

[v2] Full Stack JS - Code Challenge - Español

- Code Challenge
 - ¿Que tomamos en cuenta al revisar el challenge?
 - ¿Que hacer si tengo dudas sobre algo de lo pedido?
 - ¿Como envío el challenge?
- Code
 - Opcional
 - API
 - libs & frameworks:
 - specs
 - opcional
 - FRONT
 - libs & frameworks:
 - specs
 - opcional

Code Challenge

El JavaScript challenge para Full Stack esta dividido en 3 partes:

- Un API usando Node + Express
- Un Frontend cliente usando Bootstrap + React
- Puntos opcionales para el challenge

¿Que tomamos en cuenta al revisar el challenge?

- El código entregado tiene que funcionar y cumplir con lo solicitado
- Las instrucciones y documentación deben ser prolijas y claras
- Debe cumplir con los requisitos técnicos solicitados (incluir las librerías/frameworks específicos, versiones pedidas, test, instrucciones para correrlo, documentación, etc.)
- Los puntos opcionales no restan en caso de no entregarse o no cumplir lo esperado, pero suman en caso de estar bien.

¿Que hacer si tengo dudas sobre algo de lo pedido?

Ponte en contacto con la persona que te envió el challenge, es preferible aclarar dudas antes de enviar mal el challenge y también nos ayuda a mejorar las consignas.

¿Como envío el challenge?

Debes dejar tu código en un repo git publico o uno al cual podamos acceder y luego enviarle la información a la persona que te envió este examen (url y datos de acceso).

Code

- El código que envíes debe correr usando NodeJS 12 y no depender de librerías que están instaladas de forma global, variables de entorno o configuraciones de algún sistema operativo específico.
- El código debe ser escrito en JavaScript (ES6+), no utilizar: TypeScript, Dart, Elm, etc
- En cuanto a las librerías y frameworks, puede usar la versión que consideres apropiadas.

Opcional

- Usar Docker o Docker Compose para correr las apps

API

Un API REST, a la cual se le envía un texto y responde con el mismo texto invertido.

libs & frameworks:

- NodeJs <https://nodejs.org/en/>
- ExpressJs <https://expressjs.com/>
- Mocha <https://mochajs.org/>
- Chai <https://www.chaijs.com/>
- SuperTest <https://github.com/visionmedia/supertest#readme>

specs

Usando NodeJs +ExpressJs se debe crear API de un solo endpoint al cual se le envía el texto de la siguiente forma:

```
GET /iecho?text=test
```

y da una respuesta 200 en caso de éxito:

```
content-type: application/json

{
  "text": "tset"
}
```

o una respuesta 400 en caso de parámetros inválidos:

```
content-type: application/json

{
  "error": "no text"
}
```

Ejemplo usando curl:

Curl

```
$ curl -v -X GET "http://apihost/iecho?text=test" -H "accept:
application/json"

> GET /iecho?text=test HTTP/1.1
> Host: apihost
> User-Agent: curl/7.68.0
> accept: application/json
>

< HTTP/1.1 200 OK
< Date: Mon, 19 Oct 2020 15:18:53 GMT
< Content-Type: application/json; charset=utf-8
< Content-Length: 15
< Connection: keep-alive

{"text":"tset"}
```

También se deben crear los tests que validan el API usando Mocha + Chai + Supertest. Los tests deben poder correrse usando "npm test"

opcional

- En la respuesta, indicar por medio del flag "palindrome": true si el texto enviado es un palindromo <https://es.wikipedia.org/wiki/Pal%C3%ADndromo>
- Usar StandarJs <https://standardjs.com/>

FRONT

Un App cliente en React que permita enviar textos a la API del punto anterior y muestre las respuestas del API desde la ultima a la primera.

libs & frameworks:

- NodeJs <https://nodejs.org/en/>
- Webpack <https://webpack.js.org/>
- Bootstrap <https://getbootstrap.com/>
- React <https://reactjs.org/>

specs

Usando React + Bootstrap se debe crear una pantalla similar a la que se muestra en el siguiente wireframe:

The wireframe shows a web application layout on a grid background. At the top is a red header bar. Inside the header, on the left, is a white text input field with the placeholder text "insert Text". To the right of the input field is a blue button with the text "Send". Below the header is a large light gray rectangular area. Centered within this gray area is a white rectangular box. Inside the white box, the word "Results:" is at the top left. Below it are three stacked white text input fields. The top input field contains the text "Third Text", the middle one contains "Second Text", and the bottom one contains "First Text".

opcional

- Usa Redux <https://redux.js.org/>
- Test unitarios usando Jest <https://jestjs.io/>