Realtor Vue Front End

1. **Create a vue project from scratch**

   ```
   vue create vue-front-end
   ```

   Walk the students through the project. Be sure to add vuex or we can add it after the fact using vue add vuex

2. **Walk through the design we want to achieve. To start we want a single page to represent the homepage: TheHome.vue**

   We will want the following components

   TheHeader.vue - The header will contain our Navbar and logo (next week we learn routing)

   HomeSearch.vue - Searchable List of Homes (list filter by zip code)

   TheFooter.vue - Basic copyright text

3. **Create TheHeader.vue file and add it to the App.vue**

TheHeader.vue:

```
<template>
  <div>
     <p>{{pageTitle}}</p>
  </div>
</template>

<script>
export default {
   name: 'the-header',
   data() {
      return {
         pageTitle: 'Java Green Home Search'
      };
   }
}
</script>

<style>

</style>
```

App.vue

```
<template>
  <div id="app">
   <the-header></the-header>
  </div>
</template>

<script>

import TheHeader from './components/TheHeader'

export default {


  name: 'App',
  components: {
     TheHeader
  }
}
</script>
```

4. **Let's add an image to the header…**

```
<img src="../assets/java-green.png"/>

img {
   width: 15%;
   height: auto;
}
```

5. **Let's add a nav bar…**

```
<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Search Homes</a>
</nav>
```

6. Let's add some CSS styling:

```
nav {
  display: flex;
  justify-content: center;
  flex-wrap: wrap;
}
nav a {
  text-decoration: none;
  display: block;
  padding: 15px 25px;
  text-align: center;
  background-color: rgb(4, 245, 85);
  color: #03722e;
  margin: 0;
  font-family: sans-serif;
}
nav a:hover {
  background-color: #59ec96;
  color: #ffffff;
}
```

7. Now lets go work on the Footer a second…..

```
<template>
  <div>
     <p>&#169; {{copyText}}</p>
  </div>
</template>

<script>
export default {
   name: 'the-footer',
   data() {
      return {
         copyText: 'Java Green Enterprises LLC. All rights reserved'
      };
   }
}
</script>

<style>

</style>
```

8. Now we need to add the footer to the App.vue

```
<the-footer></the-footer>

import TheFooter from './components/TheFooter'

TheFooter
```

9. Now let's go work on the HomeSearch.vue component

```
<template>
  <div>
    <p> Placeholder text </p>
  <div>
</template>

<script>
export default {
   name: 'home-search',
   data() {
      return {};
   }
}
</script>

<style scoped>

</style>>
```

10. Now let's add this to App.vue …

```
<home-search></home-search>

import HomeSearch from './components/HomeSearch'

HomeSearch
```

11. Now let's go to HomeSearch.vue and let's set up the template for our main search results.

(a) Go to: https://divtable.com/table-styler/ and pick out a style we like, adjust colors, etc
(b) Add the CSS and HTML code to the component.
(c) Add the HTML for the search box… place it in its own div

```
<div id="search">

    <label for="zip">Enter Your Zip Code To Find Your Next Dream Home:</label>
    <input name="zip" type="text" />

</div>
```

(d) Adjust CSS as necessary

```
#main-div {
   margin: 30px;
}

#search {
  margin: 30px;
}
```

If needed, you can center the div containing the table using margin...:

```
div.minimalistBlack {
    margin: auto;
```

Let’s clean up the text box

```
input[type=text] {
  margin: 30px;
  width: 15%;
  padding: 12px 20px;
  box-sizing: border-box;
  border: 2px solid green;
  border-radius: 6px;
}
```

12. Now let's set up our data in the Vuex store. Let's start with the Homes array (Next week we will get this from a web service)

```
  homes: [
    {
      mlsId: '1000',
      address: '123 Java Green Lane',
      city: 'Columbus',
      state: 'Ohio',
      zip: '43023',
      price: '1,222,345.00',
      imageName: '1000.jpg'
    },
    {
      mlsId: '1001',
      address: '123 Vue Street',
      city: 'Grandview',
      state: 'Ohio',
      zip: '43015',
      price: '952,345.72',
      imageName: '1001.jpg'
    },
    {
      mlsId: '1002',
      address: '123 Java Blue Court',
      city: 'Columbus',
      state: 'Ohio',
      zip: '43023',
      price: '750,000.00',
      imageName: '1002.jpg'
    },
    {
      mlsId: '1003',
      address: '999 C-Sharp Rd.',
      city: 'Dublin',
      state: 'Ohio',
      zip: '43017',
      price: '99.97',
      imageName: '1003.jpg'
    },
    {
      mlsId: '1004',
      address: '555 Cohort Lane. Apt. 1',
```

```
        city: 'Columbus',
        state: 'Ohio',
        zip: '43022',
        price: '1,000,000.01',
        imageName: '1004.jpg'
      }
  ],
```

13. Now let's add a data property to hold the default value for all homes.. And use that in a computed property to filter out the homes:

```
data() {
  return {
     zipFilter: ''
  };
},

computed: {
  filteredHomes() {
    const homes = this.$store.state.homes;
    return homes.filter(home => {
      return zipFilter ==='' ? true : zipFilter === home.zip;
    });
  }
}
```

14. Now let's go back to the template section and update the div containing the row data..

```
<div class="divTableRow" v-for="home in filteredHomes" v-bind:key="home.id">
```

Walk through the v-for logic… and the v-bind-key

15. Let's start to fill in the data and make sure we can see the rows populated:

(a) Let's put the image in the first column

```
<img v-bind:src=home.imageName />
```

But this gives us a broken link for the image, so we have to go back to the store and fix the image links:

```
imageName: require('../assets/1000.jpg')
```
(file paths need to be wrapped in a require function)

16. Fill in the rest of the data (you will need to add additional columns and adjust the CSS

```
<div class="divTable minimalistBlack">
  <div class="divTableHeading">
    <div class="divTableRow">
      <div class="divTableHead"></div>
       <div class="divTableHead">MLS Number</div>
      <div class="divTableHead">Address</div>
      <div class="divTableHead">City</div>
      <div class="divTableHead">State</div>
      <div class="divTableHead">Zip</div>
      <div class="divTableHead">Price</div>
    </div>
  </div>
  <div class="divTableBody">
    <div class="divTableRow" v-for="home in filteredHomes" v-bind:key="home.id">
      <div class="divTableCell">
          <img v-bind:src=home.imageName />
      </div>
      <div class="divTableCell">{{home.mlsId}}</div>
      <div class="divTableCell">{{home.address}}</div>
      <div class="divTableCell">{{home.city}}</div>
       <div class="divTableCell">{{home.state}}</div>
      <div class="divTableCell">{{home.zip}}</div>
      <div class="divTableCell">${{home.price}}</div>
    </div>
  </div>
</div>
```

17. Fix any CSS …

```
img {
  width: 150px;
  height: auto;
}
```

```
div.minimalistBlack {
  margin: auto;
  border: 2px solid #000000;
  width: 80%;                          ← adjusted
  border-collapse: collapse;
}


.divTable {
  display: table;
  table-layout: fixed;          ←    added
}
```

18. Ok, let's make this searchable….We need to add a v-model to the input box…

```
<input type="text" v-model="zipFilter" />
```

The results should now update… Try 43023, 43015, 43017

19. What if we wanted to NOT show the empty table if there are no results? Ask the students what could we try?

Well, we shouldn't add a v-if to the v-for because the v-for will have a higher priority, so it will render an empty table… What we can do, is add it to the parent div and test for filteredHomes.length.

```
<div class="divTable minimalistBlack" v-if="filteredHomes.length">
```

Now the table should disappear….