**Diseño integral de la arquitectura técnica: especicación de los elementos del backend, tecnologías y relaciones entre ellos**

**Cliente.**

En nuestro sistema, el cliente estará compuesto por varios agentes; la aplicación móvil para el usuario, la app web para empleados y administrador, las pulseras con tecnología NFC y el lector de las mismas pulseras.

Nuestro cliente será creado con las tecnologías HTML, CSS, Angular2.js, el sistema será multiplataforma gracias a que usaremos el framework Ionic2 para ello.

Conectaremos con el servidor gracias al protocolo HTTP mediante peticiones POST, GET, PUT… Trataremos de utilizar en el mayor número de ocasiones, verbos idempotentes (PUT, GET, HEAD, DELETE...), que permiten recuperar cualquier petición que no haya llegado al servidor, simplemente repitiéndola.

**Servidor.**

Este será un servidor Apache y estará desarrollado con PHP, utilizará como hemos dichos anteriormente el protocolo HTTP y será de código abierto. Estará diseñado de modo que cumpla los principios REST, estilo arquitectónico de por excelencia de los servicios web.

**Base de datos.**

El sistema de gestión de bases de datos que utilizaremos será PostgreSQL este ofrece fiabilidad, integridad de datos y características integradas enfocadas al desarrollador. A su vez, permite unir consultas a tablas relativamente grandes de manera eficiente. Emplearemos para el diseño y gestión del sistema una única base de datos relacional o SQL que mostraremos más adelante en el modelo de datos.

**Relaciones entre las distintas entidades**

Entre los diferentes protocolos que usaremos (además del protocolo HTTP petición-respuesta ya nombrado), estarán DNS, NFS y SMTP.

En primer lugar, para la interpretación de las IP’s de los diferentes recursos a los que se quiere acceder utilizaremos el protocolo DNS.

Para el servicio de mailing entre cliente y servidor utilizaremos el protocolo SMTP como transferencia simple de correo entre las entidades.

En nuestro sistema se realizarán copias de seguridad periódicamente para poder restaurar nuestro sistema a un punto anterior y evitar grandes pérdidas en el sistema, para lo que utilizaremos el protocolo NFS.

**Especicación de elementos y tecnologías de seguridad y autenticación**

**Seguridad**

* No se puede acceder a todos los datos del sistema a no ser que se haga desde la cuenta de Administrador.
* Cifrado del saldo disponible en la cuenta de un usuario.
* El identificador del usuario también se encriptará, este constará del DNI del mismo usuario añadiendo un secreto a la combinación de números.
* El acceso a los distintos servicios estará restringido por los diferentes roles que hay en nuestro sistema.
* El diseño de la API no reflejará la estructura de BBDD para evitar posibles inyecciones SQL.
* La pulseras de uso serán solo de modo lectura, evitando que al ser de escritura y lectura algún usuario con conocimientos más avanzados pueda cambiar los identificadores de las mismas pulseras.
* Validación de los campos tanto en el lado del cliente como en el lado del servidor.
* Para mantener la sesión de usuario, se llevará a cabo el cifrado de login y contraseña en un fichero, de modo que cuando el usuario acceda de nuevo, pueda acceder a dichos datos de sesión y entrar sin autenticarse.

**Autenticación**

* Inicio de sesion con usuario y contraseña, que será cifrada con un conjunto de funciones hash criptográficas SHA512.
* La seguridad en el proceso de autenticación se va a basar en la implementación de token para garantizar la identidad del usuario.
* Cada vez que se mande una petición al servidor, se adjuntará el token actual, y el servidor devolverá uno nuevo. Dicho token, caducará tras un periodo determinado de tiempo. Por eso decimos que utilizaremos tokens de un solo uso.

**Especicación API Rest y del modelo de datos**

**Especificación API Rest**

**Login**

POST /rest/login -> Login usuario, empleado y administrador.

**Usuario**

GET/rest/usuario -> Listar todos los usuarios.

PUT/rest/usuario -> Crear un nuevo usuario.

GET/rest/usuario/id -> Detalle de un usuario.

PUT/rest/usuario/id -> Modificar un usuario (modificación parcial).

DELETE/rest/usuario/id -> Eliminar un usuario.

**Ocio**

GET/rest/ocio -> Listar todos los productos.

PUT/rest/ocio -> Crear un nuevo producto.

GET/rest/ocio/id -> Detalle de un producto.

PUT/rest/ocio/id -> Modificar un producto.

DELETE/rest/ocio/id -> Eliminar un producto.

PATH/rest/ocio/categoria -> Asignar una categoría a un producto.

PUT/rest/ocio/categoria -> Crear categorías de un producto.

DELETE/rest/ocio/categoria/id -> Eliminar categoría.

PUT/rest/ocio/categoria/id -> Modificar una categoría.

GET/rest/ocio/categoria -> Listar categorias de los productos

PATH/rest/ocio/estancia -> Asignar una estancia a un producto.

GET/rest/ocio/usuario ->Devolver el usuario asignado a un producto

**Empleado**

GET/rest/empleado -> Listar los empleados.

PUT/rest/empleado -> Crear un empleado.

GET/rest/empleado/id -> Detalle de un empleado.

PUT/rest/empleado/id -> Modificar un empleado.

DELETE/rest/empleado/id -> Eliminar un empleado.

**Estancia**

GET/rest/estancia -> Listar las estancias.

PUT/rest/estancia -> Crear una estancia.

GET/rest/estancia/id -> Detalle de una estancia.

PUT/rest/estancia/id -> Modificar una estancia.

DELETE/rest/estancia/id -> Eliminar una estancia.

**Accesoestancia**

GET/rest/accesoestancia -> Listar los accesos a una estancia.

PUT/rest/accesoestancia -> Crear un acceso a una estancia

GET/rest/accesoestancia/id -> Detalle de un acceso a una estancia.

DELETE/rest/accesoestancia/id -> Eliminar un acceso a una estancia.

**Pago**

GET/rest/pago -> Listar pagos (tickets).

PUT/rest/pago -> Crear pago (ticket).

GET/rest/pago/id -> Información de un pago (ticket).

DELETE/rest/pago/id -> Eliminar un pago.

GET/rest/pago/producto/id -> Detalles de un producto pagado.

PUT/rest/pago/producto -> Añadir un producto al tickets.

DELETE/rest/pago/producto/id -> Eliminar una producto de un tickets.

**Pulsera**

GET/rest/pulsera -> Listar pulseras.

PUT/rest/pulsera -> Dar de alta una pulsera.

GET/rest/pulsera/id -> Detalle de una pulsera.

PUT/rest/pulsera/id -> Modificar una pulsera.

DELETE/rest/pulsera/id -> Dar de baja una pulsera.

**Esquema de la base de datos relacional**

