Kévin Jean Follow
May 23, 2017 · 4 min read

A Progressive Web Application with Vue JS, Webpack & Material Design [Part 2]

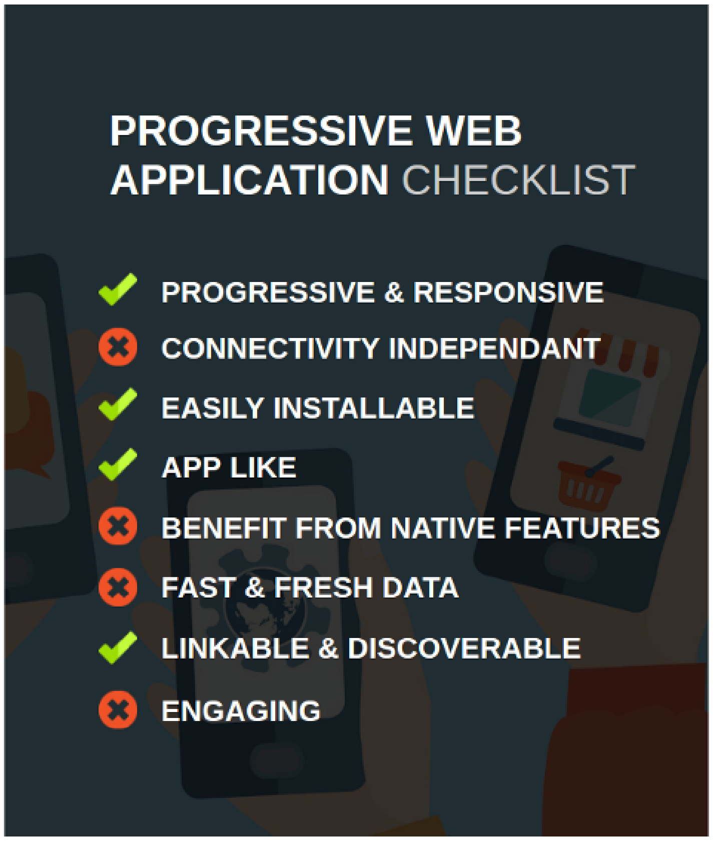
A Progressive Web App with VueJS, Webpack & Material Design
Part II
A realtime PWA with Firebase

[Updated on 10/30/2017]

This article is part of a serie that aims to build a basic but complete Progressive Web Application with VueJs, Webpack & Material Design, step-by-step and from scratch. If you have not yet read it, you can find the initial part here.

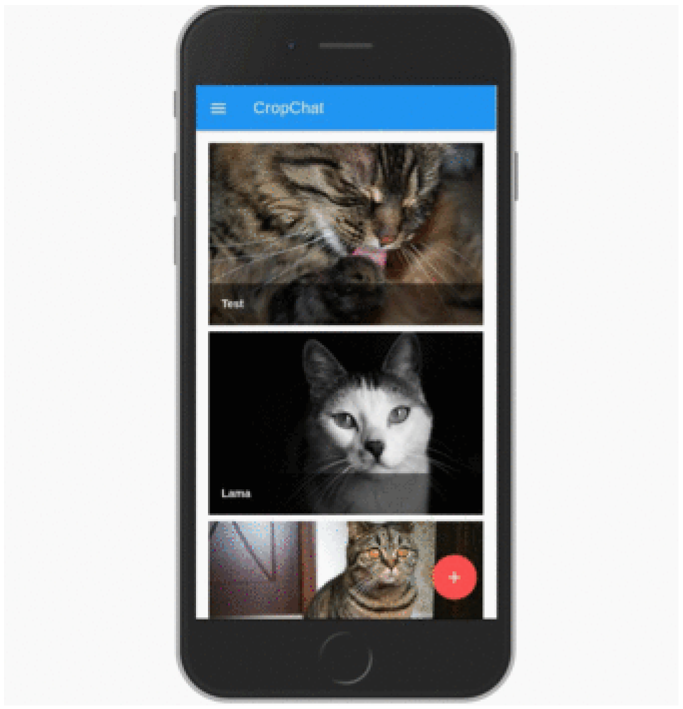
Code source available on this GitHub repository:
https://github.com/charlesBochet/vueJSPwa

In Part 1, we have learned how to create a single page application with VueJS, Webpack and MDL. We have already met first requirements of our PWA checklist:


PROGRESSIVE WEB APPLICATION CHECKLIST
✓ PROGRESSIVE & RESPONSIVE
✗ CONNECTIVITY INDEPENDANT
✓ EASILY INSTALLABLE
✓ APP LIKE
✗ BENEFIT FROM NATIVE FEATURES
✗ FAST & FRESH DATA
✓ LINKABLE & DISCOVERABLE
✗ ENGAGING

This second tutorial focuses on providing fast and fresh data for our CropChat application. We are going to:

- Setup a new Firebase database ;
- Connect our Vue App to Firebase with Vuefire ;
- Post and retrieve pictures from Firebase.



CropChat with fresh data

[PART 2] A Realtime PWA with Firebase

Firebase is a NoSQL Realtime Database developped by Google. It's Cloud hosted so you don't need to install anything on your server.

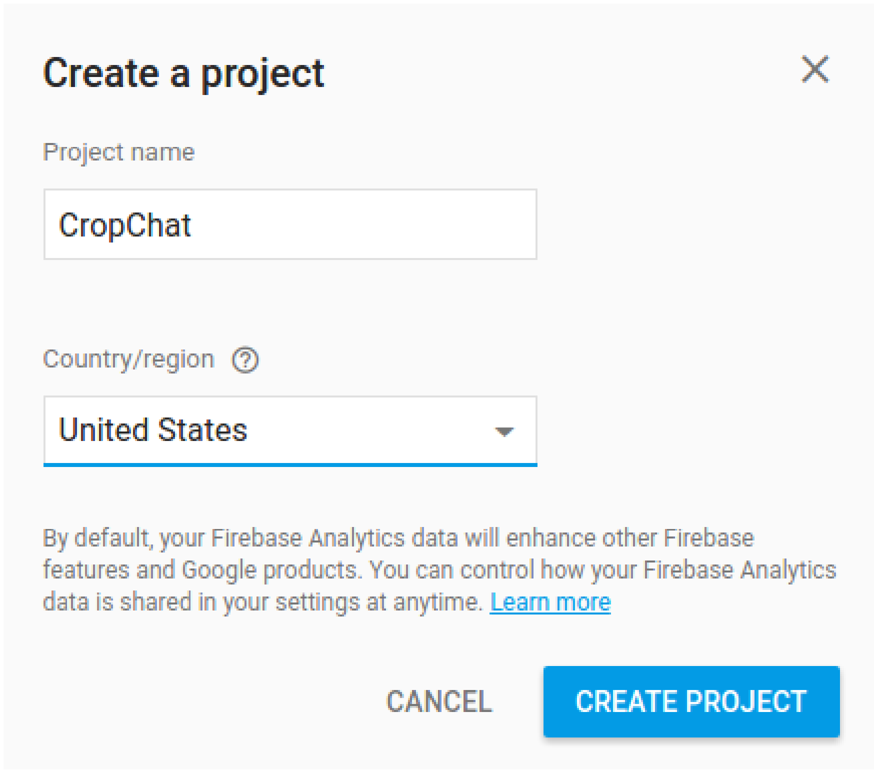
Data is synced across all clients in realtime. That makes Firebase the perfect tool to create a chat: it provides an easy way to keep a message feed up-to-date. Plus it's free.

To learn more about Firebase, please see the official page.

Let's start.

Configure firebase

First of all, go to the Firebase Console, sign up and create a new project:



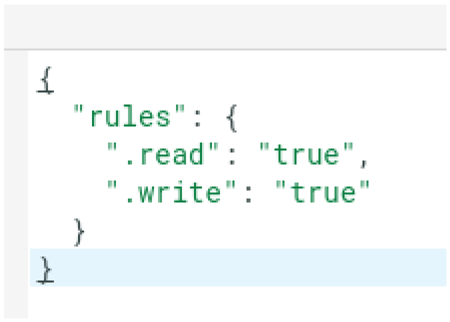
Create a project

Project name: CropChat

Country/region: United States

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime. Learn more

We also need to change the database rules :



```
{
  "rules": {
    ".read": "true",
    ".write": "true"
  }
}
```

Note: This makes your database readable and writable by everyone, only good for prototyping.

Add Firebase package to Cropchat:

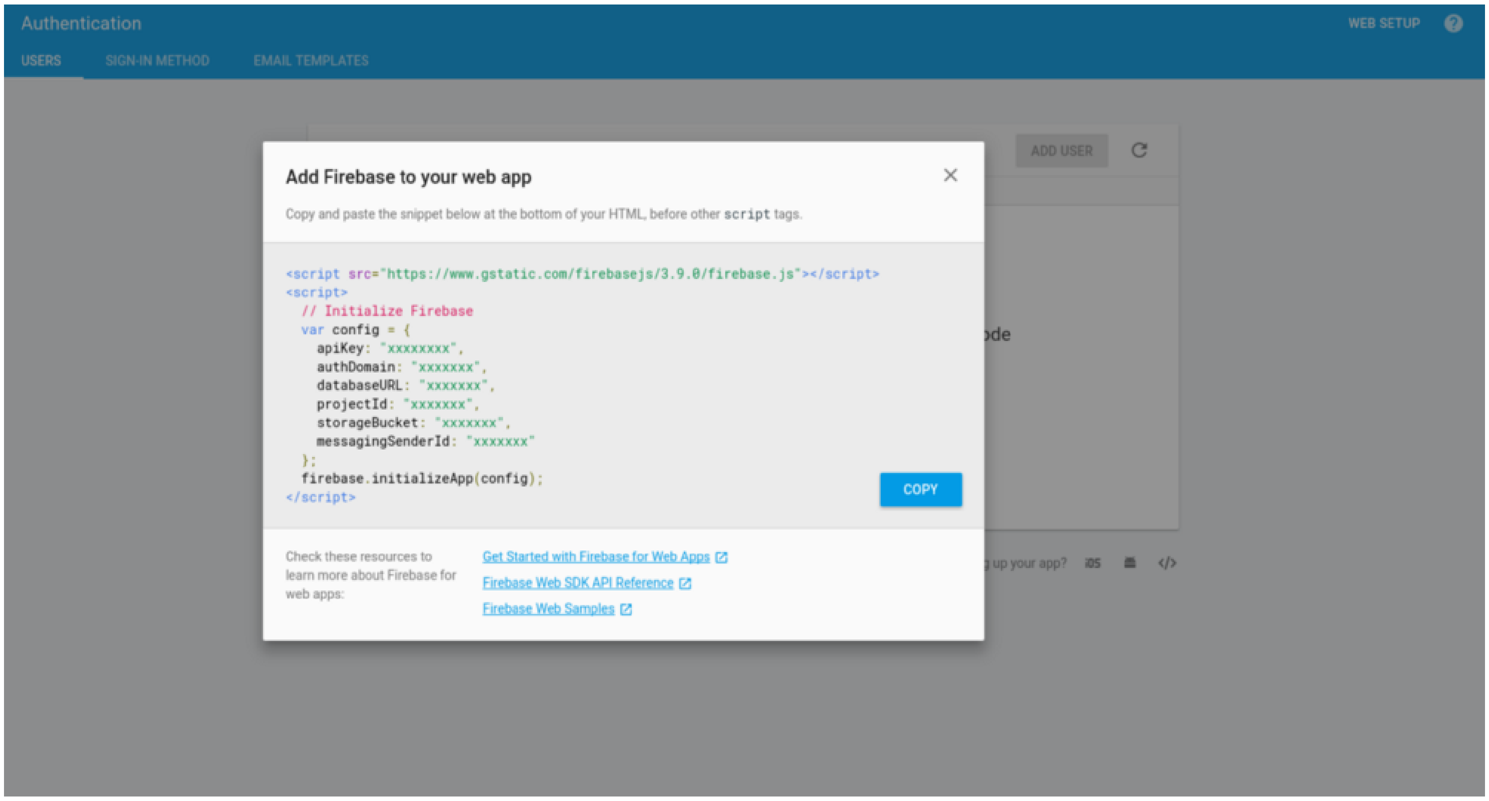
```
npm install firebase --save
```

Then we need to establish Firebase connection. Create a service `src/service/firebase.js` containing the following code:

```
import firebase from 'firebase'

var config = {
 apiKey: <Your api key here>,
 authDomain: <Your auth Domain here>,
 databaseURL: <Your databaseUrl here>,
 storageBucket: <Your storageBucket here>,
 messagingSenderId: <Your messagingSenderId here>
}
firebase.initializeApp(config)
```

Go back to the Firebase Console and select the Authentification tab. Click on “WEB SETUP” button to get the database information.


Authentication
USERS
SIGN-IN METHOD
EMAIL TEMPLATES
WEB SETUP
ADD USER
Add Firebase to your web app
Copy and paste the snippet below at the bottom of your HTML, before other script tags.
COPY
Check these resources to learn more about Firebase for web apps:
Get Started with Firebase for Web Apps
Firebase Web SDK API Reference
Firebase Web Samples

```
<script src="https://www.gstatic.com/firebasejs/3.9.0/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "xxxxxxxx",
    authDomain: "xxxxxxx",
    databaseURL: "xxxxxxx",
    projectId: "xxxxxxx",
    storageBucket: "xxxxxxx",
    messagingSenderId: "xxxxxxx"
  };
  firebase.initializeApp(config);
</script>
```

We are now connected to our database.

Vuefire: release the power of Firebase + VueJS

Configure VueFire

We will use VueFire, a node package built to wrap firebase hooks in a vueJS application.

Add VueFire to the project:

```
npm install vuefire --save
```

Import VueFire into our `src/main.js` :

```
import Vuefire from 'vuefire'

Vue.use(Vuefire)
```

Import the firebase service into `src/main.js` :

```
import firebase from './services/firebase'

new Vue({
 el: '#app',
 firebase: {
   cat: firebase.database.ref('cat').orderByChild('created_
 },
 router,
```

We just created a hook between our app prop `cats` and the firebase node `cats` . Vuefire will keep them synced for us. We add `.orderByChild('created_at')` to have the newest cat on the top of the feed.

Post a cat

We can start pushing to data on our firebase database. Add a form `src/components/PostView.vue` :

```
<template>
  <form>
    <div class="mdl-grid">
      <div class="mdl-cell mdl-cell--8-col">
        <div class="card-image__picture">
          <img :src="this.catUrl"/>
        </div>
      </div>
      <div class="mdl-cell mdl-cell--4-col mdl-cell--8-col-
        <div class="mdl-textfield mdl-js-textfield mdl-text
          <input id="username" v-model="title" type="text"
          <label for="username" class="mdl-textfield__label
        </div>
        <div class="actions">
          <a @click.prevent="postCat" class="mdl-button mdl
```

Add Vue-resource to have an HTTP client for our app and a xml parser :

```
npm install vue-resource --save
npm install xml-parser --save
```

Import Vue-resource into our `src/main.js` :

```
import VueResource from 'vue-resource'

Vue.use(VueResource)
```

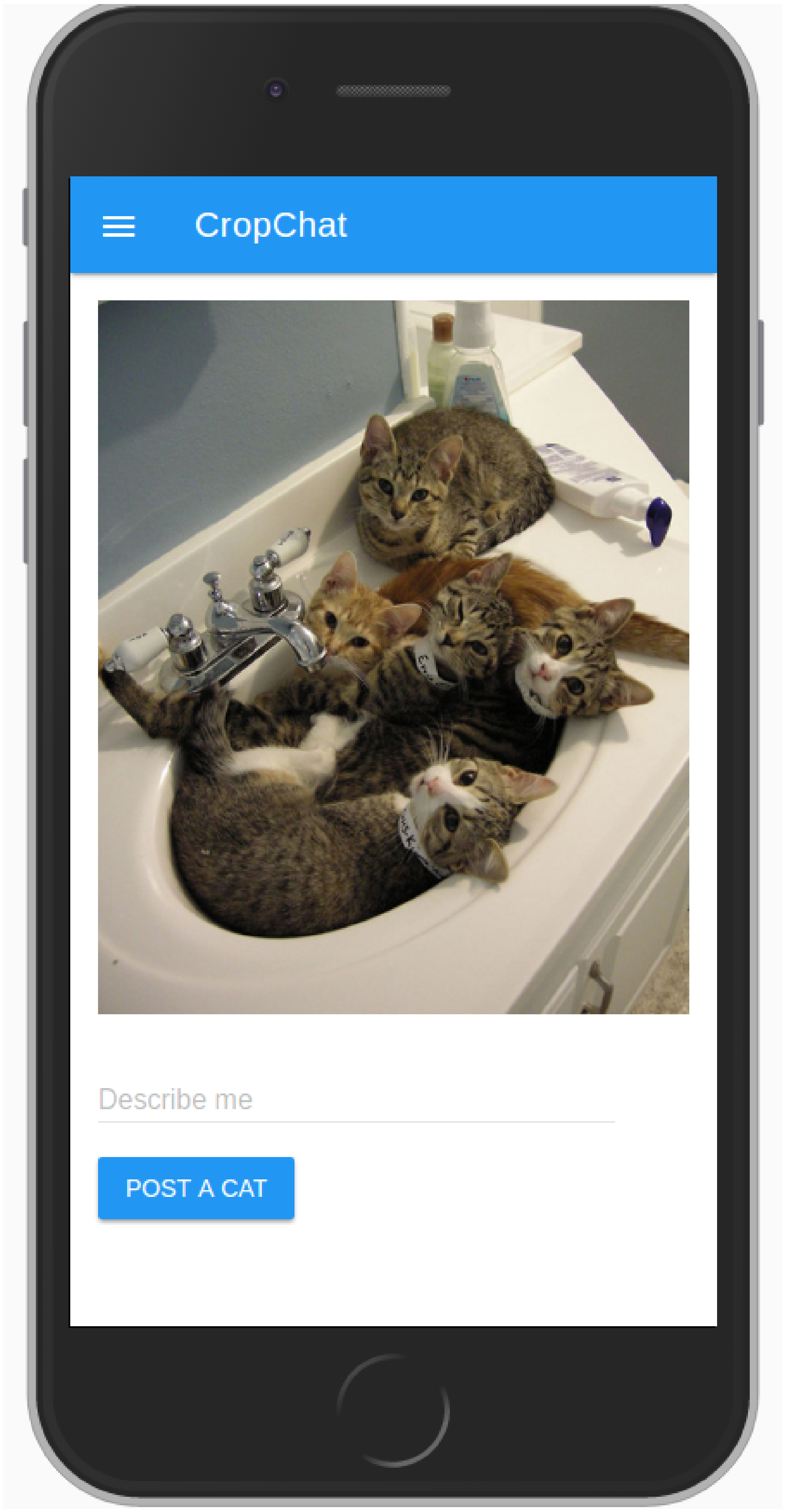
Update `PostView.vue` mounted() function to load a random cat from the CatAPI (which is a great place to find cat pictures by the way). This function is triggerd when PostView component is mounted:

```
<script>
  import parse from 'xml-parser'
  export default {
    data () {
      return {
        'catUrl': null
      }
    },
    mounted () {
      this.$http.get('https://thecatapi.com/api/images/get?
        this.catUrl = parse(response.body).root.children['0
      })
```

Finally, add postCat() function to `PostView.vue` methods that will push this image to Firebase instance:

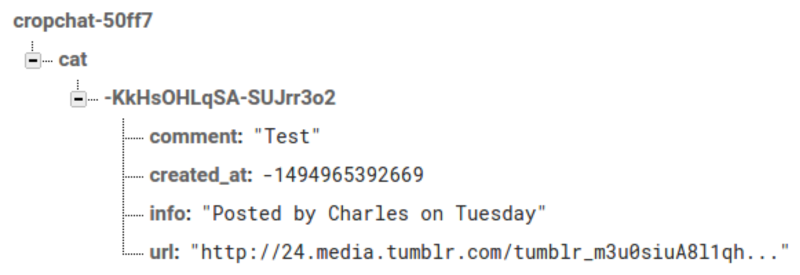
```
postCat () {
  this.$root.$firebaseRefs.cat.push(
    {
      'url': this.catUrl,
      'comment': this.title,
      'info': 'Posted by Charles on Tuesday',
      'created_at': -1 * new Date().getTime()
```

That’s it! Browse your application, click on the “Plus” button to access the Post View:



Screenshot from the Post view

And click on the“Post A Cat” button. You should see a new entry in your Firebase dashboard:



```
cropchat-50ff7
  cat
    -KkHsOHLqSA-SUJrr3o2
      comment: "Test"
      created_at: -1494965392669
      info: "Posted by Charles on Tuesday"
      url: "http://24.media.tumblr.com/tumblr_m3u0siuA8l1qh..."
```

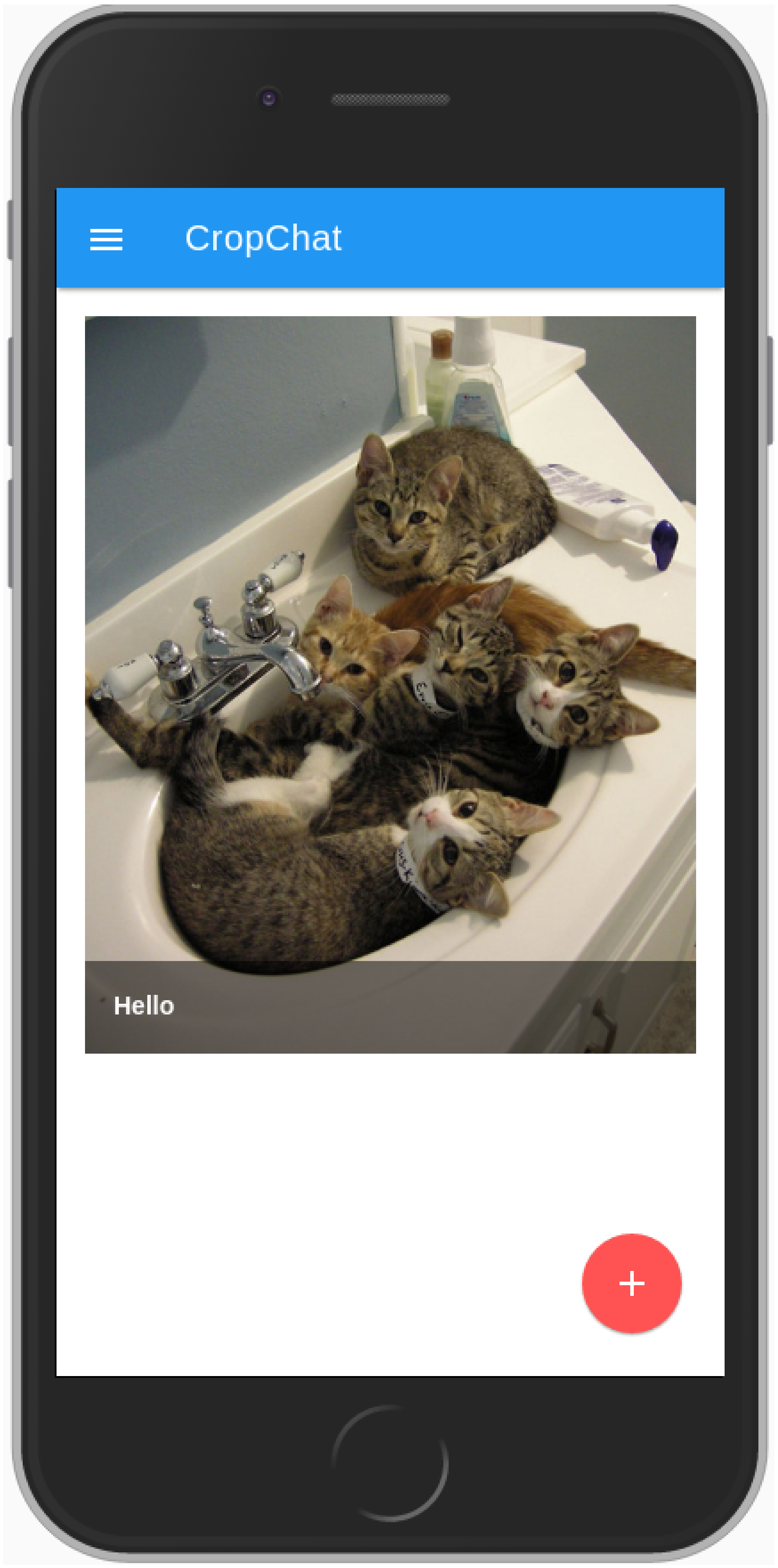
Display the cat list

We are almost done. We still have some work to do on Home View to display the cat images we stored in Firebase. Update the `HomeView.vue` template:

```
<div v-for="picture in this.$root.cat" class="image-card" @c
  <div class="image-card__picture">
    <img :src="picture.url" />
  </div>
  <div class="image-card__comment mdl-card__actions">
    <span>{{ picture.comment }}</span>
```

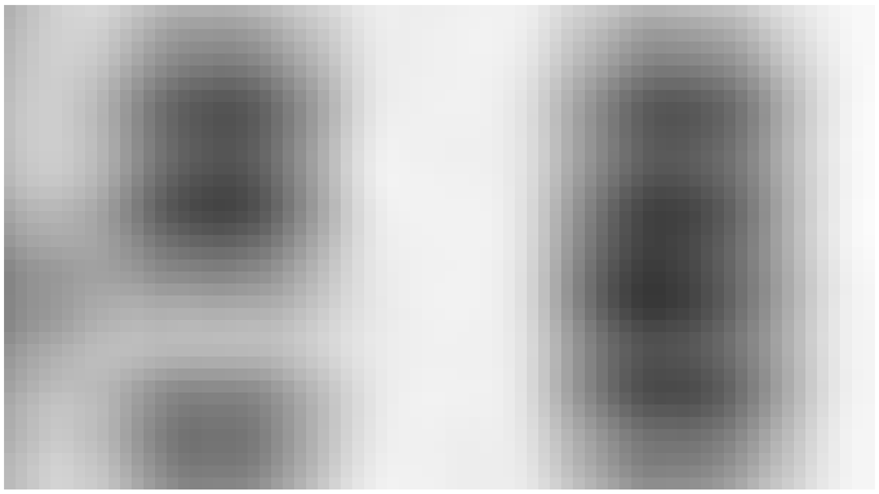
Do not forget to remove our static data file data.js and import line in `HomeView.vue` .

Here we go:



Home View is now synced on Firebase

Futermore, this is hot-synced. Open an second tab with the app, post a cat, and see the magic happening:


Hot-synced

Update the detail view

Add lodash:

```
npm install lodash --save
```

And update DetailView.vue:

```
<template>
  <div class="mdl-grid">
    <div class="mdl-cell mdl-cell--8-col">
      <div class="picture">
        <img :src="cat.url" />
      </div>
      <div class="info">
        <span>{{ cat.info }}</span>
      </div>
    </div>
    <div class="mdl-cell mdl-cell--4-col mdl-cell--8-col-ta
      <div class="comment">
        <span>{{ cat.comment }}</span>
      </div>
      <div class="actions">
        <router-link class="mdl-button mdl-js-button mdl-bu
          ANSWER
        </router-link>
      </div>
    </div>
  </div>
</template>
<script>
```

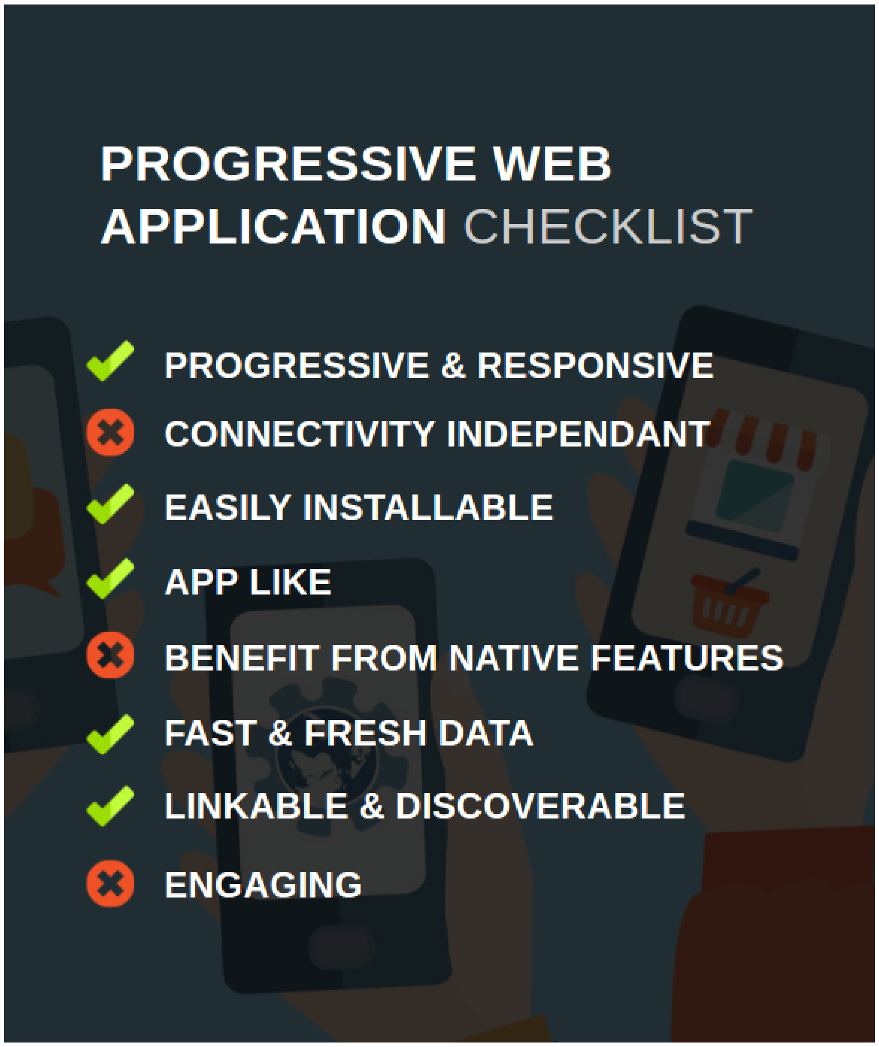
src/components/DetailView.vue

Conclusions

I hope I have convinced you of the capabilities of Firebase and VueJS to improve your app with fast and fresh datas. To cut a long story short:

- Vuefire offers an quick way to integrate a Firebase Database in your VueJS app ;
- Firebase is a very powerfull tool for creating a real time data in an app, and make sure that your users always have fresh and up-to-date data.

We checked a new point on our PWA requirements checklist:



PROGRESSIVE WEB APPLICATION CHECKLIST

- ✓ PROGRESSIVE & RESPONSIVE
- ✗ CONNECTIVITY INDEPENDANT
- ✓ EASILY INSTALLABLE
- ✓ APP LIKE
- ✗ BENEFIT FROM NATIVE FEATURES
- ✓ FAST & FRESH DATA
- ✓ LINKABLE & DISCOVERABLE
- ✗ ENGAGING

Not yet done! In Part III, we will learn how to make our Cropchat app work offline thanks to Service Workers.