

CAMPAÑA DE VERANO

Memoria de proyecto

Desarrollado por:

Inmaculada López Ugalde

AGRADECIMIENTOS

Aunque siempre estuvo en mi cabeza volver a estudiar, nunca era una idea real, siempre algo pasajero, por épocas.

Siempre con la idea de que no había elegido, en mi juventud, los estudios de manera correcta, que no me habían orientado bien, que no había sido valiente.

En 2023 llega alguien a mi vida, que aunque fuese por poco tiempo, me hizo creer que era posible, esa persona era un buen ejemplo de que se podía. Y si, un curso después estoy presentando este proyecto.

Agradezco a esa persona por encender la primera bombilla y darme seguridad para dar el primer paso. A los profesores, por darme más bombillas y cable para llegar al final del túnel, bueno, de este primer túnel. También por facilitarme los conocimientos para la estructura de este alumbrado. Gracias

Sirva esta memoria como etiqueta de apertura de un nuevo proyecto de vida.

Sirva de clase de muchos objetos.

TÍTULO:

Aplicación para la gestión de la Campaña de Verano

DESCRIPCIÓN:

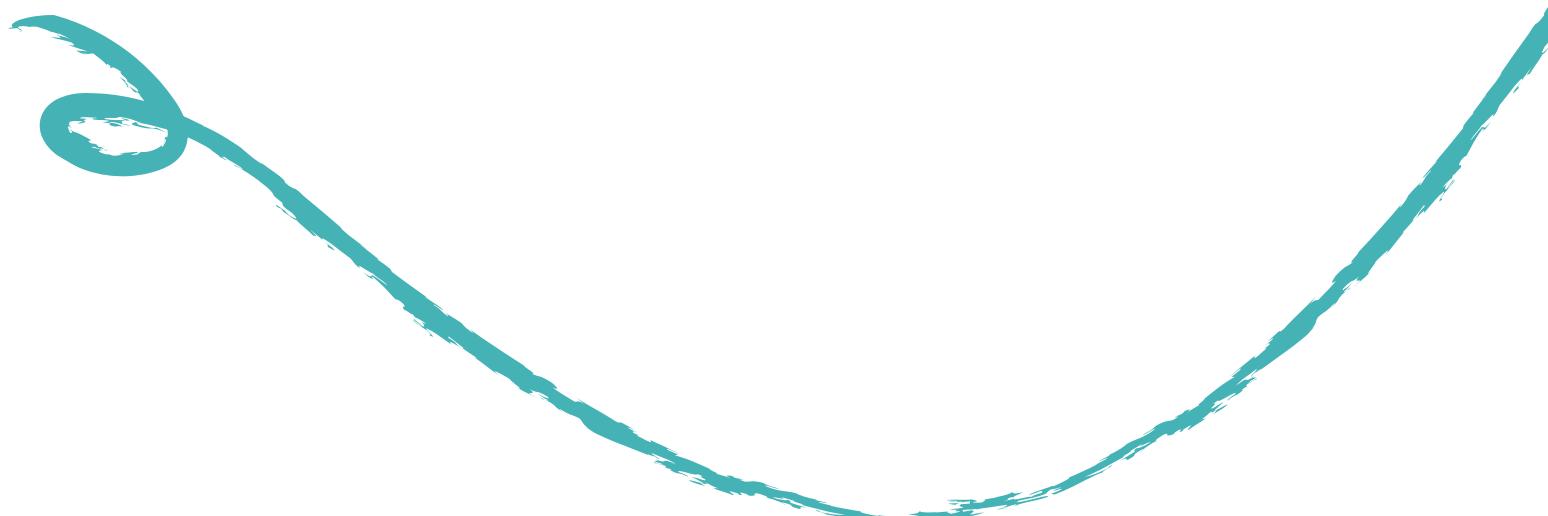
Aplicación web para la inscripción de usuarios y gestión de los mismos para la participación en la Campaña de Verano.

REPOSITORIO GITHUB

<https://github.com/llopuga/CVerano/>

VÍDEO PRESENTACIÓN

[Enlace a la presentación](#)



ÍNDICE

RESUMEN	6
ABSTRACT	7

01

INTRODUCCIÓN

1.1 Estructura del documento	8
1.2 Justificación del proyecto	10
1.3 Caso de estudio	11
1.4 Descripción general	13
1.5 Objetivos	19
1.6 Requisitos	20

02

ANÁLISIS

2.1 Casos de uso	21
- Usuario externo	22
- Usuario registrado	27
2.2 Diagrama de clase	36
2.3 Diagrama Entidad-Relación	39

03

DISEÑO

3.1 Arquitectura	41
3.2 Capa de presentación	42
3.3 Capa de aplicación	43
3.4 Capa de datos	45

ÍNDICE

04

IMPLEMENTACIÓN

4.1 Tecnologías	<u>46</u>
4.2 Herramientas	<u>49</u>
4.3 Código	
4.3.1 Front-end	<u>54</u>
4.3.2 Back-end	<u>56</u>
4.3.3 Consultas	<u>57</u>
4.3.4 Dificultades	<u>59</u>
4.4 Documentación. JavaDoc	<u>61</u>
4.5 Control de versiones	<u>62</u>
4.6 Pruebas unitarias	<u>63</u>
4.7 Despliegue	<u>64</u>

05

FUTUROS TRABAJOS

70

06

CONCLUSIONES

71

07

BIBLIOGRAFÍA

72

RESUMEN

Esta aplicación se desarrolla para solucionar el problema existente en la gestión de usuarios y modo de participación en actividades lúdicas en verano, convocadas por las administraciones públicas.

Este programa es accesible a través de web, donde se completará un formulario para la inscripción de usuarios y posteriormente se gestionarán las participaciones. Se utilizarán las tecnologías web HTML, CSS, JAVA, JAVASCRIPT, MYSQL a priori.

El portal será escalable, ya que en futuras acciones se pretende ampliar y mejorar las funcionalidades. Estas mejoras irán en el camino de hacer un sistema más justo, abierto y modificable, inclusivo y accesible.

ABSTRACT

This application has been developed to solve the existing problem in the management of users and how to participate in summer leisure activities organised by public administrations.

This programme is accessible through the web, where a form will be filled in for the registration of users and then the participations will be managed. The web technologies used will be HTML, CSS, JAVA, JAVASCRIPT, MYSQL a priori.

The portal will be scalable, as in future actions it is intended to expand and improve the functionalities. These improvements will go in the direction of making the system fairer, more open and modifiable, inclusive and accessible.



1. INTRODUCCIÓN

1.1 ESTRUCTURA DEL DOCUMENTO

En este documento encontraremos en cada uno de los capítulos las explicaciones necesarias para entender el proyecto global y cómo se ha desarrollado.

Introducción: Descripción inicial del proyecto donde se explica el objetivo del mismo, así como los motivos que han llevado a realizarlo.

Análisis: Descripción de la estructura y funcionalidad que tendrá nuestro portal, así como los diagramas más significativos para entender su funcionamiento.

Diseño: En la parte de diseño se explican los diferentes niveles de la arquitectura de la aplicación (presentación, lógico y de persistencia).

Implementación: Las tecnologías implementadas y descripción de las herramientas utilizadas para la creación, también se muestran ejemplos de desarrollo y dificultades encontradas

Futuros trabajos: Algunas funcionalidades interesantes de incluir.

Conclusiones: Conclusiones desde un punto de vista personal sobre la aplicación y su finalidad y sobre el proceso.

Bibliografía: Relación de la documentación consultada para la realización de esta memoria.

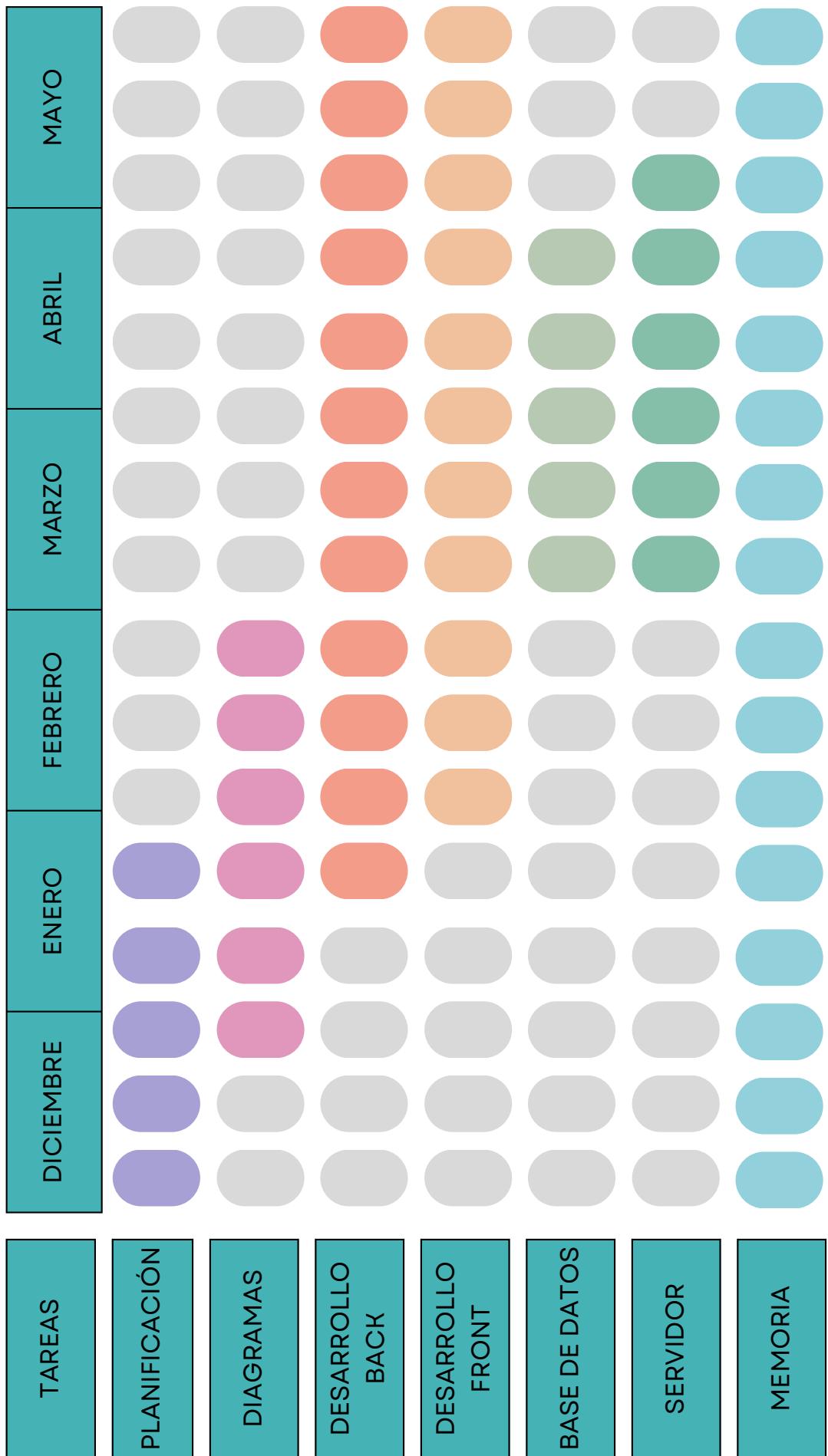
Guía del usuario: Formato video explicando cada una de las partes

Anexos: Junto con el proyecto se entregará una carpeta con material y gráficas referenciado en esta memoria. Se presenta así para una mayor comprensión y facilitar la lectura. Carpeta [Material de referencia](#)



DIAGRAMA GANTT. PROYECTO INTEGRADOR

CAMPAÑA DE VERANO



1.2 JUSTIFICACIÓN DEL PROYECTO

Me gustaría empezar explicando que es Campaña de Verano, ya que trabajo en el organismo de mi comunidad encargado de los temas de juventud y es el ámbito en el que quiero centrar mi proyecto.

Para dar solución a esta problemática actualmente existe una aplicación que hace los apartados básicos de este proceso. Por todo ello, quiero trabajar para conseguir un proceso más automatizado e independiente, que facilite el trabajo y lo haga más justo.

Campaña de Verano es un programa que actualmente existe en todas las Administraciones públicas de todas comunidades de España. Se trata de una serie de actividades que se ofertan desde las entidades públicas para que todos los jóvenes tenga la opción de tener un verano diferente.

El objetivo del programa también se basa en dar acceso a personas que no podrían participar si estas actividades no tuviesen precios públicos.

La Campaña de Verano está enfocada a jóvenes de 18 a 30 años. Las actividades son llamadas Campos de voluntariado, cuyo fin es aprender, compartir y ayudar.

En nuestro sitio web dispondremos de toda la información básica del programa, información detallada de cada una de las actividades, formulario de participación y datos básicos de contacto.

Una vez recibidas las solicitudes, se examinarán y gestionarán por parte de los administradores, obteniendo así la relación de participantes en las actividades.

1.3 CASO DE ESTUDIO

Para esta Campaña de Verano necesitamos una web que muestre toda la información pública referente al programa, algunos comentarios de anteriores participantes, formulario de inscripción, buscador para la consulta del estado de la solicitud y características específicas de las actividades. También, un apartado de acceso para los administradores, un apartado de contacto y redes sociales.

En esta parte pública, el formulario recogerá los datos básicos de cada participante. Uno de los campos fundamentales de este formulario será el campo actividad, desde este desplegable podrá elegir en qué actividad desea participar. Las opciones serán las que los administradores hayan incluido previamente. Otro de los campos fundamentales es el DNI, ya que solo permitiremos una participación por persona, controlado por este campo. Igualmente, este campo DNI nos da la posibilidad de que, cuando añadimos una solicitud, comprobamos que ha sido añadida de manera correcta. Además, en cualquier momento, podemos acceder al apartado buscador de solicitud y ver el estado de nuestra solicitud y como avanza en el proceso.

En la parte privada, accesible desde el apartado login, necesitaremos que los administradores, puedan gestionar todas las variables involucradas en el proceso:

1. Creación y gestión de Administradores.
2. Creación y gestión de Actividades.
3. Gestión de Solicitudes.
4. Procesos
 - a. Asignar número. Recoge todas las participaciones, las ordena por id y le asigna un número de participación.
 - b. Número aleatorio. Genera un número aleatorio desde el 1 hasta el número de solicitudes válidas que tenemos(número que hay que introducir).

Estos métodos son necesarios ya que las plazas disponibles no son suficientes para cubrir la demanda.

Una vez realizado el sorteo, la administración contactará con los adjudicatarios y les informará que tienen un plazo para remitir la documentación necesaria.

Finalizado el plazo, el usuario con plaza asignada y documentación correcta pasará a un listado definitivo. Si hay usuarios que no finalizan el proceso por incumplir el procedimiento en tiempo o forma, perderán su derecho a disfrutar de la plaza obtenida y su puesto será ofrecido al siguiente usuario en la lista.

Toda la información del proceso será publicada en el portal web de manera pública.

1.4 DESCRIPCIÓN GENERAL

Nuestro portal web se dividirá en una parte pública, Front-end, en la que mostraremos toda la información general del proceso, además de imágenes para hacerlo más atractivo.

Ya en el apartado de Back-end, crearemos la base de datos y la lógica relacional para realizar el proceso, la gestión de actividades, solicitudes y administradores y los distintos procesos necesarios (asignación de número, sorteo, etc.).

Todas las imágenes se pueden encontrar en la carpeta **Material de referencia/HTML**, [archivo_portal_web.pdf](#) y [responsive.pdf](#) para la versión responsive

Nuestra página de inicio sigue este estilo:

The screenshot shows the homepage of the "Summer Campaign 2024" website. At the top, there is a navigation bar with a logo on the left, followed by the text "INICIO ACTIVIDADES SOLICITUD" and a user icon. Below the navigation bar is a colorful, stylized illustration of a beach scene with boats, umbrellas, and people. The main heading "¡Bienvenidos a la Campaña de Verano 2024!" is centered above a text block. This text block contains information about the national volunteer program, mentioning its purpose, target audience, and participation details. Below this are three testimonial boxes with quotes from students. At the bottom of the page is a footer section with the campaign logo, navigation links (Inicio, Actividades, Solicitudes), an address (Calle Voluntariado 123, Mérida), and social media links for Facebook, Twitter, and Instagram.

INICIO ACTIVIDADES SOLICITUD

¡Bienvenidos a la Campaña de Verano 2024!

Los programas de voluntariado nacional ofrecen una alternativa única para disfrutar del verano mientras se realiza una contribución social importante. Estos programas atraen a jóvenes con intereses similares, tanto locales como de otras regiones, incluyendo participantes internacionales.

Las actividades de estos programas cubren una amplia gama de temas, tales como medio ambiente, conservación del patrimonio, arqueología, y apoyo a comunidades con necesidades especiales de integración. Este enfoque multidisciplinario asegura que hay oportunidades para todos, independientemente de sus intereses específicos.

Estos campos se llevan a cabo principalmente durante los meses de verano, ofreciendo opciones en diversas localidades. La participación está abierta a jóvenes adultos, típicamente entre 18 y 30 años.

La cuota de participación generalmente incluye alojamiento, alimentación, materiales necesarios y actividades adicionales. Los participantes suelen ser responsables del transporte hacia y desde el lugar del programa. La organización del campo cubre seguros de responsabilidad civil y de accidentes.

"Una experiencia transformadora que combinó mi amor por la biología con acciones prácticas de conservación. Todo ahora es diferente ¡Inolvidable!"
- Julia, estudiante de biología

"Participar en la conservación del patrimonio me permitió conectar profundamente con la historia. Aprendí mucho y conocí gente maravillosa."
- Marcos, entusiasta de la historia

"Un verano dedicado a apoyar a comunidades necesitadas. Profundamente gratificante y educativo, reforzó mi pasión por el trabajo social."
- Ana, estudiante de trabajo social

Campaña de Verano 2024

Inicio
Actividades
Solicitudes

Calle Voluntariado 123
Mérida

Facebook
Twitter
Instagram

ACTIVIDADES



INICIO ACTIVIDADES SOLICITUD 

¡Bienvenidos a la Campaña de Verano 2024!
Estas son nuestras actividades



DEPORTE



ACUÁTICO



INTERNACIONAL



AVENTURA



COOPERACIÓN



OCIO

ACTIVIDAD DETALLE



INICIO ACTIVIDADES SOLICITUD 

Rompiendo barreras

Código de actividad: 33
Lugar: Badajoz
Tema: Internacional

Descripción: En la ciudad fronteriza de Badajoz llevaremos a cabo esta actividad de voluntariado con participantes de toda Europa. Ven y cambia el mundo que te rodea, ven y cambia tu manera de ver las cosas.

Fecha de inicio: 01-07-2024
Fecha de fin: 16-07-2024
Edad mínima: 18
Edad máxima: 30
Plazas: 10



SOLICITUD



INICIO ACTIVIDADES SOLICITUD 

¿Quieres comprobar el estado de tu solicitud?
[Consulta aquí](#)

Formulario de solicitud

DNI:

Código de actividad:

Nombre:

Primer apellido:

Segundo apellido:

Email:

Dirección completa:

Teléfono:

Fecha de nacimiento:

BUSCADOR SOLICITUD



INICIO ACTIVIDADES SOLICITUD 

Compruebe el estado de su solicitud

DNI a buscar

Identificador: 62
DNI: 89624478F
Código de la actividad seleccionada: 38
Nombre: Katniss
Primer apellido: Everdeen
Segundo apellido:
Email: arquera@hungergames.com
Dirección: C/ de la mineria, 89. Distrito 11
Teléfono: 647518454
Fecha de nacimiento: 14/06/1998
Número para el sorteo: No asignado
¿Está seleccionado? Sorteo no realizado
¿Pagado? No corresponde
Estado: Recibida

LOGIN



RESPONSIVE



INICIO ACTIVIDADES SOLICITUD

¡Bienvenidos a la Campaña de Verano 2024!

Los programas de voluntariado nacional ofrecen una alternativa única para disfrutar del verano mientras se realiza una contribución social importante. Estos programas atraen a jóvenes con intereses similares, tanto locales como de otras regiones, incluyendo participantes internacionales.

Las actividades de estos programas cubren una amplia gama de temas, tales como medio ambiente, conservación del patrimonio, arqueología, y apoyo a comunidades integración. Este enfoque multidisciplinario asegura que hay oportunidades para todos, independientemente de sus intereses específicos.

Estos campos se llevan a cabo principalmente durante los meses de verano, ofreciendo opciones en diversas localidades. La participación está abierta a jóvenes adultos, típicamente entre 18 y 30 años.

La cuota de participación generalmente incluye alojamiento, alimentación, materiales necesarios y actividades adicionales. Los participantes suelen ser responsables del transporte hacia y desde el lugar del programa. La organización del campo cubre seguros de responsabilidad civil y de accidentes.

"Una experiencia transformadora que combina mi amor por la biología con acciones prácticas de conservación. Todo ahora es diferente ¡inolvidable!"
- Julia, estudiante de biología

"Participar en la conservación del patrimonio me permitió conectar profundamente con la historia. Aprendí mucho y conocí gente maravillosa."
- Marcos, entusiasta de la historia

"Un verano dedicado a apoyar a comunidades necesitadas. Profundamente gratificante y educativo, reforzó mi pasión por el trabajo social."
- Ana, estudiante de trabajo social

Campaña de Verano 2024

- Inicio
- Actividades
- Solicitudes

Calle Voluntariado 123
Mérida

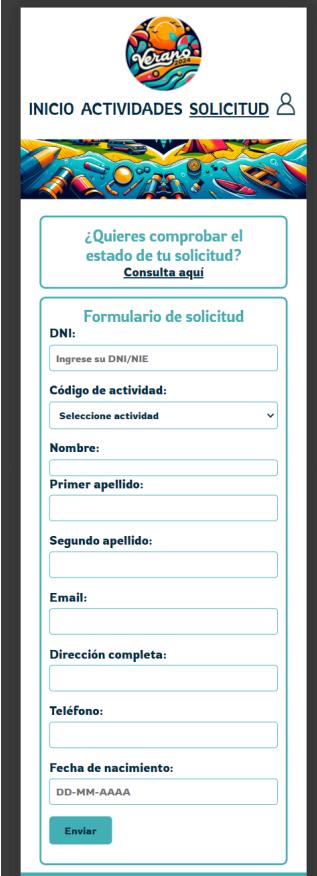
Facebook Twitter Instagram



INICIO ACTIVIDADES SOLICITUD

¡Bienvenidos a la Campaña de Verano 2024! Estas son nuestras actividades

DEPORTE	ACUÁTICO
INTERNACIONAL	AVENTURA
COOPERACIÓN	OCCIO



INICIO ACTIVIDADES SOLICITUD

¿Quieres comprobar el estado de tu solicitud?
[Consulta aquí](#)

Formulario de solicitud

DNI:

Código de actividad:

Nombre:

Primer apellido:

Segundo apellido:

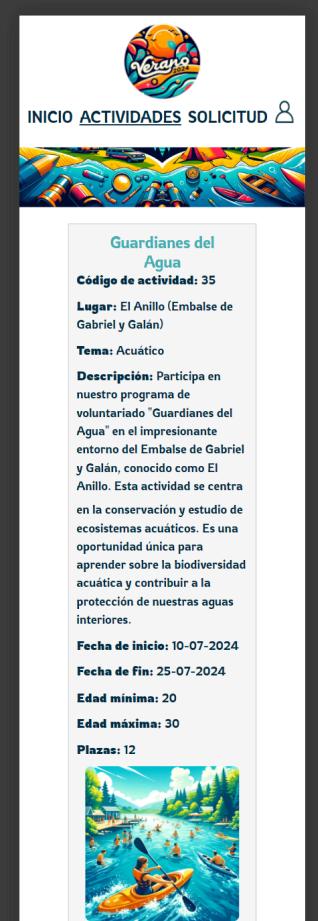
Email:

Dirección completa:

Teléfono:

Fecha de nacimiento:

Enviar



INICIO ACTIVIDADES SOLICITUD

Guardianes del Agua
Código de actividad: 35

Lugar: El Anillo (Embalse de Gabriel y Galán)

Tema: Acuático

Descripción: Participa en nuestro programa de voluntariado "Guardianes del Agua" en el impresionante entorno del Embalse de Gabriel y Galán, conocido como El Anillo. Esta actividad se centra en la conservación y estudio de ecosistemas acuáticos. Es una oportunidad única para aprender sobre la biodiversidad acuática y contribuir a la protección de nuestras aguas interiores.

Fecha de inicio: 10-07-2024

Fecha de fin: 25-07-2024

Edad mínima: 20

Edad máxima: 30

Plazas: 12





INICIO ACTIVIDADES SOLICITUD

Compruebe el estado de su solicitud

DNI a buscar

Buscar

Identificador: 64
DNI: 87758841A
Código de la actividad seleccionada: 38
Nombre: Cersei
Primer apellido: Lanister
Segundo apellido:
Email: cersei@hieloyfuego.com
Dirección: C/ del oro, 1. Roca Casterly
Teléfono: 698745879
Fecha de nacimiento: 01/01/2000
Número para el sorteo: No asignado
¿Está seleccionado? Sorteo no realizado
¿Pagado? No corresponde
Estado: Recibida

RESPONSIVE



PARTE INTERNA

[Usuarios](#)

[Actividades](#)

[Solicitudes](#)

[Procesos](#)

[Logout](#)

Bienvenido al panel de gestión de la aplicación

Este panel te permite administrar y gestionar diferentes aspectos de la aplicación. Desde aquí podrás realizar tareas como agregar, buscar, modificar y eliminar actividades, solicitudes y administradores. Explora las diferentes opciones del menú de gestión para acceder a las funcionalidades disponibles.

[Usuarios](#)

[Actividades](#)

[Solicitudes](#)

[Procesos](#)

[Logout](#)

Bienvenido al panel de procesos

[Asignar número](#) [Realizar sorteo](#)

NÚMEROS

SORTEO

Número aleatorio:

1.5 OBJETIVOS

Los objetivos básicos para un usuario externo en esta aplicación se resumen en:

- Difundir la información sobre las actividades ofertadas.
- Conseguir un entorno atractivo y accesible.
- Dar cabida a la recepción de todas las solicitudes, no importa el lugar o dispositivo desde el que se envíen.
- Que los usuarios conozcan, en todo momento, el estado de su solicitud.

Los objetivos básicos para un usuario interno en esta aplicación se resumen en:

- Añadir y gestionar las actividades
- Añadir y gestionar los Administradores
- Gestionar Solicitudes
- Control de los procesos

Además del objetivo principal se requiere un apartado más técnico como:

- Interfaz amigable para facilitar el uso por parte del administrador.
- Ahorro de costes: forma eficiente y económica de distribuir la información entre los clientes, sustituyendo los medios clásicos.
- El interfaz de la web será desarrollado para que soporte diferentes dispositivos.
- Fácil adaptación y configuración a la infraestructura tecnológica de la organización, así como gestión y manipulación de la información.
- Disponible en todas las plataformas informáticas.
- Sencilla integración de información multimedia.
- Utilización de estándares públicos y abiertos, independientes de empresas externas, como puede ser TCP/IP o HTML.

1.6 REQUISITOS

Parte pública

- Se podrá acceder a los contenidos informativos de la Campaña de verano (actividades, plazos, tasas, bases de participación, etc.)
- Se mostrarán las rr.ss del proyecto, en las cuales se podrá seguir el curso de las mismo.
- Añadir solicitudes
- Seguimiento del proceso de selección y estado de solicitud.

Parte privada

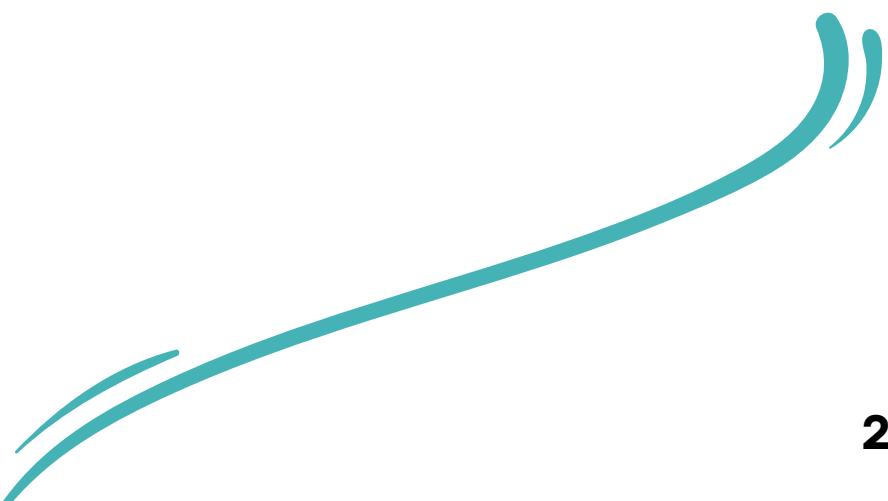
- Creación de administradores, modificación y eliminación.
- Creación de actividades, modificación y eliminación.
- Modificación y eliminación de solicitudes.
- Procesos del programa:
 - Asignación de número
 - Generar número aleatorio
- Modificación de los campos relativos a la selección y continuación del proceso del participante seleccionado

2. ANÁLISIS

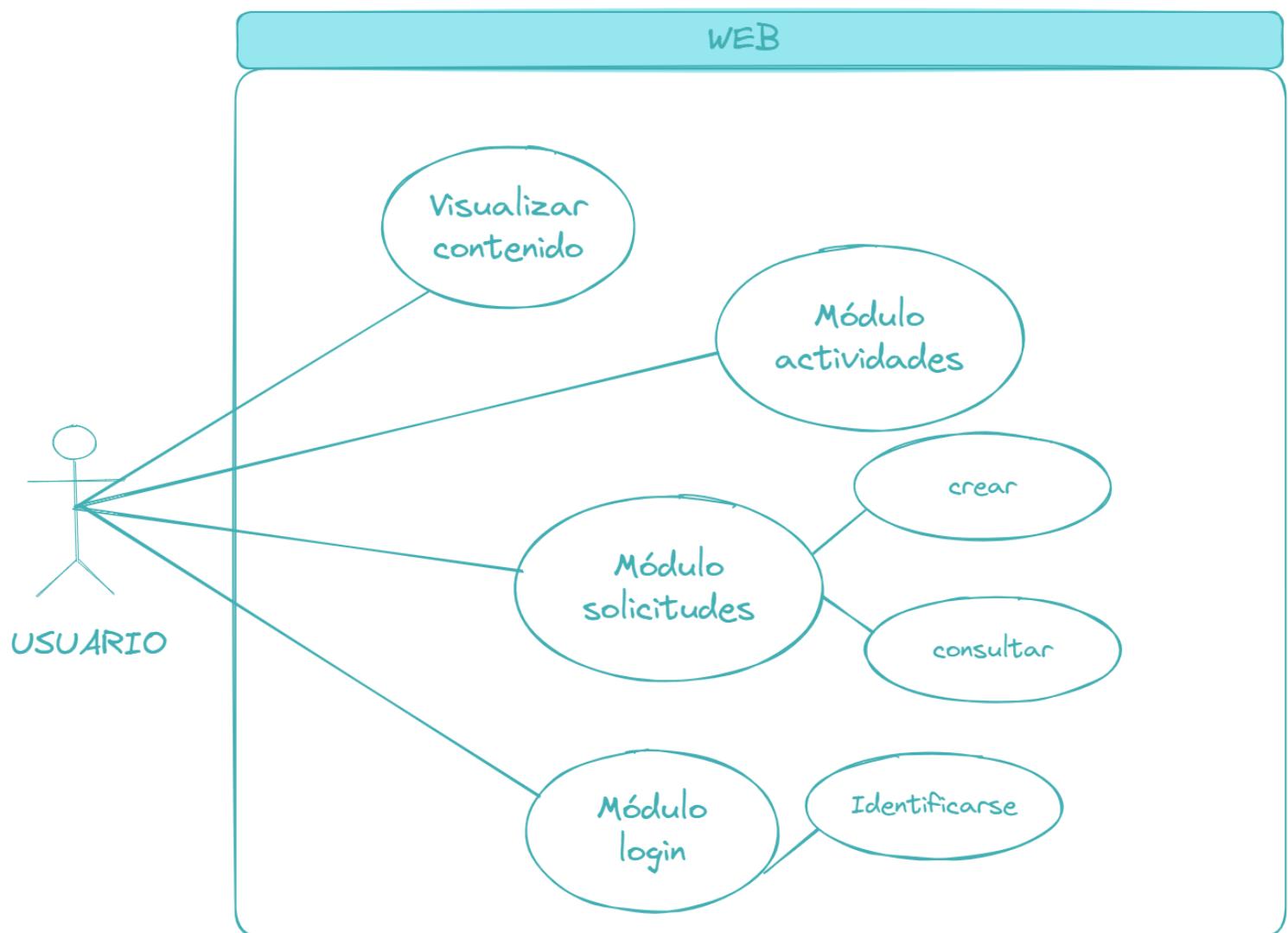
2.1 CASOS DE USO

En esta sección, abordaremos la fase de análisis del proyecto, detallando su estructura y funcionalidad mediante el uso de diagramas. Esto nos proporcionarán un modelo con los distintos actores que interactúan en nuestro proyecto, así como las relaciones y acciones que cada uno de ellos llevará a cabo, consiguiendo así una representación global de nuestro “sistema”

Cuando hablamos de casos de uso, nos referimos a descripciones detalladas de los pasos o actividades que deben ejecutarse para realizar un determinado proceso. Los personajes o entidades que participan en un caso de uso son denominados actores. En el contexto de la ingeniería de software, un caso de uso representa una secuencia de interacciones entre un sistema y sus actores. En resumen, los casos de uso nos permiten comprender cómo los usuarios interactúan con el sistema y qué acciones pueden realizar en respuesta a eventos específicos.



2.1 CASOS DE USO USUARIO EXTERNO



2.1 CASOS DE USO USUARIO EXTERNO

Nombre: Visualizar contenido

ID: CUE-01

Descripción: Navegación a través del portal para obtención de información

Actores: Usuario anónimo

Precondiciones: Ninguna

Curso normal del caso de uso: Usuario externo que quiere informarse sobre el programa, entra a través de la URL y localiza la información publicada

Nombre: Modulo actividades

ID: CUE-02

Descripción: Acceso al *apartado actividades*

Actores: Usuario anónimo

Precondiciones: Estar dentro del sitio

Curso normal del caso de uso: El usuario externo accederá a este apartado para conocer en detalle las actividades ofertadas y elegir entre las que puede participar, dependiendo de los requisitos y sus preferencias.

2.1 CASOS DE USO.

USUARIO EXTERNO

Nombre: Módulo solicitudes. Crear
ID: CUE-03

Descripción: Acceso al *apartado solicitud*

Actores: Usuario anónimo

Precondiciones: Acceder al apartado conociendo en qué actividad quiere participar. Disponer de los datos personales necesarios para la solicitud

Curso normal del caso de uso:

1. Usuario externo que ya ha decidido participar en el proceso y ha elegido una actividad en la que participar.
2. Rellenará los datos que se le piden en el formulario
3. Envío de los datos. Comprobación de que no hay una solicitud anterior con ese DNI.
4. Visualización de la solicitud enviada. Ya está recibida por el sistema

2.1 CASOS DE USO.

USUARIO EXTERNO

Nombre: Módulo solicitudes. Consultar

ID: CUE-04

Descripción: Acceso al *apartado buscador de solicitudes*

Actores: Usuario anónimo

Precondiciones: Acceder al apartado disponiendo de un DNI a consultar

Curso normal del caso de uso:

1. El usuario externo accederá a este apartado para conocer el estado de su solicitud
2. Introducirá el DNI en la casilla habilitada para ello y buscará.
3. En caso de coincidencia, se mostrarán los datos y el estado de esa solicitud.
4. En caso de no hallar resultado, se avisará de tal hecho.

2.1 CASOS DE USO.

USUARIO EXTERNO

Nombre: Módulo login. Identificarse

ID: CUE-05

Descripción: Acceso al *apartado login*.

No existe creación de cuentas. Solo se pueden generar cuentas de manera interna, por lo que un administrador generará otro administrador.

Campos requeridos **usuario y contraseña**.

La contraseña se almacena en el sistema de manera cifrada.

Esta parte privada solo es accesible para usuarios identificados ya que la **carpeta admin**, donde están todas las páginas que gestionan los administradores, tiene un **filtro para permitir el acceso** solo a los usuarios registrados.

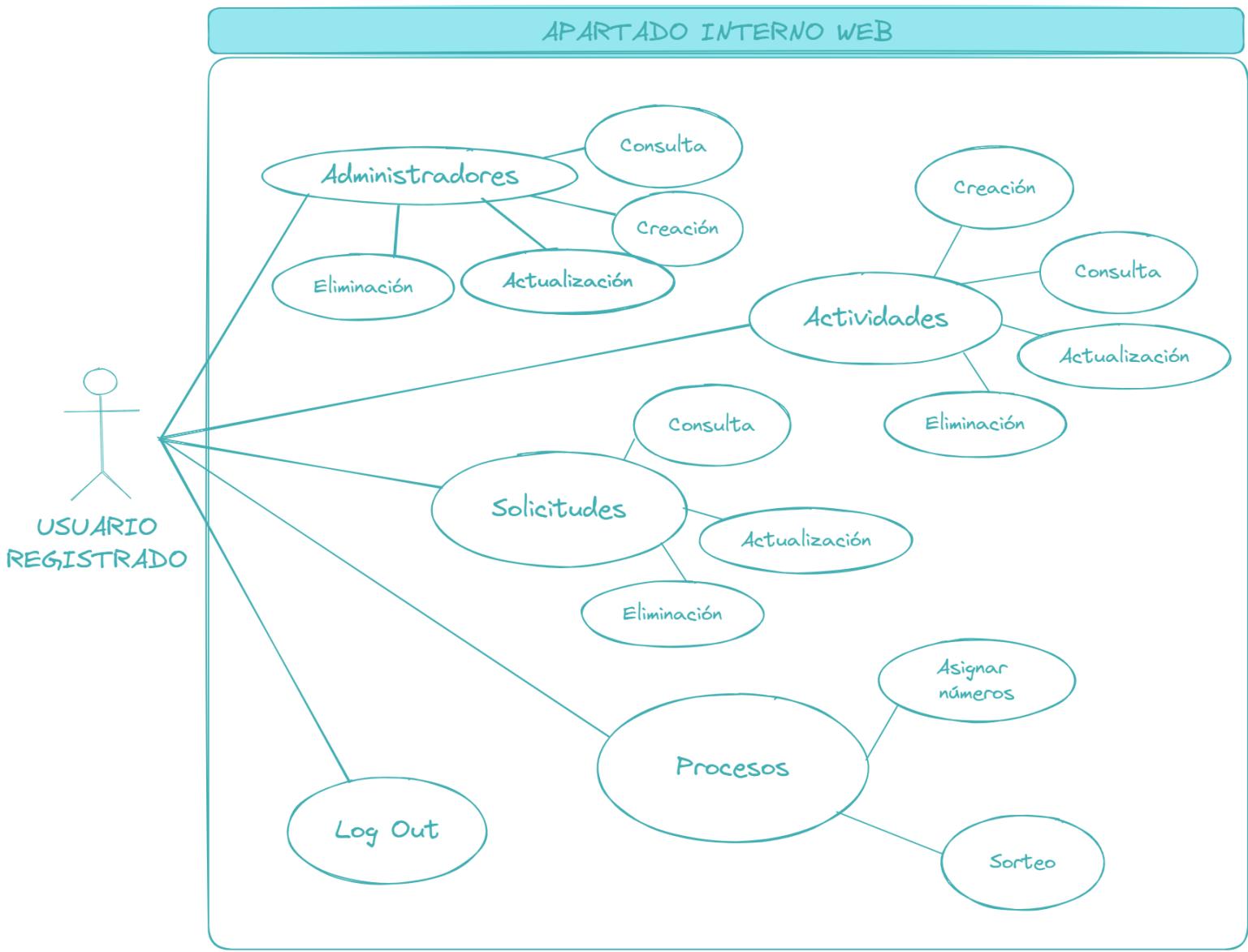
Actores: Usuario anónimo

Precondiciones: Acceder al apartado login y disponer de cuenta administrador.

Curso normal del caso de uso:

1. El usuario administrador accederá a este apartado con las credenciales que le hayan facilitado.
2. Si estas credenciales son correctas, accederá a la parte privada
3. Si estas credenciales son erróneas aparecerá un mensaje en cada una de las casillas informando de ello.
4. En la parte privada podrá cambiar su contraseña
5. En la parte privada dispondrá de todos los permisos para la gestión del programa en su totalidad.

2.1 CASOS DE USO USUARIO REGISTRADO



2.1 CASOS DE USO

USUARIO REGISTRADO

Nombre: Administradores. Creación

ID: CUR-01

Descripción: Acceso al *apartado Usuarios/Añadir*.

Desde este apartado podemos crear un nuevo usuario de acceso al sistema.

Se debe proporcionar usuario único y contraseña.

Se comprobará que no haya un usuario ya existente con ese nombre.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al *apartado Usuarios/Añadir*.

Curso normal del caso de uso:

1. Se facilitará un nombre de usuario único.
2. Se facilitará una contraseña.

2.1 CASOS DE USO

USUARIO REGISTRADO

Nombre: Administradores. Consulta - Actualización - Eliminación
ID: CUR-02

Descripción: Acceso al *apartado Usuarios/Gestión*

Al acceder a este apartado listará los usuarios dados de alta en el sistema. Al final de la tabla se nos presenta los enlaces para editar y borrar un usuario mediante su **id**.

Si accedemos a **Editar** se nos mostrará un formulario con los datos del administrador seleccionado, desde ahí podemos modificar los datos que queramos y guardar los cambios enviando de nuevo.

Con el enlace **Borrar** eliminaremos ese registro directamente.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder a el *apartado Usuarios/Gestión*

Curso normal del caso de uso:

1. Se listará una tabla con todas los usuarios dados de alta en la aplicación.
2. Se seleccionará el usuario administrador a editar o borrar
3. **Editar.** Se modificará los datos y se enviará de nuevo a la base de datos.
4. **Borrar.** Eliminaremos de forma permanente el registro.
5. Volveremos al listado de actividades

2.1 CASOS DE USO

USUARIO REGISTRADO

Nombre: Actividades. Creación

ID: CUR-03

Descripción: Acceso al apartado *Actividades/Añadir*.

Desde este apartado podemos crear una nueva actividad.

En el momento que es añadida pasa a estar disponible para seleccionar desde el **formulario Solicitud** accesible a los usuarios externos.

Habrá que cumplimentar los datos básicos de la actividad, como otros no obligatorios para hacer la actividad más atractiva, como son imagen o descripción.

El formulario tiene una función **verificar** para el correcto funcionamiento. Se comprueba que la *fecha fin* sea posterior a la *fecha de inicio*, o que la *edad máxima* sea mayor a la *edad mínima*, ambas entre 18 y 30 años. Se alerta de que ciertos campos están vacíos.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al apartado *Actividades/Añadir*.

Curso normal del caso de uso:

1. Se llenarán todos los datos que se piden en el formulario.
2. Descripción e imagen no son obligatorios, se alertará antes del envío del formulario si no hay contenido en estos campos.
3. El resto son campos obligatorios, con lo cual son imprescindibles para continuar con el envío.
4. El campo **tema** será mostrado en un desplegable con las opciones posibles. Opciones controladas desde el front y en la base de datos

2.1 CASOS DE USO USUARIO REGISTRADO

Nombre: Actividades. Consulta - Actualización - Eliminación

ID: CUR-04

Descripción: Acceso al apartado *Actividades/Gestión*.

Al acceder a este apartado se nos presentará una lista con todas las actividades almacenadas. Desde esta tabla podemos visualizar todos los atributos de cada actividad.

Al final de cada fila se nos presenta los enlaces **Editar** y **Borrar** específicos de cada registro. Controlado por **cod_actividad** que es el identificador.

Si accedemos a **Editar** se nos mostrará un formulario con los datos de la actividad seleccionada, desde ahí podemos modificar los datos que queramos y guardar los cambios enviando de nuevo.

Con el enlace **Borrar** eliminaremos ese registro directamente. Aviso de borrado permanente. Este botón será funcional siempre y cuando se mantenga la **persistencia** en la base de datos, por ejemplo, no haya una solicitud que haya elegido esa actividad.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al apartado *Actividades/Gestión*.

Curso normal del caso de uso:

1. Se listará una tabla con todas las actividades incluidas en la base de datos
2. Se seleccionará la actividad a editar o borrar
3. **Editar.** Se modificará los datos y se enviará de nuevo a la base de datos.
4. **Borrar.** Eliminaremos de forma permanente el registro. Siempre y cuando no afecte a la persistencia de la base de datos.
5. Volveremos al listado de actividades

2.1 CASOS DE USO USUARIO REGISTRADO

Nombre: Solicitudes. Consulta - Actualización - Eliminación

ID: CUR-05

Descripción: Acceso al apartado *Solicitudes/Gestión*.

Al acceder a este apartado se nos presentará una lista con todas las solicitudes recibidas. Desde esta tabla podemos visualizar todos los datos de cada solicitud.

Al final de cada fila se nos presenta los enlaces **Editar** y **Borrar** específicos de cada registro. Controlado por **id** que es el identificador.

Si accedemos a **Editar** se nos mostrará un formulario con los datos de la solicitud seleccionada, desde ahí podemos modificar los datos que queramos y guardar los cambios enviando de nuevo.

Con el enlace **Borrar** eliminaremos ese registro directamente.

Desde aquí también podrán los administradores gestionar si es un participante seleccionado, si está pagado, observaciones y estado, campos solo modificables por el administrador pero visibles por usuarios externos.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al apartado *Solicitudes/Gestión*.

Curso normal del caso de uso:

1. Se listará una tabla con todas las solicitudes recibidas en la base de datos
2. Se seleccionará la solicitud a editar o borrar
3. **Editar.** Se modificará los datos y se enviará de nuevo a la base de datos.
4. **Borrar.** Eliminaremos de forma permanente el registro.
5. Volveremos al listado de solicitudes.

2.1 CASOS DE USO

USUARIO REGISTRADO

Nombre: Procesos. Asignar número

ID: CUR-06

Descripción: Acceso al *apartado Procesos*.

Al acceder a este apartado se nos presentarán la lista de procesos disponibles. Serán accesibles pinchando en la imagen.

Necesitamos este proceso ya que para participar en el sorteo de plazas debemos hacerlo con un número. Debido a que hay solicitudes que pueden ser eliminadas por incumplimiento de los requisitos, no podemos utilizar el **id** de la solicitud y necesitamos asignarle un número una vez revisadas todas las solicitudes recibidas.

Este proceso recoge el total de las solicitudes, las ordena por **id** y les asigna un número creciente según concurrencia.

Este número será visible para los usuarios antes del sorteo.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al *apartado Procesos/Asignar número*

Curso normal del caso de uso:

1. En el apartado *Procesos* aparece este proceso **Asignar número**
2. Pinchar en la imagen para activar el proceso
3. Redirige al listado de solicitudes ya con el número asignado a cada solicitud

2.1 CASOS DE USO

USUARIO REGISTRADO

Nombre: Procesos. Sorteo (Realizar sorteo)

ID: CUR-07

Descripción: Acceso al *apartado Procesos*.

Al acceder a este apartado se nos presentarán la lista de procesos disponibles. Serán accesibles pinchando en la imagen.

Necesitamos este proceso para generar un número aleatorio a partir del cual se ordenan las solicitudes recibidas y se asignan las plazas.

El número aleatorio será generado desde el 1 hasta el número máximo de solicitudes aptas, el cual será indicado por el administrador.

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al *apartado Procesos/Realizar sorteo*

Curso normal del caso de uso:

1. En el apartado *Procesos* aparece este proceso **Realizar sorteo**
2. Pinchar en la imagen para activar el proceso
3. Nos pide el número máximo, es decir, el número de solicitudes aptas
4. Visualizaremos el número en pantalla

2.1 CASOS DE USO USUARIO REGISTRADO

Nombre: LogOut

ID: CUR-08

Descripción: Acceso al *apartado Logout*.

Al acceder a este apartado cerraremos la sesión de nuestro usuario

Actores: Usuario registrado

Precondiciones: Acceder con privilegios de administrador. Acceder al *apartado Logout*

Curso normal del caso de uso:

1. Acceder al menú Logout
2. Cerraremos la sesión activa
3. Redirección a página externa **index.html**

2.2 DIAGRAMA DE CLASE

Los diagramas de clases son fundamentales en el desarrollo de software, especialmente en las fases de análisis y diseño, ya que permiten visualizar la estructura del sistema. Estos diagramas ofrecen una representación gráfica de las clases, sus atributos y cómo interactúan entre sí, facilitando la comprensión de las relaciones y comunicaciones internas del sistema. Además, son cruciales para diseñar conceptualmente los componentes del sistema y su funcionamiento.

En la representación que veremos a continuación, presentamos el Diagrama UML de nuestro sistema, incluyendo nuestras clases principales. Es un esquema básico que proporciona una visión clara de cómo las entidades se conectan y gestionan a través de sus DAOs y cómo se interrelacionan dentro del sistema. (Gráfico 1)

En el segundo gráfico, mostraremos la clase **Actividad** al detalle.

1. Clase Modelo:

- **Actividad:** Representa la clase actividad dentro de la aplicación. Detallamos las variables y el método `dameJson` para obtener su representación en formato JSON.

2. Clase de Acceso a Datos (DAO):

- **DaoActividad:** Encargada de interactuar con la base de datos para ejecutar operaciones **CRUD** (Crear, Leer, Actualizar, Eliminar) relacionadas con la clase actividad. Utiliza una conexión proporcionada por la clase **DBConexion**.

3. Clase de Conexión:

- **DBConexion:** Utiliza el patrón **Singleton** para asegurar una única instancia de conexión a la base de datos. Proporciona la conexión a **DaoActividad** para realizar operaciones de base de datos.

4. Servlets:

- **BuscarActividad, GestionActividad, ListarActividad y SelectActividad.** Todos interactúan con **DaoActividad** para realizar sus tareas.

El diagrama refleja cómo los datos de las actividades son gestionados dentro de la aplicación, pasando por la lógica de la clase modelo, la interacción con la base de datos a través de DAOs y la interfaz de usuario mediante servlets. (Gráfico 2)

GRÁFICO 1

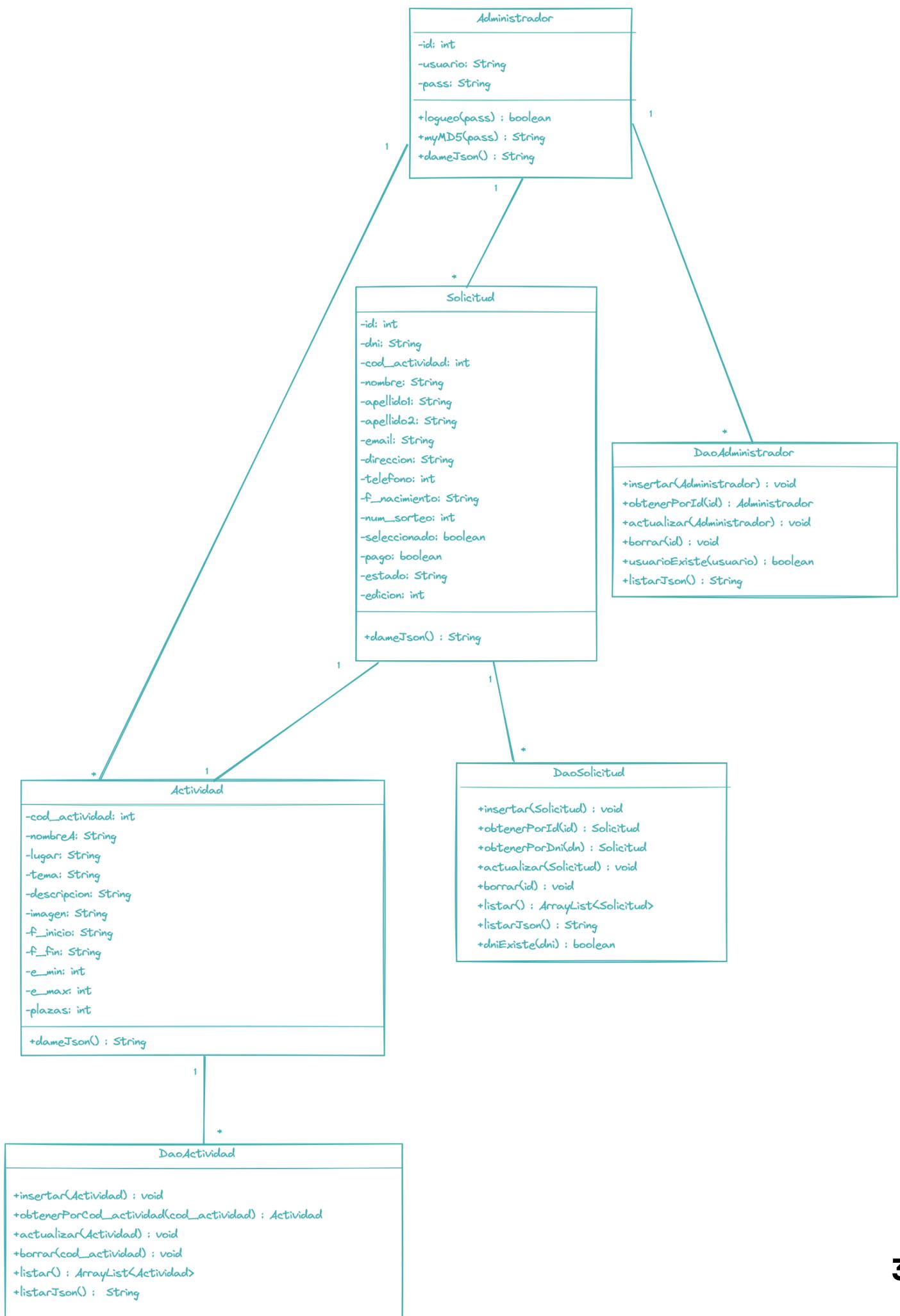
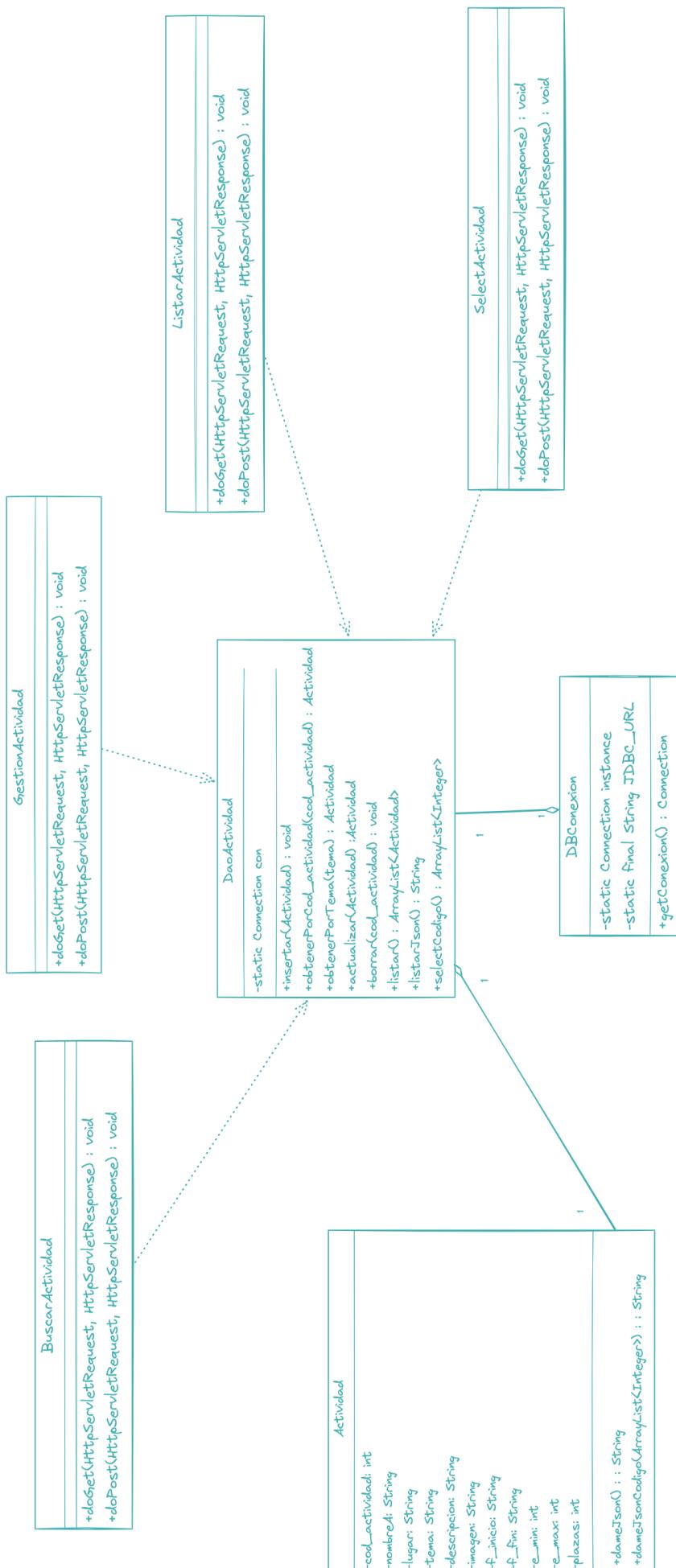


GRÁFICO 2



2.3 DIAGRAMA ENTIDAD - RELACIÓN

Un diagrama de entidad-relación (ER) es una herramienta fundamental en el diseño de bases de datos que ilustra la estructura lógica de nuestra base de datos. Detalla cómo las entidades están interrelacionadas dentro del sistema. (Gráfico 3)

Contexto:

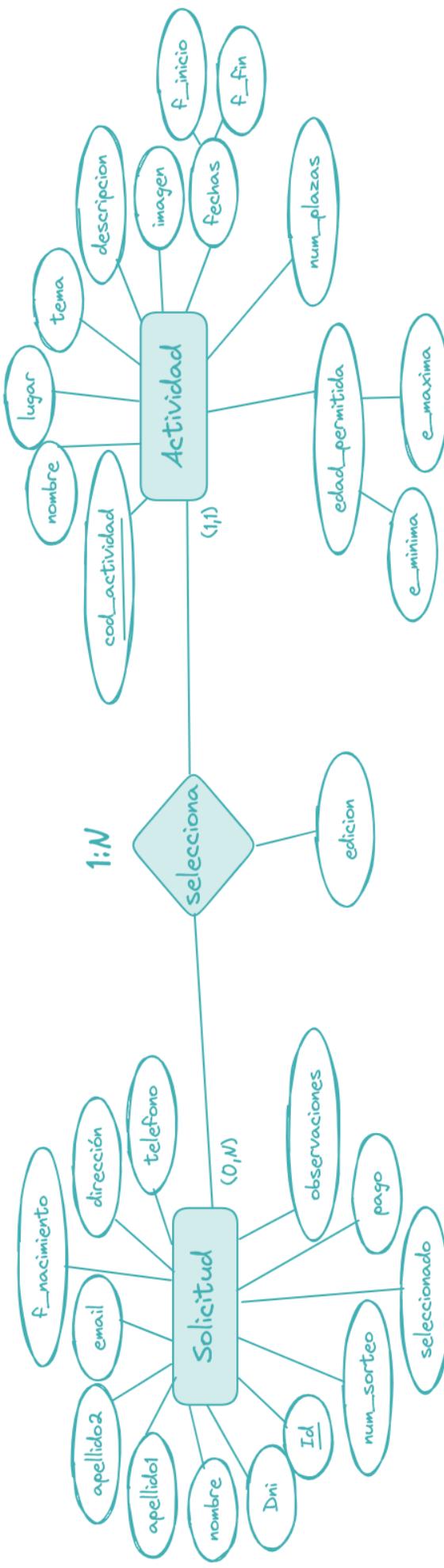
- El usuario externo accederá a la web y rellenará con sus datos personales el formulario **Solicitud**.
- Dependerá de que actividad sea seleccionada(diferentes actividades por año) para tomar el valor **edición** por ello pertenece a la relación.
- Dentro de la solicitud se seleccionará una actividad mediante su **cod_actividad**.

Actividad tendrá los atributos necesarios para identificarla únicamente.

Vemos también representada la **tabla admin**, entidad aparte que si necesitan una tabla para la gestión del **login, con su id, usuario y pass**, el cual se almacenará de manera cifrada, pero que no tienen relación con las tablas anteriores.(Gráfico 3 carpeta **Material de referencia**)

Encontraremos más información sobre el paso a tablas, construcción de la base de datos y temas relacionados en próximos epígrafes de la presente memoria.

GRÁFICO 3

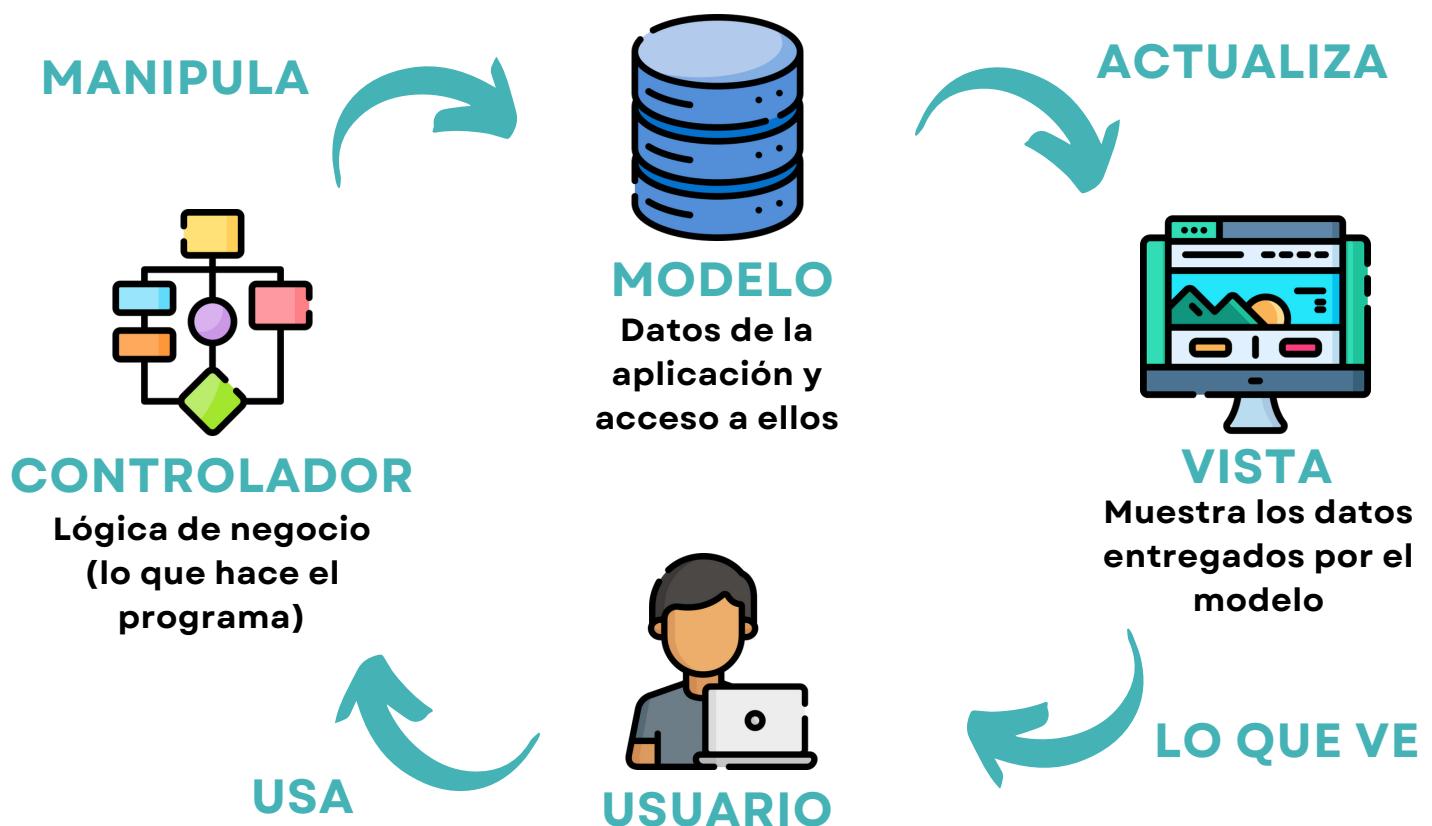


3. DISEÑO

3.1 ARQUITECTURA

Arquitectura cliente/servidor es un sistema en el que los servicios y recursos se centralizan en un servidor, el cual los distribuye a dispositivos clientes que los solicitan. Estos clientes pueden ser dispositivos como ordenadores o móviles que se conectan al servidor para realizar diversas funciones.

Una vez establecida la arquitectura, diseñamos el modelo de desarrollo de la aplicación usando el patrón **Modelo-Vista-Controlador**. Este enfoque divide la aplicación en tres componentes interconectados: la capa de presentación para gestionar las interacciones del usuario, la capa de lógica de negocio para procesar la información y la capa de datos que asegura la persistencia de los datos de la aplicación. Este diseño permite una clara separación de responsabilidades, facilitando el mantenimiento y la escalabilidad del sistema.



3. DISEÑO

3.2 CAPA DE PRESENTACIÓN

Front-end: Es la interfaz a través de la cual los usuarios interactúan con la aplicación. Esta capa incluye todo lo que el usuario puede ver y manipular, como páginas web, elementos de diseño y procesos interactivos. Su principal objetivo es proporcionar una experiencia de usuario clara y eficiente.

Este portal ha sido creado utilizando las últimas tecnologías y estándares web como **HTML5, CSS3, y JavaScript**.

Para su diseño se ha tenido en cuenta los diferentes dispositivos que actualmente usamos, por ello es un diseño *responsive*. Diseño que garantiza su perfecta visualización en los diferentes tamaños de dispositivos que a día de hoy utilizamos.

Se realiza validación de formularios por el lado del cliente utilizando JavaScript. Lo utilizaremos para prevenir errores involuntarios como alteración de datos. También ciertos controles en el back para evitar alteraciones en el proceso, p.e. añadir dos solicitudes con el mismo DNI.

Dentro del diseño se han incluido diversos efectos para presentar nuestra web de una manera más atractiva al usuario, entre los cuales podemos destacar; ventanas emergentes informativas, sombreados, iluminación de bordes, bordes redondeados o la inclusión de favicon entre otros elementos.



AVENTURA



COOPERACIÓN



OCIO

Ejemplo de sombreado y como se resalta un objeto en nuestra web.

3. DISEÑO

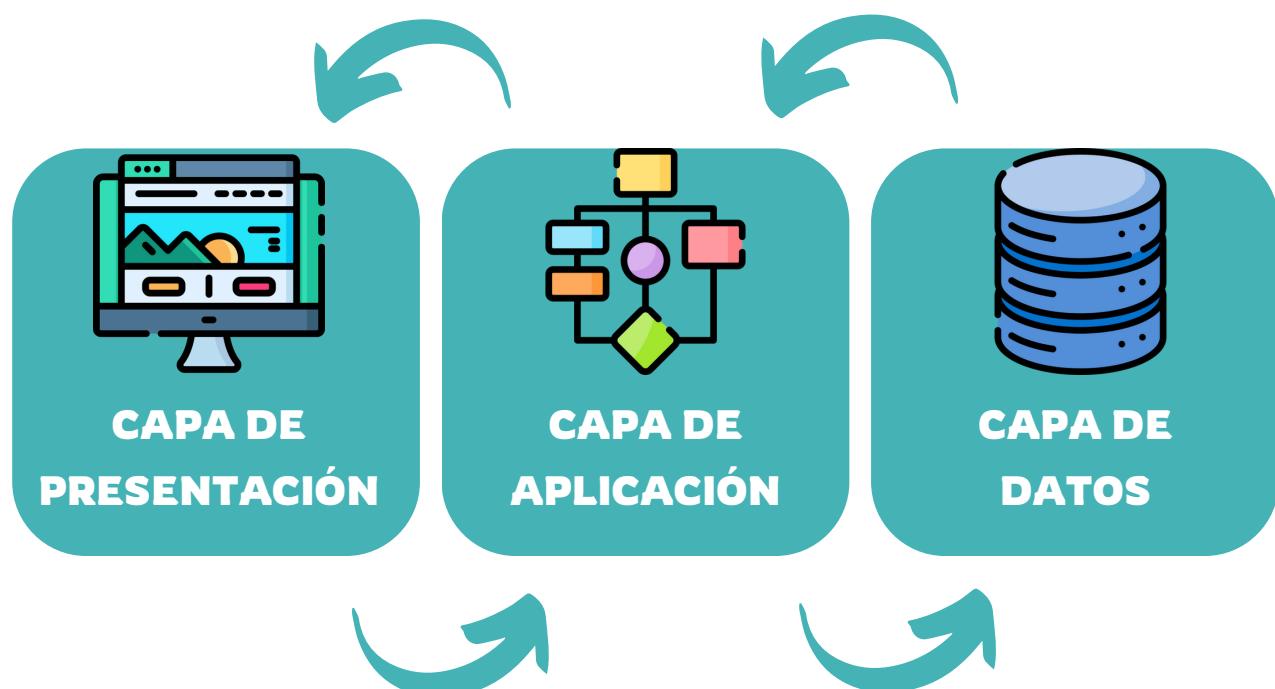
3.3 CAPA DE APLICACIÓN

Back-end: Actúa como el motor de procesamiento de la aplicación, donde se ejecuta la lógica de negocio. Esta capa maneja las solicitudes del usuario, procesa datos y toma decisiones basadas en las reglas de negocio definidas. También se encarga de las interacciones con la base de datos a través de la capa de datos.

Haciendo otra vez repaso de lo que supone **MVC** en nuestra aplicación sería;

1. El usuario accede a nuestra web e interactúa. P.e. rellena el formulario de solicitud. Formulario desarrollado con HTML, CSS y es enviado gracias a la programación en JavaScript.
2. Esos datos son recibidos por un Servlet, controlador Java
3. Junto con diversos métodos como el de *añadir* o *dniExiste* crea y controla la instancia a añadir. Utilizaremos Java
4. Siempre pasando por la capa DAO, que gestiona las conexiones con la base de datos, en cuanto a envío y recepción. En nuestro ejemplo, insertaremos la instancia en nuestra base de datos MySQL.
Utilizaremos Java y SQL para las consultas.

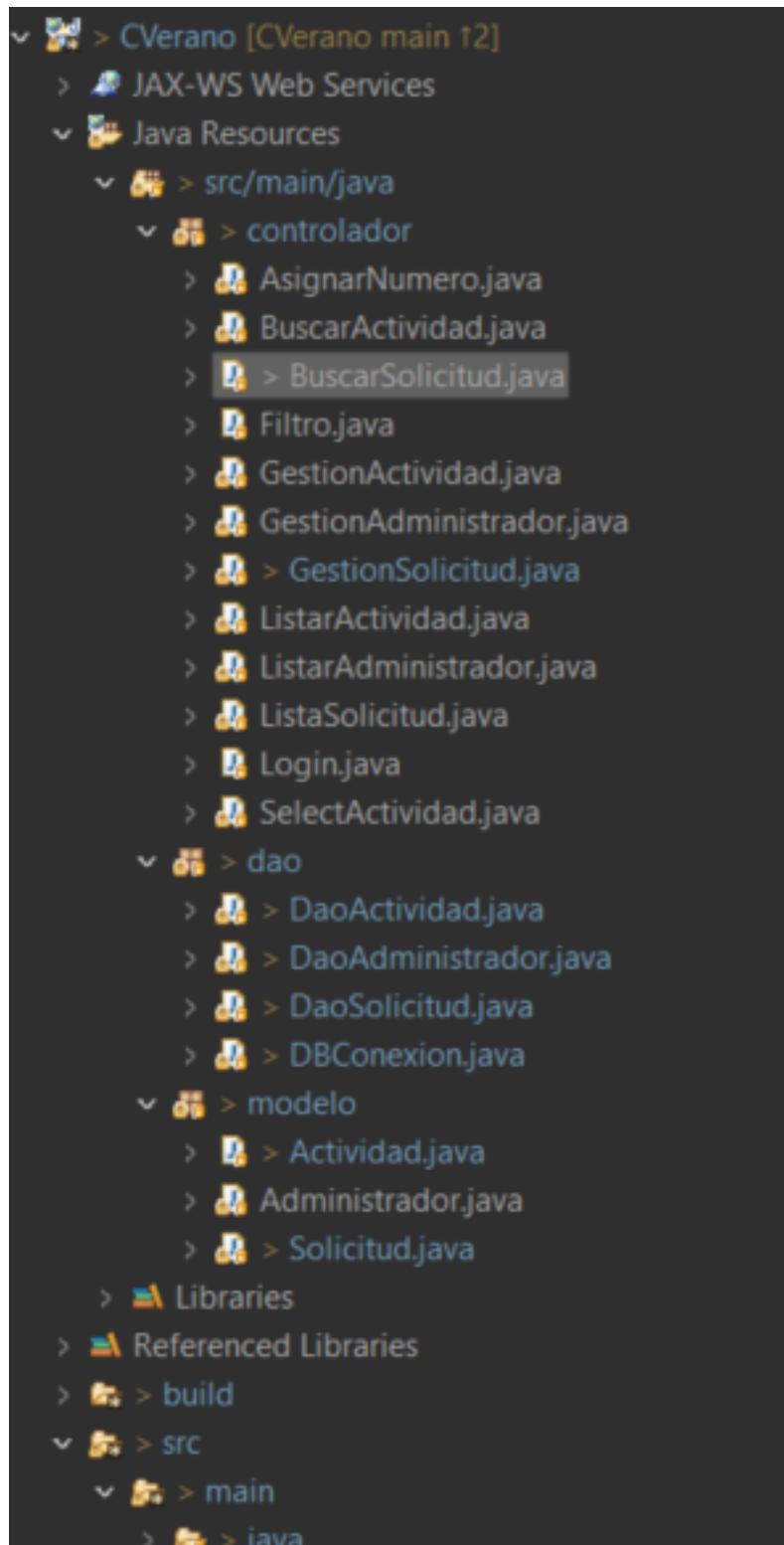
Esta podría ser la representación del viaje de ida de nuestro datos desde la vista del usuario, hasta su almacenamiento en la base de datos. Hay operaciones que también incluye el “*viaje de vuelta*” p.ej. en nuestro proyecto la consulta en el buscador por DNI. Busca un DNI y devuelve todo el registro asociado.



3. DISEÑO

3.3 CAPA DE APLICACIÓN

Aunque ya conocemos la estructura de nuestro proyecto gracias al diagrama UML, a continuación mostraremos la estructura resultante en nuestro IDE Eclipse



3. DISEÑO

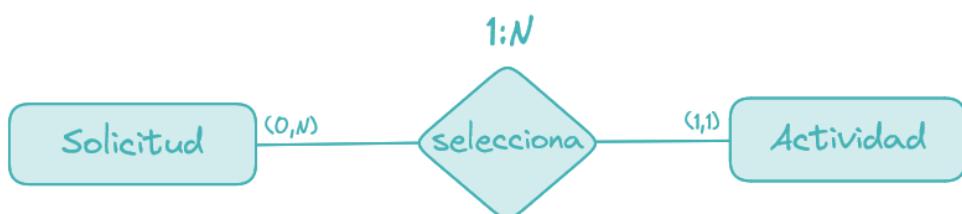
3.4 CAPA DE DATOS

Es donde se almacenan y gestionan todos los datos de la aplicación. Esta capa asegura que los datos estén disponibles de forma segura y eficiente para ser utilizados por la capa de aplicación.

Para nuestra aplicación utilizamos **MySQL** y **MySQL Workbench** para su gestión.

Anteriormente ya hemos mostrado el modelo entidad-relación, ahora vamos a dar más información sobre el desarrollo.

En cualquier modelo relacional una etapa fundamental es el paso a tablas, como un Diagrama Entidad-Relación es convertido a unas tablas. Para ello nos ayudamos de las *12 reglas de Codd*, que son un conjunto de principios para guiar en el desarrollo de los sistemas de bases de datos y asegurar que estos sistemas son relacionales.



Relación **Solicitud--selecciona--Actividad 1:N**

La relación no genera tabla, pero si se propaga la clave de quien tenga la cardinalidad máxima, en este caso **Actividad**.

El atributo de la relación también se propaga de igual manera.

Se puede consultar el gráfico completo en la página X o en la carpeta **Material de referencia/Imagenes grafico3_ER.png**

Con ello el resultado del paso a tablas sería el siguiente:

SOLICITUD (**Id, cod_actividad**, dni, nombre, apellido1, apellido2, email, dirección, teléfono, f_nacimiento, num_sorteo, seleccionado, pago, observaciones, edición)

cod_actividad (foreign key) por ser primary key de ACTIVIDAD

ACTIVIDAD (**cod_actividad**, nombre, lugar, tema, descripción, imagen, f_inicio, f_fin, num_plazas, e_minima, e_maxima)

4. IMPLEMENTACIÓN

4.1 TECNOLOGÍAS

HTML5

Hypertext Markup Language

Lenguaje de marcado utilizado para estructurar y presentar contenido en la WWW. Esta versión fue finalizada en octubre de 2014.

HTML5 introduce una serie de nuevas etiquetas y atributos que permiten un mayor alcance para incorporar multimedia y aplicaciones gráficas.

HTML es el componente más básico de la web, utilizamos sus etiquetas para incluir el contenido y darle estructura y diseño junto con CSS. Es uno de los componentes básicos para la creación web.

HTML



CSS

Cascading Style Sheets

Principal responsable de diseño y apariencia de la vista. Es un lenguaje de diseño gráfico para definir y crear la presentación de un documento HTML o XML. CSS maneja el diseño y el estilo de las páginas web, incluyendo colores, fuentes y la disposición de los elementos, permitiendo que los desarrolladores separen el contenido de su presentación visual. Esto facilita el mantenimiento y mejora la accesibilidad.

CSS



4. IMPLEMENTACIÓN

4.1 TECNOLOGÍAS

JAVASCRIPT

Lenguaje de programación dinámico ampliamente utilizado que es esencial para desarrollar páginas web interactivas. A menudo abreviado como JS, este lenguaje permite a los desarrolladores implementar funciones complejas en páginas web, desde mostrar contenido actualizado sin necesidad de recargar una página (conocido como AJAX), hasta animaciones y manejo de eventos del usuario.

JavaScript



JAVA

Lenguaje de programación orientado a objetos, diseñado para ser portátil y accesible en diversas plataformas de software y hardware.

Fue desarrollado por Sun Microsystems en 1995 y es conocido por su filosofía de "*escribir una vez, ejecutar en cualquier lugar*", lo que significa que el código compilado de Java (bytecode) puede ejecutarse en cualquier dispositivo que tenga instalada la máquina virtual de Java (JVM).

Esto lo hace extremadamente popular para desarrollar aplicaciones Android, aplicaciones empresariales, software de servidor, videojuegos, y sistemas embebidos. Java se destaca por su robustez, seguridad y capacidad de manejo de grandes volúmenes de datos, siendo una elección común para aplicaciones bancarias y financieras.



4. IMPLEMENTACIÓN

4.1 TECNOLOGÍAS

MySQL

Sistema de gestión de bases de datos relacional (RDBMS) de código abierto, ampliamente utilizado para crear y mantener bases de datos. Fue desarrollado originalmente por una compañía sueca llamada MySQL AB, la cual fue adquirida por Oracle Corporation en 2010.



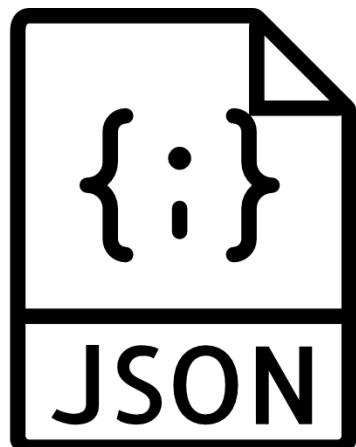
MySQL utiliza SQL (Structured Query Language) para acceder y gestionar los datos almacenados, ofreciendo un rendimiento eficiente y la capacidad de manejar grandes cantidades de datos.

Es popular en aplicaciones web y es una parte esencial de la pila LAMP/XAMP, utilizada para el desarrollo de sitios web y aplicaciones web. MySQL es conocido por su facilidad de uso, escalabilidad y compatibilidad con una amplia gama de plataformas de hosting. También es frecuentemente utilizado en combinación con PHP para crear sitios web dinámicos.

CSS

JavaScript Object Notation

Formato ligero de intercambio de datos. Es fácil de leer y escribir para humanos y simple de parsear y generar para máquinas. JSON se basa en un subconjunto del lenguaje de programación JavaScript y se utiliza principalmente para transmitir datos entre un servidor y una aplicación web como una alternativa a XML.



La simplicidad de JSON ha resultado en su adopción generalizada en la API web y los servicios de configuración de aplicaciones.

4. IMPLEMENTACIÓN

4.2 HERRAMIENTAS

ECLIPSE JAVA WEB DEVELOPER

Entorno de desarrollo integrado (IDE) ampliamente utilizado para programación en Java, aunque también soporta otros lenguajes mediante plugins.



JAVADOC

Herramienta de documentación para Java que genera documentación en formato HTML a partir de comentarios en el código fuente.

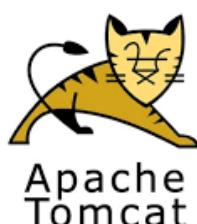
JUNIT

Framework de pruebas unitarias para el lenguaje de programación Java, que ayuda a escribir y ejecutar pruebas repetibles de manera fácil y eficiente.



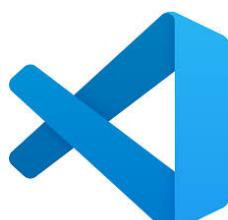
APACHE TOMCAT

Servidor web de código abierto y contenedor de servlets que facilita el despliegue y manejo de aplicaciones web Java



VISUAL STUDIO CODE

Editor de código fuente que soporta múltiples lenguajes de programación y cuenta con una gran cantidad de extensiones disponibles, lo que lo hace altamente personalizable y eficiente para el desarrollo de software.



4. IMPLEMENTACIÓN

4.2 HERRAMIENTAS

MYSQL WORKBENCH

Herramienta visual de diseño de bases de datos y gestión desarrollada por Oracle. Permite a los usuarios crear, modelar, administrar y mantener bases de datos MySQL de forma gráfica, así como desarrollar consultas SQL y configurar servidores MySQL.



EXCALIDRAW

Herramienta de dibujo en línea que simula el estilo de dibujo a mano alzada. Es excelente para crear diagramas, wireframes y esquemas rápidos de forma intuitiva. Ofrece la posibilidad de importar distintas bibliotecas para disponer de los recursos que se pueden necesitar para los diferentes diagramas.



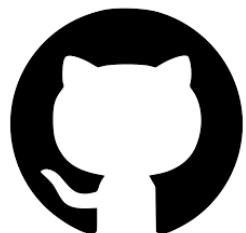
GITKRAKEN

Herramienta visual para manejar Git que facilita la gestión de repositorios con una interfaz gráfica intuitiva, ideal para visualizar historiales, gestionar ramas y resolver conflictos de manera eficiente.



GITHUB

Plataforma de alojamiento de código que utiliza Git para el control de versiones, permitiendo a los desarrolladores almacenar y gestionar sus proyectos, colaborar con otros y rastrear y controlar cambios en cualquier conjunto de archivos.



4. IMPLEMENTACIÓN

4.2 HERRAMIENTAS

XAMP

Paquete de software que facilita la instalación y uso de Apache, MySQL, PHP y Perl, proporcionando un entorno de servidor local completo para el desarrollo y prueba de aplicaciones web.



VIRTUALBOX

Software de virtualización de código abierto que permite ejecutar múltiples sistemas operativos simultáneamente en una misma máquina física, facilitando la creación y gestión de máquinas virtuales.



UBUNTU

Sistema operativo basado en Linux. Es popular por su facilidad de uso y amplio soporte, siendo una de las distribuciones de Linux más utilizadas tanto para escritorio como para servidores.



CHROME

Cliente FTP gratuito y de código abierto que permite la transferencia de archivos entre un ordenador y un servidor a través de la red.



CHROME

Navegador web desarrollado por Google, conocido por su velocidad, simplicidad y eficiencia. Es ampliamente utilizado en diversas plataformas como Windows, macOS, Linux, Android e iOS.



4. IMPLEMENTACIÓN

4.2 HERRAMIENTAS

CANVA

Herramienta de diseño gráfico en línea que facilita la creación de contenido visual, como presentaciones, carteles, documentos y otros gráficos para medios digitales o impresos.



GIMP

Programa gratuito y de código abierto para la edición de imágenes, que ofrece herramientas avanzadas para manipular gráficos y fotografías.



OBS

Software libre y de código abierto utilizado para la transmisión en vivo y la grabación de video.



CHATGPT

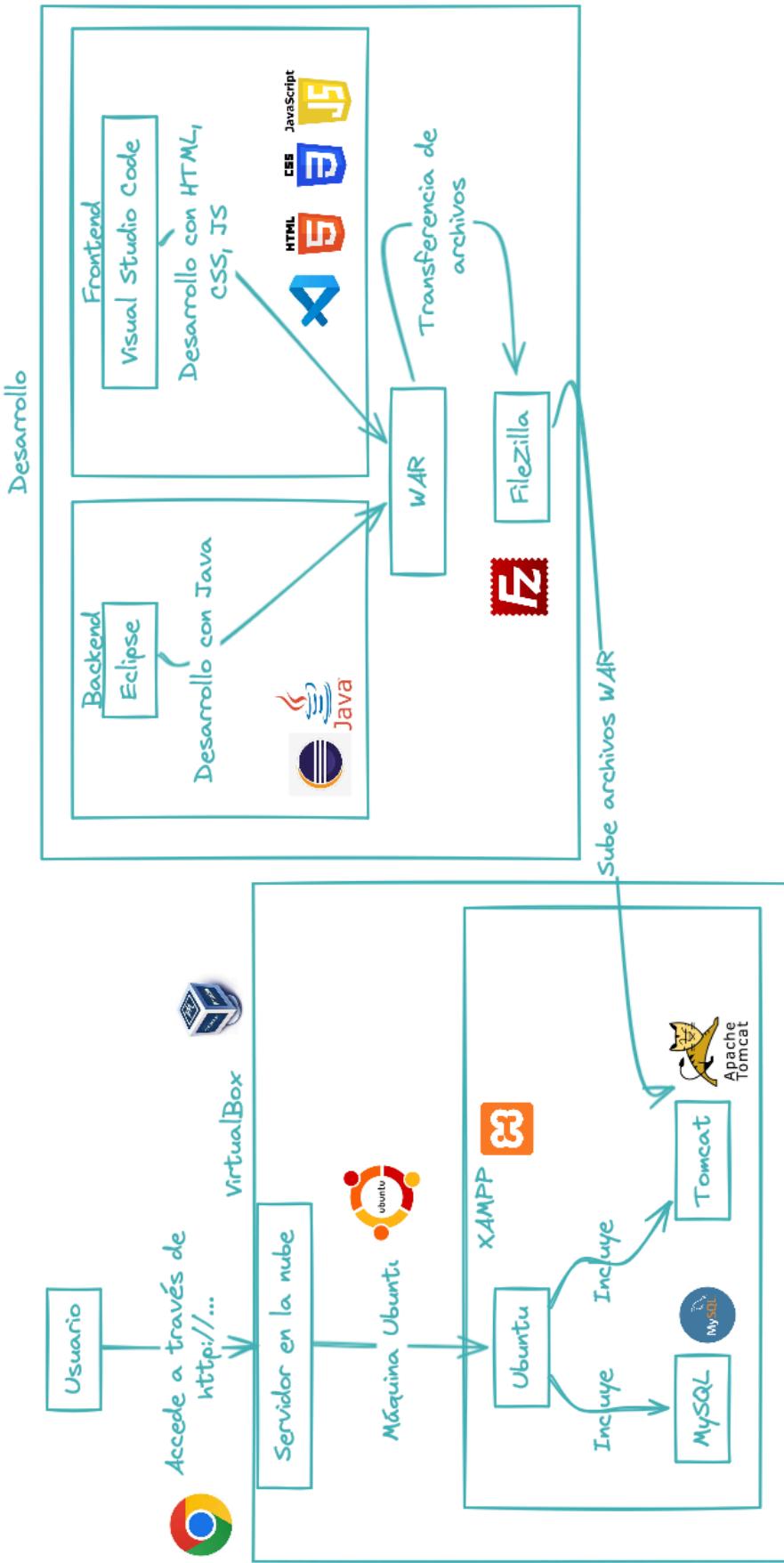
Modelo de lenguaje desarrollado por OpenAI diseñado para mantener conversaciones naturales y proporcionar respuestas coherentes y útiles a una amplia variedad de preguntas y temas.



4. IMPLEMENTACIÓN

REPRESENTACIÓN ESQUEMÁTICA

GRÁFICO 4



4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.1 FRONT-END

Anteriormente hemos presentado todas las páginas que conforman nuestro proyecto, ahora vamos a pasar a ver en detalle.

Un elemento gráfico bastante notable, en cuanto a su efecto visual, que incluimos en nuestra web es el carrusel de imágenes. Presente en todas las páginas públicas, hace una transición entre las imágenes que le hemos proporcionado. Pasamos a explicarlo en profundidad



En nuestra página **index.html** encontramos las imágenes listadas. Solo habría que añadir una nueva entrada en la lista **** para incluir una nueva imagen

```
<div class="carrusel">
  <ul>
    <li id="imagen1">
      
    </li>
    <li id="imagen2">
      
    </li>
    <li id="imagen3">
      
    </li>
  </ul>
</div>
```

4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.1 FRONT-END

Y en nuestros archivos JavaScript la función que lo carga y controla.

```
function iniciarCarrusel() {
  const imagenes = document.querySelectorAll(".carrusel li")
  let indice = 0

  setInterval(() => {
    imagenes[indice].style.opacity = 0
    indice = (indice + 1) % imagenes.length
    imagenes[indice].style.opacity = 1
  }, 3500)
}
```

1. En la primera línea se selecciona todos los elementos **** dentro del elemento con la **clase carrusel**. El resultado es una lista que contiene cada imagen dentro del carrusel.
2. Declaramos la **variable indice** y se inicializa en 0. Esta variable se utilizará para llevar la cuenta de la imagen actual que se muestra en el carrusel.
3. Empezamos a controlar el movimiento del carrusel.
 - a. Establece la opacidad de la imagen actual (indice) a 0, haciendo que desaparezca gradualmente.
 - b. Incrementa el valor de **indice** en 1 para pasar a la siguiente imagen. El uso de la operación **% imagenes.length** garantiza que el índice vuelva a 0 cuando se haya alcanzado el final de la lista, creando un efecto de bucle continuo.
 - c. Establece la opacidad de la nueva imagen actual a 1, haciéndola visible. Esto sucede inmediatamente después de ocultar la imagen anterior, dando la impresión de que una imagen se desvanece para dar paso a la siguiente.
 - d. Realiza las operaciones cada 3500 milisegundos (3.5 segundos)

4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.2 BACK-END

Desde nuestro back somos capaces de controlar los datos que recibimos desde el front, enviarlos o consultarlos en nuestra base de datos y devolverlos al front para su visualización. Un buen ejemplo de la lógica de negocio es el formulario añadir solicitud disponible para todos los usuarios de nuestra web. Veámoslo en profundidad.

El servlet **buscarSolicitud** recibirá desde el front un DNI

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    PrintWriter respuesta = response.getWriter();
    String dni = request.getParameter("dni");
    DaoSolicitud dao;
    try {
        dao = new DaoSolicitud();
        if (dao.dniExiste(dni)) {
            // Json de ese registro
            Solicitud s = new Solicitud();
            try {
                s.obtenerPorDni(dni);
                respuesta.print(s.dameJson());
                System.out.println(s.dameJson()); // Comprobar que me llega desde la base de datos
            } catch (SQLException e) {
                e.printStackTrace();
                response.sendRedirect("error.html");
            }
        } else {
            System.out.println("DNI No encontrado");
            response.sendRedirect("error.html?error=1");
        }
    } catch (SQLException e) {
        e.printStackTrace();
        response.sendRedirect("error.html");
    }
}
```

En caso de que exista, utilizará métodos como **obtenerPorDni** y **dameJson**, del modelo y el DAO para obtener los datos y mandará el JSON resultante de nuevo al front. En caso de que el DNI no existiese u otro error o excepción se gestionará con muestra de errores.

4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.3 CONSULTAS

Podemos referenciar algunas consultas más particulares de nuestro proyecto

```
public boolean dniExiste(String dni) throws SQLException {
    String sql = "SELECT COUNT(*) FROM solicitud WHERE dni=?";
    PreparedStatement ps = con.prepareStatement(sql);
    ps.setString(1, dni);
    ResultSet rs = ps.executeQuery();
    rs.next();
    return rs.getInt(1) > 0; // Verifica si hay al menos un registro
}
```

COUNT es una función que devuelve el número de filas(*) donde coincide el dni proporcionado. Si es mayor a 0, el DNI existe, en caso contrario, no existe.

```
public void asignarNumeros() throws SQLException {
    // Obtener todas las solicitudes ordenadas por id y ordenadas =
    String sql = "SELECT id FROM solicitud ORDER BY id";
    PreparedStatement ps = con.prepareStatement(sql);
    ResultSet rs = ps.executeQuery();

    // Actualizar el campo num_sorteo con un número correlativo
    int contador = 1;
    while (rs.next()) {
        String sqlUpdate = "UPDATE solicitud SET num_sorteo = ? WHERE id = ?";
        PreparedStatement updateStatement = con.prepareStatement(sqlUpdate);
        updateStatement.setInt(1, contador);
        updateStatement.setInt(2, rs.getInt("id"));
        updateStatement.executeUpdate();
        contador++;
    }
}
```

Método para el proceso asignarNúmero, el cual recoge todos los **id** ordenados por **id** y mediante un **bucle while** actualiza el contenido del atributo **num_sorteo** con un número secuencial, controlamos este incremento con la **variable contador(contador++)**

4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.3 CONSULTAS

En el repositorio compartido, dentro del directorio **Material de referencia/BD** podemos encontrar:

1. Archivo **Archivo_verano.sql**. Instrucciones SQL para la creación de la base de datos y las tablas. La tabla admin es creada utilizando el asistente.
2. Archivo **inserts_ejemplo.sql**. Algunos ejemplos de inserts mediante instrucciones SQL. Se crea para agilizar el proceso de incluir datos para el testeo.
3. Archivo **update_edicion.sql**. Debido a una evaluación posterior a la creación de la base de datos, se determina que el campo edición debe ser incluido para ganar persistencia. En el archivo podemos ver el proceso de creación y como rellenamos las tuplas existentes.
4. Archivo **consultas_ejemplo.doc**. Documento que muestra capturas de pantalla donde se muestra la consulta y el resultado. No implementadas.

4. IMPLEMENTACIÓN

4.3. CÓDIGO

4.3.4 DIFICULTADES

Me gustaría recoger en este pequeño apartado, al menos una de las dificultades encontradas a la hora de desarrollar mi aplicación y digo una, porque han sido numerosas, y la mayoría de las veces con la sensación y certeza de que no tenía los conocimientos para resolverla.

Gestión de error para el servlet que maneja el **Buscador de DNI**

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    PrintWriter respuesta = response.getWriter();
    String dni = request.getParameter("dni");
    DaoSolicitud dao;
    try {
        dao = new DaoSolicitud();
        if (dao.dniExiste(dni)) {
            // Json de ese registro
            Solicitud s = new Solicitud();
            try {
                s.obtenerPorDni(dni);
                respuesta.print(s.dameJson());
                System.out.println(s.dameJson()); // Comprobar que me llega desde la base de datos
            } catch (SQLException e) {
                e.printStackTrace();
                response.sendRedirect("error.html");
            }
        } else {
            System.out.println("DNI No encontrado");
            response.sendRedirect("error.html?error=1");
        }
    } catch (SQLException e) {
        e.printStackTrace();
        response.sendRedirect("error.html");
    }
}
```

El código parece correcto y cuando busco un DNI muestra su resultado de manera correcta y eficaz, el problema viene cuando introduzco un DNI no existente. Eclipse lo gestiona hasta el punto que puede ponerme por consola *DNI No encontrado*, también cualquiera de las otras excepciones pero el problema llega cuando quiero mandar esa respuesta al usuario para que la visualice en su navegador.

Eclipse no muestra ningún error, en cambio el navegador muestra:
Uncaught (in promise) SyntaxError: Unexpected token '<', '<!DOCTYPE ... is not valid JSON'

Buscando y leyendo sobre este error, parece ser que JavaScript espera recibir un JSON, el JSON con el resultado de la búsqueda, pero en lugar de eso le estoy mandando HTML. Para hacer que funcione necesito asegurarme de mandar siempre un JSON aún cuando haya errores.

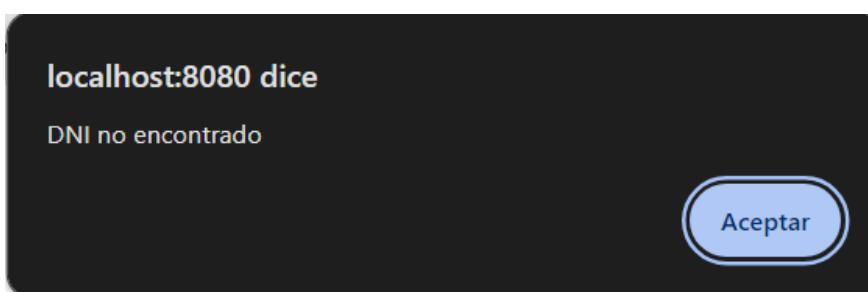
Modifico el servlet:

```
        respuesta.print("{\"error\":\"Error al obtener la solicitud por DNI\"}");
    }
} else {
    System.out.println("DNI No encontrado");
    respuesta.print("{\"error\":\"DNI no encontrado\"}");
}
} catch (SQLException e) {
    e.printStackTrace();
    respuesta.print("{\"error\":\"Error en la base de datos\"}");
}
```

Y modifco el JavaScript para que interprete las respuestas

```
function llamoSol(dni) {
    fetch("BuscarSolicitud?dni=" + dni)
        .then((response) => {
            if (!response.ok) {
                throw new Error("Problema con la respuesta del servidor")
            }
            return response.json()
        })
        .then((data) => {
            if (data.error) {
                alert(data.error) // Muestra el mensaje de error al usuario
            } else {
                pintarResultados(data) // Función para procesar y mostrar los datos
            }
        })
}
```

Ahora ya muestra con un alert el error que hayamos tenido.



4. IMPLEMENTACIÓN

4.4 DOCUMENTACIÓN. JAVADOC

Javadoc es una herramienta utilizada para generar documentación en formato HTML a partir de comentarios en el código fuente Java. Esta documentación facilita la comprensión del código y su uso por parte de otros desarrolladores. Los principales beneficios de Javadoc son:

1. Documentación Estandarizada: Genera documentación siguiendo un formato estándar que es ampliamente reconocido y utilizado en la comunidad Java.
2. Automatización: Permite crear documentación detallada de clases, métodos y campos automáticamente a partir de los comentarios en el código.
3. Navegación fácil: La documentación generada incluye enlaces y estructuras que facilitan la navegación entre diferentes partes del código.
4. Integración: Se puede integrar con entornos de desarrollo como Eclipse y herramientas de construcción como Maven para generar documentación automáticamente durante el proceso de compilación.

Javadoc contribuye a mantener un código bien documentado y fácil de entender, mejorando la colaboración y el mantenimiento del software.

Toda la documentación generada puede consultarse en el repositorio de GitHub

4. IMPLEMENTACIÓN

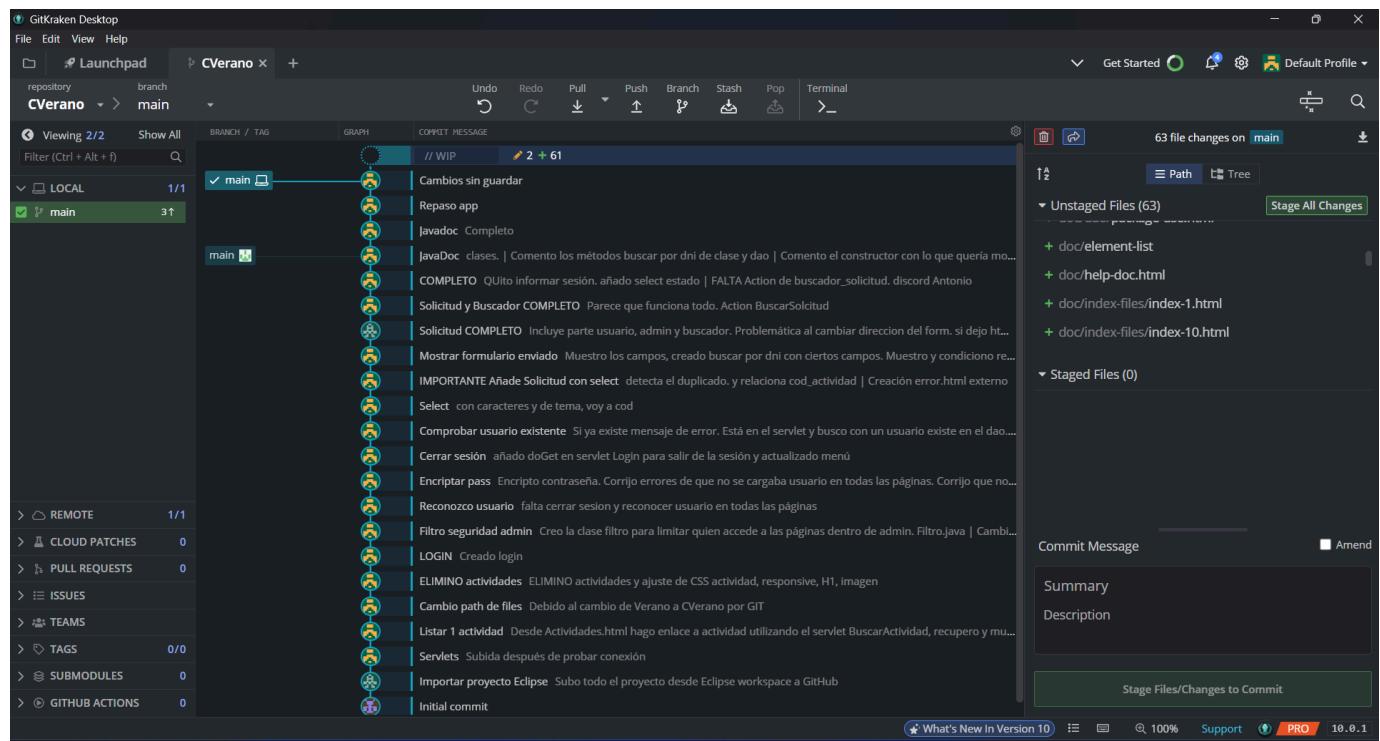
4.5 CONTROL DE VERSIONES

El control de versiones es una práctica esencial en el desarrollo de software que permite gestionar los cambios en el código fuente de un proyecto. **GitKraken** y **GitHub** son dos herramientas populares que facilitan esta tarea.

GitKraken es una interfaz gráfica para Git que nos ofrece una experiencia visual e intuitiva para gestionar repositorios, realizar commits, crear ramas y resolver conflictos.

GitHub es una plataforma de alojamiento de repositorios Git que permite la colaboración en proyectos de software. Ofrece herramientas para revisión de código, gestión de proyectos y seguimiento de problemas. Al combinar GitKraken con GitHub, podemos aprovechar la facilidad de uso de GitKraken junto con las capacidades de colaboración y gestión de proyectos de GitHub.

Juntas, estas herramientas nos proporcionan un entorno robusto para el control de versiones, facilitando la colaboración y mejorando la eficiencia.



4. IMPLEMENTACIÓN

4.6 PRUEBAS UNITARIAS

El objetivo de las pruebas unitarias es comprobar que las respuestas que estamos obteniendo, son las adecuadas. En otras palabras, que el programa “*hace lo que tiene que hacer*”.

Utilizaremos la herramienta **JUnit**, integrada en Eclipse, para realizar dichos test de comprobación.

Vamos a hacer una prueba para comprobar que las contraseñas cifradas se están generando correctamente. Por ello en el test pondremos una contraseña sin cifrar, su versión cifrada y comprobaremos que el test nos responde el mismo resultado.

```
@Test
void test() {
    // Contraseña sin cifrar
    String password = "mipass";
    // El hash MD5 esperado para "mipass"
    String expectedHash = "019c9fbe91025390efb1c65ce5eee5d6";
    // Llamamos al método
    String actualHash = Administrador.myMD5(password);
    // Comparamos el string que tenemos con el generado con el método.
    assertEquals(expectedHash, actualHash, "El hash MD5 generado no coincide con el esperado");
}
```

Ejecutamos la prueba con **Run as JUnit Test** y comprobaremos los resultados del test. En caso de error el panel JUnit nos dará más información. Si la prueba ha tenido resultados positivos y el método funciona según lo esperado visualizaremos algo así

The screenshot shows the Eclipse IDE interface. On the left, the JUnit runner output window displays: "Finished after 0.09 seconds", "Runs: 1/1", "Errors: 0", and "Failures: 0". Below this, it says "passTest [Runner: JUnit 5] (0,000 s)". On the right, the code editor shows the Java source code for the `passTest` class:

```
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class passTest {
6
7     @Test
8     void test() {
9         // Contraseña sin cifrar
10        String password = "mipass";
11        // El hash MD5 esperado para "mipass"
12        String expectedHash = "019c9fbe91025390efb1c65ce5eee5d6";
13        // Llamamos al método
14        String actualHash = Administrador.myMD5(password);
15        // Comparamos el string que tenemos con el generado con el método.
16        assertEquals(expectedHash, actualHash, "El hash MD5 generado no coincide con el esperado");
17    }
18}
19
20
21
22}
23
```

4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

Para el despliegue de la aplicación hemos montado un **Ubuntu** al que le hemos dedicado 40GB y 1 CPU.

En cuanto a la configuración de red hemos establecido **Adaptador puente o bridged adapter**. Esto significa que permite conectarse directamente a la red física de la máquina anfitriona. La máquina virtual tiene su propia dirección IP en la red local.

```
usuario@Ubuntu22:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 22.04.4 LTS
Release:        22.04
Codename:       jammy
usuario@Ubuntu22:~$
```

Una buena práctica es siempre mantener el S.O. actualizado

```
usuario@Ubuntu22:~$ sudo apt-get update
Obj:1 http://es.archive.ubuntu.com/ubuntu jammy InRelease
Obj:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Obj:3 https://ppa.launchpadcontent.net/linuxuprising/java/ubuntu jammy InRelease
Obj:4 http://es.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:5 http://es.archive.ubuntu.com/ubuntu jammy-backports InRelease
Leyendo lista de paquetes... Hecho
```

Instalaremos un servidor LAMP.

1. Descargaremos el fichero de instalación
2. ***sudo chmod 777xampp-linux-x64-7.2.9-0-installer.run*** para cambiar los permisos del archivo de XAMPP para que pueda ser ejecutado por cualquier usuario.
3. ***sudo ./xampp-linux-x64-7.2.9-0-installer.run*** Ejecutamos el archivo con privilegios de superusuario.

4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

Una vez instalado:

1. Arrancamos ***sudo /opt/lampp/lampp start***

Para hacer visible a phpmyadmin

1. ***sudo /opt/lampp/lampp stop***

2. ***sudo nano /opt/lampp/etc/extrahtdp-xampp.conf***

3. Editar el fichero. Incluir "***Required all granted***"

4. ***sudo /opt/lampp/lampp restart***



4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

Instalación de Apache Tomcat para Ubuntu

- Crear y descargar el repositorio para Java
 - ***sudo add-apt-repository ppa:linuxuprising/java***
- Actualizar el repositorio
 - ***sudo apt update***
- Instalar el JDK ejecutando el siguiente comando
 - ***sudo apt install oracle-java17-installer***
- Cuando se finalice la instalación, se verificará la versión de la instalación de Java disponible
 - ***java -version***
- Actualizarel JDK
 - ***sudo apt update***
- Indicar que se use la versión descargada por defecto
 - ***sudo apt install oracle-java17-set-default***

```
usuario@Ubuntu22:/home$ java -version
java version "17.0.6" 2023-01-17 LTS
Java(TM) SE Runtime Environment (build 17.0.6+9-LTS-190)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.6+9-LTS-190, mixed mode, sharing)
```

- Para instalar Tomcat, se necesitará instalar la versión 9 de tomcat
 - ***sudo apt install tomcat9 tomcat9-admin***
- Comprobamos el estado
 - ***sudo systemctl status tomcat9***
- Si vemos que no está arrancado bien, instalamos el JDK por defecto
 - ***sudo apt install default-jdk***
- Comprobamos que el puerto 8080 está a la escucha
 - ***ss -ltn***
- Reiniciamos el servidor
 - ***sudo systemctl restart tomcat9***
- Volvemos a comprobar el estado
 - ***sudo systemctl status tomcat9***

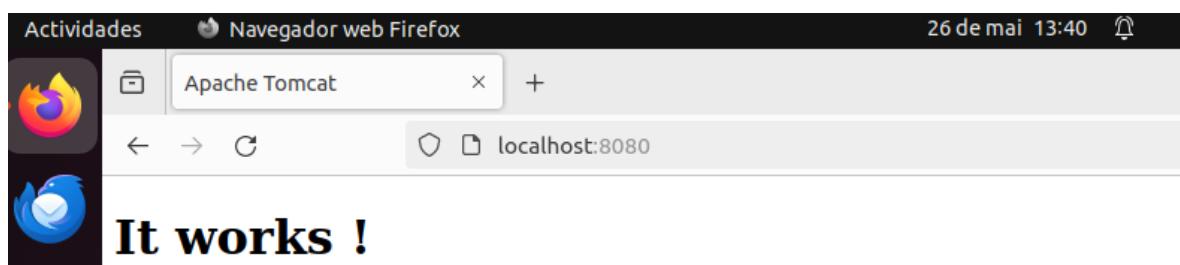
4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

```
● tomcat9.service - Apache Tomcat 9 Web Application Server
   Loaded: loaded (/lib/systemd/system/tomcat9.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-05-26 12:18:42 CEST; 1h 12min ago
     Docs: https://tomcat.apache.org/tomcat-9.0-doc/index.html
         PID: 677 (java)
       Tasks: 28 (limit: 3438)
      Memory: 121.1M
        CPU: 32.617s
      CGroup: /system.slice/tomcat9.service
              └─677 /usr/lib/jvm/default-java/bin/java -Djava.util.logging.config.file=/

mai 26 12:19:47 Ubuntu22 tomcat9[677]: Deployment of deployment descriptor [/etc/tomcat9/conf/web.xml]
mai 26 12:19:47 Ubuntu22 tomcat9[677]: Despliegue del descriptor de configuración [/etc/tomcat9/conf/web.xml]
mai 26 12:19:47 Ubuntu22 tomcat9[677]: The path attribute with value [/host-manager] in
mai 26 12:19:52 Ubuntu22 tomcat9[677]: Al menos un JAR, que se ha explorado buscando TLDs
mai 26 12:19:52 Ubuntu22 tomcat9[677]: Deployment of deployment descriptor [/etc/tomcat9/conf/web.xml]
mai 26 12:19:52 Ubuntu22 tomcat9[677]: Desplegando el directorio [/var/lib/tomcat9/webapps]
mai 26 12:19:57 Ubuntu22 tomcat9[677]: Al menos un JAR, que se ha explorado buscando TLDs
mai 26 12:19:57 Ubuntu22 tomcat9[677]: Deployment of web application directory [/var/lib/tomcat9/webapps/host-manager]
mai 26 12:19:57 Ubuntu22 tomcat9[677]: Starting ProtocolHandler ["http-nio-8080"]
mai 26 12:19:58 Ubuntu22 tomcat9[677]: Server startup in [35976] milliseconds
~
```

- Los usuarios de Tomcat se definen en **/opt/tomcat/conf/tomcat-users.xml**.
 - Abrir el archivo para editararlo con el siguiente comando
 - **sudo nano /etc/tomcat9/tomcat-users.xml**



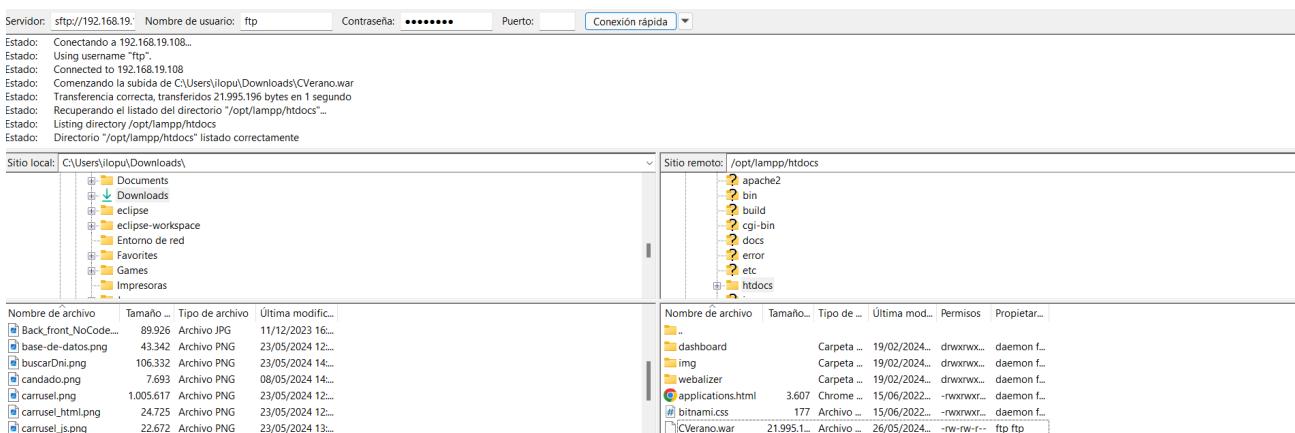
4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

Transferencia de archivos FTP

Creamos un usuario para acceder al directorio default que está en **/opt/lampp/htdocs**

- Comandos alta usuario:
 - **sudoadduser ftp**
 - **sudo passwd ftp**
- Creamos un grupo de usuarios para que tengan permisos en la carpeta de **/opt/lampp/htdocs**
 - **sudogroupadd ftp_group**
- Incluimos a usuario ftp al grupo
 - **sudousermod -G ftp_group ftp**
- Damospermisos al grupo para que maneje los ficheros del directorio **/opt/lampp/htdocs**
 - **sudochmod -R 775 /opt/lampp/htdocs**
 - **sudochown -R daemon:ftp_group /opt/lampp/htdocs**

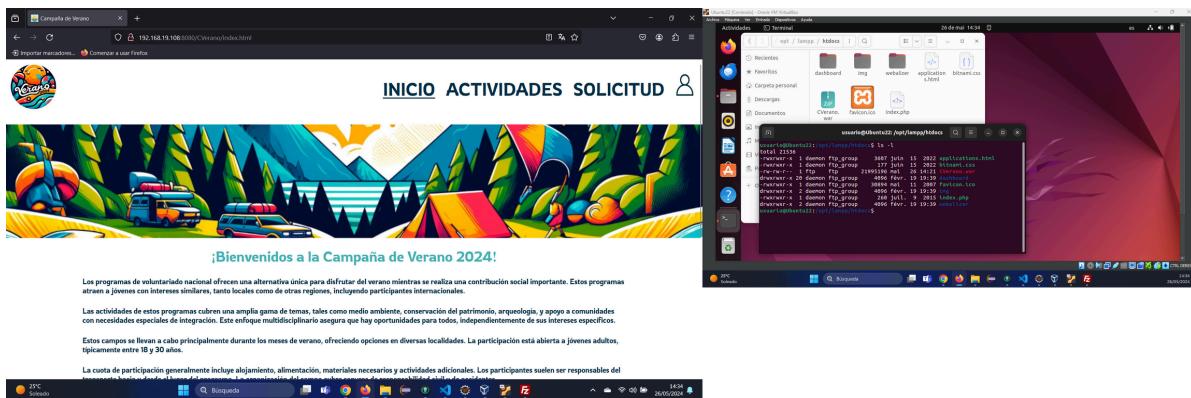


4. IMPLEMENTACIÓN

4.7 DESPLIEGUE

Una vez finalizado este proceso, podremos acceder a nuestro proyecto desde un dispositivo externo.

Para hacer la comprobación accedemos desde nuestro SO Windows a la IP de la máquina virtual. Se incluye captura de pantalla en la carpeta **Material de referencia/Imagenes** para su mejor visualización, archivo **windows_ubuntu.png**



5. FUTUROS TRABAJOS

Un punto importante de todo proyecto es que sea escalable.

Funcionalidades que no han sido posible implementarlas o porque son necesidades que en un principio no se habían planteado. Listaré alguna de esos procesos y funcionalidades que sería positivo añadir:

- Gestión de listado de participantes por actividad. Mostrar participantes que han obtenido plaza, calculado por el número de plazas disponibles, una vez superado ese número, mostrar los suplentes.
- Restricción de actividades por fecha de nacimiento. Una vez introducida la fecha de nacimiento, el programa filtrará en cuales actividad puede participar según el rango de edades permitido.
- Creación un espacio para el usuario externo en el cual, una vez logueado, pueda acceder a su solicitud y modificarla. También sería interesante, a medida que avanza el proceso, poder adjuntar documentos que se le soliciten.
- Envío de mensajes predefinidos para recibir un informe de su solicitud, estado según se avanza en el proceso y reclamación de documentación.
- Buscador avanzado de actividad que incluya filtrado por temática, fechas y edad.
- Apartado blog para poner de manera más detallada experiencias de los participantes.

CONCLUSIONES

Podría llenar este apartado solo de palabras bonitas, pero no serían representativas del todo el proceso.

Entiendo que estudiar de manera online es completamente diferente a presencial, para mi era la primera vez. Quizá si he sentido esa falta de respuesta inmediata que te da una clase presencial, pero haré de debilidades, fortalezas y diré que de eso también se aprende y me aporta valor profesional.

Se que acabo el curso con una visión completamente diferente, con más luz en el túnel, aunque a veces haya cortes de electricidad. Mi cabeza ya cierra el círculo, ya sabe el caminito que toman los datos, hacia dónde van y de dónde vienen. Esto quizá siempre ha estado claro, lo complicado era como viajaban y parece que eso también ya está afianzado.

POO, ese primer momento de ¡boom! forzarme a pensar así y finalmente ver sus posibilidades y ventajas. Me quedo con los momentos en los que funcionan las cosas, un bucle o un condicional que hace lo que tiene que hacer, me quedo con ese subidón, con esa sensación ¡Algo parecido a ganar un Grand Slam!

Gracias por las nuevas gafas que permiten ver las webs por secciones, en grid. Aumento el nivel de TOC++

La satisfacción cuando se monta una máquina virtual y puedes acceder a ella desde fuera, feliz como si hubiese mandado a orbitar un satélite.

Por las consultas SQL y los planteamientos E/R, esos pequeños rompecabezas que dan vicio, espero seguir ganando niveles para episodio final.

Orgullosa de mí, porque en este momento que cierro esta memoria, me siento satisfecha con el resultado. Por las dificultades que me he encontrado y por haber podido afrontarlas.

Esta aficionada a los juegos de lógica ha encontrado un filón en el desarrollo y se quita una espinita que tenía desde los 18 por no haber elegido de manera correcta sus estudios.

BIBLIOGRAFIA |

OpenAI. (2024). ChatGPT (Versión GPT-4) [Modelo de lenguaje de inteligencia artificial]. Disponible en <https://www.openai.com/>

Wikipedia. (n.d.). Wikipedia, La enciclopedia libre. Disponible en <https://www.wikipedia.org/>

Flaticon. (n.d.). Flaticon. Disponible en <https://www.flaticon.com/>

Freepik. (n.d.). Freepik. Disponible en <https://www.freepik.com/>

Google. (n.d.). Google Fonts. Google. Disponible en <https://fonts.google.com/>

Oracle. (n.d.). Java API documentation. Oracle. Disponible en <https://docs.oracle.com/javase/8/docs/api/>