Ejercicios React

[**1. Software de base 2**](#_l4raksjmxsrk)

[Plugins de VSC 2](#_f1sjqfvzvgag)

[Comandos utiles 2](#_cu1yne99dw5e)

[**2. Setup local de un CRUD preexistente 2**](#_e7v0mbdo6rel)

[Ejercicios para el caso presentado 5](#_j49nkc1lxjsd)

[**3. Challenges técnicos 6**](#_8vw7nz2521a3)

[3.a) Chat con efectos 6](#_cv3psom9fais)

[3.b) State as a Snapshot 6](#_axyi4h3upg2d)

[3.c) Pasar una función como prop 6](#_gjjt89n9ovb2)

[3.d) JSX in depth 6](#_dh0u5xdkraan)

[**4. Proyectos de cero 7**](#_4jci9zu401xc)

[4.a) Tic-tac-toe to hangman 7](#_6n7znj9c170o)

[4.b) CRUD 3 capas 7](#_stkmlftsed6n)

[4.C) React con servicios REST y CORS 7](#_9xd2pt65r3b0)

# 

# 1. Software de base

React es un framework de lado cliente pero usaremos diferentes productos de servidor para los endpoints. Inicialmente usaremos Neo4J: <https://nodejs.org/en/download>

## Plugins de VSC

Existen varios plugins de React para VSC sin embargo no añaden mucha funcionalidad, vamos a instalar el marcado como “starred” en este listado.

## 

## Comandos utiles

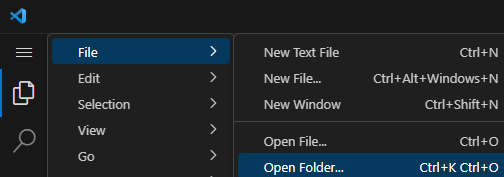
| **Utilidad** | **Comando** |
| --- | --- |
| Actualizar npm (para instalaciones antiguas) | npm install -g npm |
|  |  |

# 2. Setup local de un CRUD preexistente

Arrancar Visual Studio Code (VSC)

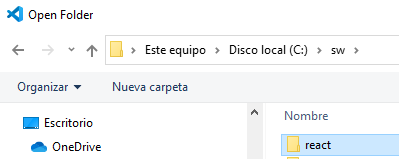


Especificar la carpeta que queremos abrir

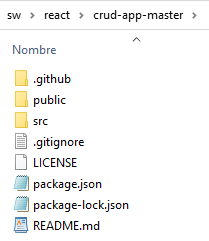


Vamos a usar algo similar a lo siguiente (una ruta corta, sin espacios en blanco, directamente en el disco principal)

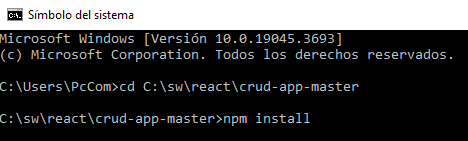
**c:/sw/react**

****

Descargar [este repositorio](https://github.com/SafdarJamal/crud-app/) de GitHub



Esta aplicación corre sobre NodeJS, abrir una ventana cmd para ejecutar NPM

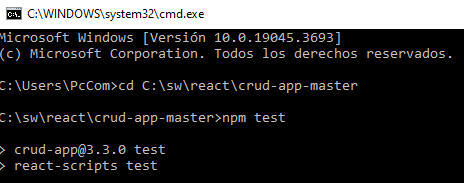


**npm install**

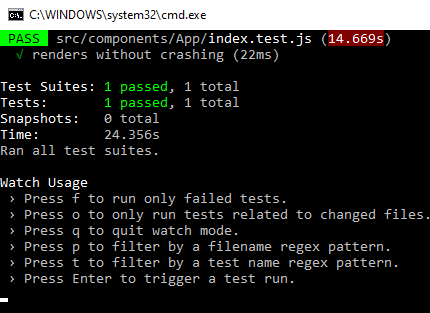
**npm start**

Vamos a ejecutar el test que viene con el proyecto

**npm test**

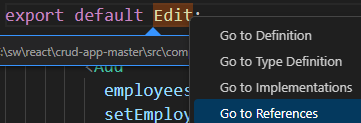


Deberíamos tener éxito en el único test existente

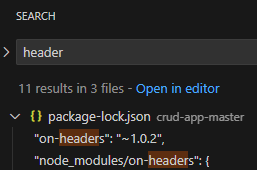


## Ejercicios para el caso presentado

1. Usar VSC para navegar el proyecto usando CTRL, evitar usarlo en tipos nativos de React o el elementos que no sean parte de la jerarquía o composición de React del propio ejemplo (es decir, queremos navegar los componentes Type, Edit, Table,...)
2. Usa “Botón derecho → “Go to references” para encontrar uso de components en otros components



1. Para algunas referencias que VSC no descubre automaticamente tenemos que usar Search



1. Identificar dónde está programada la prueba
2. Por último diagramar todo el caso práctico usando la herramienta que desees (Visio, Paint, Gimp, StarUML, Diagrams.net,...)

# 3. Challenges técnicos

## 3.a) Chat con efectos

Analizar cada caso del uso de efectos presentados en estos ejemplos y diagramarlos individualmente

<https://react.dev/reference/react/useEffect#examples-connecting>

## 3.b) State as a Snapshot

Vamos a estudiar en detalle el caso de state como un snapshot que se crea durante la fase de render, esto se entiende mejor sí se compara con el diagrama de ciclo de vida que vimos en teoría.

<https://react.dev/learn/state-as-a-snapshot>

## 3.c) Pasar una función como prop

Las lambdas se pueden pasar como parámetros para que se ejecuten “a posteriori”, analizar el caso presentado en este link

<https://react.dev/reference/react/Children#calling-a-render-prop-to-customize-rendering>

## 3.d) JSX in depth

Solucionar estos retos y analizar la utilidad del fallo (como de fiable es la descripción del problema en relación al problema real) en cada caso

<https://react.dev/learn/javascript-in-jsx-with-curly-braces#challenges>

# 4. Proyectos de cero

## 4.a) Tic-tac-toe to hangman

El tic-tac-toe (3 en raya) es el [tutorial básico de react](https://react.dev/learn/tutorial-tic-tac-toe). Vamos a programar el proyecto entero, y después vamos a usarlo como ejemplo para montar nuestro propio juego del ahorcado. Iremos copiando trozos de código y adaptándolo para montar el juego completo

¿Qué métodos “de negocio” debería tener la clase que implemente la funcionalidad del juego?

Intenta programar de cero el caso práctico (existen varias soluciones en Internet pero no las uses para la implementación).

## 4.b) CRUD 3 capas

Monta un crud de 3 capas desde cero sobre un ámbito de negocio que te resulte interesante (videoclub, gestor de videojuegos, música), no hace falta montar backend, se puede simular con JSONs en el lado cliente.

## 4.C) React con servicios REST y CORS

Fuente: <https://www.stackhawk.com/blog/react-cors-guide-what-it-is-and-how-to-enable-it/#enable-cors-on-server-side>

CORS no es realmente un problema de lado de cliente pero vamos a programar una aplicación React que sirva de front a una aplicación faceless implementada con soporte CORS.