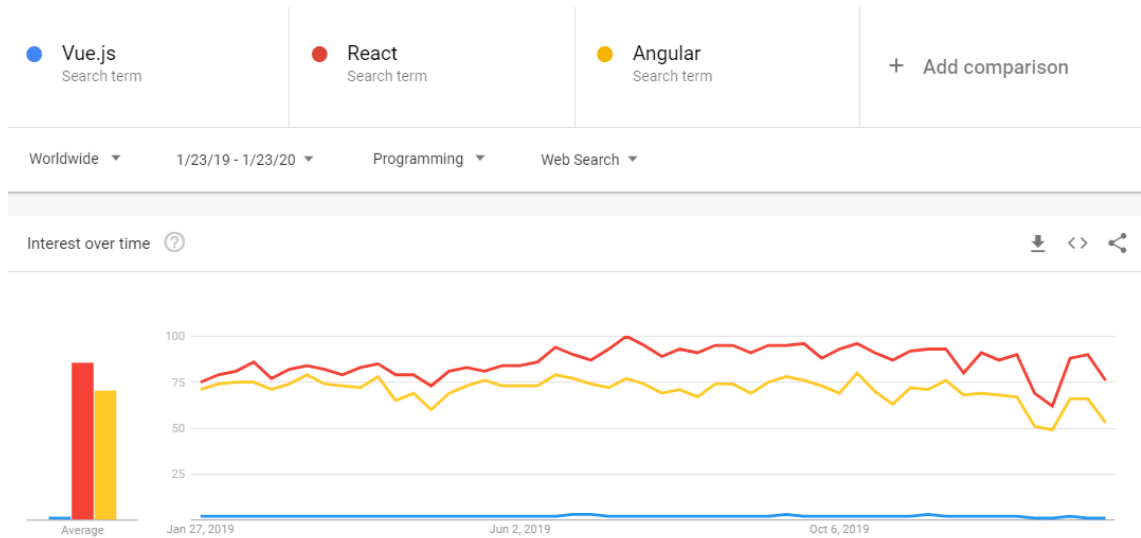


Figure 16. Seaches summary on Google Trends. Source:  
<https://trends.google.com/trends/explore?cat=31&date=2019-01-23%202020-01-23&q=Vue.js,React,Angular>

The statistics in Figure 16 proved that React and Angular are competing for the first position in searches on Google. And, developers behind Vue should consider this to improve the popularity of this framework. The author of this thesis continued to take another test on the number of downloads on npm trends and there was a significantly different result. In Figure 16, during the past six months, React stayed on top by the number of npm downloads and the other two are competing to bypass each other.



## react vs vue vs @angular/core

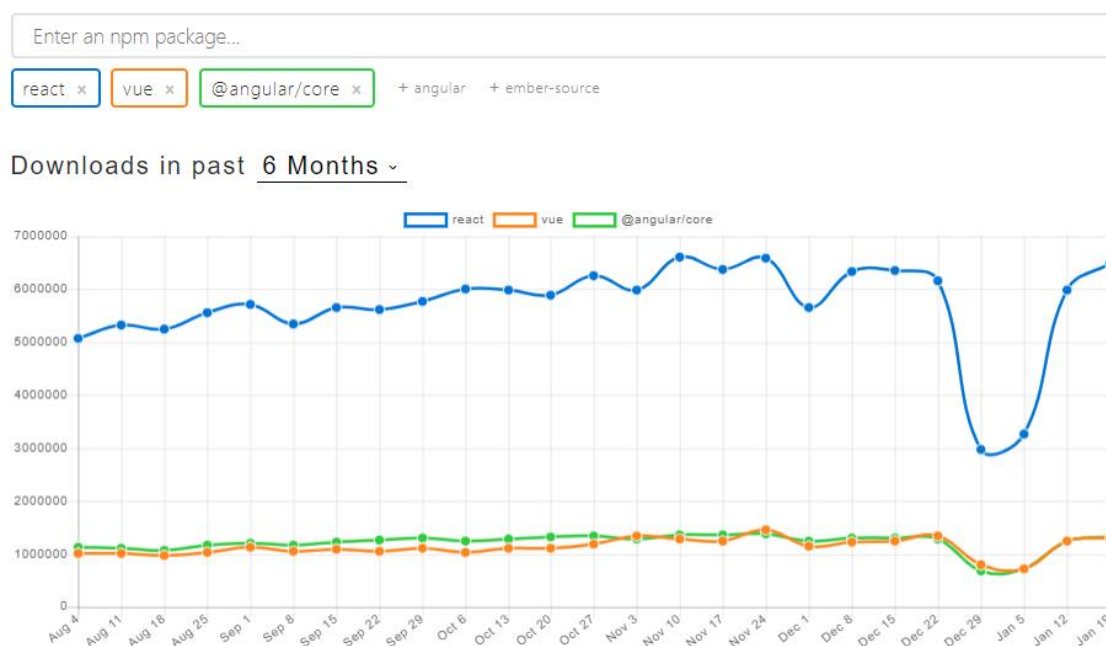


Figure 17. Npm trends. Source: <https://www.npmtrends.com/react-vs-vue-vs-@angular/core>

Based on the statistics and surveys conducted for this section, the popularity of these three frameworks is considered unstable and unpredictable. Creators behind the frameworks are improving their products to satisfy the demand of the market by reducing the drawbacks and adding more suitable features. Nowadays, there are tremendous numbers of projects that vary from light-weight static pages to extremely complex web applications. Therefore, the choices of frameworks and libraries mainly depend on the projects' scales and developers' interests. And, even though the learning curve is an important factor to be considered, each framework has its own advantages that match specific types of projects. Hence, developers need to do researches thoroughly and consider the essential factors to choose the most suitable JavaScript framework for their projects.

As presented in the previous sections, each framework has its strengths and can be a useful tool for web application development. However, the Lean 5S is a light-weight project that is not required to use extra advanced features from complete frameworks such as Angular. Also, working with a flexible framework like Vue might increase the risk

of human mistakes. Besides, the project needed to be launched as expeditiously as possible and the team members have decent skills with the React framework. Therefore, after thoroughly considered the main factors, the project team agreed that React is the most suitable framework for this project and the members' experience.

### 3.2 Google Firebase

In the modern age, based on the demand of customers and businesses, applications need to be planned and developed swiftly. However, practical work is taking a huge amount of time especially when it comes to backend development. Numerous factors to be considered such as security, processing time, usability, convenience, programmers' knowledge, etc. To face these matters, in 2012, Google launched an application development tool on Google Cloud Platform called Firebase. It is a backend service (BaaS) that includes powerful features to support developers in modern application development. Especially, Firebase offers cloud services (e.g. database, hosting, analytics, etc.) that help developers with storing and fetching data from their applications on any devices or web browsers. When working with Firebase, developers will save their time from creating backend parts such as SQL database or server-side scripts. Hence, developers will be able to focus on creating high-quality user interfaces and functionalities. The most powerful feature of Firebase is the real-time database which means data will be synced automatically across accounts instantly. To gain a deeper understanding of Firebase and the reasons why it was chosen as the backend technology for this project, the following information will explain in detail about its most used features for web application development.

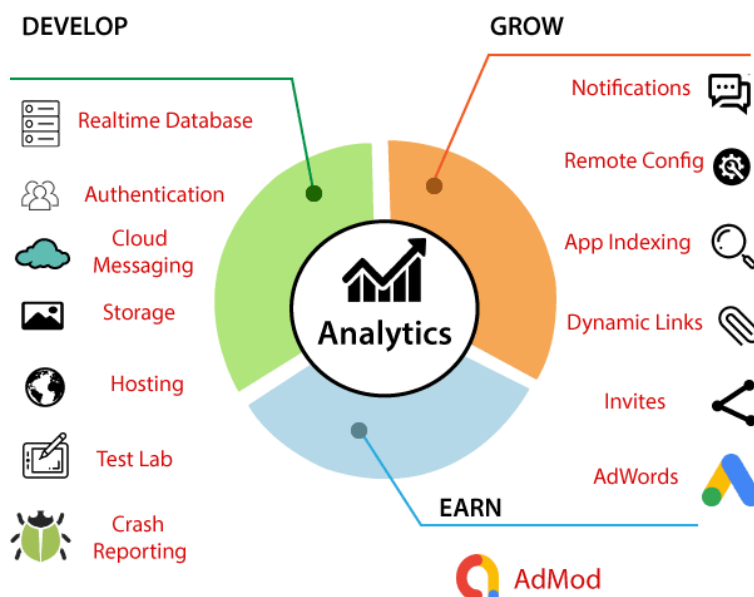


Figure 18. Firebase features. Source: <https://www.javatpoint.com/firebase-introduction>

### 3.2.1 Realtime Database

In the traditional way of transferring data between the client-side and the server-side, developers need to make HTTP requests to retrieve data. This method is passive and insecure if not strictly concerned. The data will only update by the HTTP requests. Therefore, Firebase changed the old concept by providing the Realtime Database. It is a NoSQL database service that allows developers to update data across accounts in realtime. The data transferring between the client and the database will be processed via WebSockets which contain a huge amount of information. Every time the data is updated from a client, they will be synchronized immediately on other associated clients. These data are stored as a JSON format on the Firebase cloud. Therefore, users can access the database from any device by APIs without connecting to a server. Besides, even if there is no Internet connection, the data will be stored temporarily on the local device by cache and updated automatically when it is back online.

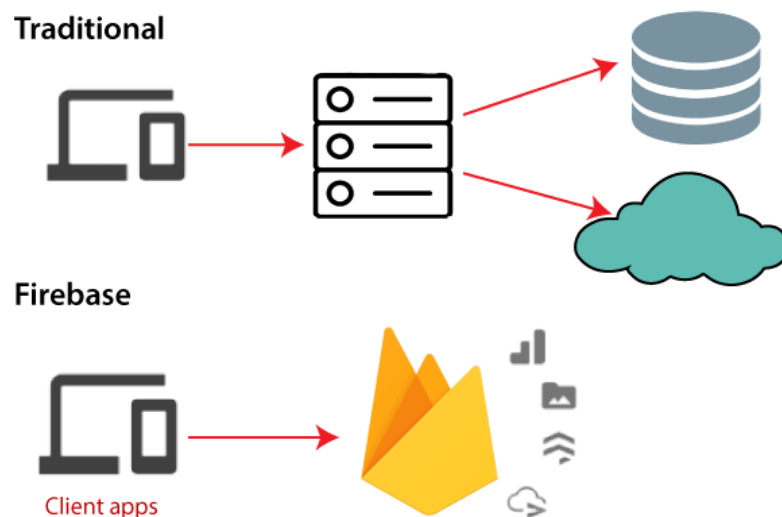


Figure 19. Firebase Database vs Traditional Database. Source: <https://www.javatpoint.com/firebase-introduction>

### 3.2.2 Authentication

Nowadays, web applications are used by many people and most of the cases they need to provide their personal information in order to perform tasks such as online shopping, communication, data storing, etc. Therefore, it is almost compulsory for an application to have an authentication system that can manage the users' accounts and keep their personal information secured. In authentication, the users will have to prove that they own a valid account to access the application by providing the required login credentials. The authentication system of Firebase supports backend services to check the users' identifications. After admins are logged in, they can view users' account basic information and manage their privileges. Also, they can create authentication tokens for users to access their applications. Besides, developers can customize the login methods they want to apply for their web apps such as username, email, password, etc. Or, they can use the FirebaseUI which is a drop-in method including an authentication interface. This FirebaseUI allows users to choose federated login methods from trustful providers such as Google, Facebook, Twitter, etc. By using this drop-in UI, developers can save their time from converting their own methods to the Firebase system.

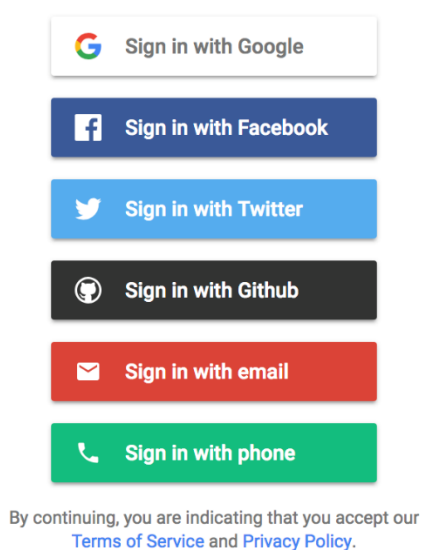


Figure 20. FirebaseUI. Source: <https://github.com/firebase/firebaseui-web>

### 3.2.3 Cloud Firestore

In reality, applications' database can be large and complex. The data structure contains numerous nested objects. Also, data appear under various types such as string, float, binary, etc. And, developers usually have to iterate through the entire database just to retrieve one piece of data. This time-consuming and unnecessary method will cause bad performance to the application and make the programming work more complicated. Therefore, Cloud Firestore was introduced to solve this problem. The Cloud Firestore is similar to the Realtime Database, it lets web applications transfer data to the Firebase cloud storage and synchronize them instantly across users. It also supports offline data storing on local devices. However, the data structure of the Cloud Firestore is quite different. The data are stored in units called "documents". Each document has a unique key and assigned value ( e.g. string, float, binary, JSON, etc.). And, associated documents are located in collections and subcollections. This structure allows nested data up to a hundred levels in depth. Additionally, Cloud Firestore allows developers to fetch data by writing simple shallow queries (e.g. sort, filter, etc.) efficiently. Hence, developers can retrieve the target data directly without fetching the subcollections from the lower level or iterating through all higher collections.



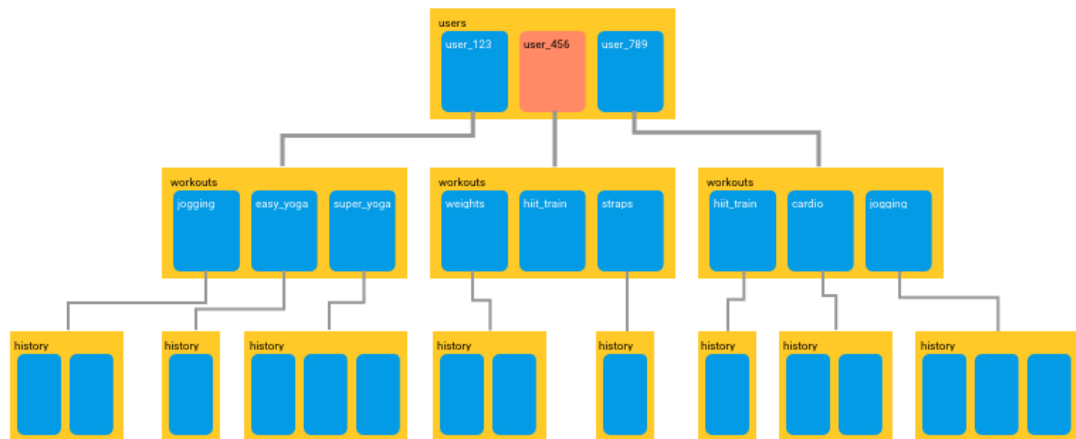


Figure 21. An example of Cloud Firestore structure. Source: <https://firebase.googleblog.com/2017/10/cloud-firestore-for-rtdb-developers.html>

The Cloud Firestore can also be used for applications that have a much higher database scale than the Realtime Database. However, the processing time of queries depends on the queries not the size of the database. This means the executing speed is similar even if the database contains a hundred documents or a hundred million documents.

#### 3.2.4 Cloud Storage

Nowadays, the demand for sharing information is spreading rapidly. Internet users not only want to compose text type data but also want to share media data such as photos, videos, etc. Cloud Storage was produced to help developers create applications that can store these types of data from the user's client. With Firebase Authentication, developers can make sure that application users' data are encrypted and protected by the Google security system. Cloud Storage SDKs have a reliable operation which means when users upload or download media files with a low Internet connection, the process will automatically resume from where it paused. Additionally, Cloud Storage allows developers to build applications that contain a gigantic database scale up to exabytes unit.

### 3.2.5 Hosting

In web application development, developers will need to publish their products to the world at some point. This requires a hosting service that serves the web content (e.g. HTML, CSS, JavaScript, etc.). Therefore, besides database services, Firebase developers added the hosting solution for web applications. Firebase Hosting provides tools and features to deploy and manipulate web apps. It is powerful and easy to integrate. Developers can deploy their apps within a few simple steps in the Firebase CLI command line. Also, they can reverse to the previous version in case of mistakes with only one click. Additionally, web applications are not required to have extra SSL configuration because they are served via HTTPS. This means transferred data are secured and protected by Firebase.

### 3.2.6 Cloud Functions

In web application development, it is required to have backend scripts that perform tasks such as sending notification messages to users' emails, checking authentication, scaling images, managing accounts, etc. Most of the time, developers have to create their own functions and locate them on servers. This approach takes a huge amount of time and might contain errors from developers' mistakes. To help reducing stress in the development process, Firebase offers a feature called Cloud Functions. This feature includes existing built-in functions to handle general backend situations. Developers can write the statements inside supported events (e.g. database events, authentication events, HTTPS requests, analytics events, etc.) to trigger the behaviors. Because the functions are stored in the Firebase cloud, developers do not have to consider about scaling their backend servers. This approach brings significant advantages:

- No backend maintenance required.
- Scalable groundwork backend in the cloud.
- The codebase for back-end scripts is separated.

To summarize, Firebase has all the features that can be used as a back-end service for modern web applications. It includes non-relational real-time database, authenticating methods, back-end functionalities, hosting service, server maintenance, notification messages, etc. And, it is compatible with modern JavaScript frameworks such as React,

Angular, etc. When using Firebase, developers do not need to install services from third-party libraries or concern about back-end scripting on servers. Therefore, after considering the benefits of Firebase and the project requirements, the development team decided to choose Firebase as the back-end service for the Lean 5S project.

## 4 DEVELOPMENT

### 4.1 Project goal

Professors and students of the chemical engineering department of Turku University of Applied Sciences use their laboratory (lab) for chemical experiments and teaching lessons. The labs usually contain numerous chemical containers and devices. During the experiments, there are many issues arise such as broken devices, expired chemical, fire accidents, leak liquid on the floor, rules violation, etc. To keep track of the issues, students are responsible for inspecting the lab condition every day and write reports by filling questions on paper. Later, they will write the results on the board on the wall and draw a graph. Professors can check the board to get information about the situations and decide solutions. In the beginning, the whole process is done manually on paper which is inconvenient and moderate. Therefore, the project 5S was launched to develop a digital solution in terms of a web application. The goal is to replace the complex handcraft system by specific simple steps in the application. To use the application, students will start by log into the system with a group account. Then, they can find the inspected lab by filling the lab number in the search box. After that, they have to answer Yes/No questions and write detail comments about the issues before submitting them. The application will calculate the point base on the answers and the data will be stored in the database. On the other side, professors can log into the application with admin accounts. They can perform tasks such as editing the lab information (e.g. add, remove, add questions, etc.), editing the questions lists (e.g. add, remove, etc.), creating group accounts, etc. Most importantly, they can view the automatically generated graph to check the labs' condition. The graph can be filtered by dates and labs based on the demands of the users. Another purpose of the project is to provide student developers with a real experience environment. By working and studying at the same time, students can quickly learn about project management and teamwork skills. Also, they gained a huge amount of knowledge of new technologies in web application development such as React, Firebase, etc. By completing personal tasks and common tasks, students are experienced with problem-solving and can work independently or together with other members. Additionally, students are more responsible and hardworking to achieve the goals of the project. This helps them prepare the necessary knowledge and technical skills that match the requirements of the jobs market and expand their career path.

## 4.2 Methodologies

During production, a development team might encounter many problems when managing the project and maintaining product quality. And, it is very difficult if each member has their own ways of working. Hence, it is necessary to apply project management methodologies to help the team function with as least problems as possible. A methodology can be described as a system that is used to manage a project including rules, principles, tools, processes. Also, it contains guidelines, documents, activities that developers need to follow to deliver a quality product on time and have an effective working routine. By using methodologies, a project team can share the common voice and keep track of their responsibilities. Most importantly, it helps to reduce the risks and duplications in the development process.

Choosing suitable management methodologies for a project is a crucial matter because it has a significant impact on project success. It is not simple to choose the best methodology that can ensure all of the projects' quality. However, developers can choose the most proper methodologies for their projects' characteristics base on several factors:

- Resources
- Skills
- Project scope
- Timeline
- Risks
- Project scale
- Communication
- Flexibility

After summarizing all these factors, the team can start to do research on methodologies' principles and select the ones that satisfy the criteria of the project. In the Lean 5S, there is not only one methodology is applied but a combination of several ones that are beneficial to the team.

- **Scrum:** the purpose of this methodology is to enhance the communication between project members and keep track of the activities of the team. In the beginning, there will be a sprint planning for the product backlogs, roles, tasks assignment, timeline, etc. Then, the development process will be divided into sprints which occur in two to four weeks. During the sprint, there will be daily

meetings among the team members to report about the task they are doing. Developers have to follow the backlog without any changes. At the end of a sprint, there will be a sprint review where the team discusses with the product owner about the backlogs. Developers will present the achievement and problems arise from the last sprint. The product owner is deciding whether the tasks are done or not. Also, discuss solutions to improve the tasks. Finally, sprint backlogs are specified for the next sprint. The Scrum lifecycle continues.

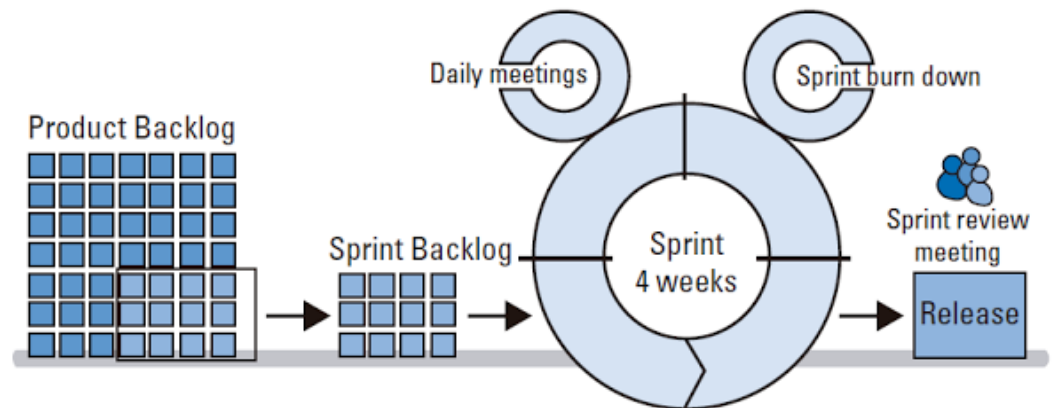
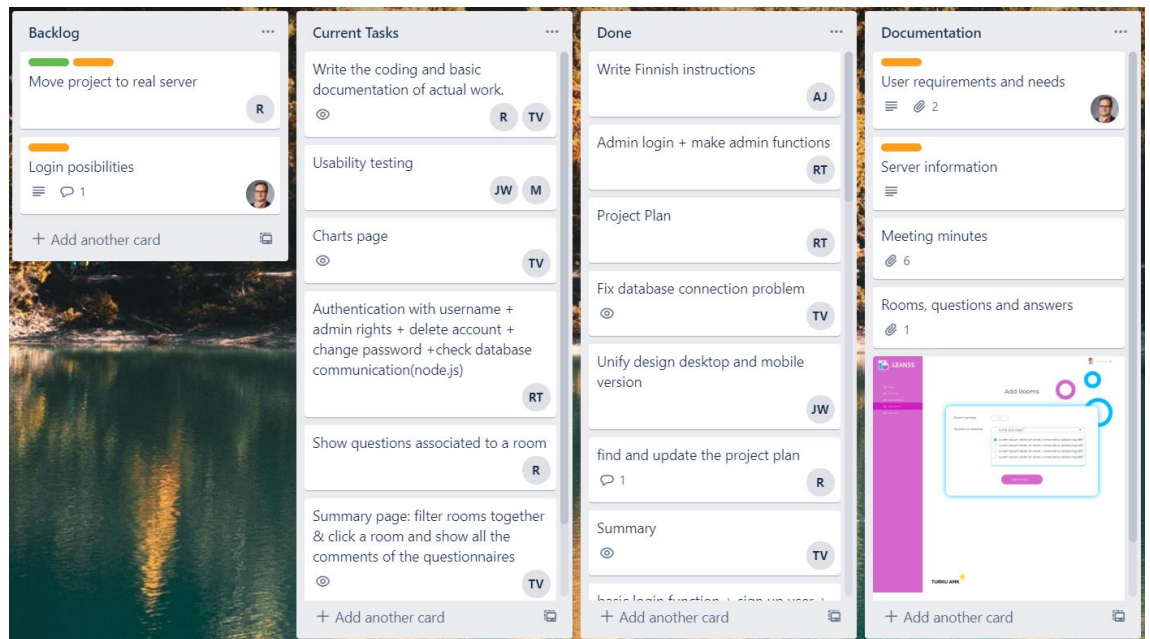


Figure 22. Scrum lifecycle. Source: <https://www.vanharen.net/blog/scrum-in-3-minutes/>

- **Kanban:** to visualize the project tasks and processes, the team uses Trello as a form of Kanban methodology. With this tool, users can create cards and locate them in columns that represent the categories of tasks (e.g. to-do, progressing, done, etc.). In each card, users can perform tasks such as writing a description, adding attachments, assigning members, etc. The cards can be dragged to other columns based on their status. Also, users can set labels base on priority and deadlines for the cards. This tool helps the developers to keep track of their progress and manage the tasks efficiently. Additionally, it is a powerful tool for project managers to apply the workflow of the Scrum methodology.



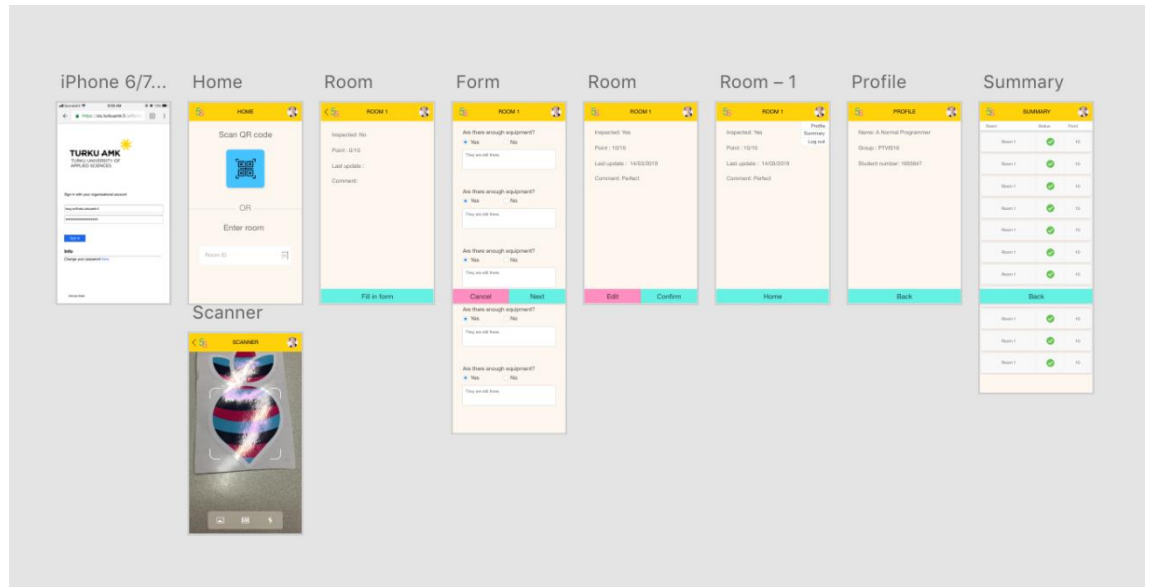
Picture 2. Kanban board of Lean 5S.

### 4.3 Tools & Software

After defining the methodologies of the project. The team begins to choose the tools for each part of the development process including UI & UX design, communication, version control, programming. Choosing a good tool is also a way to help the project proceed properly.

- UI & UX design:** Adobe XD was used to create the mockup interfaces of the application. This tool was built to design websites and applications. It allows users to choose the screen sizes of the most popular devices (e.g. Apple or Android products, laptops, etc...). Users can create layouts that represent the components and screens of the app. These layouts can be connected and seen in the preview mode to demonstrate the workflow of the application. Also, users can create trigger buttons to switch from one layout to another. With this tool, developers can see how the application works easily in a short amount of time. During the development, three members participated in daily discussions to exchange feedback and potential improvements before having the sprint review. In the sprint review, the product owner shares his thoughts on the designs and make new changes if needed. The user experience is also affected by the user interface. Therefore, developers have to modify their code constantly to apply the

changes. Besides, to evaluate the usability of the application, the team conducted user testing with a group of students. In the testing day, the users will have to use the application without instruction from the team. The results show that there is a lack of visual instruction to use the application and the users are confused with the purpose of the application.



Picture 3. The first UI mockup.

- **Communication:** Whatsapp and Outlook are the main communicating channels of the team. Developers will explain and inform each other in case of arising issues via Whatsapp. The project manager discusses with the product owner by email to ask for opinions and report the situation of the team. Also, the product owner arranges sprint reviews by using the Outlook calendar. All of the participants will be reminded about the meetings' time by the alarm features.
- **Version control:** developers usually work on different tasks and create new code or make changes from time to time. To keep track of the modification in the project source code, the team uses the GitLab repository. It provides public and private use for free. Also, the amount of participants in a project is not limited like GitHub. Using a version control method can help prevent conflicts between the changes. Each developer can create their own branch to work on and merge them into the master branch after solving the conflicts. Also, if there is any mistake from the commits, users can easily roll back to the previous version to fix the problems. Developers can see the changes made by other people. Version control is becoming compulsory in any project because of its benefits.



- **Programming:** there are approximately five developers that participate in the project stages. The tasks are divided into backlogs and assigned to each member during the sprint review meetings base on their skills and knowledge. Also, they can choose other tasks if they are eager to learn or help. Sprint review meetings between the team and the product owner are held every three weeks to discuss the work of the team. Developers have time to present their experience in the previous working weeks and explain whether the tasks are manageable or overwhelming for them. The team uses the Visual Studio Code for code editing. It was developed by Microsoft. It is a fast and light-weight editor that works on multiple platforms such as Windows, Mac, Linux. There are several features that make this tool popular among developers.
  1. The IntelliSense feature supports various actions such as code completion, code hints, etc. This feature can be applied for many languages (e.g. JavaScript, HTML, CSS, etc.) by default and other languages with extensions. This feature help developers to edit code quickly and accurately.
  2. The integrated command panel allows developers to run command lines inside the IDE instead of switching to an external terminal.
  3. The built-in Git allows developers to connect their Git repository to control the versions of their work. Developers can perform tasks such as push, pull, commit, etc. This synchronized the commits immediately to the repository cloud. Also, this tool demonstrates the changes between the old and new versions so that developers can effortlessly spot the differences.
  4. Debugging is an important part of web application development where developers can run and fix bugs for their code. Visual Studio Code contains a built-in debug tool to help developers compile and run debug iteration quickly.
  5. The side-by-side layouts create a convenient observation for developers to keep track of multiple files at the same time.

#### 4.4 Architecture

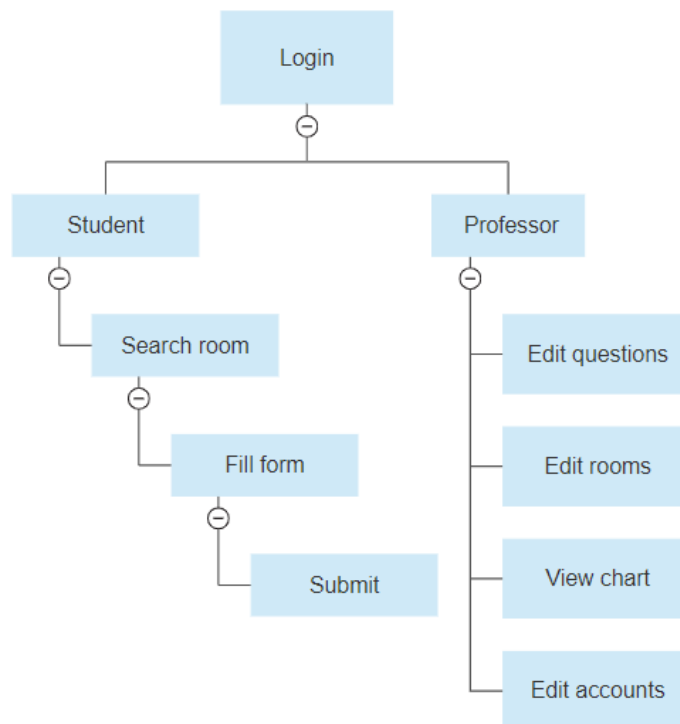


Figure 23. Lean 5S workflow.

The application contains two main roles: Professors and Students. Each role has a specified account. The authentication was designed in a way that both roles can access through the same login interface but will be directed to different panels based on their account extensions.

##### 4.4.1 Student role

When users log in with a student account, they will be directed to the page where they can search for the rooms they need to inspect. The search engine automatically finds the related room code based on the input and shows the list of rooms that contain the input. Each result contains a link to the room page.

After choosing the room, the page will be redirected to the room information page including the room's code, inspection point and checked time.

Then, users can click the "Fill in form" button and transfer to the questions form page that displays all the questions related to the selected room. In this form, users can inspect

the room and answer the Yes/No questions. If the answers are not expected by professors, users have to write some text to describe the issue in the pop-up comment box. This helps professors spot the issue easily and offer solutions. After the form was filled and submitted, the points are calculated and all data including room code, point, lists of answers, checked time will be pushed to the cloud database.

#### 4.4.2 Professor role

When users log in with the admin account, they will be directed to the admin panel that contains buttons. These buttons represent basic actions. They can perform various tasks such as editing questions, editing rooms, editing accounts, checking the data graph, etc.

The "Edit Questions" button contains a link to the questions page. In this page, users can create new questions by adding the text and choose the expected answer (Yes/No). Also, users can delete unwanted questions by clicking the red cross button next to them. The list of existing questions is located at the bottom of the page.

When clicking on the "Edit Labs" button, users can redirect to the lab page where they can create new labs by typing in the lab code and check the boxes to add questions list to the lab. Also, users can delete the existing lab with the delete button next to them.

In the summary and the chart pages, users can check the status of the rooms by using the filter functions. The graph is generated automatically based on the filter results. Also, users can check for details with the summary table by clicking on the entries.

Besides, users can edit accounts (e.g. add, delete, reset passwords, etc.) and set the privileges (Student/Professor) for them. This feature is in progress and operated by other members of the team.

#### 4.5 Implementation

After the workflow was defined, the team started to implement the designed objects. This section demonstrates guidance on initiating the development environment. Also, it will clarify the tasks and elements that were achieved by the author of this thesis during the commission period.

#### 4.5.1 Create React app

When starting a new React project, developers need to make sure that Node.js and npm are installed on their devices. The detail information can be found at this page: <https://nodejs.org>. Then, developers can follow these steps to create a React app:

- To write command lines, a terminal needs to be opened in the workplace folder.
- After that, a react application can be installed and launched by the command lines in the following order:
  1. `npx create-react-app yourappname`
  2. `cd yourappname`
  3. `npm start`

After that, a React application is created and launched on the default browser via localhost. Inside the application folder, developers can find initial folders such as `node_modules`, `public`, `src`, etc. Also, there are several files such as `.gitignore`, `package.json`, `README.md`, etc.

#### 4.5.2 Firebase database

After setting up the React app, developers can continue to integrate the Firebase database by following these steps:

1. First, a Firebase project needs to be created. The detail guidance can be found at this link: <https://firebase.google.com/docs/web/setup>.
2. To use the Firebase library, its package needs to be installed by Node.js using the following command line:
 

```
npm install --save firebase
```
3. The next step is to create a folder named `components` (This folder is used to store all components of the project.) inside of the `src` folder.
4. Then, developers can create a file named `firebase.js` inside the `components` folder to store their Firebase project configuration information. Inside the file, the code is written as Figure 23. Also, the sample information needs to be replaced by the Firebase project configuration.

```
import firebase from 'firebase';

var firebaseConfig = {
  apiKey: "api-key",
  authDomain: "project-id.firebaseio.com",
  databaseURL: "https://project-id.firebaseio.com",
  projectId: "project-id",
  storageBucket: "project-id.appspot.com",
  messagingSenderId: "sender-id",
  appId: "app-id",
  measurementId: "G-measurement-id",
};

firebase.initializeApp(firebaseConfig);
export default firebase;
```

Figure 24. Firebase database configuration sample code.

5. To connect with the database, developers can use the import syntax at the top of their component files:

```
import firebase from 'firebase';
```

#### 4.5.3 Database summary page

Even though the author of this thesis participated in the Lean 5S project from the beginning and performed various tasks, this section will specifically focus on the task that was achieved during the commission period - the summary page. After students finished their inspection and submitted the form. The data will be stored in the Firebase database cloud. However, professors need to examine the data in the most convenient way. Therefore, the requirement is to create a summary page that displays a table of results and a generated visualized graph. Professors can use the filter to view the data by date and lab number. To demonstrate the logic and functionality of the summary page, the following information will provide the audience with an explanation of the core parts of the source code.

## Initiating the component

First, the summary page is treated as a component. Therefore, a file named `Summary.js` was created and located in the `components` folder. The import statements are written at the top of the file to get the required libraries (e.g. React, Firebase, jQuery, etc.).

```
import React, { Component } from 'react';
import firebase from './firebase';
import Navigation from './Navbar';
import CanvasJSReact from './canvasjs.react';
import Table from 'react-bootstrap/Table';
import ToggleAnimation from './ToggleAnimation';
import { withRouter } from 'react-router-dom';
import $ from "jquery";
import 'jquery-ui-bundle';
import 'jquery-ui-bundle/jquery-ui.css';
```

Figure 25. Import statements.

The content of the component will be included inside a class named `Summary`. This class inherited the characteristic of the React Component library.

```
class Summary extends Component {
}
```

The props and state of the component are declared inside a constructor.

```
class Summary extends Component {
  constructor(props) {
    super(props);
    this.state = {
      answers: [],
      rooms: [],
      filterAnswers: [],
      filtered: false,
```

Figure 26. Declaring state and props.

To be used in other places, a component needs to be exported. The export statement is written at the bottom of the file.

```
export default withRouter(Summary);
```

## Fetching data

To fetch the data from Firebase and assign it to the component state, the code needs to be written inside of a React lifecycle function.

```
componentDidMount() {  
  }  
}
```

Before fetching the data from firebase, a variable was declared above the class declaration to store a copy of the answers.

```
var tempAnswer = [];
```

The list of answers is fetched by using a reference variable that links to the answers table the Firebase database. The `on()` event will iterate through the answers and push them to the temporary arrays. Next, the arrays will be sorted in descending direction to get the latest answers first. After that, the state will be updated by the temporary array.

```

const answerRef = firebase.database().ref('answers');
answerRef.on('value', (snapshot) => {
  let answers = snapshot.val();
  let newState = [];
  for (let item in answers) {
    newState.push({
      id: answers[item].room,
      point: answers[item].point,
      time: answers[item].time,
      answer: answers[item].answer
    });
    tempAnswer.push({
      id: answers[item].room,
      point: answers[item].point,
      time: answers[item].time,
      answer: answers[item].answer
    });
  }
  //sort descending by date
  newState.sort(function (a, b) { return a.time - b.time }).reverse();
  tempAnswer.sort(function (a, b) { return a.time - b.time }).reverse();
  this.setState({
    answers: newState
  });
}
)

```

Figure 27. Fetching answers.

Next, the rooms list also needs to be fetched to create a filter with room options. It was processed the same way with fetching answers but with a different reference variable. Also, while iterating through the room table, the `defaultOptions` state property is assigned with the new data. This property is used to create the graph when the page is loaded for the first time.



```

let roomList = [];
let newState1 = [];
let tempDefaultOptions = { ...this.state.defaultOptions };
let tempData = [];
let sortedTempArray = tempAnswer;
let roomsRef = firebase.database().ref('rooms').orderByKey().limitToLast(100);
roomsRef.on('child_added', snapshot => {
  /* Update React state when room is added at Firebase Database */
  let room = { text: snapshot.val().roomId, id: snapshot.key };
  newState1.push(room);
  roomList.push(room.text);
  this.setState({ rooms: newState1 }, function () {
  });
  //Visualized data for default option
  sortedTempArray.sort(function (a, b) { return a.time - b.time }).reverse();
  let pointArray = [];
  for (let j = 0; j < sortedTempArray.length; j++) {
    if (sortedTempArray[j].id === room.text) {
      let timeString = sortedTempArray[j].time.split('-');
      let timeDate = new Date(timeString[2], timeString[1] - 1, timeString[0]);
      pointArray.push({ x: timeDate, y: sortedTempArray[j].point });
    }
  }
  tempData.push({
    yValueFormatString: "#",
    xValueFormatString: "D-M-YYYY",
    name: room.text,
    type: "line",
    showInLegend: true,
    dataPoints: pointArray
  });
});
tempDefaultOptions.data = tempData;
this.setState({ defaultOptions: tempDefaultOptions }, function () {
  console.log(this.state.defaultOptions);
});

```

Figure 28. Fetching rooms and data for the graph.

## HTML elements

After the state was initiated, the HTML elements were created to render the summary page. First, the answers data are displayed in a table. Each row of the table contains the information of the answers including date, point, lab number. In the first page load, the table will display all answers as default. The author used a `map()` function to iterate through the answers state of the component.

```

<Table striped bordered hover className="summary-table">
  <thead>
    <tr>
      <th>Date</th>
      <th>Room</th>
      <th>Point</th>
    </tr>
  </thead>
  <tbody>
    {this.state.filterAnswers.map((item) => {
      return (
        <tr>
          <td className="summary-point">{item.time}</td>
          <td className="summary-room">{item.id}</td>
          <td className="summary-status">{item.point}</td>
        </tr>
      )
    })}
  </tbody>
</Table>

```

Figure 29. Table of answers.

The next element is a graph that visually displays the data. This graph is drawn inside a canvas. The author used the CanvasJS library to perform this graph. The canvas will use the state data to show the data point and link them by the lines. Detail information can be found at this link: <https://canvasjs.com/>.

```
<CanvasJSChart options={this.state.defaultOptions} />
```

After the table and the graph were created, users need a filter where they can change the results base on the date and lab number. This filter is written as an HTML form. With this form, users can choose the answers that belong to the labs they want within a range of dates. The form contains inputs, checkboxes, apply button, clear filter button.

```

<form onSubmit={this.handleFilter.bind(this)} className="filter-container grid-layout">
  <div className="input-filter">
    <div>
      <label htmlFor="from">Start Date</label>
      <input readOnly type="text" id="from" name="from"></input>
    </div>
    <div>
      <label htmlFor="to">End Date</label>
      <input readOnly type="text" id="to" name="to"></input>
    </div>
  </div>
  <div className="room-list-filter">
    {
      this.state.rooms.map((room) => {
        return (
          <div className="list-filter-item"><input type="checkbox" className="room-checkbox" name={room.text} value={
            room.text}></input>{ ' '}{room.text}</div>
        )
      })
    }
  </div>
  <div className="apply-filter">
    <button type="submit">Apply</button>
  </div>
  <div className="clearFilter">
    <button onClick={this.clearFilter.bind(this)}>Clear filter</button>
  </div>
</form>

```

Figure 30. Filter form.

## Event handlers

To apply the filter, a function was created to handle the submit events.

```
handleFilter = (e) => {}
```

In the beginning, variables were declared as references to the DOM elements and temporary arrays.

```

handleFilter = (e) => {
  e.preventDefault();
  this.setState({ filterAnswers: [] }, function () {
  });
  let roomList = [];
  let checkList = document.getElementsByClassName("room-checkbox");
  let from = document.getElementById("from").value.split('-');
  let valueFrom = $.trim($("#from").val());
  let to = document.getElementById("to").value.split('-');
  let valueTo = $.trim($("#to").val());
  let tempArray = [];
  let tempData = [];

```

Figure 31. DOM references.

This function includes if-else statements to filter the answers.

```
//filtering
if (roomList.length !== 0) {
  if (valueFrom.length === 0 && valueTo.length === 0) {
    for (let i = 0; i < tempAnswer.length; i++) {
      for (let j = 0; j < roomList.length; j++) {
        if (roomList[j] === tempAnswer[i].id) {
          tempArray.push(tempAnswer[i]);
          console.log(tempArray);
        }
      }
    }
  } else if (valueFrom.length > 0 && valueTo.length > 0) {
    for (let i = 0; i < tempAnswer.length; i++) {
      let timeString = tempAnswer[i].time.split('-');
      let timeDate = new Date(timeString[2], timeString[1] - 1, timeString[0]);
      for (let j = 0; j < roomList.length; j++) {
        if (tempAnswer[i].id === roomList[j] && timeDate >= fromDate && timeDate <= toDate) {
          tempArray.push(tempAnswer[i]);
        }
      }
    }
  };
} else if (roomList.length === 0) {
  if (from && to) {
    for (let i = 0; i < tempAnswer.length; i++) {
      let timeString = tempAnswer[i].time.split('-');
      let timeDate = new Date(timeString[2], timeString[1] - 1, timeString[0]);
      if (timeDate >= fromDate && timeDate <= toDate) {
        tempArray.push(tempAnswer[i]);
      }
    }
  };
}
}
this.setState({ filterAnswers: tempArray.reverse() }, function () {
});
```

Figure 32. Filter statements.

And, statements to change data for the graph base on the filter results.

```

//Visualized data
let sortedTempArray = tempArray;
sortedTempArray.sort(function (a, b) { return a.time - b.time }).reverse();

for (let i = 0; i < roomList.length; i++) {
  let pointArray = [];

  for (let j = 0; j < sortedTempArray.length; j++) {
    if (sortedTempArray[j].id === roomList[i]) {
      let timeString = sortedTempArray[j].time.split('-');
      let timeDate = new Date(timeString[2], timeString[1] - 1, timeString[0]);
      pointArray.push({ x: timeDate, y: sortedTempArray[j].point });
    }
  }
  tempData.push({
    yValueFormatString: "#",
    xValueFormatString: "D-M-YYYY",
    name: roomList[i],
    type: "line",
    showInLegend: true,
    dataPoints: pointArray
  });
}

```

Figure 33. Filter data for the graph.

After getting the filtered results, the state properties will be updated and the component re-renders.

```

//Set new state

this.setState({ filtered: true }, function () {
});
var tempOptions = { ...this.state.options };
tempOptions.data = tempData;
this.setState({ options: tempOptions }, function () {
  console.log(this.state.options);
});

```

Figure 34. State updating.

The next event is to clear the filter. This function will remove the input texts and change the filtered state back to the default state.

```

clearFilter = (e) => {
  e.preventDefault();
  let checkList = document.getElementsByClassName("room-checkbox");
  for (let i = 0; i < checkList.length; i++) {
    if (checkList[i].checked == true) {
      checkList[i].checked = false;
    }
  }
  document.getElementById("from").value = "";
  document.getElementById("to").value = "";
  this.setState({ filtered: false }, function () {
  });
  let tempOptions = { ...this.state.options };
  tempOptions.data = [];
  this.setState({ options: tempOptions }, function () {
    console.log(this.state.options);
  });
}

```

Figure 35. Clearing the filter.

#### 4.6 Results

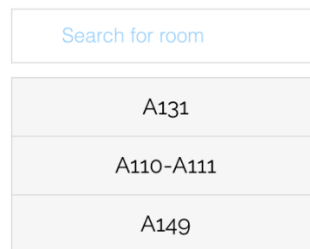
After the first development period, the core features of the application are working precisely the way they were planned. Students and professors can test the application using the localhost on their mobile devices. They can log in with existing accounts that were created by the professors.



Picture 4. Log in interface.

In the student site, users can choose the room they intend to inspect by filling out the room number in the search box and click on the wanted results. Then, they can start filling the question by clicking on the "Fill in the form" button. The questions have two answers and a comment box appears if the answer is not expected by the professors. After filling the form and submit, the data are stored in the Firebase database and update in real-time on other pages. The "Fill in the form" button is disabled until the next day.

ENTER ROOM NUMBER



The image shows a search interface for room numbers. At the top is a text input field with the placeholder text "Search for room". Below this field is a list of three room numbers: "A131", "A110-A111", and "A149". Each room number is displayed in a separate, light gray rectangular box, suggesting a list of search results.

Picture 5. Room search.

### ROOM A131

- Check point: 0 / 10
- Check time:

Picture 6. Room information.

### ROOM A131

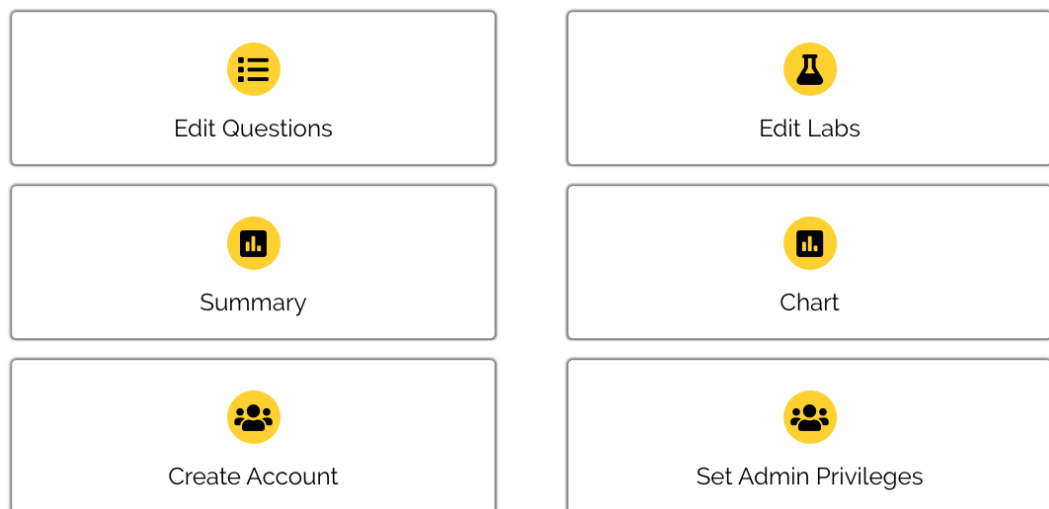
Onko tilan yleisilme ja työpisteet siistit?

- ☐ Kyllä
- ☒ Ei

Comment

Picture 7. Questions form.

In the admin site, professors can edit the question and room lists by adding new entries, removing the existing ones or choosing suitable questions for each room. Especially the summary page which was conducted during the commission period. With this summary page, users can view the results visually with the graph and the table. The table contains detail information (e.g. point, lab number, date, answers, etc.) about each report while the graph shows the differences in the lab condition. The visualized information can be modified by the filter tool. This tool uses the given date and lab number from users to render the table and the graph.



Picture 8. Admin panel .

NEW QUESTION

EXPECTED ANSWER:

☐ Kyllä ☐ Ei

ADD

LIST OF QUESTIONS:

Onko käyttämätön kuumasamaaja sammutettu? - Kyllä

Onko tavarat ja laitteet oikeilla paikoilla? - Kyllä

Picture 9. Edit questions page.



New Room

LIST OF QUESTIONS

☐ Onko käyttämätön kuumasamaaja sammutettu?

☐ Onko tavarat ja laitteet oikeilla paikoilla?

☐ Onko tasoille jäänyt puhtaita tai likaisia astioita?

☐ Onko astianpesukoneessa puhtaita astioita tai onko likaisia astioita täynnä oleva astianpesukone jäänyt käynnistämättä?

☐ Onko astianpesukoneelle ja autoklaaveille pääsy esteetön?

☐ Onko tilojen yleisilme ja työpisteet siistit?

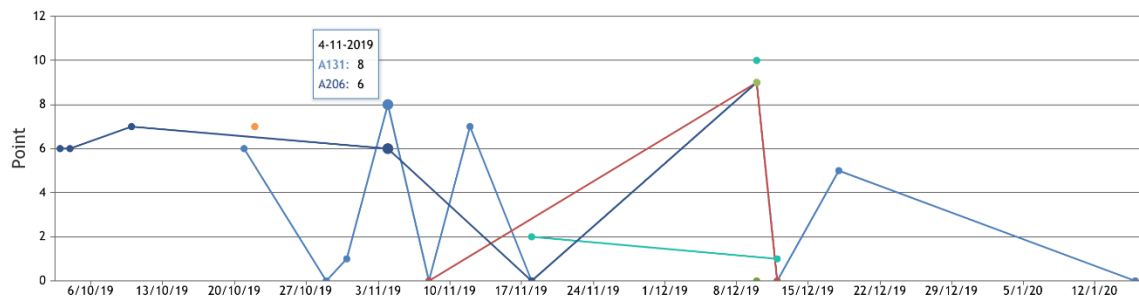
☐ Onko tilan A110 sulaketaululle pääsy esteetön?

ADD

LIST OF ROOMS

✖ A140-A148

Picture 10. Edit labs page.



Picture 11. The visualized graph.

Besides the results of the application, the team in general and the author, in particular, have gained significant experience and skills (e.g. teamwork, programming, project management, designing both UI/UX and application architecture, etc.) after achieving the tasks throughout the development stages. Most importantly, the author of this thesis can use React, Firebase, and JavaScript skillfully to solve complex problems in programming.

From the product owner's point of view, the team has done a beneficial work by processing and carrying out a usable application that satisfies the requirements.

However, the deadlines are exceeded because of the team's unstable schedules and the difference in the skills of each member.

## 5 CONCLUSION

After the first development stage, a beta Lean 5S application was produced to satisfy the main requirements of the product owner. The application lets students inspect the laboratory and report the status by filling the forms. The teachers can view the reports and perform administration tasks. The summary page that includes the inspection information and a graph were delivered successfully during the commission period.

The thesis introduced the technologies including the front-end framework React and the back-end service Google Firebase. This information gives the audience a basic understanding of how these tools are beneficial for developers and modern web applications. It suggests a full-stack option for developers who are having difficulties choosing the suitable tools to build their web applications. Also, it provides a brief history of web application development.

Besides, the practical part of the thesis described how these technologies were applied to create an application that can be used to visualize the data from the laboratory inspection. Even though numerous technologies support more advanced features for developers, based on the author's experience and the project characteristics that were explained in this thesis, React and Firebase is a suitable full-stack combination for the Lean 5S project or projects that have similar scales and features.

The results of the work significantly improved the laboratory inspection process and offered digital solutions to reduce the workforce and save time for students and lecturers.

Even though the results meet the main requirements of the product owner, it can be improved in the future with more advanced features (e.g. pdf generator, QR reader, school accounts, etc.) and more professional programming routines (e.g. code refactor, code consistency, detail documentation, etc.).

## REFERENCES

AltexSoft. (n.d.). The Good and the Bad of ReactJS and React Native. [online] Available at: <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/> [Accessed 16 Jan. 2020].

Campesato, O. (n.d.). Web 2.0 Fundamentals: With AJAX, Development Tools, and Mobile Platforms. 1st ed. David Pallai, p.128.

CanvasJS. (n.d.). Beautiful HTML5 JavaScript Charts | CanvasJS. [online] Available at: <https://canvasjs.com/> [Accessed 4 Feb. 2020].

Daityari, S. (2019). Angular vs React vs Vue: Which Framework to Choose in 2020. [online] CodeinWP. Available at: <https://www.codeinwp.com/blog/angular-vs-vue-vs-react/> [Accessed 22 Jan. 2020].

DeGroat, T. (2019). The History of JavaScript: Everything You Need to Know - Springboard Blog. [online] Springboard Blog. Available at: <https://www.springboard.com/blog/history-of-javascript/> [Accessed 14 Jan. 2020].

Dev.to. (2019). What is React.js and why it's worth to learn? - DEV Community. [online] Available at: <https://dev.to/duomly/what-is-react-js-and-why-it-s-worth-to-learn-1m9o> [Accessed 16 Jan. 2020].

Dev.to. (2020). [COMPARISON] Angular vs Vue.js vs React.js - which one you should choose in 2020? - DEV Community . [online] Available at: <https://dev.to/duomly/angular-vue-js-or-react-js-find-out-which-front-end-framework-you-should-choose-in-2020-3a3b> [Accessed 22 Jan. 2020].

Devsaran. (2016). From History of Web Application Development. [online] Available at: <https://www.devsaran.com/blog/history-web-application-development> [Accessed 13 Jan. 2020].

Emery, C. (2016). A Brief History of Web Development. [online] Techopedia.com. Available at: <https://www.techopedia.com/2/31579/networks/a-brief-history-of-web-development> [Accessed 13 Jan. 2020].

EN | Van Haren Publishing. (n.d.). Scrum - in 3 minutes. [online] Available at: <https://www.vanharen.net/blog/scrum-in-3-minutes/> [Accessed 30 Jan. 2020].

En.wikipedia.org. (n.d.). HTML. [online] Available at: <https://en.wikipedia.org/wiki/HTML> [Accessed 13 Jan. 2020].

Eschweiler, S. (2017). Introduction To Firebase Cloud Functions. [online] Medium. Available at: <https://medium.com/codingthesmartway-com-blog/introduction-to-firebase-cloud-functions-c220613f0ef> [Accessed 27 Jan. 2020].

Esplin, C. (2016). What is Firebase?. [online] Medium. Available at: <https://howtofirebase.com/what-is-firebase-fcb8614ba442> [Accessed 25 Jan. 2020].

Firebase. (n.d.). Firebase. [online] Available at: <https://firebase.google.com/> [Accessed 4 Feb. 2020].

GitHub. (n.d.). firebase/firebaseui-web. [online] Available at: <https://github.com/firebase/firebaseui-web> [Accessed 25 Jan. 2020].

Hackernoon.com. (2017). Introduction to Firebase. [online] Available at: <https://hackernoon.com/introduction-to-firebase-218a23186cd7> [Accessed 25 Jan. 2020].

Hackernoon.com. (2019). Angular vs React vs Vue: Which is the Best Choice for 2019?. [online] Available at: <https://hackernoon.com/angular-vs-react-vs-vue-which-is-the-best-choice-for-2019-16ce0deb3847> [Accessed 23 Jan. 2020].

Hoffmann, J. (2019). What Does AJAX Even Stand For? - The History of the Web. [online] The History of the Web. Available at: <https://thehistoryoftheweb.com/what-does-ajax-even-stand-for/> [Accessed 14 Jan. 2020].

Home.cern. (n.d.). A short history of the Web | CERN. [online] Available at: <https://home.cern/science/computing/birth-web/short-history-web> [Accessed 11 Jan. 2020].

Kerpelman, T. (2017). Cloud Firestore vs the Realtime Database: Which one do I use?. [online] The Firebase Blog. Available at: <https://firebase.googleblog.com/2017/10/cloud-firestore-for-rtdb-developers.html> [Accessed 26 Jan. 2020].

Kumar, A. (2018). React vs. Angular vs. Vue.js: A Complete Comparison Guide - DZone Web Dev. [online] dzone.com. Available at: <https://dzone.com/articles/react-vs-angular-vs-vuejs-a-complete-comparison-gu> [Accessed 22 Jan. 2020].

Line-mode.cern.ch. (2013). Line Mode Browser 2013. [online] Available at: <http://line-mode.cern.ch/> [Accessed 11 Jan. 2020].

Lyonnais, S. (2015). The Evolution of CSS | Creative Cloud blog by Adobe. [online] Adobe Creative Cloud. Available at: <https://blogs.adobe.com/creativecloud/the-evolution-of-css/> [Accessed 13 Jan. 2020].

Mc, A. (2019). What Does Create React App Actually Do?. [online] Medium. Available at: <https://levelup.gitconnected.com/what-does-create-react-app-actually-do-73c899443d61> [Accessed 2 Feb. 2020].

Medium. (2018). React vs Angular vs Vue.js—What to choose in 2019? (updated). [online] Available at: <https://medium.com/@TechMagic/reactjs-vs-angular5-vs-vue-js-what-to-choose-in-2018-b91e028fa91d> [Accessed 22 Jan. 2020].

Medium. 2019. Javascript Trends In 2020. [online] Available at: <<https://codeburst.io/javascript-trends-in-2020-b194bebc5ef8>> [Accessed 7 April 2020].

Morris, S. (n.d.). React JS—What is It? What is Used For? Why Should You Learn It? - Skillcrush. [online] Skillcrush. Available at: <https://skillcrush.com/2019/05/14/what-is-react-js/> [Accessed 15 Jan. 2020].

Node.js. (n.d.). Node.js. [online] Available at: <https://nodejs.org/en/> [Accessed 4 Feb. 2020].

Projects.wojtekmaj.pl. (n.d.). React Lifecycle Methods diagram. [online] Available at: <http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/> [Accessed 19 Jan. 2020].

Raynor, C. (2014). Introducing Firebase Hosting. [online] The Firebase Blog. Available at: <https://firebase.googleblog.com/2014/05/introducing-firebase-hosting.html> [Accessed 27 Jan. 2020].

Reactjs.org. (n.d.). Introducing JSX – React. [online] Available at: <https://reactjs.org/docs/introducing-jsx.html> [Accessed 20 Jan. 2020].

Reactjs.org. (n.d.). React – A JavaScript library for building user interfaces. [online] Available at: <https://reactjs.org/> [Accessed 17 Jan. 2020].

Search Engine Land. (n.d.). What Is SEO / Search Engine Optimization? - Search Engine Land. [online] Available at: <https://www.reactenlightenment.com/react-jsx/5.1.html> [Accessed 20 Jan. 2020].

Sinicki, A. (2017). An introduction to Firebase - the easiest way to build powerful, cloud-enabled Android apps. [online] Android Authority. Available at: <https://www.androidauthority.com/introduction-to-firebase-765262/> [Accessed 25 Jan. 2020].

Smartsheet. (n.d.). How to Choose the Right Project Management Methodology. [online] Available at: <https://www.smartsheet.com/content-center/best-practices/project-management/project-management-guide/how-choose-project-management-methodology> [Accessed 30 Jan. 2020].

Thadani, R. (2018). Advantages And Disadvantages of Application Software You Didn't Know. [online] Techspirited. Available at: <https://techspirited.com/advantages-disadvantages-of-application-software> [Accessed 12 Jan. 2020].

Thakur, N. (2018). What is Visual Studio Code and its advantages - Webner Blogs - eLearning, Salesforce, Web Development & More. [online] Webner Blogs - eLearning, Salesforce, Web Development & More. Available at: <https://blog.webnersolutions.com/visual-studio-code/> [Accessed 31 Jan. 2020].

W3schools.com. (n.d.). JavaScript HTML DOM. [online] Available at: [https://www.w3schools.com/js/js\\_htmlDOM.asp](https://www.w3schools.com/js/js_htmlDOM.asp) [Accessed 15 Jan. 2020].

W3schools.com. (n.d.). React Lifecycle. [online] Available at: [https://www.w3schools.com/REACT/react\\_lifecycle.asp](https://www.w3schools.com/REACT/react_lifecycle.asp) [Accessed 20 Jan. 2020].

W3schools.com. (n.d.). React State. [online] Available at: [https://www.w3schools.com/REACT/react\\_state.asp](https://www.w3schools.com/REACT/react_state.asp) [Accessed 19 Jan. 2020].

www.javatpoint.com. (n.d.). Firebase Introduction - Javatpoint. [online] Available at: <https://www.javatpoint.com/firebase-introduction> [Accessed 25 Jan. 2020].