Consumir servicios de una Api en un componente React mediante Ajax JQuery

Queremos consumir un servicio API en un componente React, para ello, voy a utilizar Ajax con JQuery para poder hacer las operaciones CRUD y, por tanto, consumir el servicio.

Debéis tener instalado el programa node.js para que el terminal nos pueda reconocer los comandos que vamos a utilizar en los siguientes apartados.

https://nodejs.org/es/

Lo primero que vamos a necesitar para poder hacerlo es crearnos un proyecto React. Lo haremos utilizando el siguiente comando:

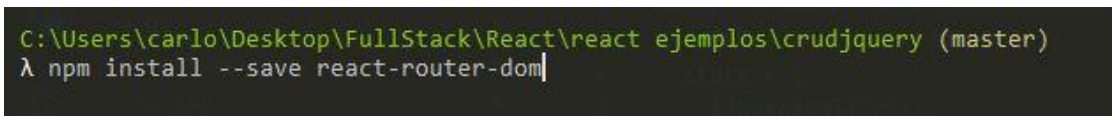
En este caso yo he llamado a mi proyecto crudjquery.

He utilizado el programa Cmder que es un terminal para poder ejecutar los comandos, pero también podéis utilizar el símbolo del sistema.

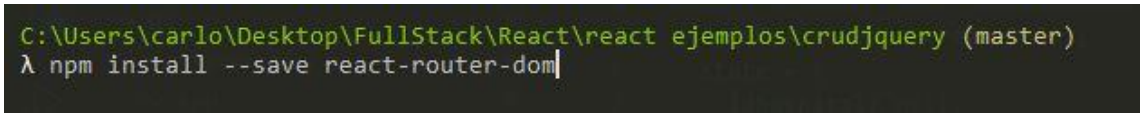
Una vez creado me voy a situar en la carpeta del proyecto y le voy a añadir unas funcionalidades al mismo mediante los siguientes comandos:

- JQuery.

- React-Router-Dom (no es necesario para el proyecto, pero lo voy a utilizar para tener más organizado el proyecto).



- BootStrap (opcional para darle diseño).



No dispongo de acceso un servicio API público para poder realizar todas las operaciones CRUD, pero nos vamos a crear un servicio API local para posteriormente poder consumirlo.

Necesitaremos tener un fichero JSON que contenga los datos que vamos consumir más tarde.

Ejecutaremos el comando “npm install -g json-server” en nuestro Cmder.

```
C:\Users\carlo\Desktop\FullStack\React\react ejemplos (master)
λ npm install -g json-server
```

Nos situaremos en la carpeta donde esta nuestro fichero JSON y ejecutaremos el siguiente comando “json-server --watch alumnos.json”. En este caso “alumnos.json” es el fichero que voy a utilizar yo para el servicio API.

```
C:\Users\carlo\Desktop\FullStack\React\react ejemplos (master)
λ json-server --watch alumnos.json
```

Ya tendríamos desplegado nuestro servicio API poder consumirlo.

En nuestro proyecto React he creado 4 componentes, en cada uno de ellos voy a realizar una operación CRUD distinta. En cada componente voy a necesitar importar el JQuery instalado mediante los comandos utilizados antes.

```
import $ from "jquery";
```

En mi primer componente voy a realizar la operación GET.

```
mostraralumnos = () => {
  var url = "http://localhost:3000/alumnos/";
  var status = false;
  var alumnos = [];
  $.ajax({
    url: url,
    dataType: "json",
    cache: false,
    method: `GET`,
    success: function (data) {
      alumnos = data;
      status = true;
    }.bind(this),
  }).then(() => {
    if (status == true) {
      this.setState({
        status: true,
        alumnos: alumnos,
      });
    }
  });
};
```

Nuestra llamada a la API se realiza introduciendo la url en el apartado “url”, el tipo de dato que vamos a leer, y la operación a realizar en el “method”. Si realiza correctamente la operación entrará en el apartado “success” y realizará el código que le indiquemos. Una vez realizada la operación, ya que el código se ejecuta más rápido que la llamada al servicio, conseguiremos, con el método “then” que ejecute un determinado código justo al acabar la operación. En este proyecto he creado un array de alumnos dentro del método y una variable booleana llamada “status” para poder cambiar los valores del “state” posteriormente, porque en el “success” del Ajax el “state” de React no es reconocido y daría un error si lo intentamos realizar de esa forma.

Realizar operación CRUD POST.

Es muy parecida que la operación “GET”. Las diferencias son que en este caso el método es el “POST”, hay un apartado “data” que le asignamos nuestro alumno en formato JSON y tenemos “content-type”, en él indicamos el formato en el que vamos a realizar la inserción.

```
postAlumno = (e) => {
  e.preventDefault();
  //variables internas del metodo
  var alumno = {};
  var status = false;
  alumno.idalumno = parseInt(this.cajaNumeroRef.current.value);
  alumno.name = this.cajaNombreRef.current.value;
  alumno.email = this.cajaEmailRef.current.value;
  var alumnoJson = JSON.stringify(alumno);
  //url de la api
  var url = "http://localhost:3000/alumnos/";
  $.ajax({
    url: url,
    type: "POST",
    data: alumnoJson,
    contentType: "application/json",
    success: function (data) {
      //si la insercion es correcta entra aqui
      status = true;
    },
    error: function () {
      console.log("Error en ajax api POST alumnos");
    },
  }).then(() => {
    //lo que queremos que realice despues de realizar la insercion
    if (status == true) {
      this.setState({
        status: true,
      });
    }
  });
};
```

Realizar operación CRUD PUT.

Se realiza de la misma manera que la operación “POST, solo modificamos el “method” que en este caso es “PUT” para realizar la modificación y en “url” debemos indicar cuál es objeto a modificar, en este ejemplo seria el alumno.

```
putAlumno = (e) => {
  e.preventDefault();
  //variables internas del metodo
  var alumno = {};
  var status = false;
  alumno.id = parseInt(this.props.idalumno);
  alumno.name = this.cajaNombreRef.current.value;
  alumno.email = this.cajaEmailRef.current.value;
  var alumnoJson = JSON.stringify(alumno);
  //url de la api
  var url = "http://localhost:3000/alumnos/" + this.props.idalumno;
  $.ajax({
    url: url,
    type: "PUT",
    data: alumnoJson,
    contentType: "application/json",
    success: function (data) {
      //si la modificacion es correcta entra aqui
      status = true;
    },
    error: function () {
      console.log("Error en ajax api PUT alumnos");
    },
  }).then(() => {
    //lo que queremos que realice despues de realizar la modificacion
    if (status == true) {
      this.setState({
        status: true,
      });
    }
  });
};
```

Realizar operación CRUD PUT.

Se realiza de la misma manera que la operación “POST, solo modificamos el “method” que en este caso es “PUT” para realizar la modificación y en “url” debemos indicar cuál es objeto a modificar, en este ejemplo seria el alumno.

```
deleteAlumno = () => {
  //url de la api
  var url = "http://localhost:3000/alumnos/" + this.props.idalumno;
  //variable interna del metodo
  var status = false;
  $.ajax({
    url: url,
    dataType: "json",
    cache: false,
    method: `DELETE`,
    success: function (data) {
      //si realiza correstamente la eliminacion entra aqui
      status = true;
    }.bind(this),
  }).then(() => {
    //lo que queremos que realice despues de realizar la eliminacion

    if (status == true) {
      this.setState({
        status: true,
      });
    }
  });
};
```

Realizar operación CRUD DELETE.

Realizara una eliminación de un objeto de la Api. Realizaremos lo mismo que en la operación “GET” con la diferencia del “method” que es “DELETE” y en “url” debemos indicar cuál es objeto a modificar, en este ejemplo seria el alumno.

```
deleteAlumno = () => {
  //url de la api
  var url = "http://localhost:3000/alumnos/" + this.props.idalumno;
  //variable interna del metodo
  var status = false;
  $.ajax({
    url: url,
    dataType: "json",
    cache: false,
    method: `DELETE`,
    success: function (data) {
      //si realiza correstamente la eliminacion entra aqui
      status = true;
    }.bind(this),
  }).then(() => {
    //lo que queremos que realice despues de realizar la eliminacion
    if (status == true) {
      this.setState({
        status: true,
      });
    }
  });
};
```