ISSN (ONLINE):2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Vol.3, Special Issue.25, February 2017

Creation of Password Management App Using Ionic Framework and Firebase

Ms.M.Gowthami[1], Rakshana.R[2] Alagammai.P[3], Sapna.A[4]
[1]Assistant Professor, Department of Computer Science and Engineering
[2, 3, 4] UG Scholar, Department of Computer Science and Engineering
[1,2,3,4] Vel tech high tech Dr.Rangarajan Dr.Sakunthala Engineering College, Avadi, Chennai-53.
gowthamimailme@gmail.com, rakshana5jav@gmail.com

ABSTRACT:
One programming language that will runs on all mobile uses. To build apps with web technology, which is free and open source, fully cross platform, first class documentations and premier native plugins. Ionic emulates native app UI guidelines and uses native SDKs, bringing the UI standards and devices features of native apps together with the full power and flexibility of the open web. Ionic uses ***Cordova*** *or phone gap to deploy natively in app.*

1. INTRODUCTION

IONIC is completely free and open source, released under the permissive MIT license, which means you can use ionic in personal or commercial projects for free. For example, MIT is the same license used by such popular projects as query and Ruby on rails. CLI, or command line interface, is a tool that provides a number of helpful commands to ionic developers. In addition to installing and updating Ionic, the CLI comes with a built-in development server, build and debugging tools, and much more. If you are using the ionic cloud, the CLI can be used to export code and even interact with your account programmatically. There are many password managers on the market right now, all of which may or may not truly keep your own password manager so you know how your data is being stored and where it is being sent. In this particular tutorial we are going to see how to encrypt password data in JavaScript using he very strong AES cipher.to keep our passwords synchronized across all our devices and platforms we are going to bind all our data to firebase .The base behind our mobile applications will be Angular with ionic framework.

2. IONIC FRAME WORK

A complete mobile toolkit, built for web developer.

Everything you need to start creating fully functional mobile apps in just minutes.

Simple CLI to create, build, test and deploy your Ionic apps onto any platform .It's like magic, but not.

Ionic **NATIVE** unlocks native APIs and features by wrapping Cordova plugins in Typescript goodness.

Live reload because to compile and redeploy your app at every step of development is for chumps.

Ion icons icon pack includes hundreds of the most common app icons .MIT licensed, and ready out of the box.

DEEPLINKINGallows you to start your app from a web link to load a specific view right from the start.

AT COMPILING means your app loads lightning fast .Don't blink on that loading screen, you'll miss it

PERFORMANCE OBSESSED

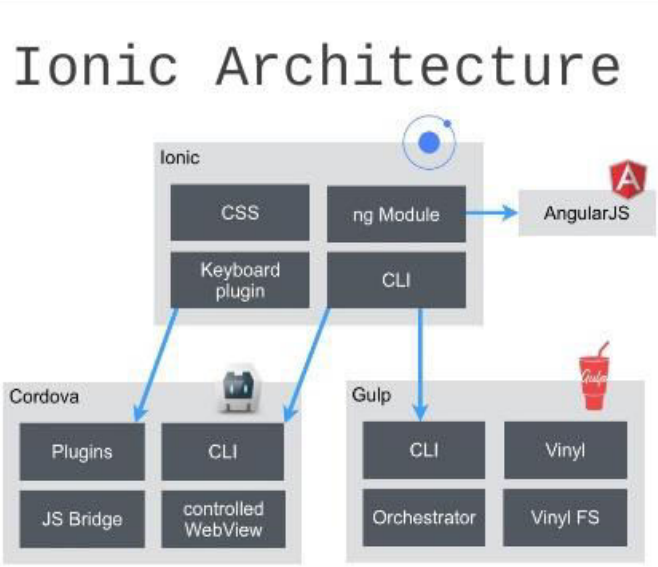
Speed is so important ,you only notice when it isn't there .Ionic is built to perform and behave great on the latest mobile devices with best practices like efficient hardware accelerated transitions ,and touch-optimized gestures .Trust me ,you'll be impressed.

ISSN (ONLINE):2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Vol.3, Special Issue.25, February 2017

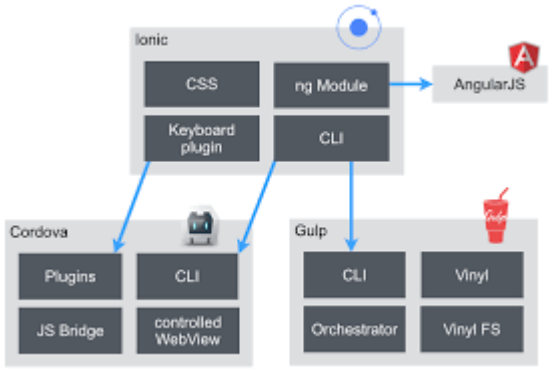
BEAUTIFULLY DESIGNED

Clean , simple ,and functional .Ionic is designed to work and display beautifully on all current mobile devices and platforms .With ready-made components ,typography ,and a gorgeous (yet extensible)base theme that adapts to each platform ,you'll be building in style.



NATIVE AND WEB OPTIMIZED

Ionic emulates native app UI guidelines and uses native SDKs, bringing the UI standards and devices features of native apps together with the full power and flexibility of the open web. Ionic uses **Cordova** or phone gap to deploy natively, or runs in the browser as a progressive web app.



2.1. IONIC FRAME WORK

FEATURES AND BENEFITS IN MOBILE APP DEVELOPMENT

Ionic (mobile app frame w) was created in the year 2013 by Drifty Co and provides tools for developing hybrid mobile apps using web technologies such as HTML 5, CSS and Sass. After taking feedback from clients& customer who tried to create Mobile apps, the team of drifty company decided to build their own

frame w mainly focus on performance and be developed with the latest web standards .A 1.0 beta was released in March 2014 and a 1.0 final in May.

WHERE DOES IONIC FIT?

Ionic, a complete open source SDK for HTML5 mobile app development framework is targeted at building hybrid mobile apps(a web app ,mainly built using HTML5 and Java script), which are basically small websites running in a browser shell in an app that have access to the native platform layer .Additionally , hybrid apps have lots of advantages in comparison to pure native apps (an application program that has been developed for use on a particular device or platform), generally in terms of platform support ,access to 3rd party code and of course speed development.
Different from a responsive framework, this comes with very native-styled mobile user interface (UI) elements or layout that you'd get with a native SDK for Android or is, but didn't exist before on the web .Also, ionic provides some opinionated but very powerful ways to develop Mobile app that eclipse existing HTML5 framework, so it required a native wrapper, such as phone Gap or Cordova so as to run as a native app.

3. FEATURES OF IONIC FRAMEWORK

Since it is a completely free and open-source framework ,ionic helps to build hybrid apps using HTML5,and makes use of Angolans for creating a powerful SDK perfectly-suited to develop highly interactive apps .It provides a great range of tools and services that make using the framework ,and once you've got node installed ,then ionic is as easy as running.

KEY FEATURES

Based on SASS and provides tons of UI components for creating robust and rich apps. Ionic comes transported with a powerful CLI, providing mobile app developers the ability to build and test ionic apps on any platform. It provides custom elements and methods for interacting with them by using angular, and one such element ,collection, repeat ,lets users to scroll through a list of thousands of items without any performance hits .Scroll-view is the another component that creates a scrollable container with which developer can interact using a native-influenced delegate system .Provides all the functionality found in native Mobile development SDKs, so users can create their apps ,and customize them for OS or Android and deploy through Cordova.

BENEFITS OF IONIC FRAMEWORK IN MOBILE APP DEVELOPMENT

Undoubtedly ,advancements in technology have landed us into a geological era ,that was beyond the ability of our hypothesis ,decades before .In addition to this ,around the clock development and innovations in technology have made us approachable to various jobs , through some quick clicks .People from across the world, spending most of their time in browsing activity with this advancement .Thus , keeping this in mind and the already got huge success of desktop computers or web applications, mobile are plenty of great benefits of mobile app development using ionic framework.

COMPLETELY FREE AND OPEN-SOURCE FRAMEWORK

You can build your apps on various platforms with this platform-independent frame work .Also, it has the ability to spit out the platform specific optimized CSS to match the native look and feel on assorted Mobile operating systems. Christo Ananth et al. [5] discussed about Positioning of a Vehicle in a Combined Indoor-Outdoor Scenario, The development in technology has given us all sophistications but equal amounts of threats too. This has brought us an urge to bring a complete security system that monitors an object continuously. Consider a situation where a cargo vehicle carrying valuable material is moving in an area using GPS (an

ISSN (ONLINE):2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Vol.3, Special Issue.25, February 2017

outdoor sensor) we can monitor it but the actual problem arises when its movement involves both indoor (within the industry) and outdoor because GPS has its limitations in indoor environment. Hence it is essential to have an additional sensor that would enable us a continuous monitoring /tracking without cutoff of the signal. In this paper we bring out a solution by combining Ultra wide band (UWB) with GPS sensory information which eliminates the limitations of conventional tracking methods in mixed scenario(indoor and outdoor) The same method finds application in mobile robots, monitoring a person on grounds of security, etc. Ionic gives codes of Mobile-optimized HTML, JS and CSS components, and reduces the need of code rewriting .Besides, ionic integrates into Angeles hence helping structure that code better as well as more manageable. Developers of apps on leading mobile platforms such as Android, widows, is, becomes possible. It is beneficial for the boost marketing and increased awareness in popularity of the apps.Ionic helps to save time, money or efforts.

FOR EXAMPLE,

FIG 2.1.EXAMPLE FOR IONIC FRAME WORK

A BEAUTIFUL DEFAULT UI

This framework comes with a lot of default CSS components and Java script components that cover most of the elementary things you would want to create into mobile application .Some of things include the Sliding Menu ,Form inputs ,Buttons ,Lists ,Navigation ,Tabs ,Sliding boxes and pop-ups and prompts. The default styles are very simple, sleek and you customize them by adding one of the pre-defined CSS classes to the element.

EASY AND FEASIBLE CROSS MOBILE APP DEVELOPMENT

Development if the app is very vital only once as well as it would be compatible with all the mobile devices. Also, it needs very limited use of time, resources and efforts, and helps in giving an integrated and look and feel. Apart from, Ionic helps in developing apps quickly with efficiency, and deploys standard tools with a single code base.

BUILT ON ANGULAR

If you've used ionic, then you would likely know that it is built on top of the Angular framework. Fundamentally, Ionic broadband Angular with a lot of stuff to make creating mobile applications with Angular super easy. Today, Angular, and so the benefits of Angular framework development can be put to use as well.

ISSN (ONLINE):2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Vol.3, Special Issue.25, February 2017

4. CREATE A PASSWORD MANAGEMENT APP USING IONIC FRAMEWORK AND FIREBASE

- ✓ There are many password managers on the market right now, all of which may or may not truly keep your own password manager so you know how your data is being stored and where it is being sent.
- ✓ In this particular tutorial we are going to see how to encrypt password data in JavaScript using he very strong AES cipher.to keep our passwords synchronized across all our devices and platforms we are going to bind all our data to firebase .The base behind our mobile applications will be Angular with ionic framework.

CREATING A NEW IONIC FRAMEWORK PROJECT

- ✓ The latest firebase JavaScript library
- ✓ The latest AngluarfireAngularJS library
- ✓ A mac with XCODE installed if building for ias
- ✓ NPK,Apachecordova,ionic,and android installed and configured

CREATING OUR PROJECT

- ✓ To start thins off, we were going to create a fresh ionic project using our terminal(mac and Linux) or command prompt(windows)

- ✓ Remember, if you're not using a Mac with Code installed, you cannot add and build for the IAS platform.
- ✓ At this point we're left with a new android and IAS project using the ionic framework BLANK template.

ADDING OUR LIBRARIES

In our IONIC PROJECTS www/Index.html we need to add all the various prerequisite libraries. The file's <head> should look something like the following:
For example,

```
<Head>
<Meta charset="utf-8">
<Meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
<Title></title>
<link her="lib/ionic/cvss/ionic.css" reel="style sheet">
<link her="cvss/style.css" reel="style sheet">
<scripter="lib/ionic/jess/ionic.bundle.js"></script>
<scripter="jess/ng-cordova.min.js"></script>
<script sec="cordova.js"></script>
<scripter="jess/firebase.js"></script>
<scripter="jess/angularfire.min.js"></script>
<scripter="jess/app.js"></script>
</head>
```

Wait! Where did jess/forge.min.js come from? Well, we cannot just download that file so we need to build it in the next step.

BUILDING THE FORGE LIBRARY

You'll remember that a segment on AES Ciphers with java library using the forge library.

DOWNLOADING FROM Godthab

- ✓ Unlike many Java script library projects on Github, forge does not offer a minified release / distribution file for including in your browser based application .This means we need to download the project and do it yourself.
- ✓ We have two options when it comes to obtaining the project. We can either download the latest master branch **archive file, or** clone the project Get:
- ✓ Get clone https://github .com/digitalbazzar/forge. It
- ✓ Both methods will accomplish the same thing.

5. INSTALLING THE DEPENDENCIES

- ✓ Because we're working with ionic framework and apache Cordova, I'm confident at this point you already have the node package manager (NPM) installed.
- ✓ With the download forge project as your current working directory your terminal or command to install all the necessary NPM dependencies:

NM install

They'll be downloaded locally to the project rather than globally on your machine.

MINIFYING FOR USE

With all the dependencies installed, it is now time to build a single minified file for use with our project .Make sure the forge is still the current working directory in your terminal or command prompt and run the following:

NM run minify
This will execute the minify script creating jess/forge.mn.js for use in our ionic framework project.

Preparing our JavaScript file
For simplicity, we are going to be doing all custom JavaScript coding in our www/jess/app.js file. It will contain all our controllers, factories, and prototypes.

Naming our Angular module
For cleanliness, we are going to give our Angular module a name.
Varpasswordapp=angular. Module (JavaScript)
In this case we've named our module passwordapp and it will be used for all controllers and factories in our application.

GETTING FIREBAS STARTED

WE'RE not quite ready to talk about firebase, but we are at a point where it would be a good idea to initialize it.At the top of your www/jess/app.js file, outside any Angular code, you want to add the following line:

Verb = new firebase ("http://instance_id.....")
Of course you need to replace INSTANCE _ID_HER with your personal firebase instance. By setting it globally, it will load before Angular and can be used throughout your application.
We also need to add firebase Tp our Angular module .In the end it will look something like this:
Varpasswordapp = angular. Module ('….")

CREATING A CIPHER FACTORY FOR ENCRYPTION AND DECRYPTION

THE FACTORY FOR CIPHER TEXT

- ✓ The $cipher factory will accomplish two things. Using the forge library, there will be an encrypt (message, password) function and a decrypt (ciphertext, password, salt, IV, options) function.

- ✓ Inside your **www/jess/app.js** file, add the Angular factory

- ✓ Encrypting will leave us with some cipher text as well as the salt and initialization vector used during encryption. All this information is required to decrypt.

Our strategy for safe storage

- ✓ The idea behind our application is simple. All data will be encrypted using AES and can only be accessed with a master password.
- ✓ The master password will never be saved in our application and an incorrect master password will only give junk data.
- ✓ So how are we going to accomplish in this?
- ✓ When the user creates a master password in our application, instead of saving the master password in our application, we are going to encrypt the static string **authenticated** and store the cipher text it produces in our application. When the user wishes to sign into the application we will decrypt the **authenticated** cipher text using the entered password. If it is anything but Authenticated when decrypted, we'll reject the user's access.

MAKING SOME HELPER PROTOTYPES

- ✓ The cipher factory is complete, but in order to stick with our safe storage strategy we need to create a few string prototypes. For examples, when decrypting our **Authenticated** string, if the password is incorrect then the dat will error when converting into a string. To avoid this we can do comparison against a hexadecimal value.
- ✓ We'll also need a prototype for generating a unique hash from a string value.
- ✓ Both these string prototypes should go at the very bottom of your www/jess/app.js file.

USING THE ANGULARJS UI –ROUTER

- ✓ Our application is going to have several different screens:
- ✓ Unlock the application
- ✓ Register a new master password
- ✓ Sign into firebase
- ✓ Viewing password categories
- ✓ Viewing password in a category
- ✓ Creating a password
- ✓ Viewing a password

To accomplish different screens or views in an Angular application we will be using the **UI –router** since it is already bundled with ionic framework.

6. PREPARING OUR STATES

With the Angular UI-ROUTER, each view or screen is considered a state. In order to use we must configure the routes and the controllers associated with them .In your ionic project's www/jess/app.js file, let's add the chunk of code which will represent the states for each of our screens.

CREATING OUR VIEWS

- ✓ We've just configured all of the routing information for our states so it is now time to set up our view templates. Essentially these are the HTML pages that will represent each of our screens.
- ✓ The first step is to prepare our ionic project's www/index.html file to use UI states .this is easy and be accomplished in just a few lines .Navigate to the <ion-pane> lines and replace them.
- ✓ Time to go through the process of designing each of our views. Brace yourself. Our templates are going to be very simplistic. Feel free to be fancier in your view design.
- ✓ Create and open the file called www/templates/locked.html and add the required codes.
- ✓ Now create and open the file called www/templates/create_vault.html and add the required codes.

- ✓ You've created the views for creating and using a master password. Owlets move onto our list screens. The first will be in a file called www/templates/categories. The second list will be in agile called www/templates/password_list.html and will be a list of passwords that reside in a particular that reside in a particular category.
- ✓ So, now we need three more views. One for creating new passwords , one for viewing passwords, create and open www/templates/password_new.html and add the codes for title, username and password. The next view, www/templates/password_view.html, will be similar to the new password screen, but this time we'll only be showing the data. For simplicity purposes, the password will be revealed upon load.
- ✓ Time to go through the process of designing each of our views. Brace yourself. Our templates are going to be very simplistic. Feel free to be fancier in your view design.
- ✓ Create and open the file called **www/templates/firebase.html** and add the required codes.

Sample coding:

```
<ion-view title="Firebase Login">
<Ion-content>
<Div>
<div class="list list-inset">
<label class="item item-input">
<input nag-model="username" type="text" placeholder="Username" />
</label>
<label class="item item-input">
<input nag-model="password" type="password" placeholder="Password" />
</label>
</div>
<div class="padding-left padding-right">
<div class="button-bar">
<a class="button" nag-click="login (username, password)">Login</a>
<a class="button" nag-click="register (username, password)">Register</a>
</div>
</div>
</div>
</ion-content>
</ion-view>
```

Now create and open the file called www/templates/secure.html and add the following code:

```
<ion-view title="My Images" nag-init="">
<ion-navy-buttons side="right">
<button class="button button-icon icon ion-camera" nag-click="upload ()"></button>
</ion-navy-buttons>
<Ion-content>
<div class="row" nag-repeat="image in images" nag-if="$index % 4 === 0">
<div class="col col-25" nag-if="$index <images. Length">
<imgng-src="data: image/jpeg; base64, {{images [$index].image}}" width="100%" />
</div>
<div class="col col-25" nag-if="$index + 1 <images. Length">
<imgng-src="data: image/jpeg; base64, {{images [$index + 1].image}}" width="100%" />
</div>
<div class="col col-25" nag-if="$index + 2 <images. Length">
<imgng-src="data: image/jpeg; base64, {{images [$index + 2].image}}" width="100%" />
</div>
<div class="col col-25" nag-if="$index + 3 <images. Length">
```

```
<imgng-src="data: image/jpeg; base64, {{images [$index + 3].image}}" width="100%" />
</div>
</div>
</ion-content>
</ion-view>
```

At first glance the code we have in our www/templates/secure.html file may look nuts. It really isn't. It is a grid system.

Configuring Our Firebase Instance

Defining Permission

- ✓ Building a Firebase To-do List or Creating a Password Manager, we're going to be using the same permission strategy:

Sample coding:

```
{
  "Rules": {
    "Users": {
      ".write": true,
      "$aid": {
        ".read": "auto! = null &&auth.uid == $aid"
      }
    }
  }
}
```

Everyone will be able to write to the users node (create a new account), but only authorized users will be able to read data. In this case data will be the users own images. You can paste the JSON rules in the Security & Rules section of the Firebase dashboard.

Allowing For Account Creation

- ✓ In the Login &Auto section of the Firebase dashboard, we must enable Email & Password authentication. There are other types of authentication, but for this particular tutorial we're going to focus on email and password based.
- ✓ Enabling this will allow people to register new accounts and create a unique simplelogin: x user key in our nosily data structure where x is an auto incrementing numeric value.
- ✓ Understanding The Data Structure
- ✓ The data stored will be of very strict formatting. It will of course be JSON, but it will look like the following:

Sample coding:

```
{
  "Images": {
    "unique_image_id_here": {
      "image": ""
    }
  }
}
```

- ✓ When storing our images, Firebase will automatically create our unique image ids. You'll see that shortly.
- ✓ Making Controllers To Pair With Our View Logic

The Firebase Controller

ISSN (ONLINE):2395-695X
ISSN (PRINT): 2395-695X

International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)
Vol.3, Special Issue.25, February 2017

- ✓ The logic behind the Firebase Controller is as follows:
- ✓ If the user chooses to sign in, check the email and password and redirect to the secure page if successful
- ✓ If the user chooses to register, create a new account and then immediately sign into and redirect to the secure page

Sample coding:

```
imageApp.controller ("Firebase Controller", function ($scope, $state, $firebaseAuth)
 {
VarfbAuth = $firebaseAuth (fob);
    $scope. Login = function (username, password) {
Faith. AuthWithPassword ({
Email: username,
Password: password
      }).then (function (outdate) {
        $state. Go ("secure");
      }).catch (function (error) {
Console. Error ("ERROR: " + error);
      });
   }

   $scope. Register = function (username, password) {
Faith. Create User ({email: username, password: password}).then (function (userdata) {
ReturnfbAuth. AuthWithPassword ({
Email: username,
Password: password
        });
      }).then (function (outdate) {
        $state. Go ("secure");
      }).catch (function (error) {
Console. Error ("ERROR: " + error);
      });
   }

});
```

Firebase stores sign in information on the device so that you can later make use of the getAuth () functions to see if you're currently authenticated.

The Secure Controller

- ✓ The logic behind the Secure Controller can be a little more complicated and is as follows:
- ✓ Clear the history stack so the back button doesn't bring us to the sign in screen again
- ✓ Create an empty array of images just in case no images exist on Firebase. Don't want any undefined errors
- ✓ Check to see if the user is signed in
- ✓ If not signed in, redirect back to the Firebase view
- ✓ If signed in, pull down the array of images from the server
- ✓ If the user chooses to upload an image, open the camera
- ✓ If the user accepts the camera image, then immediately push it to Firebase

```
imageApp.controller ("Secure Controller", function ($scope, $ionic History, $firebase Array, $Cordova Camera)
 {
        .
```

```
        .
        .
        .
});
```

- ✓ Notice the Camera.DestinationType.DATA_URL. This means that we are going to store the image as base 64 data in Firebase.
- ✓ (Properly Testing Our Apposing the Web Browser
- ✓ Don't do it. This app makes use of native device plugins that are loosely compatible with the web browser. You're setting yourself up for failure if you even attempt to open this application in a web browser.)

Using a Device or Simulator

- ✓ To test this application on your device or simulator, run the following in your Command Prompt or Terminal:
- ✓ ionic build android
- ✓ dab install -r platforms/android/ant-build/CordovaApp-debug.apk
- ✓ The above will get you going on Android. If you wish to track down errors and view the logs, you can use the Android Debug Bridge (ADB) like so:
- ✓ adblogcat

7. CONCLUSION

Ionic framework is the dominant HTML5 mobile development framework at present. You don't know how things will play out in this fast changing app development world, but ionic is well placed to go on its domination and become more and better in future. You've just made your own little photo repository. Is it the next Integra? Not in its current state, but you can definitely expand upon the material described in this paper. We took the base 64 image data obtained from the device camera and immediately pushed it into our Firebase storage. This storage can be synchronized between all of your devices and platforms.

REFERENCES:

[1] Sarah Perez (10 March 2014). "Drifty, Makers Of The Ionic Mobile Framework, Raise $1 Million". Retrieved 14 March 2015
[2] Retrieved 24 January 2017. "Software base".
[3] "Ionic Creator". Retrieved 3 February 2017.
[4] "Appery.io Builder for Ionic". Retrieved 3 February 2017.
[5] Christo Ananth, S.Silvia Rachel, E.Edinda Christy, K.Mala, "Probabilistic Framework for the Positioning Of a Vehicle in a Combined Indoor-Outdoor Scenario", International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE), Volume 2, Special Issue 13, March 2016, pp: 46-59
[6] "Ionic Documentation Overview, Browser Support". Retrieved 17 July 2015.