

Administración de Sistemas

Proyecto: Aplicación Web con Docker

Patricia Ortega García

3 de diciembre de 2023

Índice

1. Introducción	2
2. Descripción de la aplicación	2
3. Docker Compose	9
4. Tareas requeridas	9
4.1. Servidor de BBDD	9
4.2. Servidor Web	9
4.3. Python	10
5. Tareas adicionales realizadas	11
5.1. API RAWG	11
5.2. phpMyAdmin	11
5.3. cAdvisor	12
5.4. Node Exporter	12
5.5. Prometheus	12
6. Atribuciones	13

1. Introducción

En este proyecto de la asignatura Administración de Sistemas (AS), en el curso 2023-2024, se ha creado una aplicación web con distintas funcionalidades que más adelante serán explicadas. Para el desarrollo del trabajo se han utilizado imágenes de Docker.

El trabajo realizado puede encontrarse en el siguiente repositorio de GitHub:

<https://github.com/patricia-ortega-garcia/AS-Proyecto>

2. Descripción de la aplicación

La aplicación desarrollada es una aplicación web llamada **Goodgames**. En esta página se podrá acceder a información sobre una gran lista de videojuegos, además de contar con un sistema de cuentas que permite a los usuarios registrarse en la página.

Desde la página principal, al usuario se le presentan 3 opciones:

- Si el usuario tiene una cuenta creada, podrá iniciar sesión introduciendo sus credenciales y pulsando el botón **Iniciar Sesión** desde la página principal (fig. 1).

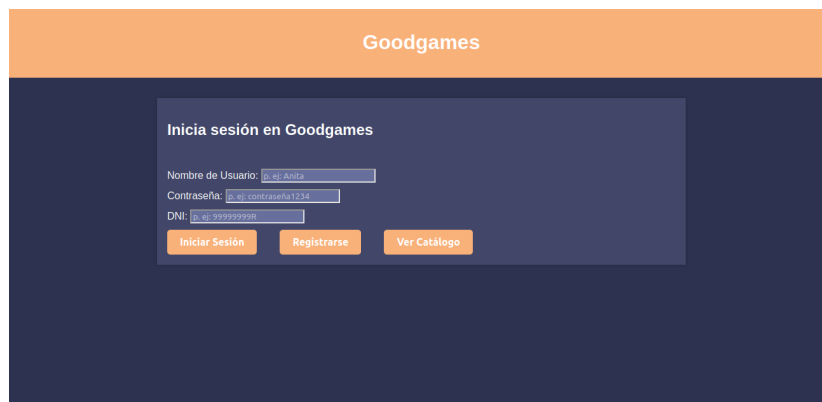


Figura 1: Página principal

- Por otro lado, si el usuario no tiene una cuenta creada, se le permite acceder a un formulario de registro pulsando el botón **Registrarse**, mediante el cual será redirigido a una página donde deberá introducir todos los datos requeridos para crear su cuenta (fig. 2). Una vez introducidos los datos, pulsará el botón **Registrarse** para llevar a cabo el registro. Si el usuario

cambia de idea respecto al registro, podrá volver a la página principal pulsando el botón **Volver a Inicio Sesión**.

Goodgames

Formulario de Registro

Nombre:

Apellidos:

DNI:

Teléfono:

Fecha de Nacimiento:

Email:

Nombre de Usuario:

Contraseña:

Figura 2: Formulario de registro

- Por último, el usuario puede acceder al catálogo de videojuegos (fig. 3) sin pasar por el registro ni el inicio de sesión, pulsando el botón **Ver Catálogo**. A esta pestaña también se accederá tras registrarse o iniciar sesión.

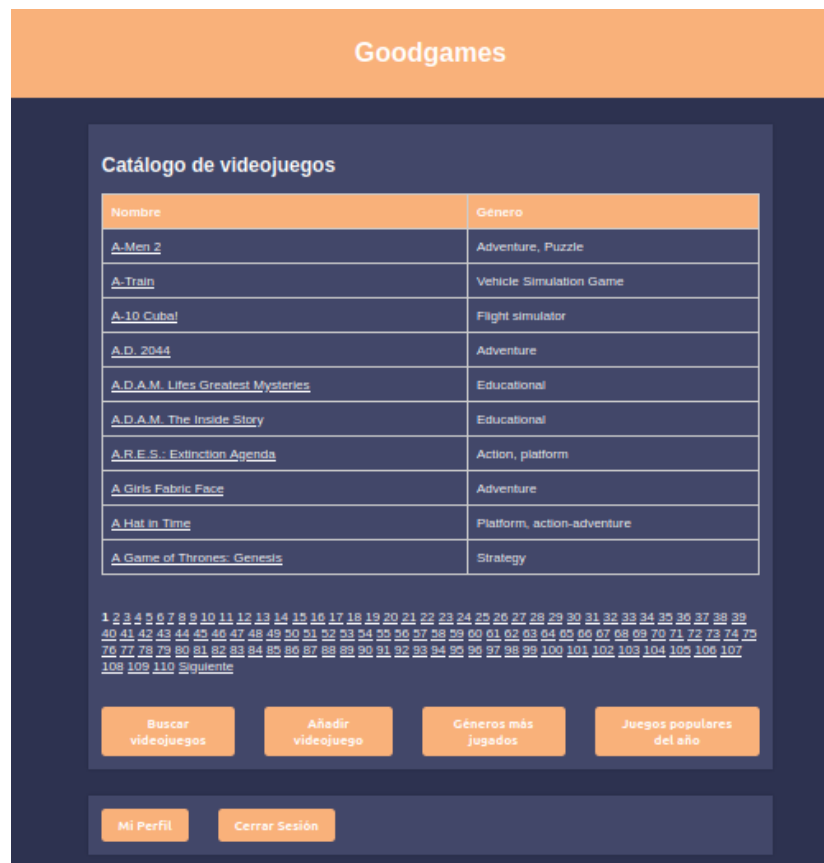


Figura 3: Catálogo de videojuegos

Desde el catálogo de videojuegos el usuario tiene bastantes posibilidades:

- Pulsando el botón **Añadir videojuego** el usuario es redirigido a un formulario (fig. 4) donde puede introducir los datos del videojuego que quiere añadir al catálogo.

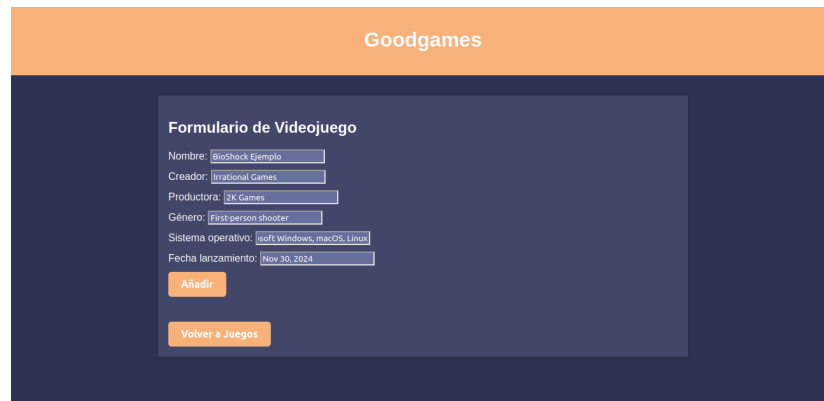


Figura 4: Añadir videojuegos

- Pulsando el botón **Buscar videojuego** el usuario es redirigido a un formulario (fig. 5) donde puede introducir los datos de los videojuego que quiere buscar en el catálogo. Puede introducir información en tantos campos como desee. Una vez pulsado el botón de **Buscar** se ve una lista con los juegos que cumplen los campos introducidos, si es que los hay (fig. 6).

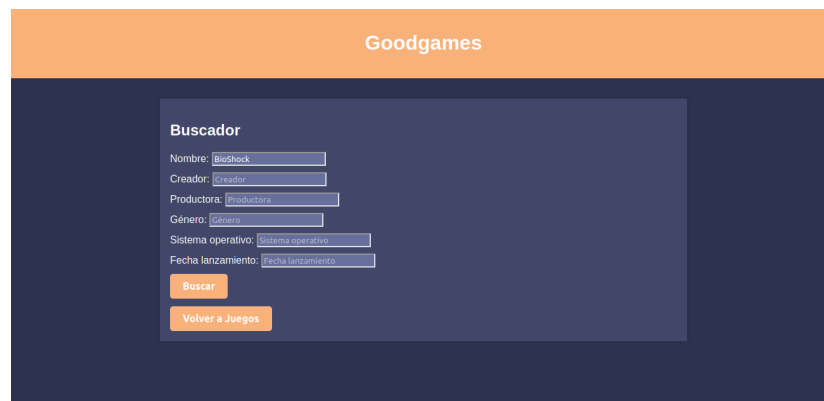


Figura 5: Buscar videojuegos

Catálogo de videojuegos	
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110	
Nombre	Género
BioShock	First-person shooter
BioShock 2	First-person shooter
BioShock Infinite	First-person shooter
BioShock Ejemplo	First-person shooter
Volver a Juegos	

Figura 6: Lista videojuegos buscados

- Cuando el usuario pulsa **Géneros más jugados**, si no ha iniciado sesión, se le redirigirá a la pantalla para iniciar sesión (fig. 1). Por otro lado, si sí ha iniciado sesión, será redirigido a una pantalla donde podrá ver un gráfico que muestra la frecuencia de géneros por videojuego (fig. 7).



Figura 7: Gráfico géneros más jugados

- Cuando el usuario pulsa **Géneros más jugados**, si no ha iniciado sesión, se le redirigirá a la pantalla para iniciar sesión (fig. 1). Por otro lado, si sí ha iniciado sesión, será redirigido a una pantalla donde podrá ver una lista de los 10 juegos más populares del último año según RAWG (fig. 8).



Figura 8: Lista juegos más populares

- El usuario puede pulsar **Mi Perfil**, y si no ha iniciado sesión, se le redirigirá a la pantalla para iniciar sesión (fig. 1). Por otro lado, si sí ha iniciado sesión, será redirigido a una pantalla donde podrá modificar sus datos de registro si así lo desea (fig. 9).



Figura 9: Ajustes de cuenta

- En todas las pestañas, pulsar el botón **Volver a Juegos** significará volver a la pantalla de catálogo de videojuegos (fig. 3), y pulsar el botón **Cerrar Sesión** significará volver a la pantalla de inicio de sesión (fig. 1), habiendo cerrado la sesión
- Por último, si el usuario pulsa el nombre de un videojuego, será redirigido a una ventana donde podrá ver información de este (fig. 10).



Figura 10: Datos del videojuego

Desde la vista de información del videojuego concreto el usuario puede hacer dos cosas:

- Si pulsa el botón **Modificar Videojuego**, el usuario será redirigido a una pantalla donde podrá modificar los datos del videojuego si así lo desea (fig. 11).

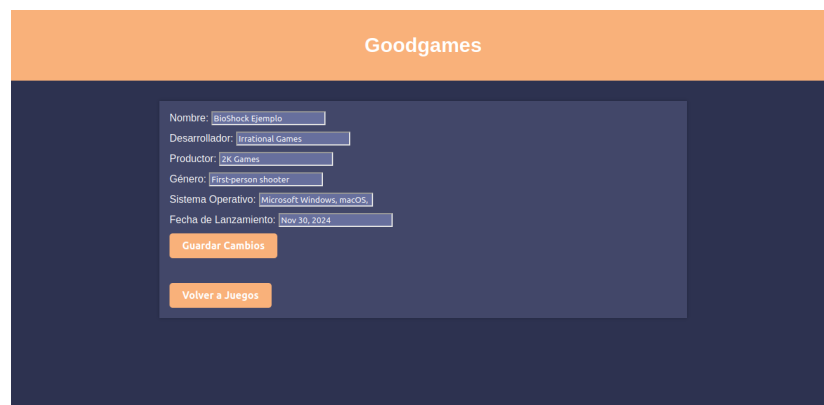


Figura 11: Editar datos del videojuego

- La otra opción que tiene es la de borrar el videojuego pulsando el botón de **Eliminar Videojuego**.

3. Docker Compose

Se ha creado un fichero YAML de Docker Compose llamado “docker-compose.yml”. Este archivo “docker-compose.yml” describe la configuración para un entorno Docker compuesto por varios servicios. Aquí hay un resumen del funcionamiento:

- Se utiliza la versión 3.7 de Docker Compose.
- Se definen varios servicios: db (MySQL 5.6), web (aplicación web con Apache), python (aplicación Python), phpmyadmin (interfaz web para MySQL), cAdvisor (monitorización de contenedores), node-exporter (exportador de métricas de nodos para Prometheus), prometheus (sistema de monitoreo) y api (otra aplicación python, utilizando la API de RAWG).
- Se define un volumen llamado *database* que se utiliza para persistir los datos de la base de datos MySQL, permitiendo que se mantengan los cambios realizados en el catálogo de videojuegos y en las cuentas creadas.

4. Tareas requeridas

Para el desarrollo de esta página web se han utilizado imágenes Docker que han permitido el correcto funcionamiento de la aplicación.

4.1. Servidor de BBDD

En la aplicación se ha utilizado una imagen de mysql como servidor de bases de datos. Los usuarios y los videojuegos son guardados en una base de datos, por lo que era necesario una imagen que permitiese gestionar la base de datos. Como se ha explicado antes, se ha definido un volumen que se utiliza para gestionar la persistencia de la base de datos. Se ha utilizado una imagen provista en Docker Hub.

Enlace: https://hub.docker.com/_/mysql

4.2. Servidor Web

La segunda imagen que se ha utilizado es la necesaria para poder tener un servidor web. Se puede acceder introduciendo en el navegador *localhost:8080*. Esta imagen ha sido creada mediante el archivo de nombre “Dockerfile”, el cual hace lo siguiente:

- **FROM php:7.2.2-apache:** Indica que este contenedor se construirá a partir de la imagen oficial de PHP con Apache en la versión 7.2.2. Esta imagen ya contiene un servidor Apache configurado para ejecutar aplicaciones PHP.

- **RUN docker-php-ext-install mysqli:** Ejecuta el comando `docker-php-ext-install` dentro del contenedor para instalar la extensión `mysqli` de PHP. Esto es necesario porque nuestra aplicación PHP utiliza MySQL, ya que `mysqli` es una extensión para la interacción con bases de datos MySQL.
- **COPY ./app /var/www/html:** Copia el contenido del directorio local `./app` al directorio de trabajo del contenedor `/var/www/html`. Este directorio es donde Apache buscará los archivos para la aplicación.
- **RUN a2enmod rewrite:** Habilita el módulo de reescritura de URL en Apache
- **RUN service apache2 restart:** Reinicia el servicio de Apache para que los cambios en la configuración surtan efecto.

Enlace: <https://hub.docker.com/r/portegar/as-web>

4.3. Python

La tercera imagen que se ha decidido utilizar es una imagen de Python Alpine. El gráfico que muestra los 5 géneros con más videojuegos es creado mediante un script de python gracias a las librerías *matplotlib* y *pandas*. Esta imagen ha sido creada mediante el archivo de nombre “`DockerfilePython`”, el cual hace lo siguiente:

- **FROM python:3-alpine3.15:** El contenedor se construirá a partir de la imagen oficial de Python 3 basada en Alpine Linux 3.15. Alpine es una distribución de Linux liviana y eficiente en espacio.
- **WORKDIR /app:** Establece el directorio de trabajo dentro del contenedor como `‘/app’`. Todos los comandos posteriores se ejecutarán en este directorio.
- **COPY ./app /app:** Copia el contenido del directorio local `‘./app’` al directorio `‘/app’` dentro del contenedor.
- **RUN pip install -r requirements.txt:** Ejecuta el comando `pip install` dentro del contenedor para instalar las dependencias especificadas en el archivo `‘requirements.txt’` (*pandas* y *matplotlib*).
- **CMD [“python”, “generar_grafico.py”]:** Define el comando por defecto que se ejecutará cuando se inicie el contenedor. En este caso, ejecuta el script Python `generar_grafico.py`. Este comando se ejecutará automáticamente al iniciar el contenedor, generando el gráfico que más tarde podrá verse en la web.

Enlace: <https://hub.docker.com/r/portegar/as-python>

5. Tareas adicionales realizadas

Además de las imágenes principales necesarias para el correcto funcionamiento de la web, se han incluido algunas imágenes y contenedores extra que añaden nuevas funcionalidades y facilitan información del estado de los distintos componentes como el estado de los contenedores, la base de datos...

5.1. API RAWG

Se ha hecho un contenedor adicional que también usa la imagen de Python Alpine. En este contenedor se obtendrán los datos que se muestran en la ventana de Juegos más populares del año (fig. 8). Esta imagen ha sido creada mediante el archivo de nombre “DockerfileApi”, el cual hace lo siguiente:

- **FROM python:3-alpine3.15:** El contenedor se construirá a partir de la imagen oficial de Python 3 basada en Alpine Linux 3.15. Alpine es una distribución de Linux liviana y eficiente en espacio.
- **WORKDIR /app:** Establece el directorio de trabajo dentro del contenedor como ‘/app’. Todos los comandos posteriores se ejecutarán en este directorio.
- **COPY ./app /app:** Copia el contenido del directorio local ‘./app’ al directorio ‘/app’ dentro del contenedor.
- **RUN pip install -r requirementsApi.txt:** Ejecuta el comando pip install dentro del contenedor para instalar las dependencias especificadas en el archivo ‘requirementsApi.txt’ (requests).
- **CMD [”python”, ”mas_vendidos.py”]:** Define el comando por defecto que se ejecutará cuando se inicie el contenedor. En este caso, ejecuta el script Python mas_vendidos.py. Este comando se ejecutará automáticamente al iniciar el contenedor, obteniendo los datos que más tarde se podrán ver en la web.

Enlace: <https://hub.docker.com/r/portegar/as-api>

5.2. phpMyAdmin

Esta imagen se ha implementado teniendo a los administradores del servicio en mente, ya que mejora la experiencia de administrar la base de datos. Se puede acceder introduciendo en el navegador *localhost:8181*. Se ha utilizado una imagen provista en Docker Hub.

Enlace: https://hub.docker.com/_/phpmyadmin

5.3. cAdvisor

Esta imagen se ha implementado teniendo a los administradores del servicio en mente, ya que provee información del estado de los contenedores, algo que puede ser útil a la hora de analizar el uso de recursos o los datos de rendimiento de los contenedores. Se puede acceder introduciendo en el navegador *localhost:8082*. Se ha utilizado una imagen provista en Docker Hub.

Enlace: <https://hub.docker.com/r/google/cadvisor>

5.4. Node Exporter

Esta imagen se ha implementado teniendo a los administradores del servicio en mente, ya que se encarga de exportar métricas de hardware y sistema operativo como uso de memoria, CPU, red, y disco. Se ha utilizado una imagen provista en Docker Hub.

Enlace: <https://hub.docker.com/r/prom/node-exporter>

5.5. Prometheus

Esta imagen se ha implementado teniendo a los administradores del servicio en mente, ya que es un servidor de monitoreo y sistema de alerta que recopila, almacena y procesa métricas de diversas fuentes, incluidos servicios, aplicaciones y, en este caso, Node Exporter. Prometheus proporciona una interfaz de consulta (PromQL) que permite realizar consultas complejas sobre las métricas almacenadas. Se puede acceder introduciendo en el navegador *localhost:9090*. Se ha utilizado una imagen provista en Docker Hub.

Enlace: <https://hub.docker.com/r/prom/prometheus>

6. Atribuciones

- Para el desarrollo de esta aplicación web ha sido necesario tener una fuente de datos de gran tamaño acerca de videojuegos, por lo que se ha empleado una base de datos obtenida en Kaggle.
Enlace: <https://www.kaggle.com/datasets/iamsouravbanerjee/computer-games-dataset?resource=download>
- Por otro lado, se ha utilizado la API de RAWG para obtener información sobre el número de jugadores de ciertos juegos.
Enlace: <https://rawg.io/apidocs>
- Por último, se ha empleado el asistente virtual ChatGPT durante el desarrollo de la aplicación. Su principal uso ha sido para solucionar errores e interpretar logs, y para obtener inspiración sobre qué funciones utilizar y qué información se podría conseguir utilizando la API de RAWG.