Lydia Yohannes

Fundamentos de ingeniería del software para sistemas cloud 2019/2020

Índice

[Nombre de los integrantes de la pareja 2](#_Toc30161495)

[Las URL del microservicio que haya desarrollado la pareja 2](#_Toc30161496)

[Las URL del frontend ya sea específico del microservicio o común para toda la aplicación 2](#_Toc30161497)

[El nivel de acabado al que se presenta 2](#_Toc30161498)

[Justificación de cómo han ido consiguiendo cada uno de los requisitos del microservicio 2](#_Toc30161499)

[Debe tener un frontend que permita hacer todas las operaciones de la API (este frontend puede ser individual o estar integrado con el resto de frontends). 2](#_Toc30161500)

[Debe estar desplegado en la nube y ser accesible en una URL 3](#_Toc30161501)

[Utilizar gestión del código fuente y mecanismos de integración continua 3](#_Toc30161502)

[Añadir validación a los formularios del frontend 4](#_Toc30161503)

[Interacción completa entre todos los microservicios de la aplicación integrando información 4](#_Toc30161504)

[Tener un front end común que integre los front ends de cada uno de los microservicios 4](#_Toc30161505)

[Análisis de los esfuerzos (en horas) dedicadas por cada uno 4](#_Toc30161506)

# Nombre de los integrantes de la pareja

Lydia Yohannes (he trabajado sola, sin pareja), Grupo 1

# Las URL del microservicio que haya desarrollado la pareja

Como he trabajado sin pareja no tenía que desarrollar un microservicio, sino centrarme en el frontend común.

# Las URL del frontend ya sea específico del microservicio o común para toda la aplicación

* Repositorio: <https://github.com/Lydia12/frontend>
* Integración continua: <https://travis-ci.com/Lydia12/frontend>
* Despliegue: <https://apidgt.herokuapp.com/>

El nivel de acabado al que se presenta

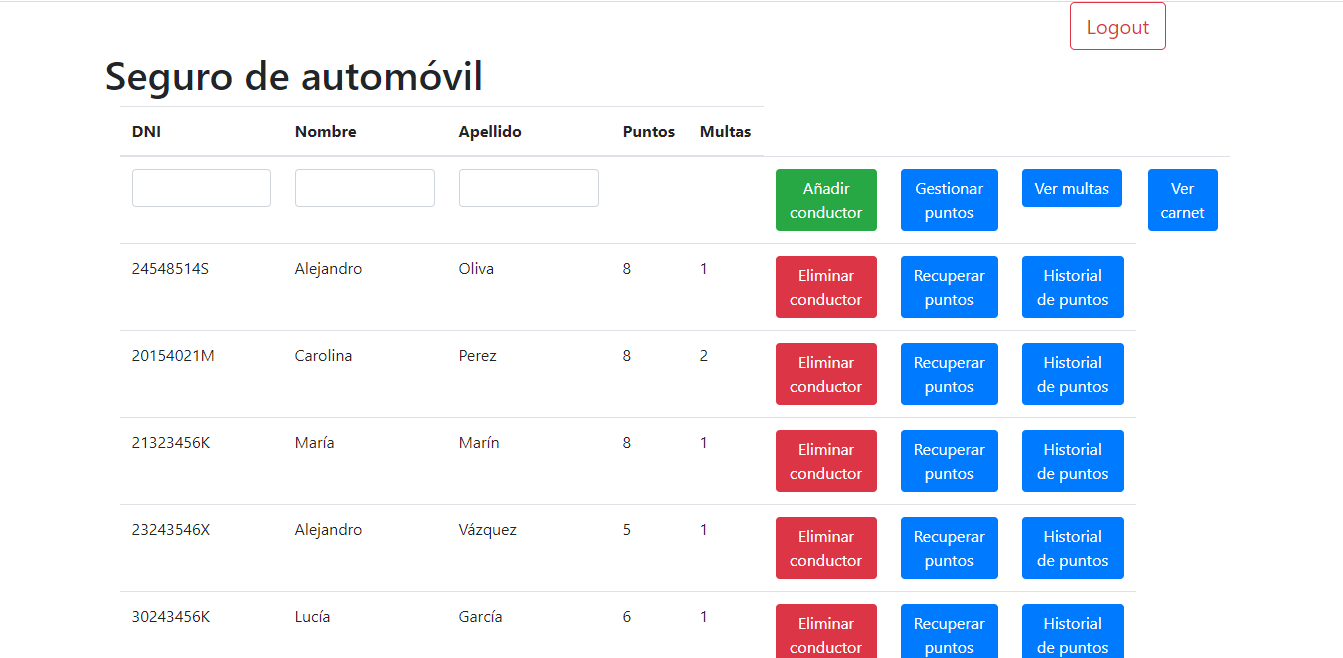
Como no tenía que trabajar en un microservicio sino en el frontend común, no podía aplicar las características exigidos para un microservicio (básico o avanzado). Pero algunos de los requisitos se puede aplicar para el frontend común básico o avanzado:

Justificación de cómo han ido consiguiendo cada uno de los requisitos del microservicio

## Debe tener un frontend que permita hacer todas las operaciones de la API (este frontend puede ser individual o estar integrado con el resto de frontends).

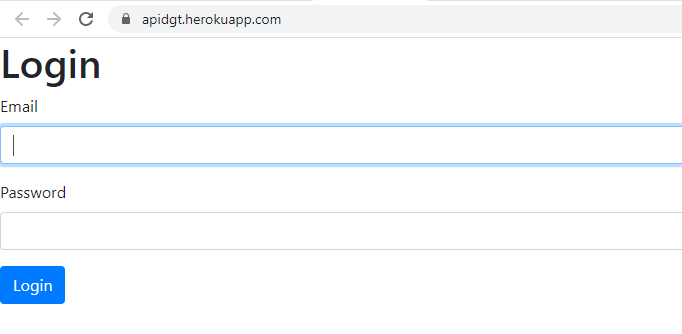
Para implementar la interacción con el API REST de los 3 microservicios he creado 3 clases: ConductoresApi (para Carnet), MultasApi y PuntosApi.

En el frontend común he integrado las informaciones más importantes de los tres microservicios en una tabla por la operacion GET de las 3 API’s en componentDidMount() para dar una vista general. El botón „Añadir conductor“ permita hacer la operación POST para crear un nuevo carnet y tambien una nueva entrada en el microservicio de puntos. Además el botón „Eliminar conductor“ permite llamar las operaciones„DELETE“ de los 3 microservicios (carnet, multas y puntos) y borrar la entrada correspondiente. He integrado el frontend común con el resto de frontends individuales donde se puede hacer todas las operaciones de cada API.



Debe estar desplegado en la nube y ser accesible en una URL

Para desplegar el frontend común he ejecutado „npm run build“ y utilizado el mecanismo de despliegue de Heroku basado en buildpacks(<https://github.com/mars/create-react-app-buildpack>). He usado este buildpack: <https://github.com/mars/create-react-app-buildpack>. El frontend común es accessible en esta URL: <https://apidgt.herokuapp.com/>.



## Utilizar gestión del código fuente y mecanismos de integración continua

He subido mi código en este repositorio de Github: <https://github.com/Lydia12/frontend> siguiendo Github flow. Y el código se compilar y se prueba automáticamente usando Travis.ci en cada commit (<https://travis-ci.com/Lydia12/frontend>)

## Añadir validación a los formularios del frontend

El frontend común tiene 3 inputs para crear un nuevo conductor. Para validar hay el metodo comprobarDNI(dni) en la clase Conductores. Esto comprueba si el DNI tiene 8 números y una letra al final, es decir, si el formato de DNI es válido. Al añadir también se comprueba, si los campos de nombre y apellido no estan vacíos.

## Interacción completa entre todos los microservicios de la aplicación integrando información

Esta interacción he hecho en la tabla principal del frontend común. De esa manera he integrado los informaciones más importantes de cada microservicio. Por ejemplo: Los puntos actuales y el numero de multas de los conductores.

## Tener un front end común que integre los front ends de cada uno de los microservicios

El frontend común integre los frontends individuales de los microservicios porque llama al a varias operaciones de la API de los 3 microservicios. Además he añadido buttons que redirigen a los frontends individuales donde se puede hacer todos las operaciones de cada microservicio.

# Análisis de los esfuerzos (en horas) dedicadas por cada uno

|  |  |
| --- | --- |
| **Requisito** | **esfuerzos (en horas)** |
| Debe tener un frontend que permita hacer todas las operaciones de la API (este frontend puede ser individual o estar integrado con el resto de frontends). | 10 |
| Debe estar desplegado en la nube y ser accesible en una URL. | 2 |
| Utilizar gestión del código fuente y mecanismos de integración continua:   * El código debe estar subido a un repositorio de Github siguiendo Github flow * El código debe compilarse y probarse automáticamente usando Travis.ci en cada commit | 5 |
| Añadir validación a los formularios del frontend. | 1 |
| Interacción completa entre todos los microservicios de la aplicación integrando información. | 10 |
| Tener un front end común que integre los front ends de cada uno de los microservicios. | 10 |