# EJERCICIO DE FORMACIÓN REACT/REDUX

[EJERCICIO DE FORMACIÓN REACT/REDUX 1](#_Toc523903770)

[Sobre este Ejercicio 2](#_Toc523903771)

[ReactReduxBack 3](#_Toc523903772)

[Explicación y Despliegue 3](#_Toc523903773)

[TAREAS 5](#_Toc523903774)

[ReactReduxFront 6](#_Toc523903775)

[Explicación y Despliegue 6](#_Toc523903776)

[TAREAS 8](#_Toc523903777)

[Enlaces de Interés 9](#_Toc523903778)

## Sobre este Ejercicio

Este ejercicio se compone de dos proyectos bien diferenciados

* ReactReduxBack
  + Java
  + Spring
  + Hibernate
  + JPA
  + SQL
* RectReduxFront
  + JavaScript
  + NodeJS
  + React
  + Redux

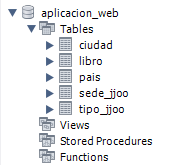
Cada una de las partes cuenta con una explicación, un pequeño tutorial de montaje y su propia sección de tareas. Comencemos.

### ReactReduxBack

#### Explicación y Despliegue

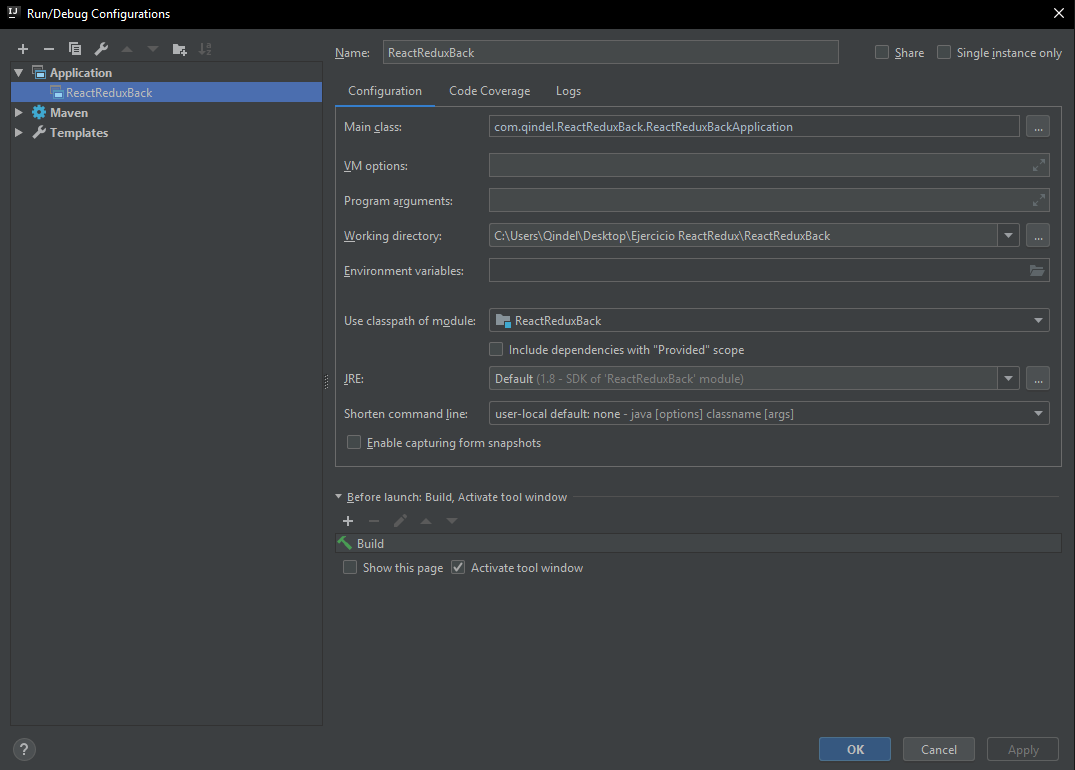
Se trata de una aplicación generada con Spring Initializr usando los módulos Web, JPA y MySQL. Debemos importarla como una aplicación Maven en nuestro IDE.

El primer paso será cargar en nuestro MySQL local el archivo inicializarBD.sql, que generará el esquema aplicacion\_web con las tablas necesarias.



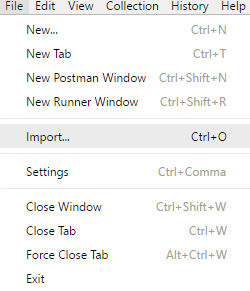
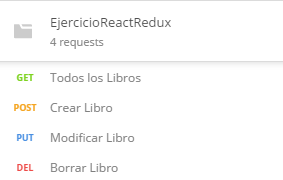
Es posible que para conectar con la misma necesites hacer alguna modificación al archivo **src/main/resources/application.properties**

En el caso de que no se genere automáticamente, es necesaria la siguiente “*run configuration”*



**IMPORTANTE: Si nos encontramos con problemas, debemos cerciorarnos de que estamos usando java 8.**

Si todo va bien hasta aquí, podemos lanzar la aplicación y usar Postman para probar los 4 endpoints ya creados y configurados en una collection importable presente en esta misma carpeta **(EjercicioReactRedux.postman\_collection.json)**

****

Probar antes POST que PUT y DELETE; está preparado para funcionar con la BD recién importada.

#### *TAREAS*

1. Mapear el resto de las entidades de la BD (La base de datos es muy similar a la de otros ejercicios, si los has llevado a cabo antes, quizás puedas reutilizar código)
2. Implementar el CRUD de Sedes (Mismo caso que el ejercicio anterior)
   1. Una buena práctica en REST consiste en que PUT realice una inserción si la ID que se intentaba modificar no existe en base de datos
   2. Establecer algún tipo de validación sobre los DTO de request y en general las operaciones de creación y modificación, de manera que si estas operaciones no pudieran completarse se devuelva un HttpStatus o mensaje personalizado.
3. Crear un endpoint que devuelva la siguiente consulta (Presente también en los anteriores ejercicios): *“Todas las ciudades dadas de alta en el sistema indicando, para cada una de ellas, el número de veces que ha sido sede de unos juegos olímpicos; distinguiendo entre el número de veces que lo ha sido para juegos de invierno y de verano. Además, se mostrará también el dato "valor" que puede ser nulo para una ciudad concreta y en cuyo caso se mostraría el definido para el país.”*

### ReactReduxFront

#### Explicación y Despliegue

Es una aplicación generada con create-react-app, a la que se le ha añadido Redux para la gestión del estado de la aplicación.

Lo primero que necesitamos hacer, si es que nuestro equipo no lo tiene todavía, es instalar NodeJS <https://nodejs.org/en/>. Cualquier versión del mismo presente en este momento en la web es válida (09/08/18). El v8 presenta mejores compatibilidades y quizá estabilidad, y el v10 usa las funciones más modernas de JavaScript. Por suerte o por desgracia, ninguna de las dos versiones tiene características específicas que estemos buscando, así que cualquiera es válida.

Cuando tengamos NodeJS instalado, habremos instalado también su gestor de paquetes, npm.

Para la creación de este ejercicio se han usado los siguientes comandos: **(no es necesario ejecutarlos para lanzar el ejercicio)**

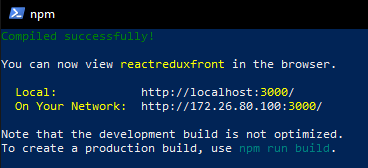
>npm install -g create-react-app

>create-react-app reactreduxfront

Con el primer comando instalamos globalmente una herramienta para generar boilerplates de aplicaciones web con react. Con la segunda generamos la aplicación con el nombre indicado en el directorio actual. No se permiten mayúsculas.

Este boilerplate contiene gran cantidad de paquetes npm ya configurados que nos facilitarán mucho la vida a la hora de desarrollar nuestra aplicación React. Para lanzar la aplicación (esta o cualquier otra aplicación NodeJS).

>npm start



**IMPORTANTE: Si estamos lanzando una aplicación NodeJS importada y no creada por nosotros (como es tu caso ahora mismo), antes de esto debemos ejecutar**

**>npm install**

**para que las dependencias se descarguen y no se produzcan errores inesperados**

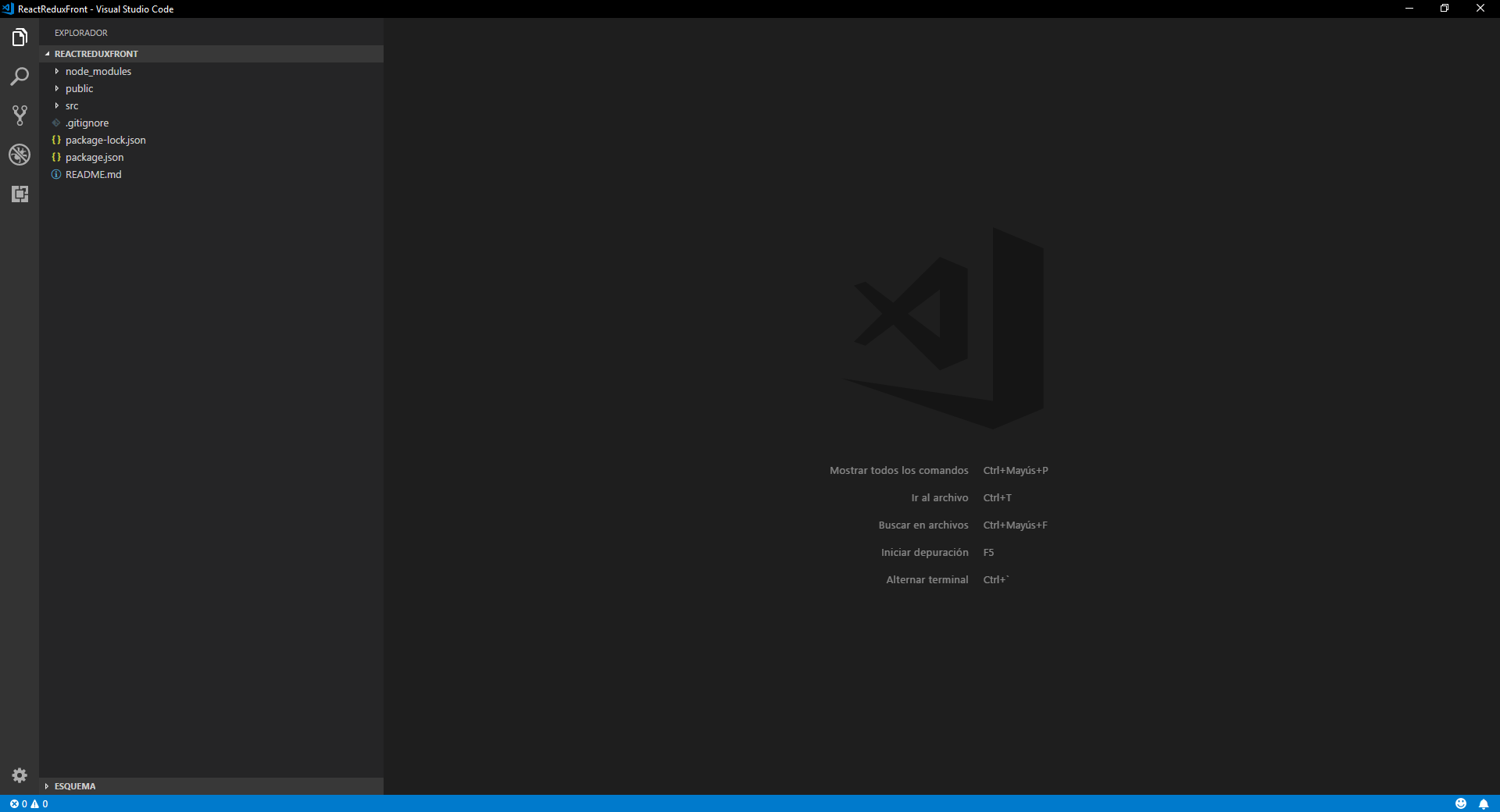
**IMPORTANTE 2: Los repositorios que npm accede pueden personalizarse en el archivo .npmrc presente en tu raíz de usuario. Este archivo podría no existir, pero también podría estar en tu máquina actual como un residuo de su propietario anterior (Si es que lo hubo). Si lo encuentras, asegúrate de moverlo a un lugar seguro antes de ejecutar la instalación para que no se produzcan aun más errores inesperados.**

En el caso de este ejercicio, obtendremos una aplicación corriendo en el puerto 3000 que aplicará cambios cada vez que guardemos un solo archivo de la carpeta del proyecto. Esta aplicación se abrirá automáticamente en nuestro navegador por defecto

La misma consola que usamos para lanzar el proyecto mostrará **errores y advertencias de errores potenciales** y buenas prácticas cuando se de el caso; es una buena idea mantener un ojo en ella y en las herramientas de desarrollador.

Además de esto, también se ***intentará*** seguir el estándar de **ESLint** para obtener un JavaScript lo más limpio y entendible posible <https://eslint.org/docs/rules/>. Si quieres asegurarte de que sigues dichas normas, puesto que son numerosas, puedes buscar un plugin para tu editor de texto o IDE que se encargue de comprobarlas por ti (*Aunque esto no se ha hecho a la hora de preparar este ejercicio, puesto que, entre otras cosas, algunas sintaxis o normas podrían quitarles simplicidad a los ejemplos que aquí se muestran)*.

Recomendación: Visual Studio Code <https://code.visualstudio.com/> (Gratis, fácil y eficiente)



El proyecto a tratar en este caso es una simple interfaz basada en pestañas. Se ha implementado una pestaña especialmente sencilla de bienvenida, que no es más que HTML plano, y una segunda pestaña que demuestra el flujo básico de una aplicación basada en React/Redux cuyas peticiones apuntan a al proyecto ReactReduxBack que debería estar funcionando en local. Se han implementado en frontal una consulta y una creación sobre la entidad “Libro” de base de datos, pero no la modificación y el borrado. De hecho, ni siquiera se ha planteado la interfaz para estas dos operaciones.

#### TAREAS

1. Del mismo modo que se ha generado una tabla de consulta para la entidad “Libro”, genera una nueva pestaña con una tabla de consulta para la respuesta del endpoint que generaste en el ejercicio 3 del anterior apartado.
   1. Una de las ventajas de la programación basada en componentes es su reusabilidad. Puedes modificar los componentes ya existentes (respetando y no rompiendo su funcionalidad actual) para que sean capaces de representar tu consulta o crear nuevos componentes con esta reusabilidad en mente.
2. Crea una nueva pestaña para el CRUD de Sedes.
   1. En el ejemplo proporcionado no se presenta interfaz alguna para la modificación ni el borrado. Puedes usar modales, botones al final de la fila, filas clickables, selección múltiple, jugar con las rutas y propiedades de los componentes para hacerlos aparecer y desaparecer…
   2. Ten siempre en mente que debes parametrizar lo máximo posible los componentes que generes, puesto que cuando más se puedan generalizar más reutilizables serán y más tiempo ahorrarás a largo plazo. Trata de que las funciones y datos con los que trabajen no estén establecidos en el código en la medida de lo posible.
3. Implementa la ordenación por columnas.
   1. Con una implementación lo suficientemente genérica, y si tus anteriores componentes son reutilizables, puede que solo necesites implementar este algoritmo una vez.
   2. Si estás devolviendo distintos tipos de dato, recuerda que números y cadenas se ordenan de distinta manera
4. Modulariza al máximo la aplicación.
   1. Extrae los formularios, botones y fragmentos de componentes más grandes que se muestren recurrentes a componentes separados, y elimina todo el [Copy and paste programming](https://en.wikipedia.org/wiki/Copy_and_paste_programming) que puedas mediante esto.
   2. Quizás encuentres componentes que dependen demasiado entre ellos. El estado de Redux se comparte entre toda la aplicación, así que te ayudará en estos casos

## Enlaces de Interés

A continuación, enlaces a las páginas de descarga y/o documentación de las tecnologías y recursos que componen la parte frontal del ejercicio.

|  |  |  |
| --- | --- | --- |
| Recurso | Enlace | Descripción |
| React | <https://reactjs.org/docs/getting-started.html> | Librería base |
| Redux | <https://es.redux.js.org/> | Gestión de un estado común de la aplicación |
| NodeJS | <https://nodejs.org/en/> | Motor de aplicaciones JavaScript |
| React-Redux | <https://github.com/reduxjs/react-redux> | Bindings oficiales de Redux para React |
| React-Router | <https://reacttraining.com/react-router/> | Gestión de rutas y renderizado condicional de componentes basado en las mismas |
| History | <https://github.com/ReactTraining/history> | Gestión del historial. Incluido en React-Router y usado en redirecciones para no forzar la recarga de la página. |
| Redux-promise-action-middleware | <https://www.npmjs.com/package/redux-promise-action-middleware> | Genera acciones correspondientes a los distintos estados de una petición HTTP (LOADING/SUCCESS/FAILURE) |
| React-Bootstrap | <https://react-bootstrap.github.io/components/alerts/> | Librería de componentes CSS de Bootstrap. Facilita mucho dar estilo a la aplicación |
| Axios | <https://github.com/axios/axios> | Cliente para realizar peticiones HTTP. Usado para la comunicación con backend |
| ESLint | <https://eslint.org/docs/rules/> | Estándar de calidad de código para JavaScript. Cuantas más reglas sigamos mejor. |
| Visual Studio Code | <https://code.visualstudio.com/> | IDE recomendado para el desarrollo de este tipo de aplicación web |
| ECMAScript 6 | <http://es6-features.org/#BlockScopedVariables> | Sintaxis específica de JavaScript 2015, muy extendida en el desarrollo con React/Redux |