Book Database - Using VueJS & Firebase

Setting Up Firebase Account & Database

- ❏ Visit: https://console.firebase.google.com/
- ❏ Click “database” and enter in new info as follows:

- ❏ Under “Rules”, change code as follows to allow all users access for now:

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

Setting Up VueJS App

- ❏ Create new VueJs project using: `vue init webpack vue-firebase-demo`
  `cd vue-firebase-demo`
  `npm run dev`
- ❏ Installing other dependencies:
  `npm install firebase vuefire --save`

Connecting VueJS App to Firebase

- ❏ Inside src/main.js, type in: `import VueFire from 'vuefire'` after importing Vue.
- ❏ Then this too: `Vue.use(VueFire)` after all import codes.
- ❏ Visit the “Project Overview” section at https://console.firebase.google.com/ and click “Add Firebase to your web app”.
  Copy the `config` section of the code and copy into `App.vue` immediately after the opening of `<script>`.
- ❏ Import Firebase for use inside *App.vue* file: `import Firebase from "firebase";`

Displaying Data from Firebase

- After the `let config` statement,

```
let app = Firebase.initializeApp(config);
let db = app.database();
let bookRef = db.ref("books");
export default {
 name: "App",
 firebase: {
   books: bookRef
 }
};
```

The code above simply initializes the app with the data from Firebase.

```
<script>
import Firebase from "firebase";

let config = {
  apiKey: "AIzaSyDwl8v4nLfJZC5h4Cja-o4OGZqFO9R_J2Y",
  authDomain: "vuejs-firebase-demo-97699.firebaseapp.com",
  databaseURL: "https://vuejs-firebase-demo-97699.firebaseio.com",
  projectId: "vuejs-firebase-demo-97699",
  storageBucket: "vuejs-firebase-demo-97699.appspot.com",
  messagingSenderId: "600367553731"
};

let app = Firebase.initializeApp(config);
let db = app.database();

let booksRef = db.ref("books");

export default {
  name: "App",
  firebase: {
    books: booksRef
  }
};
</script>
```

- Display data in table in web browser:

```
<tbody>
<tr v-for="book in books" :key="book.title">
 <td>
  <a v-bind:href="book.url" target="_blank">{{book.title}}</a>
 </td>
 <td>{{book.author}}</td>
</tr>
</tbody>
```

Enter in new Data into the Firebase database

-

```
data() {
   return {
     newBook: {
       title: "",
```

```
        author: "",
        url: "",
      }
    };
  },
```

- ❏ Create the method “addBook” for the form submission:

```
methods: {
   addBook: function() {
     booksRef.push(this.newBook);
     this.newBook.title = '';
     this.newBook.author = '';
     this.newBook.url = '';
   }
 }
```

- ❏ Then type in the HTML form input as below:

```
<div class="panel-body">
  <form id="form" class="form-inline" @submit.prevent="addBook">
    <div class="form-group">
      <label for="bookTitle">Title:</label>
      <input type="text" id="bookTitle" class="form-control" v-model="newBook.title">
    </div>
    <div class="form-group">
      <label for="bookAuthor">Author:</label>
      <input type="text" id="bookAuthor" class="form-control" v-model="newBook.author">
    </div>
    <div class="form-group">
      <label for="bookUrl">URL:</label>
      <input type="text" id="bookUrl" class="form-control" v-model="newBook.url">
    </div>
    <input type="submit" class="btn btn-primary" value="Add Book">
  </form>
</div>
```

Deleting Data from the Database and UI Table

- ❏ Create extra space in the table for delete button:

```
<td>
  <span @click="removeBook(book)"><i class="fas fa-trash-alt"></i></span>
</td>
```

- ❏ Create new method “`removeBook`” that accepts parameter `(book)` inside the script section:

```
removeBook: function(book) {
  booksRef.child(book['.key']).remove();
}
```

This removes that particular row of data from the `booksRef` object.

Create Notification on Deletion of Data

- Install toastr: `npm install toastr --save`
- Link to Toastr CDN: `<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.css">`
- Import Toastr inside of `<script>` of App.vue: `import Toastr from 'toastr';`
- Lastly, make use of Toastr instance where needed as below:

```
removeBook: function(book) {
  booksRef.child(book['.key']).remove();
  Toastr.success("Book has been removed.");
}
```

Show Dialog Box before Deletion of data

Reference: https://www.npmjs.com/package/vuejs-dialog

- Install VueJS Dialog: `npm install vuejs-dialog --save`
- Inside src/main.js, `import VueDialog from 'vuejs-dialog';`
  Then: `Vue.use(VueDialog)`
- Inside App.vue:

```
let message = "Are you sure?";
let options = {
 html: false, // set to true if your message contains HTML tags. eg: "Delete <b>Foo</b> ?"
 reverse: false, // switch the button positions (left to right, and vise versa)
 okText: "Continue",
 cancelText: "Cancel",
 backdropClose: true // set to true to close the dialog when clicking outside of the dialog window, i.e. click landing on the mask
};
```

- Then `remove()` function is changed to:

```
removeBook: function(book) {
    this.$dialog
      .confirm(message, options)
      .then(function() {
        booksRef.child(book[".key"]).remove();
        Toastr.success("Book has been removed.");
      })
      .catch(function() {
        Toastr.success("Whoops! Let's be careful next time.");
      });
  }
```