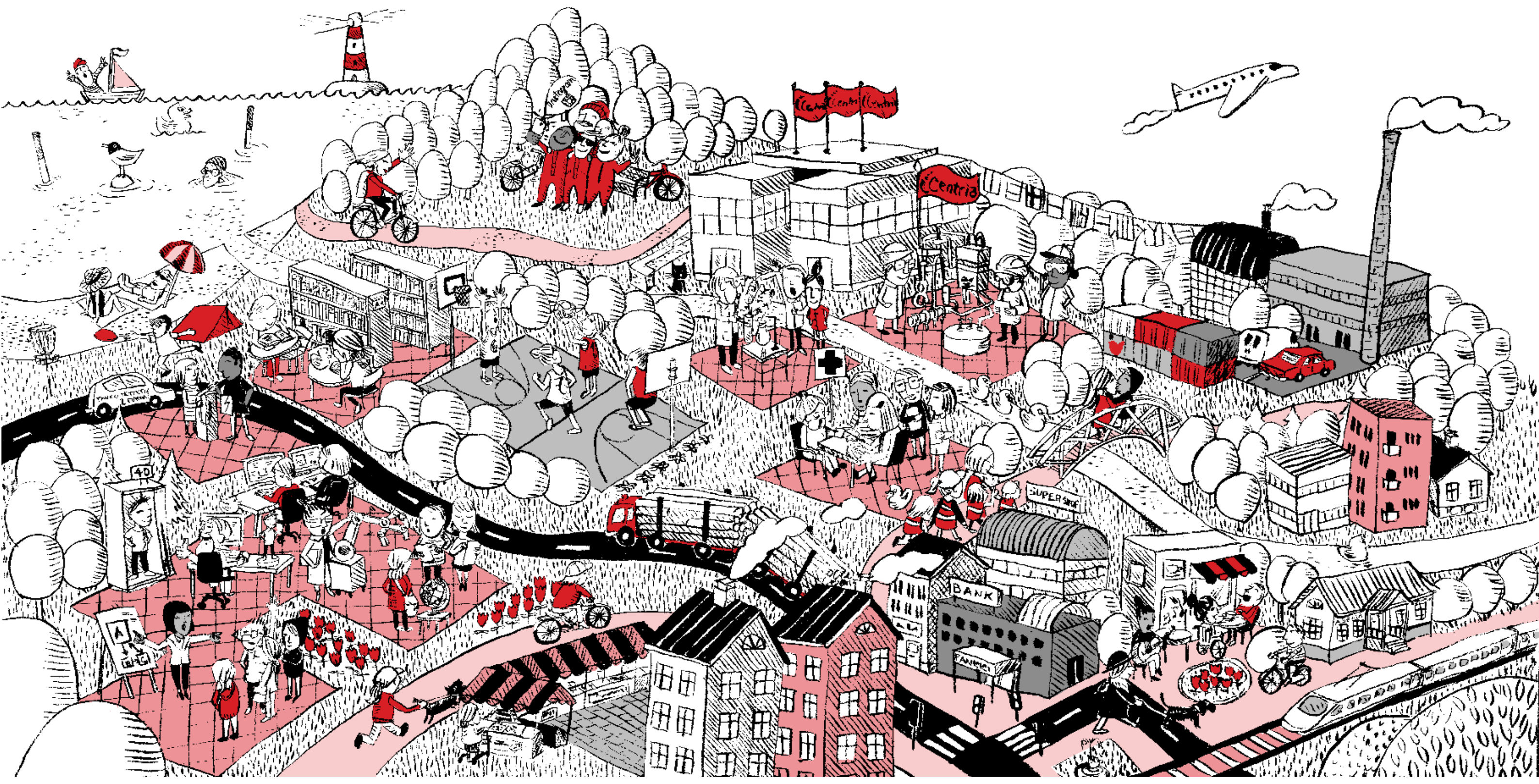
Amrit Bastakoti

USING NATIVE MOBILE SERVICES IN REACT JS

Thesis
CENTRIA UNIVERSITY OF APPLIED SCIENCES
Bachelor of Engineering, Information Technology
January 2022



ABSTRACT

Centria University of Applied Sciences	**Date** January 2022	**Author** Amrit Bastakoti
Degree programme Bchelor of Engineering		
Name of thesis USING NATIVE MOBILE SERVICES IN REACT JS		
Centria supervisor Jari Isohanni		**Pages** 22+ 3

The main objective of this thesis is to show the use of native mobile services in React.JS. Native services are hardware features of mobile devices, they are usually accessed through native programming. Developer needs to build different applications for iOS, Android and desktop using native programming. With React.JS, programmer can access some native services without writing separate native applications.

Native services uses React which is a framework for building native iOS and android application using JavaScript. It uses native components to render a UI and to access native services like camera, GPS and microphone.

In this thesis methods to access native services with React JS are presented. Some native services can be accessed through React JS and some of them cannot be accessed, this also varies between platforms and devices. In this thesis native services and their coverage support is researched.

Key words
Native Development, React JS, Web-based applications, JavaScript, Camera, Microphone, GPS.

ABSTRACT
CONTENTS

FIGURES

TABLES

1 INTRODUCTION

The main objective of the thesis is to represent how React JS is used in hardware components and mobile software. Native mobile services are specially designed focusing on building a single platform mobile app. The services are build using specific languages and editing tools. For example, we can use Java or Kotlin to create a native android app and use Swift and Ziel-C for iOS apps. Using native apps, we can automatically access smartphone capabilities such as camera and microphone. While using hybrid app we use plug-ins like Cordova to access native user capabilities. (Fletcher, 2021).

React JS is an open source JavaScript library which can be called React or React.js. It is mainly popular for single page applications that manages the view layer of web and mobile applications from which non-functional UI objects can be created. The developer of React is Jordan Walke, a software engineer on Facebook. React was featured on the Facebook news feed and on Instagram.com in 2012. The data in React can be changed without reloading the page. The speed and convenience of React framework has made it popular. It is focused on user interfaces that works same as MVC template view. It can be used in combination with other libraries or JS frameworks like as Angular JS in MVC. (Pandit, 2018)

In 2011, developers on Facebook began to face some problems with coding. The growing number of team members and app features has diminished a company. And at this time it was difficult to manage, as they faced many cascading updates. Their code requires emergency upgrades to make it work properly. All what they need to do was to do something with the user experience. Jordan Walke created a model that makes the process more efficient, and this marks the birth of React JS. The timeline mentioned below is the history of React JS. (Papp, 2018).

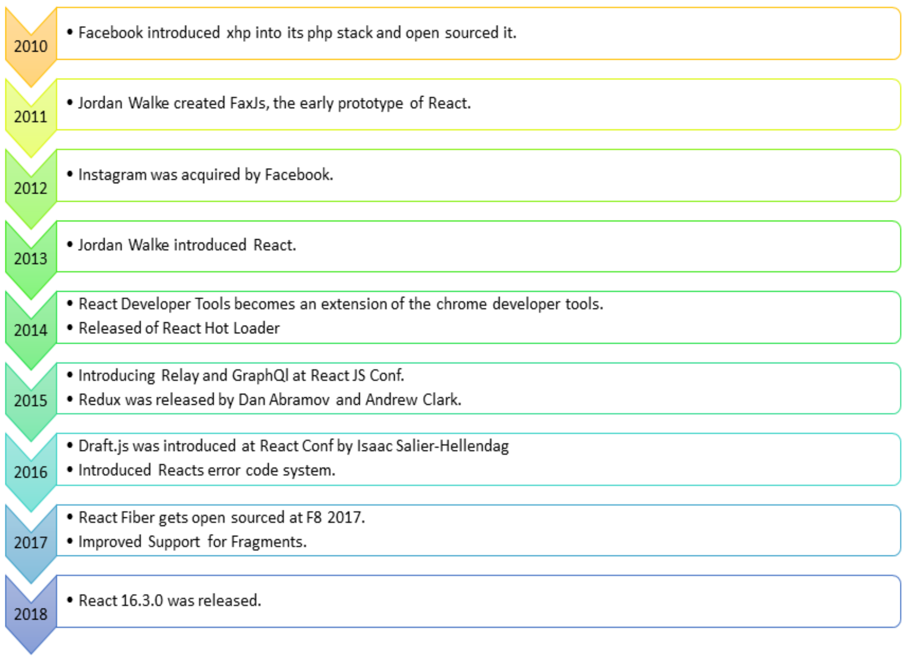

2010
• Facebook introduced xhp into its php stack and open sourced it.
2011
• Jordan Walke created FaxJs, the early prototype of React.
2012
• Instagram was acquired by Facebook.
2013
• Jordan Walke introduced React.
2014
• React Developer Tools becomes an extension of the chrome developer tools.
• Released of React Hot Loader
2015
• Introducing Relay and GraphQl at React JS Conf.
• Redux was released by Dan Abramov and Andrew Clark.
2016
• Draft.js was introduced at React Conf by Isaac Salier-Hellendag
• Introduced Reacts error code system.
2017
• React Fiber gets open sourced at F8 2017.
• Improved Support for Fragments.
2018
• React 16.3.0 was released.

Figure 1. History of React JS (Papp 2018).

With the native apps which has the ability to tap into specific resources can quickly access multiple services on a device, such as camera for recording an image of an object on a light-sensitive surface. GPS for allowing accurate determination of geographical locations, microphone for translating sound vibrations in the air into electronic signals and scribing them to recording medium, messages to send text, pictures, audio, video files and many more. The data which is associated or related to native app is stored on the device or remotely, for example cloud-based storage. (S. Gillis, 2021).

Most of the organizations are interested in investing in native mobile services development because of the benefits that are offered in comparisons to other types such as hybrid development or web development. For hardware intensive apps native apps in recent time is the best choice for every organization. However native mobile development is a challenging task concerning about different platforms but easy to access with its features in specific devices which allows developers to access the full feature of devices.

With React JS companies can engage mobile users into native mobile experiences without developing native applications. And some of the most popular native service (camera and GPS for example) can be accessed by using React JS.

However, native mobile services are exciting but there are many more challenging tasks and problems to consider in order to build programs which will help end users and work efficiently. The major problems and issues while using React JS in native mobile services are concerning platforms, design patterns and user interface design. For example, for native mobile services the library of JavaScript like React JS is not much more popular. But while developing hybrid app most of the libraries and framework of JavaScript is used. (Falessi et al. 2009).

Regarding native mobile services using React JS in fast moving technologies, this thesis will aim to present whether the native mobile services can be accessed through React JS or not and different platforms that can be built with React JS. The different libraries of JavaScript that helps to access the native mobile services, comparison of React JS with other frameworks, pros and cons of React JS are the main topic that will be elaborated in this thesis. Finally, the use of libraries to access native mobile services will be presented.

2 REACT JS FRAMEWORK

React JS is a flexible JavaScript Library that is declarative and efficient for building user interfaces. It is a component- based open source front end library which can be responsible only for the view layer of the application. It is maintained by Facebook. While building an application React JS helps to design simple views for each and every state, and efficiently update and render the right component when the data changes. The declarative view of React makes code more predictable and easier to debug. An application in React JS is made up of multiple components, each responsible for rendering a small, reusable piece of HTML. To build out of simple building blocks, components can be nested within other components to allow complex applications. The internal state can also be maintained by a component. (Latiyan, Yadav and Ghosh, 2017).

Being smooth, flexible, and accessible is the major goal of React. As well it enables to develop software application which is used to modify data without uploading or refreshing the page. Reacts pairs only with user interface while constructing the application. Collection of JavaScript like Angular JS in MVC is used as combination. Declarative, easier, portion, considerable, supporting towards server side are the main assets of React JS.(Pandit, 2021).

2.1 Components

Component is one of the crucial elements of the React.js i.e. core building blocks of React. The applications are developed in such a way that it is made up of small pieces called components. With the help of components, the task of building UIs is much easier. For example, when we access any sites in browser we can see many more components in the left, right, bottom, top and in the centre of the page. The UI is broken into small individual pieces and work them independently and merge them all in a parent component for the final UI. Components are independent and they serve the same purpose as JavaScript functions, but work in isolation and return HTML. (Rai and Yadav, 2018).

In React JS we mainly have two types of components which are functional components and class components.

2.1.1 Functional Components

The functional component in React can be created by simply writing JavaScript function. Simply JavaScript functions are functional components. Functional components is also known as stateless components and they are responsible for rendering UI. There is no render method used in functional components. It takes props and return JSX and do not have any state or lifecycle methods. Functional components are easy to test, can potentially have a better performance, are easy to debug, more reusable and they can reduce coupling are the main reasons why we choose functional components. (Yadav and Gumber, 2020).

2.1.2 Class components

Class components are used to add functionality to the application using multiple functions and all class-based components are child classes for the component class of React JS. When a component is declared in a program then it can be used in other components so as to present the use of class-based components in other components. While distinguishing class-based components from functional components, they have access to a state which dictates the current behaviour and appearance of the component. Any of the state in React JS can be modified by calling a setState() function.(Yadav, 2021). It is a regular ES6 class that extends the component class of React library. Class component is called statefull component because it determines the state changes and the implementation of the component logic. (Jain 2019.)

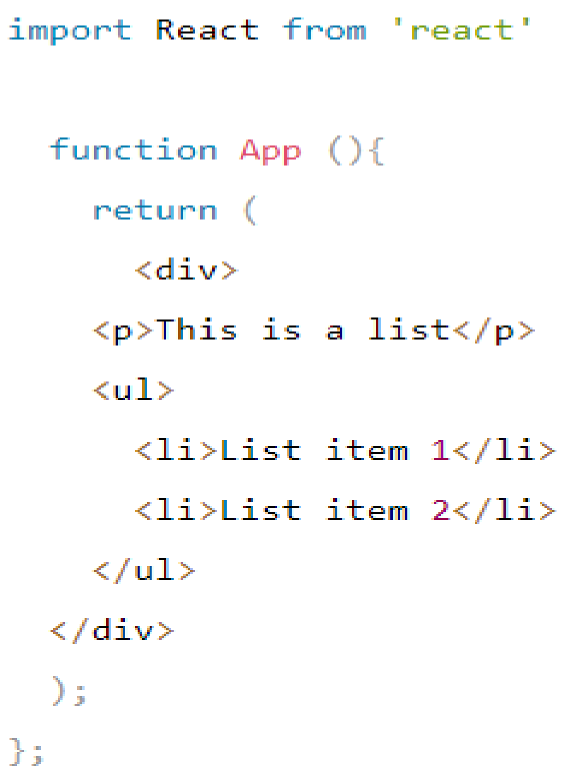
2.2 JSX

.
JSX can be defined as extension of JavaScript and it stands for JavaScript XML. The main purpose of JSX is to write HTML directly in React. While using JSX in React it is easy to create a template but not a simple template language instead it comes with the full power of JavaScript. In comparison to normal JavaScript, it is faster and performs optimization while translating to regular JavaScript. (Rao and Yadav, 2021).

2.2.1 Characteristics of JSX

JSX uses camel case notation while naming attributes and expressions is converted into regular plain JavaScript function calls after compilation. It follows the XML rule and produce react elements. In JSX large block of HTML can be inserted by writing in parenthesis. It allows writing expression in curly bracket and the expression can be any JavaScript expression or React variable. Overall, JSX with React makes easier to develop React application though it is not mandatory to use. (Srivastava, 2021)

Let us elaborate the example of JSX which shows the easy and simple process in React application.



```
import React from 'react'

  function App (){
    return (
      <div>
    <p>This is a list</p>
    <ul>
      <li>List item 1</li>
      <li>List item 2</li>
    </ul>
  </div>
  );
};
```

Figure 2. Example of JSX (Imoh 2021.)

JSX is not valid code of JavaScript, in which browsers can't compile or read it directly. At the time the process of transpiling is needed where Babel or TypeScript is used. There are many ways of returning JSX which can be returning the element directly, returning out of a ternary statement, returning early. JSX is an example of syntactic sugar in JavaScript that is designed to make things easier to read or to express. (Imoh, 2021).

2.3 A comparison of React JS to other popular frameworks

React JS is very flexible language that is used in front-end development. Due to the popularity and wide spread usage of JavaScript, many libraries and frameworks have been created to ease the development process for JavaScript developers. Among them React is one of the frameworks which is an open source JavaScript library that was created to help developers build user interfaces and UI components specially for single page applications. React is compared to other established frameworks like Angular JS and Vue JS.

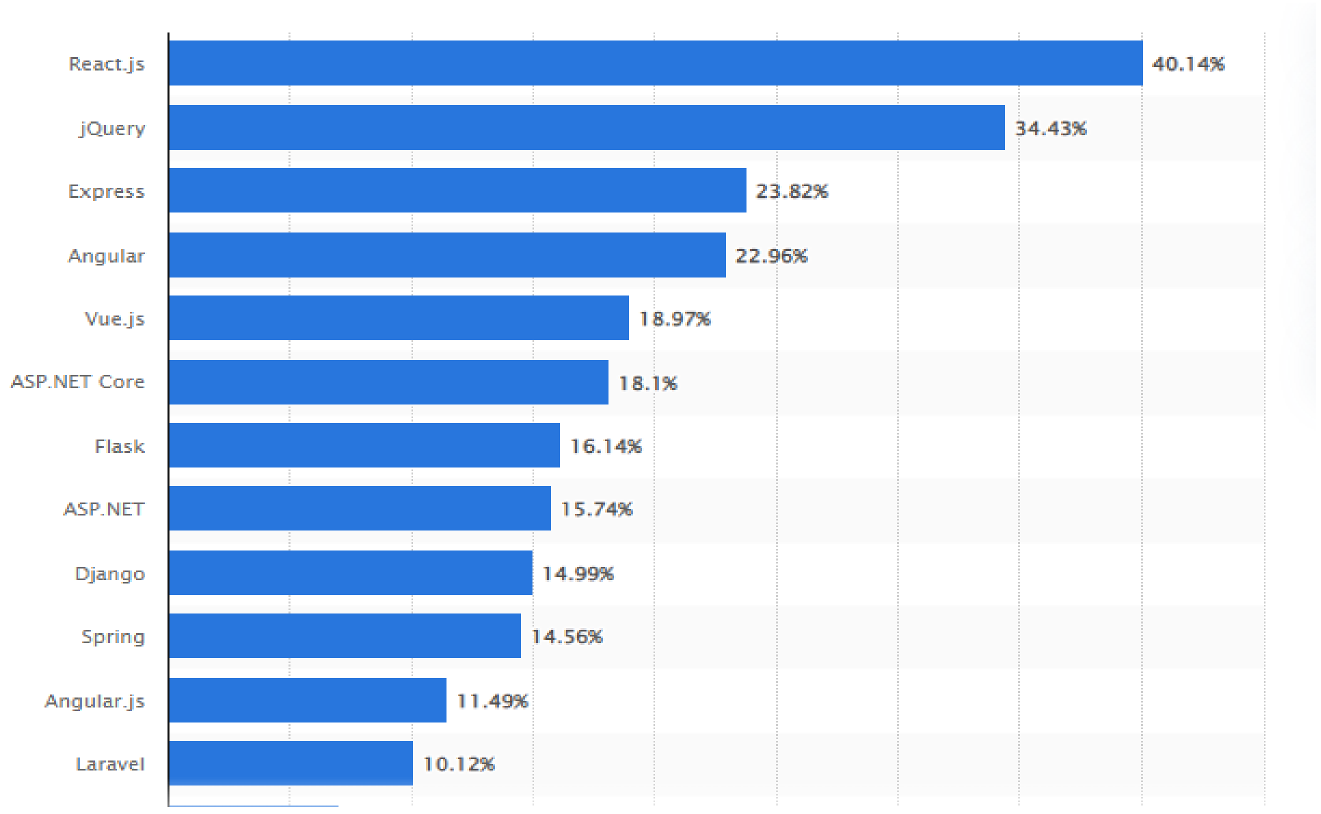


Figure 3. Most used web frameworks among developers worldwide, as of 2021 (Liu 2021.)

Runtime performance is the main topic to be discussed while comparing different frameworks. When it comes to speed, both React JS and Vue JS are different. Every time when a component changes in React, a render is triggered for the entire component sub-tree. If re-rendering is not used then it can be replaced with pure components. Next point is HTML and CSS application. In React HTML structure is expressed using JSX and CSS is managed inside React as well. However Vue JS holds classic web technologies and improves on them. (Thinkwik, 2018).

The learning step for a new developer is a lengthy process which requires many more concepts. But in React one learner only needs to get familiar with JavaScript and HTML. The developers having less experience feels or find it difficult to get started with Angular JS compared to React. (Thinkwik, 2018).

2.4 Pros and Cons of React JS

About the advantages of React, it is easy to learn and simple to use with the help of JavaScript concepts. The other factors that makes easier to use are its documentation, tutorials, and training resources. The components in React are reusable, we can use the components where it is needed. And with the help of virtual DOM, it helps to improve the performance. High pace of development, poor documentation, view part and JSX as a barrier are the disadvantages of React. The pros and cons of React are clearly distinguish in the table 1.

Table 1. Pros and Cons of React JS (Insignares, 2021)

Pros	Cons
React is component based, so it is not difficult to learn and use.	Due to the changes in the environment frequently, it may have problem in high pace of development.
The components can be reused wherever we need it.	There is updating of react technologies which may face the problem of poor documentation.
The JavaScript library provides flexibility.	For the new developers, JSX can be a barrier.
Due to virtual DOM, it improves the performance.	React has only ability to cover UI layers, which may need another tool for view part.

3 ACCESING NATIVE SERVICES IN REACTJS

Native mobile development approach is the traditional way of building applications for each mobile application, using different languages. Apps are built and adjusted exclusively for specific mobile operating system such as Android, iOS, and Windows. The developers uses tools and programming languages that are original for the device and the operating system. Apple, Google, Microsoft and other providers design specific tools, SDK and interface elements for their platforms. (Sakovich, 2020).

While creating iOS solutions we use Objective-C or Swift, environment such as XCode or AppCode, plus iOS SDK, Cocoa Touch and other tools. Likewise, for Android apps we use Java, Kotlin or C++, plus Android SDK, Mockplus. For the Windows phone apps the languages like C#, C++, plus Windows Phone SDK, Silverlight SDK are used. (Sakovich, 2020).

A native Android solution will not function on the iOS platform and vice-versa. There is full access to hardware functionality and software services which makes native apps feel like home in target devices. The hardware functionality like camera, GPS, voice recorder, audio, and video files and software services like calendar, notifications are accessed. The one another important feature that native apps, it can function or work in offline mode. Apps are downloaded from respective digital stores like App store and Google Play. These apps keep functioning in the background when the user are not actively using them. (Sakovich, 2020).

3.1 Camera

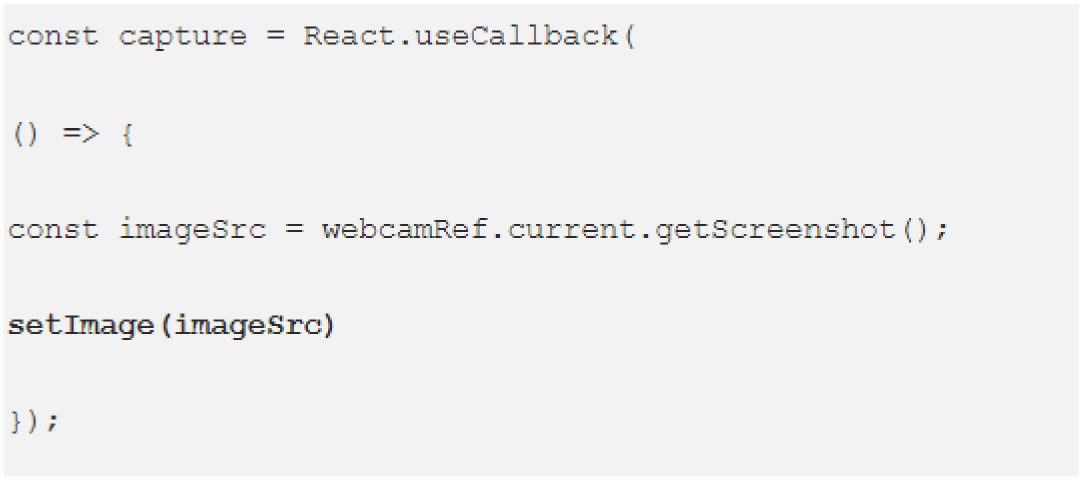
Camera can be accessed using different libraries of JavaScript in android, iOS and Windows devices. Expo- Camera is an open source platform which is used to create and make universal native apps with react. It runs on android and iOS devices and also in the web. React-native-camera is a camera component for react native and it supports barcode scanning. React-webcam is a webcam component which requires few node JavaScript modules. React-native-image-picker is a module of react native which allows to use native UI to select media from the device library or directly from the camera. CameraRoll is also a react-native module that provides access to local camera roll or photo library. Multicoder is the library that helps to record video and snapshots from device camera or desktop. The features like uploading and downloading of video and snapshots is supported. Camera-preview is one if the native features for mobile apps built with Cordova/PhoneGap and open web technologies. Videojs-record is a plugin for recording audio/video and image files. (Openbase, 2021).

Among different libraries to access camera let us take an example of React Webcam which is used to capture images using webcam. Among different modules react webcam requires a few Node.js modules. The first step is to create a form with the input elements which will capture the image of the user and some more details. After the form is created while start typing inside the input box, the values typed in the input field will be stored in the hook variable. The form can be styled inside the JSX file using CSS. Now for the webcam component we use some piece of code which is mentioned below in the figure 4.

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

import './cameraStyles.css'

import Webcam from "react-webcam";

const WebcamComponent = () => <Webcam />;

const videoConstraints = {
  width: 220,
  height: 200,
  facingMode: "user"
};

const WebcamCapture = () => {

const webcamRef = React.useRef(null);

  const capture = React.useCallback(
    () => {
      const imageSrc = webcamRef.current.getScreenshot();
    },

    [webcamRef]
  );

  return (
    <div className="webcam-container">
      <Webcam
        audio={false}
        height={200}
        ref={webcamRef}
        screenshotFormat="image/jpeg"
        width={220}
        videoConstraints={videoConstraints}
      />
      <button
      onClick={(e)=>{e.preventDefault();capture();}}>
      Capture</button>
    </div>
  );
};
```

Figure 4. Accessing Web Cam (Chowdhury 2021.)

The piece of code that is presented in Figure 4. helps to open the webcam from the device and it captures the images. The concept of hooks is used in the code. The different functional components like const WebcamComponent, videoConstraints, WebcamCapture are declared, and with the help of these functional components in React, web cam can be accessed. In the next Figure 5. we can see the piece of code that helps to view the image or photo that is captured in the webcam. (Chowdhury, 2021).



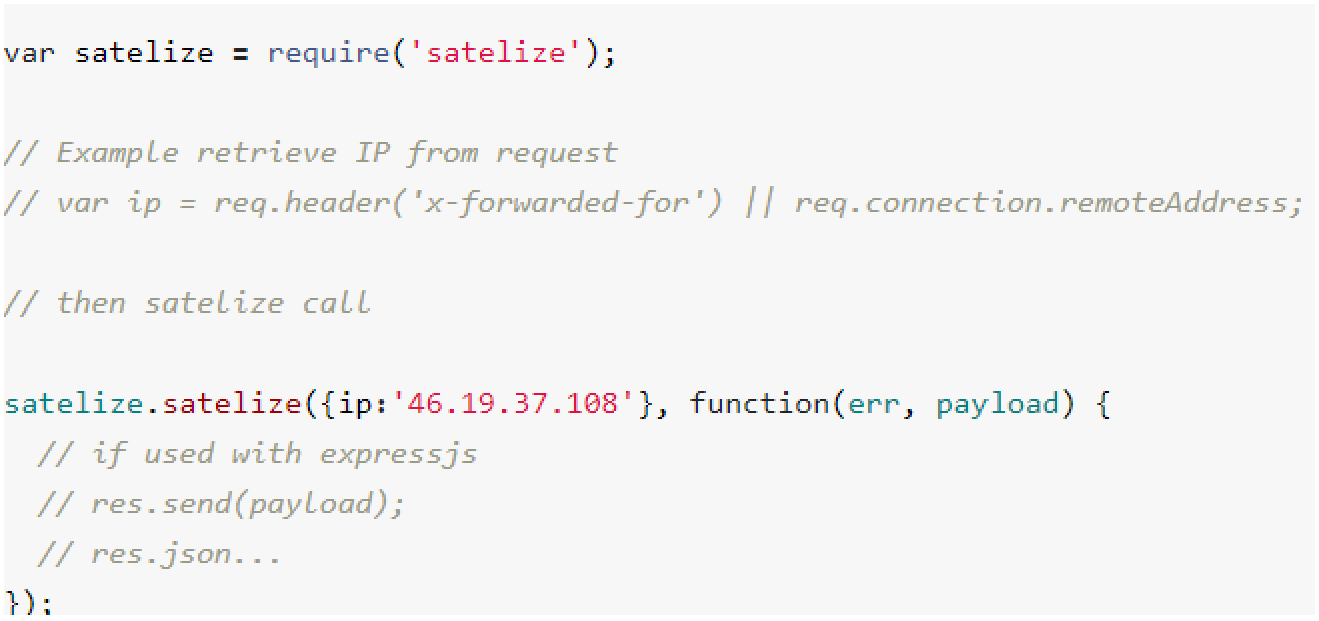
```
const capture = React.useCallback(

() => {

const imageSrc = webcamRef.current.getScreenshot();

setImage(imageSrc)

});
```

Figure 5. Example of code to view photo (Chowdhury 2021).

3.2 GPS

There are different popular libraries that can be used while accessing GPS. Fingerprintjs2 is a browser finger printing library with the highest accuracy and stability. Geoip-lite is a native Node JS implementation of MaxMind's GeoIP API which works in node 0.63. Leaflet-geosearch is a geocoding address-lookup library which supports various API providers. Express-ipfilter is a lightweight IP address based connection filtering system. Request-IP is a Node.js module which can be used for retrieving a request's IP address on the server. Ipstack is geolocation based on API stack API package for node.js. Node-IP locate which helps to find the geolocation data from IP address for example city, country, time zone using the IPLocate.io API. Satelize is a Node.js module which used for visitor geo localization by IP. Geoplugin is the library that can provides geolocation data of user's browser using services provide by http:// www.geoplugin.net/. (Openbase, 2021).

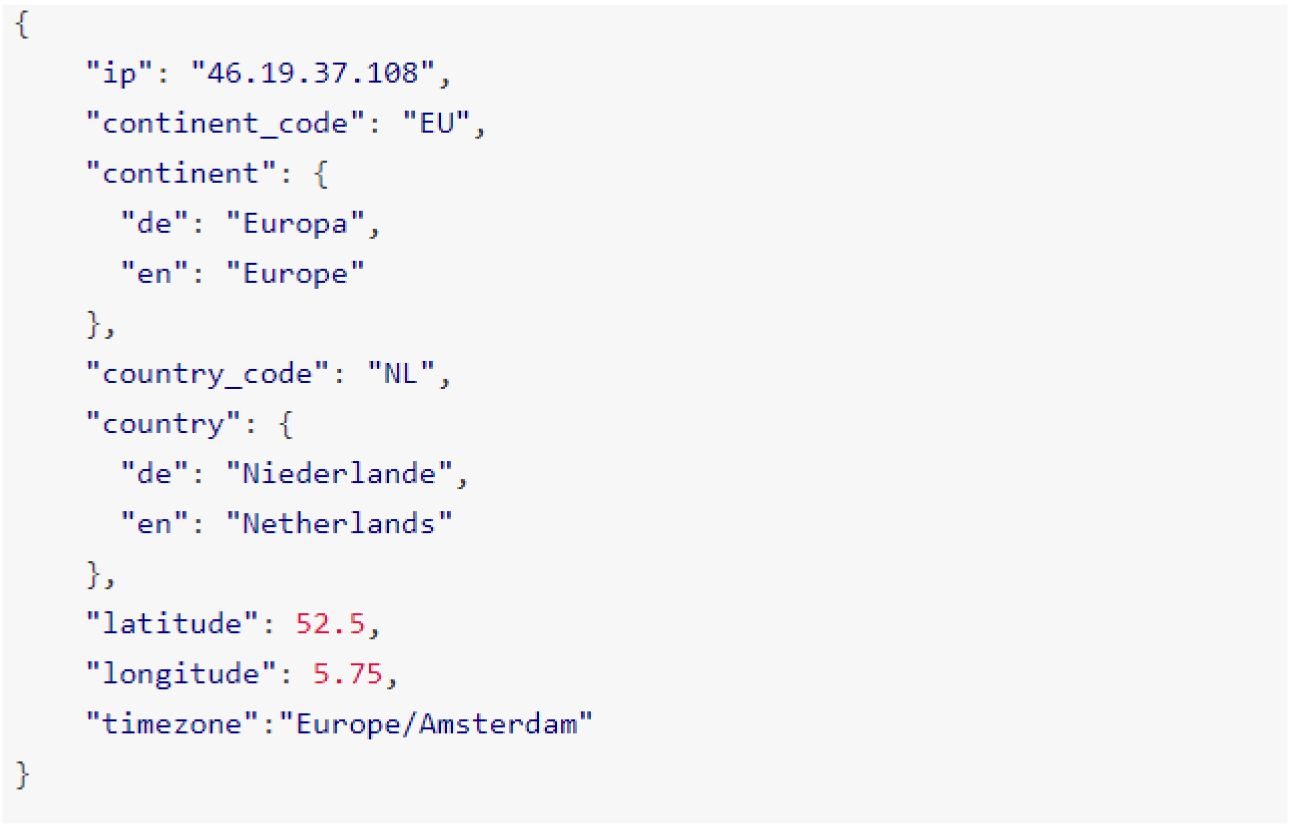
Let’s take an example of Satelize how GPS can be accessed by using this library. It works on the server side. This library is used after getting request IP that can help to make the call to get user location data. For installing Satelize library we use npm install satelize. The usage of satelize library can be represented by following piece of code shown in Figure 6.



```
var satelize = require('satelize');

// Example retrieve IP from request
// var ip = req.header('x-forwarded-for') || req.connection.remoteAddress;

// then satelize call

satelize.satelize({ip:'46.19.37.108'}, function(err, payload) {
  // if used with expressjs
  // res.send(payload);
  // res.json...
});
```

Figure 6. Feature of hook (Stack Overflow)

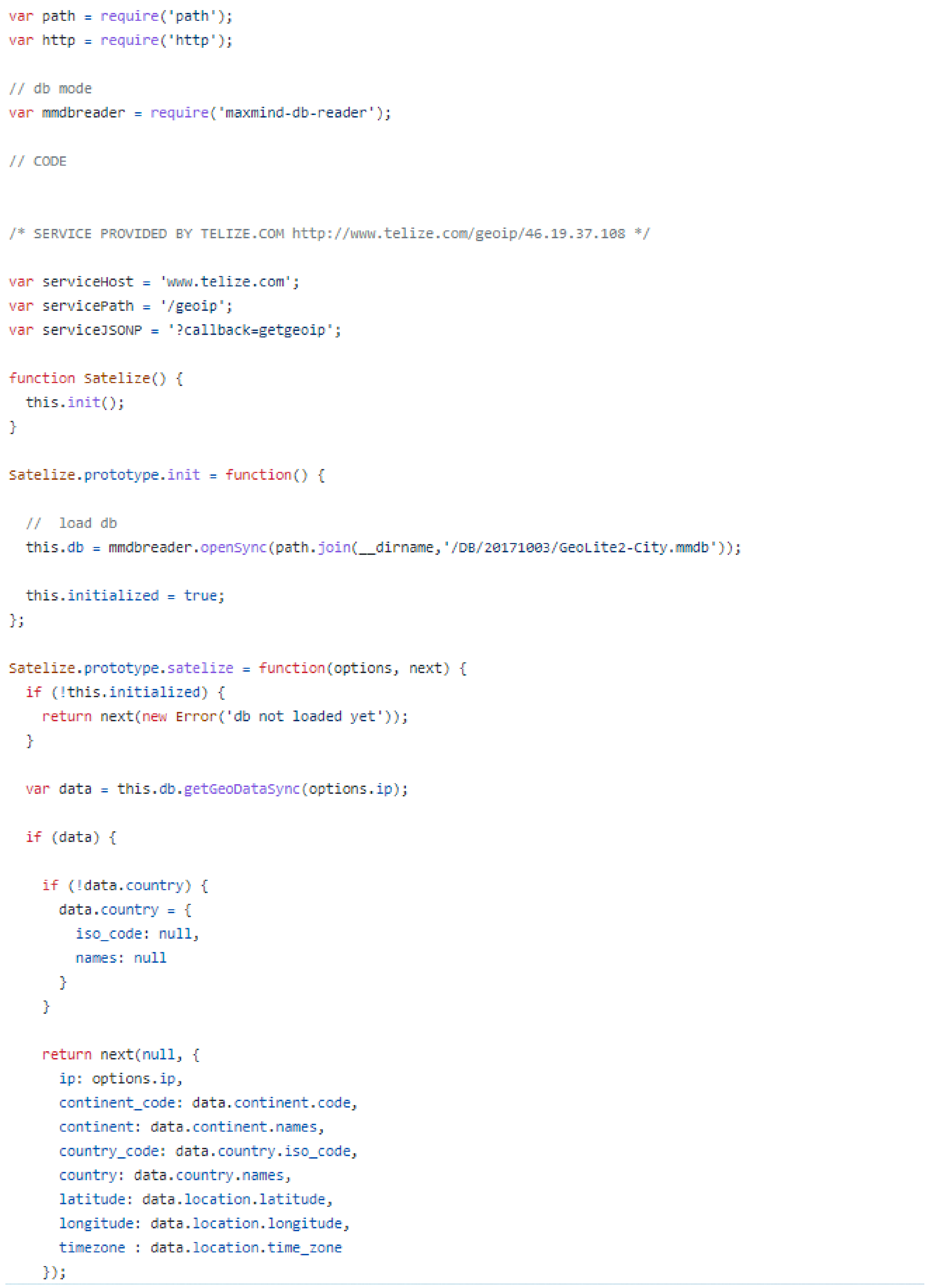
From Figure 6. we can understand the feature of hook which declares var satelize which helps to retrieve IP from request. Satelize.satelize with the ip number is defined with the function, which can be used in expressjs. Now in the Figure 7. We use node.js to arrange the data like IP, continent_code, country, latitude, longitude and timezone.



```
{
    "ip": "46.19.37.108",
    "continent_code": "EU",
    "continent": {
      "de": "Europa",
      "en": "Europe"
    },
    "country_code": "NL",
    "country": {
      "de": "Niederlande",
      "en": "Netherlands"
    },
    "latitude": 52.5,
    "longitude": 5.75,
    "timezone":"Europe/Amsterdam"
}
```

Figure 7. Data arranged in Node.js (Stack Overflow)

When all the data are arranged properly with the help of backend technologies, the arranged data needs to be returned so that anyone who access the GPS can locate and find everything they need. In the Figure 8. the variables like serviceHost, servicePath, and serviceJSONP are declared so that they can be passed with the help of function. It helps to access GPS with Satelize library.



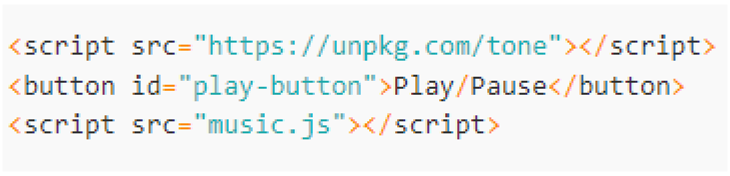
```
var path = require('path');
var http = require('http');

// db mode
var mmdbreader = require('maxmind-db-reader');

// CODE


/* SERVICE PROVIDED BY TELIZE.COM http://www.telize.com/geoip/46.19.37.108 */

var serviceHost = 'www.telize.com';
var servicePath = '/geoip';
var serviceJSONP = '?callback=getgeoip';

function Satelize() {
  this.init();
}

Satelize.prototype.init = function() {

  //  load db
  this.db = mmdbreader.openSync(path.join(__dirname,'/DB/20171003/GeoLite2-City.mmdb'));

  this.initialized = true;
};

Satelize.prototype.satelize = function(options, next) {
  if (!this.initialized) {
    return next(new Error('db not loaded yet'));
  }

  var data = this.db.getGeoDataSync(options.ip);

  if (data) {

    if (!data.country) {
      data.country = {
        iso_code: null,
        names: null
      }
    }

    return next(null, {
      ip: options.ip,
      continent_code: data.continent.code,
      continent: data.continent.names,
      country_code: data.country.iso_code,
      country: data.country.names,
      latitude: data.location.latitude,
      longitude: data.location.longitude,
      timezone : data.location.time_zone
    });
```

Figure 8. Accessing GPS using Satelize Library (Stack Overflow)

3.3 Microphone

As the libraries of camera and GPS same wise there are number of libraries that can be used for app’s audio processing. Howler.js is a library that is free and has reliable feature which is rich in library. In howler.js, the power of Web Audio API is utilized and HTML5 is used to give reliable cross-platform audio support. Likewise, Audo AI is used for Speech Enhancement API which is user friendly. The main objectives of Audo AI is to dismiss the background music from the audio and it is categorized into two parts as Batch Processing and Stream Processing. Next Dolby.io is one the best libraries for commercial level web audio solution. It provides the services like building audio conferencing, virtual classrooms, live-streaming apps, social apps in real time. Tone.js is a framework for web audio that especially focuses on allowing to create music with the help of JavaScript. Likewise, Twilio is used for generating automated audio communication features in JavaScript- based applications. (Wick-ramasinghe, 2021).

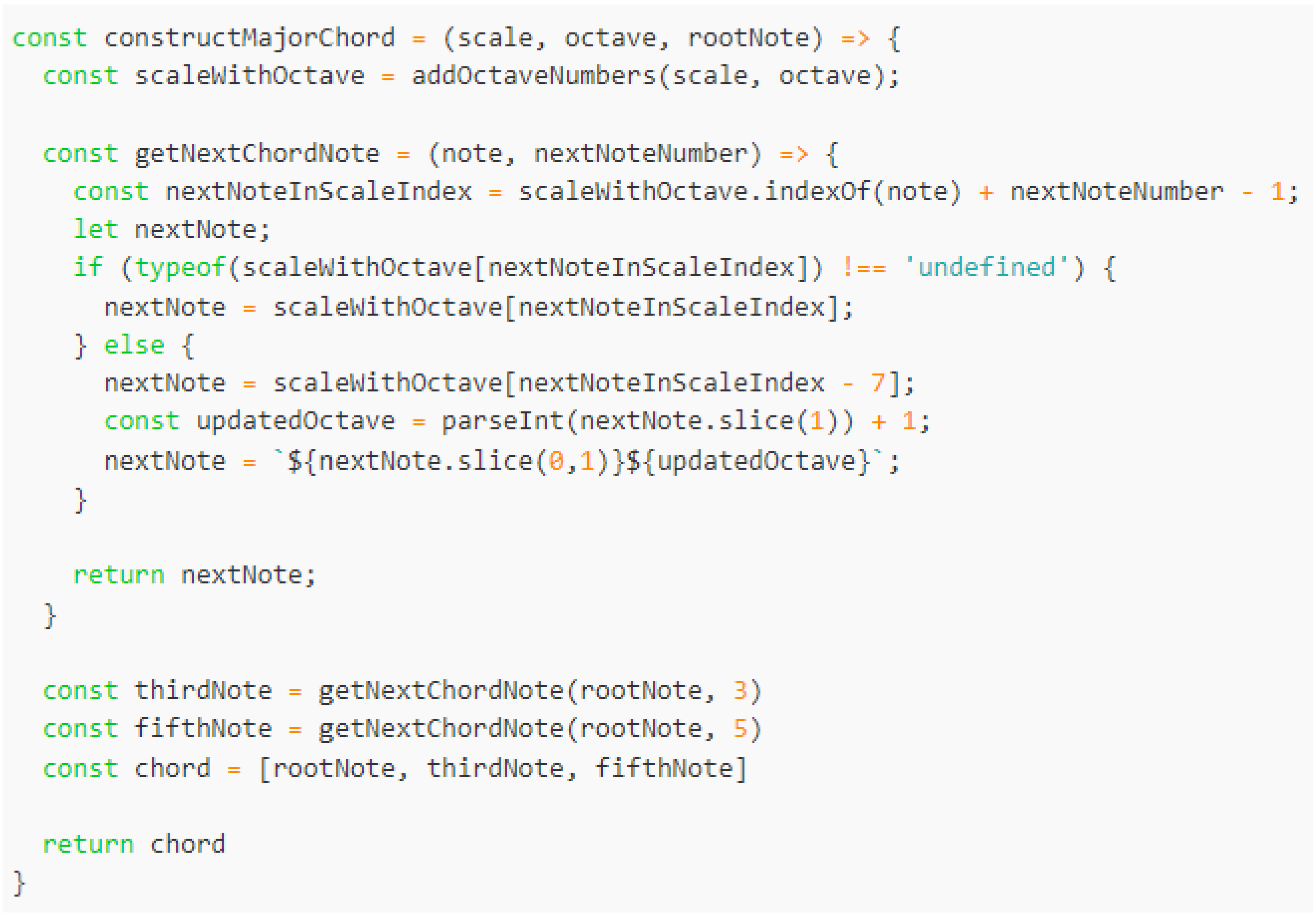
Let’s take a look how Tone.js is used to create music with JavaScript with some pieces of code. The script source are linked with the some audio and this audio can be accessed with the piece of code mentioned in the Figure 9. First, we need to set up the framework using node package manager and create a single HTML file as shown in the Figure 9.



```
<script src="https://unpkg.com/tone"></script>
<button id="play-button">Play/Pause</button>
<script src="music.js"></script>
```

Figure 9. Single HTML file with linked script (Faturovas 2020.)

After the single HTML file is created with script source, in the next step different functions likecon-structMajorChord, scaleWithOctave, getNextChordNote, nextNoteInScaleIndex are declared with their own specific tasks that helps to call the functions. This process generates major triads which is based on the root note, the octave and the scale. The process can be clearly seen in Figure 10.



```
const constructMajorChord = (scale, octave, rootNote) => {
  const scaleWithOctave = addOctaveNumbers(scale, octave);

  const getNextChordNote = (note, nextNoteNumber) => {
    const nextNoteInScaleIndex = scaleWithOctave.indexOf(note) + nextNoteNumber - 1;
    let nextNote;
    if (typeof(scaleWithOctave[nextNoteInScaleIndex]) !== 'undefined') {
      nextNote = scaleWithOctave[nextNoteInScaleIndex];
    } else {
      nextNote = scaleWithOctave[nextNoteInScaleIndex - 7];
      const updatedOctave = parseInt(nextNote.slice(1)) + 1;
      nextNote = `${nextNote.slice(0,1)}${updatedOctave}`;
    }

    return nextNote;
  }

  const thirdNote = getNextChordNote(rootNote, 3)
  const fifthNote = getNextChordNote(rootNote, 5)
  const chord = [rootNote, thirdNote, fifthNote]

  return chord
}
```

Figure 10. Creation of function that generates major traids (Faturovas 2020.)

The major steps that are described Figure 9 and Figure 10 are: setting up the beats per minute, creating main chords. The beats of the music per minute are set and main chords, chord progressions are created for the music or any sound. The main melody with the drums are added in the next step. Finally the bass is called and put all these in finishing touches. The steps created with the tone.js library makes the music audible to human being. (Faturovas, 2020).

4 RESULTS

Mobile devices and the handheld devices have changed the way people use the internet and have been very reliable and helpful to the end-users. Number of mobile applications companies has built the apps for better performance, cost-effectiveness, and more reach to current and potential customers. There are number of advantages of native features which is built for android and iOS that can facilitates between UI/UX professionals and the designer. Before developing any features, the designers will specify the UI methods what are common to users of a particular operating system and uses them. The developer of Android and iOS have idea to maximize the use of UI features for a particular platform. The user opens the native services and can interact with the interface directly. (Didukh, 2017).

User experience plays a vital role in any native applications. The best user experience is provided by the looks and feel of the application. So, in native applications it has user friendly interface that is inherited in a particular platform. Due to this reason there is no any difficulties at the time of using or dealing with the new program. It is because the developers will integrate the actions in the applications with UI/UX experts so that the applications are identical as possible. The native features in mobile devices are free to access and the applications are built up with maximum device's hardware. (Didukh, 2017).

Native features in mobile devices are easy to use due to its performance that run much faster than any other non-native applications. It is due to the language written that are supported by platform's ecosystems. Security in native features are fully protected with different layers of operating system so that it eliminates the risk of exploit. The security system is not dependent in third party frameworks. Native features are developed with the long development process of official platform SDKs which can deduct the risk of security issue. The implementation in hardware features is quick and easy due to the concerned APIs by Google and Apple operating systems. So that the native applications can take advantages of all the hardware features. Scalability, access to developers, stability, better UI/UX are the reasons that it is easy to use native features in mobile devices. (Samojlo, 2019).

The JavaScript libraries and JavaScript widget were developed focusing on dynamic interfaces that allows developers focusing on distinctive applications of Ajax. There are number of libraries that work on different area. CHR.js and cassowary is used for constraint programming. Google polymer, Dojo

Toolkit, jQuery, midori, Moo Tools and prototype JavaScript are used for DOM manipulation. The libraries like AnyChart, Babylon.js, Chart.js, Cytoscape are used for graphical or visualization. Angular JS, Bootstrap, Dojo widgets, jQuery UI are used for GUI and widgets. Google Closure Library, Joose, JsPHP, Microsoft Ajax library PDF.js are used for the purpose of pure JavaScript and Ajax. For template systems the libraries like Jasmine, Mocha, QUnit and Unit,js are used. Angular JS, Backbone.js, cappuccino, echo, ember.js, Ext js are the libraries that is used for web-application. (Paul Mueller, 2018).

All the libraries work under a certain criteria or environment. The most common environment that JavaScript libraries work are browser and Node.js. Node.js is a runtime environment that helps or allows the code to run outside the browser that is servers. Figure 10 briefly shows how the JS engine works, in which there is flow-diagram of JavaScript source code, Parser, Abstract Syntax Tree, Interpreter and compiler. Every engine have their own specifications to work under a certain module.

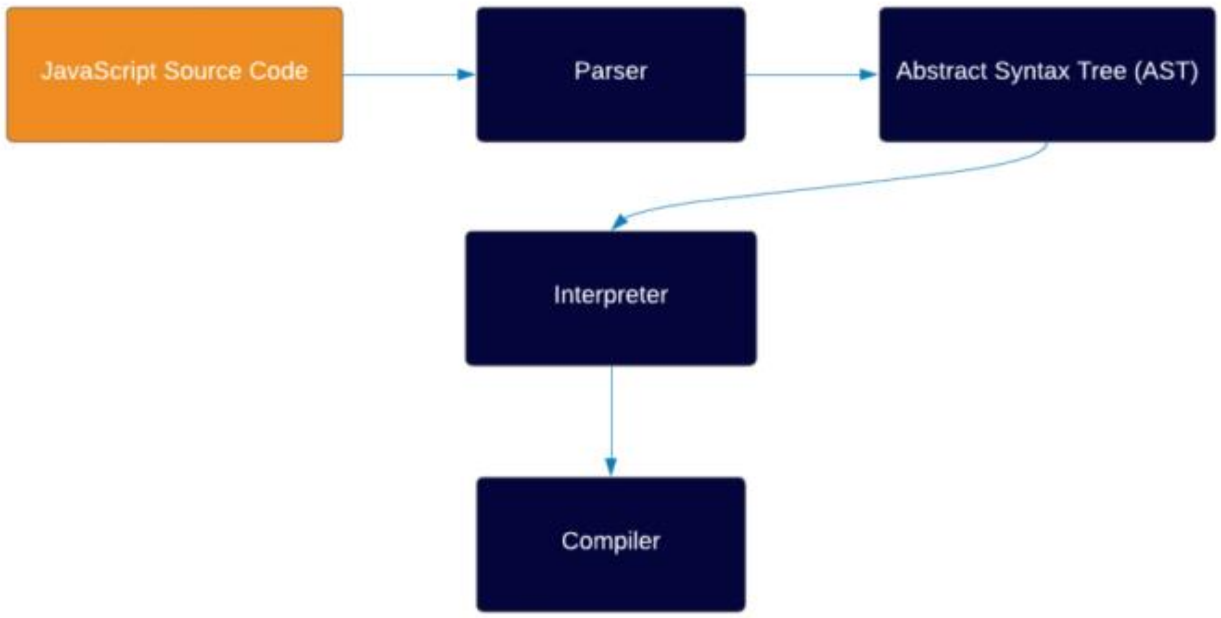


Figure 11. How JS engine works (Cohen 2021.)

Each environment needs an engine to transform the human readable code which is written in alphabets and numbers, into machine code that can only run by the computer processor. The examples of engine are Chrome V8 also used by Node, Firefox, SpiderMonkey, JavaScript core by Safari and Chakra by IE. (Cohen, 2021).

The engine uses parser to check the code after the human readable code. The parser is the one who knows all about the JS syntax and its rules. It checks the code line by line which conclude whether the syntax of the code is correct or not. If the parser finds the code error, then it stops the process and if the syntax code is valid then it generates Abstract Syntax Tree or AST. AST is a tree representation of the code which is used for converting to machine code when the code is inside a tree data structure. After the work of AST, the role of interpreter is to take the AST and transform the AST into an Intermediate Representation of the code. (Cohen, 2021).

An IR represents the source code with the data structure or code whose role is to play as an intermediate between JS code and machine code. Bytecode is a popular example of IR in JS engine. After the work of IR, there is a crucial role of compiler that takes the IR created by interpreter. And it transforms into machine code with certain optimizations. WebStorm, Komodo Edit, Visual Studio Code, Atom IDE are the popular examples of JavaScript Compiler. (Cohen, 2021).

To access different services in native applications, desktop applications and web applications the plain JavaScript can be used which is also called Vanilla JavaScript. Vanilla JS is the purest form of JavaScript, easy to handle and do not need to install libraries and do not have to go through NPM and the compilation steps. Without using any libraries and not downloading any tools Vanilla JS can access all the features used in different types of applications.

5 CONCLUSIONS

This research aimed to demonstrate the use of native services using React framework. The process of building mobile apps in the native official language for Android and iOS is complex and hard to learn with ease. So, there is use of web-native mobile application framework where developers can use the web languages that is already built. Mobile app framework plays a vital role to build the app one time and deploy it to Android and iOS, only with few changes in code. The app is distributed through the app stores like Android's play Store and Apple's App Store and through the browser. If the app is not distributed within the rules of stores then it is compulsory to choose the browser. And the app is chosen to be distributed through the app stores rules then we use a native stack and a web stack via WebView. The native code is generated in native stack where source code is translated to another language and finally will be executed on the platform with the same native controls. Web stack uses a web engine to render elements and run on the logic in browser. (Firtman, 2016).

The web stack approach while building native web apps has made it easier to access the native services in mobile or in any handled devices using libraries and framework of JavaScript. Number of web-based APIs are being used for the process of data storage, data transfer and for hardware devices. If the native apps cannot be built with the web stack approach or got a problem with the process, then we can depend upon the pure native code with Java, Swift or objective C which can create a bridge between the web stack and the native stack. Finally, with the help of bridge, the app will contain both the native code and web – based code. WinJs framework and Visual Studio can be used in windows 10 that supports the creation of universal apps using JavaScript and web technologies. (Firtman, 2016).

With the use of framework and different libraries in JavaScript has made a great experience to good developers because of well documented, flexibility, actively maintained, future thinking, tested, clean code and responsive community. The different elements like reference of function signatures, demos of actual size and narrative to use guide can be easily found in the libraries. If there is no availability of documentation developers might have complexity in building projects. The code with libraries works fast, smoothly and without problems due to a unidirectional data flow. The libraries and framework are actively maintained so that good developers can work efficiently. With the demands of the users and to upgrade the new version browser is being changed timely. At the time working libraries can suddenly stopped. The libraries is hosted in GitHub so that it can be maintained at the moment. We can find many more libraries at the time of building the projects, the libraries which is used once can be replaced with another libraries which helps for future projects. The libraries are well tested in which their functionality works as expected. During the time of building projects, a developer has a responsive community where they can discuss about the projects, errors, bugs. (Khan Academy, 2016).

Good developers are mostly focused on the user’s interfaces and for them users are everything. The developers has been always exploring user-centred design with comparative assessment, paper prototypes, wireframe prototypes and staging. With all these, users have almost great experience like with file size, performance, browser support, accessibility and responsive. (Khan Academy, 2016).

Though React has some drawbacks with the risk associated with architectural flexibility but finally, when concluding the use of React in native services it is light, fast and stable. It helps us to save time due to its reusable components and it is the one appropriate library for developing Single Page Applications. React is supported by a large group of community and it is easy to start coding.

REFERENCES

Chowdhury, S. (2021). *How to Capture Images Using React Webcam*. [online] Medium. Available at: https://javascript.plainenglish.io/capture-images-via-webcam-using-react-9282bb87de5a [Accessed 11 Oct. 2021].

Cohen, Y. (2021). *How JS Works Behind The Scenes—The Engine - Coralogix*. [online] Coralogix. Available at: https://coralogix.com/blog/how-js-works-behind-the-scenes%E2%80%8A-%E2%80%8Athe-engine/?fbclid=IwAR1VId-FcBUYeQEi2nRIZ_oK9y1dCEqacIL9qgqdtY6tHR5sBJwr1l6UTRuE [Accessed 17 Dec. 2021].

Didukh, B. (2017). *What You Need to Know About Native and Cross-Platform Apps | Agilie app development company blog*. [online] agilie.com. Available at: https://agilie.com/en/blog/what-you-need-to-know-about-native-and-cross-platform-apps [Accessed 22 Dec. 2021].

Faturovas, R. (2020). *How to create music with Tone.js · Devbridge*. [online] How to create music with Tone.js · Devbridge. Available at: https://www.devbridge.com/articles/tonejs-coding-music-production-guide/ [Accessed 15 Nov. 2021].

Firtman, M. (2016). *Web-native mobile app frameworks: How to sort through the choices*. [online] TechBeacon. Available at: https://techbeacon.com/app-dev-testing/web-native-mobile-app-frameworks-how-sort-through-choices [Accessed 18 Dec. 2021].

Fletcher, A. (2021). *Native Mobile App Development – An Explanation*. [online] Amico Hoops. Available at: https://amicohoops.net/native-mobile-app-development-an-explanation [Accessed 12 Sep. 2021].

Imoh, I. (2021). [online] www.telerik.com. Available at: https://www.telerik.com/blogs/how-jsx-react-works-under-hood? [Accessed 17 Sep. 2021].

Insignares, A. (2021). *React Pros and Cons: What are the Advantages and Disadvantages of ReactJS?* [online] Koombea. Available at: https://www.koombea.com/blog/react-pros-and-cons-what-are-the-advantages-and-disadvantages-of-reactjs/ [Accessed 18 Sep. 2021].

Jain, P. (2019). *Components In React*. [online] www.c-sharpcorner.com. Available at: https://www.c-sharpcorner.com/article/component-in-react/ [Accessed 14 Sep. 2021].

Khan Academy. (2016). *Which JS library should you use? (article)*. [online] Available at: https://www.khanacademy.org/computing/computer-programming/html-css-js/using-js-libraries-in-your-webpage/a/which-js-library-should-you-use [Accessed 22 Dec. 2021].

Latiyan, M., Yadav, S. and Ghosh, S. (2017). *React.js (Introduction and Working)*. [online] Geeksfor-Geeks. Available at: https://www.geeksforgeeks.org/react-js-introduction-working [Accessed 12 Sep. 2022].

Liu, S. (2021). [online] Available at: https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/ [Accessed 17 Sep. 2021].

Mueller, J.P. (2018). *10 Amazing Libraries and Frameworks for Your Web-based JavaScript Applications | New Relic*. [online] newrelic.com. Available at: https://newrelic.com/blog/best-practices/best-javascript-libraries-frameworks [Accessed 16 Dec. 2021].

Openbase. (2021). *10 Best JavaScript Camera Libraries in 2021*. [online] Available at: https://openbase.com/categories/js/best-javascript-camera-libraries [Accessed 5 Oct. 2021].

Pandit, N. (2018). *What Is ReactJS and Why Should We Use It?* [online] C-sharpcorner.com. Available at: https://www.c-sharpcorner.com/article/what-and-why-reactjs/ [Accessed 12 Sep. 2021].

Papp, A. (2018). *The History of React.js on a Timeline | @RisingStack*. [online] RisingStack Engineering - Node.js Tutorials & Resources. Available at: https://blog.risingstack.com/the-history-of-react-js-on-a-timeline/ [Accessed 13 Sep. 2021].

Rai, A. and Yadav, S. (2018). *ReactJS | Components*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/reactjs-components/ [Accessed 13 Sep. 2021].

Rao, S. and Yadav, S. (2018). *ReactJS | Introduction to JSX*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/reactjs-introduction-jsx/ [Accessed 16 Sep. 2021].

S. Gillis, A. (2019). *What is native app? - Definition from WhatIs.com*. [online] SearchSoftwareQuality. Available at: https://searchsoftwarequality.techtarget.com/definition/native-application-native-app [Accessed 13 Sep. 2021].

Sakovich, N. (2020). *Native Mobile App Development: Five Key Benefits | SaM Solutions*. [online] www.sam-solutions.com. Available at: https://www.sam-solutions.com/blog/native-mobile-app-development/ [Accessed 18 Sep. 2021].

Samojlo, G. (2019). *7 Reasons Why You Should Go for Native App Development*. [online] www.netguru.com. Available at: https://www.netguru.com/blog/why-native-app-development [Accessed 15 Dec. 2021].

Srivastava, A. (2021). *JSX Full Form*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/jsx-full-form/ [Accessed 15 Sep. 2021].

Thinkwik (2018). *Why ReactJS is gaining so much popularity these days*. [online] Medium. Available at: https://medium.com/@thinkwik/why-reactjs-is-gaining-so-much-popularity-these-days-c3aa686ec0b3 [Accessed 17 Sep. 2021].

Wickramasinghe, Y.S. (2021). *5 Top Audio Processing Libraries for JavaScript*. [online] Medium. Available at: https://blog.bitsrc.io/high-fidelity-web-audio-with-javascript-2e5fff0f071d [Accessed 10 Dec. 2021].

Yadav, S. and Gumber, V. (2020). *ReactJS Functional Components*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/reactjs-functional-components/?ref=lbp [Accessed 14 Sep. 2021].