Lab Exercises

Web Development Frameworks

zenika
ARCHITECTURE INFORMATIQUE

Lab 3: Vue Todo App

In this lab, we will explore building a Todo app using Vue!

Create a new Vue project

Install the Vue CLI. In you terminal run,

```
npm install -g @vue/cli
```

Once it's installed, run

```
vue create vue-todo-app
```

Select `Default (Vue 3) ([Vue 3] babel, eslint)` when prompted.

Change directory into `vue-todo-app` folder and run this command,

```
yarn serve
```

Open `http://localhost:8080/` in your browser and you should see Vue's welcome page,

Create a new Vue component

Create a new file called `Todos.vue` in `src/components` folder and paste the folowing code as the UI mockup.

```
<template>
  <main>
    <section class="todo-wrapper">
      <h2 class="todo-title">My Todo List</h2>
      <input type="text" />
      <div class="btn btn-add">+</div>
      <ul class="todo-list">
        <!-- list of todos here -->
      </ul>
    </section>
  </main>
</template>
```

```
<script>
export default {
  name: "Todos"
};
</script>

<style>
html,
body {
  margin: 0;
  height: 100%;
  font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto,
    Oxygen-Sans, Ubuntu, Cantarell, "Helvetica Neue", sans-serif;
}

#app {
  height: 100%;
}

main {
  display: flex;
  align-items: center;
  justify-content: center;
  background-color: #fefefe;
  background-image: linear-gradient(#fc6c48 0%, #ef5081 100%);
  background-repeat: no-repeat;
  background-size: cover;
  height: 100%;
}

.todo-wrapper {
  width: 400px;
  max-width: 100%;
  min-height: 500px;
  margin: 20px auto 40px;
  border: 1px solid #eee;
  border-radius: 4px;
  padding: 40px 20px;
  -webkit-box-shadow: 0 0 15px 0 rgba(0, 0, 0, 0.05);
  box-shadow: 0 0 15px 0 rgba(0, 0, 0, 0.05);
  background-color: #f4f7fc;
  overflow: hidden;
  position: relative;
}

.todo-title {
  font-size: 1.2em;
  color: #f65c65;
  font-weight: normal;
}

.btn,
input {
  line-height: 2em;
  border-radius: 3px;
```

```
  border: 0;
  display: inline-block;
  margin: 15px 0;
  padding: 0.2em 1em;
  font-size: 1em;
}

input[type="text"] {
  border: 1px solid #ddd;
  min-width: 70%;
  transition: all ease-in 0.25s;
}

input:focus {
  outline: none;
  border: 1px solid #a3b1ff;
}

.btn {
  text-align: center;
  font-weight: bold;
  cursor: pointer;
  border-width: 1px;
  border-style: solid;
}

.btn-add {
  background: #6664ff;
  border-color: #6664ff;
  pointer-events: visible;
  color: #fefefe;
  min-width: 12%;
  transition: all ease-in 0.25s;
  font-size: 2.2em;
  line-height: 0.5em;
  padding: 0.3em 0.3em;
  float: right;
}

ul.todo-list {
  padding: 0;
  margin-bottom: 30px;
}

ul.todo-list li {
  position: relative;
  list-style-type: none;
  display: block;
  margin: 10px 0;
  background: #e0e8f5;
  border-radius: 3px;
  padding: 12px;
  overflow: hidden;
}
</style>
```

Open `App.vue` file and amend the code to this,

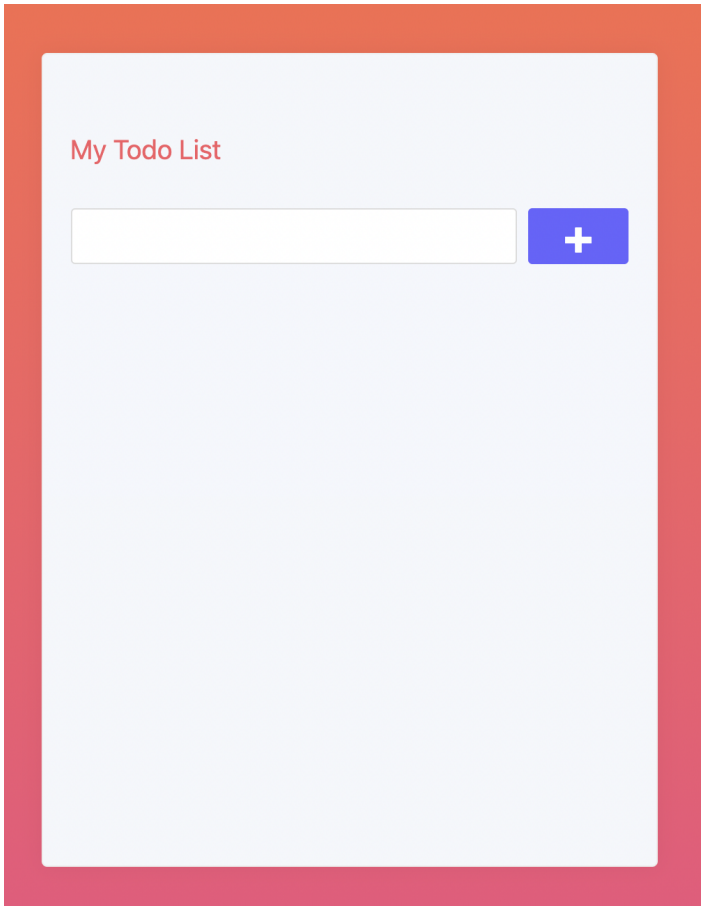
```
<template>
  <Todos />
</template>

<script>
import Todos from "./components/Todos.vue";

export default {
  name: "App",
  components: {
    Todos,
  },
};
</script>

<style></style>
```

The app should now look like this.



Back to the `Todos.vue` file, within the `script` tag, we will declare to variables for the component, `inputValues` and `todos` variables.

```
<script>
export default {
  name: "Todos",
  data() {
    return {
      inputValue: "",
```

```
      todos: [],
    };
  },
};
</script>
```

Update the input element to use the `inputValue` variable.

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

Create a method function to push the new `inputValue` when the user press on the add button.

```
<script>
export default {
  name: "Todos",
  data() {
    return {
      inputValue: "",
      todos: [],
    };
  },
  methods: {
    addTodo() {
      this.todos.push(this.inputValue);
      this.inputValue = "";
    },
  },
};
</script>
```

Let's now call the `addTodo()` method into the template!

```
      <input v-model="inputValue" @keyup.enter="addTodo" type="text" />
      <div class="btn btn-add" v-on:click="addTodo">+</div>
```

Now all that's left is to display the todos in the `ul` tag.

```
      <ul class="todo-list">
        <li v-for="todo in todos" :key="todo">
          {{ todo }}
        </li>
      </ul>
```

That's all! You can now insert new todo into the list!