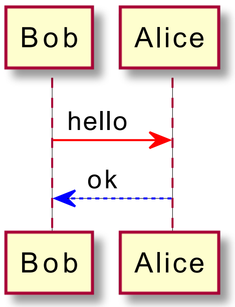
learning-vue3

from vue 2 to vue 3: composition API

this is a test paragraph.



```
<template>
  <div v-if="user">
    You are already logged-in !
    <div><button @click="handleLogout">Logout</button></div>
    <pre>{{ user }}</pre>
  </div>
  <div v-else>
    <form @submit="handleSubmit">
      <div>
        <input v-model="fields.identifier.value.value" type="text" />
        <div style="color: red">{{ form.errors.value.identifier }}</div>
      </div>
      <div>
        <input v-model="fields.password.value.value" type="password" />
        <div style="color: red">{{ form.errors.value.password }}</div>
      </div>
      <div>
        <input
          :disabled="submitButtonIsDisabled()"
          type="submit"
          :value="submitButtonValue()"
        />
      </div>
    </form>
  </div>
</template>

<script>
import { ref } from "vue";
import useRequest from "@/use/request";
import { useForm, useField } from "vee-validate";
import { localLogin, localLogout, getUser } from "@/utils/auth";

export default {
  setup() {
    const user = ref(getUser());
    const loginRequest = useRequest();

    const form = useForm({
      initialValues: {
        identifier: "",
        password: "",
      },
    });

    const fields = {
      identifier: useField("identifier", (value) =>
        value && value.trim() ? true : "Mail is required"
      ),
      password: useField("password", (value) =>
        value && value.trim() ? true : "Password is required"
      ),
    };

    const handleSubmit = form.handleSubmit((values) => {
      loginRequest
        .request("/auth/local", {
          method: "post",
          data: {
            identifier: values.identifier,
            password: values.password,
          },
        })
        .then((response) => {
          localLogin({
            jwt: response.data.jwt,
            user: response.data.user,
          });
          user.value = response.data.user;
        });
    });

    function submitButtonValue() {
      return loginRequest.state.value === "PENDING" ? "pending..." : "Login";
    }

    // désactiver le bouton de soumission si le formulaire est invalidate
    // ou que la requête vers l'API est en cours de traitement
    function submitButtonIsDisabled() {
      return loginRequest.state.value === "PENDING" || !form.meta.value.valid;
    }

    function handleLogout() {
      localLogout();
      user.value = null;
    }

    return {
      handleSubmit,
      submitButtonValue,
      submitButtonIsDisabled,
      loginRequest,
      form,
      fields,
      user,
      handleLogout,
    };
  },
};
</script>
```