# Práctica 2

**Descripción:**

La práctica consiste en el desarrollo de un conjunto de servicios web (*Empleados, Seguridad, Utilidades*) que realizamos en la *Práctica 1*, solo que en lugar de hacer uso de contratos *WSDL,* haremos uso de contratos RAML que serán consumidos por una aplicación cliente.

Para el desarrollo haremos uso las siguientes herramientas:

- **XAMPP** como servidor de base de datos de MySQL consumible por el servidor.
- **Visual Studio 2019 Enterprise** como IDE de desarrollo tanto del servidor como del cliente que consume dichos servicios haciendo uso de los contratos publicados por los mismos.

Hemos desarrollado los diferentes servicios con sus accesibles métodos (aunque el funcionamiento es el mismo, los contratos y su forma de trabajar con ellos difiere):

**Empleados:** servicio que ofrece el CRUD básico de empleados

- **nuevo:** crea (si las entradas son correctas y no existe ya) un nuevo empleado
- **consultar:** devuelve un empleado (si existe en la DB) que coincida con el DNI
- **modificar:** modifica (si las entradas son correctas y existe) un empleado de la DB, devolviendo verdadero o falso según si se realizó la modificación
- **borrar:** borra (si existe en la DB) un empleado, devolviendo verdadero o falso según si se realizó el borrado

**Seguridad:** servicio que ofrece el CRUD básico entre permisos de empleados a salas

- **asignarPermiso:** se crea una relación entre una sala y el empleado otorgándole el permiso
- **validarUsuario:** devuelve verdadero o falso si el usuario tiene en la DB permiso a dicha sala
- **obtenerNiveles:** devuelve listado con todas las salas a las que tiene acceso el empleado
- **eliminarPermiso:** devuelve verdadero o falso si se eliminó el acceso a la sala indicada para el usuario indicado

**Utilidades:** servicio que ofrece el CRUD básico entre permisos de empleados a salas

- **validarNIF:** devuelve verdadero o falso si el dato de entrada (NIF) es válido
- **validarIBAN:** devuelve verdadero o falso si el dato de entrada (IBAN) es válido
- **validarNAFSS:** devuelve verdadero o falso si el dato de entrada (NAFSS) es válido

**Puesta en marcha:**

1. Iniciar XAMPP, iniciando servidor de MySQL con el puerto por defecto.
2. Iniciar proyecto de servidor con Visual Studio 2019 Enterprise.
3. Iniciar el cliente con Visual Studio 2019 Enterprise, y consumir el servicio web.

**Aspectos a destacar:**

- **No se ha incorporado la parte opcional del *SoapKey***.
- Se ha facilitado un *String* de error que proporciona información en aquellas llamadas que **en la especificación** lo requería. Además de no incluirlo en llamadas *True/False* como por ejemplo *ValidarIBAN*, donde el IBAN puede ser válido o no serlo, siendo esta respuesta *booleana* con suficiente autodescripción.
- Se han añadido ***más detalles de errores y mejoras de control***.
- **Se ha cambiado el dato de DateTime a String**, almacenando el número de *Ticks* (que en realidad es un tipo Long) como una cadena de texto debido a errores de conversión y para facilitar la implementación final.
- A pesar de no ser requerido ningún control real, cabe destacar que por dificultad y falta de tiempo **se han simplificado las funciones** de *ValidarNIF y ValidarNAFSS*, las cuales se comprueba el formato y no que sea real, es decir, 12345678A y 12345678B són válidos, sin embargo 1234567A o 1234567AA no lo son, al igual que se requiere un control mejorado del NAFSS (más real) pero simplemente que siga el siguiente formato (DDD-DD-DDDD) respetando guiones siendo cada D un valor numérico entre 0 y 9, y si no se respeta dicho formato se muestra un error, pero no se realizan comprobaciones de error. Sin embargo en *ValidarIBAN* se ha añadido una implementación correcta.
- Se ha añadido el uso de respuestas lo más parecidas a caso de uso real respetando el protocolo HTTP (GET/POST/PUT/DELETE y códigos como 200, 201 o 404).
- Se ha reciclado la interfaz del cliente anterior para respetar la interfaz de la práctica 1 y poder hacer una comparativa más limpia dirigida a funcionalidad y no a interfaz además de ahorrar tiempo de desarrollo.

**Dificultades encontradas:**

1. Complicaciones con RAML en el aspecto de respetar tabulaciones.
2. Complicaciones con tipos de datos que difieran de los más básicos (Date y DateTime por ejemplo).
3. Dificultad para entender errores con Visual Studio 2019, ya que la conexión devolvía errores sin devolver respuestas, o respondía con códigos de estado sin añadir la respuesta pasada como parámetro, etc).