

Sobhan Niroula

# Comparing a Framework-less Application to a React Application

Metropolia University of Applied Sciences

Bachelor of Engineering

Degree Programme

Thesis

11 September 2018

Author Title	Sobhan Niroula Comparing a Framework-less Application to a React Application
Number of Pages Date	45 pages 11 September 2018
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Professional Major	Software Engineering
Instructors	Janne Salonen, Thesis Supervisor
<p>The purpose of this thesis was to learn about the basics of React framework and compare details about the traditional web development method to the newly developed modern method.</p> <p>There are two similar-looking small-scale projects done for this thesis using two completely different approaches. The projects are also deployed in two different servers whereas both the codes are managed in GitHub.</p> <p>The main reason for choosing React was to learn this new framework in the IT market. And the purpose of the projects done in this thesis was to create an apartment searching website (named 'Movemandu') for the vastly growing student population in the Kathmandu City of Nepal. However, only signup and login portals were developed during this thesis just to show the functionality of the two different approaches of web development, where new user can create accounts and registered users can log into the system securely.</p> <p>Out of the two projects done for this thesis, first project is created using the old traditional method i.e. without using any framework. The technologies used here are HTML, CSS &amp; JavaScript (for front-end), PHP &amp; MYSQL (for back-end) and Metropolia's own server (for deployment). The second project is built using the modern method i.e. using framework – React. So, in this project, only one programming language is used i.e. JavaScript. Here, NPM is used for installing dependencies and AWS is used for database and deploying.</p> <p>The ultimate part of this thesis was to compare these two differently built projects with various factors such as outlook, quality/ reliability, security, speed, popularity, weight, future, and others.</p> <p>Summarizing this paper, the study showed that the React is purely a solid framework to build modern web pages and a strong candidate for future king of web development frameworks. However, it is also not the solution for all the web development problems in the market.</p>	
Keywords	React, Front-end, NodeJS, HTML, CSS, Bootstrap, SQL, PHP, web development, AWS

## Contents

### List of Abbreviations

1	Introduction	1
2	Web Development	3
2.1	Introduction	3
2.2	History	4
2.3	Client-Side & Server-Side Programming	5
2.4	Mobile Apps	6
2.4.1	Web Apps	6
2.4.2	Native Apps	6
2.4.3	Hybrid Apps	7
2.5	Types of web development	7
2.5.1	Front-end web development	8
2.5.2	Back-end web development	9
2.5.3	Full-stack web development	11
2.6	Version Control System	11
2.7	Serverless / Cloud Computing	12
2.8	Single Page Application	13
3	JavaScript	14
3.1	Introduction	14
3.2	History	14
3.3	Vanilla JavaScript & ES6	15
3.4	TypeScript	16
3.5	Frameworks	16
3.6	Difference between software development using framework and without framework	17
3.7	JavaScript frameworks	18
3.7.1	Angular	18
3.7.2	React	19
3.7.3	Vue	19
4	React	21

4.1	JSX	21
4.2	Virtual DOM	22
4.3	Components used in React application development	23
4.3.1	Components	23
4.3.2	Props	23
4.3.3	State	24
4.4	Forms	24
4.5	React Native	25
5	Amazon Web Services	28
5.1	Amazon S3	28
5.2	Amazon DynamoDB	29
5.3	Amazon CloudFront	29
5.4	Amazon Route 53	29
5.5	Amazon Lambda	29
5.6	AWS Amplify	30
5.7	Amazon Cognito	30
6	My Project – Movemandu	32
6.1	Project-I: Without Framework	33
6.1.1	Development Process	34
6.1.2	Characteristics of the project	35
6.1.3	Pros and Cons of the project	35
6.2	Project-II: With Framework	35
6.2.1	Development Process	36
6.2.2	Characteristics of the project	37
6.2.3	Pros and Cons of the project	37
6.3	Comparison of the two projects	38
6.4	Future Implementation	41
7	Conclusion	42
	References	43

## List of Abbreviations

HTML	Hypertext Mark-Up Language
WWW	World Wide Web
W3C	World Wide Web Consortium
JS	JavaScript
AWS	Amazon Web Services
PHP	Hypertext Pre-processor
SQL	Structured Query Language
DOM	Document Object Model
VCS	Version Control System
SPA	Single Page Application
OOP	Object-Oriented Programming
JSX	JavaScript XML
API	Application Programming Interface
ECMA	European Computer Manufacturers Association
NPM	Node Package Manager
UI	User Interface
UX	User Experience

CLI	Command-Line Interface
URL	Uniform Resource Locator
PWA	Progressive Web Application
OS	Operating System
PDF	Portable Document Format
CDN	Content Delivery Network
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
TCP	Transmission Control Protocol
FAQ	Frequently Asked Questions

## 1 Introduction

Computer Programming has come very far till date. When the first time a computer program was built by Lady Augusta Ada Lovelace in 1840 [1], people might have never thought the programming would reach this far. Forget about that very old days, people might haven't thought about it even when JavaScript was first introduced in 1995 [2]. From the initial version of JavaScript 'LiveScript 1995'[2] to the latest version 'ECMAScript 2018'[3], the world of web development has developed by thousand folds. There has been diverse of new technologies in the last few years, and even more are coming soon.

Software have become one of the most essential parts of human life. We encounter with high-tech devices almost everywhere daily and those devices are run by software. Health, engineering, business, transport, etc. are only a few examples of areas where software is used. We now, cannot even imagine our life without software and software-run devices.

Computers and mobile phones are the two top devices to use the highest number of software and applications. Today, almost all the people have mobiles phones in their hand. They go through millions of applications and billions of webpages daily. There are different types of mobile operating systems, for example Android, iOS, Windows, etc. Though having different systems among them, their main purpose is to give users a good interface to use mobile applications, their main work is to process the HTML files while visiting webpages.

The first name that comes into mind while talking about web development is HTML. It is also the first language to be taught during computer programming classes in schools or institutions. All the professional programmers today started their learning process from HTML. Therefore, it is also called the basic language for web programming. A big project can still be developed by html files, but it needs some other supporting languages such as CSS to make web pages look attractive, JavaScript to make the pages interactive, backend languages to work as a bridge from the html file to the database, and database languages to make tables in database. But, this method is said to be the old traditional method to build web pages. It is because this method is less secure, more time consuming and has less support in the web market. It is also because it doesn't have any framework. Modern programmers use frameworks for building webpages because it is fast and

easy to create multiple webpages at a single time. Framework also can support different types of latest security measures to restrict unauthorized access and are very fast and reliable. That is why, there is a huge demand of framework technology at the present.

HTML is now slowly getting less popular and is being replaced. We used to see pure HTML files, but recently those files are being compiled under other languages. One of the biggest examples is JSX [4]. It is a syntax extension to JavaScript. It basically contains html codes inside JavaScript codes, and the output file is a JavaScript file. This means, there is no HTML files as it used to be. We can now develop a full-stack web pages using only one language - JavaScript. There is no need of any other programming languages. One example framework where it is used is React.

There is still an option to use a database of your own and server whichever you like to upload in, even if you are using a framework. But recently, cloud computing has been more popular as it is easy to setup and deploy webpages there. It is one of the most secure platforms to upload files as servers and some even provide database for the application. One of the examples is AWS (Amazon Web Service). We don't need to worry about backend and database security as AWS does it all. This is also called Serverless computing.

The aim of this thesis is to build projects by trying both kind of web development methods mentioned above - traditional frameless method and modern framework method to build two different but similar-looking projects and compare their functionality. The traditional frameless method project is built by using HTML, CSS, JavaScript, PHP & MYSQL, whereas the modern framework method project is built by using React, NPM & AWS. The traditional project is deployed in Metropolia's own server, whereas the modern project is deployed in AWS.

## 2 Web Development

### 2.1 Introduction

Dictionary defines web as a complex system of interconnected elements. Either it is a natural web or internet web, they both define the same. But in case of internet web, it means the interconnection between electronic documents, stored in computers. They are called web pages. The collection of these web pages is called web site. And, the process of developing these web sites is called web development.

Web development is a vast term. Developing a single web page to a big network, everything comes under it. But basically, it is the work involving development of web sites for the internet (also called as World Wide Web). Web development is also called web engineering or web design.

Web development is the planning and creation of website. There are two main parts of web development. They are: Front-end development and Back-end development. Front-end deals with the design, architecture, user interface and layout of the webpage, whereas back-end deals with connection of webpages to servers and databases. Front-end deals with client side of network whereas back-end deals with server side. The combination of both is called full-stack web development and the model is called client-server model.

The main page of a website is the home page. The home page is referred via protocol which is called Hyper-text Transfer Protocol (HTTP). This protocol is completed by visiting the web site by users, and it is possible by entering a link in the user's browser. The link is called Uniform Resource Locator (URL).

There are various programming languages used for web development. Different types of programming languages are used for front-end and back-end. For front-end, the most common is HTML (Hypertext Markup Language) for creating contents in a webpage, along with other languages like CSS, JavaScript, etc. including libraries like jQuery or Sass. For back-end, there are even more languages such as PHP, Perl, Ruby, Python, Java, C, C++, Node.js, etc. Among these, JavaScript now can work for both ends and no other languages are required.

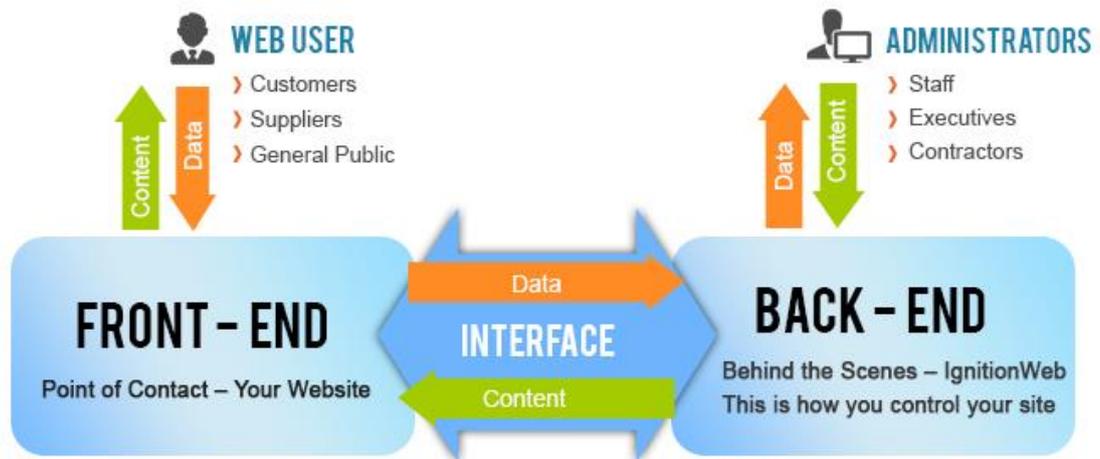


Figure 1. Web Development Chart.

## 2.2 History

The history of web development started from the development of World Wide Web (WWW) by Tim Berners Lee in 1989 with the help of his company CERN. He had built it to interact with researchers in different parts of the world. But it turned into Internet and became popular all over the world. He then termed his forum as World Wide Web Consortium (W3C) [5].

Internet turned from research to household material, and then became people's daily life. The United Nations (UN)'s International Telecommunications Union (ITU) published a report in 2012 which stated that the number of Internet users per 100 people of the population rose from 2 out of 100 in 1997 to 33 out of 100 in 2011. If we consider only of the developed countries, the data would be 67 out of 100 [6].

With the evolution of WWW, many things came into existence. HTML was built in 1990, the world's first browser - Mosaic Browser was first built in 1993. Cookies then came into market with the development of Mosaic Netscape Browser in 1994. CSS (Cascade Style Sheet) also came along with it. PHP was introduced in 1996. These are only a few examples, there were a lot of programming languages and web development tools came along with them.

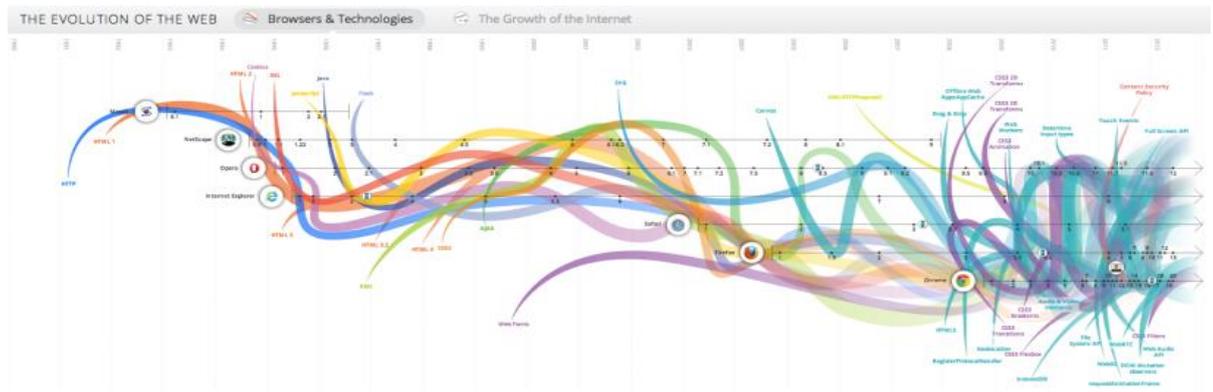


Figure 2. Timeline of the evolution of web technologies.

### 2.3 Client-Side & Server-Side Programming

There are basically two main areas in web programming. The first area is where users get involved in. This area is called client-side. The second area is where all the computation takes place. This area is called server-side.

Client-side basically deals with user interfaces and designs. Its main use is to make interactive webpages, send requests to the server and retrieve data from it. It also provides a remote service to clients such as user registration and login. The common languages used for client-side programming are HTML, CSS, JavaScript.

Server-side deals with everything that is needed to connect a server and retrieve data from it. The programs created for server-side are run on the server itself. It processes the input provided by the client/ user, compile them and connect with database. The popular server-side programming languages are: PHP, Python, Node.js, C, C++, Ruby, etc.

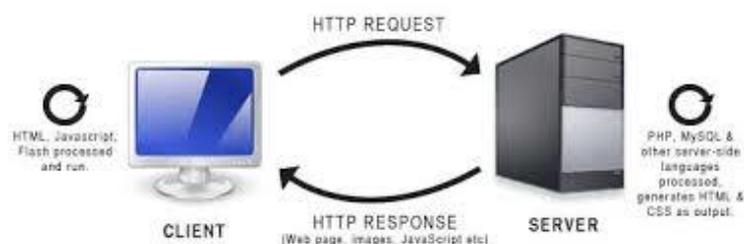


Figure 3. Client-side and server-side concept.

## 2.4 Mobile Apps

With the increase in technology, the number of electronic gadgets has also increased. Among all the electronic gadgets, the most popular is mobile phones. According to Statista - the statistics portal, the number of mobile phone users in the world has reached 4.57 billion in 2018 [7]. With this highly increasing number of mobile phone users, mobile apps have also increased a lot. Different types of mobile apps are available with different functionalities and availability of different category of phones. Mobile apps are categorized into 3 main types. They are:

### 2.4.1 Web Apps

Web apps are the applications which are accessible through a web browser over the internet. Web app is also a web page but is interactive. For example, Wikipedia is a website as it provides information to the users, but Facebook is a web app as it provides an interactive way of communication. These apps don't need to be downloaded and can load in most of the browsers. They don't occupy phone's memory or storage and don't need to be setup by logging into any stores and download from there. Because of these advantages, web apps are being popular nowadays. Mobile apps are built in HTML, CSS and JavaScript and do not need any developing environment. These apps can also be run in PC via browsers. So, they are more versatile. Developing web apps is also easier than native or hybrid apps. However, these apps can't be run offline.

The web apps lack many functionality that native apps can handle, for example: sending push notifications, working offline, loading on home screen, etc. So, the web apps are being improved over the days and adding these functionalities. These improved apps are called Progressive Web Apps (PWA).

### 2.4.2 Native Apps

Native apps are software that run natively on a device and are made particularly to perform on those devices. All the installed apps in our mobile phones are native apps. As they are developed for a particular type of device and particular use, the development of one native app can be different from another. The operating system of devices plays a big role here. Native apps are built to perform faster and smoother and can use device's functions like camera, GPS, notifications, etc. to the optimum level. As a native app is

built for a particular OS or device type, it doesn't work in other types or OS, and occupy memories and storage in those devices. Development of these apps also require more hard work and developing environment, users should login to their respective device stores to download the apps from there. However, these apps can run offline.

### 2.4.3 Hybrid Apps

Hybrid apps are the combination of web apps and native apps. The engine of hybrid apps works as like native apps whereas the outer part such as browsing is like web apps. So basically, the mechanism of hybrid apps is that they need to be installed in the device and users browse through the app. These apps are built so that users can still access device's features like GPS, notifications, camera, etc. while browsing. The biggest disadvantage for these apps is UI or responsiveness. Also, these apps can't be run offline, same as like web apps.



Figure 4. Mechanism of mobile apps.

## 2.5 Types of web development

Web development is a very vast sector. Many segments are involved to make a single website. A website contains everything from the layout to design to interactive to authorization to database connection and to server loading. These all steps are divided mainly into two parts - front-end and back-end.

## 2.5.1 Front-end web development

Everything that deals with the making of front-end of a website is called Front-end web development. It is responsible for building the part of website that users see. From layouts such as fonts, colours, dropdown menus or sliders to architecture of webpage, pictures, interactive components and forms that are made ready to connect with the database are all developed in front-end. So, in a common language, Front-end development is designing the interior of a house while back-end development is the pillar of the house.

The three main languages used in front-end web development are HTML, CSS & JavaScript. Almost all the website is made using these three languages. Along with these, there are also other frameworks used such as Bootstrap (to make webpages more attractive), Foundation, Backbone, AngularJS, EmberJS, ReactJS, etc. and libraries such as jQuery and LESS.

There are two topics which shouldn't be missed out while talking about front-end. They are UI and UX. UI stands for User Interface as UX stands for User Experience. UI refers to graphics design of a website as UX refers to analytical and technical field of the website. UI deals with how the front-end technologies have worked together to make the website work, while UX deals with how the website has worked with the users. So, they are two different terms.

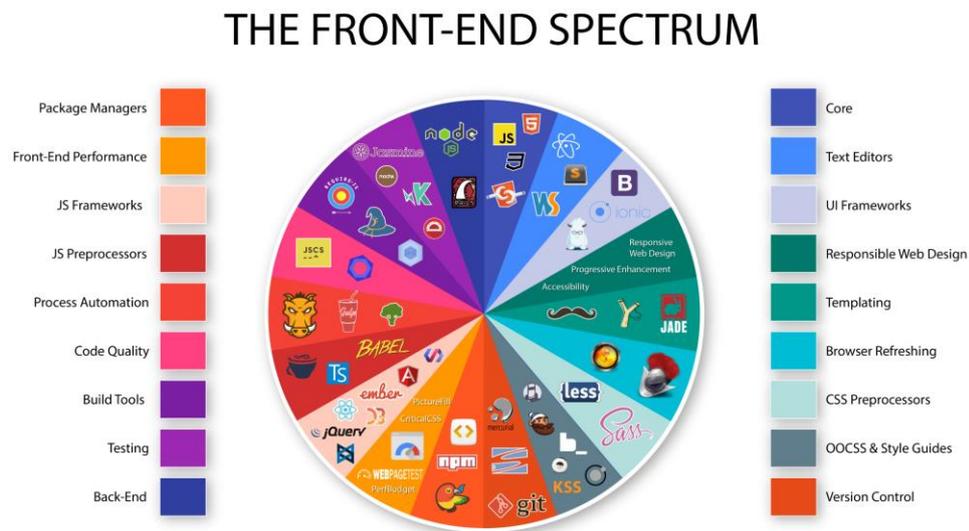


Figure 5. Front-end Spectrum.

## 2.5.2 Back-end web development

Everything that deals with the making of back-end of a website is called Back-end web development. Front-end makes a skeleton structure of website and back-end makes the skeleton able to work. It deals with the segment of a website that users don't see. All the internal workings of the website are developed in back-end. It mainly deals with the server and database connection, making sure the authorization is correct and making the website more secure. It collects data from servers and apps, filters the information and process the user requests. It is also responsible for making APIs (Application Programming Interfaces).

There are various languages used for back-end, but only one at a time. Some of them are: PHP, Python, C, C++, Ruby, etc.

There are other two components used in back-end development. They are:

### 2.5.2.1 Databases

A database is a collection of information that is stored and organized which can be easily accessed, managed and updated. The data entered into the database is stored in rows, columns and tables. These data are indexed so that they can be found easily. These data can be updated, expanded and deleted while a new information is added.

History of database started in 1960s with hierarchical and network database. In 1980s, object-oriented database was introduced [8]. In current situation, there are SQL & NoSQL databases. Recently, there is another new form of database. It is Cloud database.

Relational database is that kind of database where data are stored in the form of rows and columns in tables. SQL (Structured Query Language) is used in it. Whereas, Non-relational or NoSQL database is that kind of database where data are stored in multiple virtual servers. This kind of database are useful for large chunks of unstructured data or large sets of distributed data. It is used by large organizations. MYSQL is an example of relational database whereas MongoDB and DynamoDB are the examples of non-relational database.

A cloud database is a collection of content, either structured or unstructured, that resides on a private, public or hybrid cloud computing infrastructure platform [9]. There is no need of physical database while developing applications. All the files and tables can be uploaded directly to remote database. This helps in reducing the time and is very easy to work on. Developers don't need to worry on maintaining the database as it is maintained from the cloud itself. Some examples of cloud database are Microsoft Azure, AWS & Google Cloud. The disadvantage of this database is that it can't be used offline.

**DynamoDB:** Amazon, Samsung, Snapchat, Netflix, Electronic Arts, New York Times, AdRoll, HTC, Dropcam, Twitch, Shazam, Twilio, Localytics and Clubhouse.

**MongoDB:** Google, UPS, Facebook, Cisco, eBay, BOSH, Adobe, SAP, Forbes, and many more.

Figure 6. Real-world uses of DynamoDB & MongoDB.

#### 2.5.2.2 Servers

A server is a machine that stores HTML files or pages and loads them when users request. It is also a computer program that fulfils requests from client. Here, the client generally refers to the web browser. There are various kinds of servers such as application server, proxy server, mail server, virtual server, file server, etc. The purpose of one type of server is different from another. The use of server depends upon the type of user requests.

Generally, while developing applications, the developers set their server locally to their own computer so that it is easier to work on. While releasing the applications, the server is changed into global server so that the html files can be accessed globally. Those global servers run every time and don't stop even for a single second. Some examples of those servers are: Heroku, AWS S3, etc.

### 2.5.3 Full-stack web development

Dealing with both front-end and back-end development is termed as full-stack web development. Because of fast and highly growing web industry, most of the companies want their employees to know both the front-end and back-end development, so that they can know the basics of the other field they are working on and can build the required application. Generally, front-end developers need to learn additional back-end skills and vice-versa. This is how full-stack development born. Full-stack technology was popularised by Facebook's Engineering Department [10].

Full-stack developers need to learn two different types of languages. But recently, JavaScript has broken this record. It has become the first language to deal with both front-end and back-end, without need to learn any other programming languages. The front-end can be built using JavaScript frameworks like AngularJS and ReactJS where as back-end is built using Node.JS. Database is managed using MongoDB or DynamoDB.

### 2.6 Version Control System

While programming, developers build a new function or feature in the software every single time. They keep updating the software, but it's not obvious that the updated version is the best. It can happen that the previous version could be better than the updated. If the developer wants to roll back the previous version, but also same time wants to keep the current version as well. In this case, Version Control System (VCS) plays a great role. A VCS is a system that allows you to revert selected files or entire project back to the selected time. This system makes files to easily recover if the files are lost or wrongly modified. VCS is the integral part of programming, because it acts as a bank for coders.

There are various types of VCSs, they are: Local VCS, Centralized VCS and Distributed VCS. Local VCS is used only for a person or computer where a programmer updates his codes locally inside the computer. Centralized VCS is for more than one person where many developers can collaborate to each other. In this system, there is a separate server for uploading files. Its disadvantage is that if the server goes down or gets corrupted, all the files get lost. Distributed VCS is the most advanced version control where files are stored in repositories and all the history of the files are stored every time when the files

are cloned. This helps to save the files even if the server gets down or corrupted. Examples of Distributed VCS are: Git, Mercurial, Bazaar, etc.

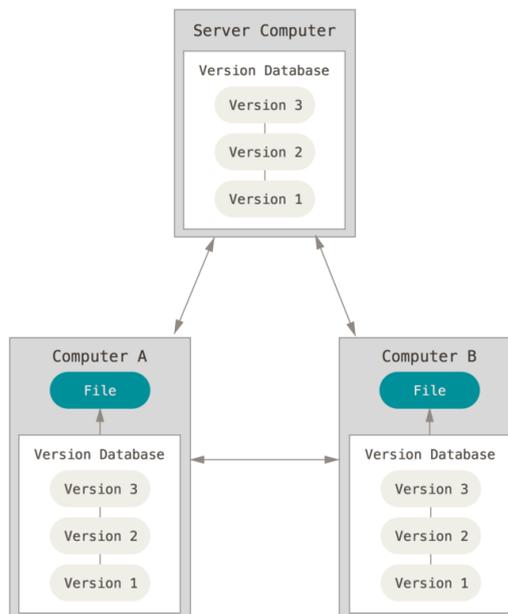


Figure 7. Distributed Version Control System Mechanism.

## 2.7 Serverless / Cloud Computing

Serverless is a new technology where programming is done without the help of servers. In this method, programmers simply write codes using their preferable programming languages. The code is deployed to a cloud provider rather than any server. The codes are triggered using the respective cloud events in the backend to connect to the cloud. Programmers need to worry only about their codes, all other mechanisms are handled by the cloud itself. It has many advantages such as it helps to develop applications easily, in less time and less effort. The main disadvantage of this technology is that the application built from it can't be run offline, and setup also costs money depending upon the use of storage and bandwidth of the cloud. This technology is also termed as BaaS (Backend as a Service) or FaaS (Function as a Service). Some of the popular cloud services are Amazon Web Services (AWS), Microsoft Azure and Google Cloud.

## 2.8 Single Page Application

Single Page Application or SPA refers to the webpage or website which has everything within a single page. When a user enters input in the website, it processes the request and rewrites it within the same page rather than loading entire new pages from the server. This method helps to maintain the same user experience throughout the session. It makes the website behave like a desktop application. The JavaScript frameworks like ReactJS, AngularJS, Ember.js, Vue.js, etc. have already adopted this approach.

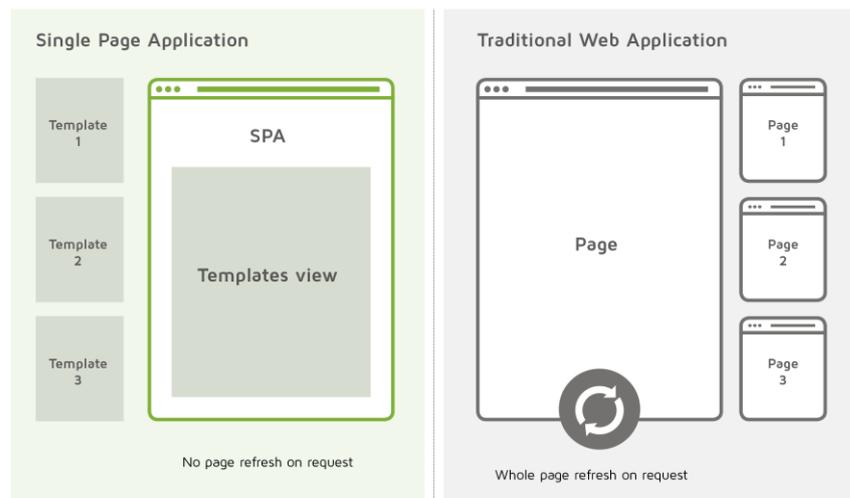


Figure 8. Single Page Application vs Traditional Web Application.

## 3 JavaScript

### 3.1 Introduction

JavaScript or JS is the most favoured programming language at this era of web development. It is the world's most widely spread programming language which is supported by all modern web browsers and devices. It is a high-level language which is easy to learn and implement and used highly for object-oriented programming (OOP). Along with HTML and CSS, it conquers the whole world's front-end web development.

Then main objective of JavaScript is to make webpages interactive. HTML and CSS make webpage look still, static and bore, but JS gives life to the webpage. JS has less memory consumptions and faster execution than other programming languages. Thus, it is being more popular. JS has been so essential in programming now that all the web browsers have a dedicated JS engine to execute JS codes.

Writing JS codes are even easier. It doesn't need any special tools or license to write the codes, just a text editor is needed. It works on all the operating systems and backend languages.

Even though the work of JS was to make webpages interactive, it has recently extended beyond its limitation. It has been able to do more than what we had thought. Emerging of several JS framework in a very short time could be one big example. Initially working only on client-side programming, JS has now been used even in server-side and databases. It has also been used in word processors and PDF software.

### 3.2 History

The world's first browser - Mosaic Browser was built in 1993. Its second version came with the name Mosaic Netscape Browser in 1994 and JavaScript came along with it in 1995. JS was built to make web pages more dynamic. Its name was LiveScript back then which was released in its beta form in Netscape Navigator 2.0 in September 1995. Then name finally changed to JavaScript in December 1995 with the release of Netscape Navigator 2.0 beta 3 [11].

Later Microsoft started its own script called JScript in 1996. JavaScript had very bad reputation at that time for not being similar in all browsers. With the introduction of JScript by Microsoft, the situation became even worse. So, JavaScript was handed over to standardize the language globally to ECMA International (European Computer Manufacturers Association) in 1997. It was named ECMAScript since then, but people usually call it JavaScript. The first version of ECMAScript came in 1997, second in 1998, third in 1999. The fourth version was abandoned, and it took 10 years to finally come a modern JavaScript to support the modern web browsers. ES5 was introduced in 2009 with a lot of improvements, and again it took 6 years to develop another version ES6 or ES2015 with a completely new structure and syntax of JavaScript. The ES9 has already released in 2018, but all the browsers still support ES5. So, to overcome this, a compiler named Babel was made so that the codes written in ES6, ES7, ES8 or ES9 are compiled to ES5 standard and run in the web browsers.

Even though Java and JavaScript sound similar, they are two different programming languages. JS was developed after influenced by programming languages Self and Scheme rather than Java. JS also supports server-side programming. It's first server-side support was developed back in 1996, but completely started supporting from 2009 when Node.js was built.

### 3.3 Vanilla JavaScript & ES6

Vanilla JavaScript is the pure form of JavaScript, or also known as Traditional JavaScript. It is a raw or plain form of JS where no any framework or even a library is used. ES5 is also common to Vanilla JS, but ES6 is different. ES6 is the new form of JS which was introduced in 2015 and has a different syntax. It computes as a function. Example:

```
// Vanilla JS and ES5
function sum (num1, num2) {
    Return num1 + num2;
}

// ES6
(num1, num2) => (num1 + num2);
```

Listing 1. Difference of codes between Vanilla JS and ES6.

### 3.4 TypeScript

As JavaScript is being popular, the developers of other languages like C++, C# and Java are attracted to JS. But, the code structure of JS is completely different from them as those language has totally object-oriented programming. So, TypeScript was generated. TypeScript allows the programmers to code in their own languages and compiles the code into JS file. It allows the syntax like classes, objects, inheritances while coding and converts them into functions and prototypes. The extension of TypeScript file is '.ts' or '.tsx'.

TypeScript is an open-source programming language developed by Microsoft, which is a superset of JS. It is used in developing large applications for both client-side and server-side.

Let's see how it works. For example, if we create a TypeScript file name as 'student.ts' and write the following code:

```
class Student {
    public StudentNumber: string = "";
    constructor() {
        this.StudentNumber = "1";
    }
}

// This creates a file called 'student.js' and converts the above codes into
the following:

var Student = (function () {
    function Student() {
        this.StudentNumber = "";
        this.StudentNumber = "1";
    }
    return Student;
})();
```

Listing 2. Conversion of TypeScript codes into JavaScript codes.

### 3.5 Frameworks

A framework or a software framework is a software environment which provides a platform to build a larger software of a type. It is a complete empty package of programmes, compilers, libraries and tools where developers just need to enter the code to execute a whole software. It is a universal and reusable environment which provides a specific functionality for a specific type of software.

The main aim of the framework is to provide all the necessary environments for developing a specific software so that developers don't need to worry about setting up those environments. They can only focus on their codes and helps saving a lot of time and effort. Frameworks have helped to explore coding into further levels. The disadvantages of using framework is that it takes time to learn a framework, and one framework is different from another. Also, framework adds the size of code in the project.

Angular & React are the examples of JavaScript frameworks whereas Laravel & Symfony are the examples of PHP frameworks. Similarly, examples of Python frameworks are Django & Pyramid.

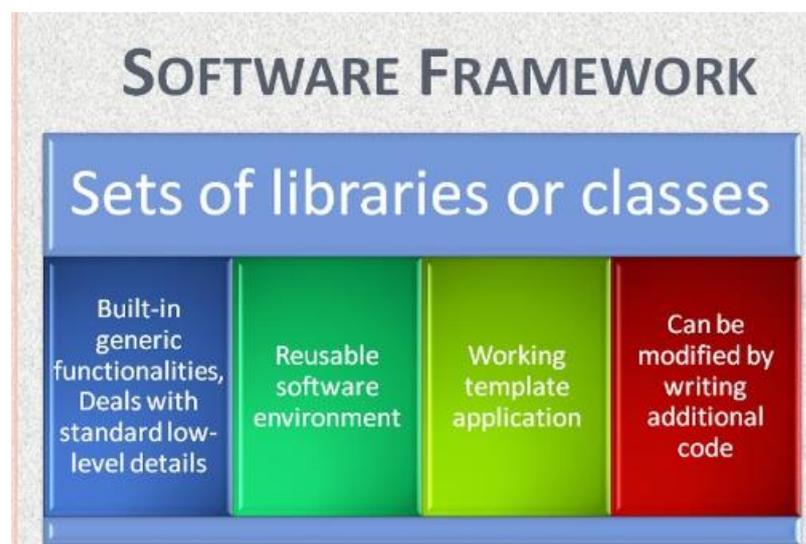


Figure 9. Framework Mechanism.

### 3.6 Difference between software development using framework and without framework

It was just few years back when programming used to be done without using any framework, or in simple word 'from scratch'. But technology changes everyday, so did the programming style. Some hardworking developers built frameworks so that programmers can get basic programming environment to code and don't have to start from scratch every time they build a new software.

Time is precious. So, by the time a programmer setup his environment for coding, he could do a lot of coding if used framework. Framework makes things simpler and offers tools that starting from scratch doesn't do. You don't need to worry about anything else

than coding. You can just focus on your code, everything else is managed by the framework. The software becomes more secured if built from framework, as maximum frameworks today offer security features and plugins as well.

But this doesn't mean that starting from scratch is a waste. It is still possible to build a highly-secured program from scratch, but it requires a lot of time and effort. It is also difficult to compete with other software in the market. But it also has some advantages. It helps to learn each functions and syntaxes, and learning frameworks takes time. So, a beginner can just start right away if he decides to do it from scratch.

### 3.7 JavaScript frameworks

JavaScript frameworks are written in JavaScript and are different from JavaScript libraries. Like other programming languages, JavaScript also has an enormous number of frameworks. Some of them are: Angular, Backbone, Dojo, Ember, Polymer, Preact, React, Svelte, Vue, etc. Among these frameworks, the 3 main JavaScript frameworks are:

#### 3.7.1 Angular

The initial version of Angular was AngularJS, which was the most popular framework before React was introduced. Angular is one of the mostly used frameworks in the world as it is a fully featured framework which provides features like data fetching, development language, state management, build toolchain, etc. The main weapon of Angular is that it uses TypeScript as its development language. So, it is the favourite framework among the developers who come from object-oriented programming languages like C++, C# and Java. It is a full-stack framework.

Angular is mostly popular in enterprise-levels, because large companies have developers working in OOP languages like Java from a long time. So, to make them move to JavaScript framework, Angular is the best option. It also has a strong corporate support from Google [12]. But, it also has some disadvantages. As TypeScript is the main development language in Angular, developers tend to follow only TS and don't go for JS even if JS is the main language for JS frameworks. In addition to it, Angular also has a poor rating in benchmarks.

### 3.7.2 React

React is the most favoured JavaScript language in the world and the history of React is also not that old. It was in 2013 when Jordan Walke, an engineer at Facebook invented React [12]. Unlike of Angular which is a full framework, React only focuses on building user interfaces. The functions like routing, data fetching and state management are handed over to third parties in React, for example React Router for routing and Redux library for state management.

React is very popular with the new generation of programmers as it focuses on functional programming and has great number of supporting libraries. It has a strong job market among developers and companies. It is an open source framework which can be used for cross-platform like web, mobile, desktop and other smart devices. It has a very strong corporate support from Facebook. However, it can be difficult to start-with for beginners due to its large platform.

### 3.7.3 Vue

Vue or generally known as Vue.js is also as new as React, introduced in 2013 by Evan You [12]. It doesn't have any corporate support as Angular has from Google and React has from Facebook. Rather it depends upon individual and corporate donations. It is popular because of its easy learning environment. It is the easiest to learn among the three major JavaScript frameworks. It is also the easiest framework to integrate with other libraries. It has both features of Angular and React and take both approaches.

The advantage of Vue is that it also has good documentation to start with. It also has the best performance among the top three frameworks. However, the job market of Vue is very less regarding to that of Angular or React.

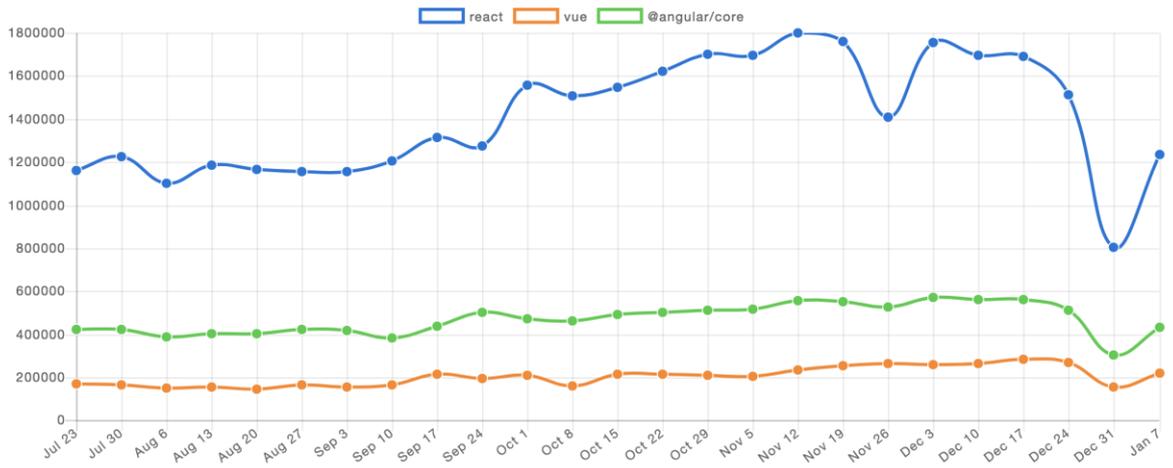


Figure 10. Downloads of main 3 JavaScript frameworks between July 2017 to Jan 2018

## 4 React

React or React.js or ReactJS is the most loved JavaScript library now. The main task of React is to build strong user interfaces. It is used for building single page applications which is developed and maintained by Facebook.

React was invented by Jordan Walke, an engineer at Facebook in 2013. He was impressed by how XHP works, which is an HTML component framework for PHP and tried to make a similar working framework. He implemented his invention in 2011 on Facebook's news feed and again in 2012 on Instagram. But, React was outsourced only in 2013.

React makes easier to create interactive user interfaces. It is also easier to design UI and render it. It is a simple yet powerful framework which can be reused again and again. It has been developed using the latest technology. Combining React with backend technologies like Node.js, PHP or Python, a complete and strong application can be built.

There are many advantages of using React. As JavaScript has been the main programming language which can work in both front-end and back-end technologies, using React could be the best option. It also has a corporate support from Facebook, so the updates are rolled out quite often. It uses JavaScript in optimum level. Unlike of other JS frameworks, it generates HTML from JavaScript. So, developers can only focus on JavaScript. It also has various libraries and plugins available. Fixing bugs is also easier and faster compared to other libraries. The motto of React is 'Learn Once, Write Anywhere' [13].

React is completely different concept in web development. It takes unique approach to solve web development problems. The technology that makes React stronger and unique is JSX and the use of Virtual DOM.

### 4.1 JSX

JSX is the short form of JavaScript XML, which is an extension to the JavaScript language syntax. JSX looks like HTML code but is a JavaScript code. It also has similar tags to HTML like `<h1>` to `<h6>`, `<p>` tags, `<li>` tags which should be nested inside `<div>` tag, and these all HTML-looking tags are then nested inside JavaScript compo-

ment. It also includes the Render function which will be rendered in DOM. It is not necessary to write React in JSX, it can also be written in pure JavaScript. But, writing in JSX provides much flexibility in performance, error handling and deployment.

JSX is closer to JavaScript than to HTML. So, JSX uses 'camelCase' structure inside of HTML attributes for naming. For example, `class` attribute in HTML becomes `className` in JSX. Similarly, an empty tag should be closed with `'/>'` attribute. JSX also prevents injection attacks. All the inputs given to JSX are first converted into strings before rendering. So, it is safe to embed user input in it. The codes written in JSX are finally compiled using Babel compiler. An example of code written in JSX is given below:

```
render() {
  return (
    <div className="container">
      <h1> Hello World! </h1>
      
      <p> Some text </p>
    </div>
  );
}
```

Listing 3. Sample JSX code.

## 4.2 Virtual DOM

Virtual DOM or VDOM is a concept in programming where a virtual representation of UI is stored in memory. The stored virtual UI is then synced with the real DOM. This process is called reconciliation and is possible by a React library called ReactDOM. In simple words, it changes only to that part of UI which we want to change, rather than the whole DOM.

The normal DOM, known as Document Object Model, has a bad reputation of being slower than other JavaScript operations. While working on JS framework, every update makes the whole DOM to reload which makes the development process very slow. To overcome this problem, React came with the concept of Virtual DOM. In Virtual DOM, only a particular object in the DOM is updated rather than updating the whole DOM. This helps in increasing the performance of the whole application. Virtual DOM is much powerful and efficient than the normal DOM.

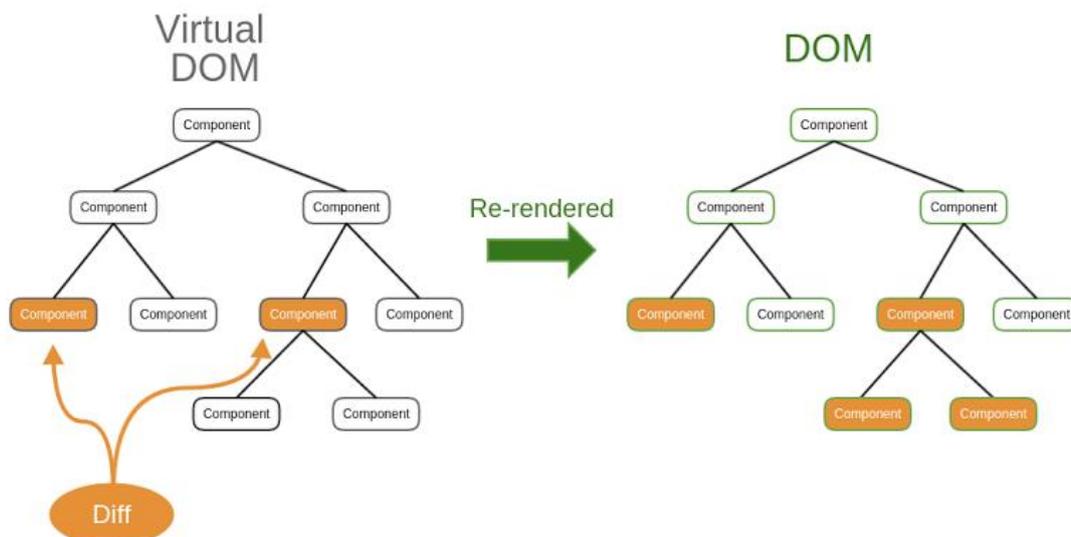


Figure 11. Mechanism of Virtual DOM.

### 4.3 Components used in React application development

React has various components which makes it unique from other JavaScript frameworks. Some of them are:

#### 4.3.1 Components

Building application in React is like building in JavaScript. React components in React are alike to functions in JavaScript. UI can be split into independent and reusable pieces using components. They accept 'props' which are the arbitrary inputs. They then return React elements which determine what should be displayed on the screen.

#### 4.3.2 Props

Props stand for properties in React. They act as a bridge between parent and child components. React passes props as JSX attributes when it sees an element representing a user-defined function. Props are created so that the parameters can be customized, but themselves are fixed throughout the process. For example, while writing the image source code in React, the component 'source' is a prop which control what image to show. Using props lets us to make a single component which can be used in many places in the app using 'this.prop' in the 'render' function.

### 4.3.3 State

State is the React element which is used to change data in React. There are only two types of data which control a component. They are `props` and `state.props`. These data are set by the parent element and are fixed throughout the component lifetime. But, for the data which should be changed, state is used instead of them. State is initialized in the constructor and called `setState` whenever the data should be changed. State makes React components more interactive and dynamic. It is mostly used in user input so that React displays the exact information that users have entered.

```
import React from 'react';

class MainApp extends React.Component {
  constructor(props) {
    super(props);

    this.State = {
      head: "This is the first line",
      "data": "This is the second line"
    }
  }

  render() {
    return (
      <div>
        <h1> {this.state.head} </h1>
        <h2> {this.state.data} </h2>
      </div>
    );
  }
}

export default MainApp;
```

Listing 4. Example of Component, Prop and State use in React

## 4.4 Forms

Creating forms in React is a bit easy. But, the forms elements here work a little bit differently than that of HTML. It is because the form elements in React keep some internal state. React handles the user submission by a different technique called 'controlled components' which also controls what happens in the form on user input. An example of form in React is given below:

```
class InputForm extends React.Component {
  constructor(props) {
```

```

    super(props);
    this.state = {value: ''};

    this.handleChange = this.handleChange.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  handleChange(event) {
    this.setState({value: event.target.value});
  }

  handleSubmit(event) {
    alert(`You have submitted an input!!: ` + this.state.value);
    event.preventDefault();
  }

  render(){
    return (
      <form onSubmit={this.handleSubmit}>
        <label>
          Name:
          <input type="text" value={this.state.value}
onChange={this.handleChange} />
        </label>
        <input type="submit" value="Submit" />
      </form>
    );
  }
}

```

Listing 5. Example of a form in React

## 4.5 React Native

The mobile version of ReactJS is the React Native. It is mainly used for creating native mobile applications. It is like React but has a different building environment with different building package. For example, while setting up the environment in CLI, React app is created by using `create-react-app` while React Native is created by using `create-react-native-app`. For this, you must install React Native by using `npm install -g create-react-native-app`.

React Native was introduced first in 2015 by the greatest social media company Facebook. Facebook is the manager for both the ReactJS and React Native.

React Native builds a real mobile app, unlike ReactJS which can create a web app and a desktop web app. So, the inner architecture is quite different in React Native. For running the app in Android OS platform, React Native should build the app which should run in Java environment whereas in Objective-C for iOS platform. The outer architecture of

React Native is however same as ReactJS. Even though it has two different architecture in the same app, it can bind the both because of a connector. The connector compiles the React codes into Java or Objective-C to run in the respective OS. It provides React with an interface into the respective OS's native UI elements. For example, the `<View>` tag in React Native will convert into Android-specific `TextView`.

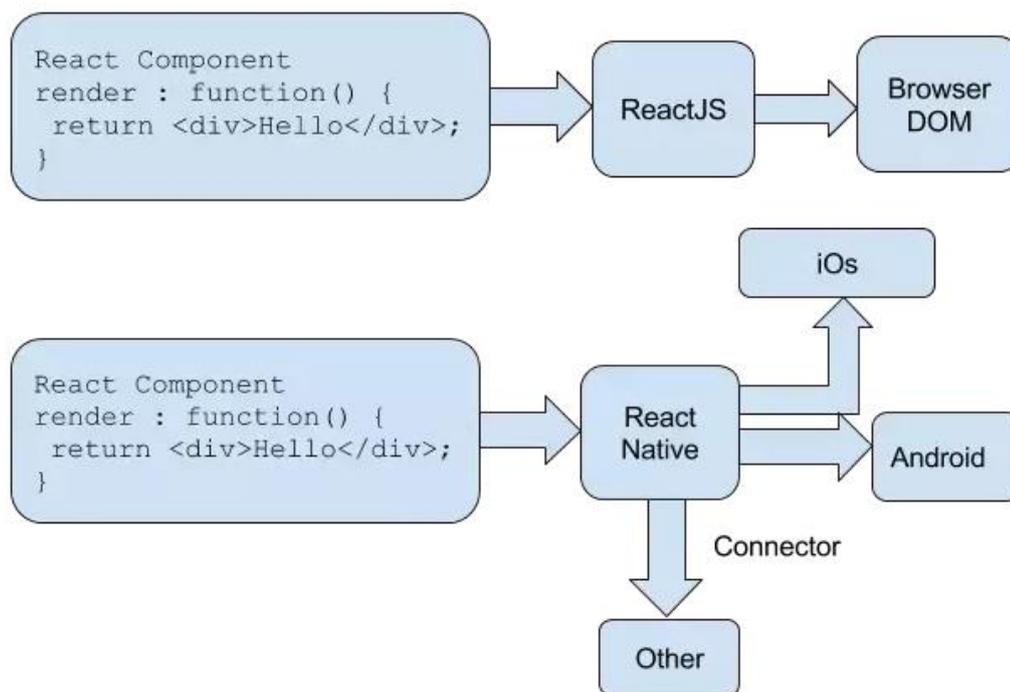


Figure 12. Difference of architectures in ReactJS and React Native.

React Native currently supports both Android and iOS. It will also support other mobile OS if someone builds the respective connector in the future. The advantage of using React Native is that you don't have to use Java or any other programming languages to build native apps, just JavaScript is enough. You also don't have to worry about any development environment for those operating systems. This also allows you to create an app in React Native and deploy in two different mobile platforms. It has made the life so much easier for the developers.

However, the React Native app is not a 100% native app. It lacks some features that a pure native app has. For example, a React Native app has difficulty in finding if the network is from mobile data or Wi-Fi. The account made on these apps are stored in the apps themselves, rather than on the mobile system.

There are many React Native apps in the market, and the number is increasing rapidly. Some of the big examples are: Facebook, Airbnb, Skype (new Android version) [14], Instagram, Tesla, Walmart, SoundCloud Pulse, Uber Eats, etc.

## 5 Amazon Web Services

Amazon Web Services or AWS is a cloud computing services provided by Amazon Inc. It is the subsidiary of Amazon.com which provides various cloud computing services to individuals, companies and governments. All the services are on the paid subscription basis and are internet-based or virtual. These services are run on servers which are placed in several cities throughout the world. The services are highly secured and easy to operate.

The history of AWS started back in 2002 where few tools and services were developed for the company's internals [15]. For public, it was launched in 2004 but still lacked many services. It then relaunched in 2006 with Amazon S3 cloud storage and some other cloud computing services. In 2010, all the retail sites of Amazon.com were moved to the AWS cloud. In 2016, on the tenth anniversary of cloud computing, AWS started its support training and skills standardization to public so that more and more developers would be interested in it.

As per 2017, AWS has 90 cloud computing services ranging from computing, storage, networking, analytics, database to deployment, management, developer tools, and many more [15]. There are several services that AWS is providing. Among them, the services that are used in this thesis are as follows:

### 5.1 Amazon S3

The full form of Amazon S3 is Amazon Simple Storage Service. It is a web storage for the internet for the developers. The interface of S3 is very simple that anyone with a little amount of knowledge can also store their data and retrieve from it at any time and anywhere. S3 is a very fast and reliable storage which runs globally. It is the place to keep the files after deploying on the server. The files are stored in the containers named as 'buckets'. It can also handle any amount of data. AWS claims it to be 99.99% durable [16].

## 5.2 Amazon DynamoDB

Amazon DynamoDB is the NoSQL database service provided by AWS that supports key-values and document data structures. It is like the normal DynamoDB but has additional features that AWS has provided. It is fully secured, and no unauthorized access can be made easily into it. It was announced in 2012 and is highly durable. It includes Auto Scaling feature which will scale the database automatically if enabled. It is built on the principles of Amazon Dynamo storage technology. The price of using this database is based on the data input inside the database.

## 5.3 Amazon CloudFront

Amazon CloudFront is a CDN. CDN means Content Delivery Network, which can be understood as a system of geographically distributed servers located at different regions of the world so that users can access to the internet as quickly as possible. When a user requests a page to load, it will be loaded from the nearest server in that region. When a file is uploaded, many copies of it are made and distributed to all the servers throughout the world so that anyone can access to the file easily from any part of the world. CDN provides best speed and performance, via the best route. Amazon CloudFront is currently located in 55 availability zones within 18 geographical regions. It is planning to add 12 more zones in 4 more regions soon.

## 5.4 Amazon Route 53

Amazon Route 53 is the Domain Name System (DNS) provided by AWS. It is very scalable and highly available. The '53' refers to the port 53 of TCP. It was introduced by Amazon in 2010. It is very reliable and cost effective service which is fully supportive to IPv6.

## 5.5 Amazon Lambda

Amazon Lambda is a serverless computing platform which runs codes in response of events. It was introduced by Amazon in 2014. It lets you run the codes without managing any server. You also don't need to worry about backend if there is Lambda. It is made

for serverless and small applications so that the developers don't have to worry about the servers and their maintenance. They just need to upload their code and Lambda takes care of everything else. Those codes can be uploaded to AWS or can also be called directly from other web or mobile apps.

## 5.6 AWS Amplify

AWS Amplify is an open source library for interacting with cloud services which uses JavaScript applications. It provides effortless interaction of JavaScript codes to the cloud services. It is installed through CLI with the line `npm install --save aws amplify` and imported into React by using `import Amplify from 'aws-amplify';`.

It has few support modules so that developers can implement new things in their applications using this library. Those modules are:

- i) Auth: provides AWS credentials to sign in using Amazon Cognito
- ii) API: interacts with RESTful APIs in a secure way
- iii) Analytics: tracks authenticated and unauthenticated users with a single line of code
- iv) Caching: provides interface across web apps and React Native to cache data
- v) Storage: provides simplified commands for storing data and files into public or private buckets in Amazon S3
- vi) i18n and Logging: provides debugging and logging capabilities, along with internationalization and localization

## 5.7 Amazon Cognito

AWS Cognito provides user signup, sign in and access control for web and mobile apps. It makes developers easy to develop these features in their apps. It scales millions of users securely and has the feature to provide social identity such as Amazon, Google or

Facebook. Developers just need to create signup and login form and rest are done by Cognito with the help of Lambda function. An example of it is given below (Source: AWS [17]):

```
// Add 'aws-amplify' library into your application

// Configure Auth category with your Amazon Cognito credentials
Amplify.configure({
  Auth: {
    identityPoolId: 'XX-XXXX-X:XXXXXXXXX-XXXX', // Amazon Cognito Identity
    Pool ID
    region: 'XX-XXXX-X', // Amazon Cognito Region
  }
});

// Call Auth.signIn with user credentials
Auth.signIn(username, password)
  .then(user => console.log(user))
  .catch(err => console.log(err));
```

Listing 6. Example of AWS Cognito usage

These above all the features can be accessed with a single login in Amazon Web Services.

## 6 My Project – Movemandu

Movemandu is the website to search apartments for highly growing student population of Kathmandu City in Nepal. This is my dream project which can give services to around 150,000 to 200,000 people if launched in a big scale. Finding rooms and apartments has been a nightmare in the Kathmandu city. People are obliged to search for months to find apartments of their choice. Even if they find, the apartments are of very high rent. There are many dealers who take unnecessary commissions for searching even a single room. Even apartment owners are worried about the type of tenants. So, to minimize these problems, I have come up with this project which can facilitate all these people in a very clean, easy and convenient way.

The website will have a good, neat and clean UI so that it can attract users from the first use. The UI will have good colours so that users won't be distracted and irritated. It will be fully responsive so that any size device can have the same great experience using it. It will be made available in web format at first, so the users from all the OS and platforms can use the Movemandu app from anywhere using any web browser. After the complete implementation in web, the app will be launched in Android, and then at last to iOS. The app will be a Single Page Application where users can find all their required services under the same page, without having to navigate to different page in a single session. The top of the homepage will have a carousel with few beautiful pictures of Kathmandu city. Above the carousel, there will be a navigation bar to navigate to the required fields, such as home, services, login, signup, search, about, etc. Below the carousel, there will be some announcements from our system, such as any new service added, any apartments added, or any other news. Below the announcements section, there will be the apartment search section where users can search apartments of their choice. They will have many options to choose from, such as location, rent range, number and types of apartment, additional facilities, etc. Below the search section, there will be information section to new and old users where they can find FAQs, information before, after and during the tenancy period, laws regarding tenancy, etc. Below the information section, there will be contact section where users can contact us with their questions. Below it, there will be the footer of the application with Site Map where links of all section of the app will be provided.

Not only the tenant, but also the apartment owners and real-estates will be benefited through this app. They can also sell their apartments through this app. There won't be any ad running in the app. The app will earn money from very small service charge from

the apartment owners after they get their tenants. Also, some apartments will be featured above others in the list with more charge. All types of members can login with their single login and their types will be determined by the system automatically. There will be two types of signup or login system. The first is filling the signup form, and the second one is through social media login such as Facebook and Google. The authorization will be very secure, no unauthorized access will be granted. Similarly, users can only search apartments after login, to prevent any security issues. Without login, they will only see the carousel and some basic information of the app.

Among the plans of the project written above, the features that are built for the thesis are a welcome page with navigation bar, sign up, login form, authorization and redirection to home page after login. These featured are built in two different projects in two different ways. The first way is the old traditional way i.e. without using any framework. The second project is built with the latest approach in web development i.e. serverless application using a framework and cloud computing. The background image and logo are both created by me. The links of the projects and GitHub files are given below:

i) Project-I: <http://users.metropolia.fi/~sobhann/movemandu/>

GitHub: <https://github.com/sobhanniroula/ThesisProjectMain>

ii) Project-II: <http://movemandu-deploy.s3-website.eu-central-1.amazonaws.com/>

GitHub: <https://github.com/sobhanniroula/ThesisProjectReact>

## 6.1 Project-I: Without Framework

Project-I is built in traditional approach in web development. There is no any frameworks used. Everything is made from scratch. For front-end, HTML5, CSS3 and JavaScript are used whereas for back-end, PHP and MYSQL are used. The PHP used here is also a conventional one, not an OOP PHP. The version of PHP however is the latest one i.e. PHP7. JavaScript in this project is used only in 'Services' page for the Parallax effect, using jQuery. The project is then deployed in Metropolia's own server.

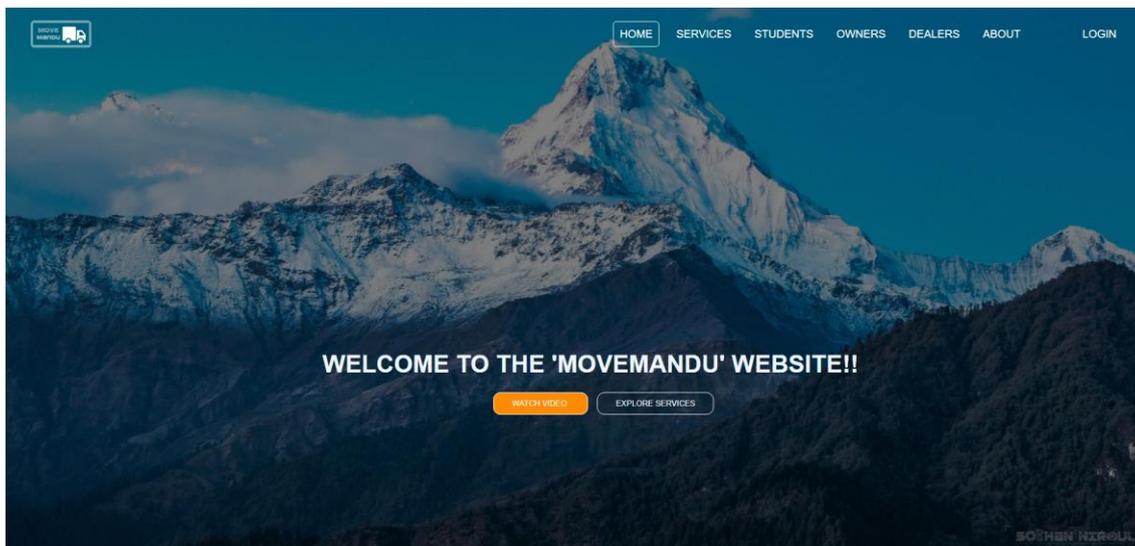


Figure 13. Home Page of Project-I.

### 6.1.1 Development Process

Front-end of this project was built first and then connected to the database using back-end. First, the logo of Movemandu was made using Adobe Photoshop, and the background image was made ready. Both the images were kept inside the folder 'img'. Then, the home page was created, with the navigation bar and its elements. CSS was made with darkening the background image, to give a better look to the front elements. Then, all the pages were made with basic codes for the empty page including the navigation bar. Then parallax effect was made using jQuery in 'services.html' page. The welcome message with two buttons were made in the homepage. Then the signup page was made where basic information of a new user is asked, such as first name, last name, username, email and password. After the signup page, the login page was made where user is asked to enter username or email and password for logging in.

After the front-end was completed, back-end was started to build. At first, all the '.html' files were converted to '.php' files. Xampp was installed on the computer, Apache and MYSQL servers were turned on, and the forms were connected to the MYSQL database. When forms were connected successfully to the database, other features such as user authentication and session were added. Change of Login button to Log out button was also implemented, and redirection to the home page after login was also implemented. Finally, session memory was created so that the user session doesn't change while refreshing the page. SQL injection prevention was also added by using the

`mysqli_real_escape_string` function. Finally, all the files were uploaded to my 'public\_html' folder in the Metropolia's server.

### 6.1.2 Characteristics of the project

There are many characteristics in this project. First, this project is made completely from scratch, no any framework or library is used. There is also no bootstrap used, so only CSS is used to its optimum level to provide a nice-looking UI. It is also responsive and protective from any sql injection. There is also a feature to know what is wrong if the user fails to sign up the account. It can be known from the address bar in the browser if the sign up has failed due to any empty field or wrong input. There is also filter for password and the password entered by the user is completely hashed before storing in the database, so that even admin won't be able to see the password.

### 6.1.3 Pros and Cons of the project

As the project is built completely from scratch, I learnt many things from it. Starting the project and developing it, as well as implementing was also easy for me even if I am a beginner programmer. I didn't have to spend time learning any framework here.

Along with the pros, there are also many cons in this project. As I haven't used any framework here, I lack many modern features and libraries. There will still be many security issues even though I have managed to protect it from sql injections. The CSS is also not up to that high mark, and the pages still lack full responsiveness. I have also used the navigation bar repeatedly in all the pages. Connection to database is also not fully secured.

## 6.2 Project-II: With Framework

Project-II is built in modern way using the latest trend in web development i.e. serverless application using a framework. React is used as the framework for the project along with bootstrap for react while AWS Lambda and AWS Cognito are used for the backend process. AWS DynamoDB is the database and AWS S3 is the deployment platform.

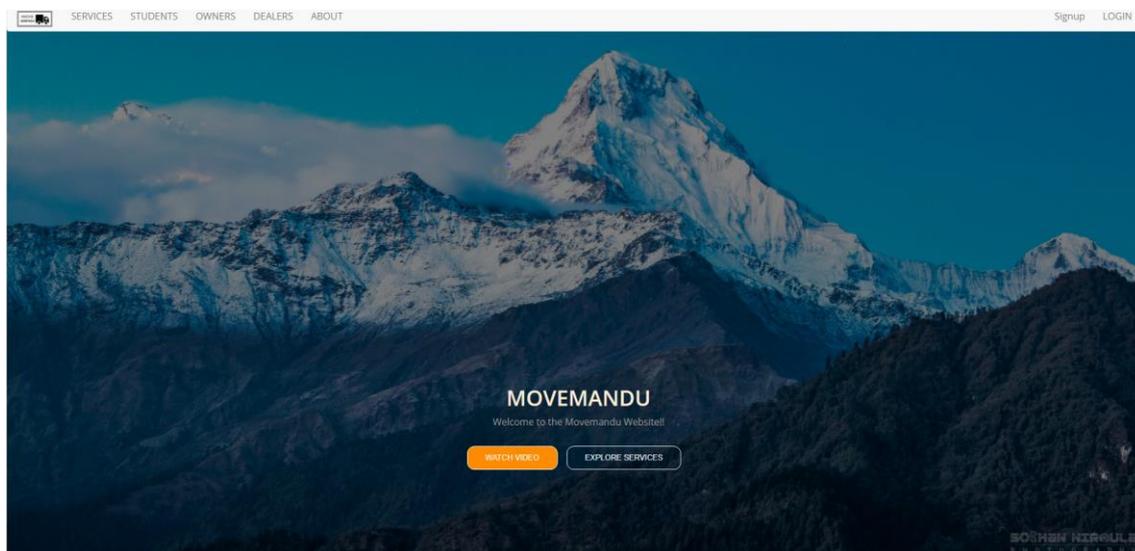


Figure 14. Home Page of Project-II.

### 6.2.1 Development Process

The development process of this project was a bit longer than the previous project. At the beginning, I downloaded and installed the latest stable version of Node Package Manager (NPM). Then, I created a boilerplate package for react inside the project folder using this code in CLI: `npm create-react-app movemandu`. Then going inside the project folder, I started the npm server with the line `npm start`. With the start of the server, the home page automatically got opened. After that, I started to edit the page. I added favicons using my logo and added custom fonts. I then added bootstrap to my project to add cool-looking feature in the UI, using the line: `npm install react-bootstrap --save`. React Router was installed to handle routes for various pages, by using `npm install react-router-dom --save`. This installed the NPM package and added the dependency to the 'package.json' file.

After these all environment setups, I started to code the front-end. I created containers and pages and route the pages using the React Router. I added the links in the navbar and created `Error 404: Page Not Found` handle. Then, I created all the elements necessary for the front-end, like that of the first project. I also built sign up and login forms. AWS Amplify was also setup using `npm install aws-amplify --save`. This basically allows the React app to talk to the AWS resources. This acts as the backend of the project, using its few simple modules. The login page was created using AWS Cognito. Sign up page was also created in a similar way. Session was added along with

logout button. Redirection to homepage was also added after each session start or end. Loading feature was added to the 'submit' button so that users can know that submitting is taking some time. After the front-end was completed, I started setting up the AWS account.

First, I setup the account in AWS. I opened a new account in AWS and got 12 months of free services. I created an IAM (Identity and Access Management) user account which basically enables you to manage users and user permissions in AWS. It is like adding a new person as an admin in a Facebook page. After that, I created a DynamoDB table and an S3 bucket for file uploads. I then created a Cognito user pool for handling user accounts and authentication in a secure and reliable way.

Deploying was the easiest process among all. As the S3 bucket was already ready, I created the build of my completed project by using `npm run build`. This converted my whole React project into html files, which I uploaded in the AWS S3 bucket by using the line `aws s3 sync build/ s3://movemandu-deploy`. Then I configured the Cloud-Front distribution so that I can get the link to my project.

### 6.2.2 Characteristics of the project

There are many characteristics in this project as well. This is also a fully responsive app and is also stable. I didn't use the Redux library, so this project is a very simple one. It is very secured as compared to the previous project. There is also a feature for blank pages: "Error 404: No Page Found". Also, the loading button was added so that the users can have some patience while submitting the form takes place.

### 6.2.3 Pros and Cons of the project

Using a framework is always an advantage, because there are so much features available. I used bootstrap so that I didn't have to worry about CSS. Debugging was very easy. React has that feature of real-time debugging. If there is any error in the code, React just shows its reason and the line where the error took place. This saves a lot of time searching for the error. Setting up the AWS helped saving a lot of time for backend programming. I didn't have to worry about backend, all the processes were handled by the AWS Amplify and Cognito. Deploying to the server was even easier. The app is also highly secured.

There are also a few cons in this project. Half of the time was spent setting up the environments and learning the basics of how the framework works. React's debugging feature didn't let me to see the result until and unless the codes are fully debugged. This created a bit irritation in some places.

### 6.3 Comparison of the two projects

I built two projects so that I can compare them, which is the main goal of this thesis. Different tests were carried out to compare them on various topics. They are:

a) Difficulty level:

As a beginner programmer, I really haven't worked in any framework before. So, it was a bit difficult to start my project in React. In the first project, however I could start and work quite comfortably. But for any intermediate or professionals in programming, it will be quite a lot easier and faster to work in React rather than from scratch.

b) Time Consumption:

The time consumption to complete the project was a bit more in React than in PHP. It is because I had to learn a new framework and its workings. Half of the time was spent on learning and setting up the environments and dependencies. But for a good-going developer, the time consumption would definitely be less in React (for implementing the same amount of features as in this thesis).

c) UI:

The UI depends upon the design you give to your application. But as in general, frameworks have availability of various libraries such as Bootstrap and other plugins to make the UI look more attractive. So, there is a high chance that the application made with a framework will have a better UI. The UIs in the projects in this thesis however are similar.

d) Weight of the project:

React has many dependencies to be installed to make it work. It even has its own engine, so the project of React or any other framework is quite heavier than the project done from scratch. Normal projects have only the files that are needed to run, so they are lighter. They also don't need to be compiled or built. So, React needs more storage. But when the React project is compiled, the build is very light as the whole React project is converted into some html files. So, it will be even lighter than the normal project to store in the server.

e) Speed:

The projects made for this thesis are very small. So, there is not much difference found in speed between them. But, if the projects are made in a large scale, there would definitely be much difference in them. React project will definitely beat the conventional project in terms of speed. JavaScript itself is known for its speed and React is fully made of JavaScript. Moreover, there is a strong engine for this framework to run on.

f) Quality or reliability:

React is obviously reliable than any other frameworks. Its popularity itself speaks for this reason. So, it is far more reliable than the application made from scratch. The quality of the product is also much better.

g) Security:

React is more secured than the other project. React itself has some security functions and also provides libraries and plugins which can make it even more secured. The other project however doesn't have any security. But still, some security measures can be implemented there as well.

h) Debugging:

It is much easier to debug in React than in the application made without any framework. React provides the feature of real-time debugging. If any error happens in the code, it suddenly shows what and where the error in the code is. This saves a lot of time finding out where the error took place in the code.

i) Features availability:

All the frameworks have some libraries and plugins available for them. Some developers are doing very hard work to bring new libraries quite often so that other developers can work on those frameworks quite freely. If you are working without framework, you still have some libraries available such as jQuery or JSON but can't beat to the availability as compared to that of frameworks. There are numerous amounts of features available for the framework as compared to without it.

j) Codes:

As comparing side-by-side between both the projects I had done, React has cleaner codes as compared to the other. There is not much difference in them but still being a object-oriented programming, React's codes look more appealing. PHP can also look cleaner if it is done with OOP. Also, the frameworks for PHP such as Laravel or Symfony are clean-looking.

k) Future and scope:

React has a very bright future ahead. The popularity that it is getting nowadays is one of the examples. Moreover, it is managed by Facebook, so the updates are rolled out quite often. There are also many libraries available for it. Many developers have changed from other programming languages and frameworks to React in recent days. Even the companies are now hiring the React developers the most. React is being kept as one of the requirements for the job seekers in almost all the companies and start-ups. This can prove that the scope of React is very high.

#### 6.4 Future Implementation

The features built in the project is just a beginning of my whole dream project. There are many other things left to develop. The project will be more secured in the future. But for now, I will be implementing first the remaining of front-end. After completing the front-end, I will be building API for the app where users will get all the information of apartments they have searched. There is not much to work on database, as AWS has already secured system there. It can allow any amount of entries and data, so I don't have to worry about it. I will add the features of social login from Facebook and Google which will make users easier to login and prevent them from exiting the app right on the first use. The app will have a great-looking UI to attract the users. The app will be launched in the Nepali market in the web format after completing the whole project and doing several tests.

When Movemandu web app becomes successful in working and delivering result to the users, I will start to develop it in Android using React Native. iOS will be developed at last. It is the least important one, as very few people use iOS devices in Nepal. Regular updates and debugging will be carried out. More and more apartments will be added to the system on the daily basis.

## 7 Conclusion

JavaScript is currently the most popular programming language in the world. Versatile and dynamic behaviour along with speed has been its main strength. Modern browsers, websites, electronic devices, servers all are compatible with JS. Similarly, the development of frameworks has been a boon to developers all around the world. Availability of all the necessary libraries and plugins to make an application is what developers had dreamt few years back. JS framework has created a revolution in the field of app and web development.

JavaScript framework is itself a powerful machine to develop web applications. But when it comes with additional functions and behaviours, it would become even more powerful. This is what React is. React has become the most powerful and dynamic framework among all the frameworks available in the market. It has the excellent performance among all. Developing a single page application has never been so easy and effortless. Not only that, it has a great future ahead with a lot of scope. Its development community is very big because it is an open source framework managed by Facebook. Developers from other programming languages are switching towards React. Small start-ups to big companies are all developing their software through React and React Native.

The goal of this thesis was to show the strength and importance of React, by comparing a project done in it with another one done without any framework used. Despite of some difficulties while doing projects, the objective of this thesis was met. Two different projects were built with similar features and comparisons were done successfully. Though the projects were very small, there were predictions for big-scale as well. The comparisons were made on the basis of speed, performance, time consumption, difficulty level, size, reliability, security, look, debugging, features and scopes. The good thing is that React won in all the topics, at least by a little margin in each. This proves that React is far better than conventional or traditional way of developing applications.

Thus, this thesis concludes that React is a very strong JavaScript framework with a very strong engine. It is the king of frameworks with a brighter future ahead. So, this thesis recommends learning React as it is going to be a compulsion to get a job in the near future. However, being technology advanced day-by-day, it cannot be ignored that there might come another even more powerful framework in the future. So, it is also recommended to learn other programming languages as well.

## References

- 1 IEEE. First computer program [online]. URL <https://ieeexplore.ieee.org/document/1253887/> [Accessed on August 9, 2018]
- 2 Archive. JavaScript introduced in 1995 [online]. URL: <https://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/news-release67.html> [Accessed on August 9, 2018]
- 3 ECMA-International. ECMAScript 2018 [online]. URL: <https://www.ecma-international.org/publications/standards/Ecma-262.htm> [Accessed on August 10, 2018]
- 4 ReactJS. JSX [online]. URL: <https://reactjs.org/docs/introducing-jsx.html> [Accessed on August 15, 2018]
- 5 W3C. W3C history [online]. URL: [https://www.w3.org/wiki/The\\_history\\_of\\_the\\_Web](https://www.w3.org/wiki/The_history_of_the_Web) [Accessed on August 18, 2018]
- 6 Infragistics. Rising of internet consumers [online]. URL: <https://www.infragistics.com/community/blogs/b/mobileman/posts/building-a-better-web-a-brief-history-of-web-development> [Accessed on August 20, 2018]
- 7 Statista. No. of mobile phone users [online]. URL: <https://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/> [Accessed on August 25, 2018]
- 8 Tech Target. History of databases [online]. URL: <https://searchsqlserver.techtarget.com/definition/database> [Accessed on August 22, 2018]
- 9 Tech Target. Cloud database [online]. URL: <https://searchcloudapplications.techtarget.com/definition/cloud-database> [Accessed on August 24, 2018]
- 10 Facebook. Full-stack development [online]. URL: <https://www.facebook.com/notes/facebook-engineering/the-full-stack-part-i/461505383919> [Accessed on August 25, 2018]
- 11 Speaking JS. History of JavaScript [online]. URL: <http://speakingjs.com/es5/ch04.html> [Accessed on August 28, 2018]
- 12 JavaScript Report. About JavaScript frameworks [online]. URL: <https://javascriptreport.com/the-ultimate-guide-to-javascript-frameworks/> [Accessed on September 1, 2018]

- 13 ReactJS. About React [online]. URL: <https://reactjs.org/> [Accessed on September 1, 2018]
- 14 Net Guru. React Native apps examples [online]. URL: <https://www.netguru.co/blog/12-great-apps-written-with-react-native> [Accessed on September 3, 2018]
- 15 Amazon. About AWS [online]. URL: <https://aws.amazon.com/about-aws/> [Accessed on September 5, 2018]
- 16 Amazon. Services from AWS [online]. URL: <https://aws.amazon.com/products> [Accessed on September 7, 2018]
- 17 Amazon. AWS Cognito usage [online]. URL: <https://aws.amazon.com/cognito/> [Accessed on September 8, 2018]

## Figure References

- 1 Cognus Technology. Web Development chart [online]. URL: <http://www.cognus-technology.com/ecommerce-development.html/> [Accessed on August 9, 2018]
- 2 Evolution of Web. Web development timeline [online]. URL: <http://www.evolution-oftheweb.com/> [Accessed on August 11, 2018]
- 3 Admec India. Client-side & Server-side concept [online]. URL: <https://www.adme-cindia.co.in/blog/what-are-client-side-and-server-side-scripting-web/> [Accessed on August 15, 2018]
- 4 Stable Kernel. Mechanism of mobile apps [online]. URL: <https://stablekernel.com/native-web-hybrid-lets-talk-about-mobile-apps/> [Accessed on August 20, 2018]
- 5 Medium. Front-end spectrum [online]. URL: <https://medium.com/@within-sight1/the-front-end-spectrum-c0f30998c9f0/> [Accessed on August 25, 2018]
- 6 Panoply. Real-world uses of DynamoDB & MongoDB [online]. URL: <https://blog.panoply.io/dynamodb-vs-mongodb> [Accessed on August 25, 2018]
- 7 GIT. Distributed VCS mechanism [online]. URL: <https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control> [Accessed on September 1, 2018]

- 8 Digital Clarity. Single page application vs traditional web application [online]. URL: <http://www.digitalclaritygroup.com/single-page-application-make-sense/> [Accessed on September 1, 2018]
- 9 Quora. Framework mechanism [online]. URL: <https://www.quora.com/What-is-the-use-of-framework-in-the-field-of-programming> [Accessed on September 3, 2018]
- 10 JavaScript Portal. Downloads of main 3 JavaScript frameworks between July 2017 - Jan 2018 [online]. URL: <https://javascriptreport.com/the-ultimate-guide-to-javascript-frameworks/> [Accessed on September 4, 2018]
- 11 Medium. Virtual DOM [online]. URL: <https://medium.com/naukri-engineering/naukriengineering-virtual-dom-fa8019c626b> [Accessed on September 4, 2018]
- 12 Difference of architectures in ReactJS and React Native [online]. [Accessed on September 5, 2018]
- 13 Home Page of Project-I [Project] [Accessed on September 9, 2018]
- 14 Home Page of Project-II [Project] [Accessed on September 11, 2018]