**MATRIZ DE FLEXIBILIDAD**

**Demo – Primera presentación**

**Esteban Bellegarde**

**Julián Gil Santos**

**C&S - 2017**

Contenido

[Introducción 3](#_Toc477513732)

[Objetivo 3](#_Toc477513733)

[Participantes 3](#_Toc477513734)

[Dominio 3](#_Toc477513735)

[Inconvenientes actuales 3](#_Toc477513736)

[Solución planteada 3](#_Toc477513737)

[Desarrollo 4](#_Toc477513738)

[Arquitectura 4](#_Toc477513739)

[Esquema principal: 4](#_Toc477513740)

[Arquitectura por capa 4](#_Toc477513741)

[Front­end 5](#_Toc477513742)

[JSX 5](#_Toc477513743)

[ECMAScript 6 5](#_Toc477513744)

[BABEL 6](#_Toc477513745)

[Webpack 7](#_Toc477513746)

[React 8](#_Toc477513747)

[Redux 9](#_Toc477513748)

[Back ­ End 10](#_Toc477513749)

[Rutas (capa de servicio) 10](#_Toc477513750)

[Modelo 11](#_Toc477513751)

[Capa de datos 12](#_Toc477513752)

[Interfaz de usuario 13](#_Toc477513753)

[Login 13](#_Toc477513754)

[Sección Empleado 13](#_Toc477513755)

[Planes a futuro 18](#_Toc477513756)

[En proceso 18](#_Toc477513757)

[Lo que falta 18](#_Toc477513758)

[Conclusiones 19](#_Toc477513759)

# Introducción

## Objetivo

Mejora de la experiencia del usuario de la Matriz de Flexibilidad, para cada perfil existente, mediante una solución provista por el área de I+D.

Objetivo secundario

Capacitación sobre tecnologías Javascript, aplicación de arquitecturas full stack junto con bases de datos relacionales, y en particular, la tecnología de front­end React.js

## Participantes

Esteban Bellegarde (desarrollador)



Julián Andrés Gil Santos (desarrollador)



Nicole Allegri (DG)



Ciro Romero (testing)



Leandro Ferrigno (líder técnico)



Ana corradi (líder DG)



## Dominio

Matriz: Archivo de Excel



Proceso:



1. Gerente (a partir de ahora, "administrador") envía notificación y archivo adjunto (mail) al empleado.
2. El empleado lo completa y lo devuelve al administrador.
3. El administrador, en caso de que haya enviado la solicitud a más de un empleado, los junta en un solo archivo del mismo formato.

### Inconvenientes actuales

Facilidad para equivocarse.



Proceso engorroso tanto para el administrador como para el empleado que completa el archivo en primera instancia.

### Solución planteada

Se propone una aplicación web, rediseñando y optimizando el flujo del proceso y experiencia del usuario.

# Desarrollo

## Arquitectura

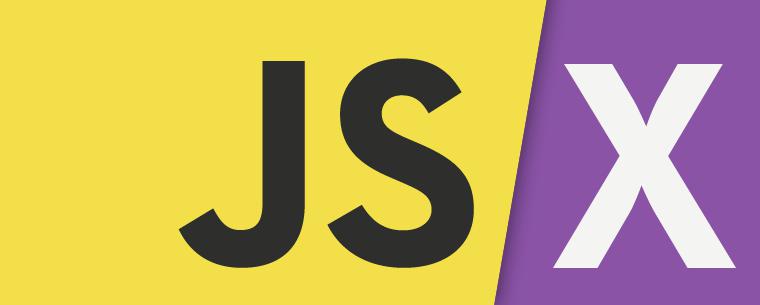
### Esquema principal:

### Arquitectura por capa

## 

## Front­end

### JSX



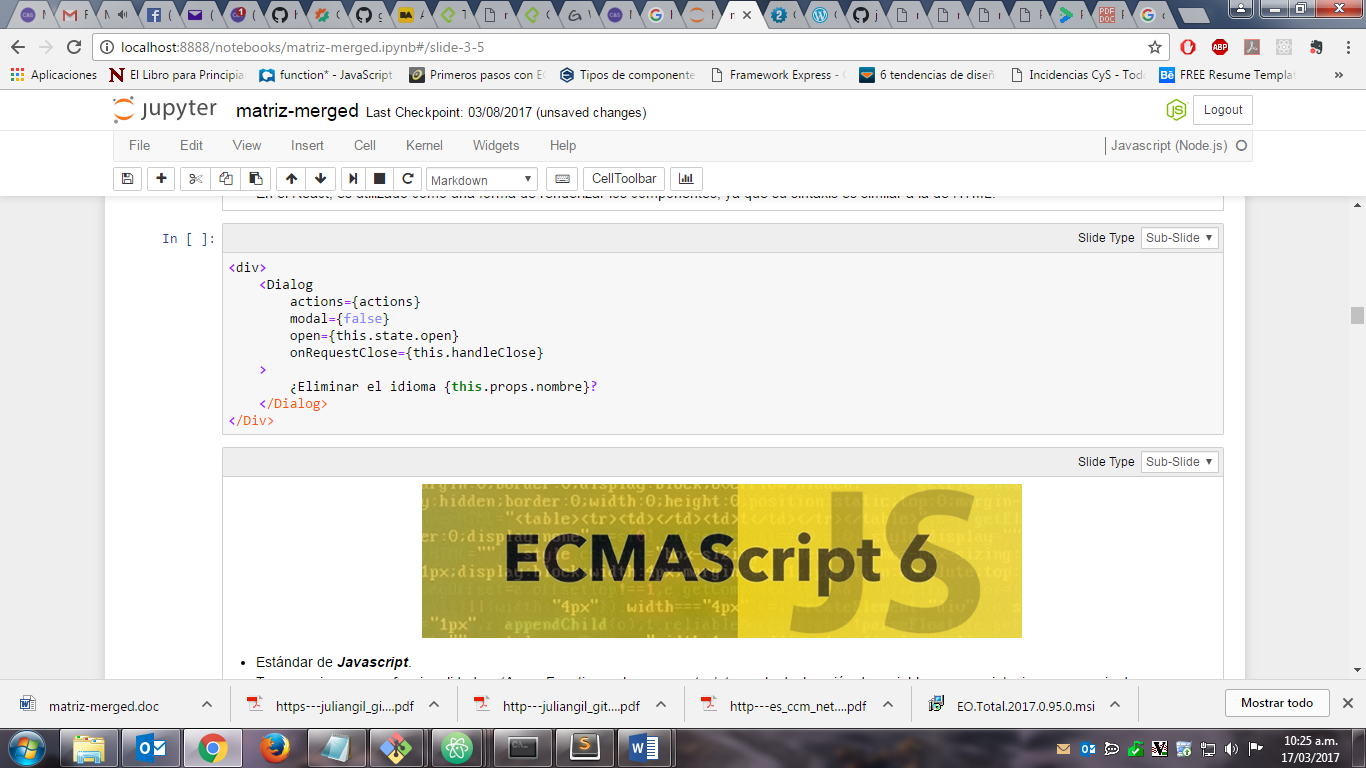
Pseudolenguaje.



Facilita el desarrollo de aplicaciones web con su sintaxis para crear elementos en el DOM.



En el React, es utilizado como una forma de renderizar los componentes, ya que su sintaxis es similar a la de HTML.



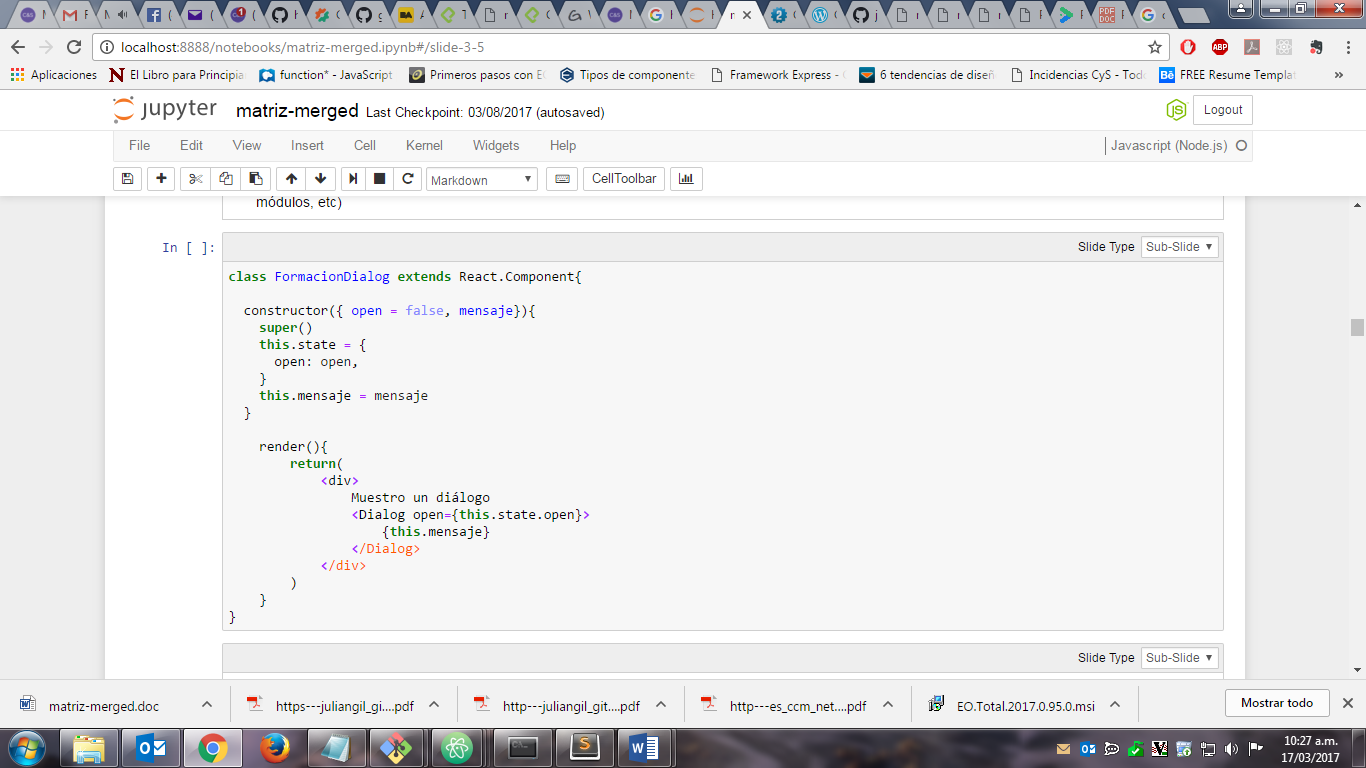
## ECMAScript 6



Estándar de *Javascript*.



Trae consigo nuevas funcionalidades: (Arrow Functions, clases, *const* y *let* para la declaración de variables, nueva sintaxis para manejo de módulos, etc)

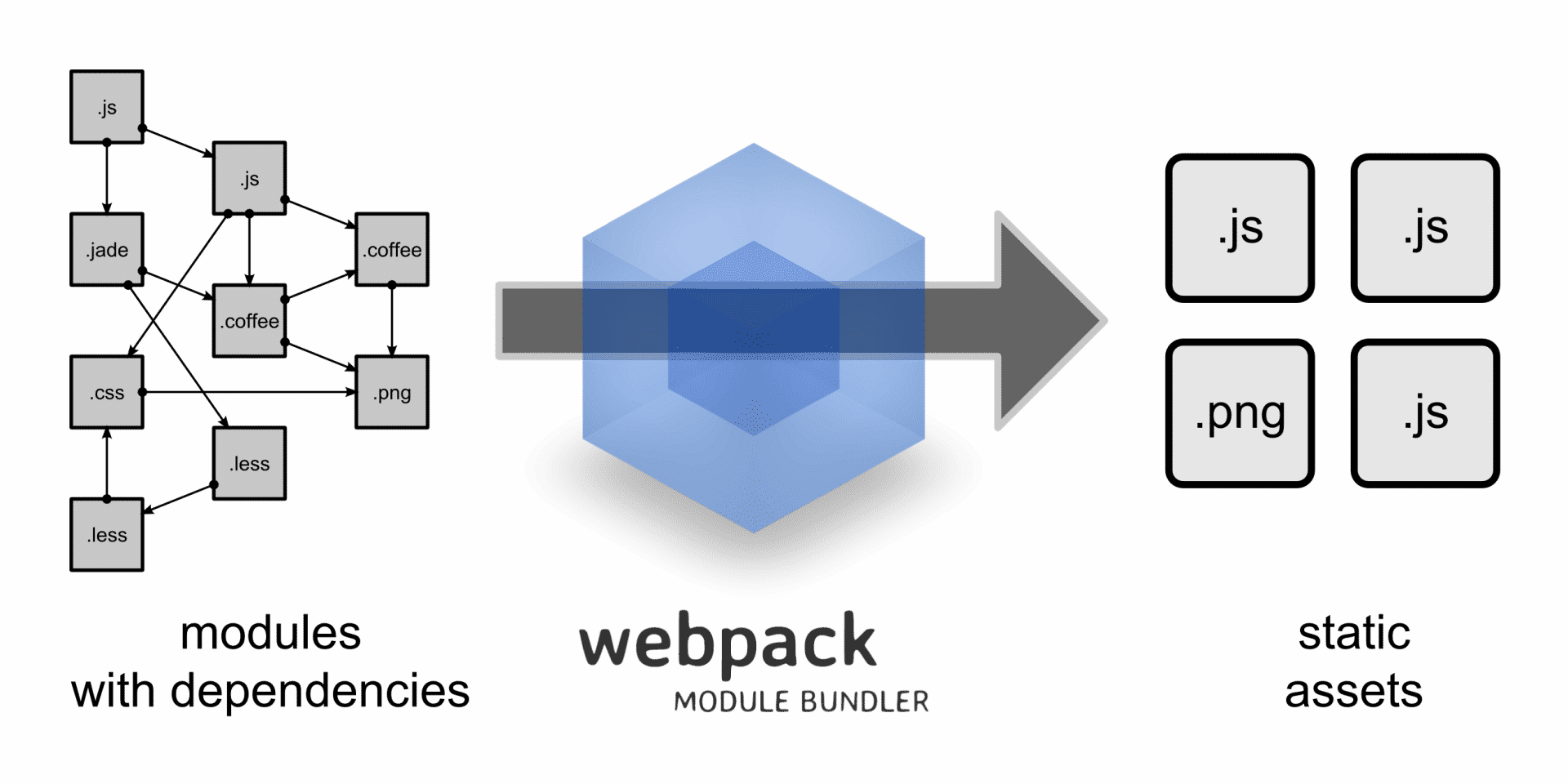


## BABEL

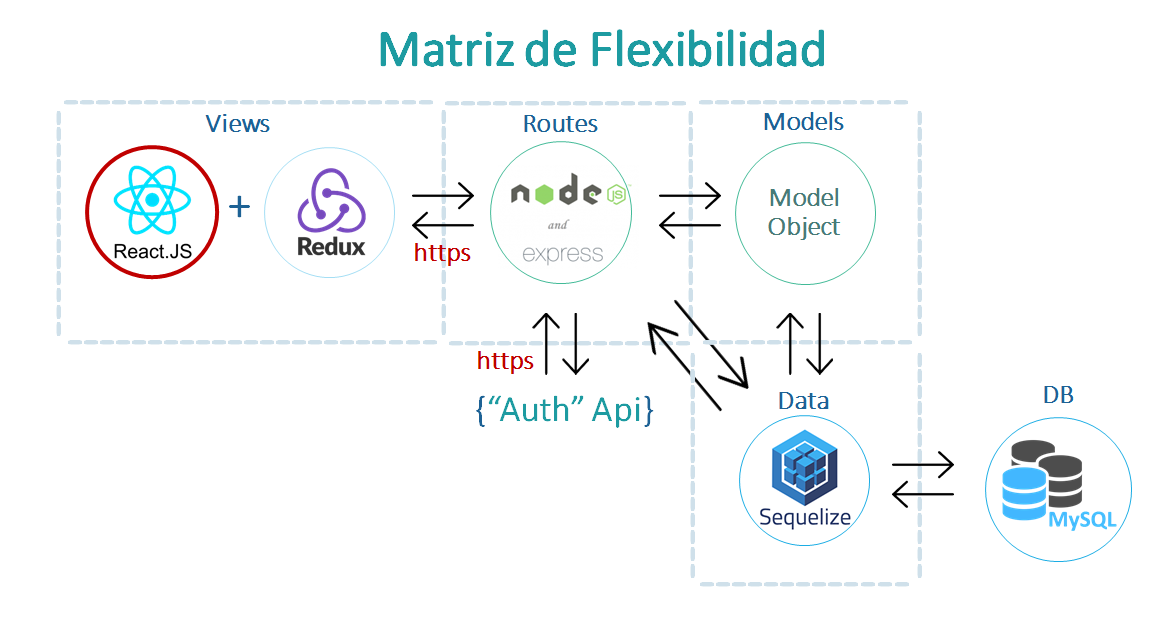
****

* Es un transpilador (traduce el código de un lenguaje a otro)
* Se transpiló de JSX y ES6 (no son soportados por el navegador) a ES5

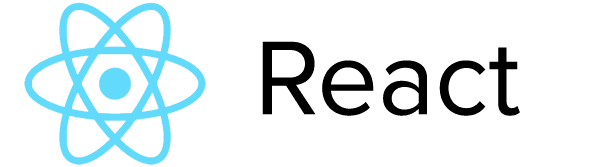
## Webpack



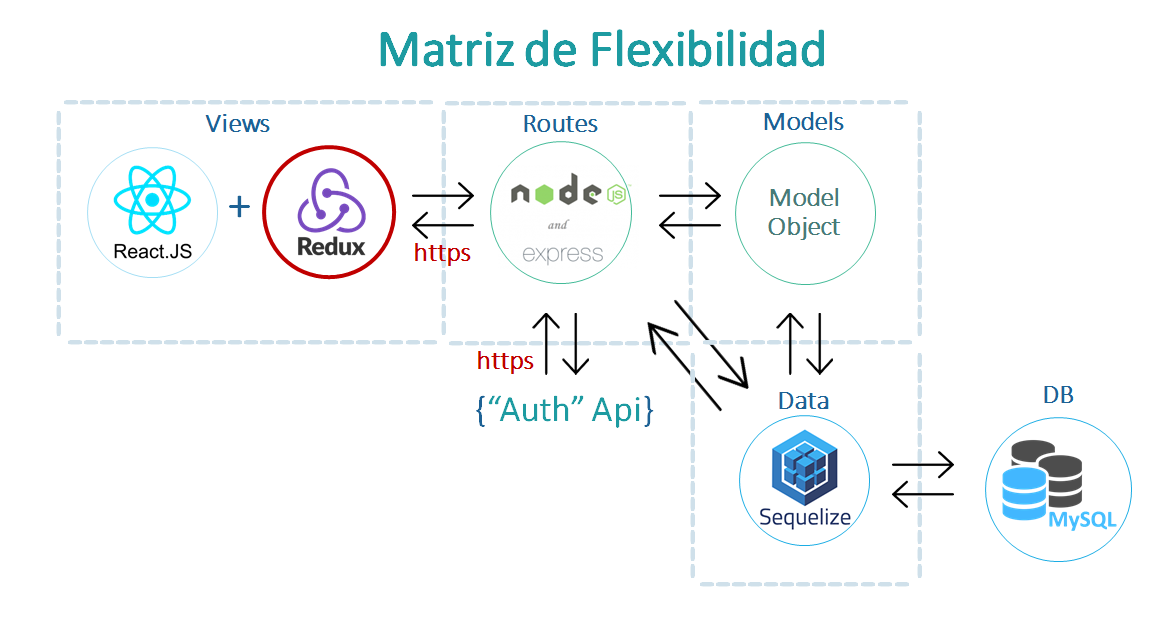
* Es un module bundler (*empaquetador de módulos*)
* Muy eficiente para aplicaciones grandes que contienen mucho código JavaScript
* Levanta la app en un server



## React



* Librería Javascript de código abierto, que sirve como herramienta para implementar la vista.
* Desarrollada por Facebook.
* Aplicaciones en una sola página.

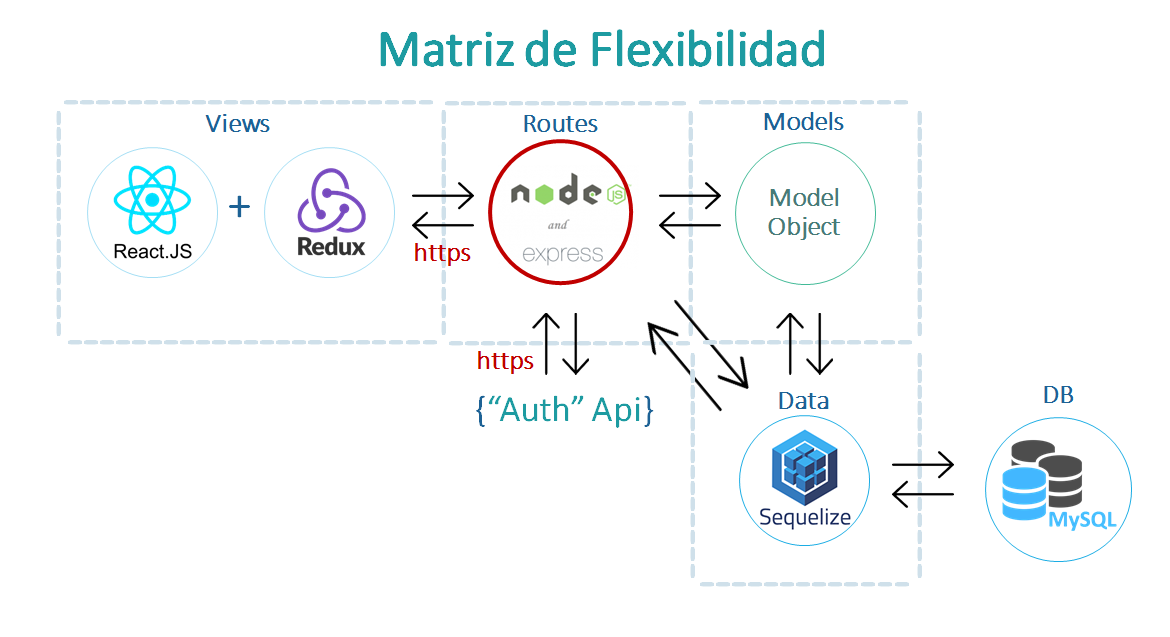


## Redux



* Su predecesor es Flux.
* Su definicion formal: *Contenedor de estados predecibles para aplicaciones javascript*.
* Se utiliza con React. Encapsula el estado de la aplicación, el comportamiento de la vista, y su comunicación con el back-end.

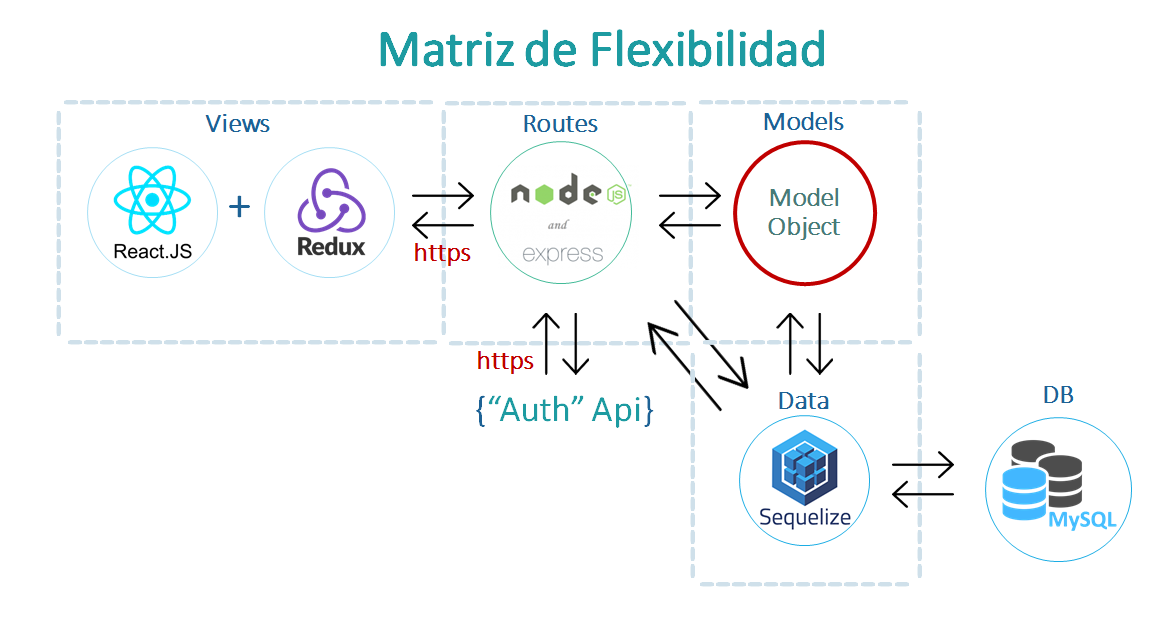
# Back ­ End



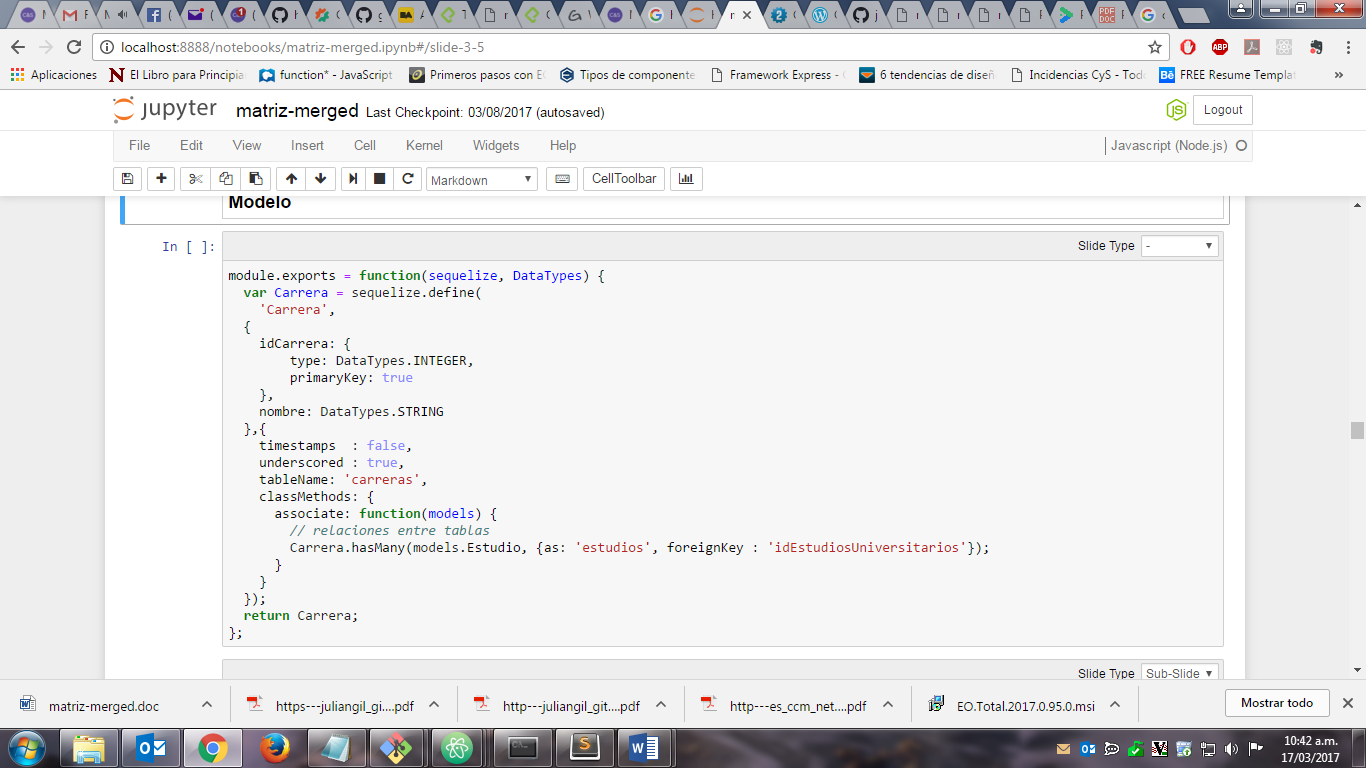
## Rutas (capa de servicio)

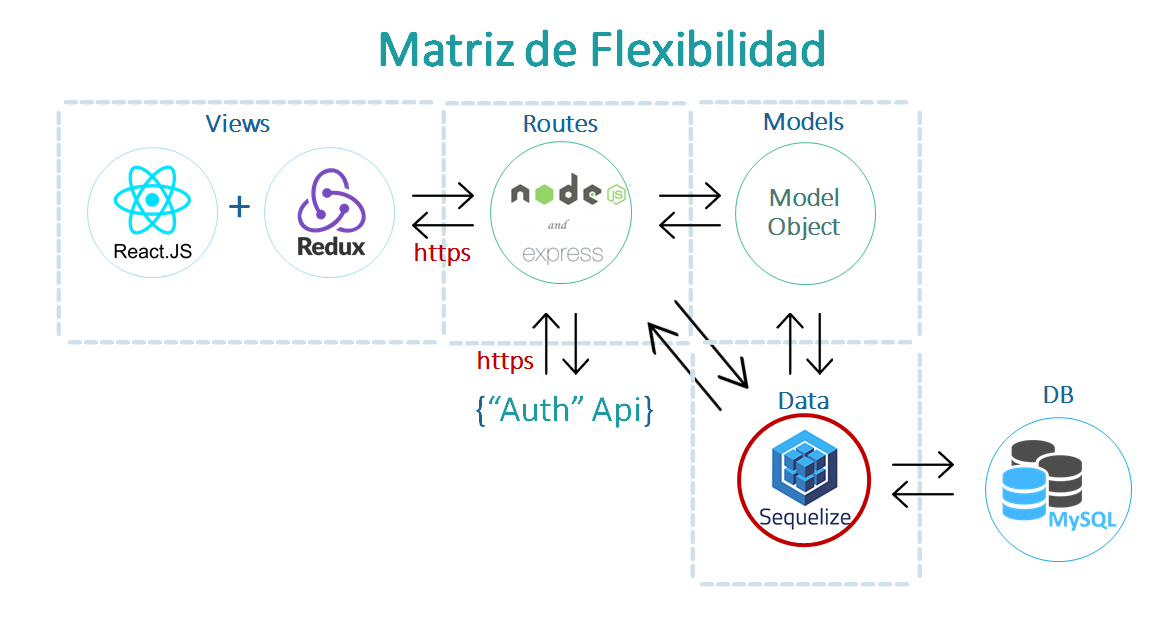


Usando **Express**, framework para Node.js, se levanta un server de backend y se especifica el comportamiento para cada petición HTTP recibida.



## Modelo





## Capa de datos



* Es un *ORM* para Node js.
* Soporta los motores de PostgreSQL, MySQL, MariaDB, SQLite y MSSQ.
* Facilita el acceso a datos para el programador - CRUD.
* Sequelize implementa *raw queries*, las cuales permite realizar SQL queries de forma fácil.

# Interfaz de usuario

Se comienza desde una pantalla de login, la cual está vinculada al servicio de LDAP de la base de usuarios de la empresa.

## Login



Ir: <http://10.2.1.69/>

Se dividió la aplicación en dos secciones, correspondientes a cada perfil:

* Empleado
* Administrador (el administrador tiene todas las funcionalidades de Empleado)

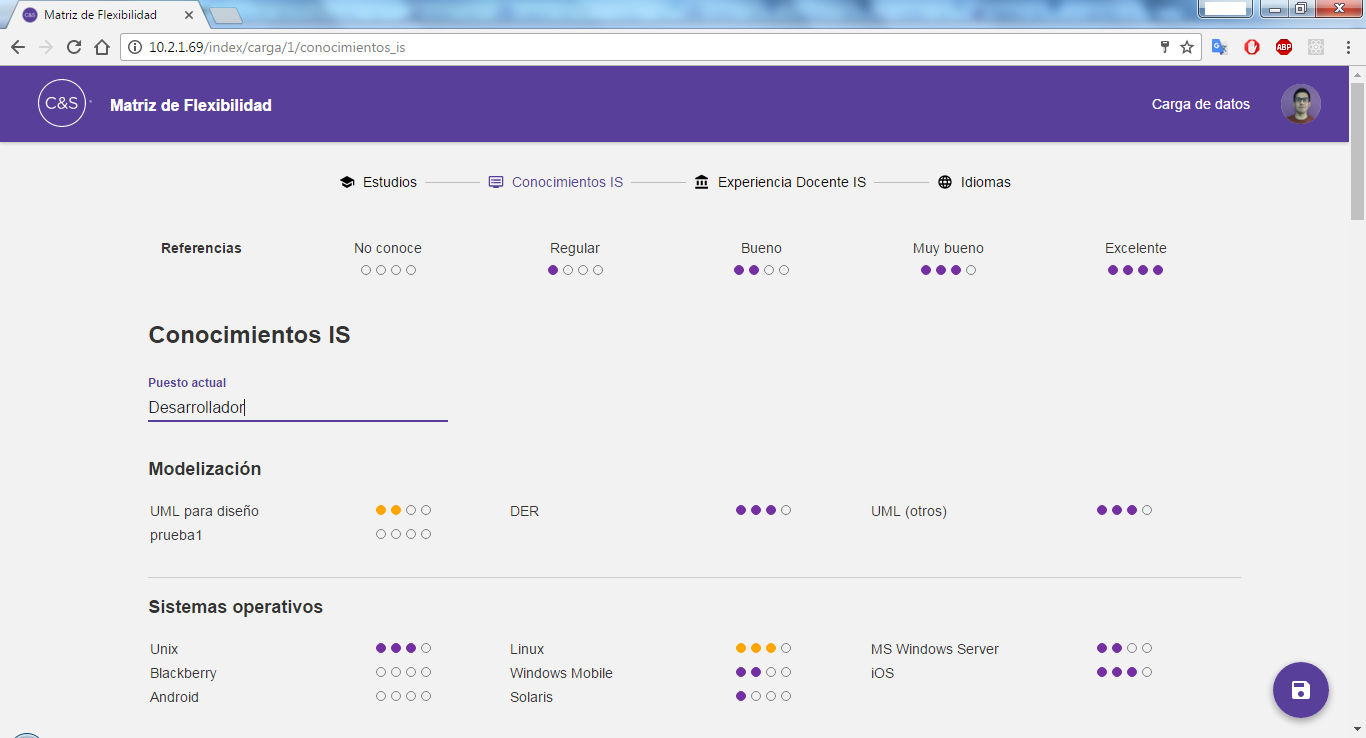
## Sección Empleado

Es la sección de carga de datos del negocio, del empleado autenticado en particular. Se divide en cuatro sub-secciones:

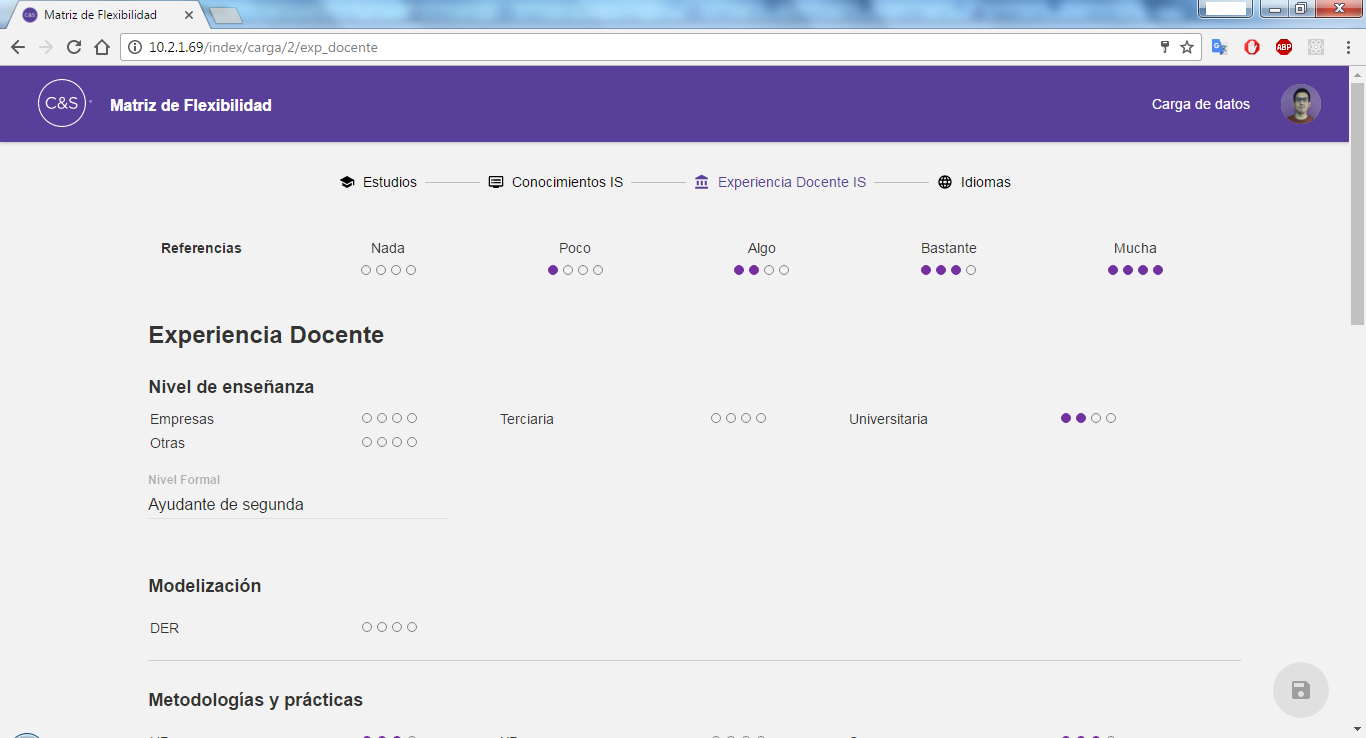
* Estudios
* Conocimientos en Ingeniería de Software
* Experiencia docente en Ingeniería de Software
* Idiomas

Ir: <http://10.2.1.69/index/carga>

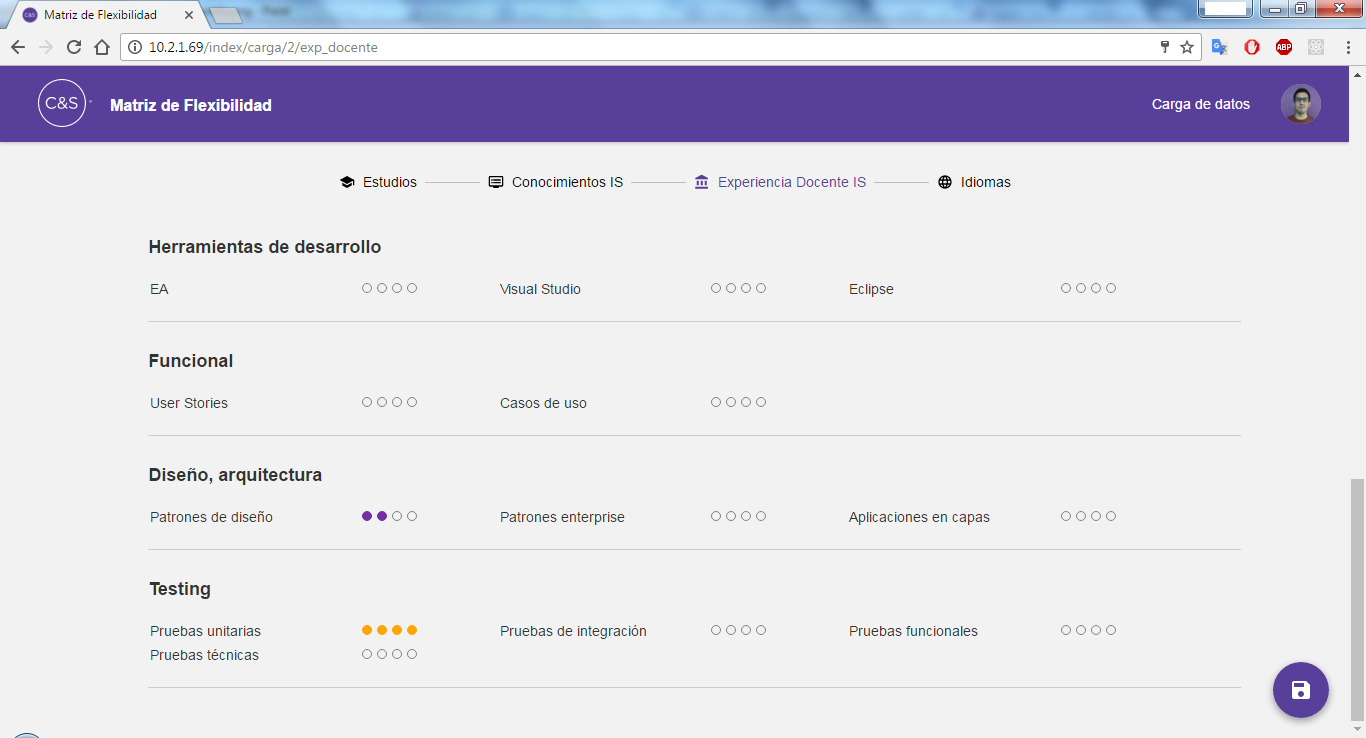
#### **Conocimientos en Ingeniería de Software**



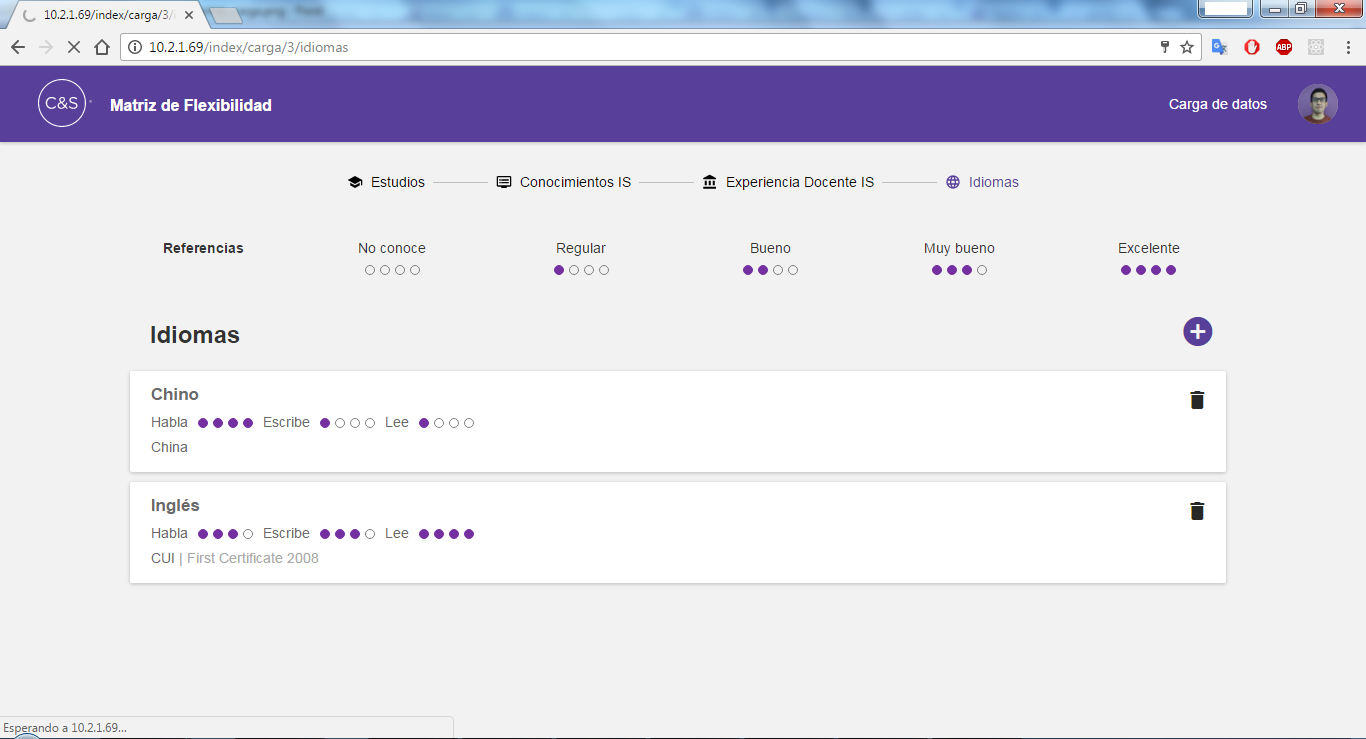
#### **Experiencia docente**



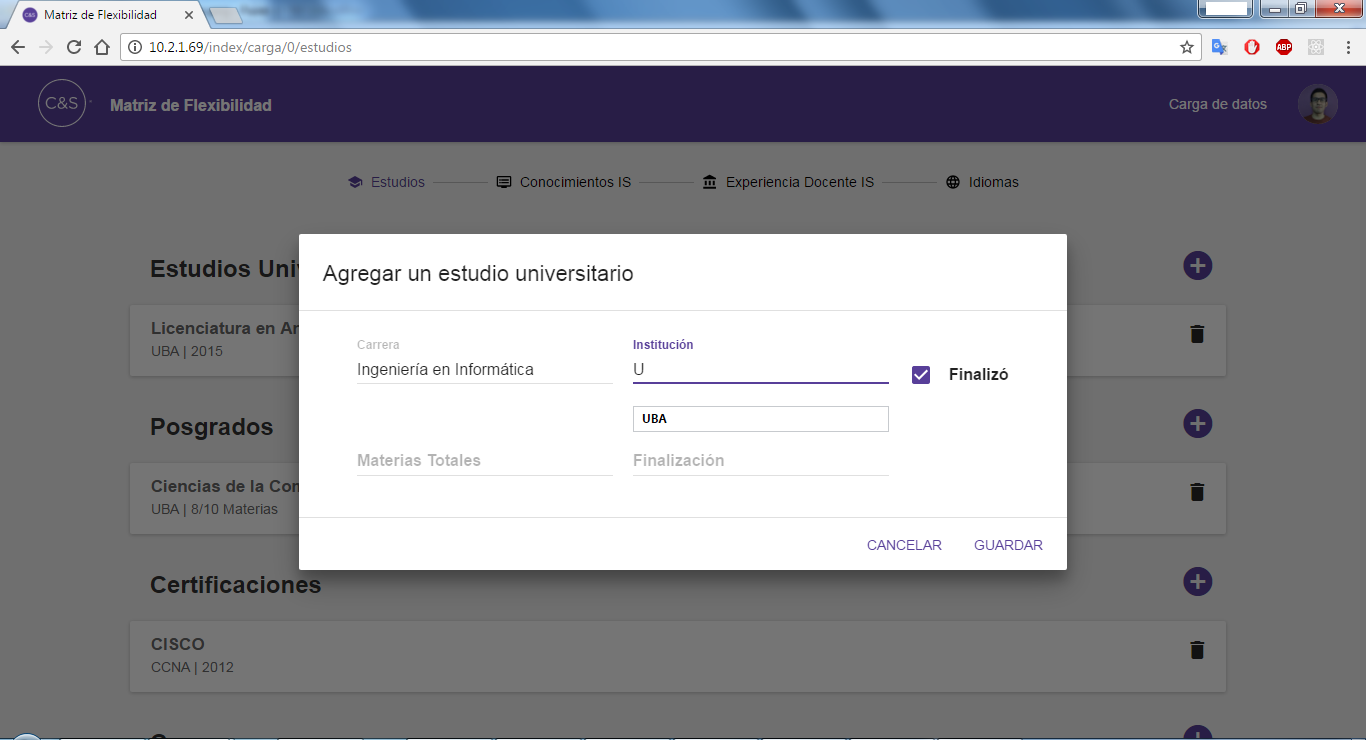
#### **Experiencia docente - categorías**



#### **Idiomas**



#### **Cargar un idioma**



#### **Sección Administrador**

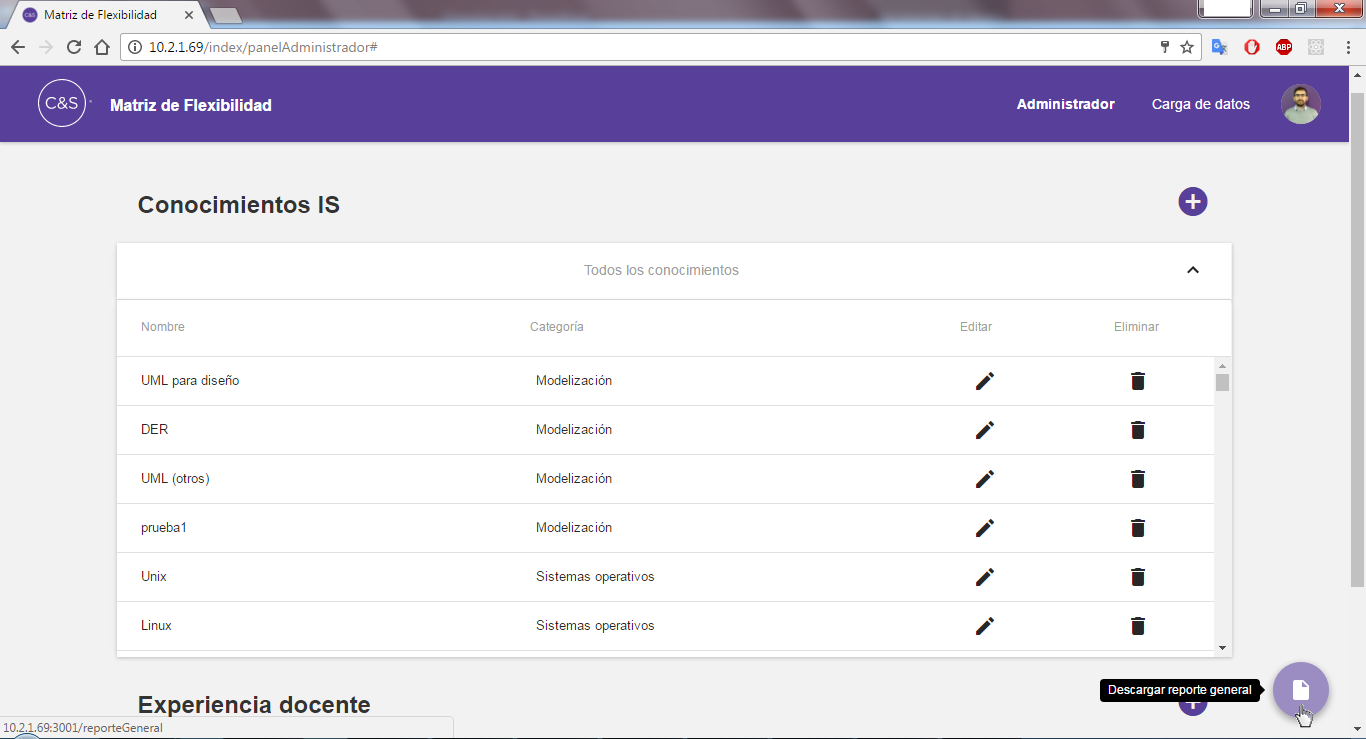
Es la sección de ABM de campos para luego ser llenados por los empleados

* Conocimientos en Ingeniería de Software
* Experiencia docente en Ingeniería de Software

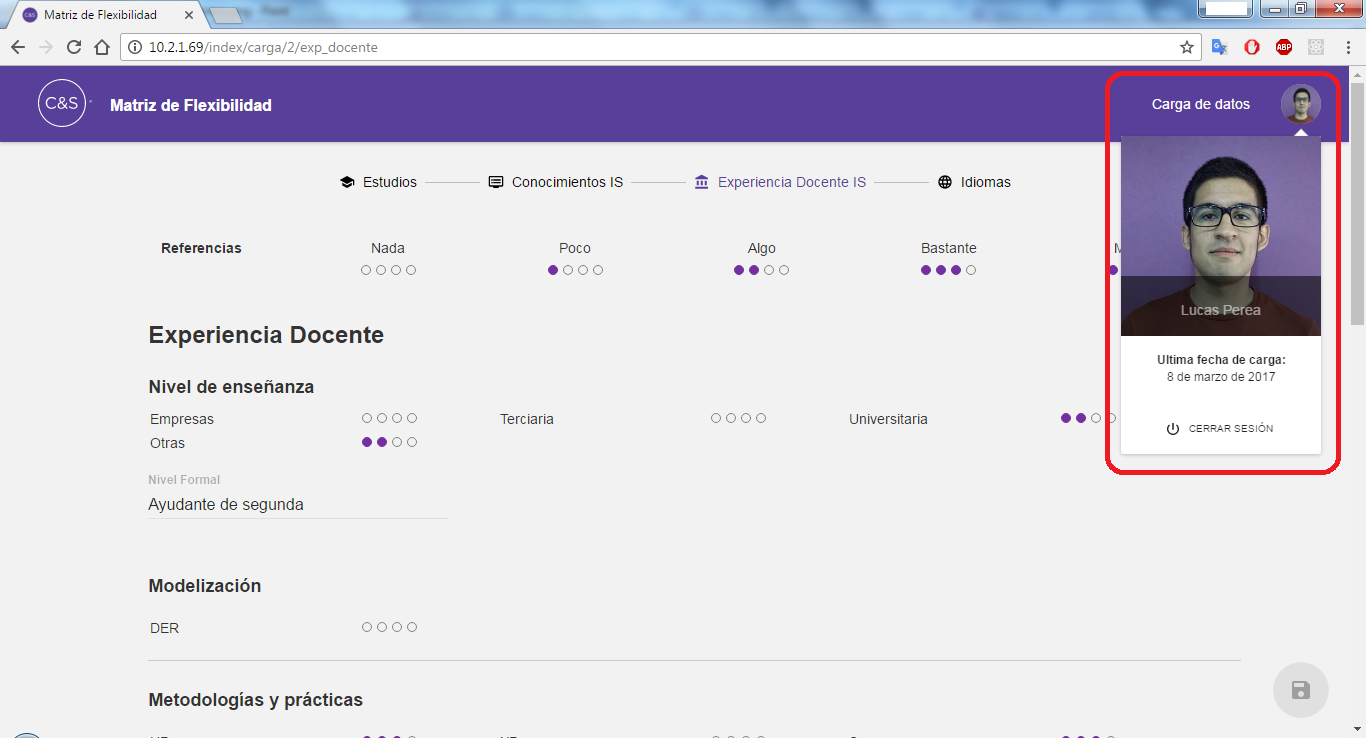
Además, podemos generar reportes

Ir: <http://10.2.1.69/index/panelAdministrador>

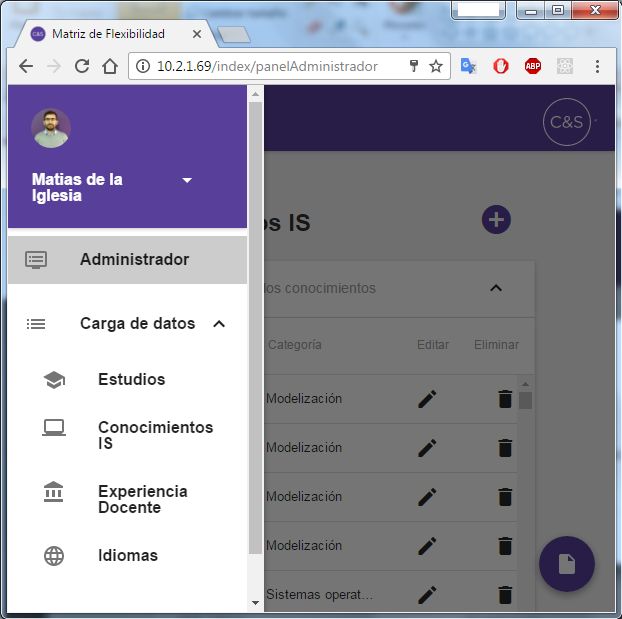
#### **Panel de administrador**



#### **Menú - pantalla grande**



#### **Menú - pantalla chica**



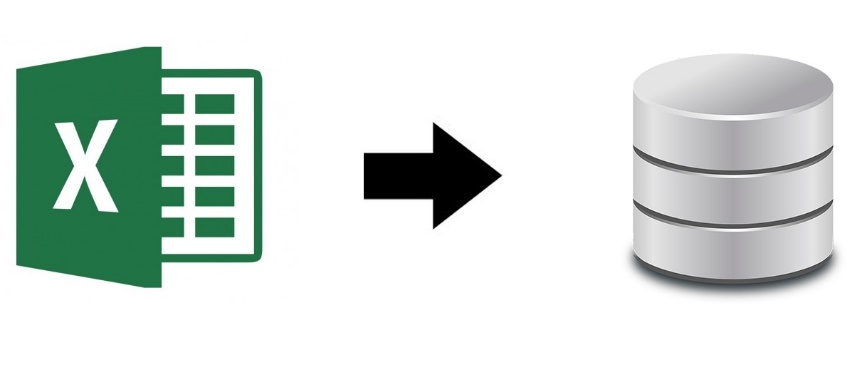
# Planes a futuro

## En proceso

* Testing (back-end, más que nada)

## Lo que falta

* Migración de datos



* Migración de MySQL a MongoDB



* Deploy a produccion

# Conclusiones

Usar tecnologías Javascript puede traer ventajas:

* Alto rendimiento
* Rápido renderizado, al usar el concepto de DOM Virtual.
* Alta escalabilidad
* Buena modularización, gracias a la arquitectura que brinda Redux.

Como tambien algunas desventajas:

* Curva de aprendizaje lenta
* Tiempo de desarrollo
* Muchas librerías y tecnologías diferentes, lo que requiere una buena investigación previa antes de desarrollar.

Se tuvo interacción con nuevas tecnologías de desarrollo web, generando un plus a C&S para futuros desarrollos o capacitaciones.