

Vue Player Tracker - Java Blue

Players:

https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190470/1_cxwinb.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190470/8_sbqriz.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/7_omvz9t.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/5_dm97cw.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/4_ec3vzk.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/2_hqe0ky.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/6_mdkfoj.webp
https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/3_mgphn7.webp

Mystery Player

https://res.cloudinary.com/drr6cwmiz/image/upload/v1701191329/mystery_exnajf.webp

Logo

-white https://res.cloudinary.com/drr6cwmiz/image/upload/v1711552221/logo_ybymjr.png
-transparent -
https://res.cloudinary.com/drr6cwmiz/image/upload/v1711554775/logo-transparent_mj2yng.png
https://res.cloudinary.com/drr6cwmiz/image/upload/v1711552221/java-bluebirds-main_xabqpf.png

STEP 1: Let's build a site header component

```
<template>
  <div>
    <h1 id="siteTitle">{{title}}</h1>
    <nav>
      <a href="#">Home</a> |
      <a href="#">About</a> |
      <a href="#">Search</a>
    </nav>
  </div>
</template>

<script>
export default {
```

```
    data() {  
      return {  
        title: 'Java BlueBirds Baseball'  
      }  
    }  
  }  
</script>  
  
<style scoped>  
  #siteTitle {  
    text-align: center;  
  }  
</style>
```

STEP 2: Add the header component to App.vue

```
<template>  
  <div class="main">  
    <the-header></the-header>  
    <h1>placeholder</h1>  
  </div>  
</template>  
  
<script>  
  
import TheHeader from './components/TheHeader.vue';  
  
export default {  
  
  components: {  
    TheHeader  
  }  
}
```

```
};  
</script>  
  
<style>  
.main {  
  font-family: "Avenir", Helvetica, Arial, sans-serif;  
  -webkit-font-smoothing: antialiased;  
  -moz-osx-font-smoothing: grayscale;  
  text-align: center;  
  color: #2c3e50;  
  max-width: 800px;  
  margin: 60px auto 0 auto;  
}  
</style>
```

After adding the component, we should have the Header added to the main page

STEP 3: Add a logo and clean up the header

In TheHeader.vue, add the following code just before the h1:

```

```

Upon saving, we can see the image is WAY too big. Let's fix it through CSS

```
img {  
  width: 10%;  
  height:auto;  
}
```

STEP 4: Let's align the header horizontally

Add an id attribute to the div and call it container and add the following CSS:

```
#container {  
  display: flex;  
  justify-content: space-around;  
  align-items: center;  
  width: 100%;  
  border-bottom: 10px solid rgb(13, 21, 139);  
  background-color: cornflowerblue;  
}
```

STEP 5: Let's make the links look like buttons

```
nav a {  
  margin: 5px;  
  padding: 13px 25px;  
  background-color: rgb(13, 21, 139);  
  color: white;  
  border-radius: 6px;  
}
```

STEP 6: Let's clean up the footer

```
#container {  
  border-top: 10px solid rgb(13, 21, 139);  
  background-color: cornflowerblue;  
  color: white;  
  margin-bottom: 10px;  
}  
  
#copyright {  
  text-align: center;  
  padding-bottom: 10px;  
}
```

STEP 7: Let's set up a Home page and basic routing

Create a HomeView.vue file...in the Views folder

```
<template>  
  <div>  
    <h1>Site home page</h1>  
  </div>  
</template>  
  
<script>  
export default {
```

```
}  
</script>  
  
<style scoped>  
  
</style>
```

Go to App.vue and add a router view element

```
<template>  
  <div class="main">  
    <the-header />  
    <router-view />  
    <the-footer />  
  </div>  
</template>
```

Go to router config file (index.js) and add a route entry (don't forget to import it)

```
import HomeView from '../views/HomeView.vue';
```

```
{  
  path: '/',  
  name: 'home',  
  component: HomeView,  
  meta: {  
    requiresAuth: true  
  }  
,  
},
```

App should now load the home page by default

STEP 8: Let's add a background image for our home page with text overlay

```
<div class="big-image">
  <div class="overlay">
    <h1>Java Bluebirds Baseball</h1>
    <p>Beating .NET is like taking candy from a baby!</p>
  </div>
</div>
```

```
.big-image {
  height: 100vh;
  width: 100vw;
  position: relative;
  background-size: cover;
  background-position: 50% 50%;
  background-image:
url('https://res.cloudinary.com/drr6cwmiz/image/upload/v1711552221/java-
bluebirds-main_xabqpf.png');
}
```

```
.overlay {
  position: absolute;
  height: 100%;
  width: 100%;
  top: 0;
  left: 0;
  background: rgb(0,0,0,0.65);
  color: white;
  display: flex;
  align-items: center;
  justify-content: center;
  flex-direction: column;
}
```

STEP 9: Add an about page

Create an AboutView.vue file and then add the route to the router

```
<template>
  <div>
    <h1>About Java Bluebirds Baseball</h1>
    <p>Random stuff would go here</p>
  </div>
</template>

<script>
export default {
}
</script>

<style scoped>

</style>
```

```
{
  path: '/about',
  name: 'about',
  component: AboutView
},
```


STEP 10: Add router links to the navigation to route between pages

```
<router-link v-bind:to="{ name: 'home' }">Home</router-link>  
<router-link v-bind:to="{ name: 'about' }">About</router-link>  
<router-link v-bind:to="{ name: 'home' }">Search Players</router-link>
```

Pages should now route accordingly

STEP 11: Adding a search page (view and component)

Create a component called **PlayerSeach.vue**

```
<template>  
  <div>  
    <p>player search page placeholder</p>  
  </div>  
</template>  
  
<script>  
export default {  
  
  data() {  
    return {  
  
    }  
  }  
}
```

```
}  
</script>  
  
<style scoped>  
  
</style>
```

Create a view page called: PlayerView.vue. And add the component to it

```
<template>  
  <player-search/>  
</template>  
  
<script>  
  
import PlayerSearch from '../components/PlayerSearch.vue';  
  
export default {  
  
  components: {  
    PlayerSearch  
  }  
}  
</script>  
  
<style scoped>  
  
</style>
```

STEP 12: Hook up Search Page to Router

In the router config file (index.js), add a route to the new search page

```
import PlayerView from '../views/PlayerView.vue';
```

```
{  
  path: '/search',  
  name: 'search',  
  component: PlayerView  
}
```

STEP 13: update the router link in the header

```
<router-link v-bind:to="{ name: 'search' }">Search Players</router-link>
```

STEP 14: Add temporary data to the Vuex Store.
(This will come from web service on another day)

```
players: [
  {
    "playerId": 1,
    "firstName": "Derek",
    "lastName": "Jibberish",
    "jerseyNumber": 2,
    "salary": 220000.0,
    "positions": [
      "Pitcher"
    ],
    "teamId": 1,
    "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190470/1_cxwinb.
webp",
    "fullName": "Derek Jibberish"
  },
  {
    "playerId": 2,
    "firstName": "Baxter",
    "lastName": "Bubbleworth",
    "jerseyNumber": 42,
    "salary": 150000.0,
    "positions": [
      "Catcher"
    ],
    "teamId": 1,
    "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/2_hqe0ky.
webp",
    "fullName": "Baxter Bubbleworth"
  },
  {
    "playerId": 3,
    "firstName": "Jasper",
    "lastName": "Jollyjinks",
    "jerseyNumber": 13,
    "salary": 250000.0,
    "positions": [
      "First Baseman"
```

```
    ],
    "teamId": 1,
    "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/3_mgphn7.
webp",
    "fullName": "Jasper Jollyjinks"
  },
  {
    "playerId": 4,
    "firstName": "Ziggy",
    "lastName": "Zipperton",
    "jerseyNumber": 3,
    "salary": 10000.0,
    "positions": [
      "Second Baseman"
    ],
    "teamId": 1,
    "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/4_ec3vzk.
webp",
    "fullName": "Ziggy Zipperton"
  },
  {
    "playerId": 5,
    "firstName": "Fiona",
    "lastName": "Fumblefizz",
    "jerseyNumber": 4,
    "salary": 430000.0,
    "positions": [
      "Shortstop"
    ],
    "teamId": 1,
    "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/5_dm97cw.
webp",
    "fullName": "Fiona Fumblefizz"
  },
  {
    "playerId": 6,
```

```
        "firstName": "Dexter",
        "lastName": "Doodlehatch",
        "jerseyNumber": 14,
        "salary": 84300.0,
        "positions": [
            "Pitcher"
        ],
        "teamId": 1,
        "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/6_mdkfoj.
webp",
        "fullName": "Dexter Doodlehatch"
    },
    {
        "playerId": 7,
        "firstName": "Judy",
        "lastName": "Wipsprocket",
        "jerseyNumber": 19,
        "salary": 230000.0,
        "positions": [
            "Shortstop"
        ],
        "teamId": 1,
        "imageUrl":
"https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190469/7_omvz9t.
webp",
        "fullName": "Judy Wipsprocket"
    },
    {
        "playerId": 8,
        "firstName": "Gilbert",
        "lastName": "Gigglesnort",
        "jerseyNumber": 29,
        "salary": 130000.0,
        "positions": [
            "Shortstop"
        ],
        "teamId": 1,
```

```
        "imageUrl":  
        "https://res.cloudinary.com/drr6cwmiz/image/upload/v1701190470/8_sbqriz.  
webp",  
        "fullName": "Gilbert Gigglesnort"  
    }  
]
```

STEP 15: Build and style the basic data table skeleton

<https://divtable.com/table-styler/>

```
<div class="divTable paleBlueRows">  
  <div class="divTableHeading">  
    <div class="divTableRow">  
      <div class="divTableHead">head1</div>  
      <div class="divTableHead">head2</div>  
      <div class="divTableHead">head3</div>  
      <div class="divTableHead">head4</div>  
      <div class="divTableHead">head5</div>  
      <div class="divTableHead">head6</div>  
    </div>  
  </div>  
  <div class="divTableBody">  
    <div class="divTableRow">  
      <div class="divTableCell">cell11_1</div>  
      <div class="divTableCell">cell12_1</div>  
      <div class="divTableCell">cell13_1</div>  
      <div class="divTableCell">cell14_1</div>  
      <div class="divTableCell">cell15_1</div>  
      <div class="divTableCell">cell16_1</div>  
    </div>  
  </div>  
</div>
```

```
div.paleBlueRows {
    font-family: Tahoma, Geneva, sans-serif;
    border: 1px solid #FFFFFF;
    background-color: #4BC5EE;
    text-align: left;
}

.divTable.paleBlueRows .divTableCell,
.divTable.paleBlueRows .divTableHead {
    border: 1px solid #FFFFFF;
    padding: 2px 2px;
}

.divTable.paleBlueRows .divTableBody .divTableCell {
    font-size: 13px;
}

.divTable.paleBlueRows .divTableRow:nth-child(even) {
    background: #D0E4F5;
}

.divTable.paleBlueRows .divTableHeading {
    background: #0B6FA4;
    border-bottom: 5px solid #FFFFFF;
}

.divTable.paleBlueRows .divTableHeading .divTableHead {
    font-size: 17px;
    font-weight: bold;
    color: #FFFFFF;
    text-align: center;
    border-left: 2px solid #FFFFFF;
}
```



```
.divTable.paleBlueRows .divTableHeading .divTableHead:first-child {
    border-left: none;
}

.paleBlueRows .tableFootStyle {
    font-size: 14px;
}

/* DivTable.com */
.divTable {
    display: table;
}

.divTableRow {
    display: table-row;
}

.divTableHeading {
    display: table-header-group;
}

.divTableCell,
.divTableHead {
    display: table-cell;
}

.divTableHeading {
    display: table-header-group;
}

.divTableFoot {
    display: table-footer-group;
}

.divTableBody {
    display: table-row-group;
}
```

STEP 16: Add a computed property to pull data

```
computed: {  
  filteredPlayers() {  
    const players = this.$store.state.players;  
    //todo add filtering later  
    return players;  
  }  
}
```

STEP 17: Add a v-for to display the correct number of rows

```
<div class="divTableBody">  
  <div class="divTableRow" v-for="player in  
filteredPlayers" v-bind:key="player.playerId">  
    <div class="divTableCell">cell1_1</div>  
    <div class="divTableCell">cell2_1</div>  
    <div class="divTableCell">cell3_1</div>  
    <div class="divTableCell">cell4_1</div>  
    <div class="divTableCell">cell5_1</div>  
    <div class="divTableCell">cell6_1</div>  
  </div>  
</div>
```

STEP 18: Update the column headers and bind the data

```
<div class="divTableRow">
```

```
<div class="divTableHead">Picture</div>
<div class="divTableHead">Jersey Number</div>
<div class="divTableHead">Full Name</div>
<div class="divTableHead">Position</div>
<div class="divTableHead">Salary</div>
<div class="divTableHead">Stats</div>
</div>
```

```
<div class="divTableBody">
  <div class="divTableRow" v-for="player in filteredPlayers"
v-bind:key="player.playerId">
    <div class="divTableCell">
      
    </div>
    <div class="divTableCell">{{ player.jerseyNumber }}</div>
    <div class="divTableCell">{{ player.fullName }}</div>
    <div class="divTableCell">{{ player.positions[0] }}</div>
    <div class="divTableCell">{{ player.salary }}</div>
    <div class="divTableCell">coming soon</div>
  </div>
</div>
```

```
#thumbnail {
  width: 100px;
  height: auto;
}
```

STEP 19: Add a search box and filter to filter the player list

First, we need to add a data property to hold the value in the search

```
data() {  
  return {  
    nameFilter: ""  
  }  
},
```

Next, we add the form to the template:

```
<div id="searchPlayer">  
  <label for="">Search Player</label>  
  <input type="text" name="playerName" v-model="nameFilter" />  
</div>
```

After testing to see if the data is being two-way data-bound in the Vue tools extension, we now modify the `filteredPlayers` function to use the `nameFilter`

```
filteredPlayers() {  
  const players = this.$store.state.players;  
  
  return players.filter( player => {  
    return this.nameFilter == '' ? true :  
player.fullName.includes(this.nameFilter);  
  })  
}
```

STEP 20: Final CSS clean up

```
input[type=text] {  
  margin: 10px;  
  width: 20%;  
  padding: 8px 16px;  
  border: 2px solid green;  
  border-radius: 5px;  
}
```