

# Memoria del proyecto

## *Plataforma web para el aprendizaje de ciberseguridad*

Trabajo de Fin de Grado  
INGENIERÍA INFORMÁTICA



**VNiVERSiDAD  
D SALAMANCA**

**Julio de 2022**

**Autor:**

*Jaime Gómez García*

**Tutores:**

*André Filipe Sales Mendes*

*Gabriel Villarrubia González*

*Juan Francisco de Paz Santana*



## **Certificado de tutores**

D. André Sales Mendes, D. Juan Francisco de Paz Santana y D. Gabriel Villarrubia González, profesores del Departamento de Informática y Automática de la Universidad de Salamanca.

HACEN CONSTAR:

Que el trabajo titulado “Plataforma web para el aprendizaje de ciberseguridad” ha sido realizado por Jaime Gómez García, con número de documento 71039763P y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Grado de Ingeniería Informática de esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca, a 1 de Julio de 2022

D. André Sales  
Mendes

D. Juan Francisco  
de Paz Santana

D. Gabriel Villarubia  
González



## Resumen

---

Actualmente la ciberseguridad suele ser un tema del que se habla mucho en los medios y en la industria, debido a la multitud de ataques que suele haber en las empresas y organizaciones gubernamentales, pero a la hora de la verdad resulta complicado a las personas que quieren especializarse en la seguridad informática saber por dónde empezar, ya que se trata de un tema muy amplio y complicado, es por esto por lo que surge este proyecto.

Este trabajo de fin de grado consiste en una plataforma web para aprender ciberseguridad a la que se le ha dado el nombre de Hacker Machines, en la cual el proceso de aprendizaje consiste en la realización de diferentes retos cibernéticos que tendrán que resolver los usuarios registrados en el sistema.

Estos retos son máquinas virtuales que se están ejecutando en un entorno controlado dentro de una red interna desarrollada para este propósito. Los retos están creados especialmente para que sean hackeados ya que tienen una serie de vulnerabilidades que deben ser explotadas. Para que los usuarios puedan conectarse a estas máquinas virtuales se deben de conectar a través de una VPN a esta red interna.

Una vez el usuario está listo para resolver el reto se le facilita la dirección IP a la que se tienen que comenzar a realizar un análisis en búsqueda de diferentes vulnerabilidades para poder ir encontrando fallos en ese sistema, una vez se vaya aprovechando de esos fallos irán encontrando las diferentes soluciones que deben ir introduciendo en la plataforma, esto hará que se le asignen una serie de puntos al usuario, que se irán acumulando, creando así un ranking con todos los usuarios.

**Palabras clave:** Ciberseguridad, aprendizaje, VPN, vulnerabilidades.



## Summary

---

Currently, cybersecurity is usually a topic that is talked about a lot in the media and in the industry, due to the multitude of attacks that usually take place in companies and organizations, but when it comes to the truth, it is complicated for people who want to computer security know where to start, since it specializes in a very broad and complicated topic, that is why this project arises.

This end-of-degree project consists of a web platform to learn cybersecurity that has been given the name of Hacker Machines, in which the learning process consists of carrying out different cyber challenges that registered users will have to solve. the system.

These challenges are virtual machines that are running in a controlled environment within an internal network developed for this purpose. The challenges are specially created to be hacked since they have a series of vulnerabilities that must be exploited. In order for users to connect to these virtual machines, they must connect through a VPN to this internal network.

Once the user is ready to solve the challenge, they are provided with the IP address to which they have to start performing an analysis in search of different vulnerabilities in order to find faults in that system, once they take advantage of those faults they should find the different solutions that must be introduced on the platform, this will make the user earn points , which will accumulate, thus creating a ranking with all users.

**Key Words:** cybersecurity, VPN, vulnerabilities.





## Índice general

---

1. Introducción.....	15
2. Objetivos .....	17
2.1. Objetivos del sistema .....	17
2.2. Objetivos personales.....	18
3. Conceptos Teóricos .....	19
3.1. Máquina Virtual .....	19
3.2. VPN.....	20
3.3. CTF .....	21
3.4. REST API.....	21
4. Técnicas y herramientas .....	22
4.1. Técnicas y herramientas usadas en el servicio web .....	22
4.1.1 Python.....	22
4.1.2. Django .....	23
4.1.3. Bootstrap .....	24
4.1.4. SQLite.....	24
4.1.5. Docker .....	25
4.1.6. Oracle VM VirtualBox.....	26
4.1.7. OpenVPN.....	27
4.1.8. Visual Studio Code .....	28
4.1.9. VulnHub .....	29
4.2. Herramientas CASE .....	30
4.2.1. Microsoft Project .....	30
4.2.2. Visual Paradigm.....	30
4.2.3. EZEstimate .....	31
4.2.4. Sphinx.....	32
4.3. Herramientas para el control de versiones.....	32

4.3.1 GIT .....	32
4.3.2. GitHub.....	33
5. Aspectos relevantes del desarrollo.....	34
5.1. Marco de trabajo .....	34
5.2. Estimación de esfuerzo .....	36
5.3. Planificación temporal .....	38
5.4. Especificación de requisitos .....	40
5.4.1. Participantes .....	40
5.4.2. Objetivos del sistema .....	40
5.4.3. Requisitos de información.....	41
5.4.4. Requisitos Funcionales.....	43
5.4.5. Requisitos no funcionales .....	45
5.5. Análisis de Requisitos .....	46
5.5.1. Modelo de Dominio .....	46
5.5.2 Realización de los casos de uso.....	47
5.5.3. Clases de Análisis.....	48
5.5.4. Arquitectura del Modelo de Análisis.....	49
5.6. Diseño del Sistema .....	49
5.6.1. Patrones de diseño .....	49
5.6.2. Subsistemas de diseño.....	50
5.6.3. Clases de diseño .....	51
5.6.4. Vista Arquitectónica .....	53
5.6.5. Realización de los Casos de Uso .....	53
5.6.6. Diagrama de Despliegue .....	54
5.7. Implementación.....	56
5.8.Pruebas.....	58
5.9. Funcionamiento del sistema de desarrollo .....	58

5.9.1. Aplicación Web .....	58
5.9.1.1. Iniciar Sesión.....	59
5.9.1.2. Registrarse.....	59
5.9.1.3. Pantalla Home.....	60
5.9.1.4. OpenVPN .....	60
5.9.1.5. Ver Perfil .....	61
5.9.1.6. Lista de Máquinas .....	62
5.9.1.7. Añadir Máquinas .....	63
5.9.1.8. Ver detalles del Reto.....	64
5.9.1.9. Resolver Reto .....	65
5.9.1.10. Ver Categorías .....	67
5.9.1.11. Ver Ranking .....	67
5.9.1.12. Ver perfil de un usuario .....	68
6. Conclusiones.....	70
7. Líneas de trabajo futuras.....	72
5. Bibliografía .....	73

## Índice de ilustraciones

---

Ilustración 1: Representación de una máquina virtual.....	19
Ilustración 2: Conexión VPN.....	20
Ilustración 3: Ejemplo de uso de Python. ....	23
Ilustración 4: Ejemplo de uso de Bootstrap en la plataforma. ....	24
Ilustración 5: Ejemplo Docker.....	25
Ilustración 6: Interfaz de Oracle Virtualbox.....	26
Ilustración 7: Uso de OpenVPN en el proyecto. ....	27
Ilustración 8: Interfaz Visual Studio Code.....	28
Ilustración 9: Página web de VulnHub.com.....	29
Ilustración 10: Interfaz de Microsoft Project.....	30
Ilustración 11: Interfaz Visual Paradigm. ....	31
Ilustración 12: Interfaz EZEstimate.....	31
Ilustración 13: Documentación del proyecto con Sphinx. ....	32
Ilustración 14: Interfaz GitHub.....	33
Ilustración 15: Proceso Unificado.....	35
Ilustración 16: Estimación de esfuerzo EZEstimate. ....	37
Ilustración 17: Iteración 1 etapa de inicio. ....	38
Ilustración 18: Ejemplo diagrama de Grantt. ....	39
Ilustración 19: Diagrama de paquetes.....	43
Ilustración 20: Actores del sistema.....	43
Ilustración 21: Paquete Gestión de Retos. ....	44
Ilustración 22: Modelo de Dominio. ....	47
Ilustración 23: Diagrama de Secuencia Añadir Reto. ....	47
Ilustración 24: Diagrama de Comunicación Gestión de Retos. ....	48
Ilustración 25: Ejemplo diagrama Modelo Vista Controlador.....	50
Ilustración 26: Modelo de Diseño. ....	51
Ilustración 27: Vista Aplicación Web. ....	52
Ilustración 28: Controlador Aplicación Web.....	52
Ilustración 29: Vista Arquitectónica. ....	53
Ilustración 30: Diagrama de Secuencia Recuperar Contraseña.....	54
Ilustración 31: Modelo de despliegue. ....	55
Ilustración 32: Estructura de la red de la plataforma. ....	57

Ilustración 33: Iniciar sesión. ....	59
Ilustración 34: Registrarse.....	59
Ilustración 35: Pantalla Home.....	60
Ilustración 36: Pantalla OpenVPN.....	60
Ilustración 37: Conexión a la VPN.....	61
Ilustración 38: Ver Perfil. ....	62
Ilustración 39: Lista máquinas siendo administrador.....	62
Ilustración 40: Lista máquinas sin ser administrador.....	63
Ilustración 41: Añadir máquina. ....	64
Ilustración 42: Ver detalles del reto. ....	65
Ilustración 43: Escaneo de la dirección IP.....	65
Ilustración 44: Ejemplo servicio http del reto. ....	66
Ilustración 45: Reto completado. ....	66
Ilustración 46: Ver categorías.....	67
Ilustración 47: Ranking de usuarios. ....	67
Ilustración 48: Buscar usuario. ....	68
Ilustración 49: Ver perfil de un usuario siendo administrador.....	68
Ilustración 50: Ver perfil de un usuario sin ser administrador.....	69

## Índice de tablas

---

Tabla 1: Objetivo Gestión de Usuarios.....	41
Tabla 2: Información respecto al reto accedido por el usuario. ....	42
Tabla 3: Caso de Uso Acceder al Reto. ....	45

## 1. Introducción

---

Hoy en día el sector de la ciberseguridad está en auge debido a la gran cantidad de vulnerabilidades que surgen y los ataques que se realizan a grandes infraestructuras. Es por esto por lo que cada vez más gente quiere iniciarse en la ciberseguridad, para intentar combatir este tipo de ataques que pueden llegar a ser muy perjudiciales para una empresa.

Al tratarse de un tema tan dinámico y nuevo existen pocas maneras en las que una persona pueda aprender y especializarse en la seguridad informática, por esta razón se crea esta plataforma, para intentar solventar esta falta de conocimiento y sobre todo falta de medios para poder practicar aplicando lo que se haya aprendido.

Desde ya hace unos años existe el formato de los CTFs, que son una serie de juegos diseñados por la comunidad con una serie de vulnerabilidades, con la única finalidad de poder encontrarlas y aprovecharlas de ellas para así conseguir las llamadas Flags para poder resolver el juego. En base al concepto de CTF está construido este sistema, en donde están en ejecución una serie de retos, que los usuarios tienen que resolver para poder ganar más puntos.

A continuación, se expone la estructura del documento:

- **Objetivos:** Se especifican cuales son los objetivos fundamentales que se deben de cumplir para el desarrollo del sistema.
- **Conceptos teóricos:** Se explican los conceptos teóricos que se van a utilizar, con el fin de poder entender de manera adecuada el documento.
- **Técnicas y Herramientas:** Se documentan las técnicas y las herramientas que se han usado para la implementación de la plataforma.
- **Aspectos relevantes del desarrollo:** Se recogen los aspectos más destacados en el proceso de desarrollo del software.
- **Conclusiones:** Se tratan las conclusiones a las que se han llegado tras realizar el trabajo.
- **Líneas de trabajo futuras:** Posibles implementaciones y aspectos que se pueden implementar en un futuro.

Este documento va acompañado de los siguientes anexos:

- **Anexo I – Planificación temporal:** Documentación en la cual se realiza la planificación temporal del proyecto.
- **Anexo II – Especificación de requisitos:** Documentación en la que se especifican los requisitos del sistema.
- **Anexo III – Análisis de requisitos:** Documentación que contiene la realización del análisis de requisitos del proyecto.
- **Anexo IV – Diseño del sistema software:** Documentación que recoge el diseño del sistema software.
- **Anexo V – Documentación técnica:** Documentación que contiene de forma detallada las funciones que se utilizan en el sistema.
- **Anexo VI – Manual de usuario:** Documentación que contiene una guía de usuario para un correcto uso de la plataforma.



## 2. Objetivos

---

En esta parte se pretende explicar cuáles son los objetivos principales del proyecto, dado que se trata de una plataforma web de aprendizaje la mayoría de estos objetivos están diseñados para dar soporte a los usuarios y cómo estos interactúan con los retos.

### 2.1. Objetivos del sistema

El proyecto se pretende considerar acabado en el momento en el que se cumplan cada uno de los objetivos del sistema que se definen a continuación:

- **Sistema de puntuación:** El sistema debe ser capaz de asignar los puntos de manera adecuada a cada uno de los usuarios cada vez que resuelven un reto o encuentran una respuesta.
- **Gestión de usuarios:** Se debe poder crear, modificar, consultar y eliminar a los usuarios que están en el sistema.
- **Gestión de retos:** El sistema debe permitir poder visualizar los retos que se encuentran disponibles para que los usuarios puedan resolverlos. En caso de que el usuario sea administrador este podrá crear, modificar, eliminar o cambiar la disponibilidad de un reto.
- **Gestión de respuestas:** Se debe ser capaz de gestionar las respuestas enviadas por parte de los usuarios, comprobando así si estas son correctas o no.
- **Gestión de categorías:** El sistema debe permitir a los usuarios administradores poder visualizar, crear, modificar o borrar las categorías que definen los retos.
- **Gestión de VPN:** El sistema se tiene de encargar de crear un archivo de configuración para cada usuario de la plataforma, con este fichero los usuarios se podrán conectar a la red interna en donde se encuentran los retos.
- **Organización del ranking:** Se debe crear un ranking de todos los usuarios del sistema, organizándolo según la cantidad de puntos que ha conseguido cada uno.

- **Gestión de roles:** Dado que el sistema tiene usuarios administradores y usuarios normales, debe poder gestionar de manera adecuada las características especiales de cada uno de ellos.
- **Búsqueda de usuarios:** Se debe permitir buscar a los usuarios que se encuentran en la plataforma.

## 2.2. Objetivos personales

Desde que comencé la carrera he tenido un gran interés por el mundo de la ciberseguridad, siempre he intentado buscar información y ver vídeos para poder aprender más y especializarme, pero este proceso al principio fue difícil debido a lo complicado que es encontrar buena formación en un tema tan actual como es el de la seguridad informática.

Debido a esto considero que es una buena iniciativa crear una plataforma web en español que pueda ayudar y formar tanto a las personas que se adentran por primera vez en este increíble mundo, como a las personas que tienen amplios conocimientos y quieren practicar.

Además de esto creo que es una gran oportunidad para poner a prueba los conocimientos adquiridos durante estos años, tanto dentro como fuera de la universidad y que mejor manera que creando un sistema de estas dimensiones, siendo además de un tema que me apasiona como es la ciberseguridad.

### 3. Conceptos Teóricos

---

A continuación, se van a explicar cuáles son los conceptos teóricos que se deben entender para poder comprender correctamente el documento.

#### 3.1. Máquina Virtual

Una máquina virtual es un software que simula el funcionamiento de un ordenador, con su correspondiente sistema operativo pudiendo ejecutar programas como si fuera una computadora real. El sistema operativo que se está ejecutando no es capaz de saber si se encuentra en un software de virtualización o en un hardware real, esto es así ya que una máquina virtual transforma los dispositivos virtuales que ofrece el software con los dispositivos reales que tiene la máquina física en donde se está ejecutando, realizando una emulación de forma que sea totalmente compatible.

En la *ilustración 1* se puede observar la representación del funcionamiento de una máquina virtual.

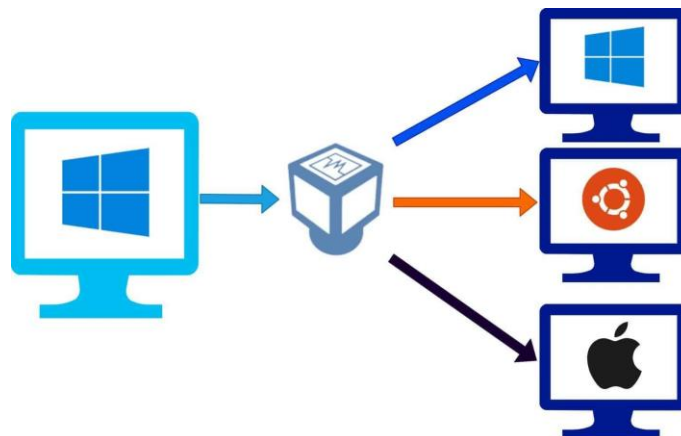


Ilustración 1: Representación de una máquina virtual.

### 3.2. VPN

Las siglas VPN significan Virtual Private Network, que en español se traduce a red privada virtual, recibe este nombre ya que es un sistema mediante la cual se permite una conexión segura de la red de área local (LAN) sobre una red pública como puede ser internet. Esto funciona realizando una conexión virtual punto a punto entre el dispositivo que se quiere conectar a la red local y el dispositivo que le permite conectarse, esta conexión por lo general está cifrada haciendo que sea una comunicación mucho más segura.

Este sistema se suele utilizar en entornos en los que se requiere conectarse a una red interna a través de internet, como por ejemplo el empleado de una empresa que quiere conectarse al centro de cómputo desde su casa, o como es el caso de este proyecto un usuario de la plataforma que quiere conectarse a la red interna en donde se están ejecutando los retos. En la *ilustración 2* se puede observar la representación del funcionamiento de una conexión VPN.

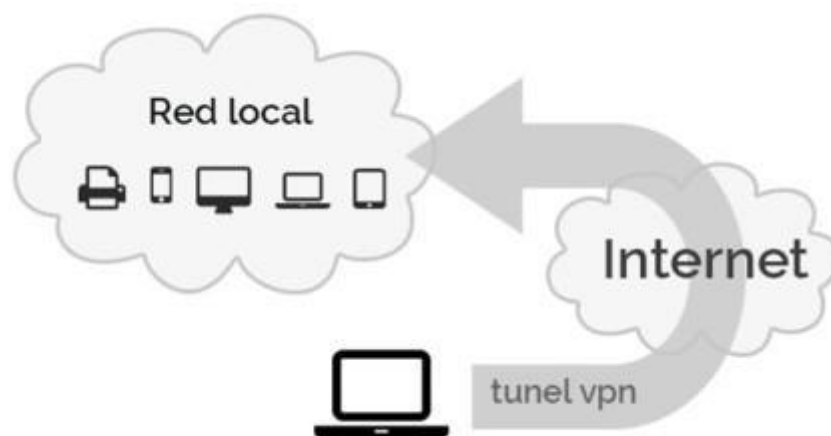


Ilustración 2: Conexión VPN.

### 3.3. CTF

Un CTF son las siglas en inglés de Capture The Flag, o en español, Capturar La Bandera. Aprovecharte de una vulnerabilidad de una página es ilegal, es por esto por lo que surgen este tipo de juegos, creados especialmente para probar las habilidades de hacking y poder practicar en un entorno controlado y seguro. Los CTFs son juegos diseñados con algún tipo de vulnerabilidad, haciendo que los jugadores investiguen y encuentren dichos fallos para ir encontrando las llamadas Flags y así conseguir resolver el reto. Este tipo de modularidad se ha popularizado mucho en los últimos años, haciendo que sea muy común que grandes empresas creen sus propios CTFs. Este tipo de retos por lo general suelen ser de las siguientes modalidades:

- **Ingeniería Inversa.**
- **Análisis forense.**
- **Criptografía.**
- **Pentesting Web.**
- **OSINT.**
- **Explotación.**

### 3.4. REST API

Una API de REST es una arquitectura software diseñada para sistemas distribuidos como Internet, fue creado en el año 2000 por Roy Fielding. Este tipo de arquitectura permite interactuar con una computadora o sistema de forma que estos comprendan cual es la solicitud y ejecutándola adecuadamente, para ello utiliza un conjunto de protocolos para integrar el software de las aplicaciones.

## 4. Técnicas y herramientas

---

En este apartado se van a definir cuáles son las técnicas y herramientas que se han utilizado para poder desarrollar el proyecto, desde la realización de los diferentes anexos hasta la programación del mismo.

### 4.1. Técnicas y herramientas usadas en el servicio web

#### 4.1.1 Python

Es un lenguaje de programación de alto nivel, siendo este de código abierto se utiliza para desarrollar multitud de plataformas debido a su gran versatilidad. Es un lenguaje multiparadigma, esto significa que soporta diferentes paradigmas de programación, como son la programación orientada a objetos, programación imperativa y programación funcional, permitiéndolo ser muy flexible y fácil de entender.

Se trata de un lenguaje interpretado, por lo que no se compila, sino que se interpreta en el momento de la ejecución haciendo que pueda ser más lento para diferentes funciones comparándolo con otros lenguajes de programación, aun así hoy en día se considera uno de los lenguajes de programación más completos, siendo este de los más utilizados en la industria.

Es por esto que se ha utilizado Python para crear determinadas funciones junto con el hecho de programar con el framework Django para el desarrollo de la página web, el cual está programado en Python. En la *ilustración 3* se puede observar un ejemplo de la utilización de Python en el desarrollo.

```

@login_required(login_url='login')
def ranking(request):
    """
    Obtiene una lista de todos los usuarios, los ordena por puntos y los muestra por pantalla en forma de tabla.
    """
    :param request: Objeto http
    :return: El render del template ranking.html
    """
    count=0
    page_obj=[]
    listaUsuarios = User.objects.all()
    listaUsuariosOrdenada = listaUsuarios.order_by('-alumno_puntos')
    myFilter = UserFilter(request.GET, queryset=listaUsuariosOrdenada)

    listaUsuariosOrdenadaFiltrada = myFilter.qs
    print(listaUsuariosOrdenadaFiltrada)

    if listaUsuariosOrdenadaFiltrada.count() == 1: #en caso de que se haya filtrado y solo haya un usuario se recorre la lista para calcular cual es su posición
        for iterator in listaUsuariosOrdenada:
            count+=1
            if iterator.username == listaUsuariosOrdenadaFiltrada[0].username:
                print(count)
                iterator.alumno.position = count
                page_obj.append(iterator)
                break
    else:
        for iterator in listaUsuariosOrdenadaFiltrada:
            count+=1
            iterator.alumno.position=count
            page_obj.append(iterator)

    paginator = Paginator(page_obj, 5)
    page_number = request.GET.get('page')
    page_obj = paginator.get_page(page_number)

```

Ilustración 3: Ejemplo de uso de Python.

#### 4.1.2. Django

Django es un framework de desarrollo web de código abierto programado en Python, utilizando el patrón de diseño Modelo Vista Controlador. En sus comienzos este framework fue creado para facilitar el desarrollo de sitios webs de noticias, pero en 2005 fue liberada al público bajo la licencia BSD.

A pesar de utilizar el patrón Modelo Vista Controlador los desarrolladores indican que su patrón es llamado MTV (Model Template View), ya que estos intentan alejarse de patrones de diseño, pretendiendo crear un sistema que funcione lo mejor posible y de la manera más intuitiva. El modelo MTV funciona de la siguiente manera:

- **Model:** Es la capa que tiene acceso a la base de datos, conteniendo toda la información de la misma.
- **Template:** Esta capa que en español significa plantilla, hace referencia a la representación visual de la interfaz gráfica del sistema, pudiendo manejar qué información se muestra y cómo, dependiendo del entorno.
- **View:** Es la capa que se encarga de controlar el acceso y la comunicación entre las capas Model y Template.

### 4.1.3. Bootstrap

Bootstrap es un framework CSS de código abierto que favorece el desarrollo web y la maquetación de páginas web desarrollado inicialmente por Twitter en 2011, pero poco tiempo después, se liberó bajo la licencia MIT y en febrero de 2012 ya era el proyecto más popular de GitHub.

Incluye numerosas herramientas y funcionalidades que permiten crear una web como por ejemplo plantillas de diseño basadas en HTML y CSS con la que es posible modificar tipografías, formularios, botones, tablas, navegaciones, menús desplegables, etc.

Se ha utilizado para crear las interfaces de la página web de una manera limpia y compatible con demás dispositivos. En la *ilustración 4* se puede observar el uso de Bootstrap para desarrollar la interfaz gráfica de la plataforma.



Máquina	Dificultad	Fecha Creada	Categorías	
 <a href="#">Mr Robot</a>	Media	June 15, 2022	Linux Top 10 Samba Dirbuster MySQL Crypto	<a href="#">Editar</a> <a href="#">Borrar</a> <a href="#">Compartir</a>
 <a href="#">Empire</a>	Media	June 26, 2022	Linux SQLi Tomcat	<a href="#">Editar</a> <a href="#">Borrar</a> <a href="#">Compartir</a>
 <a href="#">Noob</a>	Fácil	June 26, 2022	Linux Redes Top 10	<a href="#">Editar</a> <a href="#">Borrar</a> <a href="#">Compartir</a>
 <a href="#">Inferno</a>	Insana	June 26, 2022	MySQL CVE SQLi Windows	<a href="#">Editar</a> <a href="#">Borrar</a> <a href="#">Compartir</a>
 <a href="#">ChillHack</a>	Fácil	June 26, 2022	Linux Apache	<a href="#">Editar</a> <a href="#">Borrar</a> <a href="#">Compartir</a>

**Ilustración 4: Ejemplo de uso de Bootstrap en la plataforma.**

### 4.1.4. SQLite

Es una herramienta de código abierto para la gestión de bases de datos relacionales, creado por D. Richard Hipp en el año 2000. Una de las ventajas de este sistema es que la biblioteca SQLite se enlaza directamente con el programa, para así formar parte de el completamente, a diferencia de otros sistemas de bases de datos que los datos son independientes del programa con el que se comunica. Esto hace que tenga una menor latencia en el acceso a los datos, también producido por el hecho de que las llamadas a funciones son mucho más rápidas que la comunicación entre diferentes procesos.

Es por esto por lo que se ha usado SQLite como sistema de la base de datos de la plataforma.



#### 4.1.5. Docker

Docker es una plataforma creada con el fin de desarrollar, implementar y ejecutar aplicaciones dentro de contenedores de Linux. Estos contenedores se encargan de empaquetar software en unidades estandarizadas que incluyen todo lo necesario para que el software se ejecute, incluidas bibliotecas, herramientas de sistema, código y tiempo de ejecución.

Docker elimina las tareas de configuración repetitivas y es usado durante todas las etapas de desarrollo de aplicaciones de manera rápida, sencilla y portátil.

Docker a su vez, es un sistema operativo para contenedores. De manera similar a cómo una máquina virtual virtualiza el hardware del servidor, los contenedores virtualizan el sistema operativo de un servidor.

Este software ha sido usado para la utilización del servicio OpenVPN, de forma que se maneja su funcionamiento mediante la plataforma web, para que cualquier usuario pueda conectarse a este sistema. En la *ilustración 5* se puede ver una representación del uso de contenedores en Docker.

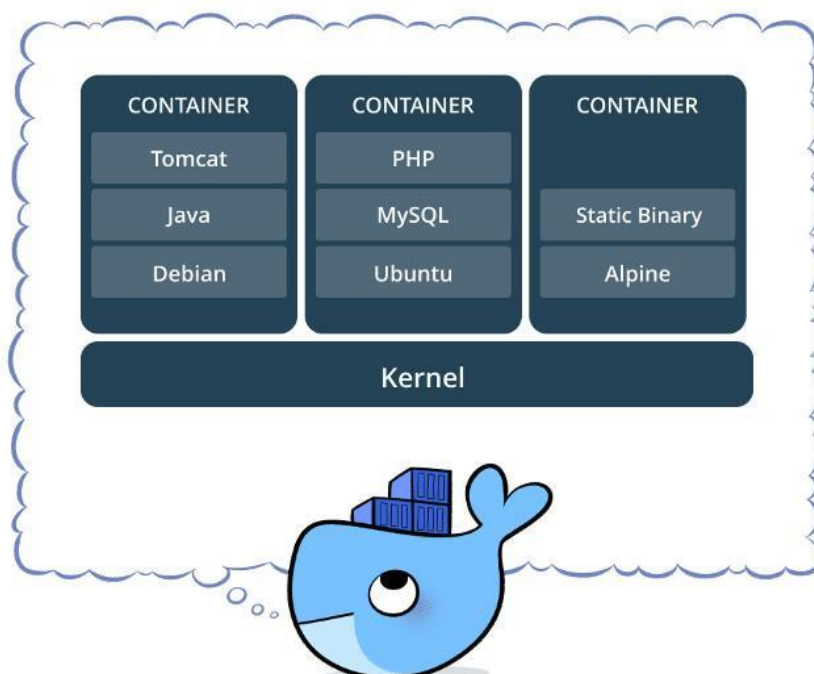


Ilustración 5: Ejemplo Docker.

#### 4.1.6. Oracle VM VirtualBox

Es un software perteneciente a Oracle cuya función es la virtualización para arquitecturas x86/amd64. Gracias a este software es posible la virtualización de sistemas operativos adicionales dentro de un dispositivo real, todo ello ofreciendo unas particularidades únicas del propio software de Virtualbox.

Este programa se utiliza para la virtualización de los retos de la plataforma, junto con la virtualización de la red interna en donde se están ejecutando y a la cual los usuarios pueden acceder a través de OpenVPN. En la *ilustración 6* se puede observar la interfaz de Oracle Virtual Box.

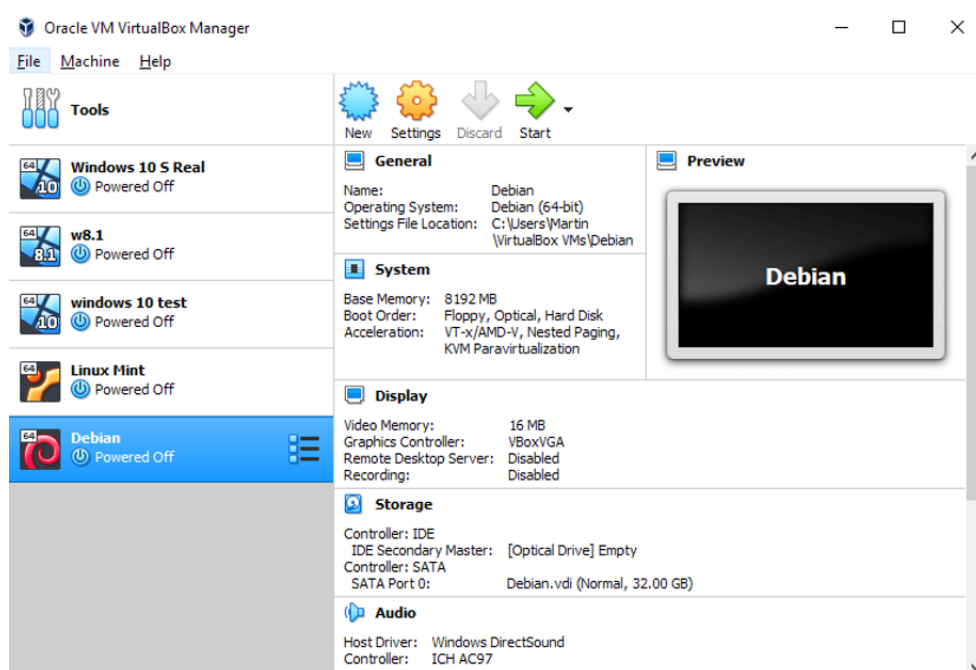


Ilustración 6: Interfaz de Oracle Virtualbox.

#### 4.1.7. OpenVPN

Es una herramienta diseñada por James Yonan en el año 2001 para realizar conectividad basada en el software libre SSL, VPN. Realiza la conexión punto a punto con validación jerárquica entre usuarios y servidores a través de certificados. Se considera uno de los softwares más versátiles debido a su gran posibilidad de configuración, encriptación y facilidad de uso. Para realizar la comunicación entre los nodos se utiliza el protocolo UDP y por lo general se suele usar el puerto 1194 UDP para usar este protocolo.

Existen dos métodos para encriptar la información que se transmite a través de este sistema:

- **Cifrado por claves SSL/TLS +RSA:** es la forma más segura de transmitir información, cada servidor y clientes poseen dos claves, una pública y otra privada, siendo la clave pública utilizada para cifrar los datos y la que utilizarán los clientes para conectarse, mientras que su clave privada es la que se utiliza para descifrar.
- **Claves pre-compartidas:** utiliza cifrado simétrico de forma que se usan mecanismos para intercambiar una clave en los dos extremos de la conexión, por lo que cualquier nodo que tenga esa clave podrá descifrar el tráfico en esta red.

En la *ilustración 7* se puede observar una representación del uso de OpenVPN en el proyecto.

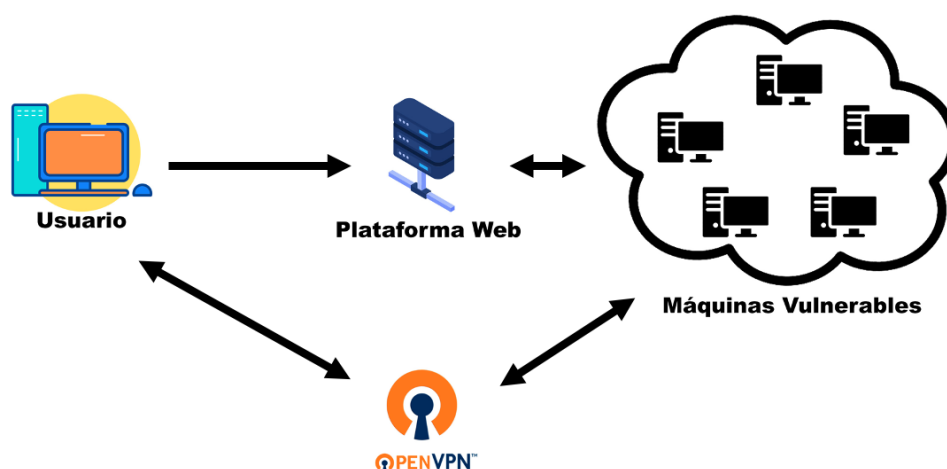


Ilustración 7: Uso de OpenVPN en el proyecto.

### 4.1.8. Visual Studio Code

Visual Studio Code es un editor de código desarrollado por Microsoft. Es un software libre y multiplataforma, es distribuido mediante una licencia de producto tradicional de Microsoft.

Cuenta con un soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código, soporte para múltiples lenguajes de programación. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias.

Visual Studio Code fue anunciado el 29 de abril de 2015 por Microsoft en la conferencia Build de 2015. Ha sido utilizado para escribir todo el código del proyecto.

En la *ilustración 8* se puede ver un ejemplo de la interfaz gráfica de Visual Studio Code.

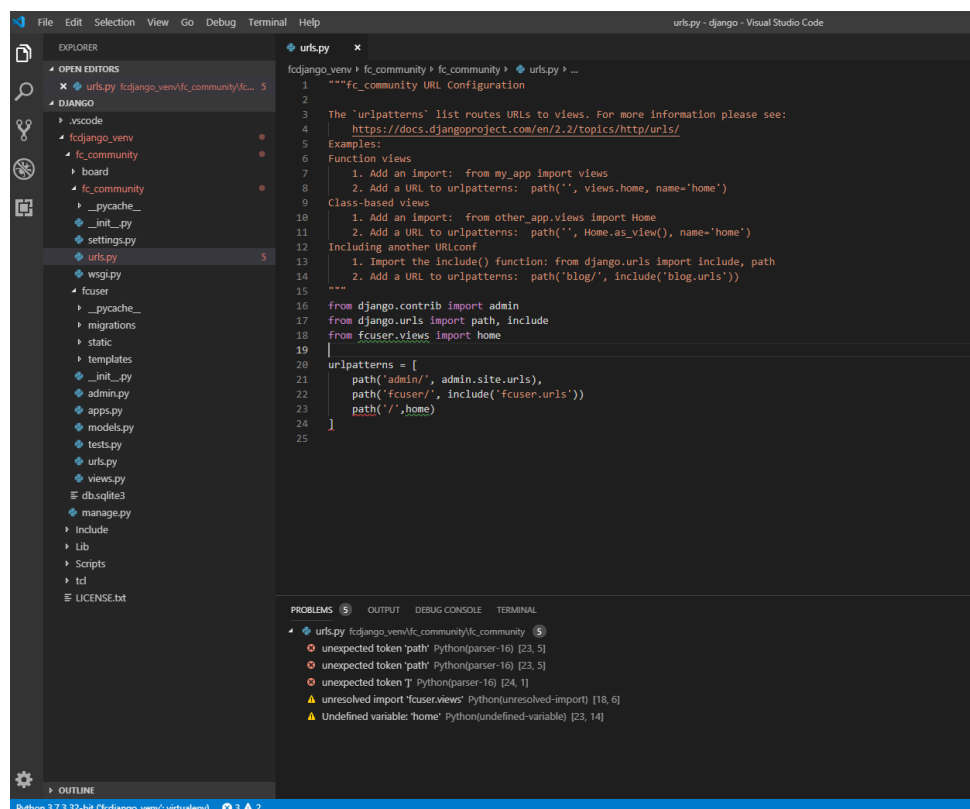


Ilustración 8: Interfaz Visual Studio Code.

#### 4.1.9. VulnHub

VulnHub es una plataforma online en donde usuarios suben CTFs que han creado ellos mismos, es decir, suben máquinas virtuales para que las personas que quieran practicar se las descarguen y se las instalen en sus respectivos software de virtualización.

Es de VulnHub de donde se van a descargar las máquinas virtuales que se ejecutarán en la red interna para poder probar la plataforma con retos funcionales.

En la *ilustración 9* se puede ver la página web de VulnHub.com.

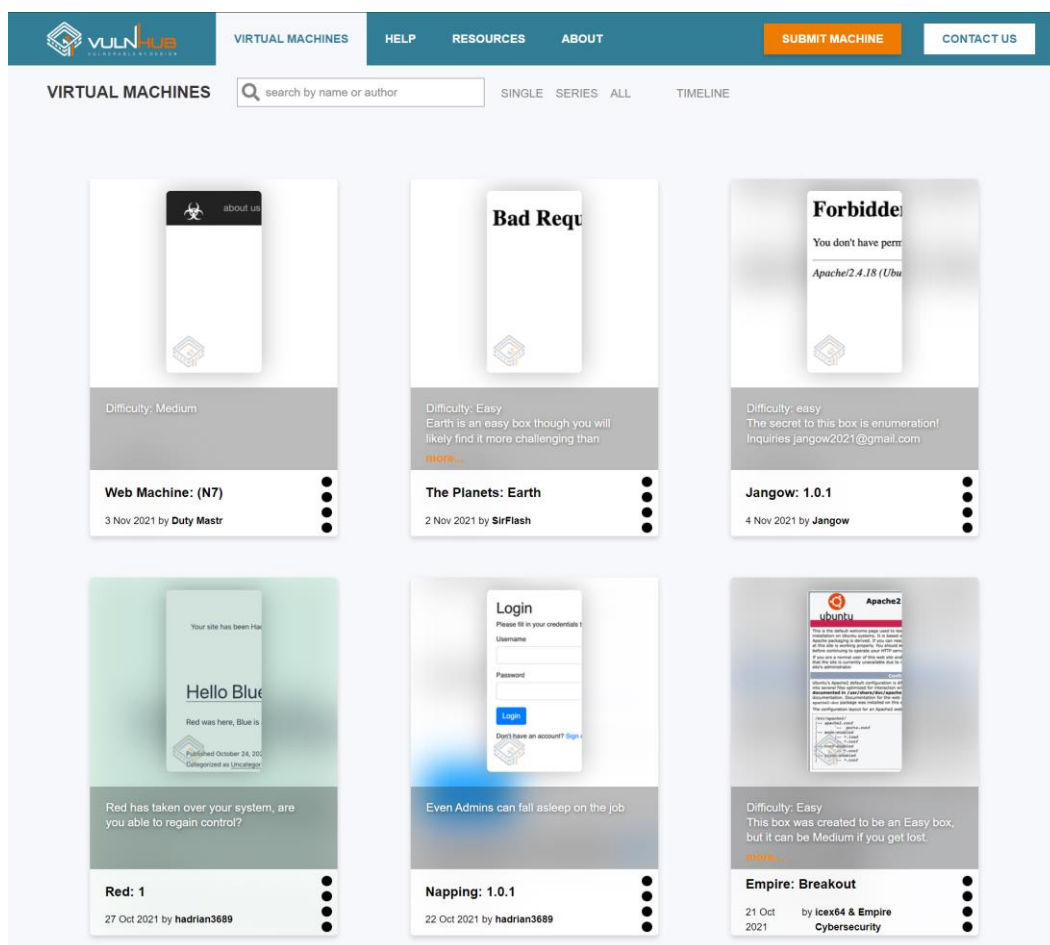


Ilustración 9: Página web de VulnHub.com.

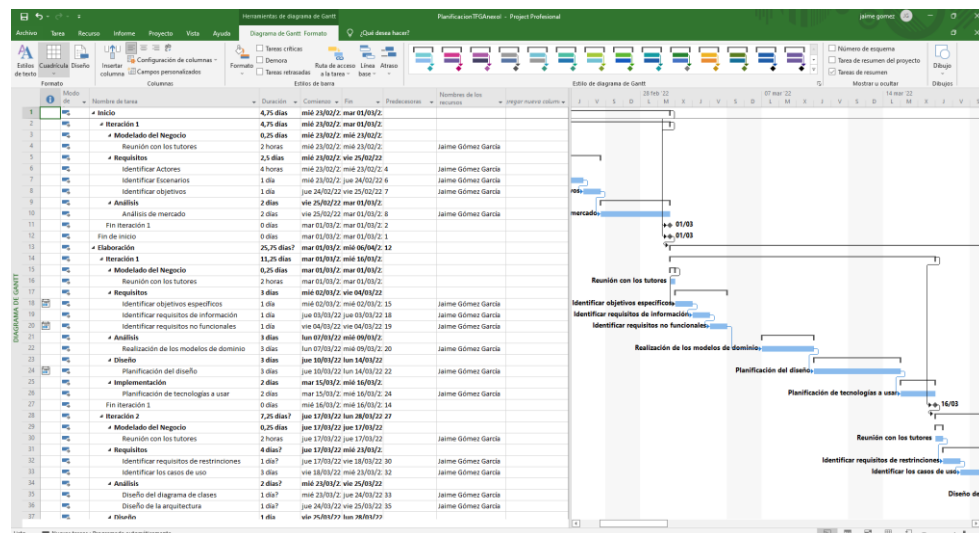
## 4.2. Herramientas CASE

### 4.2.1. Microsoft Project

Este programa es un software desarrollado por Microsoft que se utiliza ampliamente para la administración de proyectos especialmente para tareas como el reparto de recursos, seguimiento del proyecto, administración del presupuesto, analizar las cargas de trabajo y el desarrollo de los planes.

Este software se ha usado para la realización del Anexo I, concretamente la planificación temporal del trabajo utilizando el proceso unificado.

En la *ilustración 10* se puede observar el uso de Microsoft Project en el desarrollo de la planificación temporal.



**Ilustración 10: Interfaz de Microsoft Project.**

### 4.2.2. Visual Paradigm

Visual Paradigm es un software altamente utilizado para realizar las tareas relacionadas con la ingeniería del software, siendo especialmente útil para soportar el ciclo de vida completo del desarrollo de la plataforma, como por ejemplo para realizar el análisis de los requisitos, realizar los diseños arquitectónicos, diagramas de casos de uso, etc.

En concreto se ha usado durante toda la fase de realización del plan del proyecto, especificación de requisitos, análisis y diseño del sistema software, para ello se ha utilizado la licencia que nos provee la Universidad de Salamanca. En la *ilustración 11* se puede ver un ejemplo de la interfaz gráfica de Visual Paradigm.

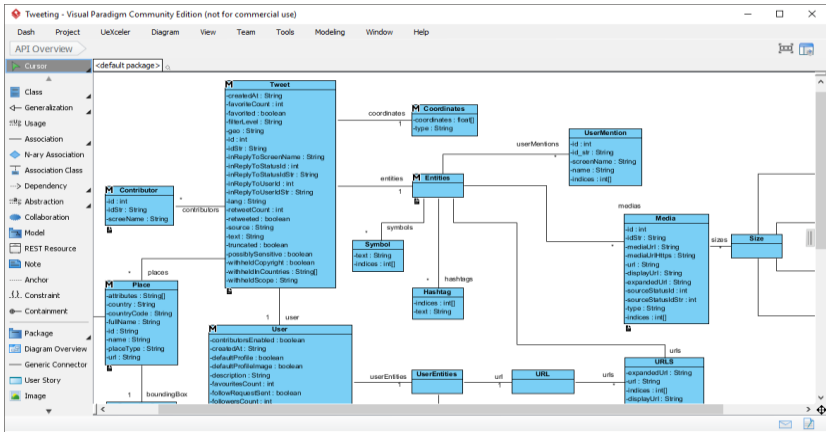


Ilustración 11: Interfaz Visual Paradigm.

### 4.2.3. EZEstimate

Es una herramienta utilizada para realizar la estimación de esfuerzo del proyecto, para ello se basa en la utilización de los puntos de casos de uso para poder crear un análisis y así estimar cual será la duración del proyecto. En la *ilustración 12* se puede el uso de EZEstimate en la estimación de esfuerzo del proyecto.

The screenshot shows the EZEstimate software interface. The main window displays project estimation data for the module 'Gestión de Autenticación'. The interface includes a 'Module' dropdown, 'Add Actor / Use case' section, 'Estimation Summary' section, and 'Use case / Actor List' table.

**Module:** Gestión de Autenticación

**Add Actor / Use case:** Actor / Use case Name, Select Type, Complexity, Add

**Estimation Summary:**

Estimation Summary	Value
UAW	9
UUCW	145
UUPC = UAW + UUCW	154
TFactor	46
EFactor	22
TCF = 0.6 + (.01*TFactor)	1.06
EF = 1.4 + (.03*EFactor)	0.74
UCP = UUPC*TCF*EF	120.7976
Total Effort@ 7 Hrs/UCP	845.5832

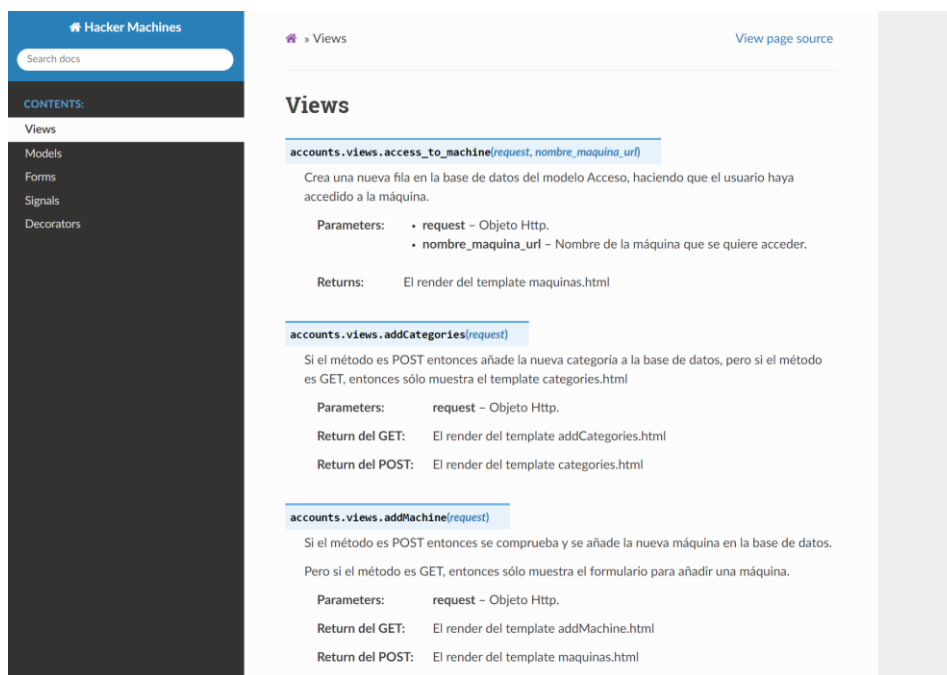
**Use case / Actor List (Double click to delete):**

Id	Module	Type	Name	complexity
1	Gestión de Aut...	Actor	Usuario Loguea...	Complex
10	Gestión de Usu...	Usecase	Mostrar Ranking	Simple
11	Gestión de Usu...	Usecase	Ver Usuario	Simple
12	Gestión de Usu...	Usecase	Buscar Usuario	Simple
13	Gestión de Usu...	Usecase	Conceder Rol d...	Simple
14	Gestión de Usu...	Usecase	Quitar Rol de A...	Simple
15	Gestión de VPN	Usecase	Descargar Arch...	Simple
16	Gestión de VPN	Usecase	Conectarse a lo...	Simple
17	Gestión de Cat...	Usecase	Ver Categorías	Simple
18	Gestión de Cat...	Usecase	Añadir Categorí...	Simple
19	Gestión de Cat...	Usecase	Editar Categoría	Simple
2	Gestión de Aut...	Actor	Usuario No Log...	Complex
20	Gestión de Cat...	Usecase	Eliminar Catego...	Simple
21	Gestión de Retos	Usecase	Ver Retos	Simple
22	Gestión de Retos	Usecase	Añadir Reto	Simple
23	Gestión de Retos	Usecase	Editar Reto	Simple
24	Gestión de Retos	Usecase	Eliminar Reto	Simple
25	Gestión de Retos	Usecase	Harar Votación R...	Simple

Ilustración 12: Interfaz EZEstimate.

## 4.2.4. Sphinx

Es un software que genera documentación de código a sitios web HTML, es por esto que se ha utilizado como ayuda para la generación de la documentación del código del proyecto. En la *ilustración 13* se puede ver la utilización de Sphinx para la generación de la documentación.



**Ilustración 13: Documentación del proyecto con Sphinx.**

## 4.3. Herramientas para el control de versiones

### 4.3.1 GIT

Es un software diseñado por Linus Torvald desarrollado para realizar el control de versiones, cumpliendo con un alto grado de eficiencia, confiabilidad y compatibilidad. Su propósito es el de llevar a cabo un registro de todos los cambios de los archivos de un proyecto, para así poder coordinarlo entre varios o un solo programador a lo largo del tiempo.

Durante el desarrollo del proyecto se ha utilizado GIT de manera muy frecuente para poder mantener una copia de seguridad pudiendo así analizar y programar el proyecto a lo largo del tiempo, concretamente se ha usado GitHub como servicio para alojar estos cambios.



### 4.3.2. GitHub

GitHub es un portal creado para alojar el código de las aplicaciones de cualquier desarrollador, y que fue comprada por Microsoft en junio del 2018. La plataforma está creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, y que como usuario no sólo se pueda descargar la aplicación, sino también entrar a su perfil para leer sobre ella o colaborar con su desarrollo.

Así pues, Github es un portal para gestionar las aplicaciones que utilizan el sistema Git. Además de permitir mirar el código y descargar las diferentes versiones de una aplicación, la plataforma también permite conectar desarrolladores con usuarios para que estos puedan colaborar mejorando la aplicación. En la *ilustración 14* se puede ver la interfaz de GitHub.

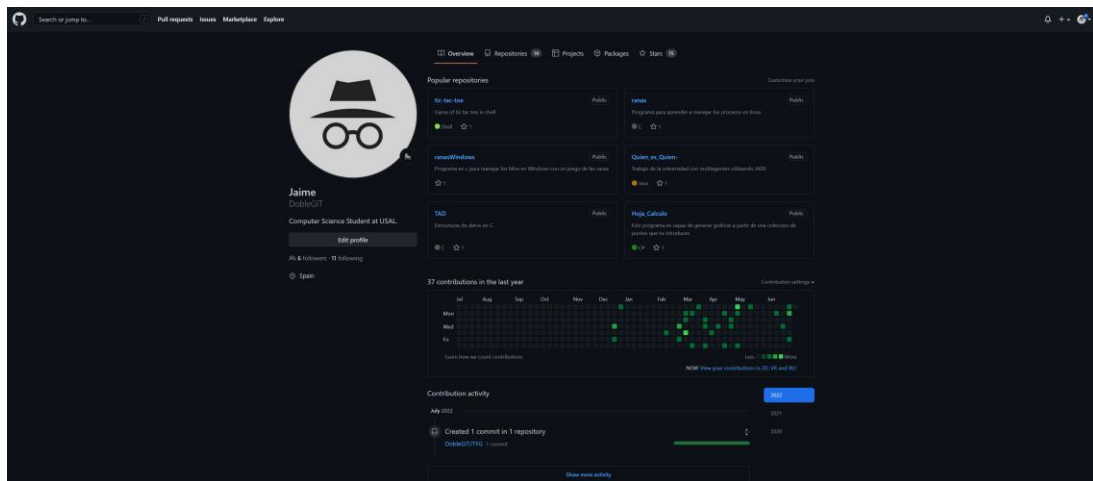


Ilustración 14: Interfaz GitHub.

## 5. Aspectos relevantes del desarrollo

---

### 5.1. Marco de trabajo

Para el desarrollo del proyecto se ha utilizado el marco de trabajo del Proceso Unificado, siendo este fundamental para la organización de las tareas y los recursos a lo largo de todo el desarrollo de la plataforma. El proceso Unificado tiene las siguientes características:

- Está basado en componentes, lo cual hace que sea muy flexible y pueda especializarse y extenderse para una gran cantidad de proyectos de software.
- Es un proceso guiado por los casos de uso, haciendo que a partir de estos se cree un conjunto de modelos de diseño y de implementación, por lo que los casos de uso se consideran un hilo conductor para avanzar con el desarrollo del proyecto.
- Se centra en la arquitectura.
- Es un proceso iterativo e incremental.

#### Ciclos de vida del proceso unificado

El proceso Unificado se realiza en diferentes ciclos de vida y al finalizar cada uno de ellos se obtiene una versión del sistema.

Este ciclo de vida se divide en diferentes fases, para así conseguir realizar un proceso interactivo e incremental, estas fases son:

- **Inicio:** durante el proceso de inicio se deben definir cuáles son los objetivos del proyecto, explorando posibles soluciones y arquitecturas que se puedan implementar. Se debe acabar con esta fase teniendo una idea de cuál va a ser la arquitectura a utilizar, una planificación de las iteraciones junto con un análisis del conjunto de necesidades y requisitos del proyecto.
- **Elaboración:** a lo largo de la fase de elaboración se especifican cuáles son los casos de uso y se planifica la arquitectura del sistema, para así conseguir tener un plan detallado para las siguientes iteraciones. Al

finalizar esta fase de debe tener planteado el plan del proyecto, la línea base de la arquitectura y la mayoría de los casos de uso.

- **Construcción:** al finalizar esta fase el proyecto debe de tener los casos de uso implementados, aunque es posible que tengan algunos errores que haya que solucionar mas adelante. La finalidad de esta fase es la de construcción del producto software en base a la línea de la arquitectura que se ha ido siguiendo.
- **Transición:** cuando acabe esta fase se debe tener una versión estable del producto software, aunque a lo largo de esta fase se continúa implementando características.

En la *ilustración 15* se muestran las diferentes fases e iteraciones del proceso unificado.

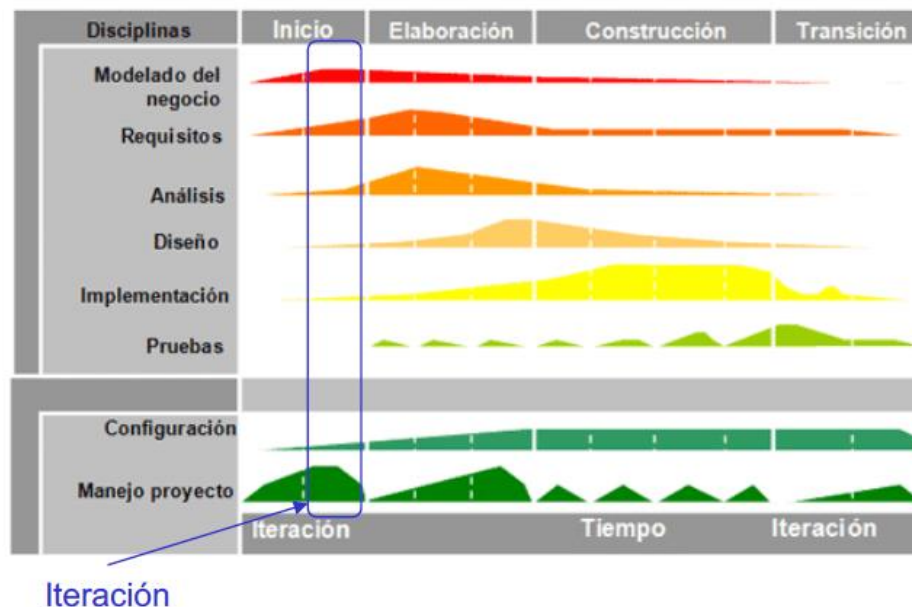


Ilustración 15: Proceso Unificado.

## 5.2. Estimación de esfuerzo

Para poder realizar la estimación de esfuerzo se ha utilizado la métrica UCP (Use Case Point), la cual tiene en cuenta los actores, escenarios, factores técnicos y de entorno, haciendo uso de tres variables:

- **UUCP (Unadjusted Use Case Weight):** Que hace uso de otras dos variables:
  - **UUCW (Unadjusted Use Case Weight):** Considera el número y la complejidad de los casos de uso.
  - **UAW (Unadjusted Actor Weight):** Considera el número y la complejidad de los actores.
- **TCF (Technical Complexity Factor):** Para calcular esta variable se hace uso de trece factores de complejidad técnica.
- **ECF (Environment Complexity Factor):** Esta variable se calcula con ocho factores de complejidad de entorno.

Una vez calculadas estas tres variables se utiliza la siguiente ecuación para calcular el UCP:

$$UCP = UUCP * TCF * ECF$$

Por último, se calcula la estimación de esfuerzo a partir del UCP con la siguiente ecuación, este esfuerzo se da en número de horas por persona:

$$Esfuerzo = UCP * F$$

Para poder realizar la estimación de esfuerzo se ha utilizado la herramienta EZEstimate, en la que se han introducido las variables descritas anteriormente.

El resultado obtenido de realizar este análisis es el que se muestra en la *ilustración 16*, en el cual podemos llegar a la conclusión de que la duración estimada del proyecto es de aproximadamente 845 horas, es decir, si se trabaja un total de 8 horas al día nos salen 105 días de trabajo para poder concluir el proyecto en su totalidad.

EZEstimate - C:\Users\jaime\Desktop\TFG\TFGANexo1\EZEstimate.ezp

File Settings Help

**Module**  
 Gestión de Autenticación  
 Add Module Delete

**Summary**  
 Total Modules 5 Excel Report Generate Report  
 Use cases Simple 29 Average 0 Complex 0  
 Actors Simple 0 Average 0 Complex 3

**Add Actor / Use case**  
 Actor / Use case Name Select Type Complexity Add

**Tech / Env Factors**  
 Set Tech Factor  
 Set Env Factors

**Estimation Summary**  
 UAW 9  
 UUCW 145  
 UUPC = UAW + UUCW 154  
 TFactor 46  
 EFactor 22  
 TCF = 0.6 + (.01\*TFactor) 1.06  
 EF = 1.4 + (-0.03\*EFactor) 0.74  
 UCP = UUPC\*TCT\*EF 120.7976  
**Total Effort@** 7 Hrs/UCP 845.5832

**Use case / Actor List** ( Double click to delete )
 

Id	Module	Type	Name	complexity
1	Gestión de Aut...	Actor	Usuario Loguea...	Complex
10	Gestión de Usu...	Usecase	Mostrar Ranking	Simple
11	Gestión de Usu...	Usecase	Ver Usuario	Simple
12	Gestión de Usu...	Usecase	Buscar Usuario	Simple
13	Gestión de Usu...	Usecase	Conceder Rol d...	Simple
14	Gestión de Usu...	Usecase	Quitar Rol de A...	Simple
15	Gestión de VPN	Usecase	Descargar Arch...	Simple
16	Gestión de VPN	Usecase	Conectarse a lo...	Simple
17	Gestión de Cat...	Usecase	Ver Categorías	Simple
18	Gestión de Cat...	Usecase	Añadir Categorí...	Simple
19	Gestión de Cat...	Usecase	Editar Categoría	Simple
2	Gestión de Aut...	Actor	Usuario No Log...	Complex
20	Gestión de Cat...	Usecase	Eliminar Catego...	Simple
21	Gestión de Retos	Usecase	Ver Retos	Simple
22	Gestión de Retos	Usecase	Añadir Reto	Simple
23	Gestión de Retos	Usecase	Editar Reto	Simple
24	Gestión de Retos	Usecase	Eliminar Reto	Simple
25	Gestión de Retos	Usecase	Hacer Visible R	Simple

Ilustración 16: Estimación de esfuerzo EZEstimate.

### 5.3. Planificación temporal

Para realizar la planificación temporal también nos basamos en el Proceso Unificado, el cual nos ayuda a determinar cuáles van a ser las tareas que se van a realizar, cuánto van a durar y en qué orden se van a realizar para evitar el mínimo bloqueo entre ellas.

Esta planificación es la base de todo el proyecto, puesto que nos ayuda a examinar si las tareas que se están realizando están cumpliendo con la duración estimada o por el contrario si necesitamos reestructurarlas sabiendo en que tareas podemos asignar más o menos duración.

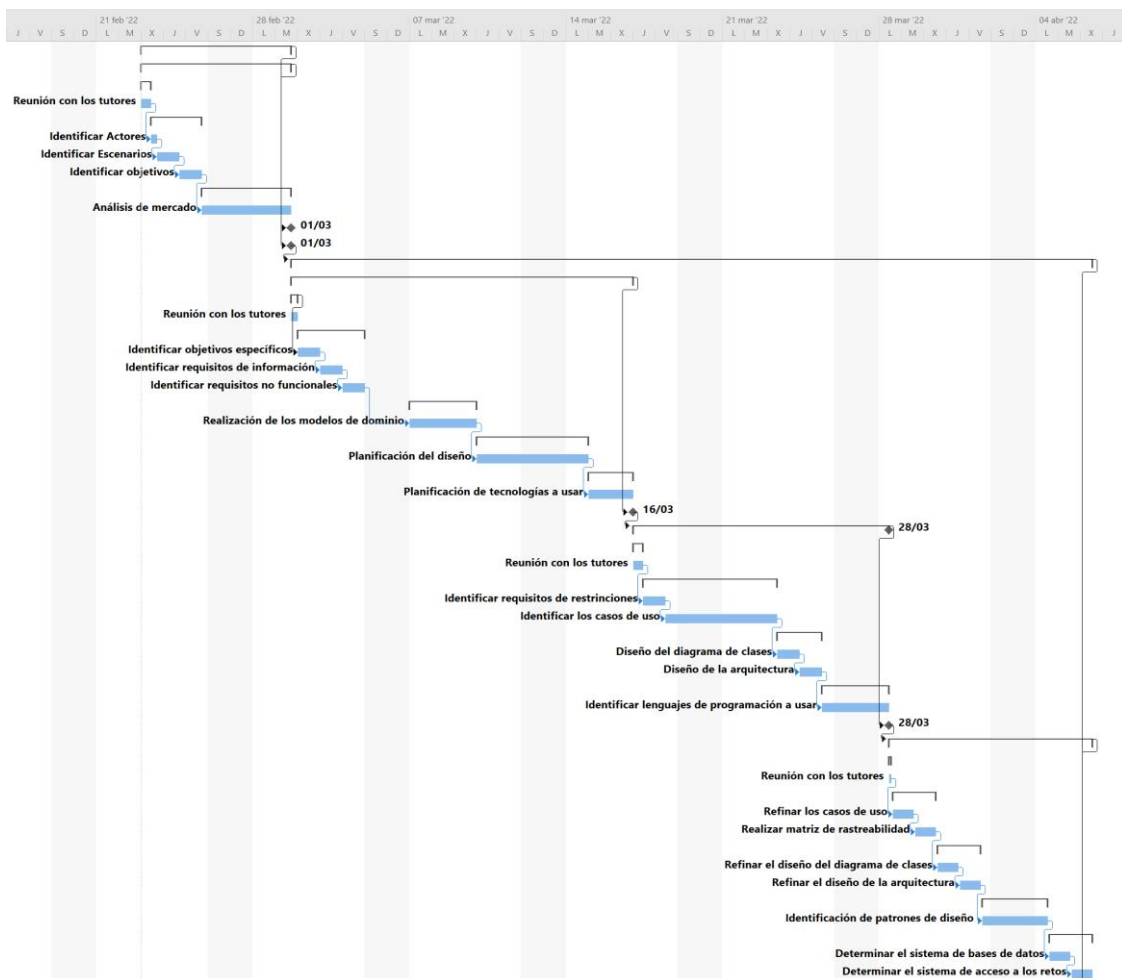
A continuación, se muestra cómo se han planificado cada una de las estas en las diferentes fases, si se desea conocer más información sobre la planificación temporal realizada se puede consultar el Anexo I – Plan de Proyecto Software. En la *ilustración 17* se puede ver la primera iteración de la etapa de Inicio de la planificación temporal.

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
• Inicio	4,75 días	mié 23/02/22	mar 01/03/22		
• Iteración 1	4,75 días	mié 23/02/22	mar 01/03/22		
• Modelado del Negocio	0,25 días	mié 23/02/22	mié 23/02/22		
Reunión con los tutores	2 horas	mié 23/02/22	mié 23/02/22		Jaime Gómez García
• Requisitos	2,5 días	mié 23/02/22	vie 25/02/22		
Identificar Actores	4 horas	mié 23/02/22	mié 23/02/22	4	Jaime Gómez García
Identificar Escenarios	1 día	mié 23/02/22	jue 24/02/22	6	Jaime Gómez García
Identificar objetivos	1 día	jue 24/02/22	vie 25/02/22	7	Jaime Gómez García
• Análisis	2 días	vie 25/02/22	mar 01/03/22		
Análisis de mercado	2 días	vie 25/02/22	mar 01/03/22	8	Jaime Gómez García
Fin iteración 1	0 días	mar 01/03/22	mar 01/03/22	2	
Fin de inicio	0 días	mar 01/03/22	mar 01/03/22	1	

**Ilustración 17: Iteración 1 etapa de inicio.**

Seguidamente podemos observar cual es el diagrama de Grantt generado, esto nos permite comprobar el reparto de tareas a lo largo del tiempo, viendo así las dependencias que existen entre ellas pudiendo optimizar el tiempo y los recursos de la manera más adecuada.

En la *ilustración 18* se puede ver un ejemplo del diagrama de Grantt generado con la herramienta Microsoft Project.



**Ilustración 18: Ejemplo diagrama de Grantt.**

## 5.4. Especificación de requisitos

Uno de los primeros pasos y de los más importantes a la hora de analizar un proyecto de estas dimensiones es la especificación de los requisitos, puesto que estos nos ayudan en gran medida a conocer cuáles pueden ser los objetivos, las necesidades, conocer a quien va dirigido la plataforma, cómo van a interactuar con ella, etc.

A continuación, se muestran las diferentes etapas de la especificación de requisitos, que se han desarrollado siguiendo la metodología de Duran y Bernández. Para más información se puede consultar el Anexo II: Especificación de Requisitos.

### 5.4.1. Participantes

El proyecto está formado por 4 participantes, los cuales son un alumno y tres tutores:

- **Jaime Gómez García.**
- **André Filipe Sales Mendes.**
- **Gabriel Villarrubia González.**
- **Juan Francisco De Paz Santana.**

### 5.4.2. Objetivos del sistema

Los objetivos del sistema son los siguientes:

- **Gestión de usuarios.**
- **Gestión de roles.**
- **Gestión de respuestas.**
- **Gestión de categorías**
- **Gestión de VPN.**
- **Organización del ranking.**
- **Gestión de roles.**
- **Sistema de puntos.**
- **Búsqueda de usuarios.**



A continuación, se muestra el ejemplo de especificación del objetivo Gestión de Usuarios:

<b>OBJ-0001</b>	<b>Gestión de Usuarios</b>
<b>Versión</b>	1.0 (07/07/2022)
<b>Autores</b>	<ul style="list-style-type: none"><li>• Jaime Gómez García</li></ul>
<b>Fuentes</b>	<ul style="list-style-type: none"><li>• André Filipe Sales Mendes</li><li>• Gabriel Villarrubia González</li><li>• Juan Francisco de Paz Santana</li></ul>
<b>Descripción</b>	El sistema debe permitir la creación, modificación y consulta de usuarios, tanto administradores como alumnos.
<b>Subobjetivos</b>	Ninguno
<b>Importancia</b>	Vital
<b>Urgencia</b>	Inmediatamente
<b>Estado</b>	Validado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	Ninguno

Tabla 1: Objetivo Gestión de Usuarios.

#### 5.4.3. Requisitos de información

En el siguiente apartado se especifican los requisitos de información del sistema, los cuales se deben almacenar y gestionar en el sistema.

- **Información sobre retos.**
- **Información de categorías.**
- **Información de ranking.**
- **Información sobre VPN.**
- **Información de usuarios.**
- **Información de puntos.**
- **Información respecto al reto accedido por el usuario.**

A continuación, se muestra el ejemplo de información respecto al reto accedido por el usuario:

<b>IRQ-0007</b>	<b>Información respecto al reto accedido por el usuario</b>
<b>Versión</b>	1.0 (07/07/2022)
<b>Autores</b>	<ul style="list-style-type: none"> <li>• Jaime Gómez García</li> </ul>
<b>Fuentes</b>	<ul style="list-style-type: none"> <li>• André Filipe Sales Mendes</li> <li>• Gabriel Villarrubia González</li> <li>• Juan Francisco de Paz Santana</li> </ul>
<b>Dependencias</b>	<ul style="list-style-type: none"> <li>• [OBJ-0001] Gestión de Usuarios</li> <li>• [OBJ-0002] Gestión de retos</li> <li>• [OBJ-0003] Gestión de respuestas</li> </ul>
<b>Descripción</b>	El sistema debe almacenar información relativa al reto al que ha accedido el usuario. En concreto:
<b>Datos específicos</b>	<ul style="list-style-type: none"> <li>• Id (Único)</li> <li>• Fecha de acceso</li> <li>• Fecha de finalización</li> <li>• Tiempo que ha tardado</li> <li>• User Flag correcta</li> <li>• Root Flag correcta</li> <li>• Completada</li> </ul>
<b>Importancia</b>	Vital
<b>Urgencia</b>	Inmediatamente
<b>Estado</b>	Validado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	Ninguno

Tabla 2: Información respecto al reto accedido por el usuario.

#### 5.4.4. Requisitos Funcionales

Estos requisitos funcionales tienen como objetivo determinar cuál es el comportamiento que debe de tener el sistema en relación a las interacciones de los diferentes actores. En la *ilustración 19* se muestran cuáles son los diagramas de paquetes que se han establecido para poder organizar y gestionar la estructura del proyecto.

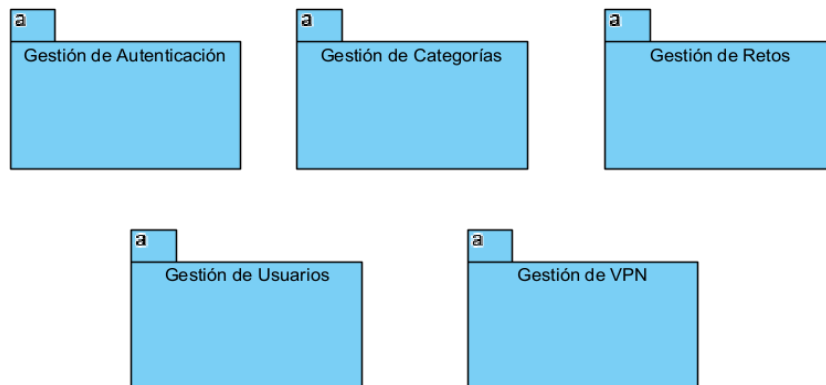


Ilustración 19: Diagrama de paquetes.

Posteriormente se muestra los actores que van a interactuar con el sistema:

- **Usuario No Logueado:** este actor es un usuario que no ha realizado la autenticación para poder acceder al sistema.
- **Usuario Logueado:** este actor es un usuario que ha iniciado sesión, teniendo así acceso a la plataforma para poder completar los retos.
- **Administrador:** este actor es un usuario con permisos adicionales para gestionar los retos y las categorías.

En la *ilustración 20* se puede ver la jerarquía de los actores del sistema.

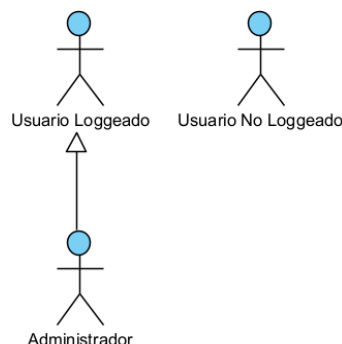
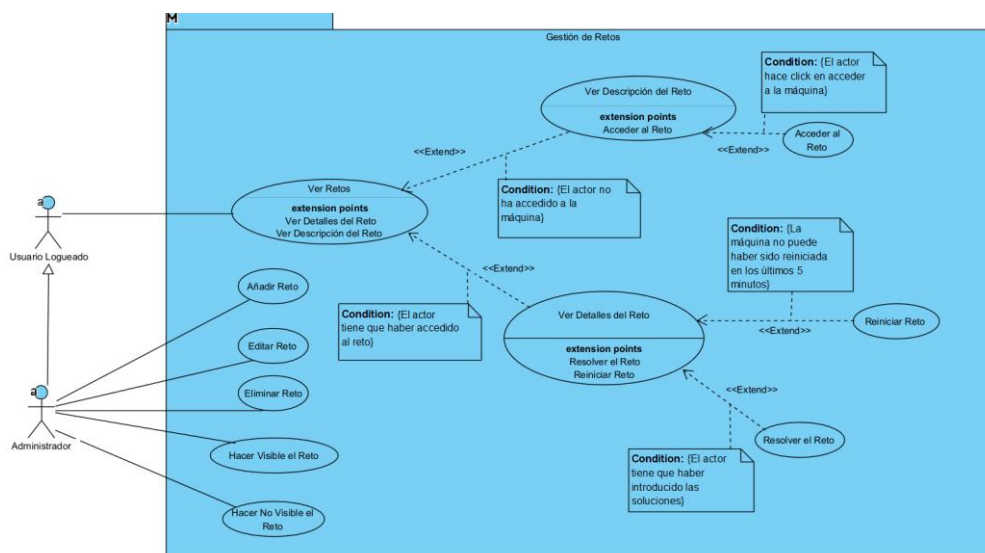


Ilustración 20: Actores del sistema.

Finalmente se especifican los diagramas de casos de uso del sistema, junto con las tablas que especifican cada uno de ellos, indicando sus acciones, dependencias, precondiciones, etc. En la *ilustración 21* se puede ver el diagrama de casos de uso del paquete Gestión de Retos.



**Ilustración 21: Paquete Gestión de Retos.**

Podemos observar en la siguiente tabla la especificación del caso de uso [UC-0029] Acceder al Reto:

UC-0029	Acceder al Reto
Versión	1.0 (07/07/2022)
Autores	<ul style="list-style-type: none"> <li>Jaime Gómez García</li> </ul>
Fuentes	<ul style="list-style-type: none"> <li>André Filipe Sales Mendes</li> <li>Gabriel Villarrubia González</li> <li>Juan Francisco de Paz Santana</li> </ul>
Dependencias	<ul style="list-style-type: none"> <li>[OBJ-0002] Gestión de retos</li> <li>[OBJ-0004] Gestión de categorías</li> <li>[OBJ-0007] Gestión de roles</li> <li>[IRQ-0001] Información sobre retos</li> <li>[IRQ-0002] Información de categorías</li> </ul>
Descripción	El sistema se debe comportar como se describe en este caso de uso, en el momento en el que el actor [ACT-0001] Usuario Logueado quiere acceder al reto para proceder a resolverlo.
Precondición	El usuario tiene que estar logueado y haber escogido un reto.

<b>Secuencia Normal</b>	<b>Paso</b>	<b>Acción</b>
	<b>1</b>	El actor [ACT-0001] Usuario Logueado le indica al sistema que quiere acceder a una máquina virtual.
	<b>2</b>	El sistema le proporciona acceso a la máquina virtual, redireccionando al actor [ACT-0001] Usuario Logueado a la pantalla para poder conectarse y resolverla.
<b>Postcondición</b>	El sistema le da acceso al reto, redireccionando al usuario a otra página para que pueda conectarse a la la máquina virtual.	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	-	-
<b>Rendimiento</b>	<b>Paso</b>	<b>Tiempo Máximo</b>
	-	-
<b>Frecuencia esperada</b>	4 veces por hora(s)	
<b>Importancia</b>	Vital	
<b>Urgencia</b>	Inmediatamente	
<b>Estado</b>	Validado	
<b>Estabilidad</b>	Alta	
<b>Comentarios</b>	Ninguno	

Tabla 3: Caso de Uso Acceder al Reto.

#### 5.4.5. Requisitos no funcionales

En el desarrollo del proyecto no sólo es importante los requisitos funcionales, sino también los no funcionales, ya que estos son ayudan en gran medida las restricciones que no están relacionadas con la funcionalidad del sistema.

En este caso se han definido los siguientes requisitos no funcionales:

- **Disponibilidad:** el sistema debe de tener un alto grado de disponibilidad, ya que queremos que los retos sean accesibles y utilizables por cualquier usuario en todo momento.
- **Usabilidad:** se debe de tener una interfaz agradable que facilite a los usuarios el proceso de acceso y resolución de los retos.
- **Funcionalidad:** el sistema debe ser capaz de satisfacer las funcionalidades para cumplir con las necesidades de los usuarios.

- **Mantenibilidad:** la plataforma debe ser capaz de mantenerse estable ante cambios que puedan surgir para mejorar o mantener las funcionalidades del mismo.
- **Seguridad:** puesto que la plataforma va a ser usada para aprender ciberseguridad y gran parte de los usuarios tendrán bastante conocimientos de esta materia es conveniente que el sistema sea seguro ante intentos de hackeos por parte de ciberdelincuentes.
- **Concurrencia:** uno de los requisitos más importantes es la concurrencia, ya que es la mayoría de retos serán utilizados a la vez por varios usuarios del sistema.

## 5.5. Análisis de Requisitos

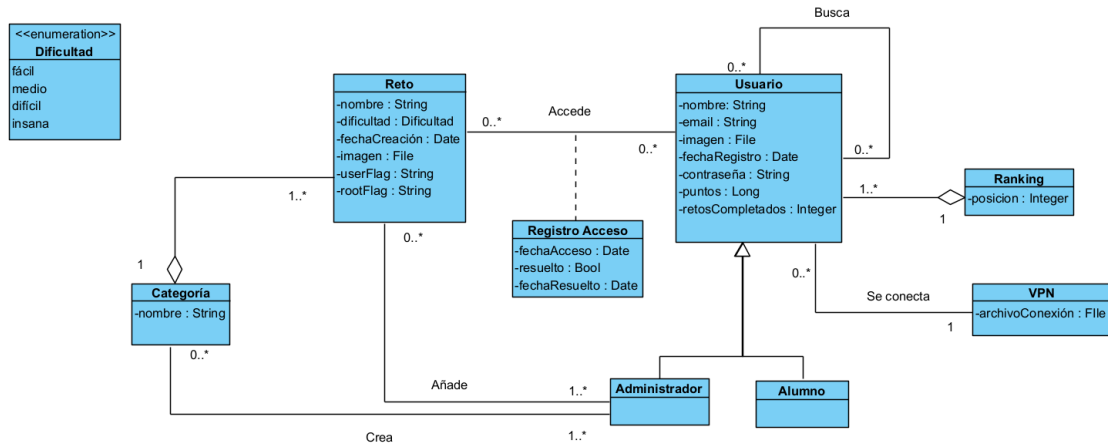
A partir de los requisitos de información que se han descrito anteriormente, se van a recoger el análisis de cada uno de ellos, para así poder acercarnos más a las necesidades reales de un usuario pudiendo tener una visión mucho más amplia de cuál es la estructura del sistema, de esta forma podremos realizar el primer contacto con la arquitectura del sistema.

Para conocer más información acerca del análisis de requisitos se puede consultar el Anexo III: Análisis de Requisitos.

### 5.5.1. Modelo de Dominio

Después del planteamiento de los requisitos de información se pretende crear un esbozo para intentar crear un modelo que se asemeje al mundo real, es por esto por lo que se crea el modelo de dominio, en donde se muestran las clases conceptuales relacionadas con los requisitos descritos.

En la *ilustración 22* se puede ver el modelo del dominio propuesto.

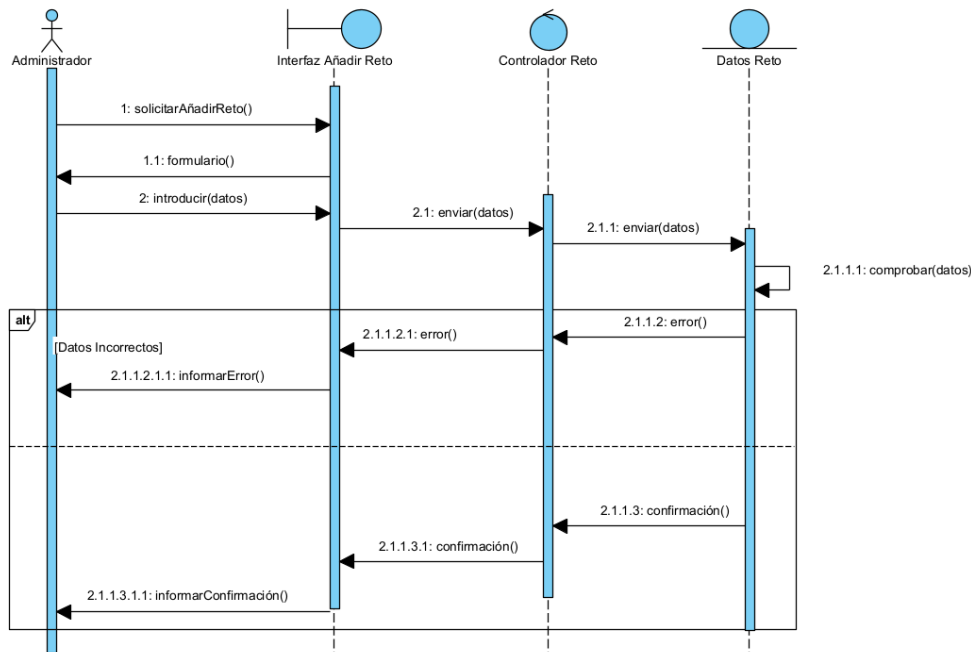


**Ilustración 22: Modelo de Dominio.**

### 5.5.2 Realización de los casos de uso

Para poder refinar los casos de uso que se han descrito antes se crean los diagramas de secuencia de los casos de uso, los cuales describen las iteraciones entre los usuarios y el sistema.

En la *ilustración 23* se puede ver un ejemplo del diagrama de secuencia del caso de uso Añadir Reto.



**Ilustración 23: Diagrama de Secuencia Añadir Reto.**

### 5.5.3. Clases de Análisis

En este punto se muestran cuáles son los diagramas de comunicación del sistema, de nuevo se han organizado según los diferentes paquetes que forman el sistema. En la *ilustración 24* se puede ver el diagrama de comunicación del paquete Gestión de Retos.

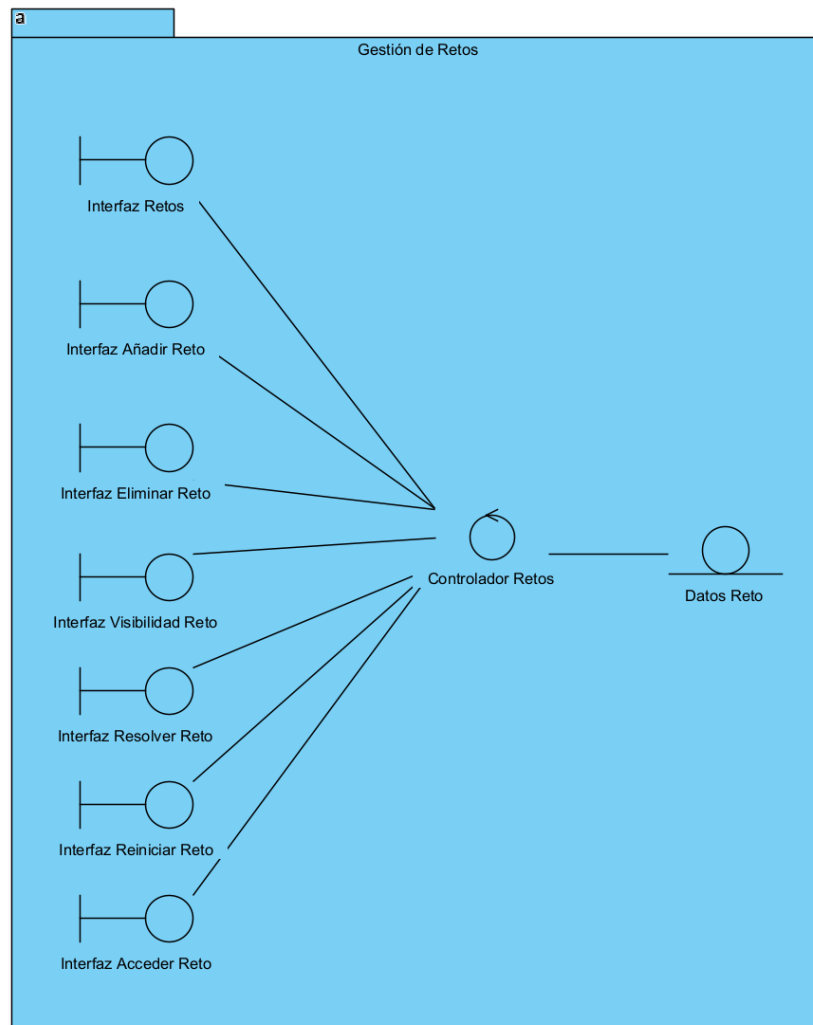


Ilustración 24: Diagrama de Comunicación Gestión de Retos.



#### 5.5.4. Arquitectura del Modelo de Análisis

A partir de todo el proceso que se ha realizado durante la fase de análisis de requisitos se crea la arquitectura final del modelo de análisis que se ha generado utilizando el patrón Modelo Vista Controlador, pudiendo diferenciar claramente cada una de las partes del proyecto según su función. Este proceso nos servirá como base para poder continuar con el desarrollo del sistema.

### 5.6. Diseño del Sistema

Una vez realizado el análisis de requisitos llega el momento de acercarnos más a implementación del sistema software, para ello se realiza el diseño del sistema en donde se van a documentar cuales van a ser las clases y los métodos que se van a utilizar en el proceso de desarrollo, junto con el desarrollo de los diagramas para poder visualizar cómo está estructurado el sistema.

Si se quiere consultar información más detallada de este proceso se puede consultar el Anexo IV: Diseño del sistema Software.

#### 5.6.1. Patrones de diseño

En este apartado se va a explicar cuál ha sido el patrón elegido para poder organizar el desarrollo, en este caso el elegido ha sido el patrón Modelo Vista Controlador, puesto que el framework Django utiliza en este patrón y esto nos ayudará a organizar las clases y métodos de una manera más optima. Seguidamente se procede a explicar en qué consiste este cada una de las partes de este patrón:

- **Modelo:** es la parte encargada de manejar los datos, utilizando mecanismos para poder acceder y actualizar los datos almacenados.
- **Vista:** especializada en gestionar las interfaces con las que se interacciona el cliente, siendo la responsable de manejar la información que se le envía y los mecanismos con los que interacciona.

- **Controlador:** esta capa es la responsable de comunicarse entre el Modelo y la Vista, siendo capaz de enlazar las diferentes peticiones que hace el cliente y de las respuestas que le da el servidor.

En la *ilustración 25* se puede ver un ejemplo del diagrama Modelo Vista Controlador.

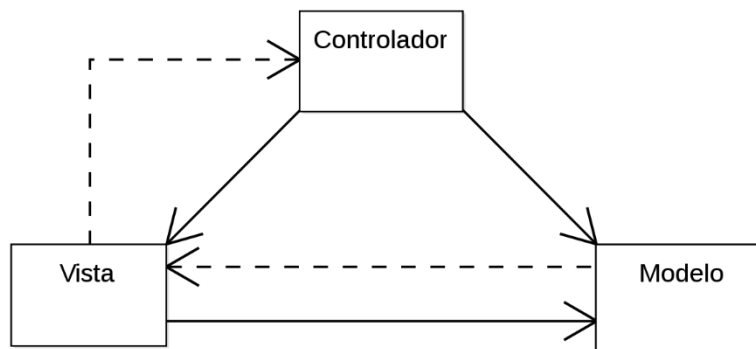


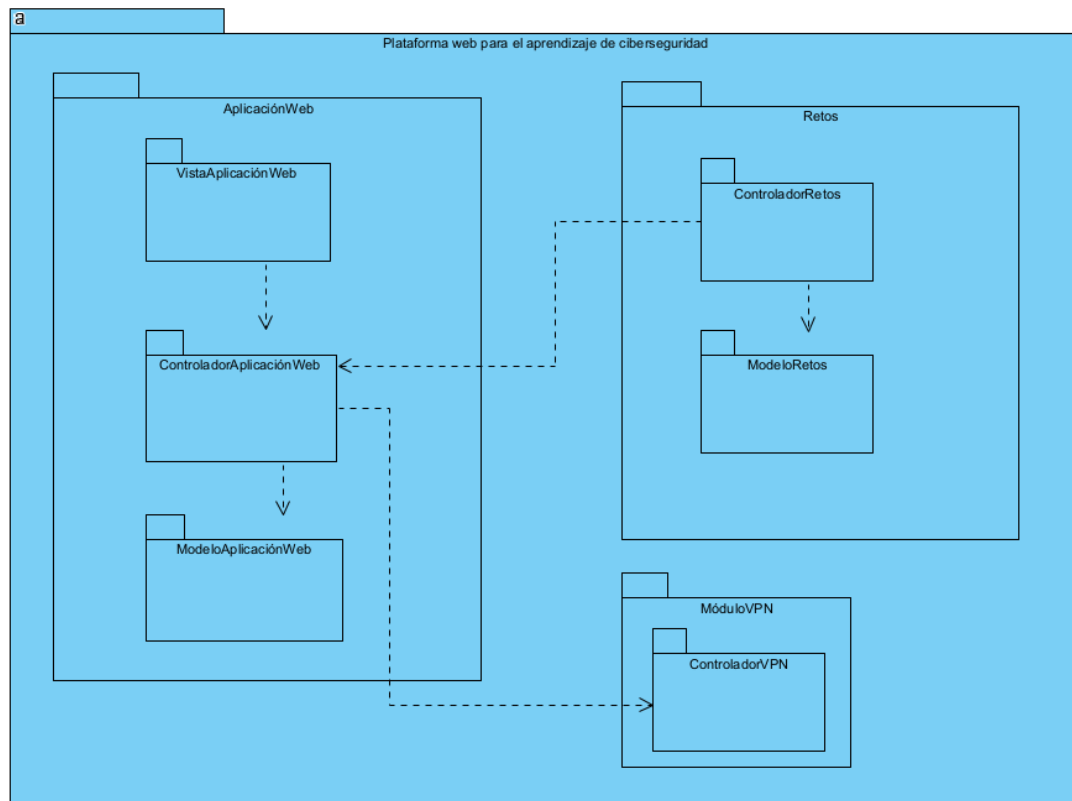
Ilustración 25: Ejemplo diagrama Modelo Vista Controlador.

### 5.6.2. Subsistemas de diseño

Para la organización del código se ha querido utilizar el siguiente subsistema de diseño, el cual está formado por:

- **Aplicación Web:** este subsistema se refiere a todo lo relacionado con la página web del sistema.
- **Retos:** Este subsistema hace referencia a los retos que van a resolver los usuarios, en donde se guarda información de los mismos y se controla el acceso a estos a través de las máquinas virtuales.
- **VPN:** Este subsistema está diseñado para manejar la información necesaria para que un usuario pueda conectarse a la red interna en donde se encuentran las máquinas virtuales.

En la *ilustración 26* se muestra el modelo de diseño generado, en donde se pueden ver las relaciones entre los diferentes paquetes.



**Ilustración 26: Modelo de Diseño.**

### 5.6.3. Clases de diseño

En la siguiente sección se van a exponer cuales son las características del patrón de diseño Modelo-Vista-Controlador, especificando los métodos de los paquetes descritos.

Como se puede ver en la *ilustración 27* se especifican cuáles son las vistas de la aplicación web, las cuales se han dividido en paquetes dependiendo del propósito de estas. Se puede observar que todos los paquetes menos Auth heredan del paquete NavBar que representa la base de las vistas, añadiendo cada una su propio contenido dependiendo de su función.

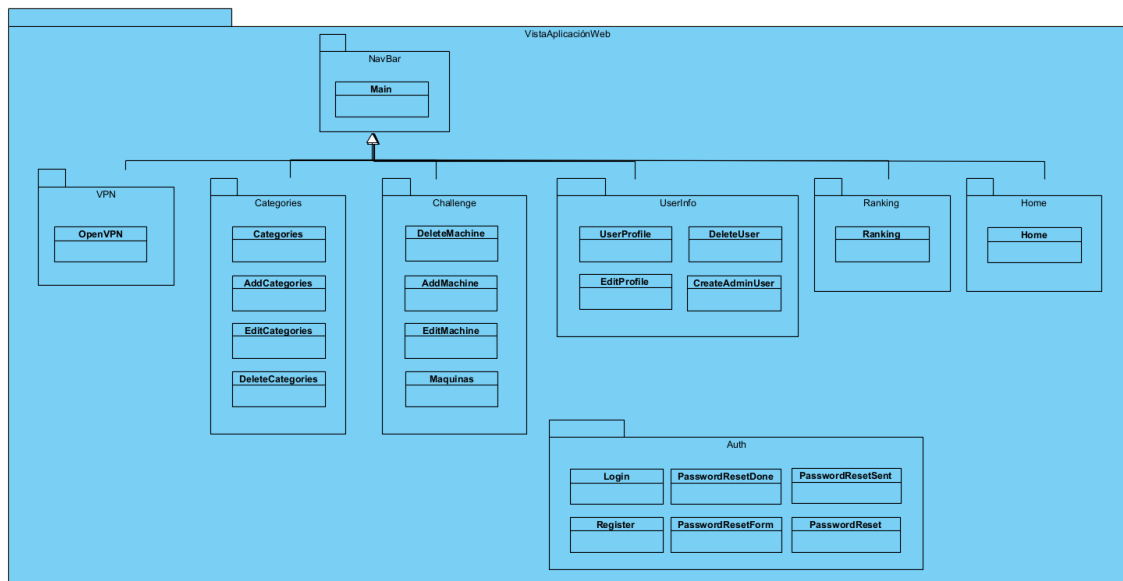


Ilustración 27: Vista Aplicación Web.

En la *ilustración 28* se muestra el Controlador Aplicación Web, en donde se indican las funciones encargadas de relacionar las vistas descritas anteriormente con los modelos.

Si se quiere consultar más información de las clases de diseño se puede consultar el Anexo IV: Diseño del sistema Software.

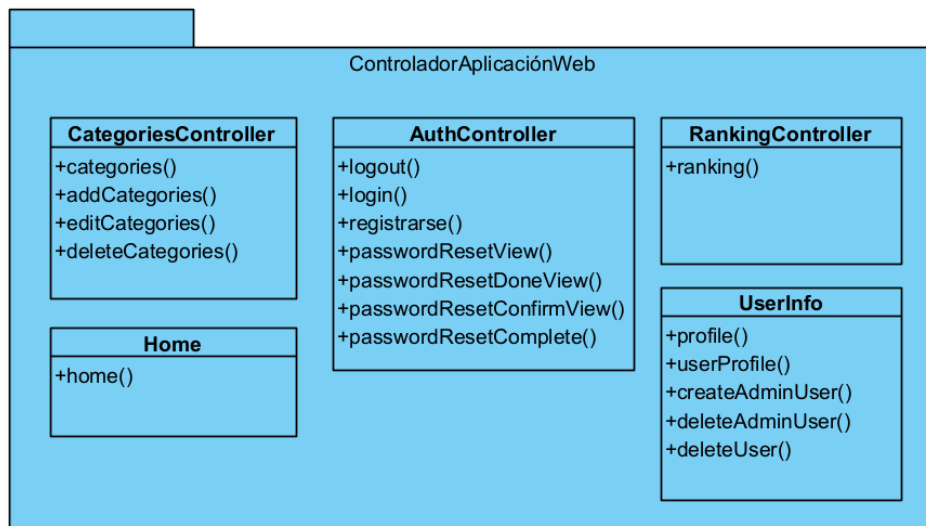
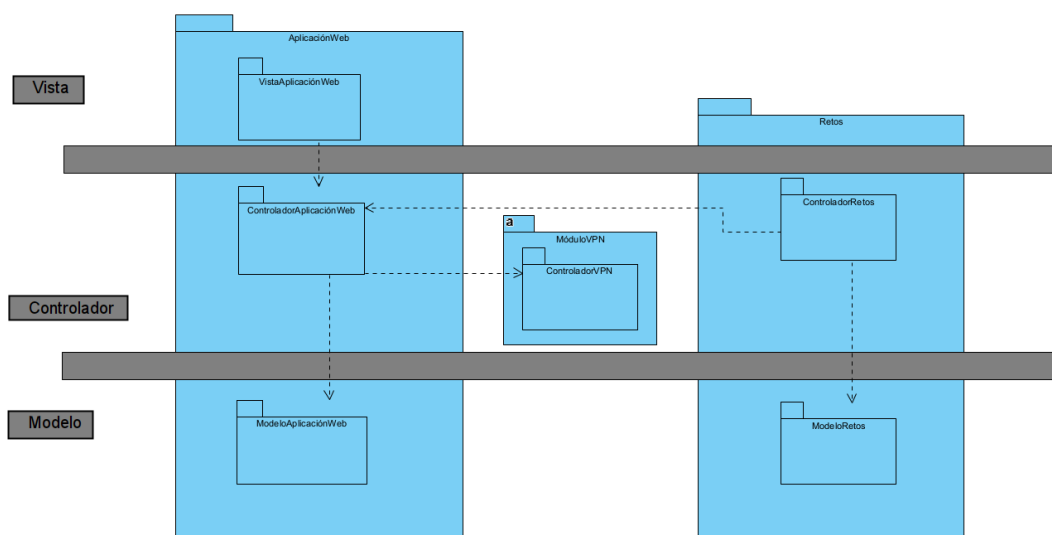


Ilustración 28: Controlador Aplicación Web.

#### 5.6.4. Vista Arquitectónica

En la *ilustración 29* se puede observar cómo se ha dividido los diferentes subpaquetes para formar la vista arquitectónica, dividiéndose en diferentes capas utilizando el patrón Modelo Vista Controlador.



**Ilustración 29: Vista Arquitectónica.**

#### 5.6.5. Realización de los Casos de Uso

En esta sección se detalla la realización de los casos de uso a partir de los diagramas de secuencia que se especificaron en el Anexo III: Análisis de Requisitos, de forma que los nombres de los métodos y funciones coincidan con el que se utilizará en el proceso de implementación. A continuación, se muestra en la *ilustración 30* la realización del caso de uso Recuperar Contraseña.

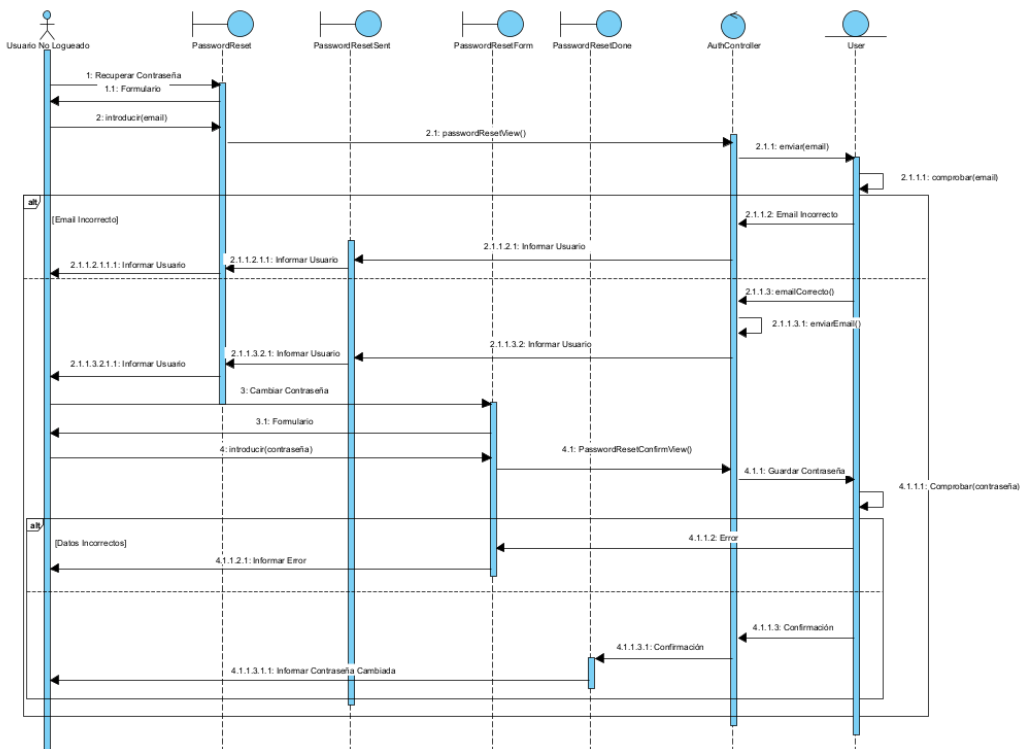


Ilustración 30: Diagrama de Secuencia Recuperar Contraseña.

### 5.6.6. Diagrama de Despliegue

Uno de los pasos más importantes para poder realizar un correcto diseño de requisitos es la creación de un diagrama de despliegue, ya que este nos ayuda a comprender cuales son los componentes de nuestro sistema y sobre todo cómo se comunican unos con otros. Este modelo de despliegue está formado por los siguientes nodos:

- **Cliente:** Representa al cliente que puede acceder a la plataforma, pudiendo realizar las acciones asociadas dependiendo de su rol, junto con la posibilidad de conectarse a los retos y poder resolverlos. Para ello utiliza el navegador web para poderse conectar al servidor de la aplicación web, esta conexión se realiza mediante el protocolo TCP/IP. Por otro lado, el cliente también tiene un entorno de ejecución de Open VPN para así poderse conectar al servidor VPN, esta conexión se realiza a través del protocolo UDP, utilizando el puerto 1194.

- **Servidor Aplicación Web:** Este nodo representa a la plataforma web, en donde los usuarios pueden realizar todas las acciones que se han especificado en los requisitos, todo esto se ejecuta en un servidor Apache y la conexión con el servidor de retos se realiza a través de SSH y con el servidor VPN se realiza a través de la API de Docker.
- **Servidor Retos:** El nodo del Servidor de Retos representa el servidor encargado de ejecutar los retos, los cuales son máquinas virtuales que se ejecutan en el software de virtualización Oracle Virtual Box, en donde la comunicación con cada una de las máquinas virtuales se realiza a través de la API perteneciente a Virtual Box.
- **Servidor VPN:** Este nodo es el encargado de ejecutar el servidor VPN encargado de dar a cada uno de los clientes acceso a la red interna en la que se encuentran los retos. Este servidor se está ejecutando en un contenedor Docker, esto es así para facilitar el despliegue y la comunicación con el servidor Web.
- **Máquinas virtuales:** Estos nodos son los encargados de representar cada uno de los retos que se están ejecutando en el Servidor de Retos, cada uno puede tener un sistema operativo diferente con diferentes servicios en ejecución.

En la *ilustración 31* se puede ver el diagrama del modelo de despliegue.

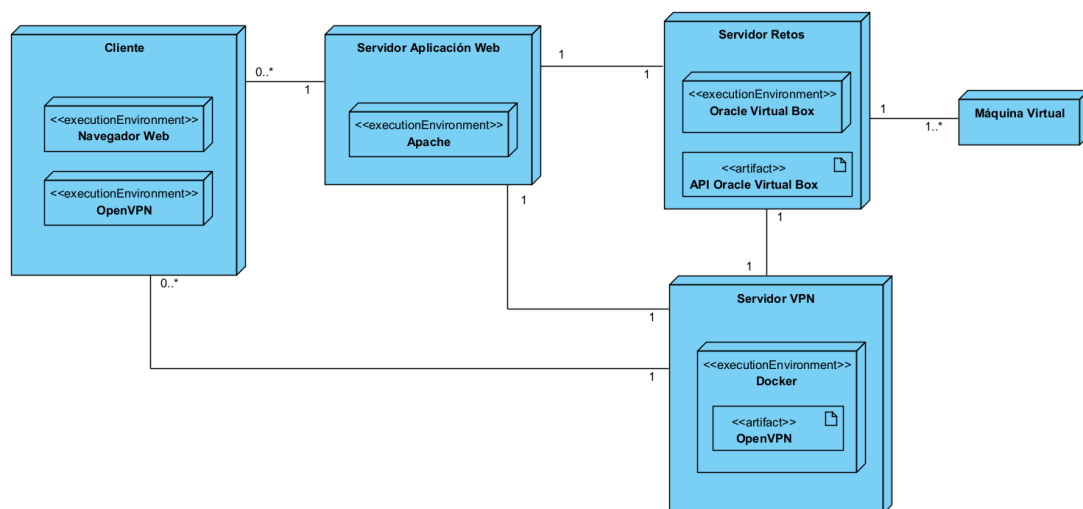


Ilustración 31: Modelo de despliegue.

## 5.7. Implementación

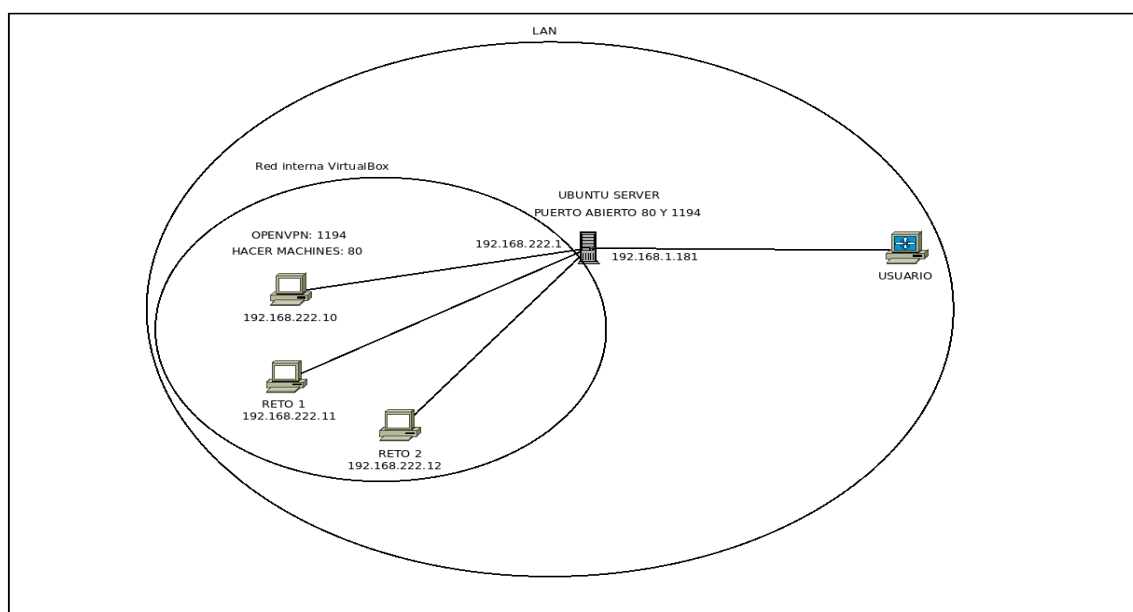
Este es el proceso que ha llevado más tiempo de desarrollo, ya que ha consistido en la codificación de todo lo planificado durante la fase de diseño. Para ello se ha seguido las técnicas y herramientas descritas anteriormente, utilizando cada una de ellas para su respectivo propósito. Se puede dividir la fase de implementación en las siguientes partes:

- **Plataforma Web:** En el proceso de implementación se decidió empezar por la plataforma web, para ello los primeros pasos han sido los propios al realizar un proyecto utilizando el framework Django. Seguidamente se comenzaron a crear los primeros modelos dentro de la base de datos, para así poder empezar a crear las diferentes acciones que se realizan en la Gestión de Usuarios, como son, iniciar sesión, registrarse, etc. Más adelante se continuó con el desarrollo de la estructura de la página web, para poder darle su conveniente estructura y apariencia, para que los usuarios puedan resolver los retos. Una vez hecho esto los siguientes pasos en el proceso de implantación fueron los de desarrollar las diferentes acciones que pueden realizar los usuarios que son administradores, como la configuración de los retos, las categorías y el manejo de los usuarios. Uno de los procesos más complicados fue la configuración de la plataforma web con el entorno virtual en donde están los retos, para ello se tuvieron que realizar multitud de pruebas y ensayos para encontrar la mejor estructura para que pudiera funcionar correctamente. Como último paso se realizó la migración del proyecto a Apache, todo ello con sus respectivas configuraciones para que toda la plataforma funcione correctamente.
- **Conexión VPN:** En esta fase la implementación se realizó utilizando un contenedor Docker en cuyo interior ya viene instalado un servidor OpenVPN, en donde sólo hace falta la configuración para el propósito de la aplicación, ya que de esta forma era mucho más sencillo el despliegue y el manejo de este a través de la API de Docker. Se tuvo que implementar las funcionalidades necesarias para que en el proceso de creación de un nuevo usuario en la plataforma web también se cree su respectivo archivo



de configuración para que una vez se lo haya descargado, este usuario pueda conectarse a la red interna en donde están ejecutando los retos.

- **Construcción del entorno virtual:** para construir el entorno virtual en el que se ejecutan los retos se ha utilizado el software Oracle Virtual Box, donde lo primero que se ha tenido que hacer es crear una red interna con la respectiva configuración del servidor DHCP de Virtual Box. Una vez hecho esto se crea una máquina virtual con un Ubuntu Server, que va a realizar el proceso de comunicación entre la red virtual y el exterior, para ello se configura este servidor con dos interfaces de red, de forma que una de ellas esté conectada con la red interna de los retos y la otra esté conectada con el exterior, de esta forma cualquiera que se quiera conectar a la red interna tiene que pasar primero por este Ubuntu Server. La estructura de la red se puede ver en la *ilustración 3*.



**Ilustración 32: Estructura de la red de la plataforma.**

Como se puede ver tanto el servidor web como el servidor OpenVPN se ejecutan dentro de una máquina virtual que se encuentra dentro de dicha red. Los retos que van a estar en la plataforma son descargados de la plataforma VulnHub como archivos .ova, una vez descargados son importarlos en Virtual Box, después son configurados para que se ejecuten dentro de la red interna.

## 5.8.Pruebas

El proceso de realización de las pruebas es fundamental para comprobar que las implementaciones que se han ido realizando durante el transcurso del desarrollo funcionan correctamente.

Es por esto por lo que se han ido realizando pruebas unitarias cada vez que se implementaba un nuevo componente, también se han realizado pruebas durante todo el desarrollo para comprobar que todos los componentes funcionan de forma correcta.

Una vez se concluyó toda la implementación se realizaron diferentes pruebas de integración, las cuales trataban de comprobar que todo el proyecto funcionaba de forma correcta y coordinada.

## 5.9. Funcionamiento del sistema de desarrollo

En este apartado se trata de explicar las principales características del funcionamiento del proyecto una vez acabada la implementación, en caso de querer consultar de manera más detallada el funcionamiento del sistema se recomienda ver el Anexo VI: Manual de Usuario.

Se van a explicar las tareas que se pueden realizar seas o no administrador, estas tareas van desde la creación de un usuario, hasta el acceso y resolución de los diferentes retos que están en el sistema.

### 5.9.1. Aplicación Web

La funcionalidad del sistema se centra en la utilización de la plataforma web, es por ello por lo que a continuación se van a explicar cada una de sus partes, divididas según su funcionalidad.

#### 5.9.1.1. Iniciar Sesión

Para que los usuarios puedan acceder a la plataforma primero deben de tener un usuario registrado, en el caso de ya tener uno se puede iniciar sesión introduciendo su nombre de usuario y su contraseña.

Regístrate'." data-bbox="188 190 803 432"/>

**Ilustración 33: Iniciar sesión.**

#### 5.9.1.2. Registrarse

Si es la primera vez que acceden a la plataforma y todavía no tienen un usuario se pueden crear uno en la siguiente pantalla rellenando el formulario que se le muestra en la *ilustración 34*.

Inicia Sesión'." data-bbox="177 604 813 856"/>

**Ilustración 34: Registrarse.**

### 5.9.1.3. Pantalla Home

Una vez iniciado sesión los usuarios pueden tener acceso para poder realizar los retos, como podemos ver la primera pantalla que se les muestra a los usuarios cuando han iniciado sesión es una breve explicación del funcionamiento de la plataforma.

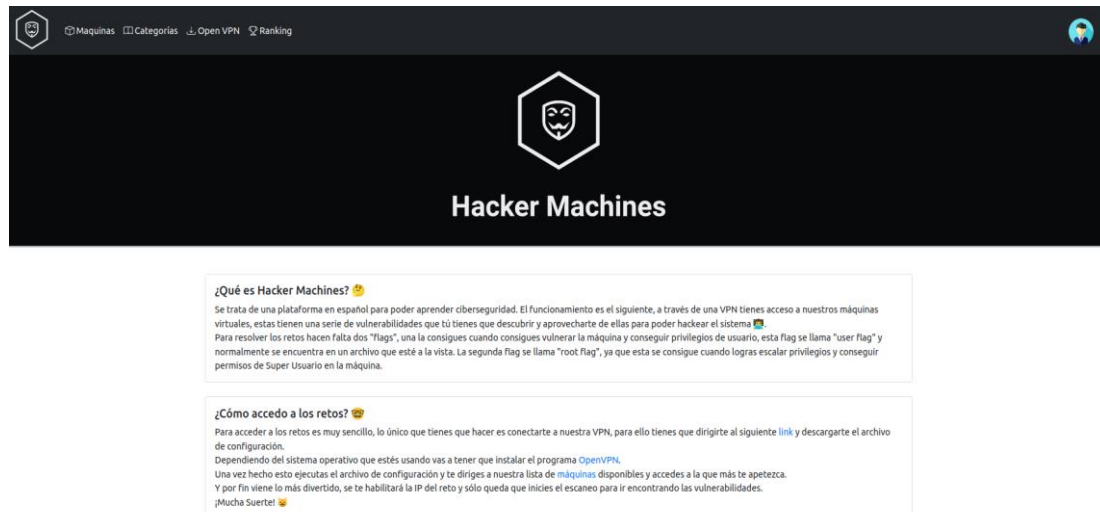


Ilustración 35: Pantalla Home.

### 5.9.1.4. OpenVPN

Los primeros pasos que debe realizar cualquier persona que quiera comenzar a resolver los retos es descargarse su archivo de configuración de la VPN, para ello dirigiéndose a la siguiente pantalla se les explica brevemente dependiendo de su sistema operativo cuales son los pasos que deben seguir para poder instalar el software OpenVPN.

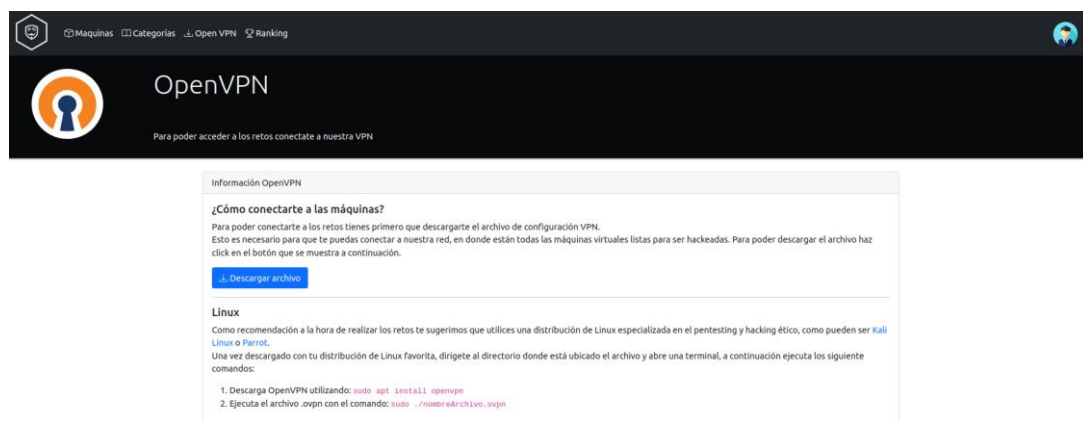
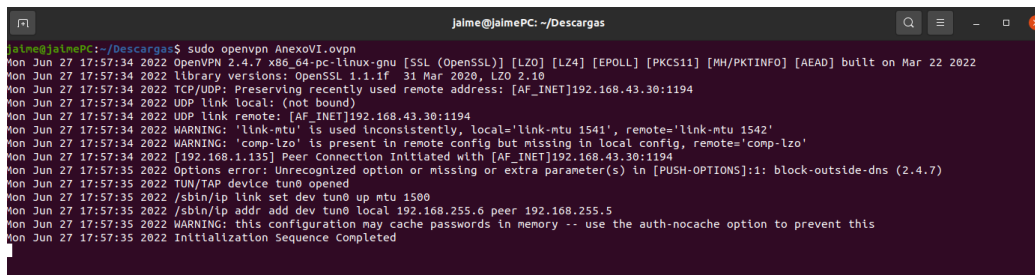


Ilustración 36: Pantalla OpenVPN.

Una vez se hayan descargado el archivo .ovpn dependiendo del sistema operativo que estén utilizando deberán ejecutarlo de diferente manera. Como por lo general se suelen realizar este tipo de retos en entornos basados en Linux, en la *ilustración 37* se muestra un ejemplo de la ejecución de este archivo y la conexión a la red.



```
jaime@jaimePC: ~/Descargas
jaime@jaimePC:~/Descargas$ sudo openvpn AnexoVI.ovpn
Mon Jun 27 17:57:34 2022 OpenVPN 2.4.7 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on Mar 22 2022
Mon Jun 27 17:57:34 2022 library versions: OpenSSL 1.1.1f  31 Mar 2020, LZO 2.10
Mon Jun 27 17:57:34 2022 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.43.30:1194
Mon Jun 27 17:57:34 2022 UDP link local: (not bound)
Mon Jun 27 17:57:34 2022 UDP link remote: [AF_INET]192.168.43.30:1194
Mon Jun 27 17:57:34 2022 WARNING: 'link-mtu' is used inconsistently, local='link-mtu 1541', remote='link-mtu 1542'
Mon Jun 27 17:57:34 2022 WARNING: 'comp-lzo' is present in remote config but missing in local config, remote='comp-lzo'
Mon Jun 27 17:57:34 2022 [192.168.1.135] Peer Connection Initiated with [AF_INET]192.168.43.30:1194
Mon Jun 27 17:57:35 2022 Options error: Unrecognized option or missing or extra parameter(s) in [PUSH-OPTIONS]:1: block-outside-dns (2.4.7)
Mon Jun 27 17:57:35 2022 TUN/TAP device tun0 opened
Mon Jun 27 17:57:35 2022 /sbin/ip link set dev tun0 up mtu 1500
Mon Jun 27 17:57:35 2022 /sbin/ip addr add dev tun0 local 192.168.255.6 peer 192.168.255.5
Mon Jun 27 17:57:35 2022 WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to prevent this
Mon Jun 27 17:57:35 2022 Initialization Sequence Completed
```

**Ilustración 37: Conexión a la VPN.**

#### 5.9.1.5. Ver Perfil

Haciendo click en la imagen de la parte superior derecha se despliega un menú con una serie de opciones, una para acceder al perfil, otra para editarlo y por último la opción de cerrar sesión. Haciendo click la primera de estas opciones se redirecciona a la siguiente pantalla, la cual nos muestra la siguiente información que es visible para los demás usuarios:

- **Imagen de perfil**
- **Número de máquinas que ha accedido**
- **Número de máquinas que ha completado**
- **Puntos totales conseguidos**
- **Fecha de registro**

Debajo de esta información se encuentra la lista de los retos que el usuario ha completado, pudiendo acceder a los mismos haciendo click sobre su foto.

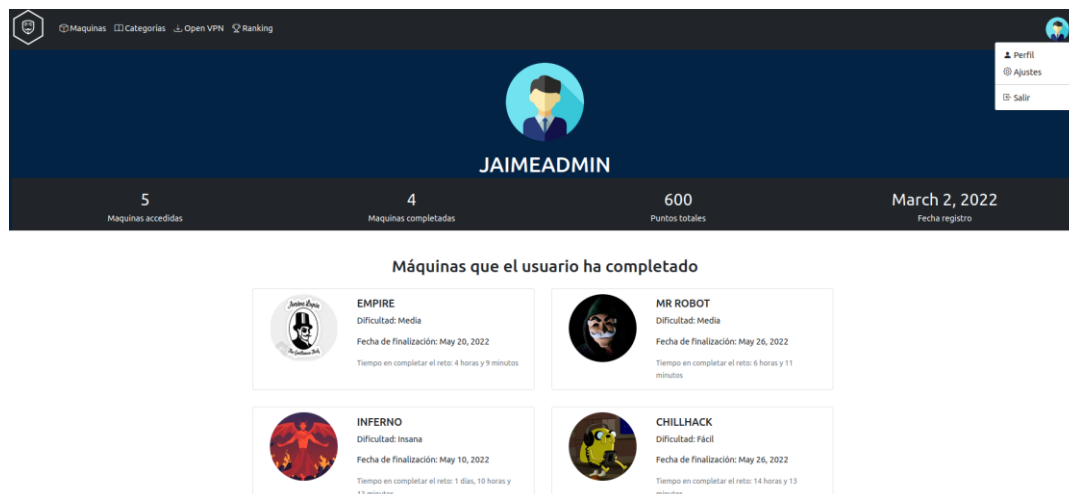


Ilustración 38: Ver Perfil.

#### 5.9.1.6. Lista de Máquinas

La *ilustración 39* representa una de las pantallas más importantes, ya que esta muestra una lista de todos los retos que están disponibles en la pantalla, indicando su nombre, la fecha de su creación, la dificultad y las categorías a las que pertenece.

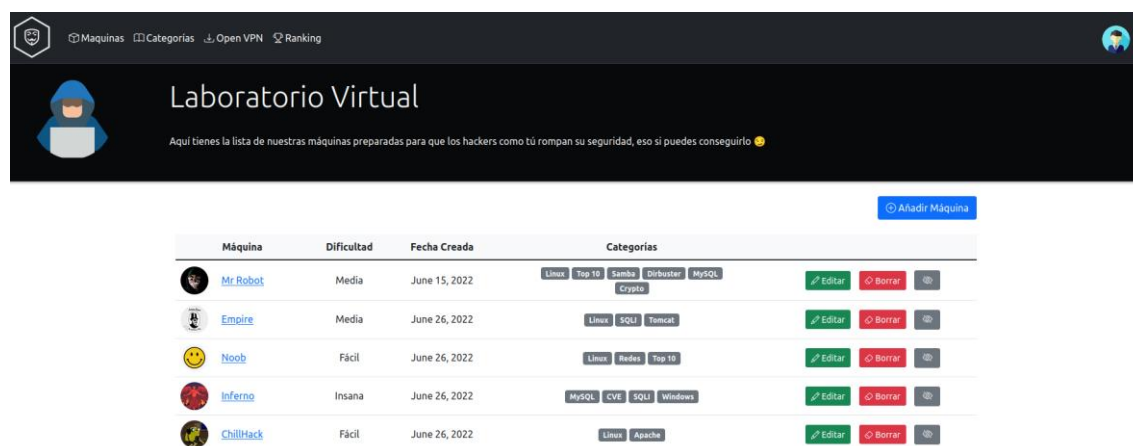


Ilustración 39: Lista máquinas siendo administrador.

Como el usuario es administrador también se pueden ver las acciones que puede realizar este tipo de usuario, pero en caso de no ser administrador la lista de las máquinas se le mostrará como se ve en la *ilustración 40*:

Máquina	Dificultad	Fecha Creada	Categorías
<a href="#">Mr Robot</a>	Media	June 15, 2022	Linux Top 10 Samba C#bester MySQL Crypto
<a href="#">Empire</a>	Media	June 26, 2022	Linux SQL Tomcat
<a href="#">Noob</a>	Fácil	June 26, 2022	Linux Redes Top 10
<a href="#">Inferno</a>	Insana	June 26, 2022	MySQL CVE SQL Windows
<a href="#">ChillHack</a>	Fácil	June 26, 2022	Linux Apache

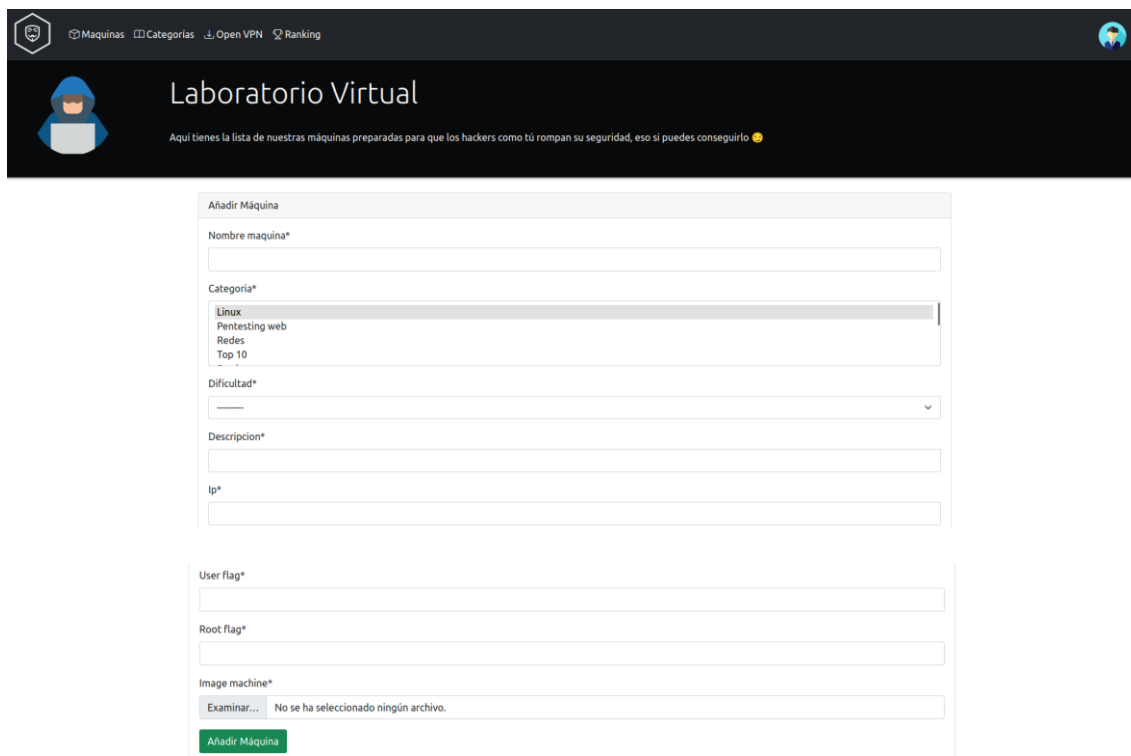
**Ilustración 40:** Lista máquinas sin ser administrador.

#### 5.9.1.7. Añadir Máquinas

En las siguientes figuras se muestra el proceso de añadir un nuevo reto a la plataforma, este consiste en un formulario en el que se deben introducir los siguientes datos:

- **Nombre de la Máquina:** Este será el título que definirá el reto y el que se mostrará en el URL de la página web.
- **Categorías:** En este apartado se pueden seleccionar las categorías que definen al reto, pudiendo así dar una idea al usuario de en qué consiste la máquina.
- **Dificultad:** Se pueden elegir cuatro opciones de dificultad, Fácil, Media, Difícil o Insana.
- **Descripción:** Esta opción permite escribir una pequeña descripción del reto.
- **IP:** Este dato es la dirección IP de la máquina virtual dentro de la red interna en la que está montada.
- **User Flag:** Esta es la primera respuesta del reto, normalmente se suele encontrar consiguiendo permisos de usuario en la máquina virtual.

- **Root Flag:** Esta es la segunda respuesta del reto, por lo general se suele encontrar una vez que has conseguido privilegios de super usuario dentro de la máquina virtual.
- **Imagen de la máquina:** Imagen que aparecerá en la portada del reto.



The screenshot shows the 'Laboratorio Virtual' web application interface. At the top, there is a navigation bar with links for 'Máquinas', 'Categorías', 'Open VPN', and 'Ranking'. Below this, a header section features a user profile icon and the text 'Laboratorio Virtual' along with a motivational message: 'Aquí tienes la lista de nuestras máquinas preparadas para que los hackers como tú rompan su seguridad, eso sí puedes conseguirlo'. The main content area displays a form titled 'Añadir Máquina'. The form includes several input fields: 'Nombre máquina\*', 'Categoría\*' (with a dropdown menu showing 'Linux', 'Pentesting web', 'Redes', and 'Top 10'), 'Dificultad\*' (with a dropdown menu), 'Descripción\*', 'Ip\*', 'User flag\*', 'Root flag\*', and 'Image machine\*'. The 'Image machine\*' field has a file selection button labeled 'Examinar...' and a message 'No se ha seleccionado ningún archivo.' At the bottom of the form is a green button labeled 'Añadir Máquina'.

**Ilustración 41: Añadir máquina.**

#### 5.9.1.8. Ver detalles del Reto

Después de que haber accedido al reto el usuario puede ver la siguiente información:

- **Dirección IP:** se muestra cual es la dirección IP de la máquina virtual.
- **Cómo acceder al reto:** un breve recordatorio de que debe descargarse el archivo de configuración OpenVPN para poder conectarse al reto.
- **Descripción del reto.**
- **Flags:** en este apartado se pueden introducir la User Flag y la Root Flag, pulsando el botón de comprobar el usuario puede ver si esa respuesta es la adecuada.



- **Reiniciar el Reto:** en caso de que la máquina virtual no funcione adecuadamente el usuario tiene la posibilidad de reiniciar el reto, a no ser que haya sido reiniciado hace menos de 5 minutos.

The screenshot shows the MR Robot challenge page. At the top, there's a navigation bar with links: Maquinas, Categorías, Open VPN, and Ranking. Below this is a header with a Mr. Robot character icon and the text "MR ROBOT".

Statistics are displayed in three columns:

- 11 Usuarios han accedido
- 4 User Flags totales
- 2 Root Flags totales

The main content area is divided into sections:

- Dirección IP del reto:** DIRECCIÓN IP 192.168.222.26
- Información de la máquina:**
  - ¿Cómo acceder a la máquina?** Para poder acceder a la máquina descargate el archivo de [OpenVPN](#) para poder acceder al entorno virtual.
  - Descripción:** Basado en la serie Mr. Robot esta máquina virtual tiene dos claves ocultas en diferentes ubicaciones, cada clave es progresivamente más difícil de encontrar. La máquina virtual no es demasiado difícil, no hay explotación avanzada ni ingeniería inversa. El nivel se considera principiante-intermedio.
  - Flags:**
    - La encontrarás al obtener privilegios de usuario en la máquina.
    - La encontrarás al obtener privilegios de Super Usuario en la máquina.
- ¿Tienes problemas con la máquina?**
  - Reiniciar el reto:** Si la máquina no funciona prueba a reiniciarla pulsando el siguiente botón, después espera unos minutos a que se termine de iniciar.

Ilustración 42: Ver detalles del reto.

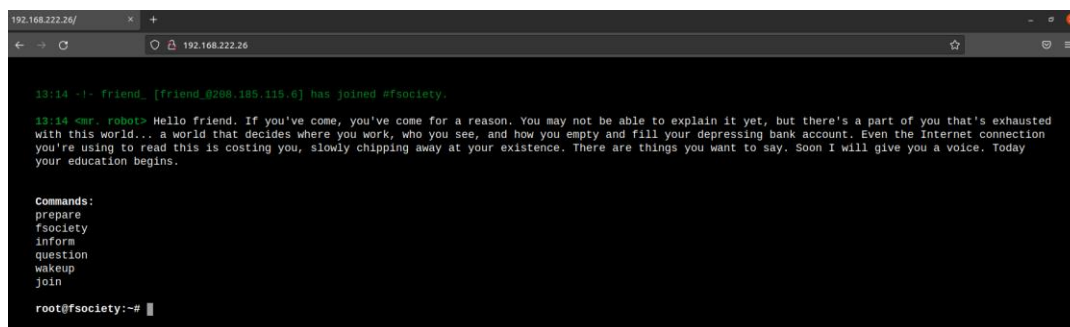
#### 5.9.1.9. Resolver Reto

Una vez el usuario ha accedido al reto, ya tiene a su disposición la dirección IP de la máquina virtual, por lo que ya puede empezar a resolver el reto. Por lo general para resolver un CTF se suele comenzar por el escaneo de puertos, para conocer qué servicios se están ejecutando en el servidor. Como se muestra la *ilustración 43* el reto MR Robot tiene tres servicios activos, ssh, http y https.

```
jai@jaiPC:~$ nmap 192.168.222.26
Starting Nmap 7.80 ( https://nmap.org ) at 2022-06-27 13:08 CEST
Nmap scan report for 192.168.222.26
Host is up (0.0019s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE
22/tcp    closed ssh
80/tcp    open  http
443/tcp   open  https
```

Ilustración 43: Escaneo de la dirección IP.

A continuación, el usuario debe realizar el análisis de cada uno de estos servicios con diferentes herramientas para poder encontrar las diferentes vulnerabilidades. En este caso este reto es muy original puesto que el servicio http que tiene ejecutándose está diseñado para parecer una terminal de Linux como se muestra en la *ilustración 44*.



```
192.168.222.26/
192.168.222.26

13:14 -!- friend_ [friend_0208.105.115.0] has joined #fsociety.

13:14 <mr_robot> Hello friend. If you've come, you've come for a reason. You may not be able to explain it yet, but there's a part of you that's exhausted with this world... a world that decides where you work, who you see, and how you empty and fill your depressing bank account. Even the Internet connection you're using to read this is costing you, slowly chipping away at your existence. There are things you want to say. Soon I will give you a voice. Today your education begins.

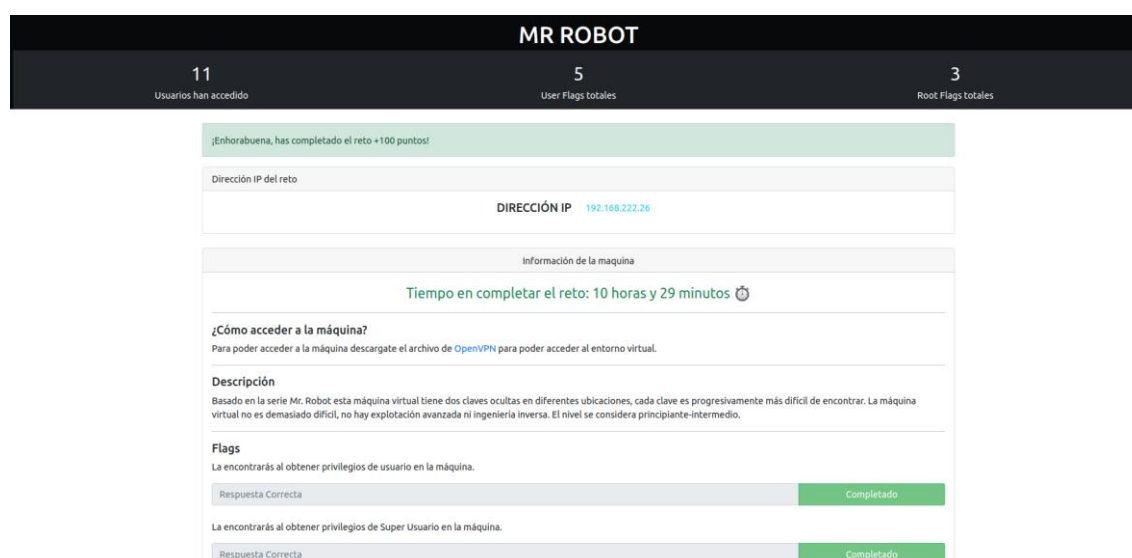
Commands:
prepare
fsociety
inform
question
wakeup
join

root@fsociety:~#
```

Ilustración 44: Ejemplo servicio http del reto.

Para resolver este reto el usuario tiene que aprovecharse de diferentes vulnerabilidades que presenta el servidor, cuando conseguida encontrarlas y aprovecharse de ellas encontrará unas flags, una vez las haya conseguido las tendrá que introducir en la parte que se indica en el reto, en caso de ser correcta se le asignarán los puntos correspondientes.

A continuación, se muestra en la *ilustración 45* la vista del reto una vez ha introducido las dos flags.



MR ROBOT		
11	5	3
Usuarios han accedido	User Flags totales	Root Flags totales

¡Enhorabuena, has completado el reto +100 puntos!

Dirección IP del reto

DIRECCIÓN IP 192.168.222.26

Información de la máquina

Tiempo en completar el reto: 10 horas y 29 minutos

¿Cómo acceder a la máquina?

Para poder acceder a la máquina descargate el archivo de [OpenVPN](#) para poder acceder al entorno virtual.

Descripción

Basado en la serie Mr. Robot esta máquina virtual tiene dos claves ocultas en diferentes ubicaciones, cada clave es progresivamente más difícil de encontrar. La máquina virtual no es demasiado difícil, no hay explotación avanzada ni ingeniería inversa. El nivel se considera principiante-intermedio.

Flags

La encontrarás al obtener privilegios de usuario en la máquina.

Respuesta Correcta Completado

La encontrarás al obtener privilegios de Super Usuario en la máquina.

Respuesta Correcta Completado

Ilustración 45: Reto completado.

#### 5.9.1.10. Ver Categorías

Para poder manejar las categorías el usuario debe de tener permisos de administrador, si es así en la siguiente pantalla se muestra cómo se pueden visualizar las categorías, pudiendo realizar las siguientes acciones:

- **Editar Categorías**
- **Borrar Categoría**
- **Añadir Categoría**

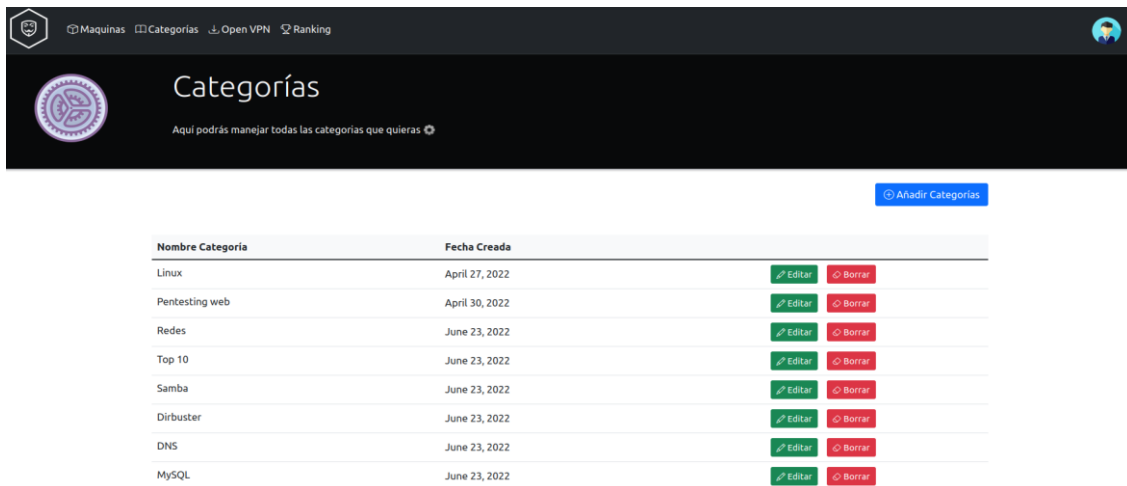


Ilustración 46: Ver categorías.

#### 5.9.1.11. Ver Ranking

A partir de los puntos conseguidos por cada uno de los usuarios se genera un ranking, pudiendo navegar explorando en que posición se encuentra cada uno y teniendo la posibilidad de buscar por el nombre de usuario.

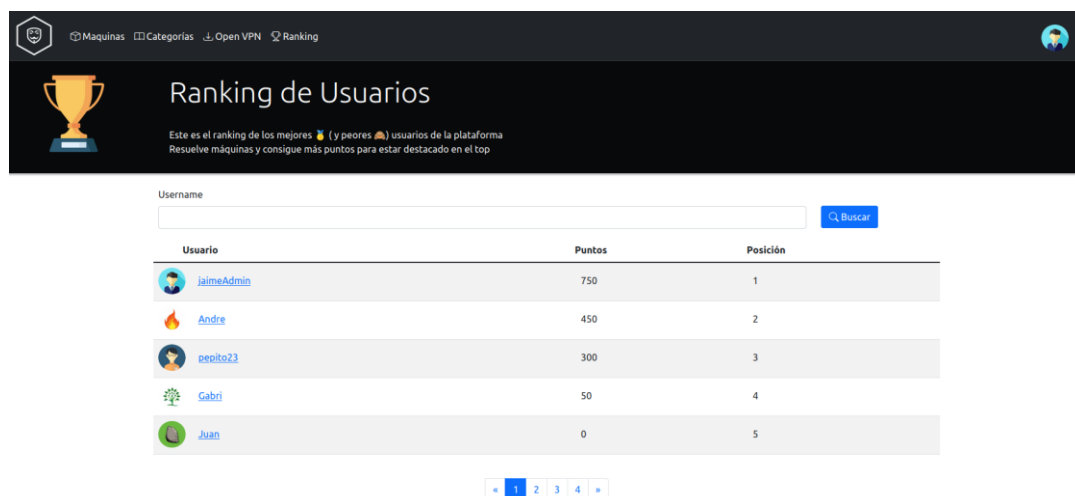
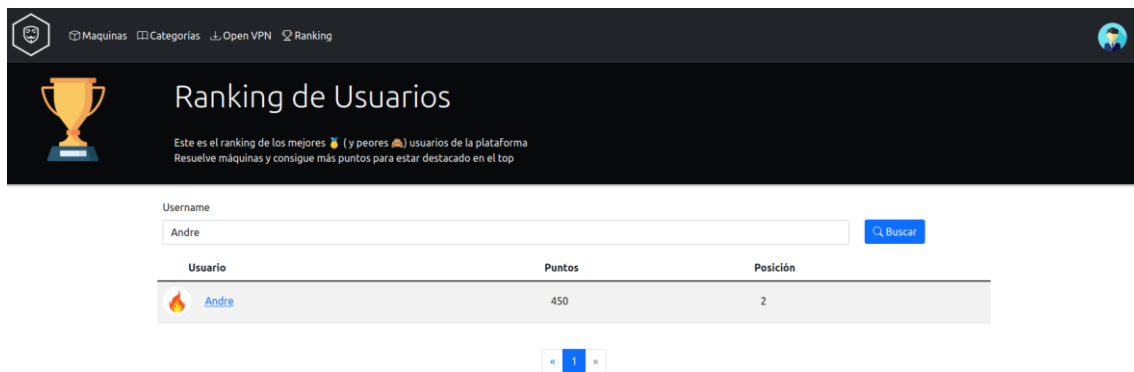


Ilustración 47: Ranking de usuarios.



**Ilustración 48: Buscar usuario.**

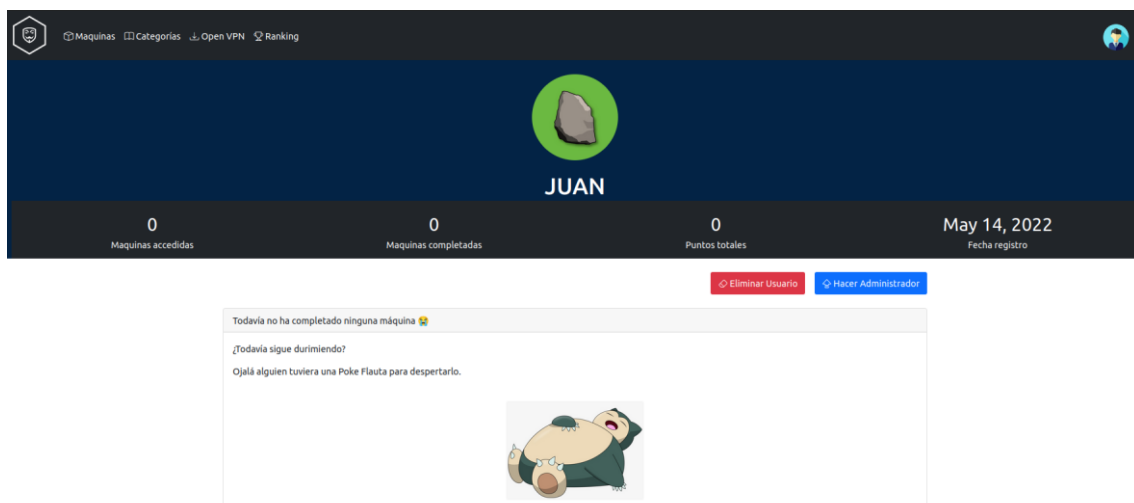
#### 5.9.1.12. Ver perfil de un usuario

Desde el ranking se puede ver el perfil de cualquier usuario que esté registrado en el sistema, en este perfil se puede ver:

- El número de máquinas que ha accedido
- El número de máquinas completadas
- La cantidad de puntos que posee
- La fecha en la que se registró
- En caso de haber resuelto alguna máquina se mostrará una lista con las que ha resuelto.

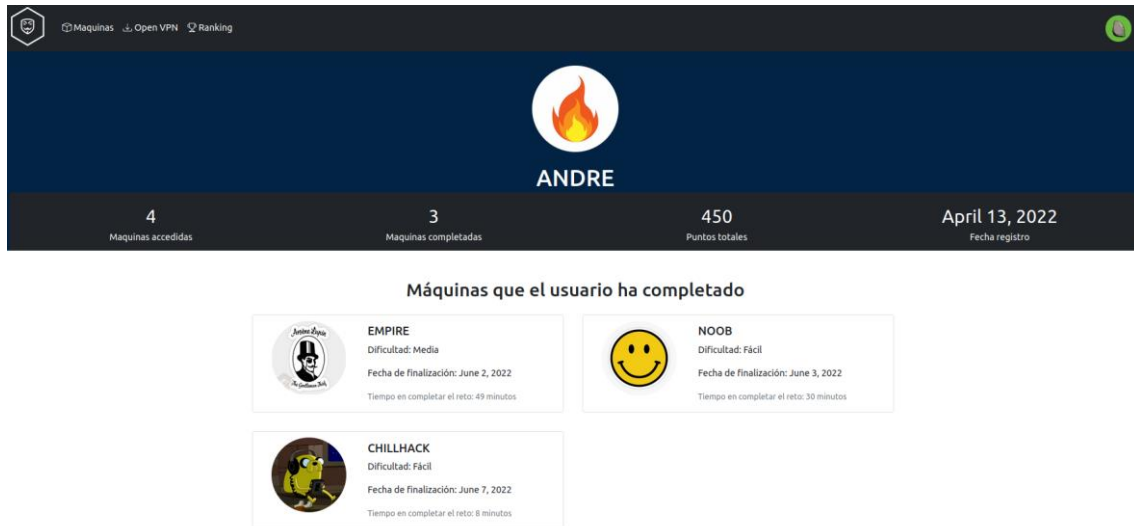
Si el usuario es administrador también se mostrarán los siguientes botones:

- Borrar Usuario
- Asignar / Quitar rol de administrador



**Ilustración 49: Ver perfil de un usuario siendo administrador.**

En caso de no ser administrador el perfil del usuario se mostrará como se ve en la *ilustración 50*.



**Ilustración 50: Ver perfil de un usuario sin ser administrador.**

## 6. Conclusiones

---

Después de desarrollar el proyecto y conseguir tener un sistema totalmente funcional se ha podido analizar desde mi punto de vista y llegar a unas conclusiones muy útiles a hora de empezar a desarrollar cualquier otro proyecto. Entre las conclusiones obtenidas podemos destacar:

- Tras mucho trabajo e investigación se ha conseguido implementar todos los objetivos que se plantearon al comienzo del trabajo:
  - **Sistema de puntuación:** Se ha podido crear un sistema que permita a los usuarios poder ganar puntos a medida que resuelven retos, pudiendo ver cuál es el total que han conseguido.
  - **Gestión de usuarios:** En cuestión a cómo se puede gestionar los perfiles de los usuarios se han cumplido con todas las características descritas.
  - **Gestión de retos:** Los usuarios pueden acceder y resolver los retos que están disponibles en la plataforma, pudiendo los administradores manejarlos correctamente.
  - **Gestión de respuestas:** Se comprueba perfectamente las soluciones de los retos propuestas por los usuarios.
  - **Gestión de categorías:** Los usuarios administradores pueden gestionar a la perfección todo lo relacionado con las categorías.
  - **Gestión de VPN:** Se ha conseguido crear la infraestructura necesaria para que los usuarios se puedan conectar a la red interna y poder resolver los retos de manera correcta.
  - **Organización del ranking:** Se ha conseguido generar un ranking en el cual se pueden visualizar los usuarios ordenados según su cantidad de puntos.
  - **Gestión de roles:** Según sean administradores o usuarios normales se ha conseguido separar las acciones que pueden realizar cada uno de ellos de manera adecuada.
  - **Búsqueda de usuarios:** Se ha permitido poder buscar a otros usuarios que se encuentren en el sistema.

- Se ha logrado realizar una aplicación con un diseño amigable y fácil de usar, esto ha sido gracias a los conocimientos que se han adquirido durante la carrera en relación con la interacción entre persona y ordenador, tratando siempre de pensar en cómo el usuario podía interactuar con la plataforma tratando de crearla de la forma más intuitiva y simple posible.
- A pesar de que en un comienzo no se tenían conocimientos en ninguna de las tecnologías que se iban a utilizar, se ha conseguido aprender a manejarlas de una manera óptima, tratando siempre de resolver cualquier problema que surgiera, buscando información en la documentación o en personas que podían haber tenido problemas similares, haciendo que haya aprendido a realizar el desarrollo de una aplicación de una manera más independiente.
- Cabe destacar la importancia de tener conocimientos en plataformas cuyo funcionamiento es similar al de mi proyecto, esto me ha ayudado enormemente a saber cómo diseñar la estructura de esta plataforma para que pueda funcionar correctamente.
- Para concluir hay que mencionar el hecho de haber aprendido lo complicado que es crear un proyecto de tales dimensiones y las dificultades que surgen al tratar de implementarlo entre una sola persona, haciendo que admire mucho más cualquier otro trabajo de programación por el hecho de pensar en las horas de trabajo que han sido necesarias para poder realizarlo.

## 7. Líneas de trabajo futuras

---

En este proyecto existen muchos aspectos que se podrían mejorar, o funciones que se podrían implementar que por el simple hecho de no tener los recursos necesarios o por falta de tiempo no se han llevado a cabo. Estas mejoras podrían ser:

- Creación de estadísticas en función de los retos que resuelven los usuarios.
- Mejorar la automatización de las máquinas para conseguir que la plataforma tenga el mayor rendimiento posible.
- Añadir la funcionalidad King of The Hill, la cual consiste en que un grupo de usuarios entren a un reto y traten de conseguir encontrar las vulnerabilidades de la máquina, consiguiendo ganar acceso lo antes posible. La persona que resuelva el reto tiene que tratar de parchear estas vulnerabilidades para que los demás usuarios no puedan aprovecharse de ellas, ganando así más puntos. La persona que consigue más puntos ganará el juego, convirtiéndose así en el King of the Hill.
- Crear la posibilidad de que los usuarios del sistema puedan proponer sus propias máquinas virtuales para que otros usuarios las resuelvan.



## 5. Bibliografía

---

- Documentación SQLite  
<https://www.sqlite.org/docs.html> [Accedido: 01-07-2022].
- Django  
[https://es.wikipedia.org/wiki/Django\\_\(framework\)](https://es.wikipedia.org/wiki/Django_(framework)) [Accedido: 01-07-2022].  
<https://docs.djangoproject.com/en/4.0/> [Accedido: 01-07-2022].
- REST  
[https://www.ics.uci.edu/~fielding/pubs/dissertation/rest\\_arch\\_style.htm](https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm)  
[Accedido: 01-06-2022].
- Pressman – Ingeniería del Software: Un Enfoque Práctico.
- Wikipedia, Proceso Unificado:
  - [https://es.wikipedia.org/wiki/Proceso\\_unificado#:~:text=El%20Proceso%20Unificado%20es%20un,varias%20iteraciones%20en%20proyectos%20grandes](https://es.wikipedia.org/wiki/Proceso_unificado#:~:text=El%20Proceso%20Unificado%20es%20un,varias%20iteraciones%20en%20proyectos%20grandes) [Accedido: 01-07-2022].
- Microsoft Project:
  - <https://docs.microsoft.com/es-es/project/> [Accedido: 01-07-2022].
- Sphinx:  
<https://www.sphinx-doc.org/en/master/> [Accedido: 01-07-2022].
- Documentación Python  
<https://docs.python.org/3/> [Accedido: 01-07-2022].
- Documentación Bootstrap  
<https://getbootstrap.com/docs/5.2/getting-started/introduction/>  
[Accedido: 01-07-2022].
- Docker  
<https://docs.docker.com/get-started/overview/> [Accedido: 01-07-2022].
- Oracle Virtual Box  
<https://docs.oracle.com/en/virtualization/virtualbox/index.html>  
[Accedido: 01-07-2022].
- Documentación OpenVPN  
<https://openvpn.net/community-resources/> [Accedido: 01-07-2022].
- Visual Studio Code  
<https://code.visualstudio.com/> [Accedido: 01-07-2022].

- VulnHub  
<https://www.vulnhub.com/> [Accedido: 01-07-2022].
- Documentación GIT  
<https://git-scm.com/doc> [Accedido: 01-07-2022].
- GitHub  
<https://github.com/> [Accedido: 01-07-2022].
- Información máquinas virtuales  
<https://www.xataka.com/especiales/maquinas-virtuales-que-son-como-funcionan-y-como-utilizarlas> [Accedido: 01-07-2022].
- Información VPN  
<https://www.xataka.com/basics/que-es-una-conexion-vpn-para-que-sirve-y-que-ventajas-tiene> [Accedido: 01-07-2022].  
[https://es.wikipedia.org/wiki/Red\\_privada\\_virtual](https://es.wikipedia.org/wiki/Red_privada_virtual) [Accedido: 01-07-2022].
- María Navarro Cáceres, María N. Moreno García, Transparencias de la asignatura de Gestión de Proyectos – “Planificación Temporal”.
- María N. Moreno García, Transparencias de la asignatura de Gestión de Proyectos – “Estimación del Esfuerzo”.
- Amador Durán Toro, Beatriz Bernández Jiménez – “Metodología para la Elicitación de Requisitos de Sistemas Software”:
  - <http://www.lsi.us.es/docs/informes/lsi-2000-10.pdf>  
[Accedido: 01-07-2022].
- Amador Durán Toro, Miguel Toro Bonilla - “Un entorno metodológico de ingeniería de requisitos para sistemas de información”
  - <https://idus.us.es/handle/11441/15365> [Accedido: 01-07-2022].
- Pressman – “Ingeniería del Software: Un Enfoque Práctico”.
- Francisco José García Peñalvo, Alicia García Holgado, Transparencias de la Asignatura Ingeniería del Software – “Modelo de Domino”.