Automate user events

```
import { render, screen } from "@testing-library/react";
import userEvent from "@testing-library/user-event";
import App from "../App";

test('Render a component', ()=>{
  render(<App/>)
  const counter = screen.getByText(/counter/i);
  const increment = screen.getByRole("button", { name: /increment/i });
  const decrement = screen.getByRole("button", { name: /decrement/i });
  // We validate the initial render
  expect(counter).toHaveTextContent("Counter: 0");
  // We fire the click 🔥
  userEvent.click(increment);
  // Finally, we verify the results on the component
  expect(counter).toHaveTextContent("Counter: 1");
  // Same for decrement button
  userEvent.click(decrement);
  expect(counter).toHaveTextContent("Counter: 0");
})
```

Let's type something 🐙

```
import { render, screen } from "@testing-library/react";
import userEvent from "@testing-library/user-event";
import {Form} from "../App";

const randomUser = {
  user: "fpetre@vairix.com",
  password: ":party-parrot:",
};

test("Simulate keyboard typing with `userEvent`🎹", () => {
    let submittedUser;
    const handleSubmit = jest.fn();
    // mockImplementation allows us to define the implementation of a jest function
    handleSubmit.mockImplementation((user) => {
      // here we save the user data to `submittedUser`
      submittedUser = user;
    });
    render(<Form handleSubmit={handleSubmit} />);
    const userInput = screen.getByLabelText(/user/i);
    const passwordInput = screen.getByLabelText(/password/i);
    const sendBtn = screen.getByRole("button", { name: /send/i });
    userEvent.type(userInput, randomUser.user); // here is the magic ✨🐙✨
    userEvent.type(passwordInput, randomUser.password);
    userEvent.click(sendBtn);
    expect(submittedUser).toEqual(randomUser);
    expect(handleSubmit).toHaveBeenCalledTimes(1);
  });
```