ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA INFORMÁTICA

DISPOSITIVO DE SEÑALIZACIÓN PARA LOS ASIENTOS DE UNA BIBLIOTECA CON ARDUINO

SIGNPOSTING DEVICE FOR LIBRARY PLACES WITH ARDUINO

Realizado por
Felipe Ruiz Pinto

Tutorizado por
Dra. Ana María Cruz Martín
Dr. Juan Antonio Fernández Madrigal

Departamento
Ingeniería de Sistemas y Automática

UNIVERSIDAD DE MÁLAGA
MÁLAGA, JUNIO 2017

Fecha defensa:
El Secretario del Tribunal

AGRADECIMIENTOS

A mis tutores Ana María y Juan Antonio por ayudarme en todo lo que he necesitado y ser tan ‘mijitas’
A mi novia Desireé por ser mi apoyo fundamental en estos duros 4 años de carrera y estar siempre a mi lado.
A mis amigos de toda la vida Irene, Eloy, Juan Carlos y Miriam por ser los mejores amigos que alguien puede tener.
A mis compañeros de clase Joaquín, Alex, Juan Antonio y muchos otros por tan buenos momentos y experiencias juntos.
A José Francisco Chicano García, profesor de la ETSI de Ingeniería Informática, ya que gracias a su recomendación pude realizar unas prácticas extracurriculares que me han permitido tener actualmente un buen trabajo en el mundo de la informática.
A Pablo Medina Córdoba y José Antonio Vázquez Montiel por realizar la maqueta del dispositivo en 3D.
A María López Bujalance y Fuensanta López Pérez, directoras de la biblioteca de la ETSI Ingeniería Informática y Telecomunicación por aportar grandes ideas para el desarrollo de este trabajo.
A Javier Vázquez Peregrina, administrador de Ticsur Consultoría Estratégica S.L. y mi actual jefe, por proporcionar el servidor donde se intentó alojar en un principio el sistema.
Sin duda a quien más le debo agradecer es a mis padres, gracias por su esfuerzo, su sacrificio, su preocupación constante por mis estudios y por anteponerlos incluso a otras necesidades. No habría páginas suficientes en esta memoria para agradecer todo lo que hacen por mí.

A todos, gracias.

Resumen

Este trabajo describe el desarrollo de un dispositivo para la señalización de los asientos de las distintas bibliotecas de la Universidad de Málaga. Mediante el mismo, cada asiento indica su estado del mismo (libre, ocupado o reservado) mediante un led, lo que ayudará al alumno a una mejor identificación de los asientos libres en los periodos de máxima ocupación.

Además, se dispone de una aplicación web que muestra la disposición y estado de los asientos mediante un mapa de cada biblioteca y en la que el alumno puede realizar reservas. Solo necesitará el carnet de estudiante para poder utilizar tanto el dispositivo como la aplicación web.

Adicionalmente, se ha construido una segunda aplicación web para la administración de la primera. Está orientada al personal de la biblioteca, permitiéndoles crear y modificar los diferentes mapas de las bibliotecas y gestionar los usuarios, además de ofrecerles un módulo de estadísticas sobre las ocupaciones de los asientos.

Palabras claves

Biblioteca, mapa, Arduino, dispositivo, señalización, aplicación web

Abstract

This project involves the development of a signposting device for booking the places of the different libraries of University of Málaga. Each place has a device attached in that shows the state (free, occupied or booked) with a led, this will help to the student to a better identification of the free places in periods of high occupation.

In addition, there is a web application that displays the location and state of the places showing a map of each library in which the student can book a seat. He/She only needs his/her student card for use the device as well as the web application.

Additionally, a second web application for the management of the library has been developed to be used by the librarians, so they can create and modify any map of the libraries, manage the users, as well as obtain some statistical information about the occupation of the places.

Keywords

Library, map, Arduino, device, signposting, web application

A mis padres

ÍNDICE

CAPÍTULO 1 | INTRODUCCIÓN

Cada vez oímos más frecuentemente el término Internet de las Cosas [1][2][3], pero, ¿en qué consiste este término/tecnología? Pues bien, el IoT (*Internet of Things*), se define como la interconexión de objetos cotidianos y no tan cotidianos a Internet con el objetivo de proporcionar a los humanos información precisa sobre su entorno. Cada uno de los objetos conectados a Internet tiene una IP específica y mediante esa IP puede ser accedido pare recibir instrucciones. Asimismo, puede contactar con un servidor externo y enviar los datos que recoja.

Qué mejor modo que utilizar este concepto para facilitar la vida de los alumnos. Todos nosotros hemos sido estudiantes en algún momento y sabemos lo complicado que es poder ocupar un sitio en una biblioteca en épocas de exámenes y lo frustrante que es dar vueltas y vueltas y no poder sentarse. Este trabajo de fin de grado propone pretende solucionar esta problemática.

Como si de un aparcamiento se tratase, cada asiento tiene un dispositivo de señalización que indica el estado del mismo, es decir, si está libre, ocupado u reservado. El dispositivo de señalización mencionado está basado en Arduino [4][5][6], al cual se han conectado diferentes componentes que permiten la correcta identificación del estado de cada asiento. Dado que montar el sistema completo sobrepasa los límites del trabajo de fin de grado, solo se ha desarrollado un único dispositivo que representa un asiento; el estado de los demás asientos ha sido simulado.

También se ha implementado una aplicación web simple, que muestra un mapa de los asientos de la biblioteca y su estado actual, asemejándose a un mapa web de los asientos de una sala de cine, desde la cual el alumno puede consultar desde casa si hay algún asiento disponible, además de poder reservar un asiento durante un periodo de tiempo determinado.

Para la gestión de todo el sistema se ha desarrollado un panel de administración desde el cual se podrán modificar los diferentes mapas de las bibliotecas, gestionar usuarios y obtener estadísticas de las ocupaciones, entre otras funcionalidades. Su uso está destinado al personal de biblioteca y/o a los administradores del sistema.

1.1 | Motivación

Con el sistema aquí descrito se pretende solucionar un incómodo problema presente en cualquier biblioteca, como es monitorizar la afluencia de alumnos en períodos de máxima ocupación, en los cuales las bibliotecas se ven desbordadas y poder encontrar un asiento libre se convierte en una ardua tarea.

No solo se pretende ayudar al alumno, sino también al personal bibliotecario mediante estadísticas que ayuden a una mejor distribución de los recursos de cada biblioteca, sin olvidar que un sistema como el desarrollado supondría una gran modernización de las instalaciones.

1.2 | Objetivos

El principal objetivo de este TFG ha sido desarrollar un sistema que permita tanto a alumnos como a personal de biblioteca identificar de un modo más sencillo, rápido y eficaz la disponibilidad de asientos de cualquier biblioteca de la Universidad de Málaga.

Concretamente nos marcamos los siguientes objetivos:

- **Desarrollo de un dispositivo basado en Arduino**, el cual estaría colocado en cada uno de los asientos de las bibliotecas e indicaría el estado del mismo. Además, se conectaría, mediante un módulo WiFi, a una base de datos para almacenar la información sobre las ocupaciones.

- **Desarrollo de una aplicación web** para presentar la información de cada dispositivo. En ella se mostraría un mapa con la distribución de los asientos de cada biblioteca que indicaría, mediante un código de colores, el estado de cada uno de ellos. Contaría con una sección de uso exclusivo por el personal de biblioteca mediante la cual se podría gestionar todo el sistema (usuarios, gestión de los mapas y de las diferentes bibliotecas, estadísticas, etc).

La arquitectura general del sistema sería la mostrada en la Figura 1

Figura 1: Arquitectura del sistema.

1.3 | Metodología utilizada

Este proyecto puede ser dividido en dos partes bien definidas: por un lado, tenemos todo lo relacionado con el dispositivo señalizador, y por el otro, el desarrollo de la aplicación web. Partiendo de esta idea, las dos partes pueden realizarse en paralelo. Para ello hemos seguido un modelo en cascada (Figura 2) [7].

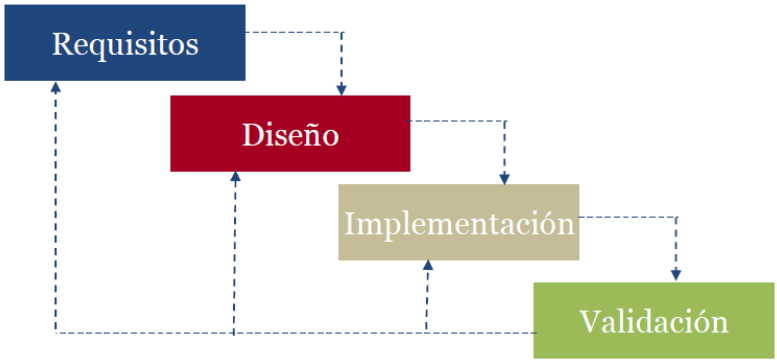

Figura 2: Modelo de desarrollo en cascada.

El modelo en cascada resulta adecuado cuando los requisitos y las fases están bien definidas, es decir, si no es un proyecto cambiante, por lo que resulta perfecto para este trabajo.

1.4 | Medios materiales

El desarrollo del sistema implica tanto desarrollo hardware como software. Todos los elementos hardware han sido adquiridos y costeados íntegramente por el alumno. En cuanto al software utilizado no ha sido necesaria la compra de ningún producto. A continuación, se listan todos los elementos utilizados:

Hardware

- Ordenador personal HP Pavilion G6 2004ss con 12 GB de RAM, Intel Core i7 de tercera generación y tarjeta gráfica AMD Radeon 7600M HD. El mismo PC se ha utilizado como servidor para alojar la aplicación web.
- Placa Arduino UNO. [8]
- Módulo Wifi ESP8266. [9]
- Placa de expansión USB. [10]
- Lector de código de barras. [11]
- Pantalla LCD de 128x64 píxeles compatible con Arduino. [12]
- Bombilla LED RGB. [13]
- Cables y resistencias. [14][15]

- Protoboard. [16]

En el Capítulo 4 se describe con más detalle la funcionalidad de cada uno de estos componentes y su implicación en el dispositivo.

Software

- Netbeans IDE 8.0.2[27] con el plugin de PHP [17][18][19] para el desarrollo del código de la aplicación web.
- Arduino IDE 1.8.1[28] para desarrollar el programa que controla cada dispositivo.
- Servidor web Apache [20][21] para alojar la aplicación web.
- Servidor de base de datos MySQL. [22][23]
- Git 2.10.0 para el control de versiones tanto del código del dispositivo como de la aplicación web.
- phpMyAdmin para el acceso a la base de datos de un modo más amigable.

1.5 | Fases del trabajo

En cualquier proyecto es fundamental una buena planificación; de ella dependerá el éxito o el fracaso del mismo. Para el desarrollo de este proyecto se propuso inicialmente la que se observa en la Figura 3.

Nombre de tarea
1-Documentación, elección y compra de Arduino y componentes
2-Elaboración del mapa de sitios disponibles de la biblioteca de la ETSII
3-Desarrollo de la aplicación web
3.1-Definición de la base de datos
3.1.1-Análisis de las entidades
3.1.2-Análisis de las relaciones
3.1.3-Implementación de la estructura de la base de datos en Microsoft SQL
3.1.4-Introducción de datos en la DB
3.2-Implementación de las diferentes vistas de la aplicación
3.2.1-Vista de login
3.2.2-Implementación de la vista con el mapa de la biblioteca
3.2.3-Conexión del mapa con la base de datos
4-Montaje e implementación del dispositivo señalizador de sitios
4.1-Conexión del shield usb e implementación de la lectura de códigos de barras
4.2-Configuración de la conexión wifi y conexión a la DB
4.3-Conexión del sensor de proximidad y programación del mismo
4.4-Conexión e implementación de la pantalla LCD.
4.5-Montaje del circuito para el led de señalización
5-Elaboración de la documentación del software y hardware desarrollado y de la memoria del trabajo

Figura 3: Planificación inicial de tareas.

Sin embargo, a medida que se sucedían algunas etapas y se encontraban posibles mejoras, modificaciones y/o dificultades sobre el diseño inicial del sistema fue necesaria la modificación de la planificación, quedando finalmente como se muestra en la Figura 4.

Nombre de tarea
1-Documentación, elección y compra de Arduino y componentes
2-Elaboración del mapa de sitios disponibles de la biblioteca de la ETSII
3-Desarrollo de la aplicación web
3.1-Definición de la base de datos
3.1.1-Análisis de las entidades
3.1.2-Análisis de las relaciones
3.1.3-Implementación de la estructura de la base de datos en MySQL
3.1.4-Introducción de datos en la DB
3.2-Implementación de la aplicación principal
3.2.1-Vista de login
3.2.2-Implementación de la vista con el mapa de la biblioteca
3.2.3-Conexión del mapa con la base de datos
3.3 - Implementación de la aplicación de administración
3.3.1-Módulo gestión usuarios
3.3.2-Módulo gestión de mapas
3.3.3-Módulo gestión de estadísticas
3.3.4-Módulo gestión de bibliotecas
3.3.5-Módulo incidencias
4-Montaje e implementación del dispositivo señalizador de sitios
4.1-Conexión del shield usb e implementación de la lectura de códigos de barras
4.2-Configuración de la conexión wifi y conexión a la DB
4.3-Conexión e implementación de la pantalla LCD.
4.4-Montaje del circuito para el led de señalización
5-Elaboración de la documentación del software y hardware desarrollado y de la memoria del trabajo

Figura 4: Planificación definitiva.

Respecto a la planificación inicial se pueden observar los siguientes cambios:

- **Implementación de la estructura de la base de datos en MySQL**: se pensó realizar la aplicación con una base de datos Microsoft SQL Server debido a que el sistema iba a estar alojado en un servidor con esta tecnología instalada. El servidor lo iba a proporcionar la empresa *Ticsur Consultoría Estratégica S.L*, donde el alumno trabaja actualmente, pero finalmente no fue posible y se optó por MySQL.

- **Implementación de la aplicación de administración**: en un primer momento no se previó la necesidad de una aplicación de administración. Sin embargo, el desarrollo de la misma ha supuesto obtener un sistema totalmente completo y funcional.

- **Conexión del sensor de proximidad y programación del mismo**: el diseño inicial del dispositivo incluía un sensor de proximidad para activar y desactivar el lector de código de barras cuando el alumno acercaba/retiraba su identificación. Esta tarea suponía una gran dificultad y finalmente se suprimió para no ralentizar el desarrollo de las demás fases. La dificultad residía en poder activar y desactivar la placa de expansión ya que a ella están conectados todos los demás componentes, por lo que, si se apagaba, se apagaría el dispositivo completo, incluido el mismo sensor.

1.6 | Consideraciones iniciales

El trabajo que se ha realizado nace por la necesidad de contar con un sistema que facilite a los usuarios de la biblioteca de la Universidad de Málaga el acceso e identificación de asientos libres, tanto en el interior de la biblioteca como desde el exterior vía web.

A continuación, se dan unas nociones básicas para que el lector comprenda el funcionamiento del sistema:

- En cada asiento habrá un dispositivo que señalizará el estado del mismo. Cada dispositivo está conectado a Internet y envía la información del estado a una base de datos.

- Hay disponible una aplicación web para mostrar dicha información mediante un plano de cada biblioteca y en la cual el usuario puede hacer reservas de asientos.

- La reserva dura una hora; cuando se haya cumplido el tiempo, el asiento volverá a estar libre.

- Todo el mundo puede ver el estado de los asientos vía web, pero solo los alumnos y personal pueden hacer reservas. Para ello deberán iniciar sesión en la aplicación con las credenciales que se le proporcionen vía email una vez se realice su registro en el sistema.

- Para ocupar/liberar un asiento es necesario el carnet universitario. El dispositivo está provisto de un lector de código de barras por el que deberán pasar su carnet.

- Un usuario solo podrá ocupar o reservar un asiento al mismo tiempo.

- La ocupación de cada asiento dura 3 horas, aunque el alumno puede volver a ocupar el asiento siempre que alguien no lo haya reservado antes.

- Las reservas siempre tienen prioridad; un alumno sentado en un asiento que esté reservado y no haya reservado él deberá dejar el puesto al usuario que realmente hizo la reserva.

- Los alumnos tendrán prioridad sobre personas externas a la universidad. Una persona que no pertenezca a la universidad deberá dejar el puesto a un alumno que se identifique correctamente en el dispositivo mediante el carnet universitario.

- Mediante la aplicación de administración se podrán gestionar usuarios, crear y modificar bibliotecas, así como sus planos y disponer de un módulo de estadísticas basado en las ocupaciones, es decir, cuando un alumno se sienta, que servirá a los empleados para poder optimizar los recursos de las bibliotecas.

- El nombre de este sistema es **Librarino** (Figura 5).

Figura 5: Logo del sistema.

Aquí no solo se ha definido y desarrollado el sistema, sino que se han aportado una serie de normas básicas que deberán acatar los usuarios para el buen funcionamiento y uso de la plataforma.

1.7 | Contenido de la memoria

En el Capítulo 2 se estudian algunos proyectos o trabajos similares, comparando sus ventajas e inconvenientes respecto al que se ha desarrollado en esta memoria. También se da una descripción de las distintas tecnologías que se han utilizado tanto en el desarrollo de la aplicación web como en el dispositivo.

El desarrollo del dispositivo, sus componentes y conexiones se estudian en el Capítulo 3, mientras que el desarrollo de la aplicación web, casos de uso, arquitectura y el modelo de datos lo podemos encontrar en el Capítulo 4.

Las diferentes pruebas realizadas para comprobar el correcto funcionamiento del sistema se describen en el Capítulo 5.

En el último capítulo, se explica, en forma de conclusiones, cómo surgió la idea para este proyecto y qué ha supuesto la realización del mismo en lo que a nuevos conocimientos se refiere. Además, se incluyen una serie de trabajos futuros que se podrían realizar para completar aún más el sistema.

Es importante, en cualquier trabajo o proyecto software, desarrollar un manual de usuario. En este caso se puede encontrar en el Apéndice A, separando el manual en los diferentes roles que intervienen en el sistema.

Por último, en el Apéndice B, se dan las instrucciones de instalación y configuración tanto de los entornos de programación utilizados para el desarrollo del código fuente del sistema, como de la aplicación en sí.

CAPÍTULO 2 | ESTADO DEL ARTE Y TECNOLOGÍAS UTILIZADAS

A lo largo de este capítulo se analizan una serie de trabajos similares al que se ha desarrollado, comentando también las ventajas e inconvenientes de cada uno con respecto a Librarino. En la segunda parte del capítulo se recorren las tecnologías que se han utilizado para construir el sistema.

2.1 | Estado del arte

Actualmente existen numerosos trabajos y patentes relacionados en mayor o menor medida con la temática de este TFG. Todos ellos tienen el mismo denominador común: el control o monitorización de los asientos de las bibliotecas. A continuación se describe el funcionamiento de algunos de ellos.

La primera de las patentes, *Library self-service seat selection system*, [24] consiste en un sistema en el cual el alumno deberá identificarse para entrar en la biblioteca a través de su carnet universitario. A continuación, seleccionará el asiento donde quiere sentarse desde un gran monitor a la entrada de la biblioteca que muestra el estado de los asientos libres (color verde) y ocupado (color rojo). Al salir de la biblioteca deberá identificarse de nuevo y se liberará el asiento donde estaba sentado.

Este sistema, para comprobar el estado de los asientos es necesario estar *in situ* en la biblioteca, además de no poder reservar un asiento fuera de ella. Otro de los inconvenientes es que personas que no sean alumnos o personal de la universidad no podrían entrar en la biblioteca. Librarino permite a dichos usuarios poder entrar a la biblioteca, aunque siempre tendrán preferencia los alumnos de la universidad.

En otra patente, *Library seat management system*, [25] no es necesaria la identificación del alumno, sino que es a través de un sensor infrarrojo dispuesto en cada uno de los asientos como se lleva a cabo la identificación del estado. Todos los sensores están comunicados mediante un circuito de control y un procesador central muestra la información en una pantalla a la entrada de la sala de lectura.

Es cierto que el modo de ocupar un asiento es más sencillo, pero es más susceptible a trampas por parte de los alumnos, ya que colocando algún objeto frente al sensor estaríamos ocupando el asiento, por lo que el alumno podría reservar varios asientos durante varias horas sin estar en la biblioteca, cosa que con nuestro sistema no sería posible. Además, tampoco permite las reservas y monitorización de los asientos fuera de las instalaciones.

Otra de las patentes, *Library seat management system*, [26] consiste en un sistema inalámbrico de dispositivos instalados en cada uno de los asientos. El estudiante introduce su identificación en el dispositivo, el cual transmite la información de manera remota al ordenador principal de la sala, que muestra el estado de cada uno de los asientos. También es necesaria la identificación a la entrada y salida de la biblioteca.

Esta última patente es muy parecida a la primera mencionada, solo cambia el modo de ocupación de los asientos, por lo que presenta los mismos inconvenientes frente a Librarino.

También podemos destacar el sistema del que dispone la *Seoul National University* (Corea) para la monitorización del estado de los asientos. En la entrada de la biblioteca principal hay un control de acceso en el cual el alumno debe pasar su carnet de estudiante para poder entrar. Una vez en el interior, hay disponibles unos monitores con un plano de la biblioteca que muestra el estado de los asientos (similar al propuesto en este trabajo) en el que el alumno selecciona el asiento que quiere ocupar, y, de nuevo mediante el carnet universitario, hace efectiva la ocupación del asiento. Como podemos ver, este sistema es muy parecido al nombrado en la primera patente.

Sin tener que salir de nuestro país, la Universidad de la Rioja está llevando a cabo un proyecto similar. Su funcionamiento es bastante sencillo:

- En primer lugar, el usuario entra en la biblioteca y ocupa físicamente el asiento.
- Cuando se quieren levantar (para descansar, ir al baño, etc), se dirigen al monitor principal y reservan el asiento en el que están sentados. El sistema les confirma la reserva y la hora límite de la misma.
- Los bibliotecarios pueden acceder a la base de reservas y conocer en tiempo real el estado de las mismas.

De nuevo, este sistema presenta algunos inconvenientes ya descritos, como son el no poder ver el estado de los asientos ni reservar fuera de la biblioteca.

Como se observa, la problemática de controlar el estado de los asientos de una biblioteca ha sido ya abordada en muchos proyectos que aportan diferentes ideas o soluciones. En este trabajo se ha querido aportar un nuevo punto de vista, dotando al sistema de monitorización y reservas vía web para mejorar la experiencia del usuario.

2.2 | Tecnologías utilizadas

Un sistema como el que se ha desarrollado requiere de la utilización de múltiples tecnologías tanto hardware como software que lleven a cabo todas y cada una de las funcionalidades.

El dispositivo físico está basado íntegramente en la popular plataforma de desarrollo electrónico Arduino [4][5][6], utilizando diferentes componentes para adaptarlo a las necesidades del trabajo.

Es en la aplicación web donde han convergido un mayor número de tecnologías. Para la construcción de las diferentes vistas de la aplicación web se ha utilizado HTML [29][30] combinado con CSS [31][32] y Bootstrap [33][34] para añadir el estilo y la maquetación a las páginas. El contenido dinámico de las vistas se ha generado combinando Javascript [35][36], jQuery [37][38] y PHP; este último ha sido utilizado también para definir toda la lógica y el back-end de la aplicación. Por último, se ha utilizado una base de datos MySQL para proporcionar el soporte de almacenamiento del sistema. Esta combinación no ha sido aleatoria, y es que PHP y MySQL es una buena combinación para construir una aplicación ligera, pero con cualquier funcionalidad que debamos desarrollar.

Para completar nuestro sistema, se ha utilizado un servidor web Apache para el alojamiento de la aplicación web, al ser de código abierto, multiplataforma y que goza de una gran fama por su buen rendimiento.

2.2.1 | Arduino

Se ha elegido esta plataforma dado que, al ser de código abierto, es uno de los más utilizados en el Internet de las cosas. Además, cuenta con una de las comunidades más grandes de desarrolladores independientes por lo que está en continuo desarrollo y evolución.

Arduino es una plataforma de desarrollo electrónico basada en hardware y software flexibles y fáciles de usar. Se presenta físicamente en forma de placa, basada en el microcontrolador ATMEL. En dicho microcontrolador se graban una serie de instrucciones que permiten controlar e interactuar con los circuitos electrónicos. El lenguaje de programación utilizado para escribir dichas instrucciones es un lenguaje derivado del C.

Arduino cuenta con entradas y salidas tanto digitales como analógicas, además de para protocolos de comunicación. En su placa impresa tiene todos los elementos necesarios para hacer funcionar el microcontrolador, pines de entrada y salida, y comunicación serie.

La comunicación serie es un protocolo de comunicación utilizado en las antiguas computadoras que disponían de puerto serie. Actualmente Arduino se comunica con los

PCs a través de USB, para ello utiliza un convertidor de serie a USB, aunque el PC lo detectará en el puerto serie.

Para extender la funcionalidad de la placa, Arduino cuenta con una gran cantidad de sensores, módulos y *shields* o placas de expansión. Un *shield* no es más que una placa compatible con Arduino que se coloca encima encajando los pines de ambas placas.

Los *shields* se pueden comunicar con Arduino bien por algunos de los pines digitales o analógicos o bien por algún bus como el SPI, I2C o puerto serie, así como usar algunos pines propios de la placa Arduino. Además, estos shields se alimentan generalmente a través del Arduino mediante los pines de 5V y GND.

En cuanto a los módulos y sensores, se comunican a través de los pines digitales y analógicos de la placa simplemente conectándolos a través de cables. Hay multitud de ellos y con funciones muy variadas, desde módulos wifi (como el que utiliza el dispositivo) hasta sensores de gas propano.

Para la utilización de algunos componentes adicionales es necesario añadir algunas bibliotecas. La mayor parte de estas bibliotecas están desarrolladas por programadores independientes a la empresa comercializadora de Arduino. Todas son de código abierto y podremos descargarlas, junto con su código completo, desde GitHub o plataformas similares.

Figura 6: Conexiones de la placa Arduino.

Conexiones de la placa

Como se ha dicho anteriormente, Arduino cuenta con una serie de entradas y salidas para aumentar su funcionalidad y llevar a cabo proyectos complejos. En la Figura 6 se pueden observar detalladamente todas las conexiones de la placa. A continuación, se describen las más importantes.

- **Alimentación:** puede alimentarse mediante la conexión USB (proporciona 5 V) o a través del jack de alimentación, el cual puede ser una pila de 9 V o una fuente de alimentación de entre 7 y 12 V.
- **Pines de alimentación:**
    - **3.3 V:** proporciona una tensión de salida de 3.3 V, y una intensidad máxima de 50 mA.
    - **5 V:** proporciona una tensión de salida de 5 V, y una intensidad máxima de 300 mA.
    - **GND:** toma de tierra.
    - **Vin:** proporciona la tensión máxima con la que está alimentada la placa.

- **Pines de entradas y de salida:** cuenta con 14 pines digitales y 6 analógicos, la intensidad máxima de todos estos pines es de 40 mA. En estos pines conectaremos los diferentes componentes del dispositivo.
    - **Digitales:** pueden suministrar o aceptar (depende del modo del pin) entre 0 V y 5 V.
    - **Analógicos:** los valores de salida van desde 0 V a 5 V en un rango de 0 a 255 (precisión de 8 bits), mientras que los valores de entrada oscilan también entre los 0 V y los 5 V en un rango de 0 a 1023 (precisión de 10 bits)**.**

Los inicios de Arduino se remontan al año 2005, donde nació como un proyecto para estudiantes en el Insituto IVREA en Italia. Por aquel entonces los estudiantes utilizaban el microcontrolador BASIC Stamp, pero su coste de 100 dólares lo convertía en un producto demasiado caro para tareas de desarrollo estudiantil. Massimo Banzi, uno de los fundadores de Arduino trabajaba en dicho instituto.

El proyecto contó con la colaboración un estudiante colombiano, Hernando Barragán, quién desarrollo la tarjeta electrónica Wiring, el lenguaje de programación y la plataforma de desarrollo.

Diferentes investigadores trabajaron para hacerlo más ligero, económico y disponible para la comunidad de código abierto. Banzi afirmó años más tarde que el proyecto no nació como una idea de negocio, sino para intentar subsistir ante el inminente cierre del instituto.

Para la producción en serie de la primera versión se tomó en cuenta que el coste no fuera mayor de 30 euros, que fuera ensamblado en una placa de color azul, debía ser Plug and Play y que trabajara con todos los sistemas operativos. Las primeras 300 unidades se las dieron a los alumnos del Instituto IVREA, con el fin de que las probaran y empezaran a diseñar sus primeros prototipos.

Al proyecto se incorporó el profesor Tom Igoe, el cual ofreció su apoyo para desarrollar el proyecto a gran escala. En la feria Maker Fair de 2011 se presentó la primera placa Arduino 32 bit para realizar tareas más pesadas.

2.2.2 | HTML

HTML significa "Lenguaje de Marcado de Hypertexto" por sus siglas en inglés *HyperText Markup Language*. El estándar HTML lo define la W3C (World Wide Web Consortium) y actualmente HTML se encuentra en su versión HTML5.

Cabe destacar que HTML no es un lenguaje de programación ya que no cuenta con funciones aritméticas, variables o estructuras de control propias de los lenguajes de programación: genera únicamente páginas web estáticas. Sin embargo, HTML se puede usar en conjunto con diversos lenguajes de programación para la creación de páginas web dinámicas.

HTML se encarga de desarrollar una descripción sobre los contenidos que aparecen como textos y sobre su estructura, complementando dicho texto con diversos objetos (como fotografías, animaciones, etc).

Es un lenguaje muy simple y general que sirve para definir otros lenguajes que tienen que ver con el formato de los documentos. Para la escritura de este lenguaje, se crean etiquetas que aparecen especificadas a través de corchetes o paréntesis angulares.

El norteamericano Tim Berners-Lee fue el primero en proponer una descripción de HTML en un documento que publicó en 1991. Allí describía veintidós componentes que suponen el diseño más básico y simple del HTML.

Entre los recursos que pueden enlazarse al código HTML se encuentran fotografías, vídeos, archivos de otras webs o incluso de la misma y todo tipo de contenido que se encuentre subido a la red.

2.2.3 | CSS

CSS, siglas en inglés de *Cascading Stylesheets*, es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de las páginas web e interfaces de usuario escritas en HTML o XHTML.

CSS está diseñado principalmente para marcar la separación del contenido del documento de su forma de presentación. Esta separación busca mejorar la accesibilidad del

documento, permitir que varios documentos HTML compartan un mismo estilo usando una sola hoja de estilo, y reducir la complejidad y la repetición de código.

En resumen, al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para crear los contenidos. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

2.2.4 | JAVASCRIPT

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar el código.

Es un lenguaje con muchas posibilidades, pudiendo ser utilizado para crear pequeñas funciones y efectos hasta grandes objetos mucho más complejos. Tiene la ventaja de ser incorporado en cualquier página web y puede ser ejecutado sin la necesidad de instalar otro programa para ser visualizado.

Sin embargo, no solo es capaz de ejecutarse en el lado del cliente, sino que existe otro JavaScript que se ejecuta en el servidor, denominado LiveWire JavaScript. En este trabajo se ha utilizado íntegramente la modalidad del lado del cliente.

Esta tecnología puede encontrarse en la inmensa mayoría de las páginas que visitamos diariamente, desde el correo electrónico hasta buscadores de información. JavaScript puede ser utilizado para insertar elementos dinámicos de diferente índole, como por ejemplo relojes, contadores de visitas, validadores de formularios, etc.

El código JavaScript se puede identificar en el cuerpo del documento HTML entre las etiquetas <script></script>. Puede ser insertado en el mismo documento o importado desde un archivo con extensión .js, de este modo conseguimos reaprovechar el código JavaScript para varias páginas HTML.

En los años 90, Netscape creo Livescript; las primeras versiones de este lenguaje fueron principalmente dedicadas a pequeños grupos de diseñadores web que no necesitaban utilizar un compilador, o sin ninguna experiencia en la programación orientada a objetos.

En diciembre de 1995, Netscape y Sun Microsystems (el creador del lenguaje Java) luego de unirse con el objetivo de desarrollar el proyecto en conjunto, reintroducen este lenguaje con el nombre de Javascript.

La estandarización de Javascript comenzó en conjunto con ECMA en noviembre de 1996. Es adoptado como estándar en junio de 1997 y luego también por la "Internacional Organization for Standardization" (ISO).

jQuery

jQuery es una biblioteca multiplataforma de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con AJAX.

Ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio

La ventaja principal de jQuery es que es mucho más simple que sus competidores, pudiendo agregar plugins fácilmente, traduciéndose esto en un ahorro substancial de tiempo y esfuerzo.

2.2.5 | AJAX

AJAX [39][40] es el acrónimo de *Asynchronous Javascript and XML*, es decir: Javascript y XML Asíncrono. AJAX es una técnica que permite que un servidor y un navegador intercambien información, posiblemente en XML (aunque también es posible en otros formatos como JSON), de forma asíncrona.

Permite que una página web que ya ha sido cargada solicite nueva información al servidor. Con AJAX no es necesario recargar toda la página web, como ocurre cuando pinchamos en un enlace o cuando pulsamos el botón *submit* de un formulario. Es posible realizar una conexión a un servidor desde dentro de una página web usando un programa Javascript. Dicho servidor enviará una respuesta; esta respuesta se almacenará en una variable del programa Javascript y, una vez almacenada en la variable, podremos hacer con ella lo que deseemos. En la Figura 7 se compara el modelo clásico de aplicaciones frente al modelo basado en AJAX.

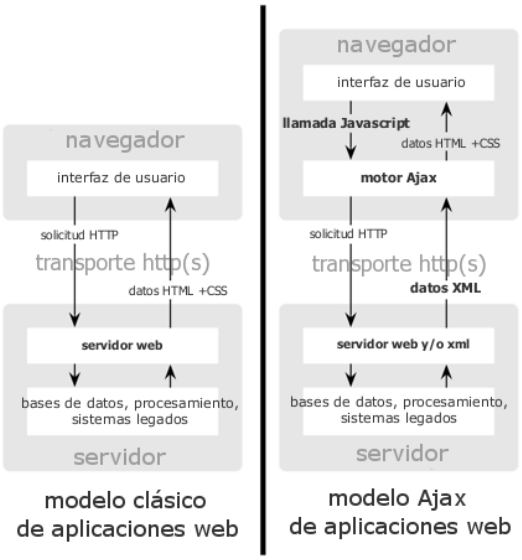

Figura 7: Modelo clásico vs modelo AJAX [39].

2.2.6 | PHP

PHP (*Hypertext Preprocessor*) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser embebido en HTML.

Este lenguaje basado en scripts, al contrario que JavaScript, se ejecuta en el lado del servidor, generando HTML y enviándolo al cliente. Con esta técnica se consigue que el cliente únicamente reciba el resultado de ejecutar el script, por lo que no puede conocer el código PHP subyacente. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. La Figura 8 muestra de un modo más visual el ciclo de una petición HTML en una aplicación basada en PHP.

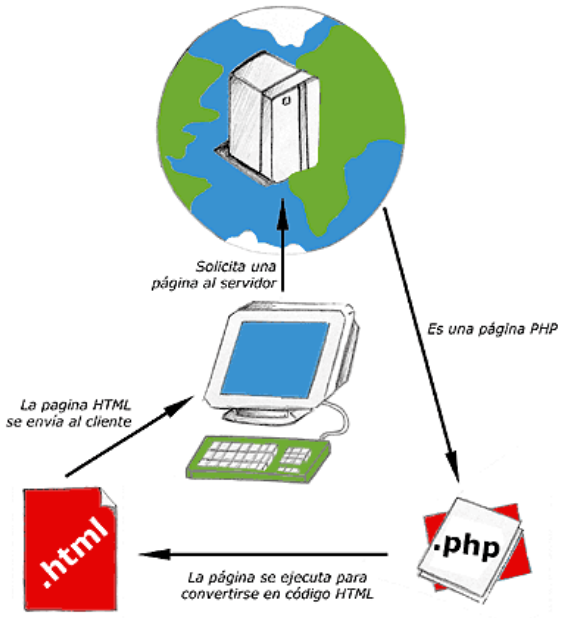

Figura 8: Petición a una aplicación web basada en PHP [19].

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos, conexiones de red, envío de correo electrónicos, creación y modificación de hojas de Excel, creación dinámica de imágenes, subida de archivos y una lista interminable de utilidades adicionales.

Además PHP es compatible con las bases de datos más comunes, como MySQL, Oracle, Microsoft SQL Server, Informix, entre otras.

Fue originalmente diseñado en Perl, con base en la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vítae y guardar ciertos datos. El 8 de junio de 1995 fue publicado "Personal Home Page Tools" después de que Lerdorf lo combinara con su propio Form Interpreter para crear PHP/FI.

Dos programadores israelíes del Technion, Zeev Suraski y Andi Gutmans, reescribieron el analizador sintáctico en 1997 y crearon la base del PHP3, además cambiaron el nombre del lenguaje por PHP: Hypertext Preprocessor. Inmediatamente comenzaron experimentaciones públicas de PHP3, y se publicó oficialmente en junio de 1998. Para 1999, Suraski y Gutmans reescribieron el código de PHP, y produjeron lo que hoy se conoce como motor Zend.

2.2.7 | MySQL

MySQL es un sistema de gestión de base de datos relacional o SGBD multihilo y multiusuario, lo que le permite ser utilizado por varias personas al mismo tiempo, e incluso, realizar varias consultas a la vez, lo que lo hace sumamente versátil.

La mayor parte del código se encuentra escrito en lenguaje C/C++ y la sintaxis de su uso es bastante simple, lo que permite crear bases de datos complejas con mucha facilidad. Además, es compatible con múltiples plataformas informáticas y ofrece una infinidad de aplicaciones que permiten acceder rápidamente a las sentencias del gestor de base de datos.

Desarrollado bajo licencia dual GPL/Licencia comercial por *Oracle Corporation*, está considerada como la base datos *open source* más popular del mundo, y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web.

2.2.8 | Apache

Apache es el servidor web más utilizado, líder con el mayor número de instalaciones a nivel mundial muy por delante de otras soluciones como el IIS (Internet Information Server) de Microsoft. Apache es un proyecto de código abierto y uso gratuito, multiplataforma (hay versiones para todos los sistemas operativos), muy robusto y que destaca por su seguridad y rendimiento.

Este servidor web sigue siendo desarrollado por la comunidad de usuarios desarrolladores que trabaja bajo la tutela de Apache Software Foundation.

Apache es utilizado principalmente para proporcionar alojamiento de páginas web, ya sean estáticas o dinámicas. Este estupendo servidor se integra a la perfección con otras aplicaciones, creando el famoso paquete XAMP con Perl, Python, MySQL y PHP, junto a cualquier sistema operativo.

2.2.9 | Bootstrap

Bootstrap es un framework CSS desarrollado en el año 2011 inicialmente por Twitter. Esta tecnología permite dar forma a un sitio web mediante librerías CSS que incluyen tipografías, botones, cuadros, menús y otros elementos que pueden ser utilizados en cualquier sitio web. En el mismo año 2011 fue liberado bajo la licencia MIT y su desarrollo continua en un repositorio de GitHub.

Esta tecnología permite crear aplicaciones web con diseño *responsive*, es decir, adaptable a todos los dispositivos (PC, tablet y smartphone) además de crear interfaces de usuario

limpias y bien estructuradas gracias a su *grid system* que permite la maquetación por columnas, concretamente 12.

Ofrece un soporte casi completo con HTML5 Y CSS3, lo que permite su uso de un modo muy flexible y como es de esperar, es totalmente compatible con todos los navegadores.

Como curiosidad, si el lector conoce algunos gestores de contenido como WordPress o Drupal, debe saber que casi todos los temas o plantillas de ambas plataformas están basadas en Bootstrap.

2.2.10 | JSON

En el apartado 2.2.5 se habla de AJAX y del tipo de respuesta que obtiene del servidor, pues bien, en la aplicación desarrollada se utiliza el formato JSON [41][42] para obtener dicha respuesta.

Se define como un formato para el intercambio de datos, proponiendo una sintaxis dedicada que se usa para identificar y gestionar dichos datos. Nació como una alternativa a XML dado su fácil uso en JavaScript. Su mayor ventaja es que puede ser tratado por cualquier lenguaje de programación, por lo que puede ser utilizado para el intercambio de información entre distintas tecnologías (PHP – JavaScript/jQuery en el caso de Librarino).

Los objetos JSON se definen como conjuntos no ordenados de pares 'clave/valor' separados por comas delimitado por llaves {…}. La clave siempre debe ser una cadena, en cuanto al valor, puede tener cualquier formato, incluso otro objeto o array JSON.

Si lo comparamos con XML, este goza de mayor soporte y muchas más herramientas de desarrollo (tanto en el lado del cliente como en el servidor), aunque JSON también cuenta con una gran cantidad de analizadores del lado del servidor, existiendo al menos uno para la mayoría de los entornos. En la Figura 9 se compara una misma respuesta obtenida en estos dos formatos.

JSON puede ser muy compacto y eficiente de manera efectiva, aunque es poco recomendable si hay datos fuertemente jerarquizados o anidaciones profundas ya que se vuelve un poco confuso dado su simple formato.

JSON

```
{
  "person" : {
    "xmlns" : "urn:ns:person",
    "firstName" : {
      "$t" : "John"
    },
    "lastName" : {
      "$t" : "Smith"
    },
    "contactInfo" : {
      "default" : "true",
      "type" : "home",
      "xmlns" : "urn:ns:contactinfo",
      "phone" : [
        {
          "type" : "voice",
          "$t" : "203-555-1212"
        },
        {
          "type" : "fax",
          "$t" : "203-555-1213"
        }
      ],
      "email" : {
        "xmlns" : "",
        "$t" : "jsmith@example.com"
      }
    },
    "photo" : {
      "xmlns" : "",
      "$t" : "http://example.com/jsmith
/profile.png"
    }
  }
}
```

XML

```
<person xmlns="urn:ns:person">
   <firstName>John</firstName>
   <lastName>Smith</lastName>
   <contactInfo
xmlns="urn:ns:contactinfo" type="home"
default="true" >
      <phone
type="voice">203-555-1212</phone>
      <phone
type="fax">203-555-1213</phone>
      <email
xmlns="">jsmith@example.com</email>
   </contactInfo>
   <photo xmlns="">http://example.com
/jsmith/profile.png</photo>
</person>
```

Figura 9: Comparación de los formatos JSON y XML.

CAPÍTULO 3 | DESARROLLO DEL DISPOSITIVO FÍSICO

En este tercer capítulo se describe el dispositivo físico completamente, sus componentes, conexiones entre ellos, el software y los distintos estados por los que puede pasar.

El dispositivo es la base del sistema ya que, además de mostrar el estado de cada asiento, hará de interfaz entre el usuario y el propio sistema.

3.1 | Componentes del dispositivo

Para el desarrollo del dispositivo se ha precisado de diferentes módulos y componentes para aumentar la funcionalidad y/o capacidad de la placa Arduino. La elección de los mismos ha facilitado desde un primer momento el desarrollo del dispositivo dada la sencillez de sus conexiones e integración con la placa principal. A continuación, se describen todos los elementos utilizados y su función.

3.1.1 | Placa Arduino Uno

Es el componente principal del dispositivo al cual conectamos los diferentes módulos y que contiene el programa principal que desempeña la funcionalidad del mismo. La Figura 10 muestra la placa Arduino UNO.

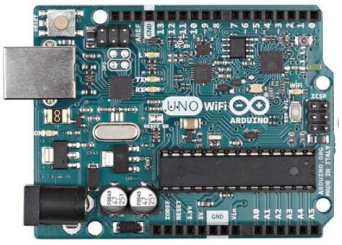

Figura 10: Placa Arduino UNO.

3.1.2 | Módulo Wifi ESP8266

Este módulo (Figura 11) dota al dispositivo de conexión a internet vía Wifi.

Figura 11: Módulo Wifi ESP8266.

Conexiones

Las conexiones (Tabla 1 y Figura 12) de este módulo son sencillas y solo hay que tener en cuenta ciertas consideraciones:

- No es tolerante a 5V. Los pines de lógica no deben de ser expuestos a más de 3.3V (aunque algunas versiones tienen protecciones hasta 6V en todos los pines menos en Vcc).

- Los pines de CHPD y GPIO2 deberán estar en HIGH (3.3V) al iniciar el modulo. Estos controlan el modo en que el dispositivo arranca.

- El pin de RST (RESTART) no es necesario conectarlo. Si se necesita reiniciar el modulo, se puede desconectar y conectar el cable que conecta CH_PD a 3.3V (o bien conectar un botón que interrumpa momentáneamente esta conexión).

- El módulo requiere más de 40mA para su funcionamiento óptimo bajo condiciones de muchas peticiones por segundo, por lo que es bueno conectarlo con una fuente externa regulada a 3.3V que pueda proveer más de 175mA.

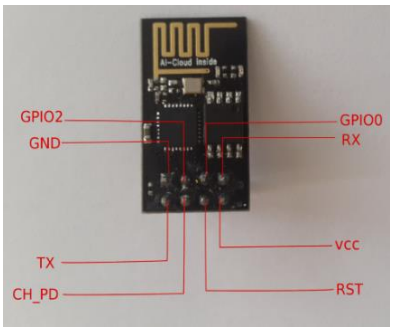

Figura 12: Conexiones del módulo ESP8266.

ESP8266	Arduino
CH_PD	3.3 V
GPIO2	3.3 V
TX	Pin 5
RX	Pin 6
Vcc	3.3 V
GND	GND

Tabla 1: Resumen de conexiones del módulo ESP8266.

3.1.3 | Lector de código de barras vía USB

Su función es la de leer el código de barras del carnet de estudiante de cada alumno con lo que se ocupará/desocupará el asiento.

Para el desarrollo del prototipo se ha utilizado un lector básico de pistola (Figura 13), como los que podemos encontrar en cualquier supermercado. En un futuro producto final se utilizaría un lector más pequeño y manejable para encajarlo en la carcasa del dispositivo.

La conexión al dispositivo se hace mediante el puerto USB del Shield USB.

Figura 13: Lector de código de barras.

3.1.4 | Shield USB

Dado de la placa Arduino no incluye el manejo de dispositivos USB, es necesario añadir un *shield* (Figura 14) que proporcione dicha funcionalidad y al que irá conectado el lector de código de barras anterior.

Figura 14: Shield USB.

Como se explicó en el apartado 2.2.1, simplemente se coloca encima de la placa Arduino encajando los pines de la misma nomenclatura. En la Figura 15 puede observarse la unión de ambos componentes.

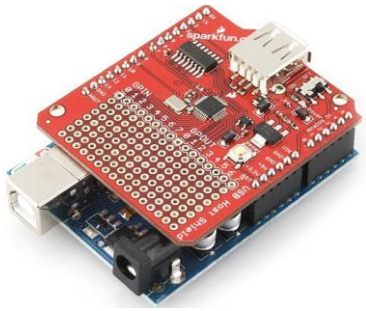

Figura 15: Conexión del Shield USB al Arduino.

3.1.5 | Pantalla LCD

Mediante este componente (Figura 16) se muestra información sobre el estado del dispositivo. Concretamente se muestra el estado actual del asiento, el número del asiento al que está asignado el dispositivo y el código de barras leído.

Figura 16: Pantalla LCD.

Conexiones

Las conexiones (Tabla 2) se sitúan en una placa en la parte trasera de la pantalla. Dicha placa actúa transformando los pines que necesitaría una pantalla LCD normal a una comunicación serie (Figura 17), por lo que solamente necesitamos utilizar 3 pines para unir la pantalla al Arduino.

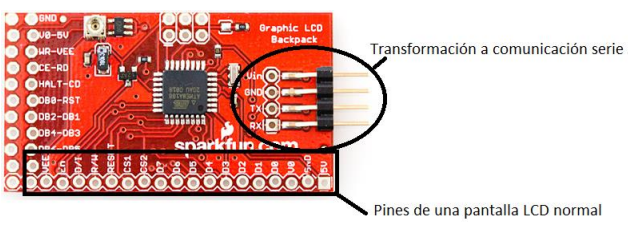

Figura 17: Conexiones de la pantalla LCD.

Pantalla	Arduino
Vin	5 V
GND	GND
RX	Pin 3

Tabla 2: Resumen de conexiones pantalla – Arduino.

3.1.6 | Led RGB

Es el componente más pequeño del dispositivo. Se encargará de indicar el estado del asiento mediante un código de colores. En la Figura 18 se puede ver un led como el que se ha utilizado para el desarrollo del dispositivo.

Figura 18: Led RGB.

Conexiones

El led RGB consta de 4 patillas (Figura 19), una para cada color básico y la última para la toma de tierra. La programación del led es simple: desde el programa se indica un valor de 0 a 255 para cada patilla, haciendo posible obtener cualquier color. La Tabla 3 muestra el resumen de conexiones entre el led y los distintos pines de la placa Arduino.

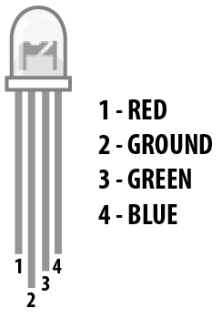

Figura 19: Conexiones del led RGB

LED	Arduino
Rojo	Pin A0
GND	GND
Verde	Pin A1
Azul	Pin A2

Tabla 3: Resumen de conexiones entre el led y la placa

En la figura 20 podemos observar el esquema de conexión del led junto con las resistencias de protección de 220 Ω y la protoboard.

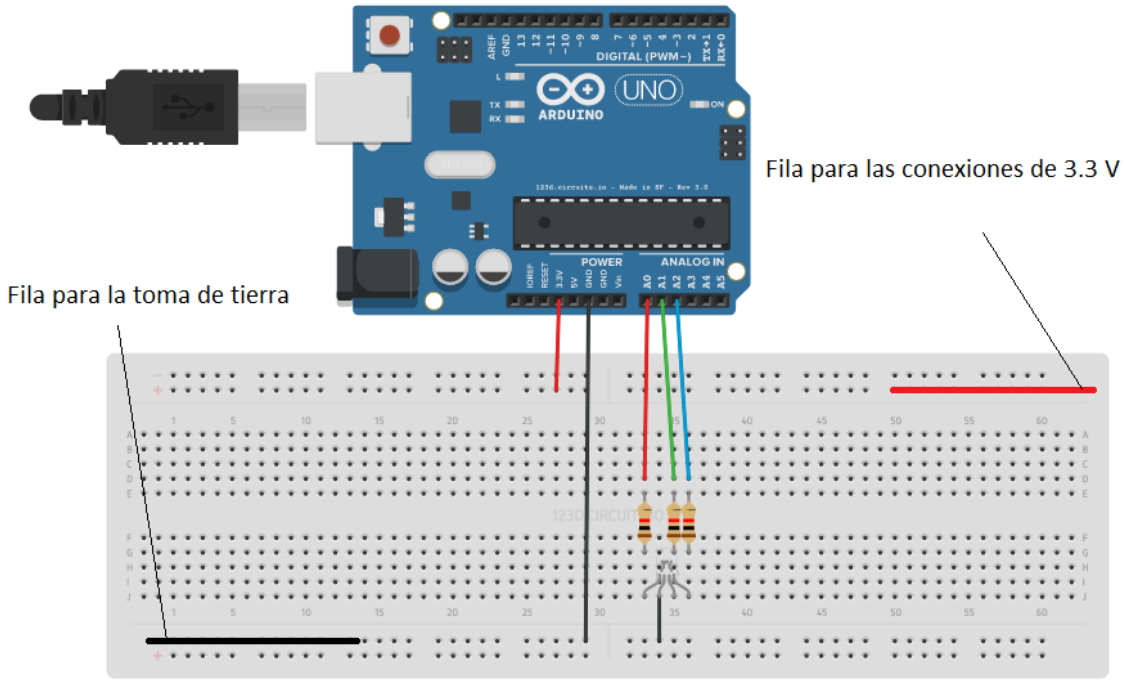

Figura 20: Esquema de conexión del led RGB

3.1.7 | Otros componentes

Los componentes restantes (Figuras 21-23) se encargan de interconectar los diferentes componentes a la placa principal.

Figura 21: Protoboard.

Figura 23: Cables.

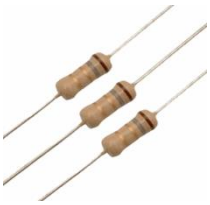

Figura 22: Resistencias.

3.2 | Coste del dispositivo

En este apartado abordamos una de las partes fundamentales en el desarrollo de cualquier proyecto, sea software o de cualquier índole: el tema económico.

Para la realización de este proyecto se ha necesitado hacer un pequeño desembolso para la adquisición de las piezas del dispositivo.

Por supuesto el departamento se ofreció a proporcionar las piezas, pero el alumno accedió voluntariamente a pagarlas de su bolsillo.

En la Tabla 4 se puede observar el coste de cada componente del dispositivo además del montante total.

Placa Arduino Uno	24.14 €
Módulo Wifi ESP8266	10 €
Lector de código de barras USB	29.95 €
Arduino USB Host Shield	29.65 €
Pantalla Serial LCD	42.35 €
LED RGB	2.30 €
Pack 10 cables Arduino (x2)	7.80 €
Resistencia 220 ohmios (x3)	0.28 €
Protoboard	4.50 €
TOTAL	**150.97 €**

Tabla 4: Costes de los componentes del dispositivo.

El dispositivo en su totalidad tendría un coste aproximado de unos 150 euros, lo que se convertiría en una cantidad más seria si se instalara en una biblioteca de aproximadamente 400 asientos, eso sin contar los costes de producción. Seguramente podríamos reducir los costes al comprar las piezas en grandes cantidades y al usar un circuito impreso en lugar de protoboard.

No cabe duda que el sistema íntegro conllevaría un desembolso económico tanto para la adquisición de los dispositivos como para la adecuación de las instalaciones. Pero debemos pensar que un producto como el que se ha desarrollado supondría un gran avance tecnológico para cualquier biblioteca, además de mejorar la experiencia del alumno.

3.3 | Software del dispositivo

Una vez definida toda la parte hardware del dispositivo, pasamos a describir el software que le permite desempeñar su función.

Para el desarrollo del código se ha utilizado el entorno de desarrollo Arduino IDE 1.8.1 [28]; en el Apéndice B se puede observar los pasos para la instalación y configuración del entorno de desarrollo.

El código fuente puede ser dividido en 4 partes fundamentales, correspondientes al control de cada uno de los componentes del dispositivo. Estas 4 partes son dependientes entre sí por lo que no pueden funcionar una sin la otra, o en otras palabras, la salida de un módulo es la entrada de otro. Dicho esto, los módulos o partes principales son, por orden de intervención en el sistema:

1) Control de la lectura del código de barras.
2) Control de la conexión con la base de datos.
3) Control del led RGB.
4) Control de la pantalla LCD.

En la Figura 24 se muestran las diferentes interacciones entre los módulos software del dispositivo (bloques rectangulares), el usuario, y el resto de elementos del sistema (bloques con esquinas redondeadas).

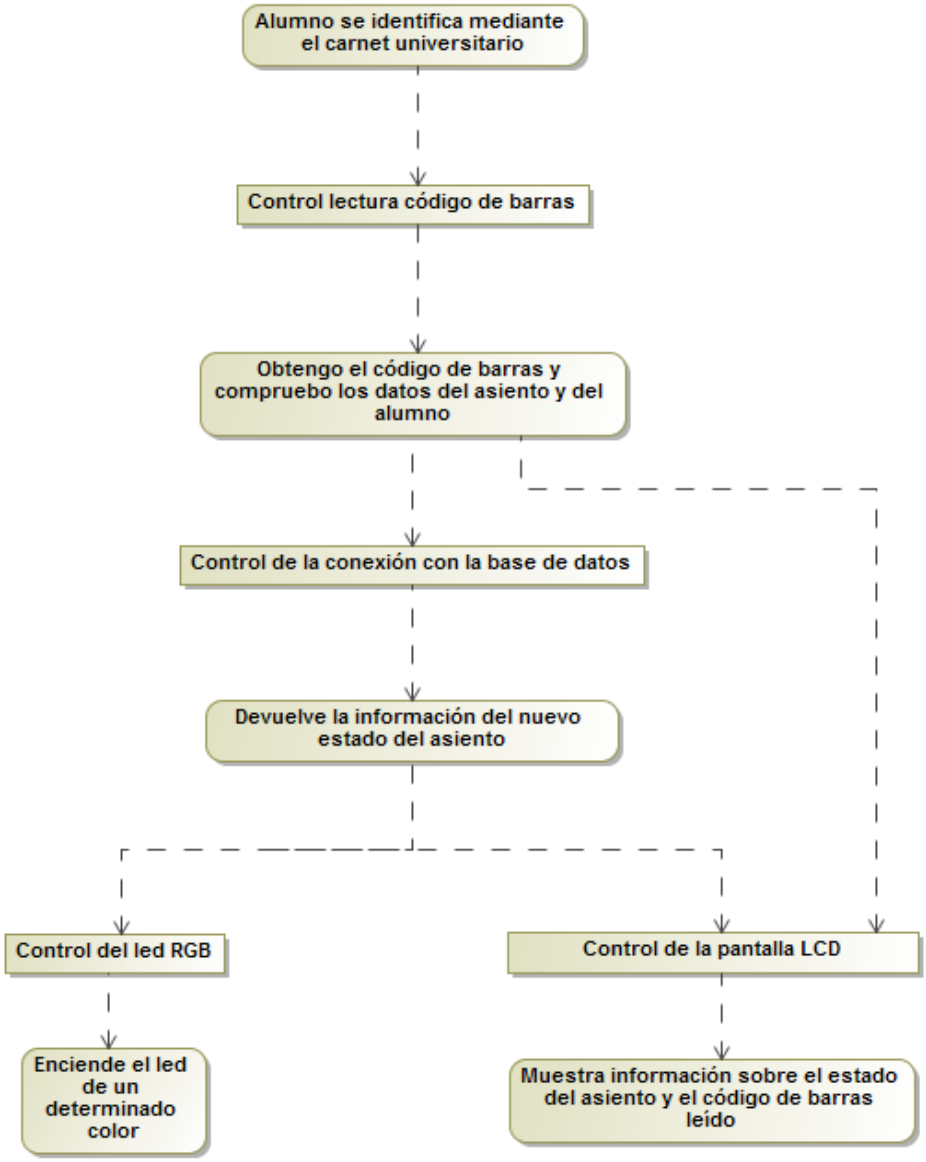

Figura 24: Dependencia entre los módulos del código fuente.

3.3.1 | Control de la lectura del código de barras

La función de este módulo será simplemente la de leer el código de barras contenido en el carnet universitario del usuario. Para la construcción de este módulo se ha adaptado un ejemplo encontrado en la web [44] además de precisar de una librería para el control del USB [43].

Prácticamente todas las librerías de Arduino vienen documentadas con una serie de ejemplos para comprender mejor la funcionalidad de la misma, por lo que el código para la lectura está basado en uno de esos ejemplos. Dicho ejemplo consiste en la lectura de caracteres desde un teclado así que ha resultado sencillo reutilizar el código, ya que al fin y al cabo la función del lector es similar a la de un teclado de ordenador.

3.3.2 | Control de la conexión con la base de datos

El código perteneciente a este módulo será el encargado de conectar el dispositivo a internet y de realizar una petición a la aplicación web para obtener los datos del asiento. También ha sido necesaria la utilización de una nueva librería [45] para poder controlar el módulo Wifi.

Para la conexión a internet es necesario proporcionar el SSID de una red WiFi además de la contraseña. Una vez conectado a internet, el programa lanza una petición GET a un script PHP que se encargará de dar una respuesta en función del estado del asiento y del código de barras leído. La petición GET tiene el siguiente formato:

{direcciónWeb}/controladores/ardController.php?id={id del asiento}&usuario={código de barras}

Mediante el analizador de tráfico Wireshark se ha podido capturar el paquete que transporta dicha petición. En la Figura 25 se resumen los campos más importantes.

Figura 25: Captura en Wireshark de la petición.

La respuesta será un número entero que identificará una situación u otra:

- **0:** el nuevo estado del asiento es "ocupado".
- **1:** el nuevo estado del asiento es "libre".
- **2:** el nuevo estado del asiento es "reservado".
- **3:** el código de barras proporcionado no corresponde a ningún usuario del sistema por lo que esta respuesta estará identificada con el estado "usuario no válido".

Cabe destacar que la petición GET se realiza en dos partes del programa. La primera se produce una vez que se ha leído un código y la segunda se hace en el bucle principal del programa dentro de un temporizador. Este temporizador comprueba cada 8 segundos el estado del asiento por si ha cambiado vía web debido a una reserva. La petición se hace sin código de barras por lo que el script PHP detecta está situación y simplemente devuelve el estado del asiento sin hacer ninguna comprobación más.

3.3.3 | Control del led RGB

A partir del estado obtenido del módulo anterior el led emitirá un color, mostrando de un modo más sencillo el estado del asiento, lo que facilitará la identificación de asientos por parte del alumno.

El código de colores es simple: si el asiento está ocupado el led se encenderá de color rojo, verde si en cambio el asiento está libre, y por último para un asiento reservado, el led emitirá un tono azul.

3.3.4 | Control de la pantalla LCD

El último de los módulos es el encargado de controlar la pantalla LCD. En esta pantalla se mostrará alguna información del asiento, como la biblioteca en la que se encuentra, el número del asiento al que está asignado el dispositivo, el estado actual del asiento y por último el código de barras que se ha leído.

Para el control de la pantalla se ha requerido la utilización de otra librería [46] que proporciona las funciones básicas para imprimir caracteres en la pantalla, borrar partes de la misma, etc.

3.4 | Interacción con el dispositivo

En el último apartado de este capítulo se explica una interacción del usuario con el dispositivo y como transitará entre los diferentes estados dependiendo del estado actual y la entrada, que en este caso será la identificación del estudiante.

El primer paso en esta interacción será la de la identificación del alumno mediante su carnet universitario. Dependiendo del estado actual del asiento transitará a otro nuevo estado:

- **Estado actual - Ocupado**: si el alumno está ocupando dicho asiento, el nuevo estado será libre. Esta interacción servirá para que el alumno desocupe el asiento cuando quiera marcharse. En cambio, si el asiento está ocupado por otro alumno, el asiento seguirá ocupado.

- **Estado actual - Reservado**: un asiento reservado solo podrá ser ocupado por el alumno que ha reservado dicho asiento, por lo tanto, el nuevo estado pasará a ocupado. Sin embargo, si no es el alumno que ha reservado el asiento, seguirá reservado.

- **Estado actual - Libre**: si el asiento está libre podrá ser ocupado por cualquier alumno que se identifique correctamente.

Para mostrarlo de un modo más sencillo y entendible se puede ver en la Figura 26 un diagrama de estados que representa la información anterior.

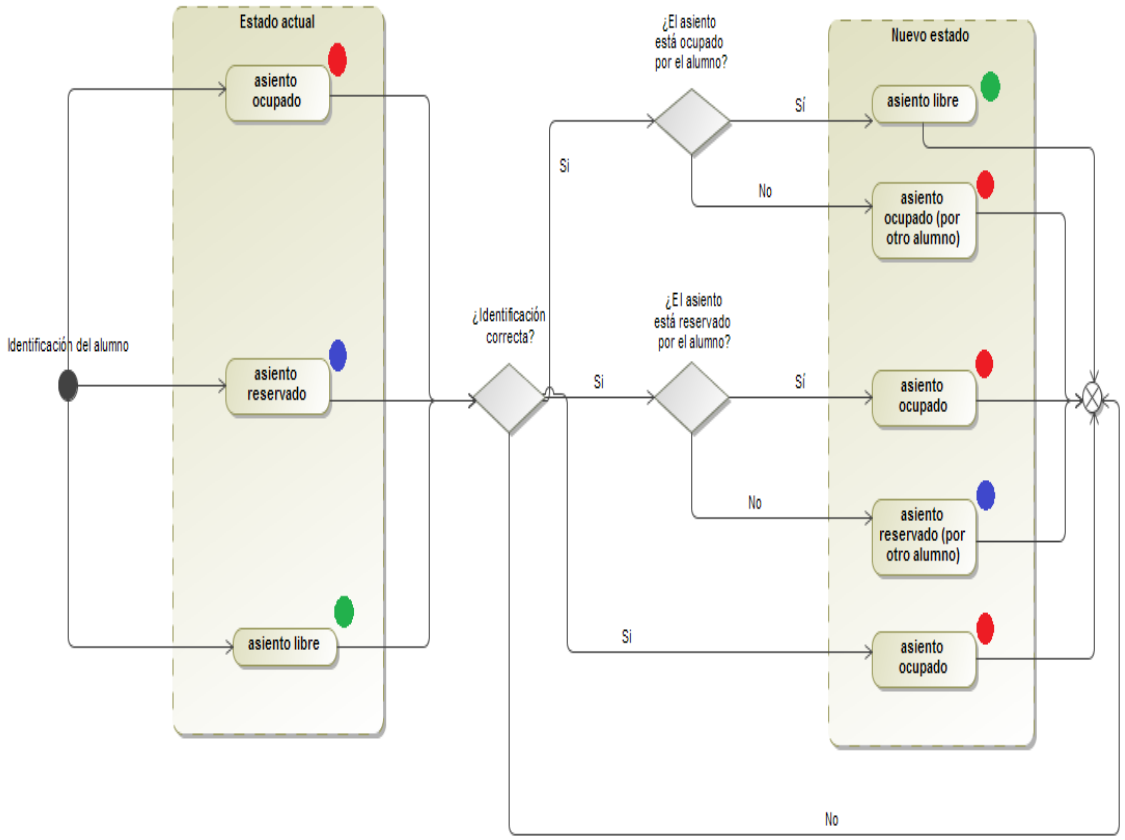

Figura 26: Diagrama de estados del dispositivo

CAPÍTULO 4 | DESARROLLO DE LA APLICACIÓN WEB

En este capítulo se abordará el desarrollo de la aplicación web, su arquitectura, componentes y descripción del modelo de datos.

4.1 | Casos de uso

En este apartado se estudiarán cómo intervienen los diferentes roles con el sistema, dando así una orientación básica de las múltiples funcionalidades de la aplicación que se ampliaran en el manual de usuario. Se recorrerán en primer lugar los casos de uso del dispositivo y a continuación los de las aplicaciones web.

En el sistema se pueden distinguir tres roles principales:

- **Usuario:** es el usuario común del sistema; se puede identificar con un alumno.
- **Bibliotecario:** se refiere al personal de biblioteca, el cual podrá gestionar distintos apartados del sistema.
- **Administrador:** es el rol que alberga todos los permisos y que puede manejar el sistema al completo.

4.1.1 | Dispositivo

En el dispositivo solo habrá dos posibles acciones: o bien ocupamos un asiento que esté libre o que hayamos reservado, o bien nos levantamos de un asiento que hayamos ocupado. Cualquier rol de la aplicación podrá realizar dichas acciones. La Figura 27 muestra con más detalle la interacción entre los roles y las acciones.

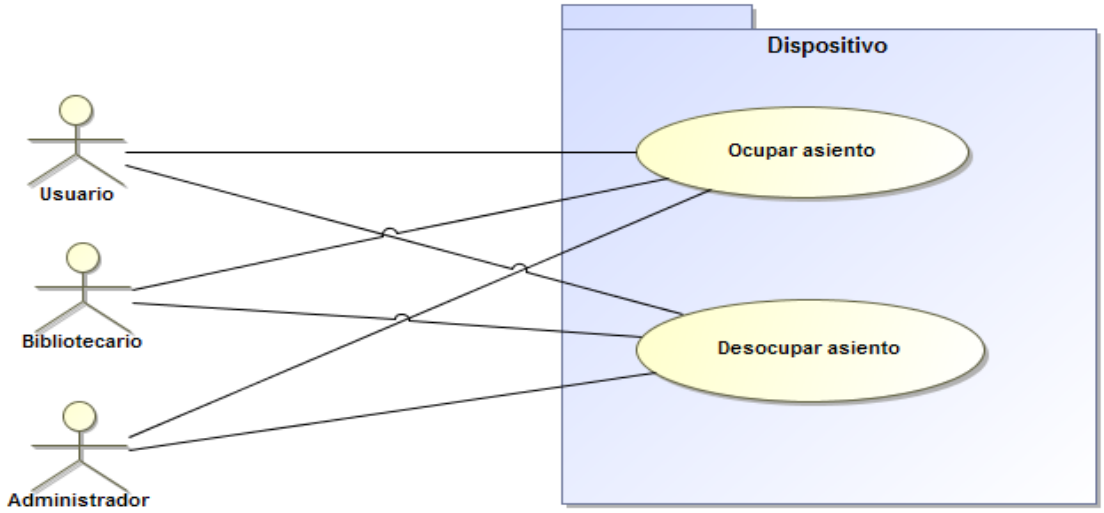

Figura 27: Casos de uso del dispositivo.

4.1.2 | Aplicación principal

En la aplicación principal es donde interviene íntegramente el rol 'usuario' (Figura 28), ya que este no tiene acceso a la aplicación de administración. Básicamente, la función de un usuario será la de identificarse en la aplicación, ver los mapas, reservar un asiento, notificar incidencias relacionadas con los asientos y cambiar su contraseña.

Los roles 'bibliotecario' y 'administración' (Figura 29), además de tener los privilegios referidos al rol 'usuario', podrán ver quién ha reservado u ocupado un determinado asiento y poder liberar dicho asiento sin necesidad de autorización de su ocupante. Obviamente se liberará aquel asiento del que se considere que se está haciendo un uso inapropiado.

4.1.3 | Aplicación de administración

A la aplicación de administración solamente tendrán acceso los roles 'administrador' (Figura 30) y 'bibliotecario' (Figura 31). Como cabe pensar, el administrador podrá desempeñar todas las funcionalidades de esta aplicación. El rol 'bibliotecario' únicamente tiene denegado el acceso a modificar los datos de las bibliotecas, que no su mapa de asientos. En las Figuras 53 y 54 se pueden ver los casos de uso para el rol 'administrador' y 'bibliotecario' respectivamente.

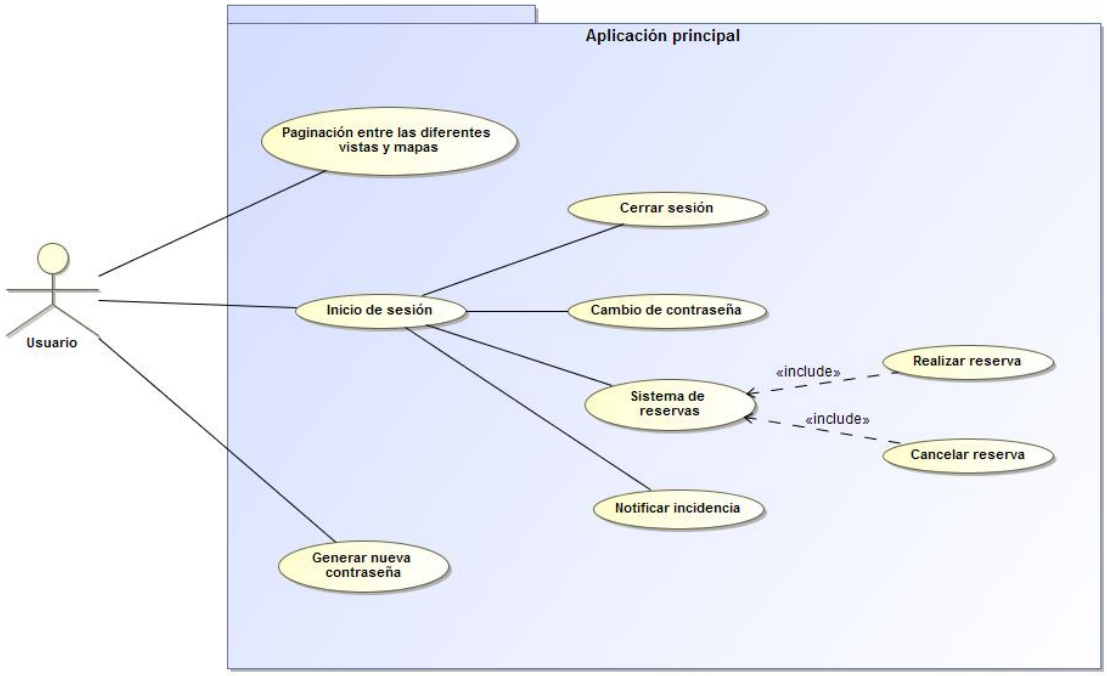

Figura 28: Casos de uso en la aplicación principal para el rol 'usuario'.

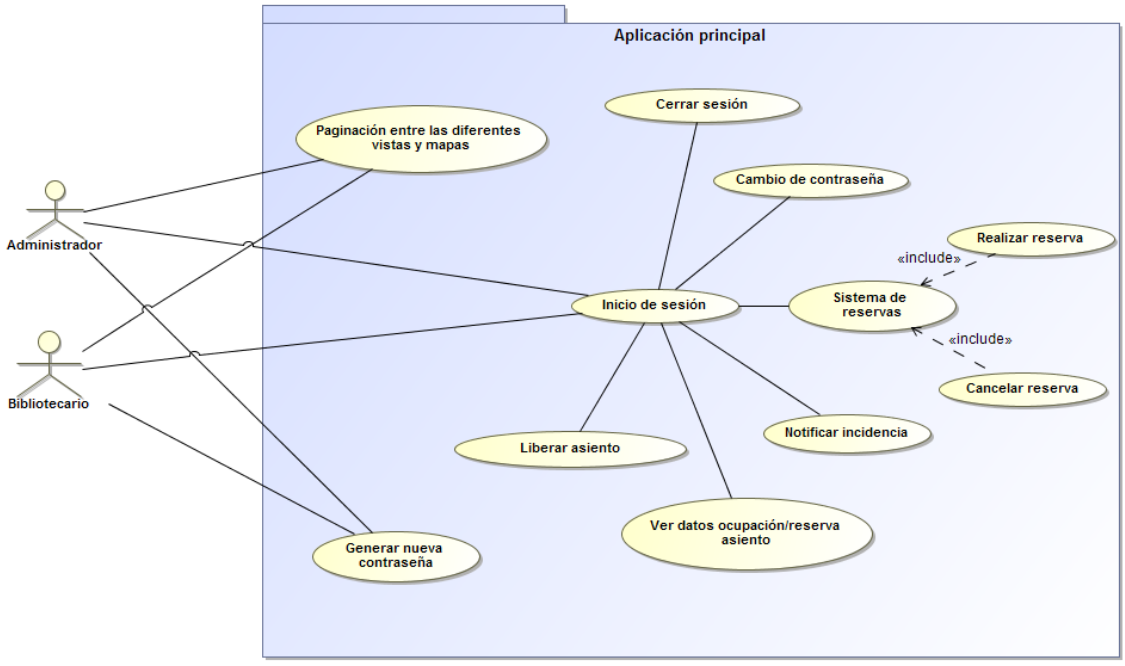

Figura 29: Casos de uso en la aplicación principal para los roles 'bibliotecario' y 'administrador'.

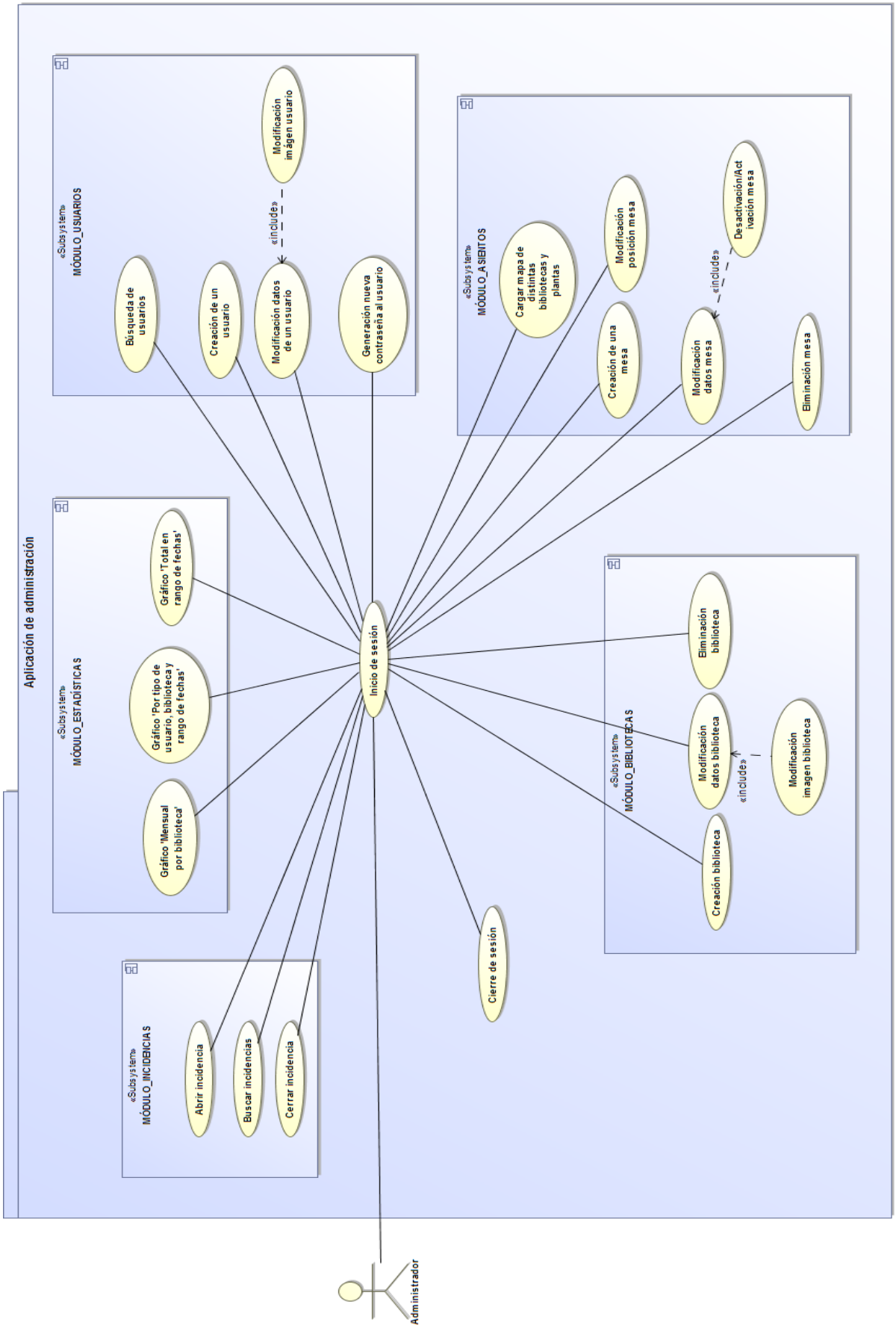

Figura 30: Casos de uso para el rol 'administrador' en la aplicación de administración.

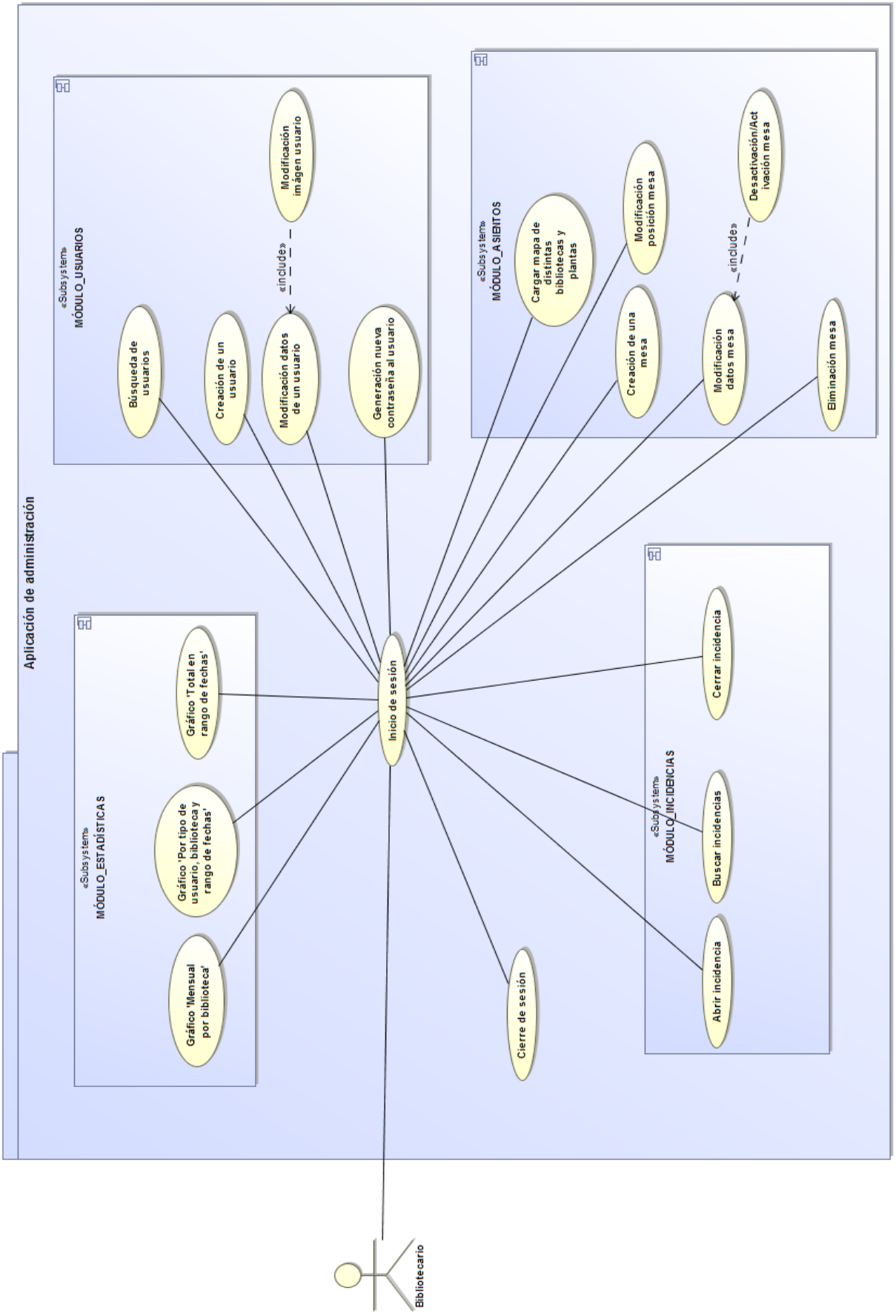

Figura 31: Casos de uso para el rol 'bibliotecario' en la aplicación de administración.

4.2 | Arquitectura de la aplicación

Una de las primeras cosas necesarias para empezar a construir cualquier software es definir su arquitectura. Debemos elegir la más adecuada según el tipo de aplicación, requisitos y funcionalidades.

Para el desarrollo de esta aplicación se ha utilizado un modelo dividido en tres capas, en la que cada una de ellas desempeña una función concreta y ofrece servicios a la capa superior. En la Figura 32 se puede ver con más detalle la interacción entre cada una de las capas.

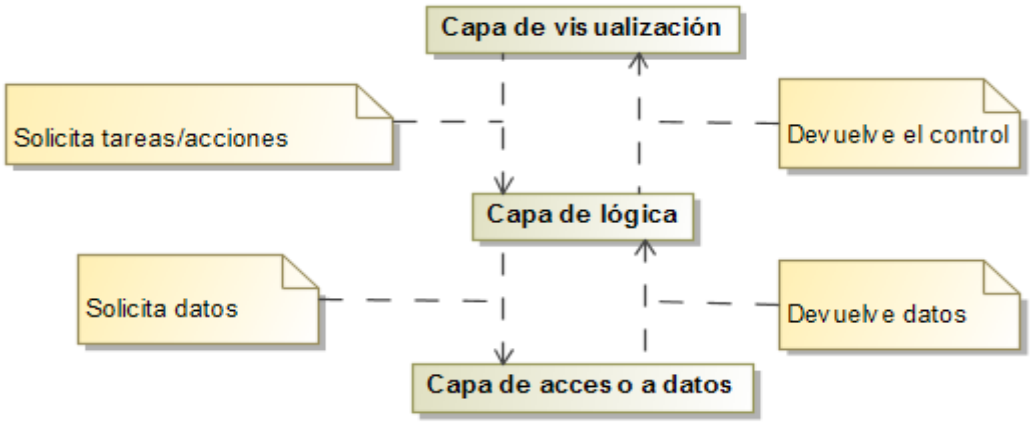

Figura 32: Arquitectura de la aplicación web.

4.2.1 | Capa de visualización

La primera de las capas contiene las vistas o páginas de la aplicación, que solicitan acciones o tareas a la capa de lógica, mediante, por ejemplo, pulsar un botón, el envío de un formulario, etc. Todas las vistas complejas tienen su propio controlador.

Para la realización de las vistas de la aplicación se ha utilizado HTML combinado con CSS y Bootstrap para dar diseño a las páginas, y para la generación de las partes dinámicas se ha utilizado PHP embebido y Javascript.

- **Index.php (Figura 37):** página principal de la aplicación web, donde se mostrará un resumen de todas las bibliotecas junto con su imagen, nombre y número de asientos libres.
- **Biblioteca.php (Figura 36):** en esta vista se muestra el mapa de cada biblioteca. A través de parámetros GET se especifica qué biblioteca y planta se debe mostrar.
- **Ayuda.php (Figura 35):** página donde el usuario puede ver algunas directrices sobre el funcionamiento del sistema.

PHP incluye, entre sus muchas funcionalidades, la posibilidad de importar código desde un script PHP a otro. En este caso se ha utilizado para realizar el menú, el pie de página y el contenido del mapa de asientos y así ahorrar escritura de código.

- **Header.php:** cabecera mostrada en todas las páginas citadas anteriormente. Además de contener el menú, contiene pantallas modales para realizar el login y el cambio de contraseña.
- **Footer.php:** pie de página importado del mismo modo que el menú.
- **Mapa.php:** este script se utiliza para mostrar el mapa de los asientos. Se utiliza en la web principal y en la aplicación de administración y se mostrará de un modo u otro en una u otra página. Las Figuras 33 y 34 muestran la comparación entre los diferentes modos de visualizar un mapa.

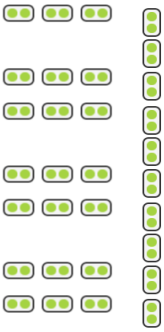

Figura 34: Mapa en modo visualización.

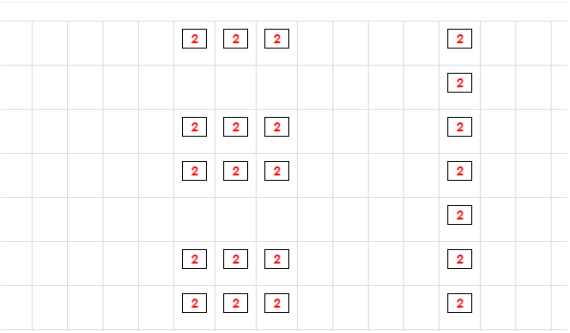

Figura 33: Mapa en modo edición.

En la aplicación de administración se ha utilizado la misma filosofía, una serie de vistas principales en las cuales importamos algunos scripts PHP.

- **Admin/inicio.php (Figura 37):** página de bienvenida de la aplicación de administración.
- **Admin/index.php (Figura 38):** página principal de la aplicación de administración donde el usuario deberá identificarse para acceder.
- **Admin/usuarios.php (Figura 39):** listar y crear usuarios.
- **Admin/verUsuario.php (Figura 40):** mostrará la información del usuario cuyo ID (NIU o DNI) se ha pasado mediante el método GET.
- **Admin/incidencias.php (Figura 41):** desde aquí se gestionarán las incidencias sobre cualquier dispositivo que haya podido detectar algún usuario, que previamente habrá notificado en la aplicación principal.
- **Admin/bibliotecas.php (Figura 42):** su función es gestionar la información de las bibliotecas, como el nombre, número de plantas, etc. Se podrán crear, modificar y eliminar bibliotecas.
- **Admin/asientos.php (Figura 43):** en esta vista se podrá modificar el mapa de cada biblioteca.
- **Admin/estadísticas.php (Figura 44):** muestra diferentes estadísticas mediante gráficos.

- **Admin/header.php:** header de la aplicación de administración. Se importa desde todas las páginas.
- **Admin/menuLateral:** menú lateral para facilitar la navegación entre páginas. Al igual que el header, se importa desde todas las páginas.

A continuación, podemos ver el diagrama de interacción entre las vistas de la aplicación principal (Figuras 35, 36, 37).

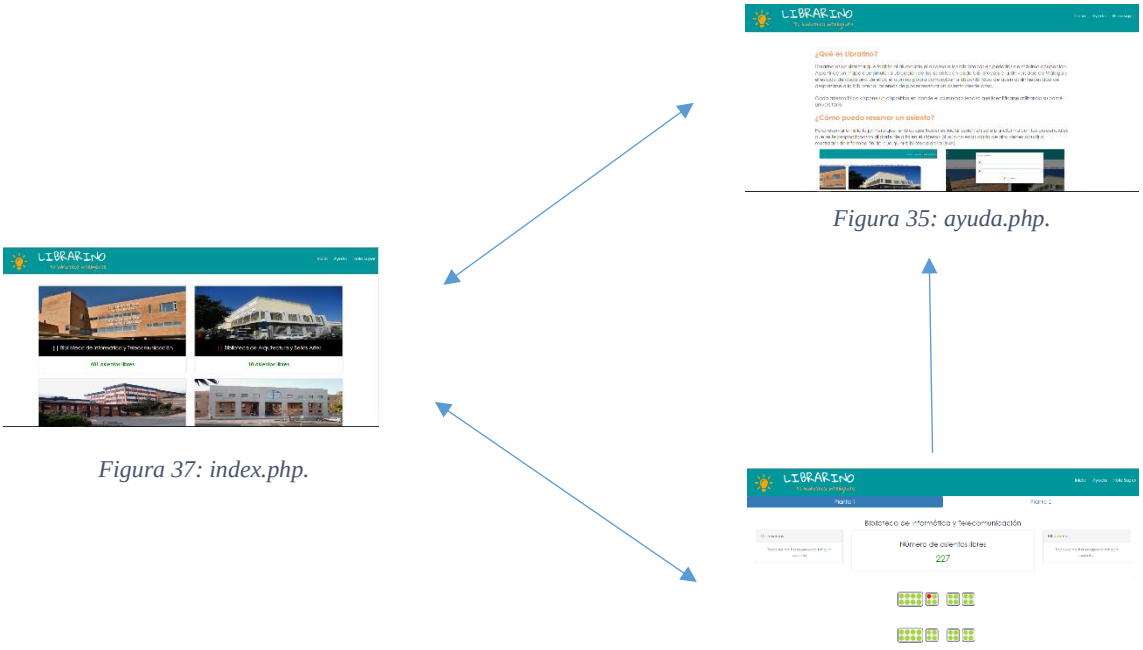

Figura 35: ayuda.php.

Figura 37: index.php.

Figura 36: biblioteca.php.

Como se ha podido observar, la aplicación principal es bastante simple, tan solo consta de 3 vistas, pero con la potencia suficiente para implementar una interfaz de usuario intuitiva y completa con la que poder visualizar la situación y estado de todos los asientos cada una de las bibliotecas.

Pasamos ahora a mostrar el diagrama de interacción entre las vistas de la aplicación de administración (Figuras 38-45).

Desde todas las vistas
(admin/*)

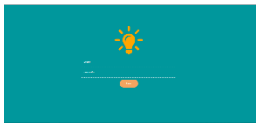

Figura 39: admin/index.php.

Figura 38: admin/inicio.php.

Hacia todas las vistas
(admin/* excepto verUsuario.php)

Figura 40: admin/usuarios.php.

Figura 41: admin/verUsuario.php.

Figura 45: admin/estadisticas.php.

Figura 43: admin/bibliotecas.php.

Figura 42: admin/incidencias.php.

Figura 44: admin/asientos.php.

Plugins jQuery/JavaScript utilizados

jQuery permite el uso de plugins de muy diversos tipos para añadir mayor vistosidad y funcionalidad a la aplicación, ahorrando gran cantidad de tiempo y código. En la aplicación se han usado cuatro de estos plugins, los cuales se describen a continuación:

- **Datatables** [47]**:** con este plugin se pueden realizar tablas complejas, con paginación, filtros, etc., lo que permite una interacción completa del usuario con una tabla HTML sin necesidad de escritura adicional de código (Figura 46).

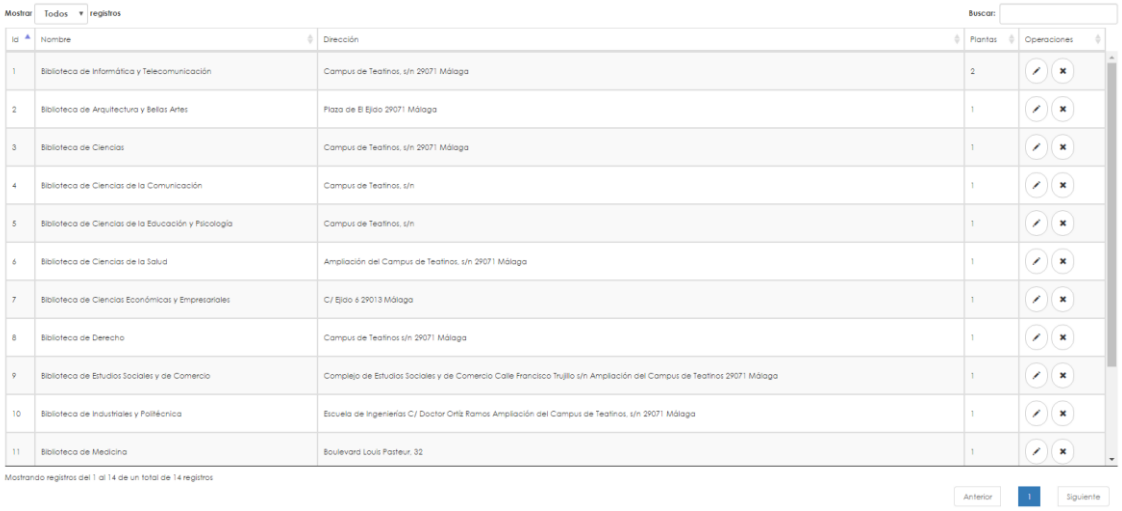

Figura 46: Ejemplo del plugin Datatables.

- **Select2** [48]**:** sustituye al desplegable tradicional de HTML proporcionando múltiples combinaciones de desplegables (filtro, selección múltiple más interactiva, etc.) (Figura 47).

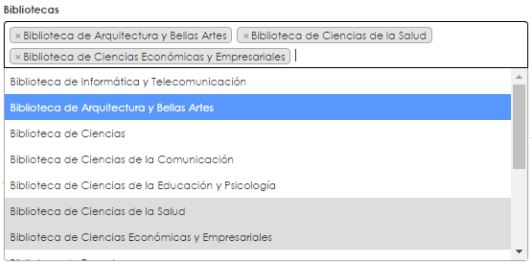

Figura 47: Ejemplo de select multiple del plugin Select2.

- **Datepicker** [49]**:** para la introducción de fechas en una web es recomendable utilizar un pequeño calendario como entrada de texto para mantener un formato fijo. El tipo 'date' de un 'input' de un formulario HTML (Figura 49) puede no ser compatible con todos los navegadores, por lo que para corregir esto se ha utilizado un plugin (Figura 48) que proporciona la misma funcionalidad, aparte de tener un diseño más atractivo.

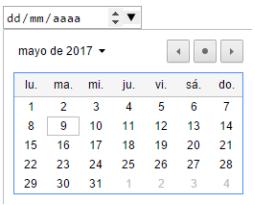

Figura 49: Calendario tradicional HTML.

Figura 48: Plugin Datepicker.

- **Chart.js** [50]**:** gracias a este plugin podemos incluir múltiples tipos de gráficos en nuestra web. Un ejemplo de esta librería lo encontramos en la Figura 50.

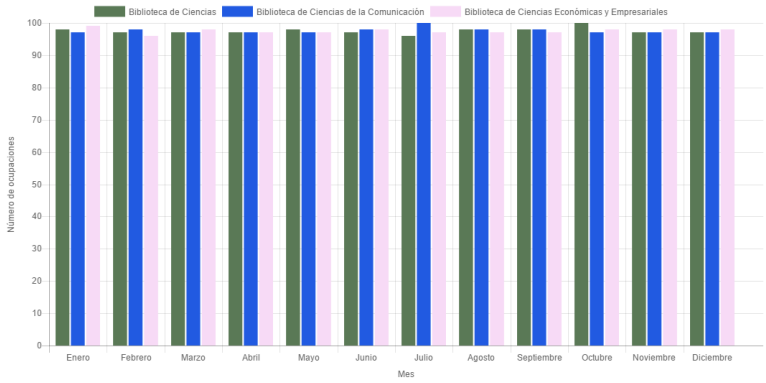

Figura 50: Ejemplo de Chart.js en la aplicación de administración.

4.2.2 | Capa de lógica

En esta capa podemos encontrar la lógica de la aplicación, donde se realizarán todas las acciones de control del sistema. Se compone de una serie de scripts PHP que ofrecen servicios a las vistas. Cada uno de estos servicios es invocado mediante una llamada GET o POST desde una vista hacia un controlador, que ejecutará una acción u otra dependiendo del tipo de llamada. Estas llamadas se hacen desde formularios HTML, funciones JavaScript o llamadas asíncronas desde AJAX.

El funcionamiento de la capa de lógica es idéntico en ambas aplicaciones. Los scripts PHP que desempeñan esta funcionalidad pueden encontrarse en la carpeta 'controladores' del directorio principal de cada una de las aplicaciones.

En el sistema existe la funcionalidad de poder enviar emails, en nuestro caso lo utilizamos en tres escenarios que se comentarán en la descripción de cada uno de los controladores. Para disponer del envío de emails, se ha utilizado la librería PHP 'PHPMailer' [51].

PHPMailer es una potente librería para el envío de emails en aplicaciones PHP, pudiendo mandar contenido en formato HTML, adjuntar ficheros, autentificación SMTP etc., además de ser la librería más popular en PHP para desempeñar funciones de correo electrónico.

Una vez dicho esto, se exponen a continuación los diversos controladores de la aplicación y la función de cada uno de ellos:

- **Controladores/ardController.php:** este controlador es el encargado de interactuar con el dispositivo y recibirá dos parámetros GET: el identificador del dispositivo que hace la petición, y el código de barras leído. Como se ha explicado a lo largo de este proyecto, se realizará una acción u otra dependiendo del estado actual del asiento y del usuario identificado.
- **Controladores/funcionesComunes.php:** contiene funciones básicas que son llamadas desde varios scripts PHP. Para la utilización de las mismas es necesario importar este controlador en el script donde se deseen utilizar.
- **Controladores/loginController.php:** se encarga de la lógica del inicio o fin de sesión tanto en la aplicación principal como la de administración.
  - **Acciones POST**
    - **login:** recibe los parámetros necesarios para comprobar que la identificación del usuario es correcta e iniciará la sesión, redirigiendo a la página principal de la aplicación correspondiente.
  - **Acciones GET**
    - **logout:** cierra la sesión del usuario.
- **Controladores/reservaController.php:** controla la reserva de los asientos de la biblioteca.
  - **Acciones POST**
    - **reservar:** reserva un asiento y envía un email al usuario informando de los datos de su reserva.
    - **cancelarReserva:** cancela la reserva de un asiento
    - **incidencia:** crea una incidencia sobre el dispositivo asociado al asiento.
    - **liberar:** el bibliotecario o el administrador pueden liberar un asiento ocupado o reservado.
- **Controladores/usuarioController.php:** se encarga de cambiar la contraseña del usuario.
  - **Acciones POST**
    - **cambiaContraseña:** cambia la contraseña del usuario.
- **Admin/controladores/asientosController.php:** controla las acciones de la edición de los mapas de las bibliotecas.
  - **Acciones POST**
    - **cargarMapa:** carga el mapa en modo edición dada una biblioteca y una planta.

    - **actualizaPosición:** actualiza la posición de una mesa dadas sus coordenadas.
    - **modiMesa:** modifica los parámetros básicos de una mesa (número de asientos, rotación y si está activada o desactivada).
    - **crearMesa:** crea una mesa con un número determinado de asientos
    - **eliminarMesa:** elimina una mesa.
- **Admin/controladores/bibliotecasController.php:** proporciona los servicios para la gestión de las bibliotecas.
  - **Acciones POST**
    - **crearBiblio:** crea una biblioteca dado su nombre, dirección, número de plantas e imagen.
    - **cargarBiblio:** carga los datos de una biblioteca en una ventana modal para su posterior modificación.
    - **modiBiblio:** modifica los datos de una biblioteca.
  - **Acciones GET**
    - **eliminarBiblio:** elimina una biblioteca.
- **Admin/controladores/estadisticasController.php:** mediante este script se cargan los datos de los diferentes gráficos del módulo de estadísticas.
  - **Acciones POST**
    - **graficaMensualTodasBiblios:** prepara los datos para la gráfica 'Mensual por biblioteca'.
    - **graficaRango:** carga los datos para la gráfica 'Total en rango de fechas'.
    - **graficaTipo:** suministra los datos en la gráfica 'Por tipo de usuario, biblioteca y rango de fechas'.
- **Admin/controladores/incidenciasController.php:** simplemente abrirá y/o cerrará incidencias.
  - **Acciones GET**
    - **nuevoEstado:** abre o cierra una incidencia.
- **Admin/controladores/usuariosController.php:** dará los servicios necesarios para gestionar los usuarios del sistema.
  - **Acciones POST**
    - **crearUsuario:** crea un usuario dados sus datos y envía un email a dicho usuario informando de sus datos de acceso.
    - **modificarUsuario:** como su propio nombre indica, sirve para modificar los datos de un usuario.
    - **nuevaContrasenia:** genera una contraseña aleatoria alfanumérica [52] y la envía al usuario por email.

Interacciones Vista – Controlador

De un modo más gráfico se muestra en la Figura 51 la interacción entre las diferentes vistas de la aplicación principal y los controladores.

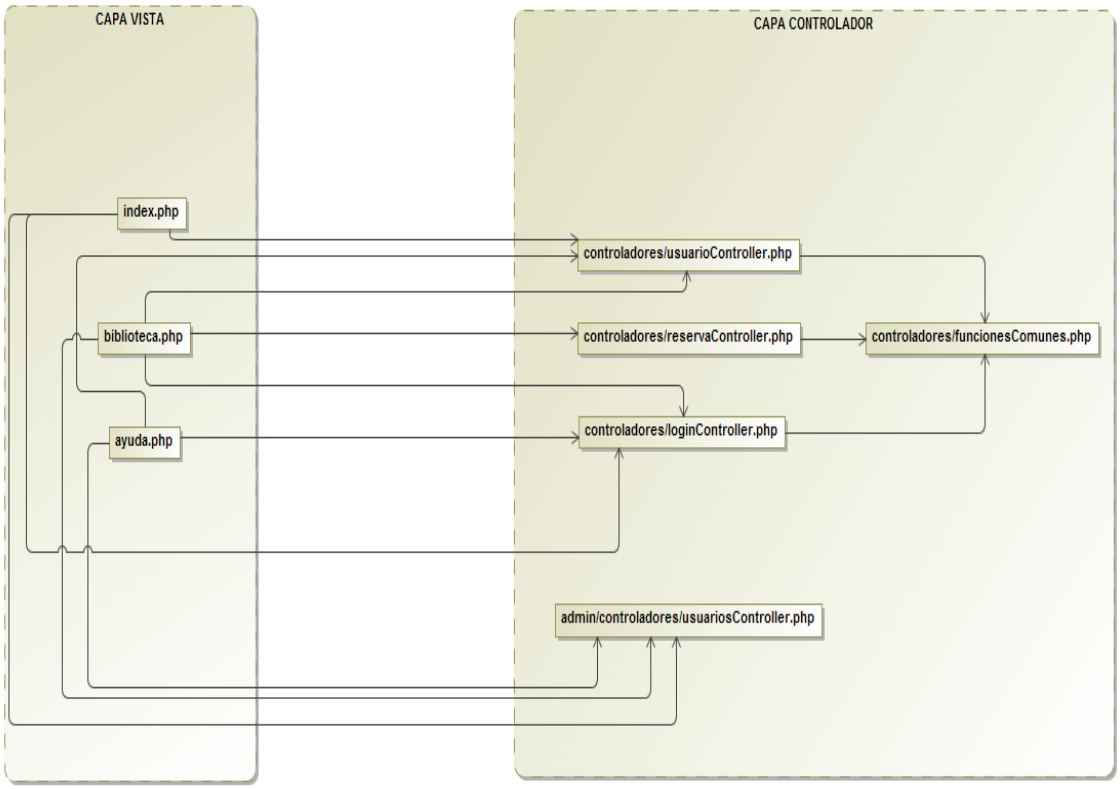

Figura 51: Interacción Vista-Controlador aplicación principal.

En la Figura 52, del mismo modo se exponen las interacciones en la aplicación de administración.

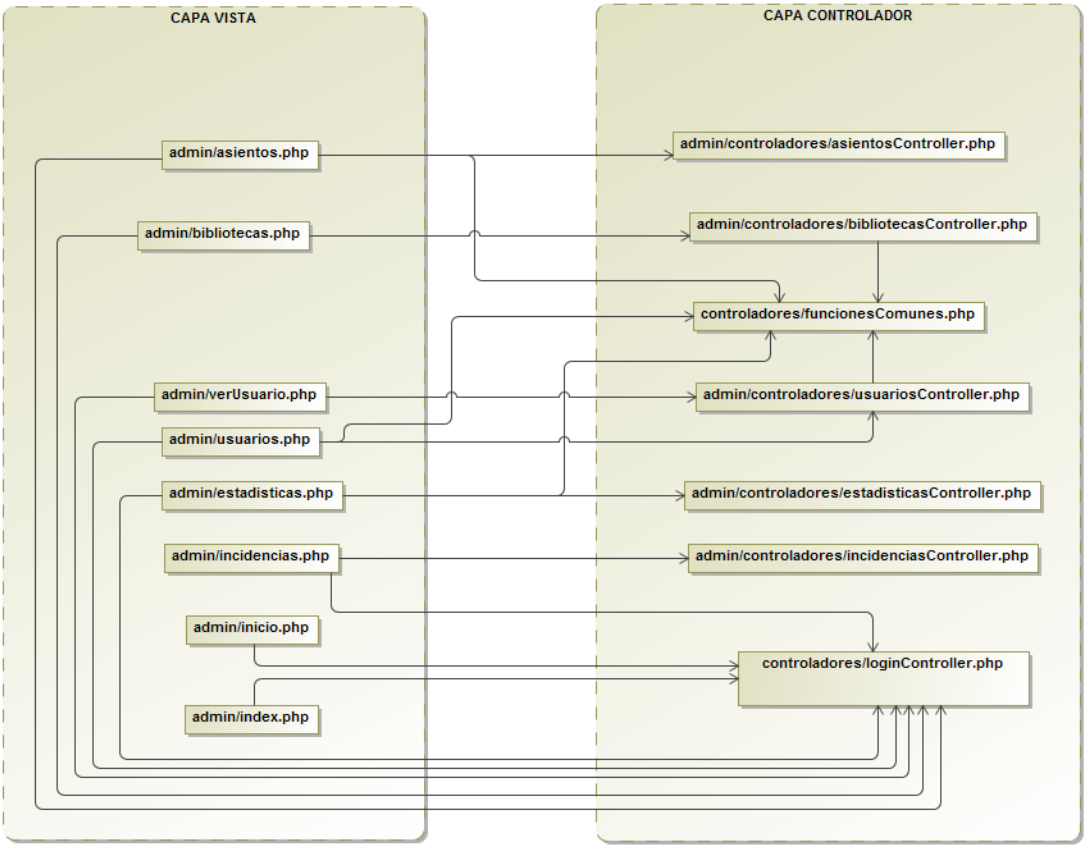

Figura 52: Interacción Vista-Controlador aplicación de administración.

4.2.3 | Capa de acceso a datos

Esta capa es la encargada de realizar las interacciones con la base de datos. No está implementada del modo tradicional mediante una serie de clases que se mapean en la base de datos como si fueran tablas, su implementación es totalmente distinta. De este modo conseguimos interactuar directamente con la base de datos sin tener clases de por medio y además su funcionamiento es bastante simple: se utiliza una clase PHP (Figura 53) que cuenta con los métodos necesarios para realizar consultas y obtener dicha información en forma de array, así como insertar, modificar y eliminar columnas de las diferentes tablas de la base de datos. Dicha clase está constituida en el script *clases/bd.class.php.*

Esta capa interactuará con prácticamente todos los scripts de la capa controlador que requieran de operaciones con la base de datos.

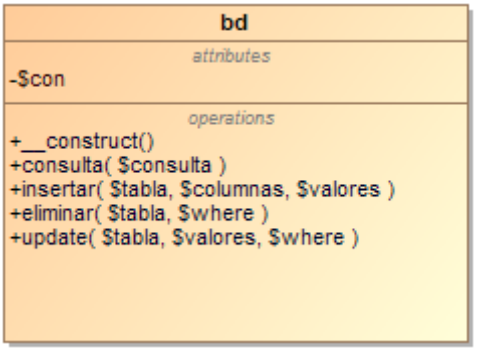

Figura 53: Clase BD.

La implementación de la clase BD es muy sencilla. El atributo *$con* es el manejador de la conexión. Dicha conexión se establece en la creación del objeto mediante su constructor (*__construct()*).

Los demás métodos se exponen a continuación:

- **consultar:** este método realizará una consulta (select) a la base de datos y devolverá el resultado en forma de array.
- **insertar:** insertará en la tabla *$tabla* un nuevo registro con un conjunto de columnas *$columnas* y unos valores *$valores* para cada una de ellas.
- **eliminar:** elimina de la tabla *$tabla* los registros que cumplan la condición *$where.*
- **update:** actualiza los registros *$valores* de la tabla *$tabla* que cumplan las condiciones *$where.*

Dado que el sistema está orientado para ser usado por múltiples usuarios al mismo tiempo, todos estos métodos implementan mecanismos de control de concurrencia en el acceso a la base de datos. Cuando se llama a uno de estos métodos, en primer lugar, se bloquea la tabla a la que se pretende acceder para que ninguna otra operación acceda o manipule la información mientras la llamada actual ejecuta las acciones pertinentes sobre la base de datos. Una vez que se ha terminado la transacción, se libera la tabla.

4.3 | Arquitectura de la base de datos

La base de datos que se ha desarrollado no es una base de datos compleja, dado que el sistema que se ha propuesto es bastante simple. Consta de 6 tablas y las distintas relaciones entre ellas. En la Figura 54 se muestra el diagrama E/R de la base de datos.

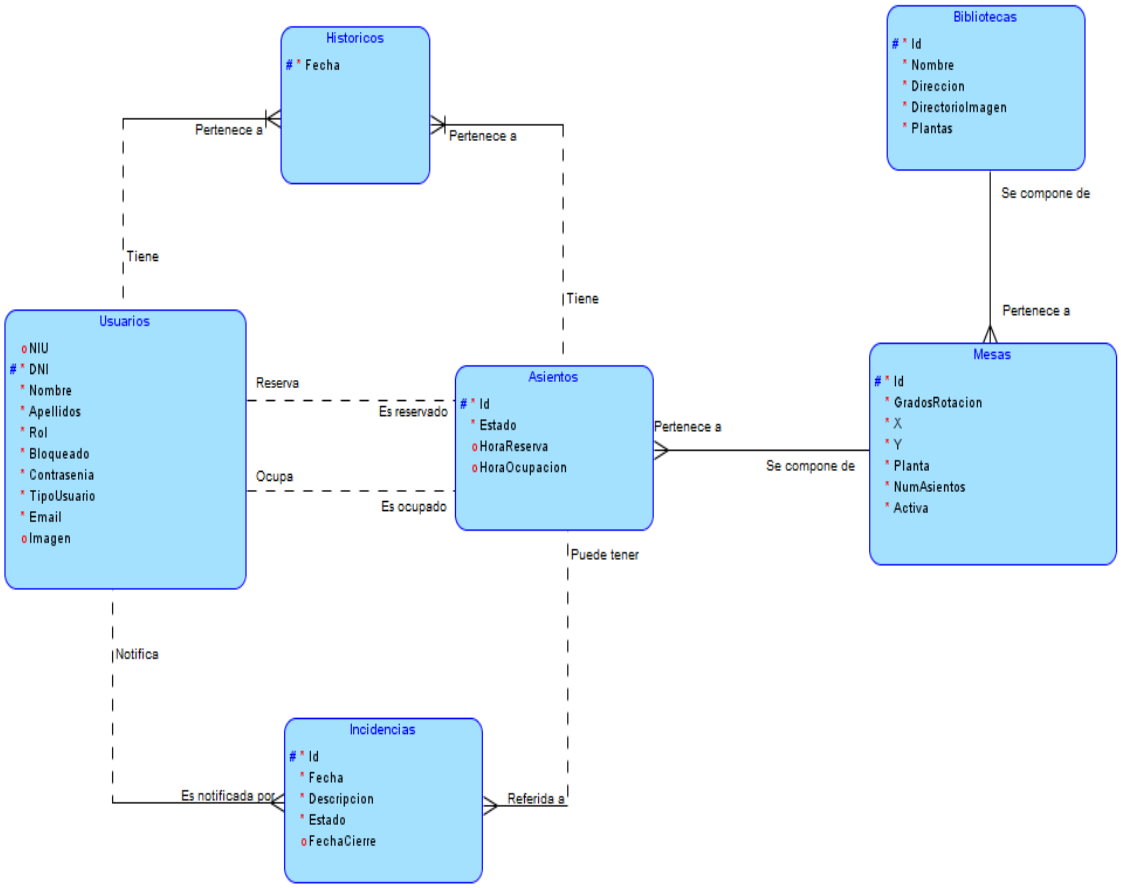

Figura 54: Diagrama E/R de la base de datos.

4.3.1 | Tabla Usuarios

En esta tabla se almacenarán todos los usuarios del sistema.

- Atributos
  - **NIU:** número de identificación universitaria del usuario. Este atributo puede ser opcional ya que puede haber usuarios que no dispongan de esta identificación (PDI o PAS).
  - **DNI:** documento nacional de identidad que actúa como clave primaria de la tabla.
  - **Nombre:** nombre del usuario.
  - **Apellidos:** apellidos del usuario.
  - **Rol:** el usuario puede tener 3 posibles roles, que son usuario simple (un alumno, por ejemplo), bibliotecario o administrador del sistema. Cada uno de estos roles se identifican mediante los enteros 1, 2 y 3 respectivamente.
  - **Bloqueado:** indica si el usuario ha sido bloqueado en el sistema por alguna conducta impropia.
  - **Contrasenia:** contraseña del alumno. Para mayor seguridad se encripta mediante un algoritmo basado en el DES de Unix. Este algoritmo genera hash

débiles e inesperados si no se utiliza una clave como base del algoritmo. En este caso de utilizará como clave el DNI del usuario.
  - **TipoUsuario:** tipo de usuario del sistema. Puede ser alumno, PDI o PAS.
  - **Email:** email del usuario.
  - **Imagen:** nombre de la imagen del usuario.

- Relaciones
  - **1:N con Incidencias:** el usuario puede abrir varias incidencias.
  - **1:1 con Asientos:** se pueden observar dos relaciones con la misma cardinalidad y de comportamiento similar. El usuario solamente podrá ocupar o reservar un asiento al mismo tiempo.
  - **1:N con Historicos:** el usuario puede tener más de un registro de la tabla que almacena el histórico de ocupaciones de los asientos.

4.3.2 | Tabla Bibliotecas

Tabla que almacena las bibliotecas de las que dispone el sistema.

- Atributos
  - **Id:** identificación de cada biblioteca y clave principal de la tabla. Esta clave es un número entero autogenerado.
  - **Nombre:** nombre de la biblioteca.
  - **Dirección:** dirección de la biblioteca.
  - **DirectorioImagen:** ruta de la imagen que representa a la biblioteca en la página principal de la biblioteca.
  - **Plantas:** número de plantas de la biblioteca. Una biblioteca tendrá tantos mapas como plantas haya.

- Relaciones
  - **1:N con Mesas:** una biblioteca debe estar constituida por una o varias mesas.

4.3.3 | Tabla Mesas

Cada una de las mesas de las bibliotecas se almacena en esta tabla.

- Atributos
  - **Id:** una mesa está identificada por este campo. Es la clave primaria del sistema además de ser numérica autogenerada.
  - **GradosRotacion:** en la creación del mapa se pueden especificar los grados de rotación de una mesa para disponerla en diferentes orientaciones.
  - **X:** coordenada X que ocupa en el mapa al que pertenece.
  - **Y:** coordenada Y que ocupa en el mapa al que pertenece.
  - **Planta:** planta en la que se sitúa en la biblioteca.

  - **NumAsientos:** número de asientos de los que consta la mesa.
  - **Activa:** una mesa puede activarse o no debido a múltiples causas (problema en uno de los dispositivos, utilización de la misma para otro fin, etc.)

- Relaciones
  - **1:N con Asientos:** una mesa está compuesta por varios asientos.

4.3.4 | Tabla Asientos

Es la tabla más importante de la base de datos, ya que todo el sistema se basa en los asientos.

- Atributos
  - **Id:** clave principal de la tabla, numérica y autogenerada para facilitar la creación de cada uno de los asientos.
  - **Estado:** estado actual del asiento. Un asiento podrá estar ocupado, libre o reservado. Cada uno de estos estados se identifican con los valores 0, 1 y 2 respectivamente.
  - **HoraReserva:** fecha y hora en la que un usuario ha reservado el asiento.
  - **HoraOcupacion:** fecha y hora en la que un usuario ha ocupado el asiento.

- Relaciones
  - **1:N con Mesas:** un asiento solo podrá pertenecer a una mesa concreta.
  - **1:1 con Usuarios:** las dos relaciones de este tipo indican el usuario que ha reservado u ocupado el sitio.
  - **1:1 con Históricos:** un asiento puede tener varios registros de ocupaciones.
  - **1:1: con Incidencias:** el dispositivo de un asiento puede tener más de una incidencia.

4.3.5 | Tabla Históricos

Está tabla es la base del módulo de estadísticas y en ella de almacenan todos los registros de ocupaciones, es decir, cuando un usuario ocupa un asiento se guarda un registro en esta tabla.

- Atributos
  - **Fecha**: fecha en la que se ha ocupado el asiento. Este atributo actúa como clave primaria junto al usuario que ha producido la ocupación y junto al asiento que ha sido objeto dicha ocupación.

- Relaciones
  - **1:N con Usuarios:** un registro debe estar asociado a un usuario.
  - **1:N con Asientos:** un registro debe estar asociado a un asiento.

4.3.6 | Tabla Incidencias

En está tabla se anotan las incidencias que haya podido tener algún dispositivo referente a alguna avería, deterioro, etc. o alguna anomalía en el funcionamiento del dispositivo.

- Atributos
  - **Id:** identificación de la incidencia y clave primaria de la tabla. Sus valores son números enteros autogenerados.
  - **Fecha:** fecha en la que se produce la incidencia.
  - **Descripción:** descripción de la incidencia.
  - **Estado:** indica si la incidencia está abierta (1) o cerrada (0).
  - **FechaCierre:** fecha en la que se soluciona la incidencia.

- Relaciones
  - **1:N con Usuarios:** relaciona al usuario que ha notificado la incidencia.
  - **1:N con Asientos:** una incidencia siempre debe estar asociada a un asiento.

CAPÍTULO 5 | RESULTADOS Y PRUEBAS

Dada las características de esta aplicación es bastante complicado la automatización de pruebas unitarias y el control de resultados. En su lugar se hicieron pruebas de cada uno de los módulos o funcionalidades del sistema conforme se iban completando, además de una gran batería de pruebas una vez estuvo acabada la plataforma al completo.

Al estar el sistema formado por una aplicación web y un dispositivo interactivo, las pruebas están fuertemente ligadas a la interacción del usuario con el sistema y por ende a los casos de uso de la aplicación.

Todas las pruebas se realizaron desde PC, y para el caso de la aplicación principal también se realizaron algunas pruebas en Smartphone. Tanto para las pruebas en PC como en Smartphone fueron en una red local. Sin embargo, no ha sido posible la realización de pruebas con varios usuarios simultáneamente, aunque como se dijo anteriormente, la base de datos implementa mecanismos para el acceso concurrente.

Tanto las pruebas relacionadas con el dispositivo como las relacionadas con la aplicación web se limitan a las funcionalidades críticas y, por ello, de las que depende el correcto funcionamiento del sistema. A continuación, se citan todas ellas brevemente ya que en el apéndice A se describen con mayor nivel de detalle.

1) Dispositivo
   a) Ocupar un asiento
   b) Desocupar un asiento

2) Aplicación principal
   a) Paginación entre las diferentes vistas
   b) Inicio de sesión
   c) Cierre de sesión
   d) Cambio de contraseña
   e) Generación de una nueva contraseña
   f) Reserva de un asiento
   g) Cancelación de la reserva
   h) Ver datos reserva/ocupación de un asiento
   i) Liberación de un asiento
   j) Notificación de una incidencia

3) Aplicación de administración
   a) Paginación entre las diferentes vistas
   b) Inicio de sesión
   c) Cierre de sesión

- Módulo Usuarios

   d) Búsqueda de usuarios mediante el filtro
   e) Creación de un usuario
   f) Modificación de un usuario
   g) Cambiar la imagen de un usuario
   h) Generación de una nueva contraseña para un usuario

- Módulo Asientos

   i) Cargar mapa de distintas bibliotecas y plantas
   j) Creación de una mesa
   k) Modificación de la posición de una mesa
   l) Modificación de los datos de una mesa
   m) Desactivación/Activación de una mesa
   n) Eliminación de una mesa

- Módulo Estadísticas

   o) Cargar diferentes datos del gráfico 'Mensual por biblioteca'
   p) Cargar diferentes datos del gráfico 'Total en rango de fechas'
   q) Cargar diferentes datos del gráfico 'Por tipo de usuario, biblioteca y rango de fechas'

- Módulo Bibliotecas

   r) Creación de una biblioteca
   s) Modificación de los datos de una biblioteca
   t) Modificación imagen de una biblioteca
   u) Eliminación biblioteca

- Módulo Incidencias

   v) Cerrar una incidencia
   w) Abrir una incidencia
   x) Buscar incidencias

CAPÍTULO 6 | CONCLUSIONES Y TRABAJOS FUTUROS

En este último capítulo se resume cómo surgió la idea para este trabajo, lo que ha supuesto su realización, lo que supondría una implantación real de Librarino y además se habla sobre posibles mejoras futuras del sistema.

6.1 | Conclusiones

Librarino surgió como una idea fugaz, una broma entre amigos durante una tarde de estudio en la biblioteca. Esa pequeña idea fue fraguándose y consolidándose con el paso de los días y tras mucho discurrir decidí transformarla en este proyecto.

Gracias a la ilusión y el tremendo esfuerzo que ha supuesto para mí la realización de este trabajo, que compatibilizo con un trabajo de jornada completa, he podido desarrollar algo que seguro supondría un gran avance en nuestra universidad que mejoraría la experiencia de todos los alumnos con las bibliotecas.

Con la misma ilusión espero que este trabajo no quede en un estante como un trabajo de fin de grado más, sino que algún día la Universidad de Málaga se plantee seriamente implantar un proyecto como este o similar, sin duda nuestras bibliotecas y la propia universidad se convertirían en referentes en toda España.

El principal objetivo de este proyecto era construir un dispositivo que facilitara identificación de asientos libres en las bibliotecas. Este objetivo se ha visto cumplido satisfactoriamente, dada la buena elección de los componentes y la facilidad para su manejo que ha supuesto que el desarrollo del mismo sea sencillo, aunque como en todos los aspectos, con alguna dificultad que se ha conseguido solventar.

El segundo gran objetivo, quizás al nivel del primero, era que la información de los asientos fuera accesible desde cualquier lugar, sin tener que estar en la biblioteca necesariamente. Para cumplir este objetivo se ha desarrollado una aplicación web, que como el lector habrá podido observar, es una aplicación bastante simple, aunque con todas las funcionalidades que se preveían y alguna extra que ha ido surgiendo. Esta simpleza ha conseguido solventar un objetivo en segundo plano que cualquier software debería cumplir: un manejo sencillo.

Cuando se desarrolla cualquier proyecto es importante aprender de él, adquirir nuevas habilidades y experiencia. Arduino era para mí una tecnología casi desconocida cuando

empecé este proyecto, y gracias a él he pasado de estar a punto de quemar un led por no ponerle una resistencia, a aconsejar a demás compañeros y amigos sobre cómo debían ser sus diseños, qué componentes debían utilizar e incluso de hacer un pequeño proyecto Arduino para mejorar una escultura de arte de una buena amiga mía.

Otro de los aspectos importantes, más allá de lo estrictamente informático, es la habilidad de organización y planificación. El tener que realizar y seguir una metodología y planificación, me ha permitido adquirir unos buenos hábitos de trabajo, fijarme unas fechas de entrega, unas metas y unas pautas a seguir, además de ir viendo con ilusión como iba creciendo mi idea poco a poco, pero con paso firme.

Llegados a este punto, seguro que el lector se habrá preguntado de donde viene el nombre de Librarino, pues bien es hora de revelar dicho secreto:

Library + Arduino = Librarino

6.2 | Trabajos futuros

El principal trabajo sería adaptar la aplicación web a Smartphones mediante la realización de las correspondientes apps en Android e iOS, aunque la aplicación se ha adaptado mediante CSS a cualquier tipo de pantalla. Sin embargo, es mucho más funcional para el usuario disponer de una app en lugar de tener que estar continuamente entrando a la web.

Otro de los objetivos importantes es construir una carcasa para el dispositivo e idear un método de montaje y configuración en serie para suministrar grandes pedidos si se decidiera comercializarlo. En la Figura 55 puede observarse una primera aproximación de cómo podría ser el dispositivo.

Figura 55: Posible diseño del dispositivo.

El tema de la seguridad de la aplicación también se ha dejado de lado. Sería conveniente realizar una pequeña auditoría de seguridad de la aplicación para la prevención de, por ejemplo, ataques de inyección de SQL, entre muchos otros tipos de agujeros de seguridad.

En cuanto a las futuras mejoras en la aplicación web, se proponen las siguientes:

- Mejorar la funcionalidad de la edición de los mapas, añadiendo funcionalidades como la posibilidad de definir zonas, poder seleccionar varias mesas y arrastrarlas a una nueva ubicación, eliminarlas, etc.

- Al igual que se realiza un histórico de ocupaciones, se propone realizar otro para las reservas y realizar estadísticas de las mismas para comparar, por ejemplo, el número de reservas que se convierten finalmente en ocupaciones.

- Diseñar un sistema de bloqueos de alumnos automatizado para prevenir conductas poco éticas. El ejemplo más práctico es el alumno que está continuamente reservando y cancelando. En este caso, al hacer 5 ciclos “reserva-cancelación” en menos de 30 minutos quedaría automáticamente bloqueado.

- Realizar pruebas de carga del sistema una vez que este montado en un servidor real.

APÉNDICE A | MANUAL DE USUARIO

En este primer apéndice el lector aprenderá a manejar el sistema, así como una descripción más detalla de sus distintas funcionalidades. El manual estará dividido en las tres figuras o roles principales que intervienen en el sistema. Una vez detallada una funcionalidad para un rol no se volverá a repetir para el siguiente, dado que muchas son comunes a todos los roles.

A.1 | Manual del rol 'Usuario'

El rol más simple únicamente tendrá acceso al dispositivo y a la aplicación principal.

A.1.1 | Instrucciones del dispositivo

Cada asiento está provisto de un dispositivo donde el usuario puede ver el estado actual del asiento. Solo podrá ser ocupado un asiento libre o un asiento reservado por él mismo.

Para la ocupación de un asiento simplemente se debe pasar el código de barras del carnet universitario por el lector que tiene el dispositivo. En ese momento la bombilla pasará a color rojo indicando que el asiento está ocupado. El usuario debe saber que esta ocupación dura 3 horas y una vez expirado ese tiempo deberá repetir el procedimiento, siempre que antes otro usuario no haya reservado el asiento.

En cuanto a la liberación del asiento, es necesario volver a pasar una tarjeta por el lector del dispositivo y observará que la bombilla está ahora verde, es decir, el asiento ha vuelto a estar libre.

El usuario deberá estar obligatoriamente dado de alta en el sistema para poder disfrutar de las ventajas de Librarino. En el caso de que no sea así, debe dirigirse a los mostradores de información de cualquier biblioteca para proceder a su alta.

En todos los dispositivos hay una pequeña pantalla que muestra algunos datos del asiento. En la Figura 56 se detallan todos los campos de la misma.

Figura 56: Detalle del display del dispositivo.

A.1.2 | Instrucciones de la aplicación web principal

Ver mapa de una biblioteca

La primera toma de contacto con la aplicación web será a través de la página de inicio (Figura 57) donde se muestra en forma de lista en dos columnas cada una de las bibliotecas con una imagen del exterior del edificio, el nombre de la misma y el número de asientos libres de los que dispone.

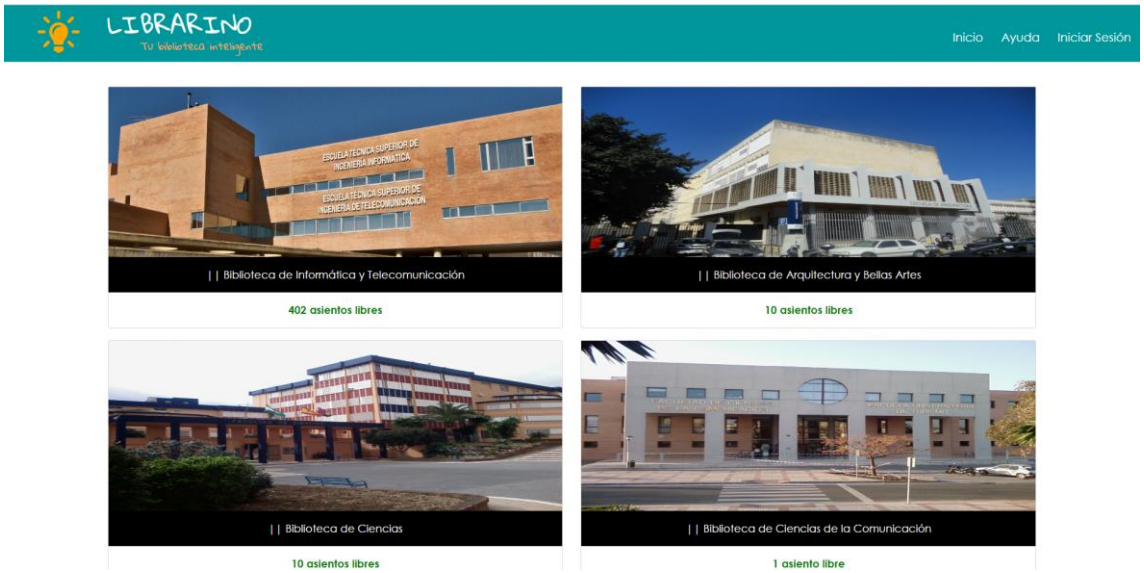

Figura 57: Página inicial de la aplicación web.

Para entrar en alguna biblioteca solo hay que pulsar sobre la imagen de la misma. Una vez seleccionada la biblioteca, se muestra en una nueva página un mapa con la disposición de los asientos en la sala de lectura seleccionada. En la Figura 58 se puede observar la página que contiene al mapa.

Es bastante que probable que alguna biblioteca disponga de más de una planta y queramos ver el mapa de alguna de ellas. Justo debajo del encabezado de la página hay un pequeño menú en forma de pestañas donde el usuario puede ir cambiando de planta.

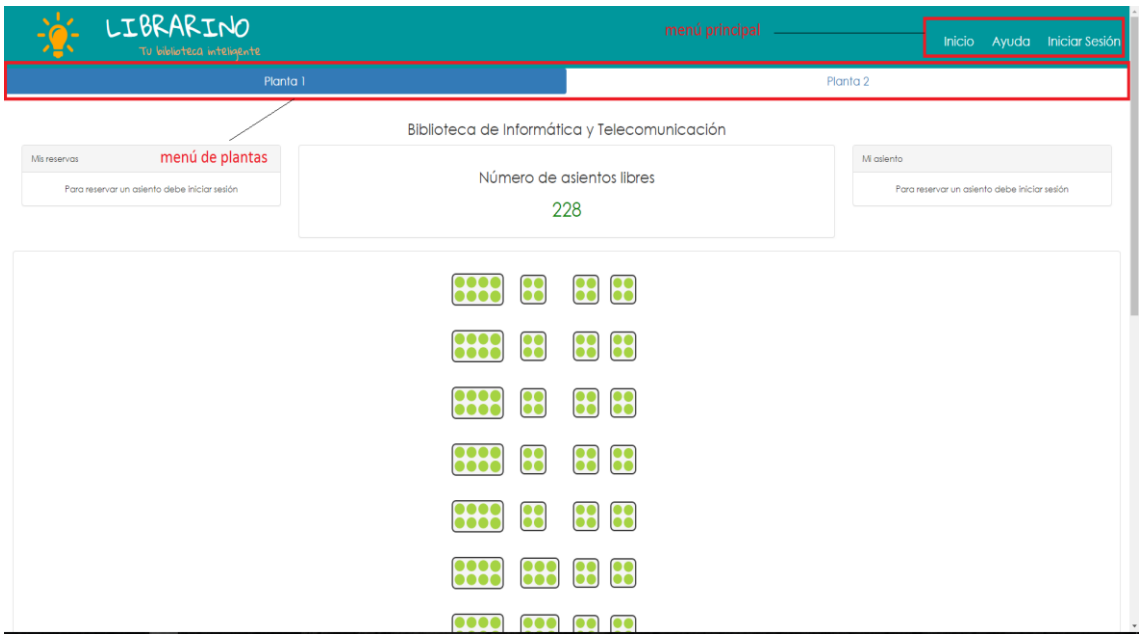

Figura 58: Detalle de los menús de la app principal.

Generación de una nueva contraseña

El olvidarnos de nuestra contraseña no supone ningún problema, pues la aplicación dispone de un generador automático de contraseñas. Seguiremos estos pasos:

1) Pulsar el botón ‘Iniciar Sesión’ en el menú principal.
2) Seleccionar en la ventana flotante ‘¿Ha olvidado su contraseña?’.
3) En la nueva ventana flotante que se abre es necesario introducir el DNI del usuario y pulsar ‘Generar’.
4) El usuario recibirá un correo de acceso con la nueva contraseña.

Inicio de sesión

El inicio de sesión puede hacerse desde cualquier página de la aplicación principal dado que el encabezado es el mismo en todas ellas. El usuario podrá acceder a la aplicación tanto con su DNI como con su NIU.

1) Seleccionar 'Inicio de sesión' en el menú principal.
2) Introducir usuario y contraseña en el formulario que se muestra en la ventana flotante.
3) Pulsar 'Iniciar sesión'.

Reserva de un asiento

La duración de una reserva tendrá una duración de una hora, momento en el cual el asiento volverá a estar libre si el asiento no ha sido ocupado por el usuario que efectuó la reserva.

1) En primer lugar, debemos de seleccionar la biblioteca que más nos interese y ver su mapa (ver B.1.2.1).
2) Iniciar sesión (ver B.1.2.2).
3) Una vez hayamos escogido un asiento libre (asientos con color verde), pulsamos sobre él y se abrirá una ventana flotante. En dicha ventana pulsamos el botón reservar.

El asiento que hayamos reservado pasará a color azul y emitirá un parpadeo [53] para identificarlo de un modo más rápido. En el panel 'Mis reservas' (Figura 59) se pueden ver los datos de la reserva. Estos mismos datos serán notificados al usuario mediante un correo electrónico.

Figura 59: Panel 'Mis reservas'.

Cancelación de una reserva

Una vez efectuada una reserva el alumno tiene la posibilidad de poder cancelarla. Simplemente debe pulsar ‘Cancelar reserva’ del panel ‘Mis reservas’.

Notificación de una incidencia

Las incidencias pueden ser notificadas en cualquier asiento sea cual sea el rol del usuario.

1) Dirigirnos al mapa de una biblioteca.
2) Iniciar sesión.
3) Pulsar sobre el asiento del mapa del que se quiera dejar constancia de la incidencia.
4) En la ventana flotante que se abrirá, realizar una descripción del problema en el apartado ‘Notificar una incidencia’.
5) Pulsar ‘Notificar’ para acabar y guardar la incidencia en el sistema.

Cambio de contraseña

Una vez dado de alta un usuario se le proporciona una contraseña alfanumérica y aleatoria que puede ser difícil de recordar, por lo que es posible cambiar dicha contraseña.

1) Dirigirnos al mapa de una biblioteca.
2) Iniciar sesión
3) En el menú principal pulsar sobre ‘Hola {nombre del usuario}’.
4) Seleccionar del desplegable ‘Cambiar contraseña’.
5) Introducir la nueva contraseña en el campo ‘Nueva contraseña’ y repetirla en el campo ‘Repita contraseña’.
6) Pulsar el botón ‘Guardar’.

Cerrar sesión

1) Iniciar sesión.
2) En el menú principal pulsar sobre ‘Hola {nombre del usuario}’.
3) Pulsar sobre la opción ‘Cerrar sesión’ del desplegable.

A.2 | Manual para el rol ‘Bibliotecario’

Las instrucciones del dispositivo y las funcionalidades de la aplicación principal ya descritas en el rol ‘Usuario’ se omiten al ser las mismas para cualquier tipo de rol.

A.2.1 | Instrucciones de la aplicación web principal

Ver los datos de una ocupación/reserva

El bibliotecario y el administrador podrán ver quién ha ocupado o reservado un determinado asiento.

1) Dirigirnos al mapa de una biblioteca.
2) Iniciar sesión.
3) Pulsar sobre el asiento que se desee.
4) En la ventana flotante (Figura 60) puede ver los datos de quién ha ocupado/reservado el asiento, así como el día y la hora.

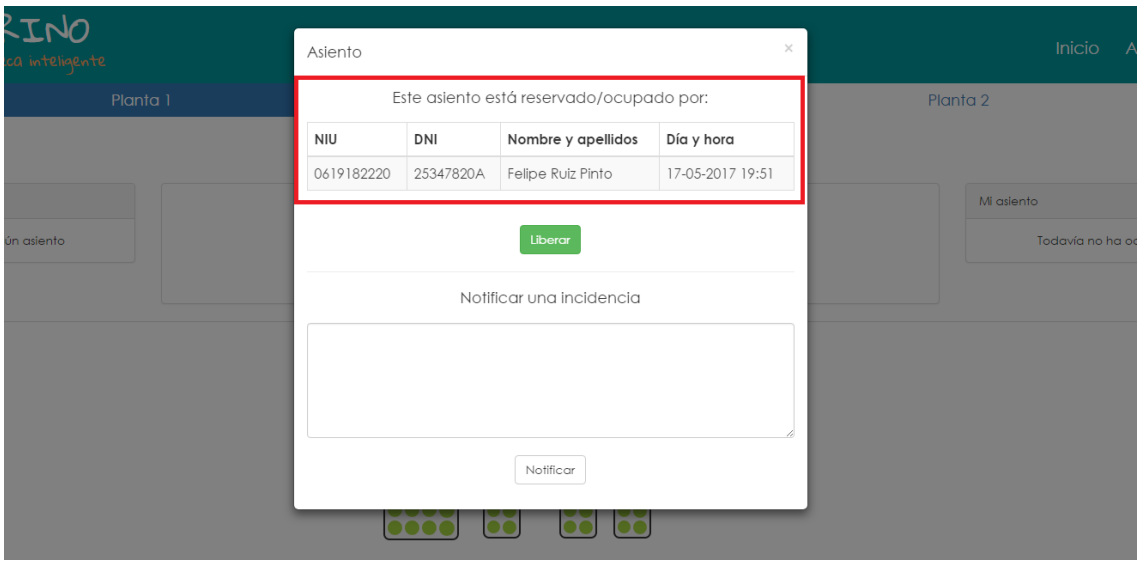

Figura 60: Datos de una reserva/ocupación.

Liberar un asiento

Es posible la liberación de un asiento sin que se tenga que interactuar con el dispositivo. Esta acción solo la podrá llevar a cabo los roles ‘bibliotecario’ y ‘administrador’.

1) Dirigirnos al mapa de una biblioteca.
2) Iniciar sesión.
3) Pulsar sobre el asiento que se desee.
4) En la ventana flotante se pulsa en el botón ‘Liberar’.

A.2.2 | Instrucciones de la aplicación web de administración

A la aplicación de administración únicamente podrá acceder los roles ‘Bibliotecario’ y ‘Administración’. La dirección para acceder a esta aplicación es *{direcciónWebPrincipal}/admin.*

Iniciar sesión

Es obligatorio, para acceder a la aplicación de administración, identificarse mediante el usuario y contraseña con las mismas credenciales que en la aplicación principal. Por tanto, esta será la primera página a la que accedamos. En el caso de que un alumno intente acceder, la aplicación detecta su rol y muestra un mensaje indicando que no tiene permisos. La Figura 61 muestra la pantalla principal.

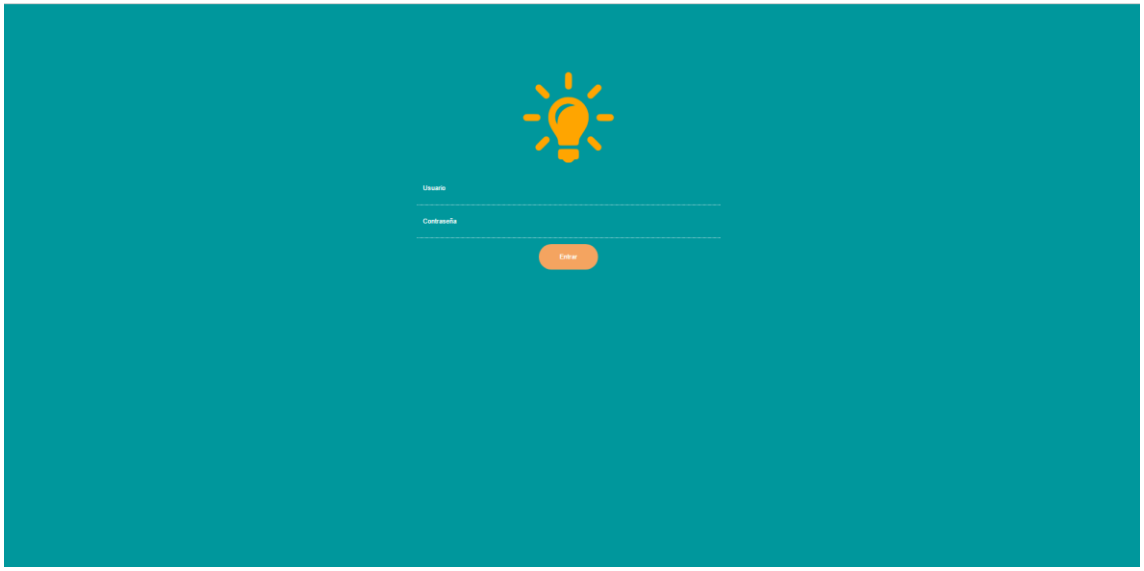

Figura 61: Formulario de identificación de la aplicación de administración.

Navegación

Para facilitar la navegación se dispone de un menú lateral (Figura 62) desde el que se podrá acceder a los diferentes módulos.

Figura 62: Pantalla de bienvenida de la app de admón.

Módulo Usuarios

El bibliotecario solo podrá gestionar usuarios con el rol ‘usuario’. El administrador tendrá los permisos necesarios para modificar usuarios de cualquier rol.

Buscar usuarios

1) Acceder desde el menú al módulo de usuarios.
2) En el filtro podrá buscar por nombre, apellidos, DNI o NIU.
3) En la tabla aparecerán los resultados que coincidan con el filtro. La búsqueda es por coincidencia en cualquier parte del campo buscado.

Modificar datos de un usuario

1) Acceder desde el menú al módulo de usuarios.
2) Buscar al usuario que se desee modificar mediante el filtro.
3) Pulsar el botón con el icono de un lápiz en la columna ‘Modi.’
4) En la nueva página se puede encontrar un formulario con los datos del usuario (Figura 63).
5) Modificar el que se desee y pulsar el botón ‘Guardar’.

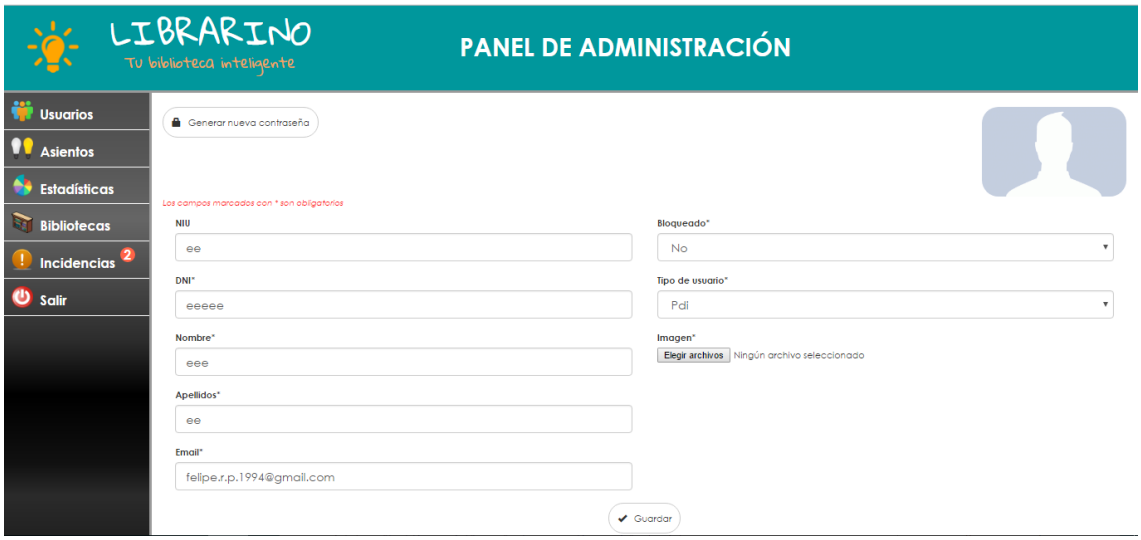

Figura 63: Formulario edición usuario

Modificar imagen de un usuario

1) Acceder desde el menú al módulo de usuarios.
2) Buscar al usuario que se desee modificar mediante el filtro.
3) Pulsar el botón con el icono de un lápiz en la columna ‘Modi.’
4) En la nueva página se puede encontrar un formulario con los datos del usuario.
5) En el apartado ‘Imagen’ pulsar el botón ‘Elegir archivos’.

6) Seleccionar la nueva imagen del alumno.
7) Pulsar el botón ‘Guardar’

Generar una nueva contraseña a un usuario

1) Acceder desde el menú al módulo de usuarios.
2) Buscar al usuario que se desee modificar mediante el filtro.
3) Pulsar el botón con el icono de un lápiz en la columna ‘Modi.’
4) En la nueva página se puede encontrar un formulario con los datos del usuario.
5) Pulsar el botón ‘Generar nueva contraseña’ en la parte superior izquierda de la pantalla.
6) El usuario recibirá su nueva contraseña mediante un correo electrónico.

Crear un nuevo usuario

1) Acceder desde el menú al módulo de usuarios.
2) Pulsar el botón ‘Nuevo usuario’ situado en la esquina superior derecha.
3) En la ventana flotante, rellenar los datos del usuario.
4) Pulsar el botón ‘Guardar’.

Módulo Asientos

El bibliotecario solo podrá gestionar el mapa de la biblioteca a la que pertenezca. El administrador tendrá los permisos necesarios para modificar usuarios de cualquier rol.

Visualización del mapa de una biblioteca

Para visualizar el mapa de edición de una biblioteca (Figura 64) se han de seguir los siguientes pasos:

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar la planta que se desea ver mediante los filtros de la parte superior de la pantalla.

Figura 64: Pantalla de edición de un mapa

Crear una nueva mesa

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar el mapa que se desea ver mediante los filtros de la parte superior de la pantalla.
3) Pulsar dos veces en la celda en la que se desee crear la mesa.
4) Introducir los datos de la nueva mesa en la ventana flotante.
5) Pulsar el botón ‘Guardar’.

Modificar posición de una mesa

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar el mapa que se desea ver mediante los filtros de la parte superior de la pantalla.
3) Mantener pulsado sobre la mesa que se desea modificar y arrastrar a una nueva celda.

Modificar datos de una mesa

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar el mapa que se desea ver mediante los filtros de la parte superior de la pantalla.
3) Pulsar sobre la mesa que se desea modificar.
4) Modificar los datos que se estimen convenientes.
5) Pulsar botón ‘Guardar’.

Activar/Desactivar una mesa

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar el mapa que se desea ver mediante los filtros de la parte superior de la pantalla.
3) Pulsar sobre la mesa que se desea modificar.
4) Modificar el estado mediante el desplegable del apartado ‘Estado’.
5) Pulsar botón ‘Guardar’.

Eliminar una mesa

1) Acceder desde el menú al módulo de asientos.
2) Seleccionar el mapa que se desea ver mediante los filtros de la parte superior de la pantalla.
3) Pulsar sobre la mesa que se desea modificar.
4) Pulsar botón ‘Eliminar’.

Módulo Estadísticas

Mediante este módulo se podrán visualizar mediante gráficas distintos aspectos sobre las ocupaciones de los asientos en todas las bibliotecas. Hay 3 tipos de gráficos y para visualizar los datos de cada uno se deben seguir los siguientes pasos:

1) Acceder desde el menú al módulo de estadísticas.
2) Introducir los datos del filtro en el apartado que se desee.
3) Pulsar el botón ‘Cargar gráfico’.

Módulo Incidencias

Se gestionarán desde aquí todas las incidencias abiertas por cualquier usuario en la aplicación principal.

Buscar incidencias

1) Acceder desde el menú al módulo de incidencias.
2) Mediante el filtro situado en la parte superior buscar por los parámetros que se desee.
3) Pulsar el botón ‘Buscar’.
4) Aparecerá un listado con las incidencias que correspondan con la búsqueda.

Cerrar incidencia

Acceder desde el menú al módulo de incidencias.

1) Se listan las incidencias abiertas al acceder a la página.
2) Pulsar el botón ‘Cerrar’ situado en la columna ‘Operaciones’ en la incidencia que se desee abrir.

Abrir incidencia

1) Acceder desde el menú al módulo de incidencias.
2) En el filtro situado en la parte superior, seleccionar en el apartado ‘Estado’ el valor ‘Cerrada’.
3) Pulsar el botón ‘Buscar’.
4) Aparecerá un listado con las incidencias abiertas.
5) Pulsar el botón ‘Cerrar’ situado en la columna ‘Operaciones’ en la incidencia que se desee cerrar.

Cerrar sesión

Para salir de la aplicación únicamente se debe pulsar en el apartado ‘Salir’ del menú lateral.

A.3 | Manual para el rol ‘Administrador’

Tanto para el bibliotecario como para el administrador las funciones en cuanto al dispositivo y la aplicación principal son idénticas así que se omiten en este apartado, al igual que las funciones comunes en la aplicación de administración.

A.3.1 | Instrucciones de la aplicación web de administración

Módulo bibliotecas

A este módulo únicamente tendrá acceso el administrador, quien tendrá la posibilidad de gestionar todos los datos de las bibliotecas, así como crear nuevas.

Crear biblioteca

1) Acceder desde el menú al módulo de bibliotecas.
2) Pulsar el botón ‘Nueva biblioteca’ situado en la parte superior derecha.
3) Introducir los datos de la nueva biblioteca en el formulario que muestra la ventana flotante.
4) Pulsar el botón ‘Guardar’.

Modificar biblioteca

1) Acceder desde el menú al módulo de bibliotecas.
2) Buscar en el listado la biblioteca que se desee modificar.
3) Pulsar el botón con el dibujo de un lápiz de la columna ‘Operaciones’.
4) Modificar los datos de la biblioteca en el formulario que muestra la ventana flotante.
5) Pulsar el botón ‘Guardar’.

Modificar imagen de una biblioteca

1) Acceder desde el menú al módulo de bibliotecas.
2) Buscar en el listado la biblioteca que se desee modificar.
3) Pulsar el botón con el dibujo de un lápiz de la columna ‘Operaciones’.
4) En el apartado ‘Nueva imagen’ pulsar el botón ‘Seleccionar archivo’ y buscar en el navegador de archivos la nueva imagen de la biblioteca.
5) Pulsar el botón ‘Guardar’.

Eliminar una biblioteca

1) Acceder desde el menú al módulo de bibliotecas.
2) Buscar en el listado la biblioteca que se desee modificar.
3) Pulsar el botón con el dibujo de una cruz de la columna ‘Operaciones’.
4) Confirmar en el diálogo que abre el navegador web.

APÉNDICE B | MANUAL DE INSTALACIÓN

En este apéndice se mostrará al lector tanto la instalación de la aplicación y su entorno (servidor web, base de datos, despliegue), como los entornos de programación utilizados para el desarrollo del trabajo.

En el CD (Figura 65) que acompaña esta memoria se encuentra todo lo necesario para instalar la aplicación siguiendo el manual de este apéndice.

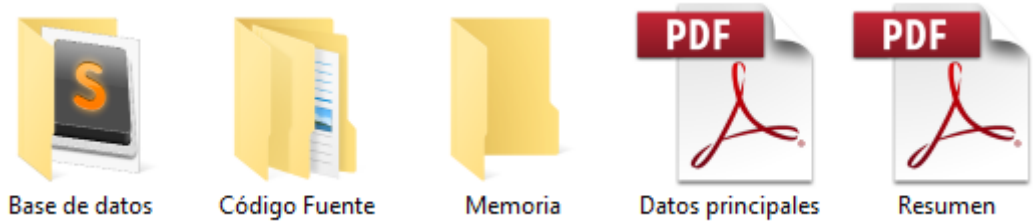

Figura 65: Raíz del CD.

- **Carpeta 'Base de datos':** se aloja una copia de la base de datos utilizada en la aplicación.
- **Carpeta 'Código Fuente':** contiene dos subcarpetas. Dentro de la subcarpeta 'Aplicación web' se encuentra el código fuente de la aplicación, del mismo modo en la subcarpeta 'Dispositivo' se puede encontrar el del dispositivo y las librerías necesarias para el correcto funcionamiento del mismo.
- **Carpeta 'Memoria':** contiene la presente memoria en formato PDF.

B.1 | Arduino IDE 1.8.1

Para el desarrollo del código del dispositivo de ha utilizado esta plataforma. Su descarga se puede realizar desde el apartado *Software* de su página oficial, donde simplemente pulsamos sobre el sistema operativo que deseemos y comenzará la descarga. La instalación es del tipo 'Siguiente – Siguiente' por lo que no alberga mayor problema.

Completada la instalación abrimos el software para proceder al código fuente:

1) Abrir *Arduino IDE*.
2) En la barra de herramientas pulsamos *Archivo* > *Abrir*.
3) Buscamos el código en el directorio *Código Fuente* > *Dispositivo* del CD.
4) Seleccionamos 'ardLibrarino y pulsamos 'Abrir'.

Es importante incluir las librerías para el manejo del módulo Wifi, la pantalla LCD y la placa USB. Para ello es nesario copiar el contenido de la carpeta *Código Fuente* > *Dispositivo* > *Librerías* del CD en el directorio del equipo *C:\Program Files (x86)\Arduino\libraries*. En la Figura 67 se puede observar el directorio que contiene todas las librerías.

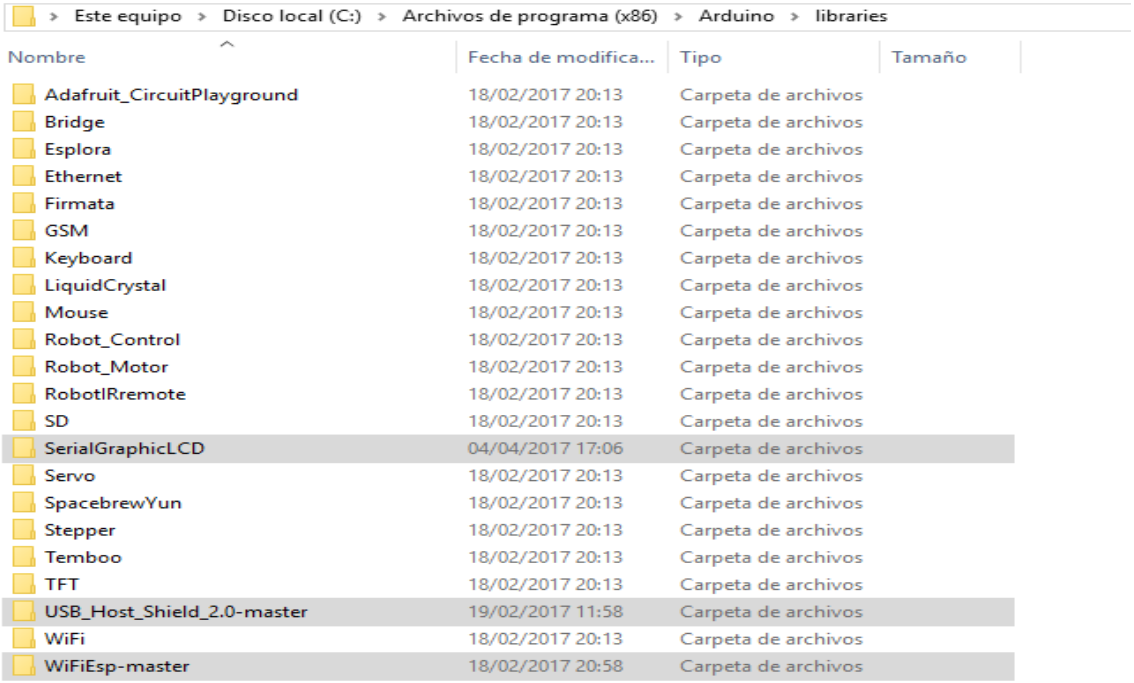

Figura 66: Directorio de librerias de Arduino IDE.

B.2 | Configuración del dispositivo

Será necesario configurar algunos parámetros del código para cargarlo en el dispositivo. Debemos indicar los datos de la conexión Wifi y el número de asiento que representa el dispositivo. En la Figura 68 se observan las variables que hay que modificar

```
// Variables para la conexión a internet
char ssid[] = "iPhone de Felipe";
char pass[] = "felipe8313";
char server[] = "172.20.10.2";

// Número de asiento
String id = "344";
```

Figura 67: Líneas de configuración del código del dispositivo.

Ahora que tenemos configurado el código para un dispositivo concreto es hora de cargarlo en la placa, previo montaje siguiendo las instrucciones de las conexiones del capítulo 4.

1) Pulsar *Herramientas* > *Puerto* y seleccionar el puerto COM donde aparezca la placa Arduino (Figura 69). Esto es necesario para conectar el dispositivo al software de desarrollo.
2) Pulsar el botón con el dibujo de la flecha hacia la derecha situado en el menú del IDE. Con esto se compila y se carga el dispositivo en la placa.
3) Esperamos a que termine el proceso y tendremos el dispositivo listo para su uso.

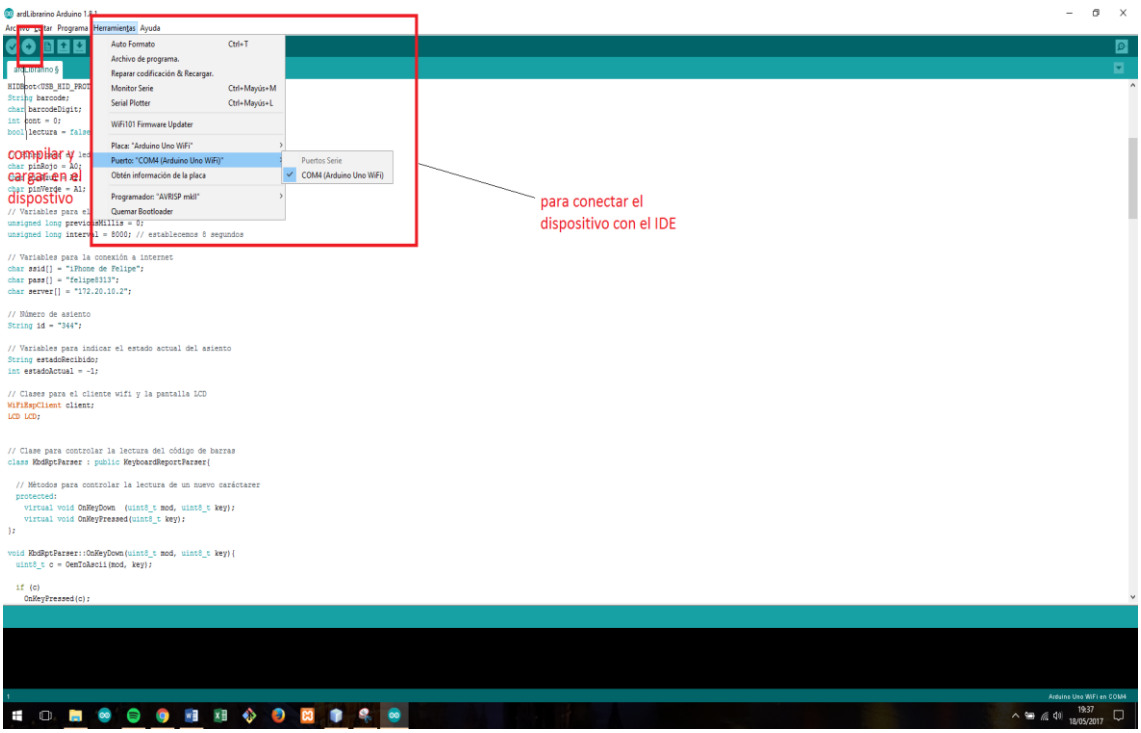

Figura 68: Instrucciones de conexión y carga del código.

B.3 | Netbeans IDE 8.0.2

Este IDE (*Integrated Development Environment*) ha sido utilizado para desarrollar el código completo de las aplicaciones. Aunque se ha usado la versión 8.0.2, se puede usar sin ningún problema una más actual. Para descargarlo es necesario ir a su página oficial y descargar la versión de Netbeans [27] que incluya PHP, por tanto, podremos descargar la versión completa o simplemente la versión con PHP.

En la parte superior seleccionamos el idioma y el sistema operativo y pulsamos 'Download'.

Figura 69: Página de descarga de Netbeans.

Una vez descargado simplemente se deben seguir los pasos del instalador y tendremos el entorno de programación preparado.

Para importar la aplicación se siguen los siguientes pasos:

1) Abrir *Netbeans*.
2) En la barra de herramientas pulsamos *Archivo* > *Abrir proyecto.*
3) Buscamos la aplicación en el CD en el directorio *Código Fuente* > *Aplicación web.*
4) Seleccionamos 'librarinoApp' y pulsamos 'Abrir proyecto'.

B.4 | Autoinstalable XAMPP

La forma más rápida y sencilla de instalar en un equipo o servidor PHP, MySQL y Apache de una sola vez es mediante un autoinstalador. Podemos encontrar varios en la web, pero en este caso se ha utilizado XAMPP. Para descargarlo nos dirigimos a su web oficial [54] y descargamos la versión correspondiente a nuestro sistema operativo.

Nada más comenzar la instalación se debe indicar las funcionalidades que debemos instalar (Figura 70). Obviamente es necesario instalar como mínimo PHP, Apache y MySQL, aunque en este caso hay que instalar también phpMyAdmin para poder acceder a la base de datos fácilmente.

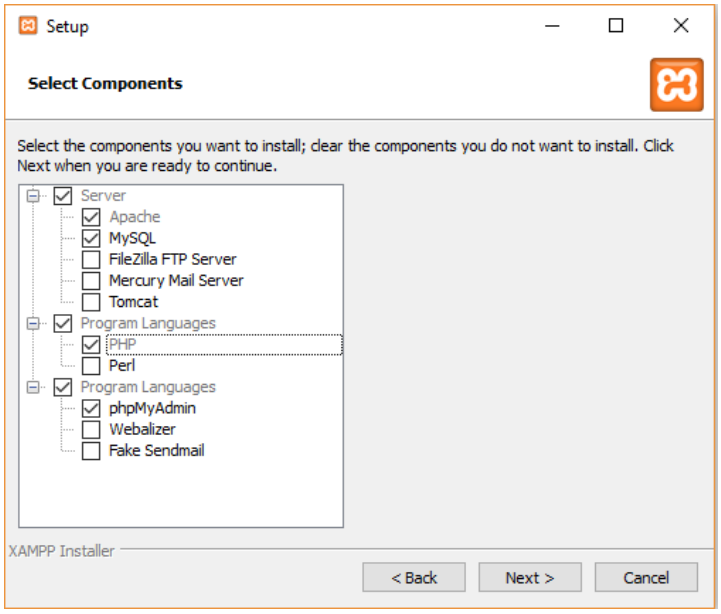

Figura 70: Módulos a instalar en XAMPP.

Para completar el proceso se deben seguir los pasos del instalador.

B.5 | Configuración de la base de datos

El método para disponer de la base de datos será mediante su importación utilizando phpMyAdmin. El archivo que debemos importar se encuentra en la carpeta 'Base de datos' y contiene tanto las tablas como los datos.

1) Abrir en panel de control de XAMPP.
2) Iniciar el servidor web y el de base de datos pulsando los botones 'Start' de cada uno de ellos.
3) Para acceder a phpMyAdmin pulsamos el botón 'Admin' del módulo MySQL.
4) En el menú superior de la página pulsamos 'Importar'.
5) Pulsar 'Seleccionar archivo' y buscamos el fichero de importación de la base de datos en el directorio nombrado anteriormente.
6) Pulsar 'Continuar'.

En las Figuras 71 y 72 se ilustra este proceso.

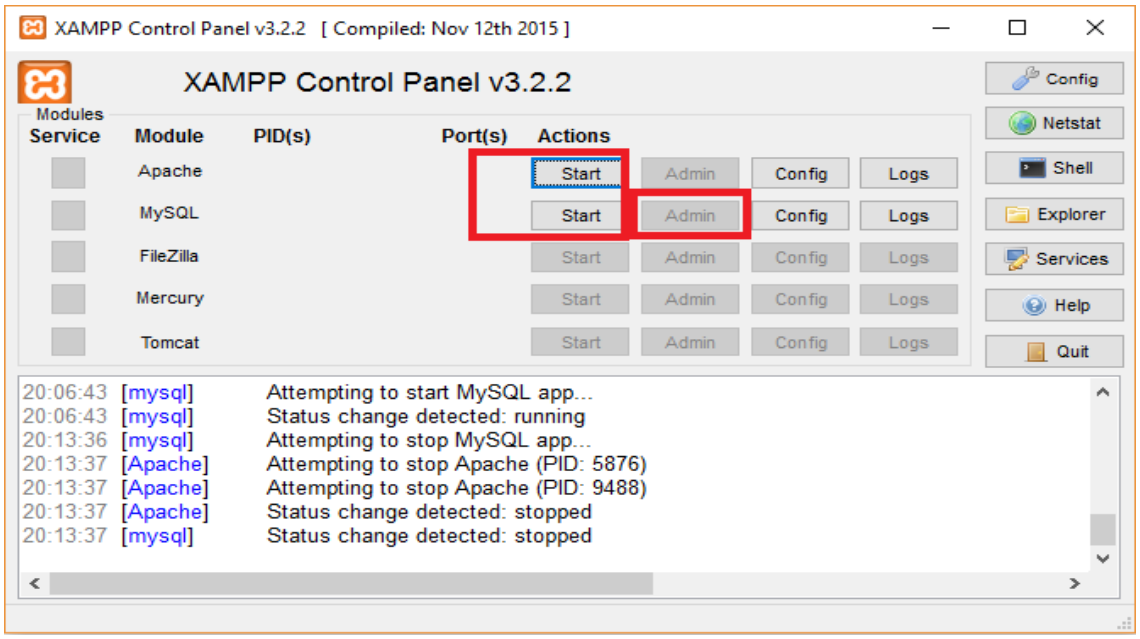

Figura 71: Panel de control de XAMPP.

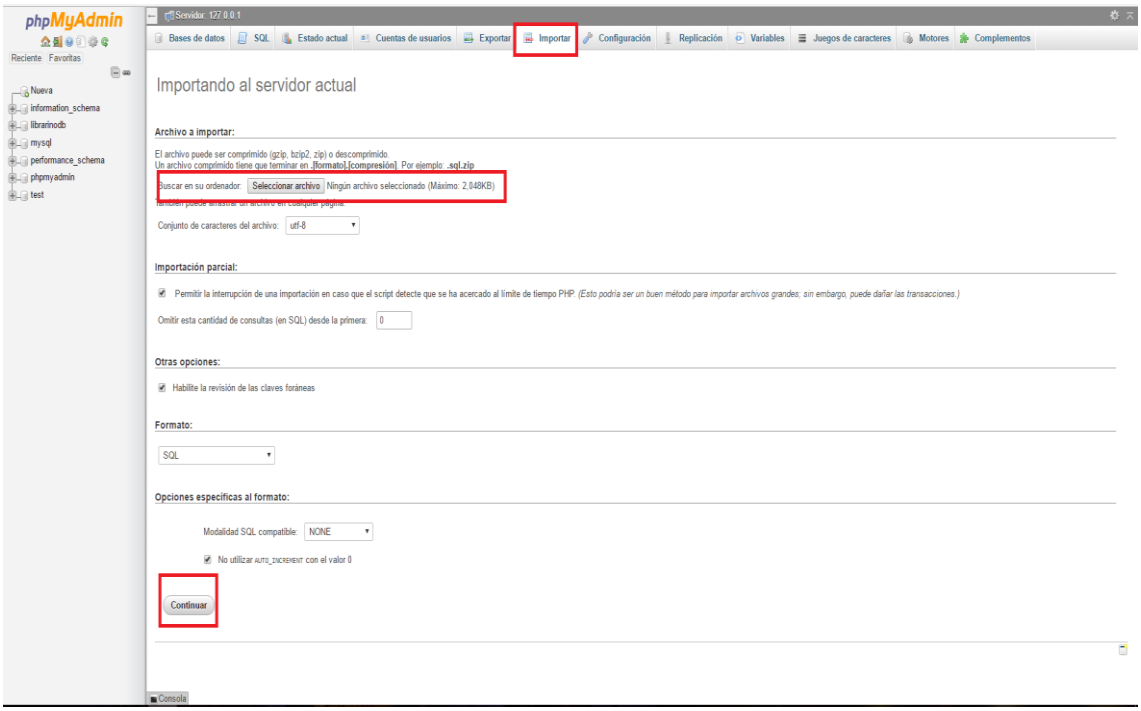

Figura 72: Importación de la base de datos.

Ya tenemos la base de datos de Librarino ejecutándose en el servidor MySQL. Lo último que queda es activar el planificador de eventos. Los eventos se utilizan en el sistema para liberar el asiento a la hora de hacer una reserva o a las 3 horas de hacer una ocupación y definen la hora exacta en que debe suceder dicha liberación.

1) En el menú lateral seleccionamos la base de datos 'librarinodb'.
2) Pulsar 'Eventos' en el menú superior.
3) Activar el planificador pulsando en el deslizador del apartado 'Estado del planificador de eventos' (Figuras 73).

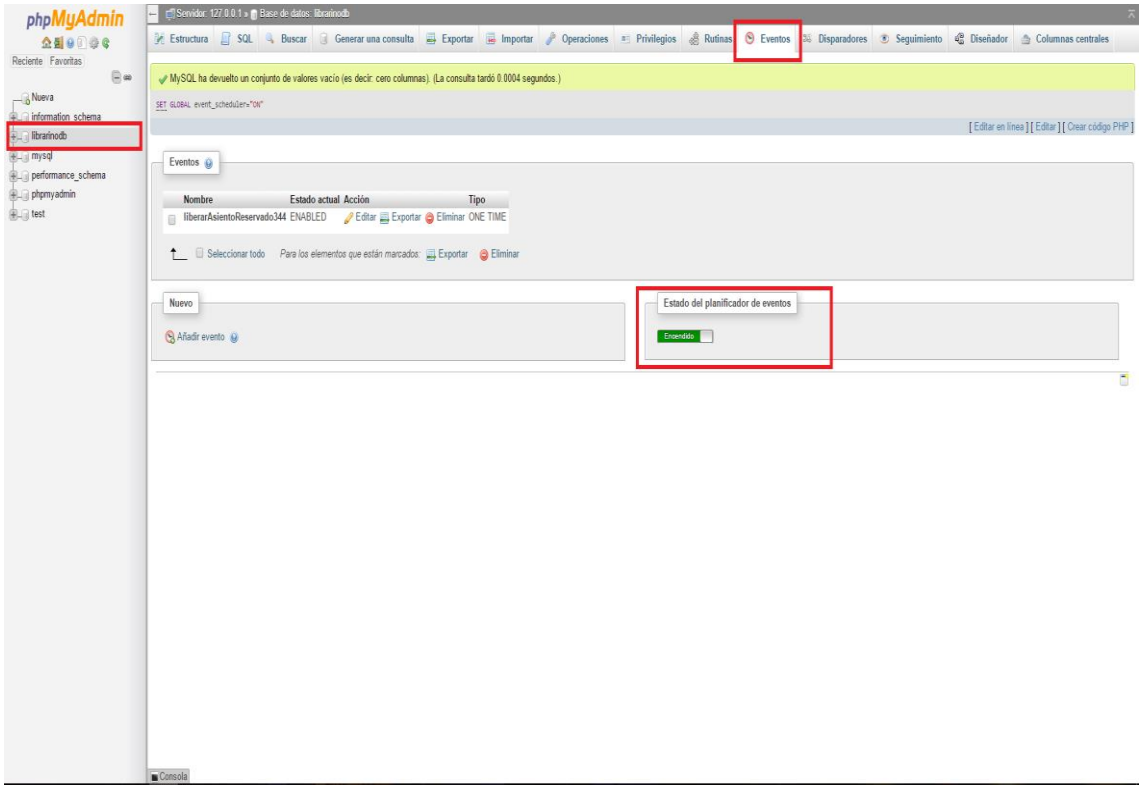

Figura 73: Activación planificación eventos.

B.6 | Despliegue de la aplicación

El despliegue de la aplicación en el servidor web es bastante sencillo pues requiere de hacer únicamente un 'copy-paste' del código de la aplicación en el directorio *C:/xampp/htdocs* (Figura 74).

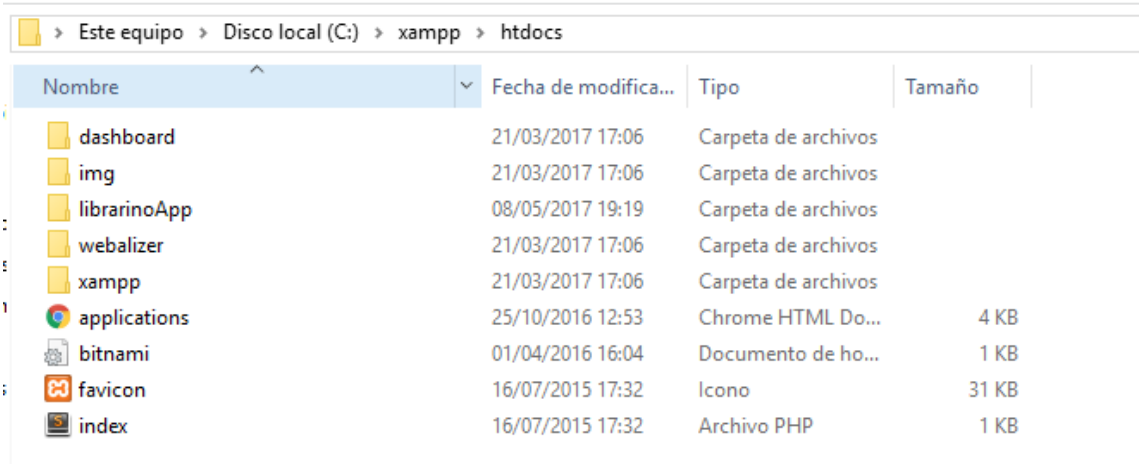

Figura 74: Contenido del directorio principal del servidor web.

Las tecnologías utilizadas son compatibles con cualquier navegador web por lo que se puede acceder a la aplicación mediante cualquiera de ellos.

- *http://localhost/librarinoApp* para la aplicación principal.
- *http://localhost/librarinoApp/admin* para la aplicación de administración.

BIBLIOGRAFÍA

[1] (2015) ¿Qué es el internet de las cosas?, de Revista “Muy Interesante”

Sitio web: http://www.muyinteresante.es/curiosidades/preguntas-respuestas/ique-es-el-qinternet-de-las-cosasq

[2] (2014) ¿Qué es y cómo funciona el Internet de las cosas?, de Hipertextual

Sitio web: https://hipertextual.com/archivo/2014/10/internet-cosas

[3] (2016) Internet de las cosas, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/Internet_de_las_cosas

[4] (2017) Arduino, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/Arduino

[5] (2014) Hardware para novatos (VII): Arduino ¿qué es y cómo funciona?, de Hipertextual

Sitio web: https://hipertextual.com/archivo/2014/03/hardware-novatos-arduino/

[6] (2016) ¿Qué es Arduino?, de Blog *Aprendiendo Arduino*

Sitio web: https://aprendiendoarduino.wordpress.com/2016/09/25/que-es-arduino/

[7] José Ignacio Peláez (2014) Tema 3: Modelado de procesos. En apuntes de la asignatura Introducción a la Ingeniería del Software.

[8] (2017) Arduino UNO, de Bricogeek

Sitio web: http://tienda.bricogeek.com/arduino/305-arduino-uno.html

[9] (2017) Módulo Wifi ESP8266, de Planeta electrónico

Sitio web: https://www.planetaelectronico.com/modulo-arduino-wifi-esp8266-esp-01-p-18348.html

[10] (2017) Arduino USB Host Shield, de Bricogeek

Sitio web: http://tienda.bricogeek.com/arduino/288-arduino-usb-host-shield.html

[11] (2017) Lector de código de barras UNOTEC, de PC Componentes

Sitio web: https://www.pccomponentes.com/unotec-lector-de-c-digos-de-barras-usb-negro

[12] (2017) Pantalla LCD Serial, de Bricogeek

Sitio web: http://tienda.bricogeek.com/pantallas-lcd/334-pantalla-serial-lcd-128x64.html

[13] (2017) Diodo LED tricolor RGB, de Bricogeek

Sitio web: http://tienda.bricogeek.com/componentes/62-diodo-led-tricolor-rgb-5mm.html

[14] (2017) Cables para Arduino, de Planeta electrónico

Sitio web: https://www.planetaelectronico.com/cable-dupont-arduino-macho-macho-15cm-pack-10-unidades-p-17368.html

[15] (2017) Resistencia 220 ohmios, de Planeta electrónico

Sitio web: https://www.planetaelectronico.com/resistencia-carbon-220r-16w-p-11558.html

[16] (2017) Protoboard de 400 contactos, de PC Componentes

Sitio web: https://www.pccomponentes.com/placa-de-conexiones-400-contactos

[17] (2017) Página oficial de PHP

Sitio web: http://php.net/manual/es/intro-whatis.php

[18] (2017) PHP, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/PHP

[19] (2017) ¿Qué es PHP?, de desarrolloweb.com

Sitio web: https://www.desarrolloweb.com/articulos/392.php

[20] (2011) ¿Qué es Apache?, de Culturacion

Sitio web: http://culturacion.com/que-es-apache/

[21] (2012) ¿Qué hace un servidor web como Apache?, de Digital Learning

Sitio web: http://www.digitallearning.es/blog/apache-servidor-web-configuracion-apache2-conf/

[22] (2017) MySQL, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/MySQL

[23] (2015) ¿Qué es y para qué sirve MySQL?, de Culturacion

Sitio web: http://culturacion.com/que-es-y-para-que-sirve-mysql/

[24] (2016) Patente *Library self-service seat selection system*, de Espacenet

Sitio web: https://worldwide.espacenet.com/publicationDetails/biblio?II=6&ND=3&adjacent=true&locale=en_EP&FT=D&date=20160203&CC=CN&NR=105303381A&KC=A

[25] (2015) Patente *Library seat managment system*, de Espacenet

Sitio web: https://worldwide.espacenet.com/publicationDetails/biblio?II=13&ND=3&adjacent=true&locale=en_EP&FT=D&date=20150701&CC=CN&NR=104751378A&KC=A

[26] (2015) Patente *Library seat managment system*, de Espacenet

Sitio web: https://worldwide.espacenet.com/publicationDetails/biblio?II=17&ND=3&adjacent=true&locale=en_EP&FT=D&date=20150513&CC=CN&NR=204331793U&KC=U

[27] (2017) Netbeans IDE 8.0.2, de Netbeans

Sitio web: https://netbeans.org/downloads

[28] (2017) Arduino IDE 1.8.1, de Arduino

Sitio web: http://www.arduino.org/previous-releases

[29] (2012) HTML, de Definicion.de

Sitio web: http://definicion.de/html/

[30] (2017) ¿Qué es HTML?, de Acerca de HTML

Sitio web: http://www.acercadehtml.com/manual-html/que-es-html.html

[31] (2017) ¿Qué es CSS?, de Librosweb

Sitio web: http://librosweb.es/libro/css/capitulo_1.html

[32] (2017) CSS, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada

[33] (2015) ¿Qué es Bootstrap?, de Railoa Networks

Sitio web: https://raiolanetworks.es/blog/que-es-bootstrap/

[34] (2016) ¿Qué es Bootstrap y cúales son sus ventajas?, de Puntoabierto

Sitio web: http://puntoabierto.net/blog/que-es-bootstrap-y-cuales-son-sus-ventajas

[35] (2017) JavaScript, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/JavaScript

[36] (2007) ¿Qué es JavaScript?, de Maestros del web

Sitio web: http://www.maestrosdelweb.com/que-es-javascript/

[37] (2017) jQuery, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/JQuery

[38] (2013) jQuery: Qué es, Orígenes, Ventajas y Desventajas, de Capacity

Sitio web: http://blog.capacityacademy.com/2013/03/16/jquery-que-es-origenes-ventajas-desventajas/

[39] (2012) ¿Qué es AJAX?, de Digital Learning

Sitio web: http://www.digitallearning.es/blog/que-es-ajax/

[40] (2017) AJAX, Capítulo 1. Introducción a AJAX, de Librosweb

Sitio web: http://librosweb.es/libro/ajax/capitulo_1.html

[41] (2014) ¿Qué es y para qué sirve JSON?, de GeekyTheory

Sitio web: https://geekytheory.com/json-i-que-es-y-para-que-sirve-json/

[42] (2017) JSON, de Wikipedia

Sitio web: https://es.wikipedia.org/wiki/JSON

[43] (2017) Repositorio GitHub de la librería para el Shield USB, de GitHub

Sitio web: https://github.com/felis/USB_Host_Shield_2.0

[44] (2014) Ejemplo práctico de conexión y configuración de un lector de código de barras con Arduino, de Electroingenio

Sitio web: http://www.electroingenio.com/arduino/bar-code-scanner-arduino-usb-shield/

[45] (2017) Repositorio GitHub de la librería para el módulo Wifi, de GitHub

Sitio web: https://github.com/bportaluri/WiFiEsp

[46] (2015) Repositorio GitHub de la librería para la pantalla LCD, de GitHub

Sitio web:https://github.com/Haven-Lau/Arduino-Libraries/tree/master/SerialGraphicLCD

[47] (2017) Plugin jQuery *Datatables*, de Datatables

Sitio web: https://datatables.net/

[48] (2017) Plugin jQuery *Select2*, de GitHub

Sitio web: https://select2.github.io/

[49] (2017) Plugin jQuery *Datepicker*, de jQuery UI

Sitio web: https://jqueryui.com/datepicker/

[50] (2017) Plugin jQuery *Chart.js*, de Chart.js

Sitio web: http://www.chartjs.org/

[51] (2017) Repositorio GitHub de la librería PHP *PHPMailer*, de GitHub

Sitio web: https://github.com/PHPMailer/PHPMailer

[52] (2009) Ejemplo de generación de una contraseña aleatoria alfanumérica en PHP, de Blog de Álvaro Pita

Sitio web: http://www.alvaropita.es/generar-contrasena-aleatoria-en-php/

[53] (2017) Ejemplo de texto que parpadea con CSS, de Codepen

Sitio web: http://codepen.io/sekane81/pen/JKuwb

[54] (2017) Página oficial de XAMPP, de Apache Friends

Sitio web: https://www.apachefriends.org/es/index.html