

Documentación de

CLIMANET

TRABAJO FIN DE GRADO

DESARROLLO DE APLICACIONES WEB

Autor: Marc Fortes Domínguez

Tutor de proyecto: Ismael Tracastro

Curso: 2019/2020

Contenido

1. Introducción.....	3
2. Objetivo.....	3
2.1. Planificación.....	3
2.2. Mockups.....	5
3. Solución tecnológica.....	6
3.1. Componentes físicos.....	6
3.1.1. Elegoo UNO R3.....	7
3.1.2. Ethernet Shield W5100.....	7
3.1.3. Sensor DHT11.....	8
3.1.4. Pantalla LCD1602A.....	9
3.1.5. Otros componentes.....	10
3.2. Lenguajes.....	11
3.2.1. HTML.....	11
3.2.2. Bootstrap / CSS.....	11
3.2.3. PHP.....	12
3.2.4. Arduino / C++.....	12
3.3. Entornos de desarrollos.....	13
3.3.1. Arduino IDE.....	13
3.3.2. Atom.....	13
3.4. Bases de datos.....	14
3.4.1. MySQL.....	14
3.4.2. PHPMyAdmin.....	14
4. Desarrollo del proyecto.....	15
4.1. Diagramas de clases.....	15
4.2. Montaje físico.....	16
4.3. Codificación.....	17
4.3.1. Código perteneciente al dispositivo físico.....	17
4.3.2. Código perteneciente a la aplicación web.....	18
5. Evaluación final.....	20
6. Conclusiones.....	22
7. Bibliografía.....	22
8. Código adjuntado.....	23

1. Introducción

Muchas empresas poseen habitaciones o espacios interiores, como cámaras frigoríficas o centros de procesamiento de datos, que necesitan llevar un control preciso de las temperaturas y / o humedades que hay en su interior. Además, se añade la necesidad de poder visualizar estos registros de una manera accesible e intuitiva, de manera que cualquier persona pueda llegar a ellos sin la presentación de obstáculos y dificultades.

Por las necesidades anteriormente mencionadas, surge la idea de este proyecto, *Climanet*. Climanet no es solo un producto, que permite obtener valores mediante componentes físicos, sino que también es un servicio, gracias a su *Web App*, que permite acceder a todos los registros de temperaturas y humedades pudiendo filtrar de diferentes maneras, y siempre de manera sencilla.

2. Objetivo

La creación de Climanet, a parte de satisfacer las necesidades de otras personas, también satisface las mías. Esto es debido a que a lo largo de su creación me permite poder desarrollar todas las habilidades y técnicas fundamentales. Al fin y al cabo, creo que ese es el objetivo de este proyecto, crear algo donde se vean expuestas todas las habilidades adquiridas durante la creación del mismo, y a lo largo del curso.

2.1. Planificación inicial

Antes de empezar a crear, es muy importante tener una visión a largo plazo de los pasos que son convenientes seguir. Es normal que con el paso de las semanas, estos pasos se vean algo modificados, ya que siempre pueden presentarse imprevistos que no hemos tenido en cuenta.

He querido dividir la planificación en tres bloques principales. El primero, la **fase de investigación**. Es el período que he creído necesario para formar la estructura lógica del proyecto al mismo tiempo que empiezo a juntar información y documentación. El segundo bloque, la **puesta en marcha**. Esta fase, se compone de la selección y compra de material, montaje del producto físico una vez obtenidos los materiales, y el desarrollo de la aplicación web. Este segundo bloque se entrelaza con la primera fase, ya que uso el tiempo de espera de la compra de materiales para ir formándome. El tercer y último bloque, **proceso de documentación**. A pesar de que a lo largo del proyecto se van tomando algunas anotaciones, he creído conveniente dejar esta fase como la última. De esta manera, se pueden reflejar de una mejor manera, todos los cambios que se hayan presentado a lo largo del proyecto.

A continuación, dejo el diagrama de *Gantt* inicial de cómo se plantearon las fases de desarrollo:



2.2. Mockups

Tener unos diseños iniciales de cómo se quiere que se vea la aplicación web es muy importante antes de ponerse a escribir cualquier línea de código. Esto facilitará mucho su desarrollo. Como los diseños están hechos para observarlos y tener una idea clara de un solo vistazo, aquí dejo algunas de mis ideas iniciales:

Página inicial

Climanet	Login	Signup
<div> <div>Descripción</div> <div> <div>Crear cuenta</div> <div><input type="text"/></div> <div><input type="text"/></div> <div>Button</div> </div> </div>		

Página Signup

Climanet	Login	Signup
<div> <div>Crear cuenta</div> <div><input type="text" value="email"/></div> <div><input type="text" value="password"/></div> <div>Button</div> </div>		

Página Login

Climanet

Login Signup

Login

email

password

Button


Página home (sesión iniciada)

Climanet

Logout

Filtros

Options ▼

4/2  ▼

Button

Result Set

3. Solución tecnológica

A continuación haré un despliegue de todo aquello que resulta necesario para poder llevar a cabo el proyecto. He decidido dividir y explicar sus partes en **hardware**, **lenguajes**, **entornos de desarrollo** y **base de datos**.

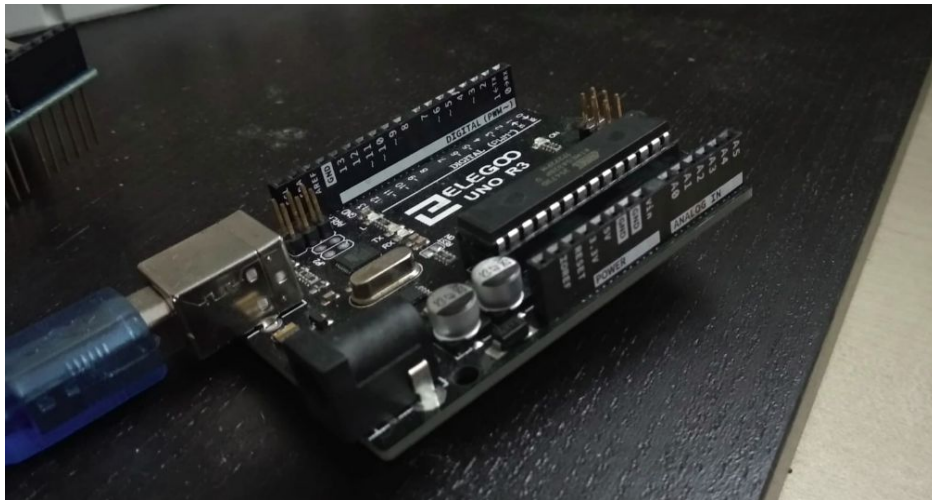
3.1. Componentes físicos

En este apartado hablaré de todos los componentes físicos que han sido necesarios, desde el dispositivo que, a grandes rasgos puede ser el más importante, hasta el que pueda ser menos. A la hora de la verdad, todos los componentes tienen la misma importancia, ya que todos se complementan y se hacen falta los unos a los otros para el correcto funcionamiento del proyecto.

La parte física del proyecto es una de las partes más interesantes de este proyecto, ya que quizás, sin ser uno de los puntos más fuerte de este curso, me ha servido mucho para aprender los conceptos básicos de la electrónica.

3.1.1. Elegoo UNO R3

Abriré este apartado físico hablando de lo que a mí me gusta llamar el centro de operaciones del proyecto. Estoy hablando de **Elegoo UNO R3**. Esta es la placa electrónica que he escogido, que me permitirá cargar los programas al resto de componentes que estén conectados a ésta. Podría haber escogido otros modelos, pero por el precio que tenía y referencias de otras personas, me parecía una opción muy buena.



Esta placa tiene **14 pines digitales de entrada / salida y 6 pines analógicos**. Estos 20 pines en total, son programables.

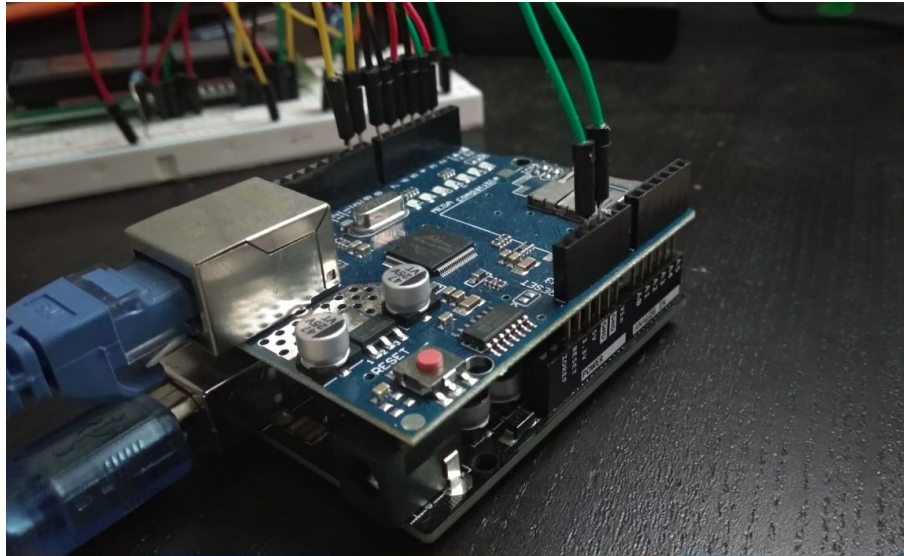
La placa puede ser alimentada por un cable USB conectado directamente a un puerto USB de nuestro ordenador, que ofrecerá unos ~5V (siempre y cuando no necesitemos de más tensión). Si necesitáramos una tensión más alta, podríamos alimentar la placa una batería externa de entre 7-20V.

3.1.2. Ethernet Shield W5100

Otro gran pilar de los componentes físicos de este proyecto es **Ethernet Shield**. Éste módulo permite conectar nuestra placa Elegoo UNO R3 a la red sin presentar dificultades. Esto es gracias al chip de ethernet **Wiznet W5100**, que proporciona una IP, que hace posible el uso de TCP y UDP.

La utilidad que le sacaré a este dispositivo será la de mensajero entre el sensor DHT11 (hablaré en los siguientes apartados de él), que será el encargado de recibir los valores, y mi Web App, que será la

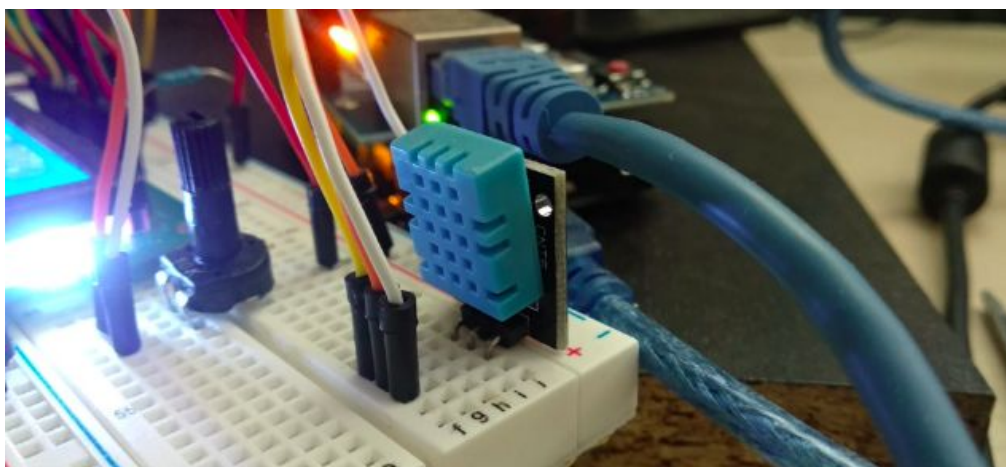
encargada de imprimirlos. Para ello, Elegoo UNO R3 estará conectada a la red, haciendo posible su visibilidad con el resto de dispositivos.



El módulo tiene una **conexión RJ-45 estándar**, que permitirá conectarlo a la red. Elegoo UNO R3 se comunicará con el chip W5100 mediante el bus SPI usando los **pines digitales 10, 11, 12 y 13**. En cuanto a la alimentación, la conectamos a los 5V de Elegoo UNO R3, y a la toma de tierra, GND.

3.1.3. Sensor DHT11

Este sensor es una de las partes más importantes debido a que va a ser el encargado de estar recibiendo los valores de temperatura y humedad del ambiente interior. Estos valores, serán los que posteriormente guardaremos en la base de datos, para poder trabajar con ellos.



El dispositivo está recubierto de plástico con unas ranuras que permite la entrada y circulación de aire donde se encuentran los propios sensores de temperatura y humedad. Mediante uno de sus tres pines podremos obtener los valores de temperatura en **grados centígrados (símbolo °C)** y **la humedad relativa**.

DHT11 - Características	
Temperatura	0 a 50°C +- 2°C
Humedad	20 a 80% +- 5%
Ciclo de muestras	1 Hz
Voltaje	3 a 5.5 V

El modelo DHT11 que empleo es con PCB, esto quiere decir que, **el sensor viene integrado dentro de un PCB** (Printed Circuit Board) que viene con una resistencia integrada. El modelo que yo he utilizado dispone de 3 pines, que son los siguientes:

- **VCC:** Alimentación (Pin izquierdo)
- **DATA:** Transmisión de datos (Pin medio)
- **GND:** Conexión a tierra (Pin derecho)

3.1.4. Pantalla LCD1602A

Otro de los componentes del grupo físico de este proyecto, es el módulo de pantalla LCD 1602A, de **16 columnas y 2 líneas**, para mostrar información de forma directa desde Arduino. Esta pantalla me servirá para imprimir los valores capturados por el sensor de temperatura, y poder así, visualizarlos sin tener la necesidad de acceder a la aplicación web.



El módulo, en su parte superior, **tiene 16 pines**, en el que cada uno de ellos corresponden a una conexión. A continuación, dejo una tabla que he realizado con las conexiones correspondientes:

Numeración de terminales / pines	Designación asociada	Significado
1	VSS	GND
2	VDD	5V
3	VO	Contraste
4	RS	Registro
5	R/W	Lectura / Escritura
6	E	Habilitado
7	D0	Datos 0
8	D1	Datos 1
9	D2	Datos 2
10	D3	Datos 3
11	D4	Datos 4
12	D5	Datos 5
13	D6	Datos 6
14	D7	Datos 7
15	A	Ánodo (Resistencia 220k necesaria)
16	K	Cátodo

3.1.5. Otros componentes

En los puntos anteriores he estado hablando sobre los componentes principales que definen este proyecto, pero la verdad es, que no hay que dejar de lado el resto de componentes que pueden pasar más desapercibidos, pero son de igual importancia. Estos componentes son los siguientes:

- **Cable Ethernet:** El cable que yo he usado es un ethernet standard con un conector RJ45. Éste me servirá para conectar el módulo Ethernet Shield al Router.

- **Potenciómetro:** El uso de este componente es necesario para poder regular el contraste de mi pantalla LCD, de tal manera que me permita regular el brillo.
- **LED:** Este componente puede decirse que no es del todo imprescindible, pero es una muy buena manera de hacer saber al usuario en qué momentos se están realizando conexiones, ya que cuando esto ocurra, el LED se encenderá y apagará.
- **Resistencias:** En todo el circuito, solo haré uso de 2 resistencias, pero que son altamente necesarias. Haré uso de dos resistencias de 220k, una para un LED, y otra para la pantalla LCD.
- **Cables:** Por último, solo me queda hablar de los cables, algo muy obvio. Estos cables me permitirán establecer las conexiones necesarias entre los componentes partícipes del circuitos.

3.2. Lenguajes

Los lenguajes son los que proporcionan la **capacidad de escribir una serie de instrucciones** (código) con el fin de controlar el comportamiento físico y/o lógico (enviar órdenes) de un ordenador. Los lenguajes que he seleccionado, son los que bajo mi criterio, mejor se adaptan a los requisitos de mi proyecto.

3.2.1. HTML

Utilizo **HTML** (*HyperText Markup Language*), un lenguaje muy simple y general, para elaborar la estructura de mi Web App. Este lenguaje de marcado sirve para definir la estructura básica y un código para la definición de contenido de una página web, como texto, imágenes, vídeos, etc.

Para definirlo de alguna manera más coloquial, me gusta compararlo con un esqueleto que se encarga de aguantar y ofrecer soporte al resto de organismo, que vendrían siendo el resto de componentes de la Web App.

3.2.2. Bootstrap / CSS

Para dar un diseño más estético a mi aplicación web he decidido utilizar el *framework* **Bootstrap**. Esta biblioteca de código abierto me permite hacer uso de tipografías, formularios, botones, menús de

navegación, etc. Esto último hace que en una proporción de resultados - tiempo invertido, sea bastante aceptable, ya que la parte visual no es el punto fuerte de este proyecto.

Los elementos de diseños están basados en HTML y CSS, así como extensiones de JavaScript adiciones.

Aún siendo Bootstrap la base del diseño de la web, también he usado CSS puro para dar estilos propios a algunas partes.

3.2.3. PHP

El lenguaje seleccionado para encargarse de realizar el trabajo de gestión de la base de datos es **PHP** (**PHP: Hypertext Preprocessor**). Este lenguaje me permite hacer de mi web app una **página dinámica**, ya que su contenido se irá actualizando y no siempre será el mismo. PHP se procesa desde el lado del servidor, por lo que éste recibe los datos de la solicitud, realiza su trabajo (consultar a una base datos, por ejemplo), y por último contesta con una página web estática, que ha sido creada de manera dinámica.

3.2.4. Arduino / C++

Arduino está basado en C y soporta todas las funciones del estándar C y algunas de C++. Ésto último posible que muchos de los comandos estándar de los lenguajes mencionados, sean utilizables.

La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contiene declaraciones, estamentos o instrucciones.

- **setup()**: Esta primera función, **es la primera en ejecutarse** en un programa Arduino. Aquí **se establecen funciones y criterios** que queramos que nuestro microcontrolador ejecute **de manera única**.
- **loop()**: Dentro de esta otra función de alta importancia, se escribe el programa que, como su propio nombre indica, *loop* o *bucle* en español, queramos que **se repita de forma indefinida** hasta que se apague o se reinicie el microcontrolador.

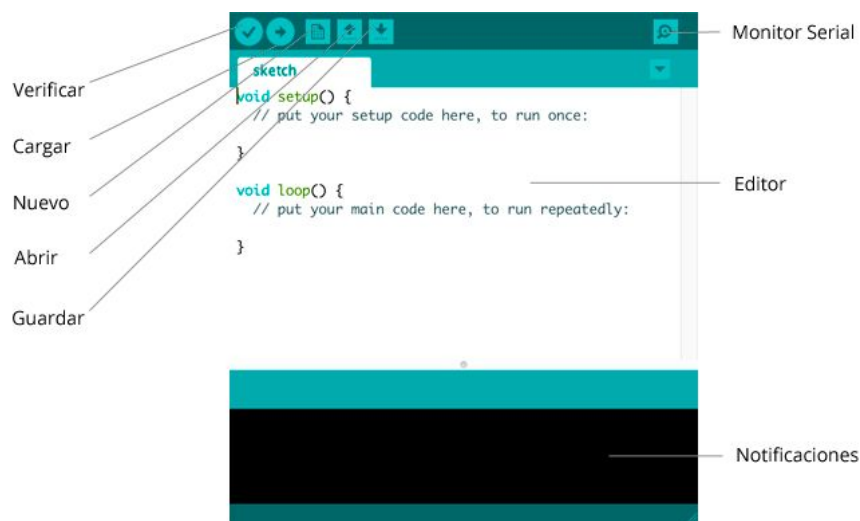
Esta última parte de estructura de la que he estado hablando, es uno de los tres pilares del lenguaje de programación Arduino. Arduino, como ellos mismos dicen (mencionado en la bibliografía), se dividen en **tres partes: funciones, valores (variables y constantes), y estructura**.

3.3. Entornos de desarrollo

Quizás esta sea una de las partes que pase más desapercibida, pero en mi opinión no es así. El editor de código será el lugar donde se pasa prácticamente la mayoría de tiempo escribiendo código. Si el editor que se usa no se ajusta a tus necesidades, tienes un problema. Por eso a continuación, hablaré de los programas que he utilizado para escribir mi código.

3.3.1. Arduino IDE

Existen varias opciones para escribir el código que será leído por la placa Elegoo UNO R3, pero yo me he quedado con la siguiente: **Arduino IDE**. Este entorno de desarrollo de la compañía Arduino, es totalmente compatible Elegoo. Desde este entorno de desarrollo se realiza la escritura y carga de programas en la placa. El IDE de Arduino admite los lenguajes C y C++ utilizando algunas reglas especiales de estructuración de códigos.

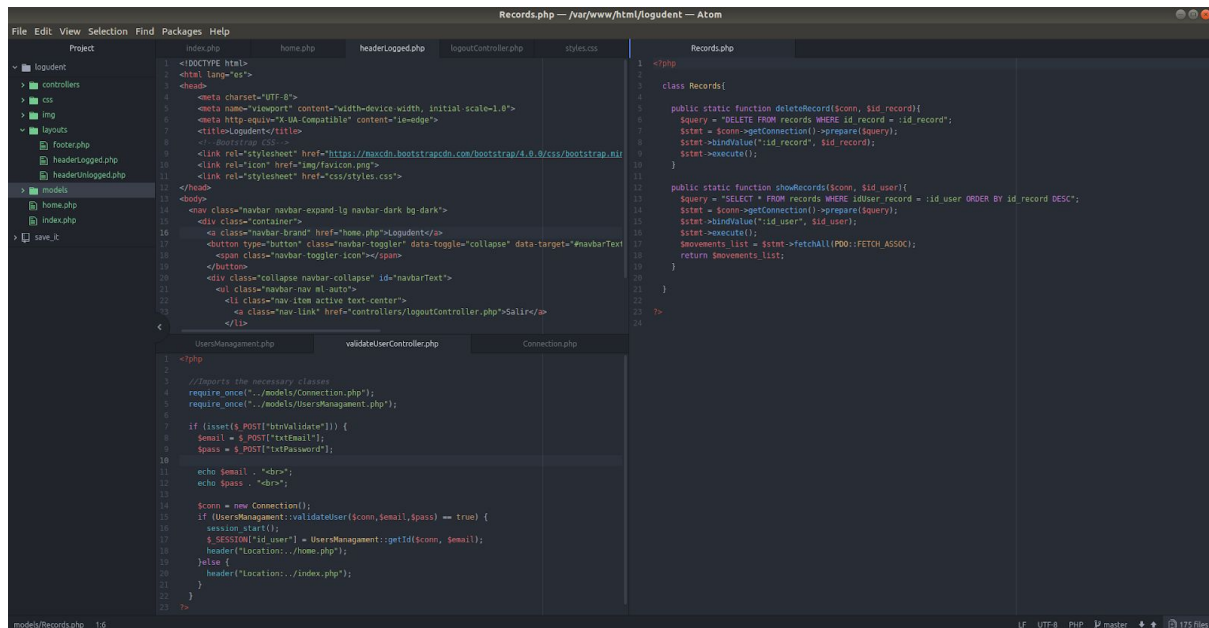


Una de las grandes ventajas que ofrece este IDE, es la suministración de una biblioteca de *software*, que proporciona muchos procedimientos comunes de E/S (Entrada / Salida).

3.3.2. Atom

Bajo experiencia propia, me he decantado por seguir trabajando con **Atom**. Este es el editor de código que llevo utilizando desde que empecé en este mundillo de la programación. Esto último que he mencionado, no me ha privado de probar otros editores como *Visual Studio Code*, *Brackets*, *Sublime Text 2* y alguno más que no recuerdo. Si me he decantado por Atom es por las necesidades que me

cubren, que son las justas. Por eso, no me merece la pena usar otros potentes editores cuando no le saco partido a la mayoría de sus funciones.



Desde Atom, he escrito todo el código requerido para la aplicación web, en sus diferentes lenguajes (HTML, Bootstrap / CSS, JavaScript y PHP).

3.4. Bases de datos

Y para finalizar la sección de *solución tecnológica*, hablaré de la base de datos. Y es que aunque esta parte sea la última, no es la menos importante.

3.4.1. MySQL

Debido a que el proyecto necesita almacenar datos, haré uso del administrador de base de datos **MySQL**. Este gestor de código abierto, es más que potente para satisfacer las necesidades del proyecto. MySQL será el encargado de ofrecer la posibilidad de almacenar, recuperar, modificar y administrar una base de datos utilizando **SQL**.

3.4.2. PHPMysqlAdmin

PHPMyAdmin es una herramienta que permite manejar la administración de MySQL de una manera mucho más cómoda, utilizando un navegador web. Todas aquellas funciones que permite MySQL, mencionadas en el punto anterior, las podremos realizar desde PHPMyAdmin haciendo uso de su interfaz.

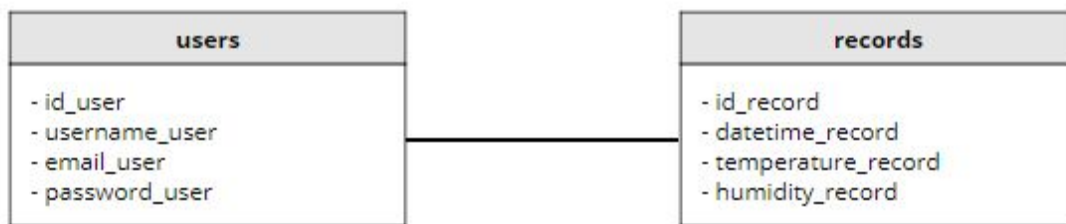
Los administradores MySQL y PHPMyAdmin se instalan en la misma máquina, aunque de manera independiente. Esto último quiere decir que, no haré uso de ningún tipo de paquete como *XAMPP* o *LAMPP*, sino que instalaré cada gestor independientemente.

4. Desarrollo del proyecto

Como había mencionado en el apartado de la planificación, esta es la parte en la que empieza la segunda fase, la **puesta en marcha**. Cuando ya he obtenido todos los materiales necesarios y creo que estoy lo suficientemente informado y documentado, es el momento de proceder al diseño lógico de la base de datos y la aplicación, junto a la construcción de las mismas junto al dispositivo físico.

4.1. Diagramas de clases

Al igual que la planificación, estructurar un buen diagrama de clases antes de proceder a la construcción es de alta importancia. Con esto, conseguiremos analizar eficaz y eficientemente la mejor manera de construir nuestra aplicación junto a la base de datos.



La base de datos se compondrá de dos tablas. Una de ellas para todos los usuarios registrados en la aplicación web, y la otra tabla para guardar todos los registros de temperatura y humedad que se hayan producido.

En la tabla **usuario** se solicitarán los siguientes atributos:

- **id_user** (int): Un número autoincremental único que servirá para identificar cada usuario.
- **username_user** (varchar): Nombre de usuario que indicará el usuario.
- **email_user** (varchar): Email que introducirá el usuario.
- **password_user** (varchar): Contraseña que introducirá el usuario.

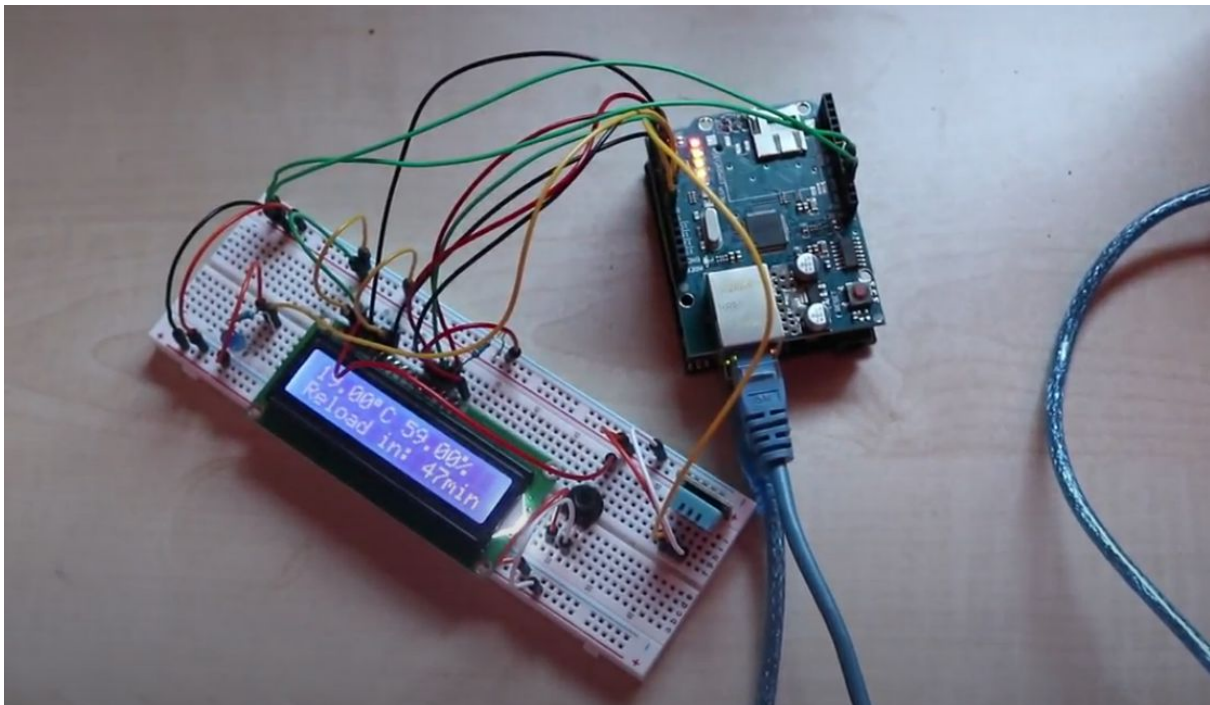
En la tabla **records** se solicitarán los siguientes atributos:

- **id_record** (int): Un número autoincremental único que servirá para identificar cada registro.

- **datetime_record** (datetime): La fecha y hora (hh:mm:ss) en la que se ha producido el registro.
- **temperature_record** (varchar): La temperatura existente durante el registro.
- **humidity_record** (varchar): La humedad existente durante el registro.

4.2. Montaje físico

Cuando me llegan las piezas necesarias para realizar el montaje del dispositivo físico, quiere decir que ha llegado el momento de ponerse manos a las obras. Al principio de este documento mencionaba que el dispositivo se basaba en cuatro componentes esenciales: Placa base Elegoo UNO R3, el módulo Ethernet Shield W5100, el sensor de temperatura y humedad DHT11 y la pantalla LCD 1602A.



Con estos componentes, lo que conseguiremos es desarrollar el siguiente flujo de trabajo:

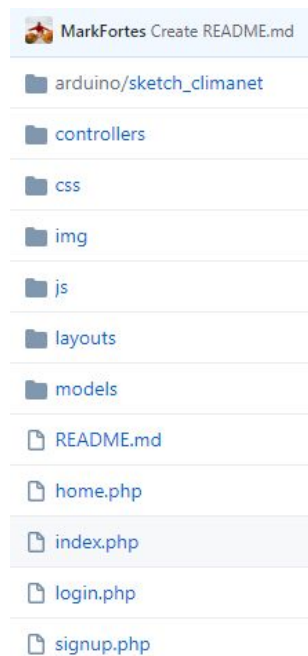
1. La placa base se conectará a una fuente de alimentación (en mi caso un puerto USB de 5V de mi ordenador portátil), de tal manera que alimentará al resto de los componentes del circuito. Además será la encargada de hacer correr el programa para que empiece a funcionar todo.
2. El sensor de temperatura y humedad DHT11 recogerá los valores existentes en el momento.
3. Los valores, anteriormente recogidos, se mostrarán gracias a la pantalla LCD1602A. Ésta mostrará el valor de la temperatura y el valor de la humedad. También mostrará un mensaje con el tiempo de espera para que se vuelva a cargar el programa.

4. Por otro lado, el módulo Ethernet Shield W5100 conectado al router mediante un cable RJ-45, también recogerá los valores y los enviará a la base de datos, estableciendo una conexión como cliente con mi servidor (un equipo dentro de mi red privada) y enviándolos a través del protocolo HTTP/1.0.

4.3. Codificación

A continuación, veremos qué metodologías y técnicas se han utilizado para el desarrollo de estos dos apartados.

Todo el código que ha sido desarrollado para el funcionamiento del dispositivo físico y la construcción de la aplicación web se encuentra subido en: <https://github.com/MarkFortes/climanet>.



4.3.1. Código perteneciente al dispositivo físico

Para hacer que toda la parte del funcionamiento físico funcione de manera correcta, se ha tenido que desarrollar un código, también llamado *sketch* donde se incluyen las librerías necesarias, se declaran las variables, funciones, conexiones, etc. Este *sketch* es el que corre en la placa base Elegoo UNO R3.

De manera resumida, el *sketch* realiza las siguientes funciones:

1. **Incluir las librerías** *LiquidCrystal.h* para la pantalla LCD, *DHT.h* para el sensor de temperatura y humedad, y por último, *SPI.h* y *Ethernet.h* para el módulo Ethernet.

2. **Crear e inicializar los objetos** de cada componentes para poder hacer uso de sus correspondientes funciones.
3. **Obtener los valores de temperatura y humedad** a través del sensor DHT11.
4. Mostrar los valores a través de la pantalla LCD1602A.
5. **Establecer una comunicación Arduino - MySQL**. La placa base actuará como cliente e intentará comunicarse con el servidor dentro de mi red privada. Si la conexión se realiza sin presentar problemas, se realiza una llamada HTTP/1.0 usando el método GET, enviando los valores obtenidos como parámetros al servidor dónde habrá un fichero llamado *createRecordController* que se encargará de **registrar los valores en la base de datos**. Si la conexión falla, el programa devuelve un mensaje de error.
6. Si anteriormente se ha abierto la conexión correctamente, **se cierra la conexión cliente-servidor**.
7. Se pone en marcha la cuenta atrás para el reinicio del programa, de forma que todos los pasos anteriores se repiten uno tras otro.

4.3.2. Código perteneciente a la aplicación web

Existen varios patrones de desarrollo de software. Para la aplicación web de este proyecto, he considerado conveniente utilizar el patrón **Modelo-Vista-Controlador (MVC)**. Éste tiene la finalidad de separar los componentes para la representación de la información y la interacción del usuario. Varios *frameworks* están basados en este patrón, pero en mi caso he decidido utilizar MVC sin hacer uso de ningún *framework* para así entender y comprender su utilidad y las mejoras que puede aportar a la hora de desarrollar el código orientado a un proyecto.

Modelos

He categorizado como modelos aquellas clases en las que declaro varias propiedades y funciones que se han de utilizar de manera repetitiva en varios ficheros.

Modelos que han sido necesarios en Climamet:

- **Connection**: Permite abrir y cerrar la conexión con la base de datos MySQL.
- **Records**: Ofrece una amplio número de maneras de poder trabajar y filtrar de maneras específicas los registros guardados en la base de datos.
- **Users**: Esta clase permite crear nuevos usuarios, además de validarlos, eliminarlos, y devolver los datos de todos aquellos que ya estén registrados en la base de datos.

Controladores

La función principal de los controladores es la de hacer de intermediario de las vistas y los modelos. De esta manera, conseguimos no ensuciar el código de las vistas y así poder declarar los objetos y hacer uso de sus funciones fuera de éstas.

Controladores que han sido necesarios en Climamet:

- **createRecordController**: Se encarga de guardar nuevos registros en la base de datos. Este fichero es de alta importancia, ya que será el que utilizará el dispositivo físico para hacer los registros de los valores de temperatura y humedad.
- **createUserController**: Se encarga de llamar a la función *createUser* de la clase *User* para crear un usuario nuevo a partir de los datos introducidos en el formulario de registro de usuario.
- **logoutController**: Permite cerrar sesión eliminando ésta.
- **showRecordsController**: Se encarga de hacer una llamada, teniendo en cuenta los filtros solicitados, a la clase *Records*. Ésta devolverá todos los registros demandados, y así podremos manejarlos a nuestro antojo para poder imprimirlos en las vistas.
- **validateUserController**: Se encarga de llamar a la función *validateUser* de la clase *User* para validar el usuario a través de los datos rellenados en los campos del formulario de inicio de sesión.

Vistas

En las vistas, como su nombre indica, se desarrolla todo el código que sea correspondiente a aquello que se visualiza en la aplicación web. Podemos englobar desde la estructura y maquetación hasta la puntos más enfocados al diseño y estético.

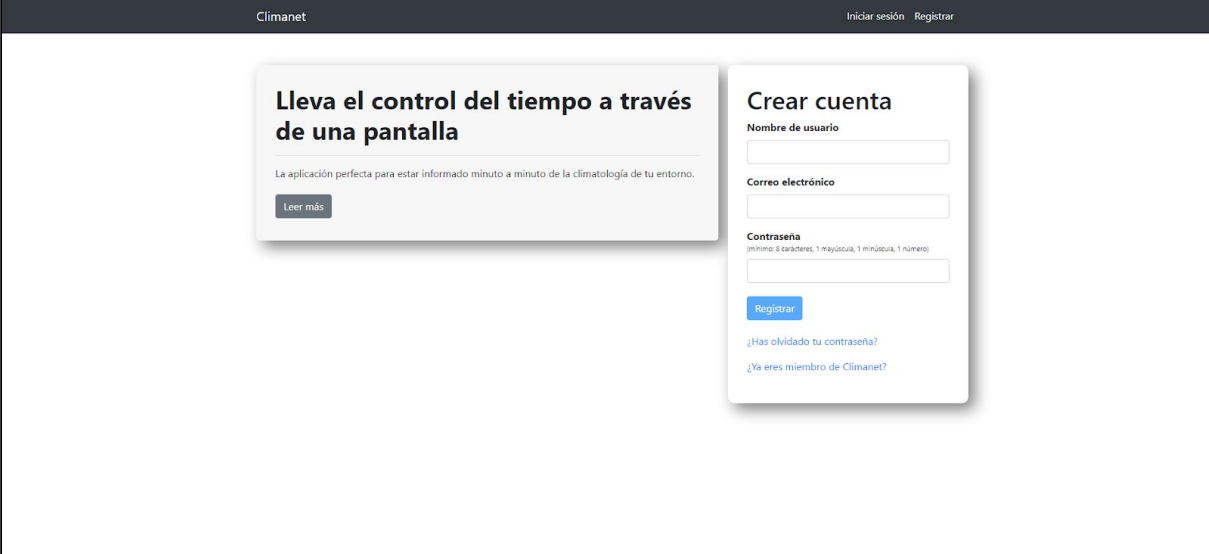
Algunas de las vistas más importantes de Climamet son las siguientes:

- **index**: pantalla inicial de bienvenida cuando no se ha iniciado sesión.
- **home**: página principal, una vez se ha iniciado sesión.
- **login & signup**: formulario de inicio de sesión y registro de nuevos usuarios.
- **header**: encabezado que se comparte a lo largo de las diferentes ventanas de la aplicación.
- **footer**: pie de página que se comparte a lo largo de las diferentes ventanas de la aplicación.

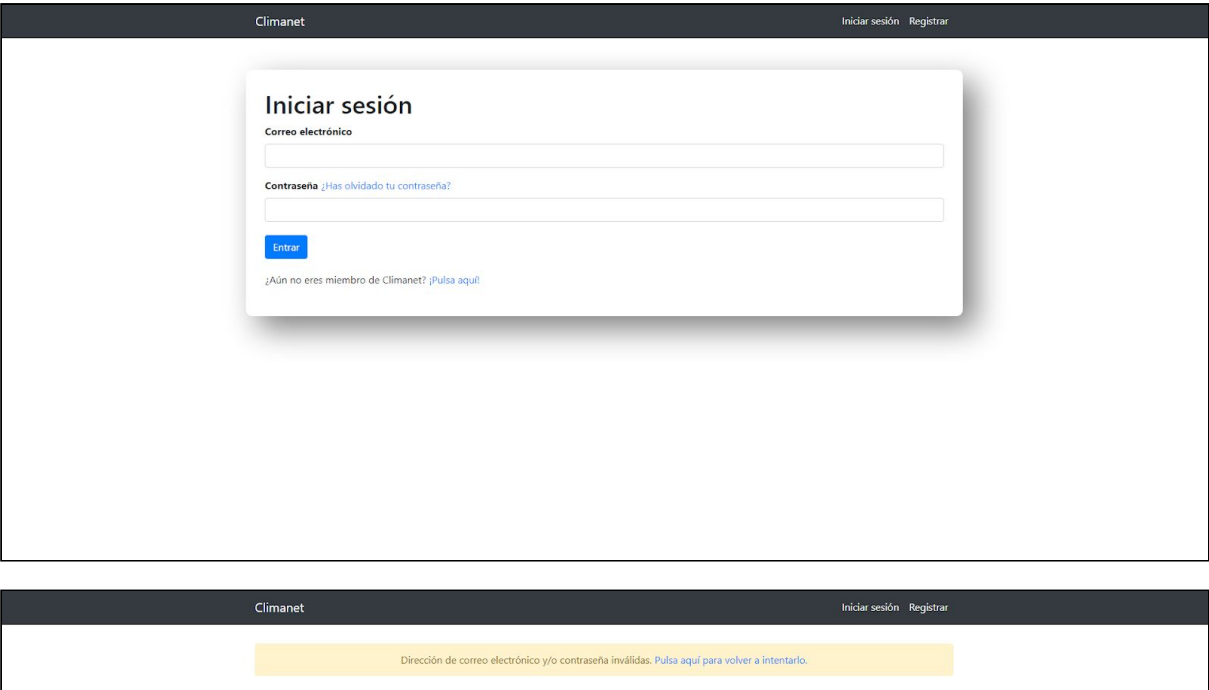
5. Evaluación final

A continuación, voy a adjuntar algunas imágenes del aspecto final de algunas de las ventanas de la aplicación web de Climanet. Se pueden apreciar las diferencias y semejanzas existentes entre los diseños realizados desde un principio, como bien se ven en el apartado *Mockups*. Además de la implementación de algunas ventanas.

Página de bienvenida



Iniciar sesión & mensajes de validación



Registrar usuario & mensajes de validación

Climanet

Iniciar sesión Registrar

Crear cuenta

Nombre de usuario

Correo electrónico

Contraseña
(mínimo 8 caracteres, 1 mayúscula, 1 minúscula, 1 número)

Registrar

¿Ya eres miembro de Climanet? ¡Pulsa aquí!

Climanet

Iniciar sesión Registrar

Este nombre de usuario ya está en uso...

Esta dirección de correo electrónico ya está en uso...

No se pudo crear la cuenta de usuario. [Pulsa aquí para volver a intentarlo.](#)

Página principal (sesión iniciada)

Climanet

Salir

¡Bienvenido a Climanet, marc!

Últimos registros visibles:

10

Filtrar

Desde:

28/05/2020

Hasta:

28/05/2020

Más especificaciones:

Todo

Filtrar por fecha

Mostrando los 10 últimos registros.

Fecha y hora	Temperatura	Humedad
2020-05-23 13:08:28	18.40 °C	58.00 %
2020-05-23 12:08:15	18.40 °C	59.00 %
2020-05-23 12:08:00	18.40 °C	59.00 %
2020-05-22 18:30:58	20.00 °C	66.00 %
2020-05-22 17:30:22	19.00 °C	58.00 %
2020-05-21 16:44:38	19.20 °C	58.00 %
2020-05-21 14:51:19	18.10 °C	60.00 %
2020-05-21 13:44:07	18.10 °C	58.00 %
2020-05-21 12:43:53	18.10 °C	58.00 %
2020-05-21 11:43:40	17.90 °C	63.00 %

6. Conclusiones

Este trabajo me ha permitido desarrollar unas capacidades y habilidades de las cuales me siento bastante satisfecho. Me gusta echar la vista atrás, cuando aún no sabía por dónde coger este proyecto, sin saber qué tecnologías seleccionar, ni cómo usarlas. Sin saber absolutamente nada de componentes electrónicos, y no solamente eso, sino cómo fusionar el mundo de la electrónica con el de la programación web. El hecho de ver que todos los puntos anteriores se han cumplido y llevado a cabo con éxito es una demostración de que el proyecto ha cumplido las expectativas iniciales. El resultado ha sido muy satisfactorio tanto a nivel personal como profesional.

Este proyecto se puede ampliar en varios aspectos, y ese es otro factor que me gusta, lo escalable que puede seguir siendo. A continuación algunas de las posibles mejoras:

- Visualizar los datos en formato de gráficas.
- Acceder a la aplicación fuera de la red privada.
- Añadir otro tipo de sensores, como lumínicos, de calidad de aire, ruido, etc.

7. Bibliografía

Jecrespom. Artículo: *Lenguaje de programación de Arduino, estructura de un programa*. Recuperado de: <https://aprendiendoarduino.wordpress.com/2015/03/26/lenguaje-de-programacion-de-arduino-estructura-de-un-programa/>

Arduino. Documentación: *Language Reference*. Recuperado de: <https://www.arduino.cc/reference/en>

Jecrespom. Artículo: *Hardware Ethernet en Arduino*. Recuperado de: <https://aprendiendoarduino.wordpress.com/tag/w5100/>

Luis Llamas. Artículo: *Conectar Arduino a Internet con módulo Ethernet*. Recuperado de: <https://www.luisllamas.es/arduino-dht11-dht22/>

Jorge. Artículo: *Tutorial LCD, conectando tu arduino a un LCD1602 y LCD2004*. Recuperado de: https://www.naylampmechatronics.com/blog/34_Tutorial-LCD-conectando-tu-arduino-a-un-LCD1602-y-LCD2004/

PHP. Documentación: *Manual de PHP*. Recuperado de: <https://www.php.net/manual/es/index.php>

Bootstrap. Documentación: *Getting started*. Recuperado de:
<https://getbootstrap.com/docs/4.1/getting-started/introduction/>

Wikipedia. Documentación: *Modelo-vista-controlador*. Recuperado de:
<https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador>

8. Código adjuntado

sketch_climanet.io

```
#include <LiquidCrystal.h> //lcd
#include <DHT.h> //sensor
#include <SPI.h> //ethernet
#include <Ethernet.h> //ethernet

//SENSOR DHT11
#define DHTPIN 9 //Digital pin conectado al sensor DHT
#define DHTTYPE DHT11 //Modelo del sensor DHT
DHT dht(DHTPIN, DHTTYPE); //Creacion del objeto del sensor pasando como parametros el pin y el
modelo

//PANTALLA LCD
LiquidCrystal lcd(7,6,5,4,3,2);

//ETHERNET SHIELD
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ipServer(192,168,1,146);
IPAddress ipClient(192,168,1,150);
EthernetClient client;

//VARIABLES PROGRAMA
int led = 8; //led verde
float t; //temperatura
float h; //humedad
const unsigned long second = 1000;
const unsigned long minute = 60000;
```

```
const unsigned long hour = 3600*second;

void setup() {
  Serial.begin(9600);

  dht.begin(); // Inicializa el sensor
  lcd.begin(16,2); //define el número de columnas y filas, respectivamente, de la pantalla lcd
  pinMode(led, OUTPUT); //inicializacion del led como salida

  while(!Serial){

  }
}

void loop() {
  delay(2000);

  t = dht.readTemperature(); //Guardamos el valor de la temperatura en t
  h = dht.readHumidity(); //Guardamos el valor de la humedad en h

  if(isnan(t) || isnan(h)){ //Si los valores son incorrectos retorna al principio del loop
    Serial.println("Error en la lectura de temperatura y humedad!");
    return;
  }

  t = t - 7;
  h = h + 10;

  lcd.setCursor(0,0);
  lcd.print(String(t) + (char)223 + "C " + String(h) + "%"); //(char)223 = °

  //-----CONEXION ARDUINO-MYSQL-----

  if(Ethernet.begin(mac) == 0){
    Serial.println("Failed to configure Ethernet using DHCP");
```



```
Ethernet.begin(mac,ipClient);
}

Serial.println("connecting...");

if(client.connect(ipServer, 80)){
  Serial.println("connected");
  digitalWrite(led, true);
  delay(1000);
  /*client.print("GET /arduino_mysql/mysql.php?temperature=61&humidity=94");*/
  client.print("GET /climanet/controllers/createRecordController.php?temperature=");
  client.print(t);
  client.print("&humidity=");
  client.print(h);
  client.println(" HTTP/1.0");
  client.println("Connection: close");
  client.println();

  client.println();
  digitalWrite(led, false);
}else{
  Serial.println("connection failed");
  return;
}

//-----

if(client.available()){
  char c = client.read();
  Serial.print(c);
}

if(!client.connected()){
  Serial.println();
  Serial.println("disconnecting");
  client.stop();
}
```

```

    while(true);
}

////////////////////////////////////

for(int timer = 60; timer > 0; timer--){
    lcd.setCursor(0,1);
    lcd.print("Reload in: " + String(timer) + "min");
    delay(minute);
}

}

```

home.php

```

<?php
//Checks if the user has logged recently. If not, it will be redirected to index
session_start();
if (!isset($_SESSION["id_user"])) {
    header("Location:login.php");
}else {
    //Imports Classes
    require_once("models/Connection.php");
    require_once("models/UsersManagment.php");

    $conn = new Connection();
    $id_user = $_SESSION["id_user"];

    //Imports header
    require_once("layouts/headerLogged.php");
}
?>

<div class="container" id="home-container">
<div class="row">
<div class="col-md-12">

```

```

    <h3>¡Bienvenido a Climamet, <b><?php echo UsersManagment::getUsername($conn, $id_user);
?></b>!</h3>

</div>

</div>

<hr>

<div class="row">
    <div class="col-md-4">
        <div class="row">
            <div class="col-md-12">
                <form action="#tableRecords" method="post">
                    <div class="form-group">
                        <label>Últimos registros visibles: </label>
                        <select class="form-control" name="txtLimit">
                            <option>10</option>
                            <option>25</option>
                            <option>50</option>
                            <option>100</option>
                            <option>Todos</option>
                        </select>
                    </div>
                    <div class="form-group text-center">
                        <button type="input" class="btn btn-primary btn-block"
name="btnFilterLimit">Filtrar</button>
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>

<!--

<hr style="border:0.5px solid black">
<div class="row text-center">
    <div class="col-md-12">
        <form action="" method="post">
            <button type="input" class="btn btn-outline-danger btn-block"
name="btnMaxTempRecord">Temperatura más alta</button>

```

```
<button type="input" class="btn btn-outline-primary btn-block"
name="btnMinTempRecord">Temperatura más baja</button>
</form>
</div>

<div class="col-md-12">
  <form action="" method="post">
    <button type="input" class="btn btn-outline-danger btn-block"
name="btnMaxHumRecord">Humedad más alta</button>
    <button type="input" class="btn btn-outline-primary btn-block"
name="btnMinHumRecord">Humedad más baja</button>
  </form>
</div>
</div>
-->
<hr style="border:0.5px solid black">
<div class="row">
  <div class="col-md-12">
    <form action="#tableRecords" method="post">
      <div class="form-group">
        <label>Desde: </label>
        <input type="date" class="form-control" name="txtCalendarFrom" id="txtCalendarFrom"
required>
      </div>
      <div class="form-group">
        <label>Hasta: </label>
        <input type="date" class="form-control" name="txtCalendarTo" id="txtCalendarTo"
required>
      </div>
      <div class="form-group">
        <label>Más especificaciones: </label>
        <select class="form-control" name="txtCalendarSpecification">
          <option>Todo</option>
          <option>Temperatura más alta</option>
          <option>Temperatura más baja</option>
```

```
<option>Humedad más alta</option>
<option>Humedad más baja</option>
</select>
</div>
<div class="form-group text-center">
  <button type="input" class="btn btn-primary btn-block"
name="btnFilterBetweenDates">Filtrar por fecha</button>
</div>
</form>
</div>
</div>
</div>
<div class="col-md-8">
  <div class="row" id="tableRecords">
    <div class="col-md-12">
      <table class="table">
        <thead>
          <tr>
            <th scope="col">Fecha y hora</th>
            <th scope="col">Temperatura</th>
            <th scope="col">Humedad</th>
          </tr>
        </thead>
        <tbody>
          <?php
            include_once("../controllers/showRecordsController.php");
          ?>
        </tbody>
      </table>
    </div>
  </div>
</div>
</div>
</div>
```

```
<!--Call to my own js functions-->
<script src="js/getDate.js"></script>
<script src="js/clsSignup.js"></script>
```

```
<?php
    //Imports the footer
    require_once("layouts/footer.php");
?>
```

index.php

```
<?php
    //Checks if the user has logged recently. If so, it will be redirected to home
    session_start();
    if (isset($_SESSION["id_user"])) {
        header("Location:home.php");
    }
?>
```

```
<?php
    //Imports the header
    require_once("layouts/headerUnlogged.php");
?>
```

```
<div class="container" id="index-container">
    <div class="row">
        <div class="col-md-8 no-padding">
            <div class="row">
                <div class="col-md-12">
                    <div class="jumbotron">
                        <h1><b>Lleva el control del tiempo a través de una pantalla</b></h1>
                        <hr>
                        <p>La aplicación perfecta para estar informado minuto a minuto de la climatología de tu
entorno.</p>
                        <button type="button" class="btn btn-secondary">Leer más </button>
                    </div>
                </div>
            </div>
        </div>
    </div>
```

```

    </div>
  </div>
</div>
<div class="col-md-4" id="login-index-container">
  <div class="row">
    <div class="col-md-12 no-padding">
      <h1>Crear cuenta</h1>
      <form action="controllers/createUserController.php" method="post">
        <div class="form-group">
          <label><b>Nombre de usuario</b></label>
          <input type="text" class="form-control" name="txtUsername" required>
        </div>
        <div class="form-group">
          <label><b>Correo electrónico</b></label>
          <input type="email" class="form-control" name="txtEmail" required>
        </div>
        <div class="form-group">
          <label><b>Contraseña</b> <div style="font-size: 11px">(mínimo: 8 caracteres, 1 mayúscula,
1 minúscula, 1 número)</div></label>
          <input type="password" class="form-control" name="txtPassword" id="txtPassword"
oninput="ValidatePasswordFormat()" required>
        </div>
        <div class="form-group">
          <button type="submit" class="btn btn-primary" name="btnSignup" id="btnSignup"
disabled>Registrar</button>
        </div>
        <div class="form-group">
          <p><a href="#">¿Has olvidado tu contraseña?</a></p>
        </div>
        <div class="form-group">
          <p><a href="/login.php">¿Ya eres miembro de Climamet?</a></p>
        </div>
      </form>
    </div>
  </div>
</div>

```

```

    </div>
  </div>
</div>

<!--Call to my own js functions-->
<script src="js/clsSignup.js"></script>

<?php
    require_once("layouts/footer.php");
?>

```

login.php

```

<?php
    //Checks if the user has logged recently. If so, it will be redirected to home
    session_start();
    if (isset($_SESSION["id_user"])) {
        header("Location:home.php");
    }
?>

<?php
    //Imports the header
    require_once("layouts/headerUnlogged.php");
?>

<div class="container" id="login-container">
    <div class="row">
        <div class="col-md-12 no-padding">
            <h1>Iniciar sesión</h1>
            <form action="controllers/validateUserController.php" method="post">
                <div class="form-group">
                    <label><b>Correo electrónico</b></label>
                    <input type="email" class="form-control" name="txtEmail" required>
                </div>
                <div class="form-group">

```



```

        <label><b>Contraseña</b><a href="#"> ¿Has olvidado tu contraseña?</a></label>
        <input type="password" class="form-control" name="txtPassword" required>
    </div>
    <div class="form-group">
        <button type="submit" class="btn btn-primary" name="btnLogin">Entrar</button>
    </div>
    <div class="form-group">
        <p>¿Aún no eres miembro de Climamet? <a href="signup.php">¡Pulsa aquí!</a></p>
    </div>
</form>
</div>
</div>

<?php
    require_once("layouts/footer.php");
?>

```

signup.php

```

<?php
    //Checks if the user has logged recently. If so, it will be redirected to home
    session_start();
    if (isset($_SESSION["id_user"])) {
        header("Location:home.php");
    }
?>

<?php
    //Imports the header
    require_once("layouts/headerUnlogged.php");
?>

<div class="container" id="signup-container">
    <div class="row">
        <div class="col-md-12 no-padding">

```

```

<h1>Crear cuenta</h1>
<form action="controllers/createUserController.php" method="post">
  <div class="form-group">
    <label><b>Nombre de usuario</b></label>
    <input type="text" class="form-control" name="txtUsername" required>
  </div>
  <div class="form-group">
    <label><b>Correo electrónico</b></label>
    <input type="email" class="form-control" name="txtEmail" required>
  </div>
  <div class="form-group">
    <label><b>Contraseña</b> <div style="font-size: 11px">(mínimo: 8 caracteres, 1 mayúscula,
1 minúscula, 1 número)</div></label>
    <input type="password" class="form-control" name="txtPassword" id="txtPassword"
oninput="ValidatePasswordFormat()" required>
  </div>
  <div class="form-group">
    <button disabled type="submit" class="btn btn-primary" name="btnSignup"
id="btnSignup">Registrar</button>
  </div>
  <div class="form-group">
    <p>¿Ya eres miembro de Climamet? <a href="login.php">¡Pulsa aquí!</a></p>
  </div>
</form>
</div>
</div>

<!--Call to my own js functions-->
<script src="js/clsSignup.js"></script>

<?php
  require_once("layouts/footer.php");
?>

```

Connection.php

```
<?php
class Connection{
    private $conn;

    function __construct(){
        try {
            $this->conn = new PDO("mysql:host=localhost;dbname=climanet;port=3306", "marc",
"Pruebashernest1");
            $this->conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        } catch (\Exception $e) {
            echo "Se ha producido un error: " . $e->getMessage();
        }
    }

    function getConnection(){
        return $this->conn;
    }
}
?>
```

Records.php

```
<?php
class Records{

    public static function createRecord($conn,$temperature,$humidity){ //arduino llama a esta funcion
para los registros
        try {
            $query = "INSERT INTO records (temperature_record, humidity_record) VALUES
(:temperature,:humidity)";
            $stmt = $conn->getConnection()->prepare($query);
            $stmt->bindValue(":temperature", $temperature);
            $stmt->bindValue(":humidity", $humidity);
            $stmt->execute();
            echo "Registro realizado";
        }
    }
}
```

```
} catch (\Exception $e) {  
    echo $e->getMessage();  
}  
}
```

```
public static function showUnfilteredRecords($conn){  
    echo "Mostrando los <b>10</b> últimos registros."  
    $query = "SELECT * FROM records ORDER BY id_record DESC LIMIT 10";  
    $stmt = $conn->getConnection()->prepare($query);  
    $stmt->execute();  
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);  
    return $movements_list;  
}
```

```
public static function showFilterLimitRecords($conn,$limit){  
    if ($limit == "Todos") {  
        $query = "SELECT * FROM records ORDER BY id_record DESC";  
    }else {  
        $query = "SELECT * FROM records ORDER BY id_record DESC LIMIT $limit";  
    }  
    $stmt = $conn->getConnection()->prepare($query);  
    $stmt->execute();  
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);  
    return $movements_list;  
}
```

```
public static function showFilterBetweenDates($conn, $from, $to, $specification){  
    if ($specification == "Todo") {  
        $query = "SELECT * FROM records WHERE datetime_record > '$from' AND datetime_record  
< '$to 23:59:59' ORDER BY id_record DESC";  
    }else if ($specification == "Temperatura más alta") {  
        $query = "SELECT *  
            FROM records  
            WHERE temperature_record = (SELECT MAX(temperature_record) FROM records  
WHERE datetime_record > '$from' AND datetime_record < '$to 23:59:59')  
            WHERE datetime_record > '$from' AND datetime_record < '$to 23:59:59')";  
    }  
    $stmt = $conn->getConnection()->prepare($query);  
    $stmt->execute();  
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);  
    return $movements_list;  
}
```

```

        AND datetime_record > '$from' AND datetime_record < '$to 23:59:59';
    }else if ($specification == "Temperatura más baja") {
        $query = "SELECT *
        FROM records
        WHERE temperature_record = (SELECT MIN(temperature_record) FROM records
WHERE datetime_record > '$from' AND datetime_record < '$to 23:59:59')
        AND datetime_record > '$from' AND datetime_record < '$to 23:59:59';
    }else if ($specification == "Humedad más alta") {
        $query = "SELECT *
        FROM records
        WHERE humidity_record = (SELECT MAX(humidity_record) FROM records WHERE
datetime_record > '$from' AND datetime_record < '$to 23:59:59')
        AND datetime_record > '$from' AND datetime_record < '$to 23:59:59';
    }else if ($specification == "Humedad más baja") {
        $query = "SELECT *
        FROM records
        WHERE humidity_record = (SELECT MIN(humidity_record) FROM records WHERE
datetime_record > '$from' AND datetime_record < '$to 23:59:59')
        AND datetime_record > '$from' AND datetime_record < '$to 23:59:59';
    }
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->execute();
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $movements_list;
}

public static function showMaxTempRecord($conn){
    $query = "SELECT * FROM records WHERE temperature_record = (SELECT
MAX(temperature_record) FROM records) ORDER BY id_record DESC";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->execute();
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $movements_list;
}

```

```

public static function showMinTempRecord($conn){
    $query = "SELECT * FROM records WHERE temperature_record = (SELECT
MIN(temperature_record) FROM records) ORDER BY id_record DESC";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->execute();
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $movements_list;
}

```

```

public static function showMaxHumRecord($conn){
    $query = "SELECT * FROM records WHERE humidity_record = (SELECT
MAX(humidity_record) FROM records) ORDER BY id_record DESC";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->execute();
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $movements_list;
}

```

```

public static function showMinHumRecord($conn){
    $query = "SELECT * FROM records WHERE humidity_record = (SELECT
MIN(humidity_record) FROM records) ORDER BY id_record DESC";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->execute();
    $movements_list = $stmt->fetchAll(PDO::FETCH_ASSOC);
    return $movements_list;
}
}
?>

```

UsersManagment.php

```

<?php
class UsersManagment{

    public static function validateUser($conn, $email, $pass){
        $query = "SELECT * FROM users WHERE email_user = :email AND password_user = :pass";
    }
}

```

```
$stmt = $conn->getConnection()->prepare($query);
$stmt->bindValue(":email", $email);
$stmt->bindValue(":pass", $pass);
$stmt->execute();
if ($stmt->rowCount() > 0) {
    return true;
} else {
    return false;
}
}
```

```
public static function createUser($conn, $username, $password, $email){
    $query = "INSERT INTO users (username_user, password_user, email_user) VALUES
(:username, :password, :email)";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->bindValue(":username", $username);
    $stmt->bindValue(":password", $password);
    $stmt->bindValue(":email", $email);
    $stmt->execute();
    if ($stmt->rowCount() > 0) {
        return true;
    } else {
        return false;
    }
}
```

```
public static function getId($conn, $email){
    $query = "SELECT id_user FROM users WHERE email_user = :email";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->bindValue(":email", $email);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row["id_user"];
}
```

```

public static function getUsername($conn, $id){
    $query = "SELECT * FROM users WHERE id_user = :id";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->bindValue(":id", $id);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row["username_user"];
}

public static function getRole($conn, $id){
    $query = "SELECT * FROM users WHERE id_user = :id";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->bindValue(":id", $id);
    $stmt->execute();
    $row = $stmt->fetch(PDO::FETCH_ASSOC);
    return $row["admin_user"];
}
}
?>

```

ValidateData.php

```

<?php
class ValidateData{
    public static function existsUsername($conn, $username){
        $query = "SELECT * FROM users WHERE username_user = :username";
        $stmt = $conn->getConnection()->prepare($query);
        $stmt->bindValue(":username", $username);
        $stmt->execute();
        if ($stmt->rowCount() > 0) { //Quiere decir que existe el username pasado por parametro
            return true;
        } else { //No existe
            return false;
        }
    }
}

```



```

public static function existsEmail($conn, $email){
    $query = "SELECT * FROM users WHERE email_user = :email";
    $stmt = $conn->getConnection()->prepare($query);
    $stmt->bindValue(":email", $email);
    $stmt->execute();
    if ($stmt->rowCount() > 0) { //Quiere decir que existe el email pasado por parametro
        return true;
    }else { //No existe
        return false;
    }
}
}
?>

```

createRecordController.php

```

<?php
require_once("../models/Connection.php");
require_once("../models/Records.php");

$temperature = $_GET["temperature"];
$humidity = $_GET["humidity"];

echo "Temperatura: " . $temperature . "<br>";
echo "Humedad: " . $humidity . "<br>";

$conn = new Connection();
Records::createRecord($conn, $temperature, $humidity);
?>

```

createUserController.php

```

<?php
if (isset($_POST["btnSignup"])) {
    //Imports necessary classes
    require_once("../models/Connection.php");
    require_once("../models/UsersManagment.php");
}

```

```
require_once("../models/ValidateData.php");

//Imports header
require_once("../layouts/headerUnlogged.php");

$username = $_POST["txtUsername"];
$email = $_POST["txtEmail"];
$password = $_POST["txtPassword"];

$conn = new Connection();

$usernameAvailable;
$emailAvailable;

if (ValidateData::existsUsername($conn, $username) == false) { //Username disponible para
asignar
    $usernameAvailable = true;
} else {
    $usernameAvailable = false;
    include_once("../layouts/existingUserMessage.html");
}

if (ValidateData::existsEmail($conn, $email) == false) { //Password disponible para asignar
    $emailAvailable = true;
} else {
    $emailAvailable = false;
    include_once("../layouts/existingEmailMessage.html");
}

if ($usernameAvailable == true && $emailAvailable == true) {
    UsersManagment::createUser($conn, $username, $password, $email);
    session_start();
    $id_user = UsersManagment::getId($conn, $email);
    $_SESSION["id_user"] = $id_user;
    header("Location:../home.php");
}
```

```
    }else {  
        include_once("../layouts/incorrectUserCreationMessage.html");  
    }  
}else {  
  
}  
  
//Imports footer  
require_once("../layouts/footer.php");  
?>
```

logoutController.php

```
<?php  
session_start();  
  
if (isset($_SESSION["id_user"])) {  
    unset($_SESSION["id_user"]);  
    setcookie("name_user", "", time() - 3600);  
    header("Location:../index.php");  
}  
?>
```

showRecordsController.php

```
<?php  
require_once("models/Connection.php");  
require_once("models/Records.php");  
require_once("models/UsersManagment.php");  
  
$conn = new Connection();  
  
if (isset($_POST["btnFilterBetweenDates"])) {  
  
    $from = $_POST["txtCalendarFrom"];  
    $to = $_POST["txtCalendarTo"];  
    $specification = $_POST["txtCalendarSpecification"];
```

```
    echo "Mostrando <b>" . lcfirst($specification) . "</b> desde el <b>" . $from . "</b> hasta el <b>" .
$to . "</b><br>";

    $records_list = Records::showFilterBetweenDates($conn, $from, $to, $specification);

} elseif (isset($_POST["btnMaxTempRecord"])) {

    $records_list = Records::showMaxTempRecord($conn);

} elseif (isset($_POST["btnMinTempRecord"])) {

    $records_list = Records::showMinTempRecord($conn);

} elseif (isset($_POST["btnMaxHumRecord"])) {

    $records_list = Records::showMaxHumRecord($conn);

} elseif (isset($_POST["btnMinHumRecord"])) {

    $records_list = Records::showMinHumRecord($conn);

} else if (isset($_POST["btnFilterLimit"])) {

    $limit = $_POST["txtLimit"];
    if ($limit == "Todos") {
        echo "Mostrando <b>todos</b> los registros.";
    } else {
        echo "Mostrando los <b>" . $limit . "</b> últimos registros.";
    }
    $records_list = Records::showFilterLimitRecords($conn, $limit);

} else { //Si no se ha hecho ningun filtro se muestran los 100 ultimos registros

    $records_list = Records::showUnfilteredRecords($conn);

}
```

```
foreach ($records_list as $record) {  
    echo "<tr>";  
    echo "<td>" . $record["datetime_record"] . "</td>";  
    echo "<td>" . $record["temperature_record"] . " °C</td>";  
    echo "<td>" . $record["humidity_record"] . " %</td>";  
    echo "</tr>";  
}  
?>
```

validateUserController.php

```
<?php  
if (isset($_POST["btnLogin"])) {  
    //Imports the necessary classes  
    require_once("../models/Connection.php");  
    require_once("../models/UsersManagment.php");  
  
    //Imports header  
    require_once("../layouts/headerUnlogged.php");  
  
    $email = $_POST["txtEmail"];  
    $password = $_POST["txtPassword"];  
  
    $conn = new Connection();  
    if (UsersManagment::validateUser($conn,$email,$password) == true) {  
        session_start();  
        $id_user = UsersManagment::getId($conn, $email);  
        $_SESSION["id_user"] = $id_user;  
        header("Location:../home.php");  
    } else {  
        include_once("../layouts/incorrectUserValidationMessage.html");  
    }  
}  
  
//Imports footer
```

```
require_once("../layouts/footer.php");  
?>
```

styles.css

```
.jumbotron {  
    padding: 0px;  
    box-shadow: 10px 10px 25px grey;  
    padding: 30px;  
    background-color: #f7f7f7;  
}
```

```
.navbar-brand-centered{  
    text-align: center;  
    width: 100%;  
}
```

```
button{  
    margin-top:5px;  
    margin-bottom: 5px;  
}
```

```
/******
```

```
#index-container{  
    margin: 50px auto 0;  
    background-color: white;  
    border-radius: 10px;  
}
```

```
#login-index-container{  
    /*padding: 10px;  
    background-color: white;*/  
    padding: 30px;  
    background-color: white;  
    border-radius: 10px;
```

```
    box-shadow: 10px 10px 25px grey;
}
```

```
#login-container{
    margin: 50px auto 0;
    padding: 30px;
    background-color: white;
    border-radius: 10px;
    box-shadow: 20px 20px 50px grey;

}
```

```
/*Sign Up Page*/
#signup-container{
    margin: 50px auto 0;
    padding: 30px;
    background-color: white;
    border-radius: 10px;
    box-shadow: 20px 20px 50px grey;
}
```

```
/*Home Page*/
#home-container{
    margin: 30px auto 0;
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    line-height: 30px;
}
```

clsSignup.js

```
function ValidatePasswordFormat(){
    var tPW= document.getElementById("txtPassword").value;
    var button = document.getElementById("btnSignup");
```

```
if (tPW.length<8){
    console.log('Password inferior a 8 caracteres');
    button.disabled = true;
    return false;
}else {
    button.disabled = true;
    var mayus = false;
    var minus = false;
    var num = false;

    for (var i = 0; i < tPW.length; i++) {
        if (tPW.charCodeAt(i) >= 65 && tPW.charCodeAt(i) <= 90) {
            mayus = true;
        }else if (tPW.charCodeAt(i) >= 97 && tPW.charCodeAt(i) <= 122) {
            minus = true;
        }else if (tPW.charCodeAt(i) >= 48 && tPW.charCodeAt(i) <= 57) {
            num = true;
        }
    }
}

if (mayus==true && minus==true && num==true) {
    button.disabled = false;
}
}
```

getDate.js

```
var date = new Date();

var day = date.getDate();
var month = date.getMonth() + 1;
var year = date.getFullYear();

if (month < 10) month = "0" + month;
if (day < 10) day = "0" + day;
```



```
var today = year + "-" + month + "-" + day;
document.getElementById("txtCalendarFrom").value = today;
document.getElementById("txtCalendarTo").value = today;
```

existingEmailMessage.html

```
<div class="container">
  <div class="row text-center">
    <div class="col-md-12 no-padding">
      <div class="alert alert-danger" role="alert" style="margin-top:30px">
        Esta dirección de correo electrónico ya está en uso...
      </div>
    </div>
  </div>
</div>
```

existingUserMessage.html

```
<div class="container">
  <div class="row text-center">
    <div class="col-md-12 no-padding">
      <div class="alert alert-danger" role="alert" style="margin-top:30px">
        Este nombre de usuario ya está en uso...
      </div>
    </div>
  </div>
</div>
```

footer.php

```
<!--Bootstrap JS-->
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCKRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hX
pG5KkN" crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b
4Q" crossorigin="anonymous"></script>
```

```
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCm
Yl" crossorigin="anonymous"></script>
</body>
</html>
```

headerLogged.php

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Climanet</title>
  <!--Bootstrap CSS-->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAi
S6JXm" crossorigin="anonymous">
  <link rel="icon" href="img/favicon.png">
  <link rel="stylesheet" href="css/styles.css">
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    <div class="container">
      <a class="navbar-brand" href="index.php">Climanet</a>
      <button type="button" class="navbar-toggler" data-toggle="collapse" data-target="#navbarText"
aria-controls="navbarText" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarText">
        <ul class="navbar-nav ml-auto">
          <li class="nav-item active">
            <a class="nav-link" href="controllers/logoutController.php">Salir</a>
          </li>
```

```

    </ul>
  </div>
</div>
</nav>

```

headerUnlogged.php

```

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Climanet</title>
  <!--Bootstrap CSS-->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAi
S6JXm" crossorigin="anonymous">
  <link rel="icon" href="/climanet/img/favicon.png">
  <link rel="stylesheet" href="/climanet/css/styles.css">
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    <div class="container">
      <a class="navbar-brand" href="/climanet/index.php">Climanet</a>
      <button type="button" class="navbar-toggler" data-toggle="collapse" data-target="#navbarText"
aria-controls="navbarText" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarText">
        <ul class="navbar-nav ml-auto">
          <li class="nav-item active">
            <a class="nav-link" href="/climanet/login.php">Iniciar sesión</a>
          </li>
          <li class="nav-item active">

```

```
<a class="nav-link" href="/climanet/signup.php">Registrar</a>
</li>
</ul>
</div>
</div>
</nav>
```

incorrectUserCreationMessage.html

```
<div class="container">
  <div class="row text-center">
    <div class="col-md-12 no-padding">
      <div class="alert alert-warning" style="margin-top:30px">
        No se pudo crear la cuenta de usuario. <a href="../signup.php">Pulsa aquí para volver a
        intentarlo.</a>
      </div>
    </div>
  </div>
</div>
```

incorrectUserValidationMessage.html

```
<div class="container">
  <div class="row text-center">
    <div class="col-md-12 no-padding">
      <div class="alert alert-warning" style="margin-top:30px">
        Dirección de correo electrónico y/o contraseña inválidas. <a href="../login.php">Pulsa aquí
        para volver a intentarlo.</a>
      </div>
    </div>
  </div>
</div>
```