DISEÑO Y DESARROLLO DE UNA WEB DINÁMICA EN UNA VPS CON MONITORIZACIÓN

ADRIÁN VILLADA MURILLO

CENTRO DIGITECH FP

CURSO 24/25

GRADO SUPERIOR Administración de Sistemas Informáticos en Red

Contenido

1 Identificación de necesidades 3
1.1 Contextualización 3
1.2 Justificación 3
1.3 Otros aspectos 3
2 Diseño del proyecto 4
2.1 Contenido 4
2.2 Objetivos y recursos 4
2.3 Viabilidad económica 4
2.4 Modelo de solución 4
3 Ejecución del proyecto 5
3.1 Planificación temporal 5
3.2 Riesgos 5
3.3 Revisión de recursos 5
3.4 Documentación de ejecución 6
4 Seguimiento y control 7
4.1 Valoración del proyecto 7
4.2 Incidencias 7
4.3 Cambios 7
4.4 Pruebas y soporte 7

Proyecto Integrado Digitech por Adrián Villada Murillo

1. Identificación de necesidades

1.1 Contextualización

El presente proyecto se enmarca en el ámbito de la administración de sistemas y el desarrollo web seguro, un sector clave en el entorno tecnológico actual. Las empresas de este ámbito demandan soluciones personalizadas, alojadas en entornos propios (como servidores VPS), que garanticen la seguridad, el control de los datos y la escalabilidad.

La empresa/organización que implementaría este proyecto sería una empresa pequeña del sector tecnológico, dedicada al desarrollo de contenido especializado para videojuegos o formación online. El objetivo principal sería ofrecer una plataforma accesible, segura y funcional que permita a los usuarios registrarse, iniciar sesión, consultar información detallada sobre personajes del juego Valorant, y visualizar vídeos con estrategias y mecánicas de juego.

La necesidad detectada es la creación de una plataforma centralizada y segura donde los usuarios puedan aprender a utilizar personajes del videojuego Valorant, consultar mapas recomendados y estrategias, y acceder a vídeos explicativos, todo ello desde una interfaz web moderna alojada en un servidor privado virtual (VPS) propio y adicionalmente se le ha implementado una VPN.

1.2 Justificación

El proyecto responde directamente a la necesidad de contar con un sistema web seguro y funcional que permita a los usuarios acceder a contenidos especializados sin depender de plataformas de terceros.

Las oportunidades de negocio en este sector son diversas, ya que el interés por los videojuegos competitivos sigue creciendo, y con ello, la demanda de contenido educativo o formativo sobre mecánicas de juego, estrategias y uso de personajes.

Entre las características más relevantes del proyecto destacan:

- Securización inicial del VPS (acceso mediante claves SSH, cambio de puerto por defecto y configuración de Fail2ban).

- Configuración completa de un VPS con Apache, MySQL y PHP.

- Instalación de un servidor VPN con WireGuard como demostración práctica de su configuración y uso.

- Desarrollo de una página web funcional con login, registro y recuperación de contraseña.

- Implementación de un sistema de filtrado de personajes por rol.

- Página de detalles con descripción, habilidades y mapas recomendados por personaje.

- Visualización de vídeos de YouTube embebidos para mostrar estrategias por mapa.

- Almacenamiento y gestión de los datos mediante una base de datos MySQL estructurada.

1.3 Otros aspectos

· Obligaciones fiscales, laborales y de prevención de riesgos:
En el caso de convertir este proyecto en una actividad económica real, habría que contemplar el alta como autónomo o la constitución de una empresa, así como la contratación de personal técnico o creativo si se desea ampliar el proyecto. Al tratarse de un entorno de teletrabajo, muchas de las exigencias relativas a prevención de riesgos laborales se ven reducidas, aunque se deben seguir buenas prácticas ergonómicas y de organización del trabajo.

· Ayudas o subvenciones disponibles:
Existen ayudas y subvenciones orientadas a jóvenes emprendedores, empresas tecnológicas o proyectos de digitalización, tanto a nivel autonómico como estatal. Entre ellas destacan las del Programa Kit Digital, Red.es, y ayudas de la Junta de Andalucía u organismos locales como Promálaga.

· Guion de trabajo del proyecto:

- Securización inicial del VPS (claves SSH, cambio de puerto, Fail2ban).

- Instalación y configuración del stack LAMP (Linux, Apache, MySQL, PHP).

- Configuración del servidor VPN con WireGuard.

- Diseño e implementación de la estructura de base de datos.

- Desarrollo de la interfaz web: login, registro, recuperación de contraseña.

- Implementación de secciones dinámicas: personajes, detalles y vídeos.

- Pruebas funcionales, revisión de errores y mejoras.

- Redacción de la documentación del proyecto.

2. Diseño del proyecto

2.1 Contenido

Este proyecto tiene como finalidad la implementación de una plataforma web funcional y segura, centrada en el análisis de personajes del videojuego Valorant, así como el despliegue de la infraestructura necesaria en un servidor VPS. Se incluyen aspectos clave de la administración de sistemas (seguridad, servidores, VPN, base de datos), desarrollo backend y diseño web.

· Viabilidad técnica

La viabilidad técnica del proyecto es alta, dado que todos los elementos utilizados (Apache, MySQL, PHP, WireGuard) son de código abierto, ampliamente documentados y compatibles entre sí. El VPS utilizado cuenta con recursos suficientes para soportar la web y su base de datos sin necesidad de ampliaciones. Además, se ha trabajado con tecnologías estándar del sector, lo que facilita su mantenimiento, escalabilidad y posible reutilización.

· Fases del proyecto

1. Fase de planificación y seguridad inicial

- Configuración del acceso seguro por SSH

- Cambio de puerto predeterminado

- Instalación y configuración de Fail2ban

2. Fase de infraestructura

- Instalación de Apache, MySQL y PHP

- Configuración del entorno de ejecución y pruebas

3. Fase de implementación de VPN

- Instalación y configuración de WireGuard como demostración de conocimientos de redes seguras

4. Fase de desarrollo backend y base de datos

- Diseño de la base de datos con sus relaciones (usuarios, personajes, mapas, habilidades)

- Desarrollo de scripts PHP para la lógica de login, registro, recuperación, y consulta de datos

5. Fase de desarrollo frontend

- Diseño de la interfaz web

- Filtros de personajes por rol

- Páginas dinámicas de personajes y vídeos

6. Fase de pruebas, depuración y mejora

- Verificación de funcionamiento de cada componente

- Pruebas de seguridad básica

- Revisión y corrección de errores

7. Fase de documentación final

- Elaboración de la memoria del proyecto

- Recopilación de evidencias gráficas

2.2 Objetivos y recursos

· Objetivos del proyecto

- Implementar un entorno web funcional y seguro en un VPS propio.

- Demostrar conocimientos en administración de sistemas (servidores, seguridad, redes).

- Desarrollar una web informativa sobre Valorant con filtrado y visualización de contenido.

- Diseñar una base de datos relacional adecuada al contenido y necesidades del sitio.

- Garantizar buenas prácticas de seguridad en la implementación.

· Recursos hardware y software

· Hardware:

- Ordenador personal

- Servidor VPS (1 CPU, 2 GB RAM, 40 GB SSD)

· **Software:**

- Sistema operativo: Ubuntu Server 24.04

- Servidor web: Apache 2

- Base de datos: MySQL

- Netdata

- Lenguaje de programación: PHP

- VPN: WireGuard

- Cliente SSH: Mobaxterm

- Navegador web: Google Chrome

- Herramientas de edición de código: Visual Studio Code

· ***Recursos materiales y personales***

- Conexión a Internet estable

- Documentación técnica (manuales oficiales, foros, tutoriales)

- Una persona encargada del diseño, implementación y documentación (yo como alumno)

2.3 Viabilidad económica

Concepto	Coste aproximado
VPS (1 año, 2GB RAM, 40GB SSD)	12€
Dominio personalizado (1 año)	7€
Electricidad / Internet propio	Incluido
Software utilizado	0€ (opensource)

Total estimado: 19 €

Financiación necesaria

No se requiere financiación externa. Los gastos son asumidos por el alumno para la realización del proyecto con fines educativos.

2.4 Modelo de solución

· Descripción de la solución

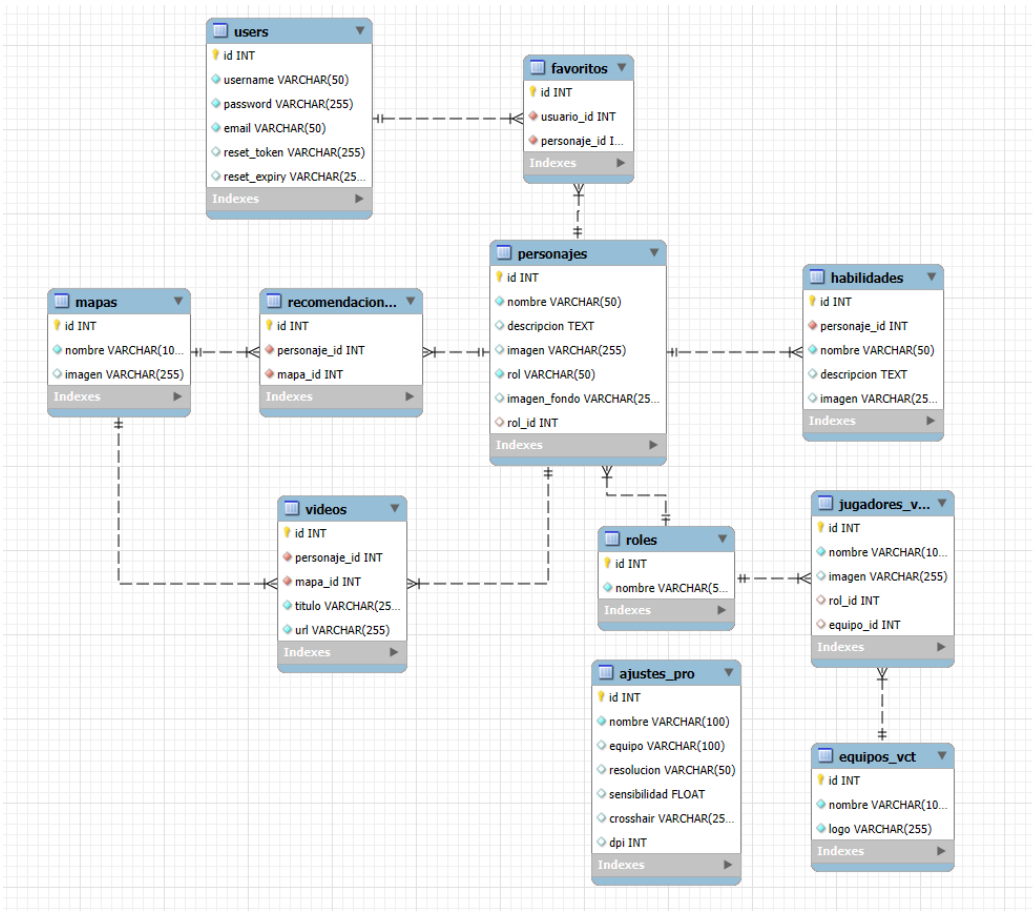
El proyecto consiste en una plataforma web orientada a jugadores de Valorant, desarrollada y alojada en un VPS configurado manualmente por mí. La web permite a los usuarios registrarse, iniciar sesión, explorar información detallada sobre personajes, ver vídeos y recomendaciones según mapas, así como recuperar su contraseña mediante correo electrónico.

Toda la información del sitio se gestiona dinámicamente a través de una base de datos MySQL, diseñada específicamente para cubrir las necesidades del proyecto. El contenido (personajes, habilidades, mapas, vídeos, jugadores profesionales, equipos VCT) se carga en tiempo real mediante consultas a esta base de datos. La seguridad del sistema ha sido una prioridad desde el inicio, incluyendo el acceso al servidor por SSH con claves, modificación del puerto por defecto, y la protección mediante Fail2ban. Adicionalmente, se ha implementado una VPN con WireGuard para demostrar la capacidad de proteger el acceso mediante redes privadas virtuales.

- Tabla de direccionamiento IP y puertos usados

Servicio	IP pública	Puerto	Protocolo	Descripción
Apache	[redacted]	[redacted]	TCP	Acceso HTTP
Apache SSL	[redacted]	[redacted]	TCP	Acceso HTTPS
SSH	[redacted]	[redacted]	TCP	Acceso remoto seguro
Wireguard	[redacted]	[redacted]	UDP	Conexión VPN
MySQL	[redacted]	[redacted]	TCP	Acceso a la base de datos solo desde el propio VPS

- Diagrama E/R de la base de datos



- Tabla de componentes software y hardware

Componente	Tipo	Descripción
VPS	Hardware	1 CPU, 2GB RAM, 40 GB SSD
Ubuntu Server 24.04	Software	Sistema operativo
Apache 2.4	Software	Servidor web
MySQL 8.0	Software	Base de datos relacional
PHP 8.3.6	Software	Backend dinámico
Wireguard	Software	VPN ligera y rápida
Visual Studio Code	Software	Entorno de desarrollo local
Cliente SSH	Software	Acceso remoto y administración
Netdata	Software	Monitorización del servidor en tiempo real

- Tabla de usuarios, permisos y funcionalidades

Rol	Funcionalidades
Administrador	Accede a toda la web, y gestiona datos si fuera necesario en el futuro
Usuario	Registro, login, ver personajes, habilidades, videos, mapas, equipos, recuperación de contraseña.

- Cronograma con fases del proyecto

Semana	Fase
1	Securización del VPS (SSH, puerto, Fail2ban)
2	Instalación de Apache, MySQL, PHP y configuración básica
3	Implementación de Wireguard VPN
4	Diseño y creación de la base de datos relacional
5	Desarrollo del backend (login, registro, recuperación)
6	Desarrollo del frontend y diseño responsivo
7	Funcionalidades dinámicas: filtros, vídeos, detalles
8	Pruebas, documentación y entrega final

- Organigrama básico del desarrollo (plan de trabajo)

Inicio del proyecto

├─ Securización VPS

├─ Instalación servicios

├─ VPN como práctica

├─ Desarrollo backend

├─ Diseño base de datos

├─ Desarrollo frontend

└─ Pruebas y documentación

- Diagramas de flujo de login, registro, y recuperación

Diagrama de flujo de registro:

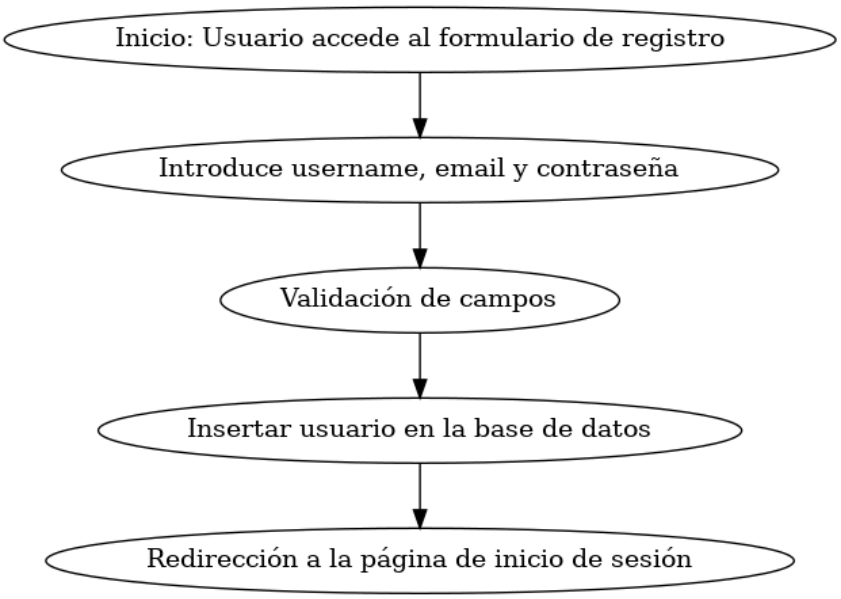


Diagrama de flujo de login:

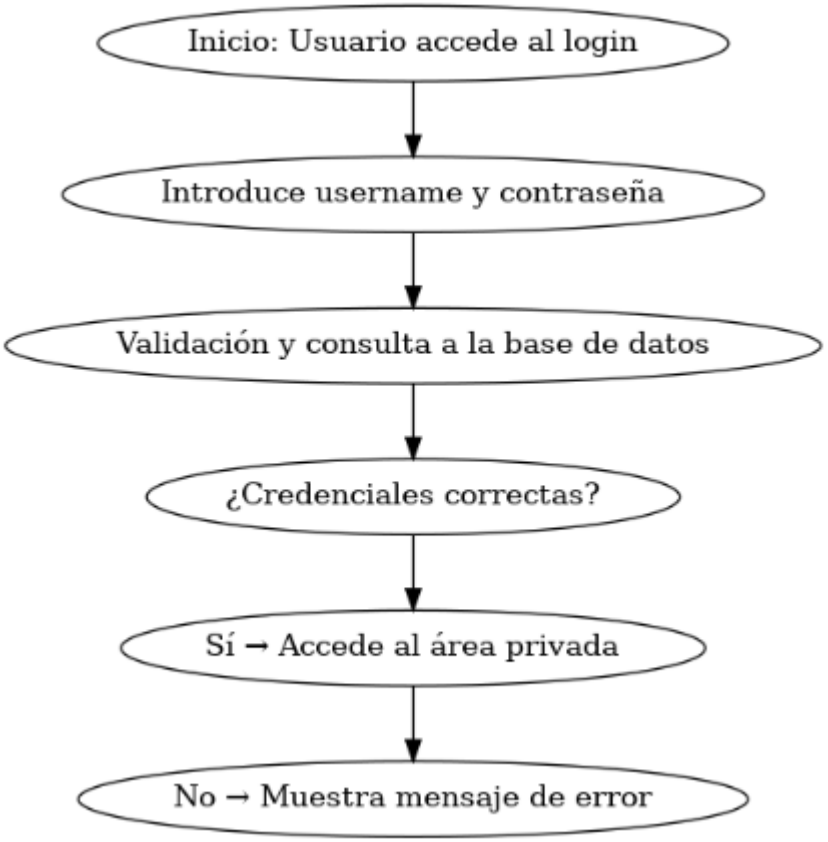
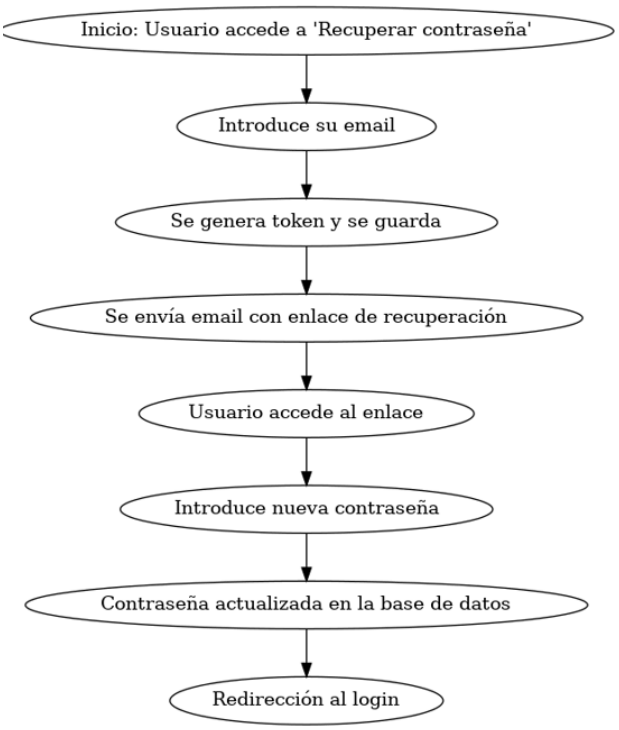


Diagrama de flujo de recuperación:



· Controles de calidad identificados

- Pruebas funcionales: comprobación de login, registro, filtrado, carga de vídeos.
- Seguridad: acceso por claves SSH, puerto personalizado, Fail2ban.
- Validación de formularios para evitar inyecciones básicas.
- Comprobación de enlaces y visualización de contenidos embebidos.
- Verificación de la estructura y relaciones en la base de datos.
- Pruebas de carga básica con varios usuarios en simultáneo.

3. Ejecución del proyecto

3.1 Planificación temporal

- **Identificar las tareas dependientes.**

El desarrollo del proyecto siguió una planificación lógica en la que algunas tareas dependían necesariamente de la finalización de otras anteriores. El esquema de dependencias fue el siguiente:

- Securización del VPS (claves SSH, cambio de puerto y Fail2ban) → Debía completarse antes de instalar cualquier servicio, para proteger la máquina desde el inicio.

- Instalación del stack LAMP (Apache, MySQL, PHP) → Requería tener el servidor asegurado.

- Configuración de WireGuard VPN → Aunque no estrictamente dependiente, se realizó tras asegurar el VPS y antes del desarrollo de la web.

- Diseño e implementación de la base de datos MySQL → Necesitaba que MySQL estuviera operativo.

- Desarrollo backend (login, registro, recuperación) → Requería tener la base de datos diseñada.

- Desarrollo frontend (estructura y diseño de la página) → Dependía de tener el backend funcional para conectar los datos.

- Implementación de funcionalidades dinámicas (filtros, vídeos, mapas) → Dependía de haber finalizado el desarrollo base del backend y frontend.

- Fase de pruebas y documentación final → Solo pudo comenzar tras terminar todo el desarrollo funcional.

- **Identificar los recursos necesarios en cada tarea.**

Tarea	Recursos necesarios
Securización VPS	Acceso SSH root, conocimientos básicos de seguridad Linux
Instalación de servicios LAMP	Conexión SSH, paquetes de software (Apache, MySQL, PHP)
Configuración VPN WireGuard	Instalación de WireGuard, conocimientos básicos de redes
Diseño de base de datos	Conocimiento de MySQL, modelado E/R
Desarrollo backend	Entorno de desarrollo local, Visual Studio Code, PHP
Desarrollo frontend	Conocimiento de HTML, CSS, PHP y nociones de usabilidad
Funcionalidades dinámicas	Conexiones a MySQL, lógica PHP avanzada
Pruebas y documentación	Acceso total al sistema, herramientas de captura de pantalla, editores de texto

- **Permisos y autorizaciones necesarias.**

Para la ejecución completa del proyecto fueron necesarios:

- Acceso root al servidor VPS para realizar configuraciones de red, instalación de servicios y seguridad.

- Permisos de administrador en la cuenta de usuario propia.

- Autorizaciones de conexión VPN para gestionar las claves y túneles.

- Control del dominio y configuración DNS para redirigir correctamente al subdominio del proyecto.

- **Identificar protocolo de actuación en cada fase.**

Cada fase se abordó siguiendo un protocolo de trabajo que asegurara la correcta ejecución:

· Planificación previa: análisis de requerimientos, definición de herramientas.

· Ejecución controlada: instalación y configuración paso a paso, con comprobaciones tras cada cambio.

· Pruebas parciales: verificación de funcionamiento después de cada implementación importante (por ejemplo, después de configurar Apache, probar la carga de una página de prueba).

· Documentación constante: capturas de pantallas y registros de cada configuración y prueba exitosa o problemática.

· Backup de seguridad: copias puntuales de archivos y base de datos a medida que se avanzaba en el proyecto.

· Corrección de errores detectados en pruebas: iterar hasta obtener el funcionamiento esperado antes de continuar a la siguiente fase.

3.2 Riesgos

- **Identificación de riesgos.**

Riesgo	Descripción
Fallo en la conexión SSH	Pérdida del acceso remoto al VPS debido a errores en la configuración de red, bloqueo de puertos o eliminación de claves SSH.
Instalaciones defectuosas	Error en la instalación de servicios críticos como Apache, MySQL o PHP, provocando mal funcionamiento de la web.
Inyección de código o vulnerabilidades	Desarrollo de formularios de login o registro inseguros que permitan ataques como SQL Injection.
Configuración errónea de WireGuard	Creación incorrecta de túneles VPN que impidan el acceso seguro o generen caídas de la red.
Caídas del VPS	Inestabilidad del proveedor de VPS o problemas técnicos que dejen inaccesible el servidor.
Pérdida de datos	Borrado accidental o corrupción de la base de datos sin copias de seguridad actualizadas.
Errores en las consultas SQL	Fallos de funcionamiento de la página web debido a errores en la programación de las consultas o falta de validaciones.

- **Creación de plan de prevención de riesgos.**

Riesgo	Medida preventiva
Fallo en la conexión SSH	Comprobación de la configuración local antes de cerrar sesiones SSH, mantener una segunda clave SSH de respaldo.
Instalaciones defectuosas	Instalación controlada y validación tras cada servicio; realizar snapshots del sistema si el proveedor lo permite.
Inyección de código o vulnerabilidades	Validación de entradas del usuario, utilización de consultas preparadas (PDO o MySQLi en PHP).
Configuración errónea de WireGuard	Seguir documentación oficial y realizar pruebas de conectividad tras la configuración.
Caídas del VPS	Elegir un proveedor fiable y monitorizar el servicio para actuar en caso de caídas.
Pérdida de datos	Realizar copias de seguridad periódicas de la base de datos y de los archivos críticos.
Errores en las consultas SQL	Realizar pruebas unitarias de todas las funciones críticas que interactúan con la base de datos.

3.3 Revisión de recursos

- **Asignación de recursos (materiales y humanos) y temporización.**

Recurso	Tipo	Función en el proyecto	Asignación temporal
VPS (1 CPU, 2 GB RAM)	Hardware	Alojamiento del servidor web y base de datos	Todo el proyecto
Ubuntu Server 24.04	Software	Sistema operativo del servidor VPS	Todo el proyecto
Apache 2.4	Software	Servidor web para la página principal	Desde semana 2
MySQL 8.0	Software	Gestión de la base de datos del sitio	Desde semana 2
PHP 8.3.6	Software	Programación del backend dinámico	Desde semana 2
WireGuard	Software	Implementación de VPN como demostración	Semana 3
Visual Studio Code	Software	Edición del código fuente localmente	Todo el proyecto
Cliente SSH (Mobaxterm/Terminal Linux)	Software	Acceso y administración del VPS	Todo el proyecto
Conexión a Internet	Material	Conectividad continua para desarrollo y pruebas	Todo el proyecto
Alumno (yo)	Humano	Diseño, implementación y documentación	Todo el proyecto

- **Revisión del presupuesto diseñado en la fase anterior.**

Concepto	Coste aproximado
VPS (1 año, 2GB RAM, 40GB SSD)	12€
Dominio personalizado (1 año)	7€
Electricidad / Internet propio	Incluido
Software utilizado	0€ (opensource)

3.4 Documentación de ejecución

- **Ejecución del proyecto.**

La ejecución del proyecto se llevó a cabo siguiendo las fases previamente planificadas, asegurando en todo momento la correcta implantación y funcionamiento de cada componente:

1. Securización inicial del servidor VPS.

- Se accedió por primera vez al VPS con credenciales proporcionadas.

- Se generó un par de claves SSH, configurando la autenticación por clave pública en lugar de contraseña.

- Se cambió el puerto de SSH del puerto 22 a uno personalizado para reducir riesgos de escaneo automatizado.

- Se instaló y configuró Fail2ban para proteger el servidor contra intentos de fuerza bruta.

2. Instalación y configuración de servicios LAMP

- Instalación de Apache como servidor web.

- Instalación de MySQL para gestionar la base de datos.

- Instalación de PHP para el desarrollo backend de la página web.

- Configuración inicial de Apache para servir la futura aplicación web en el subdominio.

3. Configuración de WireGuard VPN

- Instalación de WireGuard en el servidor VPS.

- Generación de claves públicas y privadas.

- Creación de túneles de red segura para conexiones privadas.

- Comprobación de conexión VPN exitosa entre cliente y VPS.

4. Diseño y despliegue de la base de datos

- Creación del esquema de base de datos en MySQL.

- Implementación de tablas para usuarios, personajes, habilidades, mapas, vídeos, equipos y jugadores VCT.

- Relación de las tablas mediante claves foráneas según el modelo E/R diseñado.

5. Desarrollo backend

- Programación de la lógica de login, registro de usuarios y recuperación de contraseña.

- Implementación de consultas seguras a la base de datos usando sentencias preparadas.

6. Desarrollo frontend

- Creación de la página de inicio (Home) explicando la finalidad del sitio web.

- Implementación de la sección de personajes con filtro por rol.

- Desarrollo de páginas de detalle de cada personaje, mostrando habilidades y mapas recomendados.

- Integración de vídeos explicativos embebidos desde YouTube.

7. Pruebas funcionales y de seguridad

- Pruebas de funcionamiento de formularios (registro, login, recuperación).

- Verificación de correcta carga dinámica de personajes y mapas.

- Validaciones para prevenir inyecciones SQL y otros ataques básicos.

- Testeo de accesibilidad de la web desde diferentes dispositivos.

8. Documentación y entrega del proyecto

- Captura de pantallas de todas las implementaciones realizadas.

- Organización de los ficheros de configuración y código fuente.

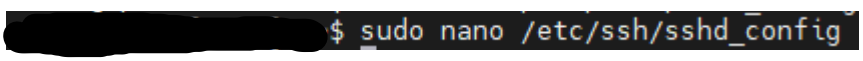
- Redacción de manuales de usuario, instalación y administración.

- **Ficheros de configuración**

Configuración de SSH

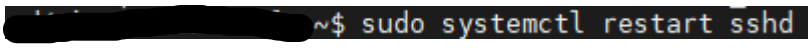
Para asegurar el acceso al servidor VPS:
- Se deshabilitó la autenticación por contraseña.
- Se configuró la autenticación por clave pública.
- Se cambió el puerto de conexión SSH para dificultar escaneos automáticos.

Fichero editado:



Cambios realizados:
· Port [redacted]
· PasswordAuthentication no
·PubkeyAuthentication yes
·PermitRootLogin no

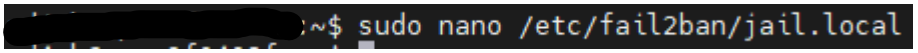
Después de editar, reinició el servicio SSH con



Configuración de Fail2ban

Fail2ban se configuró para proteger el servidor de ataques de fuerza bruta en SSH y otros servicios.

Archivo de configuración usado:



Configuración básica aplicada:
[DEFAULT]
bantime = 1d
findtime = 10m
maxretry = 3

[sshd]
enabled = true
port = [redacted]
logpath = /var/log/auth.log
bantime = 1d
findtime = 10m
maxretry = 3

Fail2ban se habilitó y se inició:
sudo systemctl enable fail2ban
sudo systemctl start fail2ban

Configuración de Apache

Se configuró Apache para servir la página web alojada en un subdominio.

Archivo de configuración principal:

```
[redacted]~$ sudo nano /etc/apache2/sites-available/tfg.adrianvillada.es.conf
```

Configuración realizada:

```
<VirtualHost [redacted]

        ServerAdmin [redacted]
        ServerName [redacted]
        DocumentRoot [redacted]
        DirectoryIndex [redacted]

        Redirect permanent / https://tfg.adrianvillada.es/

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

<VirtualHost [redacted]
    ServerAdmin [redacted]
    ServerName [redacted]
    DocumentRoot [redacted]
    DirectoryIndex [redacted]

    SSLEngine on
    SSLCertificateFile [redacted]
    SSLCertificateKeyFile [redacted]

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Activación del sitio y reinicio de Apache:

```
[redacted]~$ sudo a2ensite tfg.adrianvillada.es.conf
```

```
[redacted]~$ sudo systemctl reload apache2
```

Configuración de WireGuard

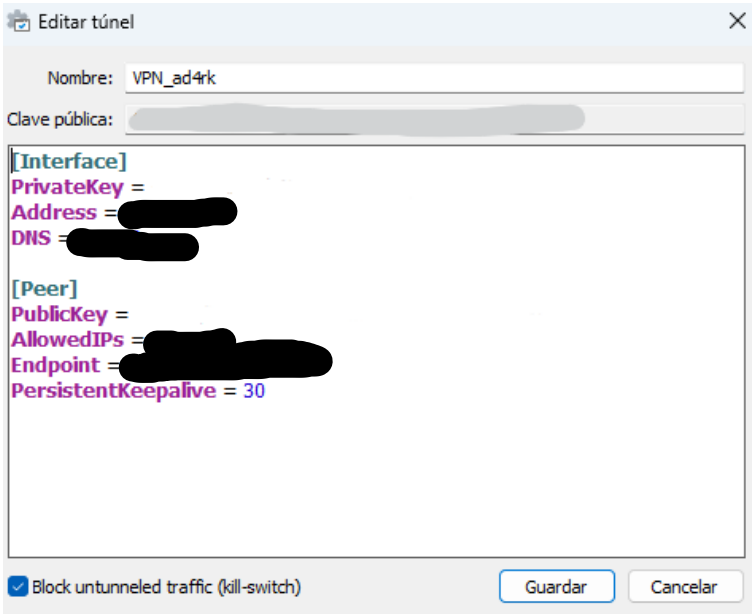
Se instaló WireGuard y se configuró una red privada para conexión VPN.

Archivo de configuración en el servidor:

```
[redacted]~$ sudo nano /etc/wireguard/wg0.conf
```

```
[Interface]
# Clave privada del servidor
PrivateKey = [redacted]

# IP interna que usará el servidor en la VPN
Address = [redacted]

# Puerto de escucha de WireGuard
ListenPort = [redacted]

# Permitir reenvío de paquetes y salida a Internet para los clientes
PostUp = iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -o ens3 -j MASQUERADE
PostDown = iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D POSTROUTING -o ens3 -j MASQUERADE

# Guardar la configuración al apagar el servidor
SaveConfig = true

[Peer]
#Adri
PublicKey = [redacted]
AllowedIPs = [redacted]
```

Archivo de configuración en el cliente:



Activación del servicio WireGuard:

```
[redacted]:~$ sudo systemctl start wg-quick@wg0
```

```
[redacted]:~$ sudo systemctl enable wg-quick@wg0
```

- **Configuración de dispositivos de red**

Aunque en este proyecto no se utilizaron dispositivos físicos de red como routers o switches, sí se configuró una red privada virtual (VPN) mediante WireGuard, lo que implicó establecer una topología lógica entre el cliente y el servidor.

Configuración de la red VPN (WireGuard)

Se configuró una interfaz de red virtual (wg0) tanto en el servidor (VPS) como en el cliente local. Esto permitió establecer un túnel cifrado punto a punto para fines de prueba y demostración de habilidades en redes privadas seguras.

· Datos de configuración:

- Interfaz del servidor: wg0

- IP del servidor VPN (interna): [redacted]

- IP del cliente VPN (interna): [redacted]

- Puerto utilizado: [redacted]

- Protocolo: UDP

- Rango de red virtual: [redacted]

· Seguridad:

- Claves públicas y privadas generadas con wg genkey y wg pubkey.

- Sólo se permitió el acceso desde el cliente autorizado [redacted].

· Comprobación de la conectividad:

Una vez levantado el túnel VPN:

- Se pudo hacer ping desde el cliente al servidor (ping [redacted]

- El tráfico entre cliente y servidor viajó cifrado por el túnel VPN.

- **Características técnicas**

A continuación, se detallan las características técnicas principales del sistema desarrollado, tanto a nivel de infraestructura como de software:

INFRAESTRUCTURA

Elemento	Detalles técnicos
Servidor VPS	1 vCPU, 2 GB RAM, 40 GB SSD, IP pública dedicada
Sistema operativo	Ubuntu Server 24.04
Conexión remota	SSH por clave pública (puerto personalizado 2222)
VPN implementada	WireGuard (IP interna [redacted], puerto [redacted]/UDP)

SERVICIOS INSTALADOS

Servicio	Función
Apache 2.4	Servidor web para alojar la aplicación
MySQL 8.0	Sistema de gestión de base de datos
PHP 8.3.6	Lógica del backend dinámico
Fail2ban	Protección contra ataques de fuerza bruta
WireGuard	Red privada virtual (VPN)

APLICACIÓN WEB

Elemento	Características clave
Página de inicio (Home)	Explicación del propósito del sitio
Login, registro y recuperación	Gestión segura de usuarios con recuperación vía email
Base de datos	Relacional, optimizada y estructurada por tablas de personajes, mapas, habilidades...
Contenido dinámico	Cargado por PHP y MySQL, incluyendo filtros, detalles de personajes, vídeos embebidos
Seguridad web	Consultas preparadas, validaciones, acceso controlado

- **Código fuente (o web)**

He enviado un correo con todos los archivos de configuración a[redacted]
Por lo que explicaré brevemente los archivos de configuración más relevantes, la función que tiene y como se representa visualmente.

archivos_css/styles.css

Función: Contiene todos los estilos visuales del proyecto, desde formularios hasta botones, navbar, tarjetas, etc.

Notas: Centraliza el diseño para asegurar consistencia entre páginas.

archivos_php/conexion.php

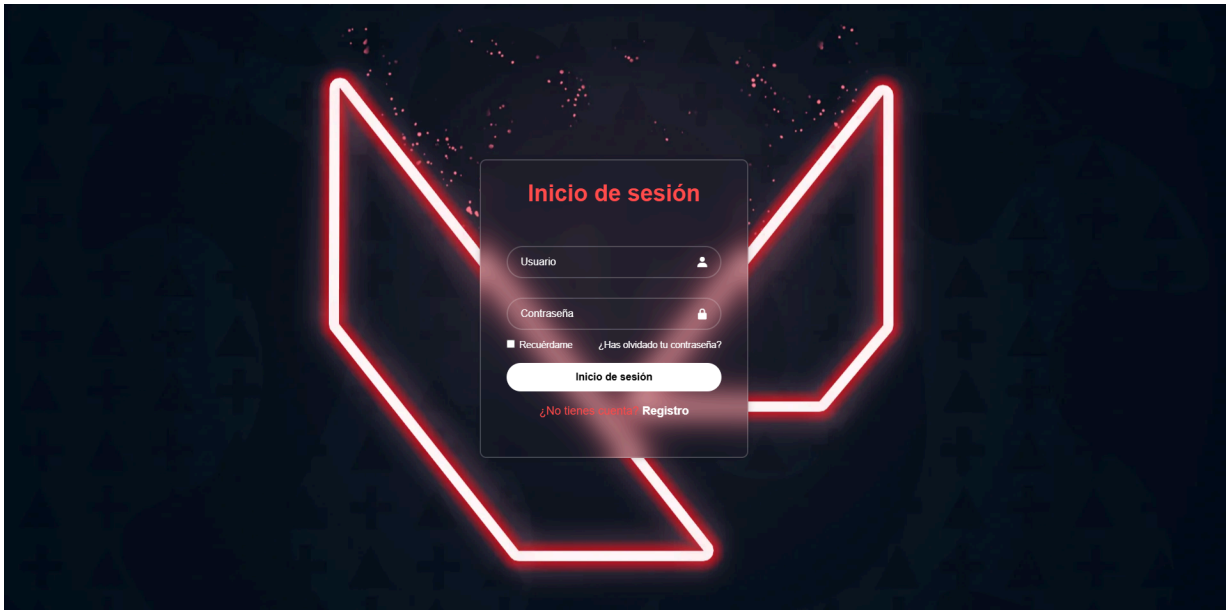
Función: Establece la conexión con la base de datos usando variables de entorno para proteger credenciales.

archivos_html/ index.php

Función: Página de inicio de sesión del usuario.

Muestra: Formulario de login con validación, recuerda usuario si está en cookies y muestra mensajes de error o éxito.

Relación: Envia los datos a login.php.



login.php

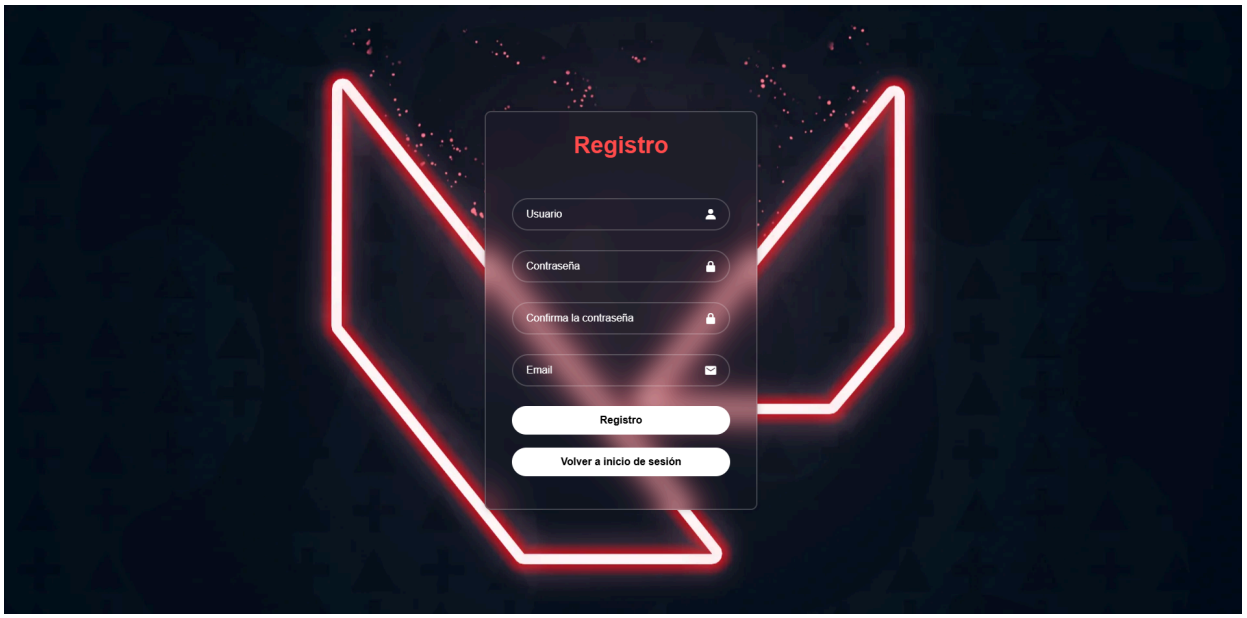
Función: Valida usuario y contraseña. Si son correctos, inicia sesión y redirige a dashboard.php.

Register1.php

Función: Página de registro de usuario.

Muestra: Formulario con campos para usuario, email, contraseña y confirmación.

Relación: Envia los datos a register.php.



register.php

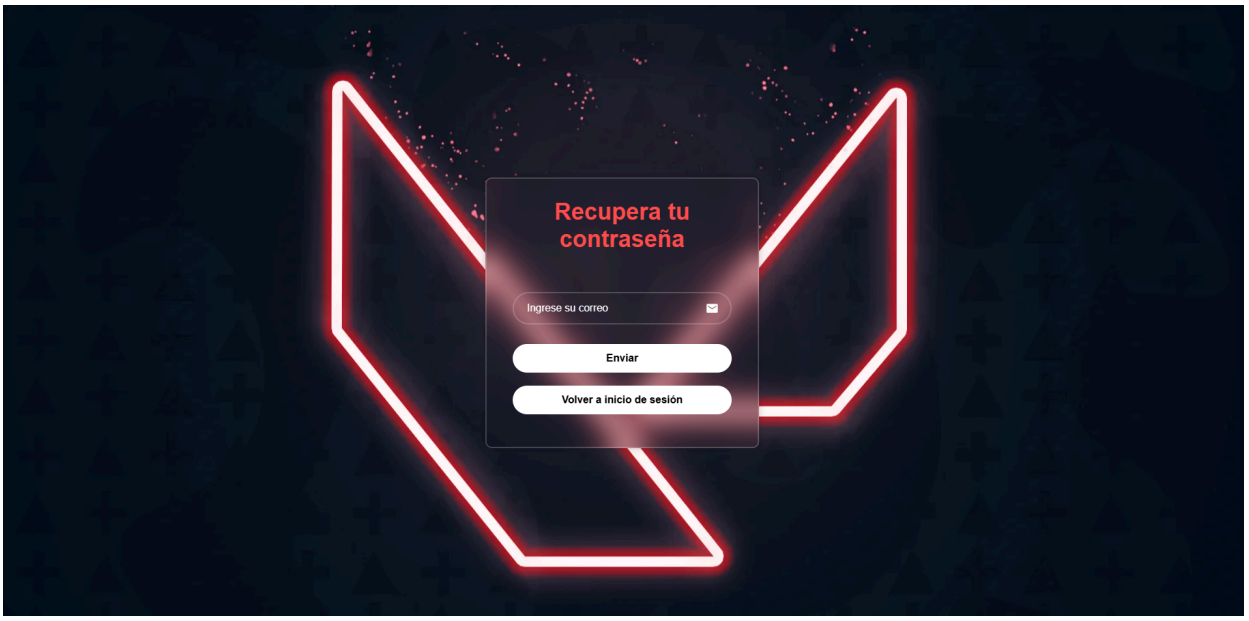
Función: Procesa el formulario de registro, valida datos, comprueba duplicados y registra en base de datos.

Redirección: Redirige a index.php?success=registro_exitoso si todo va bien.

lostpassword.php

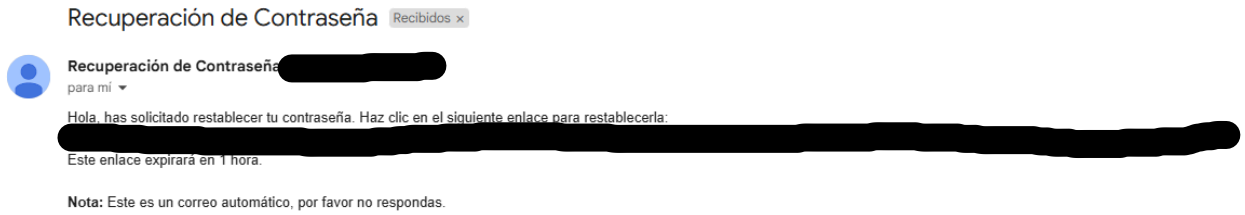
Función: Formulario para solicitar restablecer contraseña.

Relación: Envia email a través de process_lostpassword.php.



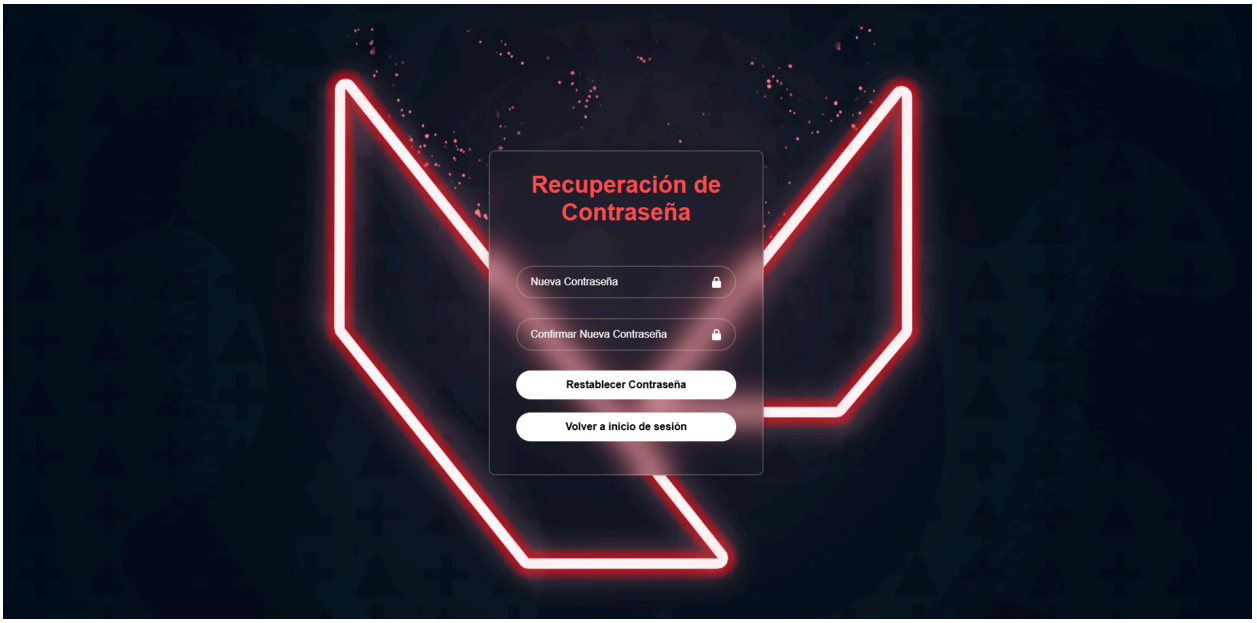
process_lostpassword.php

Función: Genera un token de recuperación de contraseña, lo almacena en la base de datos y envía un correo al usuario con el enlace.



reset_password.php

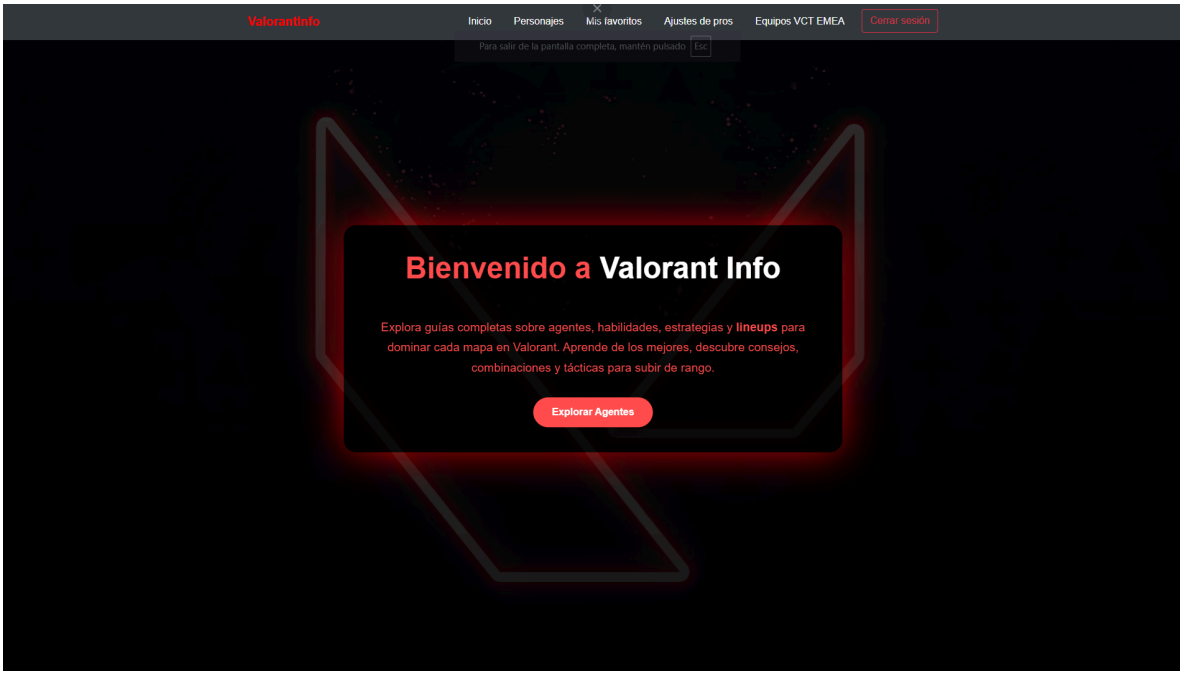
Función: Permite al usuario establecer una nueva contraseña si el token es válido y no ha expirado.



dashboard.php

Función: Página de bienvenida tras iniciar sesión, con mensaje y botón para explorar personajes.

Muestra: Sección de presentación.

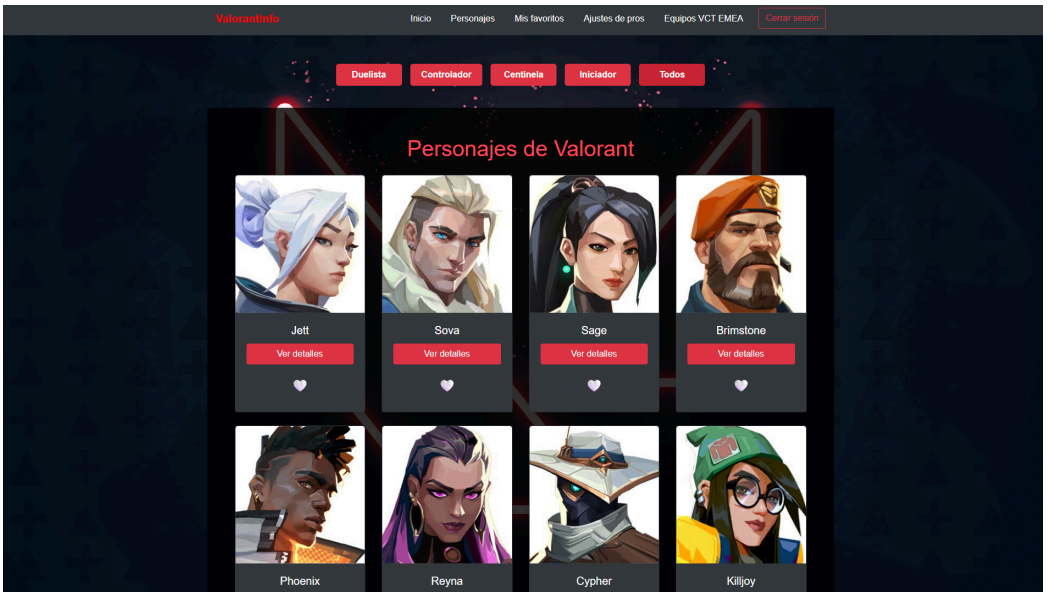


personajes.php

Función: Página principal de visualización de personajes.

Filtros: Permite filtrar por rol (Duelista, Centinela, etc.).

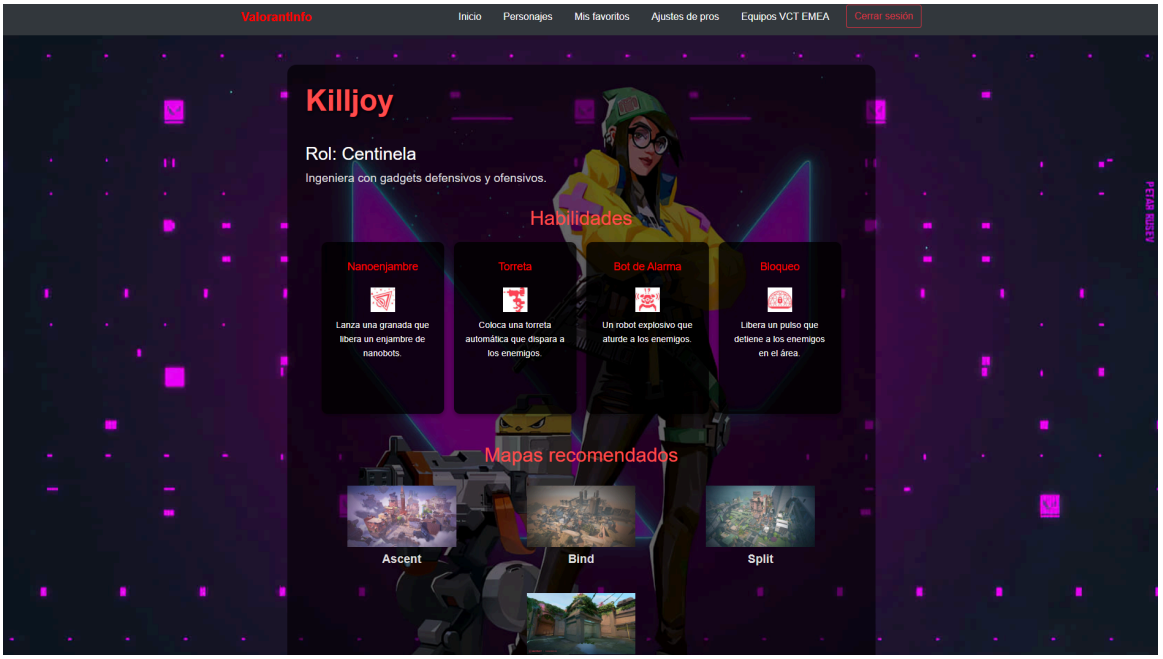
AJAX: Incluye sistema de favoritos (corazón).



detalle_personaje.php

Función: Página de detalle de un personaje con sus habilidades, mapas recomendados y vídeos embebidos de YouTube.

Consulta: Datos dinámicos desde base de datos personajes.



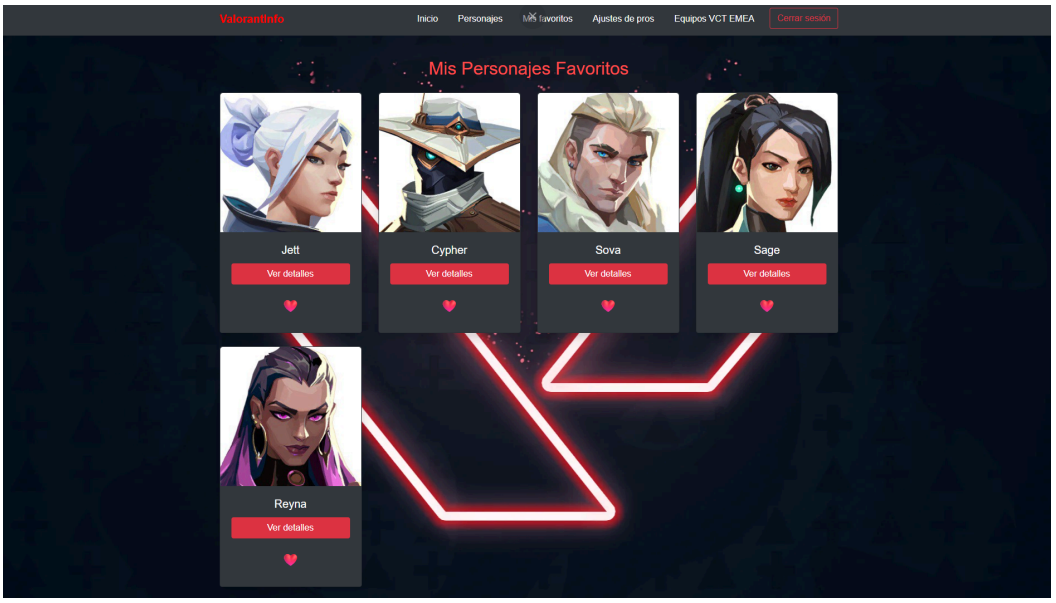
favorito_toggle.php

Función: Script que añade o elimina personajes favoritos mediante peticiones AJAX.

Consulta: Tabla favoritos.

mis_favoritos.php

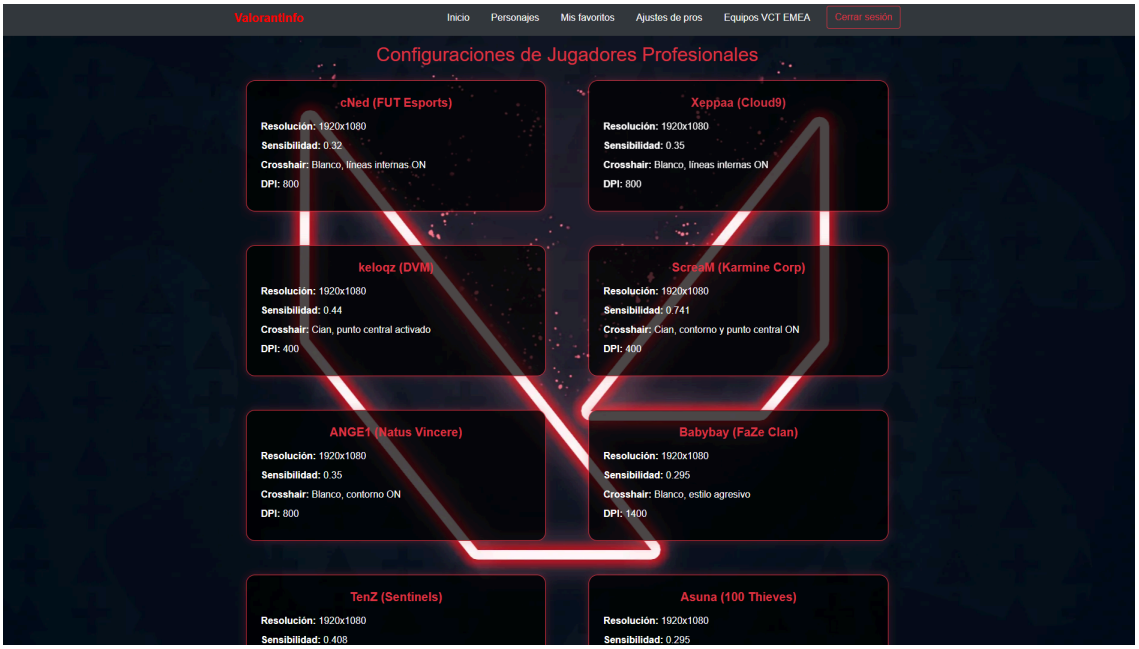
Función: Página que muestra los personajes marcados como favoritos por el usuario actual.



pro_settings.php

Función: Página de ajustes de jugadores profesionales (resolución, crosshair, sensibilidad...).

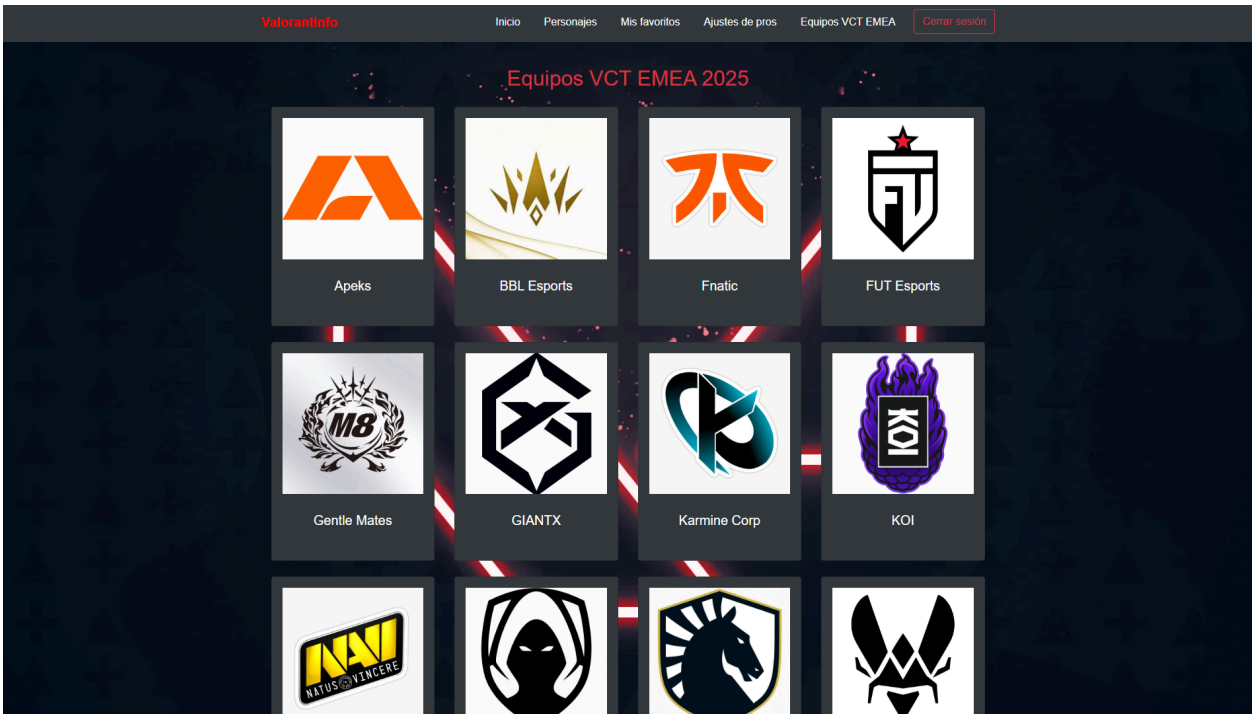
Datos: Ahora cargados dinámicamente desde la base de datos (ajustes_pro).



equipos_vct.php

Función: Página con todos los equipos de la VCT.

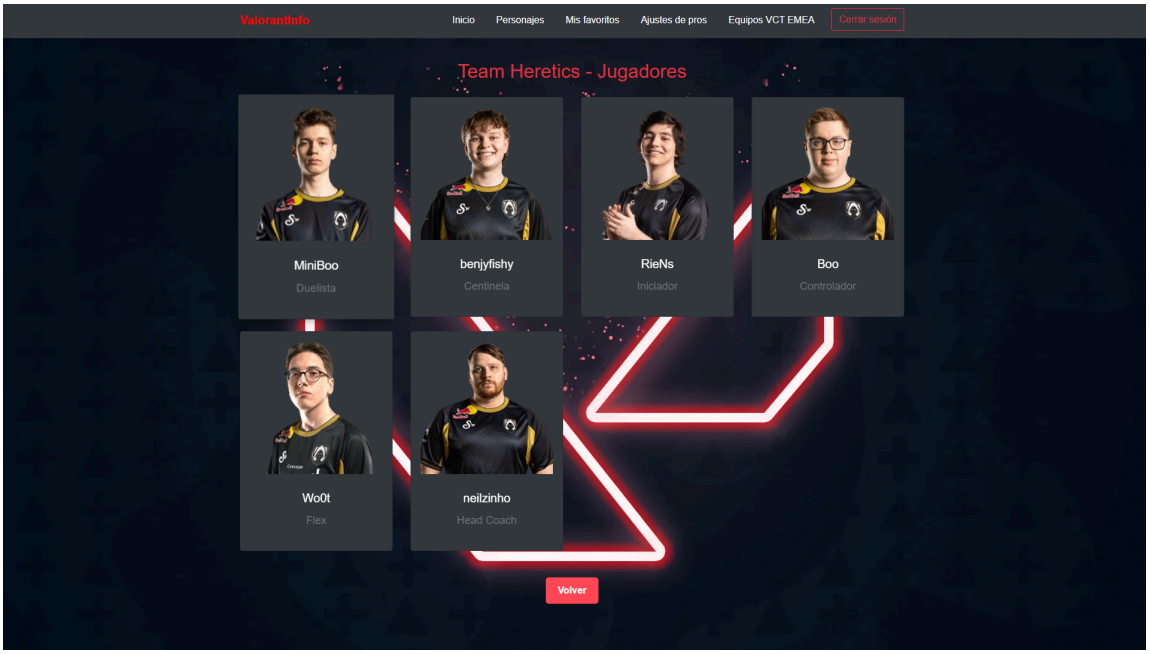
Muestra: Tarjetas con logos de equipos que enlazan a detalle_equipo.php.



detalle_equipo.php

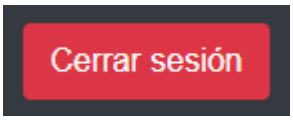
Función: Muestra los jugadores de un equipo seleccionado desde equipos_vct.php.

Consulta: Relaciona equipos_vct con jugadores_vct.



logout.php

Función: Cierra la sesión y redirige a index.php.



- **Base de datos implementada**

La base de datos implementada está optimizada para almacenar de forma relacional toda la información necesaria para el funcionamiento de la página web, excepto una tabla que estará de forma aislada. Se compone de las siguientes tablas principales:

Tabla users:
Almacena los usuarios registrados en el sistema.

Campo	Tipo	Descripción
id	INT (PK)	Identificador único
username	VARCHAR	Nombre de usuario
password	VARCHAR	Contraseña cifrada
email	VARCHAR	Correo del usuario
reset_token	VARCHAR	Token de recuperación de contraseña
reset_expiry	DATETIME	Fecha de expiración del token

Tabla personajes:
Contiene la información básica de cada personaje.

Campo	Tipo	Descripción
id	INT (PK)	Identificador del personaje
nombre	VARCHAR	Nombre del personaje
descripcion	TEXT	Descripción general
imagen	VARCHAR	Ruta de imagen del personaje
imagen_fondo	VARCHAR	Imagen de fondo
rol_id	INT (FK)	Relación con la tabla roles

Tabla roles:
Define los roles posibles de los personajes (por ejemplo: Duelista, Centinela, etc.).

Campo	Tipo	Descripción
id	INT (PK)	Identificador único
nombre	VARCHAR	Nombre del rol

Tabla habilidades:
Relaciona las habilidades con los personajes.

Campo	Tipo	Descripción
id	INT (PK)	Identificador de habilidad
personaje_id	INT (FK)	Relación con personajes
nombre	VARCHAR	Nombre de la habilidad
descripcion	TEXT	Descripción funcional
imagen	VARCHAR	Imagen ilustrativa (opcional)

Tabla mapas:
Listado de mapas disponibles en el juego.

Campo	Tipo	Descripción
id	INT (PK)	Identificador del mapa
nombre	VARCHAR	Nombre del mapa
imagen	VARCHAR	Imagen del mapa

Tabla recomendaciones:
Asocia personajes con mapas recomendados.

Campo	Tipo	Descripción
id	INT (PK)	Identificador
personaje_id	INT (FK)	Personaje recomendado
mapa_id	INT (FK)	Mapa recomendado

Tabla videos:
Contiene vídeos relacionados con personajes y mapas.

Campo	Tipo	Descripción
id	INT (PK)	Identificador
personaje_id	INT (FK)	Relación con personajes
mapa_id	INT (FK)	Relación con mapas
titulo	VARCHAR	Título del vídeo
url	TEXT	URL del vídeo embebido (YouTube)

Tabla favoritos:
Permite a los usuarios marcar personajes como favoritos.

Campo	Tipo	Descripción
id	INT (PK)	Identificador
usuario_id	INT (FK)	Usuario que marca el favorito
personaje_id	INT (FK)	Personaje favorito

Tablas equipos_vct y jugadores_vct:

Relacionadas con el contenido extra de jugadores profesionales.

Tabla equipos_vct:

Campo	Tipo	Descripción
id	INT (PK)	Identificador del equipo
nombre	VARCHAR	Nombre del equipo
logo	VARCHAR	Ruta al logo

Tabla jugadores_vct:

Campo	Tipo	Descripción
id	INT (PK)	Identificador del jugador
nombre	VARCHAR	Nombre del jugador
imagen	VARCHAR	Imagen del jugador
rol_id	INT (FK)	Rol del jugador (relación con roles)
equipo_id	INT (FK)	Relación con equipos_vct

Tabla ajustes_pro:

Tabla independiente y estática que almacena configuraciones utilizadas por jugadores profesionales, streamers o creadores de contenido. No está relacionada directamente con otras tablas, ya que su objetivo es únicamente informativo para los usuarios.

Campo	Tipo	Descripción
id	INT (PK)	Identificador único de cada configuración
nombre	VARCHAR(100)	Nombre del jugador profesional o creador de contenido
equipo	VARCHAR(100)	Nombre del equipo o marca asociada (si aplica)
resolucion	VARCHAR(50)	Resolución de pantalla utilizada
sensibilidad	FLOAT	Sensibilidad del ratón usada
crosshair	VARCHAR(255)	Configuración del punto de mira (texto o JSON simple)
dpi	INT	DPI del ratón

- **Política de seguridad implementada**

La seguridad fue uno de los pilares fundamentales del proyecto, y se implementaron diferentes mecanismos de protección desde el inicio para garantizar la integridad, confidencialidad y disponibilidad del sistema.

Seguridad en el servidor (VPS)

· **Acceso SSH seguro**

- Se generó un par de claves SSH y se configuró el servidor para aceptar únicamente autenticación mediante clave pública.

- Se cambió el puerto por defecto (22) por un puerto alternativo personalizado ([redacted]) para evitar escaneos automatizados.

- Se desactivó el acceso por contraseña modificando /etc/ssh/sshd_config.

· **Fail2ban**

- Se instaló y configuró Fail2ban para proteger el acceso SSH de ataques de fuerza bruta.

- Se estableció un número máximo de intentos fallidos antes de banear temporalmente la IP del atacante.

· VPN con WireGuard

- Se configuró un túnel VPN como demostración funcional de conexión segura entre cliente y VPS.

- Se generaron claves criptográficas y se utilizaron direcciones privadas para proteger las conexiones.

Seguridad en la aplicación web

· Gestión de contraseñas

- Las contraseñas de los usuarios se almacenan cifradas mediante password_hash() con algoritmo bcrypt.

- Para el login, se valida la contraseña usando password_verify().

· Protección contra inyecciones SQL

- Se utilizaron consultas preparadas (prepared statements) en todos los formularios que interactúan con la base de datos (login, registro, recuperación...).

- Se validan y escapan las entradas del usuario.

· Recuperación segura de contraseñas

- Se genera un token único y aleatorio para cada recuperación.

- Este token tiene una fecha de expiración y se almacena de forma segura en la base de datos.

- Se envía un correo con el enlace de recuperación personalizado.

· Control de acceso y sesiones

- Se utilizan sesiones ($_SESSION) para mantener la autenticación del usuario una vez accede correctamente.

- Se restringe el acceso a páginas privadas si el usuario no ha iniciado sesión.

Buenas prácticas generales

- Se deshabilitaron índices de directorio en Apache para evitar listar archivos por error.

- Se revisaron los permisos de archivos y carpetas, limitando la escritura a lo estrictamente necesario.

- Se restringieron los métodos HTTP permitidos, aceptando únicamente GET y POST.

- Se realizó una revisión final del código para detectar errores comunes de seguridad.

- **Reglas (GPOs, Iptables, ACLs, …) implementadas**

- **Copia de seguridad del sitio web**

Actualmente no tengo configurada una copia de seguridad automatizada que se ejecute cada cierto tiempo. Sin embargo, realizo copias manuales de forma periódica para guardar los archivos importantes del proyecto.

Utilizo MobaXterm, una herramienta de administración remota que permite conectarse al VPS vía SSH y acceder a su sistema de archivos.
Desde su entorno gráfico, simplemente arrastro y copio el directorio completo donde se encuentra mi página web (a mi equipo local.

Si en un futuro quiero tenerlo automatizado, implementaría copias automatizadas mediante cron.

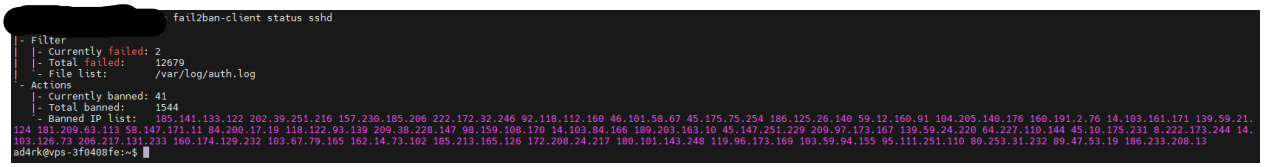
- **En general:**

- **Texto: Descripción de las implementaciones realizadas**
- **Capturas de pantallas de las implementaciones realizadas**

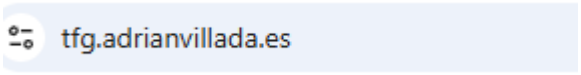
· Acceso SSH con clave privada y puerto cambiado
Durante la fase de securización del servidor, se configuró el acceso SSH por clave pública y se modificó el puerto por defecto

· Fail2ban en funcionamiento

Configuración del servicio Fail2ban para evitar ataques de fuerza bruta en el servicio SSH.

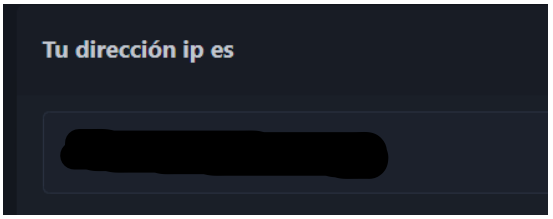


· Apache sirviendo la página en el subdominio



· Conexión VPN activa con WireGuard

Ejemplo dirección IP pública con VPN activa.



Implementación adicional: Monitorización del servidor con Netdata.

Para supervisar el rendimiento del servidor VPS donde se aloja la web, se ha instalado Netdata, una herramienta de monitorización ligera y en tiempo real. Esta solución permite obtener información detallada del sistema, como el uso de CPU, memoria, tráfico de red, actividad de disco y procesos activos.

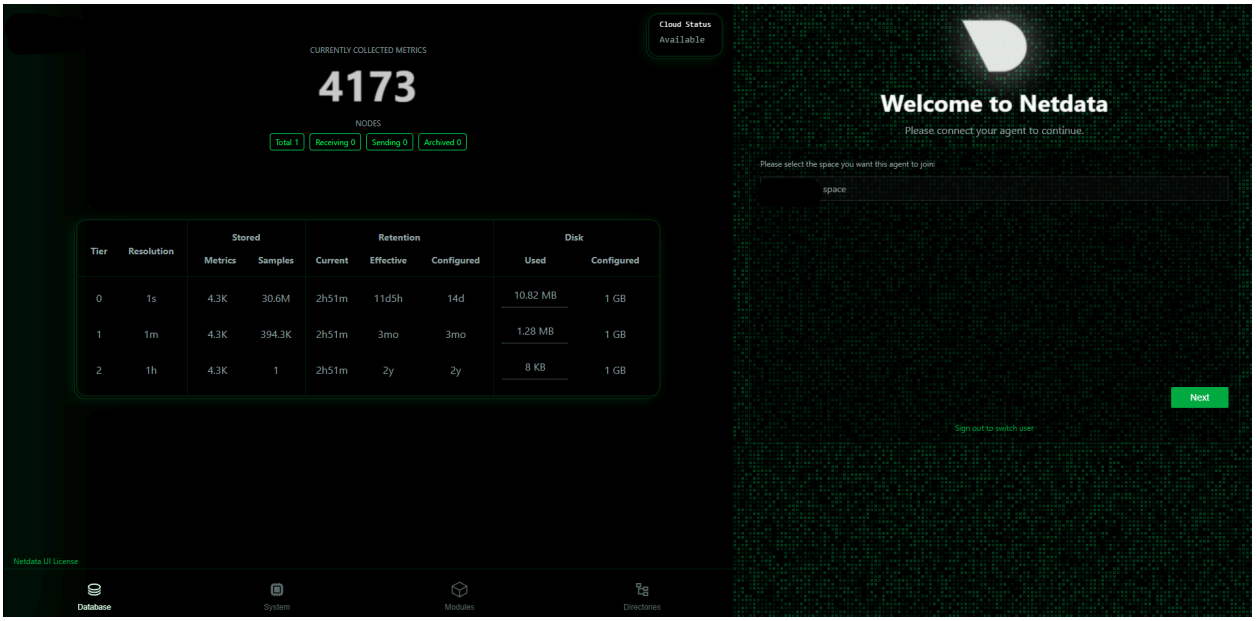
La instalación se ha realizado mediante el siguiente comando:

```
bash <(curl -L -Ss https://my-netdata.io/kickstart.sh)
```

Tras la instalación, el panel de Netdata es accesible desde el navegador web a través del puerto 19999

A continuación, se muestra una captura del panel principal de Netdata:

Netdata facilita el diagnóstico de posibles problemas de rendimiento y permite tener un control continuo del estado del servidor de forma visual e intuitiva.



- **Manuales de usuario**

Instrucciones básicas:

1. Acceder a la web mediante el subdominio configurado.

2. Usar el formulario de registro para crear una cuenta.

3. Acceder con el formulario de login usando su nombre de usuario y contraseña.

4. Visualizar los contenidos disponibles:

- Página de inicio con explicación del sitio.

- Filtrar personajes por rol.

- Ver detalles de cada personaje, sus habilidades, mapas recomendados y vídeos.

- Ver equipos profesionales (VCT) y sus jugadores.

- Consultar la sección de ajustes profesionales de jugadores.

5. Si olvida su contraseña:

- Acceder a "¿Has olvidado tu contraseña?".

- Introducir el correo.

- Acceder al enlace recibido para establecer una nueva contraseña.

- **Manuales de instalación**

Manual de instalación.

· Requisitos:
- VPS con Ubuntu Server 24.04

- Acceso por SSH

- Dominio y subdominio configurado

· Pasos de instalación:

1. Actualizar el sistema:

```
sudo apt update && sudo apt upgrade
```

2. Instalar Apache, MySQL y PHP:

```
sudo apt install apache2 mysql-server php libapache2-mod-php php-mysql
```

3. Copiar los archivos del proyecto a /var/www/html/dominio_tfg
Puedes hacerlo con MobaXterm o SCP.

4. Importar la base de datos:

```
mysql -u root -p < backup_bd.sql
```

5. Configurar Apache con un VirtualHost para el subdominio.

6. Dar permisos a los archivos:

```
[redacted]~$ sudo chown -R www-data:www-data /var/www/html/dominio_tfg
[redacted]~$ sudo chmod -R 755 /var/www/html/dominio_tfg
```

7. Reiniciar Apache:

```
sudo systemctl restart apache2
```

- **Manuales de Configuración y administración**

Tareas de administración habituales:

· Crear nuevas copias de seguridad manuales:

- Archivos: con tar
- Base de datos: con mysqldump

· Modificar usuarios o contenido: directamente desde phpMyAdmin o MySQL.

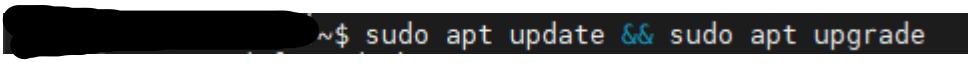
· Revisar funcionamiento del sistema:

- Logs de Apache: /var/log/apache2/

- Logs del sistema: journalctl, /var/log/syslog

· Verificar el estado de la VPN:

· Actualizar el sistema:



· Recomendaciones:
- No modificar la base de datos manualmente sin respaldo previo.

- Mantener las claves SSH seguras.

- Realizar mantenimiento mensual del VPS y la web.

4. Seguimiento y control

4.1 Valoración del proyecto

- **Definir medios de evaluar el proyecto.**

· El proyecto puede ser evaluado mediante los siguientes criterios:

- Funcionamiento correcto de la aplicación web.

- Seguridad aplicada en el servidor y en el desarrollo.

- Acceso a través de subdominio correctamente configurado.

- Organización estructurada de base de datos y código.

- Documentación completa y estructurada del proceso.

- **Definir los indicadores de calidad del proyecto.**

Indicador	Criterio de éxito
Funcionamiento del login	Acceso con credenciales válidas sin errores
Recuperación de contraseña	Enlace funcional y cambio correcto
Visualización de personajes	Filtrado por rol y detalle dinámico
Seguridad en el servidor	Clave SSH, puerto personalizado, fail2ban activo
Respuesta del servidor	Página accesible desde cualquier navegador
Navegación de usuario	Fluidez y comprensión del sitio sin errores

- **Elaboración del protocolo de evaluación con el cliente y su documentación.**

· Protocolo de evaluación

- Comprobación funcional de cada módulo.

- Validación de la base de datos con datos reales.

- Test de seguridad básica: acceso SSH, protección contra inyecciones, bloqueo de IPs con fail2ban.

- Evaluación externa (profesor/tutor) basada en funcionamiento real y documentación entregada.

4.2 Incidencias

- **Definir un protocolo para resolución de incidencias:**

- **Recopilación de información**
- **Posible solución**
- **Registro**

Protocolo para resolución de incidencias:

1. Recopilación de información

Anotar con precisión el error, qué lo genera y cómo reproducirlo.

2. Análisis del problema

Consultar logs (/var/log/apache2/error.log, /var/log/syslog, etc.) o mensajes de error en pantalla.

3. Solución propuesta y aplicación

Buscar en documentación o foros, aplicar parches o ajustes en código/configuración.

4. Registro y verificación

Documentar la incidencia y cómo se resolvió. Verificar que el problema no se repite.

4.3 Cambios

- **Adaptación a los cambios y registro:**

- **Migración**

Elemento	Cambio realizado	Justificación
Acceso web	De entorno local (XAMPP) a VPS con Apache	Publicar la web real desde la nube
Base de datos	De pruebas locales a MySQL en el VPS	Mantener datos centralizados

- **Actualización**

Elemento	Actualización	Justificación
Estructura de la BBDD	Añadida tabla ajustes_pro y relaciones extra	Adaptación a nuevas funcionalidades
Código fuente	Refactorización y mejora del login/registro	Mayor seguridad y limpieza de código
Seguridad	Ajustes en Fail2ban y reglas de firewall	Reforzar protección ante amenazas

- **Escalabilidad**

Elemento	Adaptación	Justificación
Base de datos relacional	Diseño modular y normalizado	Permite añadir más tablas fácilmente
Sistema de usuarios	Previsto soporte para roles diferenciados	Escalable a sistemas multirol en el futuro
Organización de código	Separación lógica (PHP, HTML, CSS)	Permite ampliaciones sin romper estructura

- **Mejoras**

Elemento	Mejora aplicada	Justificación
Diseño web	Secciones nuevas: equipos VCT, ajustes pro	Aumentar el valor del contenido
Seguridad en el backend	Consultas preparadas, token de recuperación	Evitar ataques comunes (SQLi)
Experiencia de usuario	Sistema de favoritos	Personalización y navegación más amigable

4.4 Pruebas y soporte

- **Registro de las pruebas realizadas:**

- **de red**
- **de carga**
- **de seguridad**
- **de acceso**

Tipo de prueba	Descripción	Resultado esperado	Resultado obtenido
Prueba de red	Verificar conectividad mediante VPN y acceso HTTP al servidor VPS	Acceso web desde cliente vía subdominio	Correcto
Prueba de carga	Comprobación de acceso simultáneo con varias pestañas y usuarios	No se detectan caídas o lentitud notable	Correcto
Prueba de seguridad	Intentos de inyección SQL, acceso a páginas privadas sin sesión	Sistema resistente, entradas validadas	Correcto
Prueba de acceso	Registro, login, recuperación de contraseña con datos válidos e inválidos	Acceso correcto o mensajes de error adecuados	Correcto

- **Copias de seguridad**

- Archivos web: copiados manualmente desde MobaXterm al equipo local.

- Base de datos: exportada mediante mysqldump.

- Se realizaron backups puntuales tras fases clave del desarrollo.

- **Elaboración de un acuerdo de servicio.**

Aunque el sistema no está destinado a producción real, se define un acuerdo básico de servicio para establecer expectativas:

Aspecto	Compromiso
Disponibilidad	El sistema debería estar accesible 24/7 mientras el VPS esté en línea
Tiempo de respuesta	En caso de error o caída, recuperación estimada < 24h
Seguridad	Contraseñas cifradas, tráfico VPN seguro, acceso controlado
Mantenimiento	Actualizaciones del sistema y código al menos una vez al mes
Copias de seguridad	Manuales, realizadas cada vez que se actualiza el contenido

Conclusión y propuesta de mejora

El proyecto ha cumplido con los objetivos iniciales: una web funcional, dinámica, segura y alojada en un entorno real administrado por el propio alumno. A nivel técnico, la estructura del backend y de la base de datos está preparada para escalar de forma progresiva si el sistema creciera.

· Escalabilidad del sistema (futuro crecimiento)

Si el número de usuarios o servicios aumentara, podrían aplicarse las siguientes mejoras:

- Separación de servicios: migrar la base de datos a un servidor independiente para mejorar rendimiento.

- Balanceo de carga: implementar un proxy inverso como Nginx o HAProxy para repartir el tráfico entre varios servidores.

- Sistema de caché: usar herramientas como Redis o memcached para reducir consultas repetidas.

- Roles avanzados: añadir nuevos tipos de usuarios con permisos distintos (moderadores, editores...).

- Panel de administración: crear un backend para gestionar contenido (CRUD completo).

- Sistema automatizado de copias de seguridad y alertas de estado del servidor.

- Contenedores y orquestación: migrar el entorno a Docker y escalar con Kubernetes si el crecimiento es muy alto.

Estas mejoras permitirían mantener la eficiencia, la seguridad y la estabilidad del sistema si este pasara de un entorno académico a un entorno profesional real.