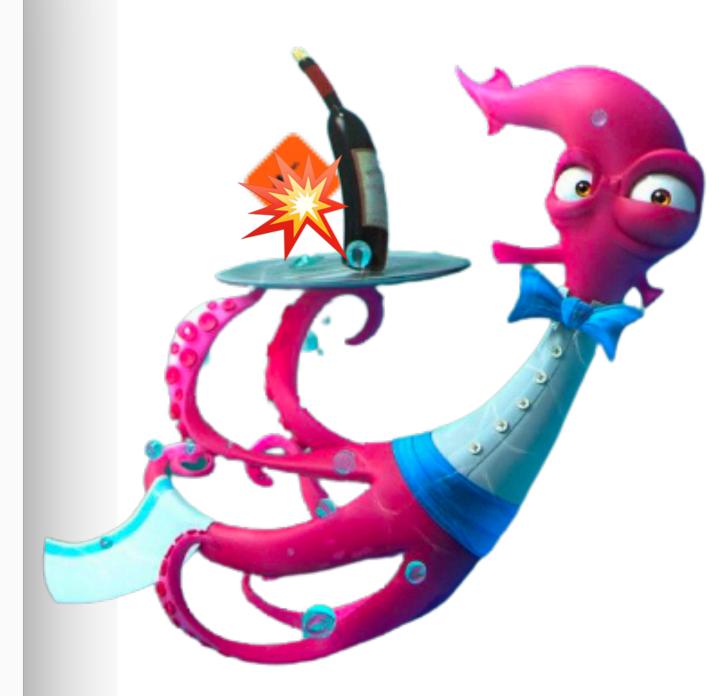
## HTTP mocking with MSW

(Unhappy Path)



```
import {
  render,
  screen,
  waitForElementToBeRemoved,
} from "@testing-library/react";
import userEvent from "@testing-library/user-event";
import { rest } from "msw";
import { setupServer } from "msw/node";
import App from "../App";
// here we go with the serverSetup
const server = setupServer();
const rickAndMortyApi = `https://rickandmortyapi.com/api/character/1`;
beforeAll(() => {
  server.listen();
});
afterEach(() => {
  server.resetHandlers();
});
afterAll(() => {
  server.close();
});
const message = `R.I.P \( \bigau^*\);
    server.use(
      rest.get(rickAndMortyApi, (req, res, ctx) => {
       // It's pretty same as before but setting an error status code
        return res(ctx.delay(1000), ctx.status(500), ctx.json({ message }));
      })
    render(<App />);
    await userEvent.click(screen.getByRole("button", { name: /get rick/i }));
    await waitForElementToBeRemoved(() => screen.getByText(/loading/i), {
      timeout: 2000,
    });
    expect(screen.getByRole("alert")).toHaveTextContent(message);
  });
```



## Ok, but show me the code

I created a sample repo on my GitHub, it has way more deeper explanation on it.