

Universidad ORT Uruguay

Facultad de Ingeniería

Obligatorio

Programación de redes

Entregado como requisito para la obtención del título de

Licenciatura en Sistemas

Guillermo Diotti - 285578

Lautaro Elosegui – 287788

Santiago Pucciarelli – 245481

Profesores:

Andrés Soria Pérez

Tomás Andrés Clavijo Vitale

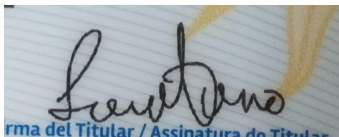
2024

<https://github.com/ArqSis-PDR-2024-2/obl-m6a-licenciatura-287788-285578-245481>

Declaración de autoría:

Nosotros, Lautaro Elosegui, Santiago Pucciarelli y Guillermo Diotti declaramos que el trabajo que se presenta en esta obra es de nuestra propia mano. Puedo asegurar que:

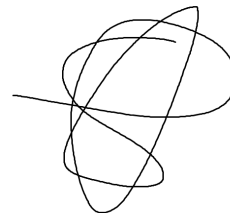
- La obra fue producida en su totalidad mientras realizamos el Obligatorio de Programación de redes;
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

A handwritten signature in blue ink on a white background. The signature is cursive and appears to read 'Lautaro Elosegui'. Below the signature, there is a small blue stamp that reads 'Firma del Titular / Assinatura do Titular'.

Lautaro Elosegui

A handwritten signature in black ink on a white background. The signature is cursive and appears to read 'Guillermo Diotti'.

Guillermo Diotti

A handwritten signature in black ink on a white background. The signature is cursive and appears to read 'Santiago Pucciarelli'.

Santiago Pucciarelli

Índice

Abstract	4
Alcance de la aplicación	5
Errores conocidos	5
Supuestos	5
Protocolo	6
Descripción de la arquitectura	7
Cliente	7
Servidor	7
Comunicación	8
Control de excepciones	8
Diagrama de componentes	9
Casos de prueba	10

Abstract

En este trabajo se presenta la documentación del proyecto obligatorio de programación de redes, primera entrega. En la misma se hará exposición tanto de los supuestos realizados, como de los errores encontrados, así como una explicación general de la estructura construida.

Alcance de la aplicación

El proyecto cumple concretamente con todos los requerimientos técnicos pautados en la letra del primer obligatorio. La aplicación cliente cuenta con autenticación, la opción de publicar un juego, adquirir un juego, modificar un juego, dar de baja un juego, buscar un juego, consultar información de un juego en específico y calificar un juego.

Por otra parte en el servidor se manejan los aspectos requeridos para el correcto funcionamiento del cliente (se manejan los pedidos de el listado en el párrafo anterior) así como el manejo de conexiones de usuarios, el borrado de las carátulas en el servidor y el cierre del servidor a través de la consola del mismo.

Finalmente, en los requisitos en común, la configuración se da de forma automática, permitiéndonos usar dispositivos diferentes para hacer la conexión cliente - servidor. Así mismo, las desconexiones y errores de un cliente no dan problemas al servidor ni a otros clientes conectados al mismo. Esto último gracias a que el servidor tiene una clase custom de excepciones la cual maneja las excepciones que podrían causar la caída del mismo.

Errores conocidos

A la hora de utilizar la función de descargar la imagen asociada a un juego buscado, existe la posibilidad de que el juego sea borrado (y por ende también la imagen) durante la descarga de la misma, ocasionando que el cliente se mantenga en un estado de espera o que en el servidor se ocasione un error al querer enviar una imagen que ya no existe. Este problema surge por la implementación que utilizamos en la cual primero se le pregunta al servidor si el juego cuenta con carátula y en caso de que el servidor responda positivamente, se comienza la descarga. El problema puede surgir en ese intermedio.

Para solucionar el problema se podría resolver bloqueando el acceso a ese juego desde el primer momento en el que se consulta y se mantenga bloqueado hasta luego de la descarga, evitando que este pueda borrarse durante el proceso de descarga, pero por el tiempo que restaba para la entrega y la poca probabilidad de que este caso ocurra, se decidió simplemente documentar y no invertir tiempo en arreglarlo, de igual manera está planteado arreglarlo para la siguiente.

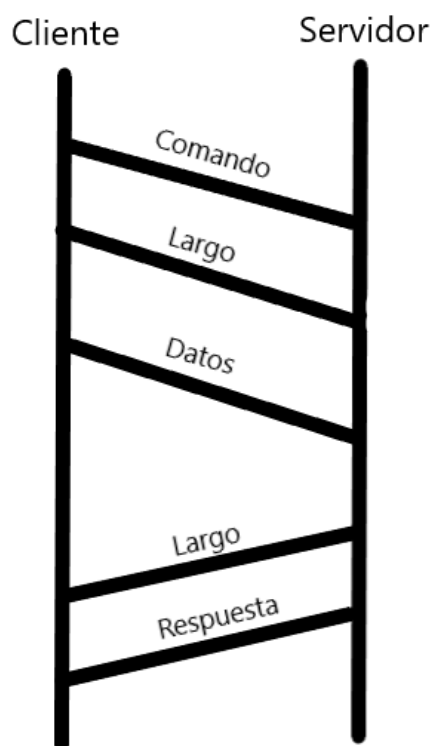
Supuestos

- Idioma del sistema: Se asume que el cliente es de habla hispana, la aplicación para el cliente y los mensajes del servidor están en español, mientras que el nombre de las clases y las variables en el código fuente se encuentran en inglés.
- Usuario único: Se asume que el nombre de usuario debe ser único en el sistema, por lo que no se permitirán múltiples cuentas con el mismo nombre de usuario
- Compra única por usuario: Se asume que un usuario solo puede adquirir un juego, por lo que múltiples compras de un mismo título por un mismo usuario no están permitidas.
- A la hora de modificar un juego, se modifican todos los aspectos de este, teniendo que colocar el mismo valor que poseía antes en caso de no querer modificar cierto campo.
- Acceso a calificaciones: Se asume que las calificaciones y comentarios de los juegos serán públicas, por lo que cualquier usuario podrá ver las reseñas y puntuaciones asociadas a un juego

- Reseñas únicas por usuario: Un mismo usuario no puede dejar una reseña en un juego más de una vez.
- Reseñas a juegos adquiridos: El usuario debe contar con el juego para poder dejar una reseña en el mismo.

Protocolo

Para la definición de este protocolo se utilizó como ejemplo el propuesto por la letra, aunque con ciertas modificaciones según lo que consideramos necesario e imprescindible para un correcto funcionamiento de la aplicación.



Campo	Comando	Largo	Datos
Valores	1-17 (Con posibilidad de expansión)	Entero	Variable
Tamaño	4 bytes	4 bytes	Largo

Este es el protocolo base que utilizamos para la comunicación del cliente con el servidor, pero no significa que se utilice en todas las peticiones, existen casos en donde no es necesario el envío de datos al servidor (al listar juegos por ejemplo), en estos casos solamente se realiza el envío del comando con el cual el servidor define el camino de acción.

El comando hace referencia a la acción que el cliente está intentando acceder, funciona para guiar al servidor sobre lo que el cliente desea hacer y traza la ruta para que (en caso de ser requerido), el servidor reciba los datos, los procesa y con ellos realiza las operaciones correspondientes.

También es debido aclarar que es el cliente quien hace uso de este protocolo para el envío de mensajes, ya que el servidor prescinde de la utilización de comandos al no necesitar que el cliente se adapte a la respuesta del servidor.

Descripción de la arquitectura

Cliente

La solución se separó en 3 proyectos, Cliente, Servidor y Comunicación. Dentro de cada proyecto además, se decidió separar algunos aspectos del código en sus propias clases. Además de la clase Program.cs y ClientConfig.cs, para el cliente se crearon 4 clases particulares. Empezando por el Program dentro del Cliente, además del main, tenemos una función TryConnectToServer que se encarga de la conexión al servidor y RunMainLoop que se encarga de mostrar el menú principal luego de cada acción en el menú y de desconectar al cliente en caso de que haga logout o de que cierre la aplicación. Por otro lado ClientConfig.cs tiene aspectos para la configuración de la conexión del cliente al servidor.

La clase Utilities se encuentra en la raíz del proyecto Cliente y tiene diferentes funcionalidades que permiten evaluar los ingresos del cliente por consola. Algunos ejemplos de funciones de esta clase pueden ser la función ReadNonEmptyInput(), la cual revisa que el input del cliente no sea un espacio vacío. ShowGameCategories() y ShowGamePlatforms(), las cuales listan las categorías de juego con las que contamos en el sistema y las plataformas en las que se pueden publicar los juegos respectivamente. Y por último, CheckIntegerValue(int min, int max), la cual revisa que el valor ingresado sea un número entre 2 valores, esta última se utiliza para la verificación del año de publicación entre otros usos.

Luego tenemos 3 clases dentro de una carpeta llamada Menu la cual contiene a la clase GameManager.cs, MenuManager.cs y SessionManager.cs. La primera se encarga de los aspectos pertinentes a los juegos (creación, modificación, eliminación, etc), la siguiente se encarga del manejo del menú principal con un switch y la última se encarga de manejar la autenticación y sesión del cliente conectado al servidor.

Servidor

Por otra parte tenemos el proyecto Servidor que tiene Program.cs, ServerConfig.cs y Utilities.cs en la raíz del proyecto. Program contiene el main, HandleServer, una función que maneja el comando exit en la consola del servidor y HandleClient una función que maneja las acciones del cliente en el servidor (como por ejemplo la creación de un juego en el servidor). ServerConfig es similar a ClientConfig, albergando la configuración del servidor para poder establecer la conexión con el cliente. Utilities tiene algunas funciones para facilitar la comprensión de los comandos ingresados por el usuario. Se encarga del uso de enums para el manejo de la lista de juegos y plataformas además de SendResponse que envía los mensajes al cliente.

Dentro de Servidor hay 3 carpetas llamadas Collections, Exceptions y Logics, las cuales cuentan con distintas librerías de clases para una mejor organización de la solución.

En la carpeta Collection tenemos 3 clases, GameCollection.cs, SocketCollection.cs y UserCollection.cs. En estas Collections es donde se presenta el manejo del acceso a los recursos. En base a lo discutido en clase se decidió rápidamente en utilizar locks para controlar el acceso al recurso ya que nos pareció lo más adecuado. SocketCollection y UserCollection tienen el mismo funcionamiento que GameCollection pero para los sockets y el user respectivamente.

Dentro de la carpeta Exceptions está la clase ServerExceptions que maneja las excepciones del servidor.

En la última carpeta, Logics, se encuentra la lógica de Game, Review y User, cada una dentro de una carpeta con su nombre seguido de Logic.

Dentro de GameLogic tenemos 4 clases, Game.cs, GameGenre.cs, GameLogic.cs y GamePlatform.cs, que serían las clases que cuentan con cierta responsabilidad sobre cómo funcionan los juegos.

Comunicación

En este proyecto se encuentran clases relacionadas al envío de mensajes entre los clientes y el servidor, además de administrar la configuración que permite una conexión más dinámica entre el cliente y el servidor.

La clase NetworkDataHelper es la encargada del envío y recepción de mensajes binarios de ambas partes, contando con los métodos Send y Receive que aseguran la integridad de la información al momento de trabajar con ella. En el caso de las clases FileCommsHandler, ConversionHandler, FileStreamHandler, FileHandler y Protocol también se utilizan para el envío y recepción de información, pero más específicamente de archivos o imágenes, siendo la clase principal FileCommsHandler, quien utiliza las demás para realizar dichas acciones.

Por último se cuenta con la clase Actions que define los códigos utilizados en el protocolo para guiar las peticiones al servidor, siendo algunos “Register”, “Login”, AcquireGame”, etc.

Decisiones de diseño

Dentro de las distintas decisiones tomadas en el transcurso de este obligatorio se encuentran, en primera mano, la decisión de qué categorías y plataformas incluir en el sistema para la creación de un juego. Por el lado de las categorías elegidas, ellas fueron Acción, Aventura, Combate, Puzzle, Carreras, Tiroteo, Simulación, Deportes, Estrategia. Por otra parte, las distintas plataformas a optar son Windows, MacOS, Android y Linux.

Uno de los requerimientos del sistema era el de la búsqueda de juegos, específicamente buscarlos en base a criterios definidos. Nuestra solución permite filtrar los juegos tanto por plataforma como por categoría, de tal manera que es posible visualizar fácilmente, por ejemplo los juegos con categoría de Acción.

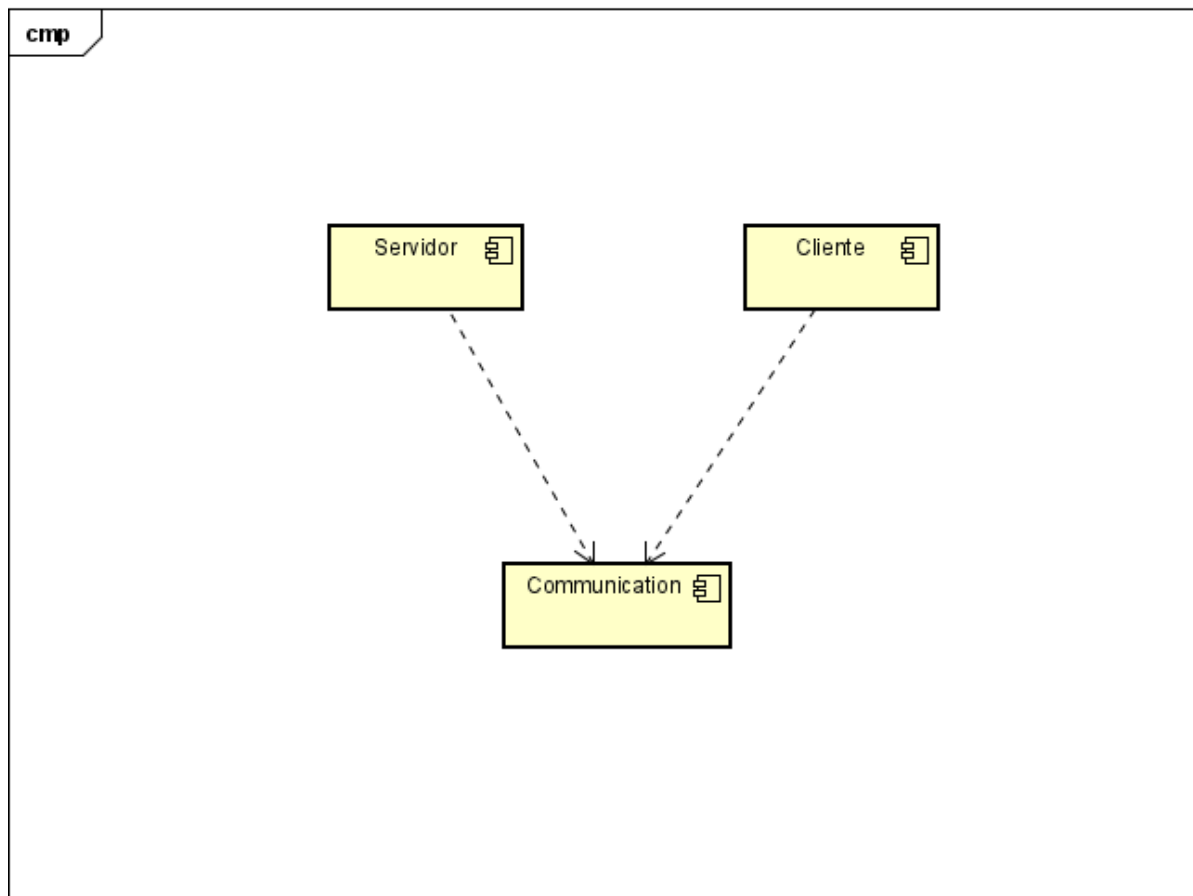
Otra decisión fue que al momento de crear el juego, automáticamente se asocia como creador al usuario conectado, ya que no consideramos que sea lógico que el usuario conectado decida quién es el creador, ya que es el mismo quien lo crea.

Además, el año de publicación de un juego está limitado a ser entre 1952 (fecha de lanzamiento del primer videojuego) y 2 años más del año actual (en nuestro caso 2026).

Control de excepciones

Como se nombró anteriormente, se cuenta con una clase ServerException que define una estructura para lanzar excepciones personalizadas cuando se intenta realizar una acción no posible, estas se lanzan con el mensaje informativo correspondiente en las diferentes colecciones cuando se cumplen ciertas condiciones. Luego en las clases UserLogic, ReviewLogic y GameLogic se utiliza la estructura try y catch para atrapar esas excepciones y retornar al cliente el mensaje de error que corresponda.

Diagrama de componentes



Finalmente, agregamos el diagrama de componentes correspondiente a la aplicación. El mismo es muy simple, al estar basado en los .dlls respectivos a cada proyecto en su release, nos damos cuenta que Cliente y Servidor dependen de Communication ya que ese .dll se encuentra en ambas carpetas de release, por lo que ambas están utilizando este componente para su funcionamiento final. Esto tiene sentido al fijarnos que, al tener todos los métodos para el envío estable de mensajes entre un emisor y un receptor, el proyecto Communication juega un papel clave tanto para el cliente como para el servidor.

Casos de prueba

A continuación se exponen distintos casos (no todos) que pueden ocurrir cuando se intenta llevar a cabo una acción no válida o incorrecta. Se mostrarán los distintos mensajes que se pueden mostrar dependiendo de cómo lleves a cabo la consulta.

Campo vacío

La mayoría de campos (si no es que todos) cuentan con una validación de que el contenido del campo no puede ser vacío, por lo que si intentas hacerlo se mostrará este mensaje

```
--- Registro ---  
Nombre de usuario:  
El valor no puede estar vacío ni contener '#'.  
Intente nuevamente: 
```

Campo con “#”

Debido a la forma que empaquetamos los datos para mandárselos al servidor, en el cual utilizamos un “#” como separador y luego un “split” para desempaquetar los, decidimos eliminar la posibilidad de que este símbolo pueda introducirse como valor a un campo, ya que el método que lo valida es el mismo que el de los campos vacíos, en el mensaje es el mismo.

```
--- Registro ---  
Nombre de usuario:  
El valor no puede estar vacío ni contener '#'.  
Intente nuevamente: 
```

Seleccionar una opción incorrecta

Ya que decidimos definir valores concretos para distintos campos (como puede ser el género y plataforma), es requerido que al intentar ingresar un valor para estos campos, el mismo esté dentro de los parámetros especificados, en el caso de que se intente seleccionar uno incorrecto, se mostrará el siguiente mensaje.

```
--- Publicar Juego ---  
Nombre del juego: Warhammer  
Género del juego:  
1- Acción  
2- Aventura  
3- Combate  
4- Puzzle  
5- Carreras  
6- Tiroteo  
7- Simulación  
8- Deportes  
9- Estrategia  
10  
Entrada no válida.  
Introduzca un valor correcto: 
```

Usuario o juego repetido

En el caso de que intentes crear un objeto usuario o juego con la misma llave primaria de un objeto del mismo tipo ya existente (nombre para usuario y título para juego), al terminar de colocar rellenar todos los campos requeridos, se mostrará un mensaje informando que la acción no es posible debido a que no se aceptan repetidos.

```
--- Publicar Juego ---
Nombre del juego: Warframe
Género del juego:
1- Acción
2- Aventura
3- Combate
4- Puzzle
5- Carreras
6- Tiroteo
7- Simulación
8- Deportes
9- Estrategia
1
Año de lanzamiento del juego: 2014
Publicador del juego: UBISOFT
Plataforma del juego:
1- Windows
2- Linux
3- MacOS
4- Android
1
Unidades del juego: 10
Servidor:

El juego ingresado ya existe.
```

```
--- Registro ---
Nombre de usuario: Law
Contraseña: 123
Servidor:

El nombre de usuario ya existe.
```

Adquirir un juego sin unidades

Uno de los atributos que poseen los juegos, es la cantidad de unidades que pueden adquirirse del mismo, por ende cuando un usuario lo adquiera, las unidades disminuyen en uno. En el caso de que un usuario intente adquirir un juego que no posea unidades, se mostrará el siguiente mensaje

```
Nombre del juego a adquirir: W
Servidor:

No hay unidades disponibles.
```

Adquirir un juego que ya posees

Luego se encuentra el caso de que el juego cuente con unidades, pero tu anteriormente ya lo hayas adquirido, por ende no se te permitirá volver a adquirirlo

```
--- Adquirir Juego ---  
  
--- Listar Juegos ---  
Servidor:  
  
1 - Warframe  
  
Nombre del juego a adquirir: Warframe  
Servidor:  
  
El usuario ya posee el juego.
```

Modificar un juego que no creaste

Pasando ahora al requerimiento de modificar un juego, está estipulado que el responsable de hacer esto es el creador de dicho juego, por lo que si otro usuario intenta modificarlo, resultaría en un mensaje de error

```
--- Modificar Juego ---  
Nombre del juego: Warframe  
Nuevo nombre del juego: Warframe  
Nuevo género del juego:  
1- Acción  
2- Aventura  
3- Combate  
4- Puzzle  
5- Carreras  
6- Tiroteo  
7- Simulación  
8- Deportes  
9- Estrategia  
1  
Nuevo año de lanzamiento del juego: 2021  
Nuevo publicador del juego: Bethesda  
Nueva plataforma del juego:  
1- Windows  
2- Linux  
3- MacOS  
4- Android  
1  
Nueva cantidad de Unidades del juego: 10  
Servidor:  
  
No tienes permisos para modificar este juego.
```

Eliminar un juego

Lo mismo que ocurre para modificar se presenta aquí, si un usuario intenta eliminar un juego que no ha creado, se mostrará el siguiente error.

```
Nombre del juego a eliminar: Warframe
Servidor:

No tienes permisos para eliminar este juego.
```

Calificar un juego que no posees

Para poder calificar un juego y dejar una reseña, es requerido que el usuario cuente con el juego que intenta calificar, por lo que si esto no se cumple, se le notificará al usuario que la acción no es válida de la siguiente forma.

```
Nombre del juego a calificar: Warframe
Calificación del juego: 10
Reseña del juego: Buen juego
Servidor:

No puedes calificar un juego que no has adquirido.
```

Agregar/actualizar una carátula con una imagen que no existe

Cuando se publica o modifica un juego, se muestra la opción para colocar una carátula al mismo, si el usuario decide hacerlo pero introduce una dirección en la que no se encuentra ninguna imagen, desde el cliente se lanzará un mensaje indicándolo.

```
¿Desea subir una portada para dicho juego? (S/N)
S
Ruta completa de la imagen: ruta incorrecta
Imagen no encontrada.
```

Descargar una imagen que no existe

```
Nombre del juego a buscar: Warframe
Servidor:

Titulo: Warframe Genero: Action Año de Lanzamiento: 2022 Plataforma: Linux Publicador: 2 Unidades: 0

1- Descargar imagen
2- Ver reseñas
3- Volver
1

--- Descargando imagen ---

El juego no cuenta con imagen
```

Reseñar dos veces el mismo juego

Un cliente solo puede reseñar un juego una vez, no se permiten múltiples reseñas del mismo usuario al mismo juego

```
--- Calificar Juego ---  
  
--- Listar Juegos ---  
Servidor:  
  
1 - W  
2 - Ark  
3 - Warframe  
  
Nombre del juego a calificar: Warframe  
Calificación del juego: 8  
Reseña del juego: Bueno  
Servidor:  
  
El usuario ya cuenta con una reseña para este juego.
```

A continuación se ejemplifican algunos de los mensajes que se muestran cuando se realiza correctamente una acción determinada.

Registrar usuario

```
--- Registro ---  
Nombre de usuario: Lawta  
Contraseña: 123  
Servidor:  
  
Registro exitoso.
```

Iniciar sesión

```
--- Iniciar Sesión ---  
Nombre de usuario: Lawta  
Contraseña: 123  
Servidor:  
  
Inicio de sesion exitoso.
```




Publicar juego

```
--- Publicar Juego ---
Nombre del juego: W
Género del juego:
1- Acción
2- Aventura
3- Combate
4- Puzzle
5- Carreras
6- Tiroteo
7- Simulación
8- Deportes
9- Estrategia
7- Simulación
8- Deportes
9- Estrategia
1
Año de lanzamiento del juego: 2022
7- Simulación
8- Deportes
9- Estrategia
1
Año de lanzamiento del juego: 2022
Publicador del juego: EA
Plataforma del juego:
1- Windows
2- Linux
3- MacOS
4- Android
1
Unidades del juego: 4
Servidor:
Registro exitoso.
```

Publicar caratula

```
¿Desea subir una portada para dicho juego? (S/N)
S
Ruta completa de la imagen: C:\Users\User\Downloads\d.jpg
Servidor:

Caratula asignada exitosamente
```

ProgDeRedes > Servidor > bin > Debug > net8.0					Buscar en net8.0
Nombre	Fecha de modificación	Tipo	Tamaño		
 runtimes	15/09/2024 10:17	Carpeta de archivos			
 a29a8f44-c4a5-48fa-ace5-0a4fe1608b5e.jpg	30/09/2024 8:46	Archivo JPG	50	KB	
 Communication.dll	30/09/2024 8:19	Extensión de la aplica...	9	KB	

Adquirir juego

```
--- Adquirir Juego ---

--- Listar Juegos ---
Servidor:

1 - Warframe

Nombre del juego a adquirir: Warframe
Servidor:

Juego adquirido.
```

Eliminar juego

```
--- Eliminar Juego ---

--- Listar Juegos ---
Servidor:

1 - W
2 - Ark
3 - Warframe

Nombre del juego a eliminar: Ark
Servidor:

Juego eliminado.
```


Modificar juego

```
--- Modificar Juego ---
Nombre del juego: Warframe
Nuevo nombre del juego: Warframe2
Nuevo género del juego:
1- Acción
2- Aventura
3- Combate
4- Puzzle
5- Carreras
6- Tiroteo
7- Simulación
8- Deportes
9- Estrategia
5
Nuevo año de lanzamiento del juego: 2012
Nuevo publicador del juego: AE
Nueva plataforma del juego:
1- Windows
2- Linux
3- MacOS
4- Android
4
Nueva cantidad de Unidades del juego: 40
Servidor:

Juego modificado.
```

Modificar caratula

```
¿Desea modificar la portada para dicho juego? (S/N)
S
Ruta completa de la imagen: C:\Users\User\Downloads\d2.jpg
Servidor:

Caratula modificada exitosamente
```

Buscar juego

```
--- Listar Información de un juego ---

--- Listar Juegos ---
Servidor:

1 - W
2 - Ark
3 - Warframe

Nombre del juego a buscar: Warframe
Servidor:

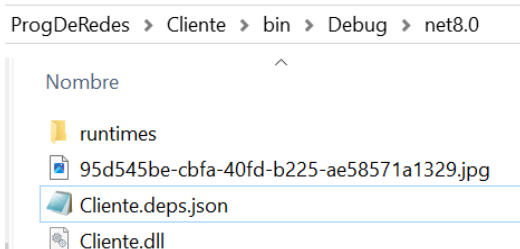
Titulo: Warframe Genero: Action Año de Lanzamiento: 2022 Plataforma: Linux Publicador: 2 Unidades: 0

1- Descargar imagen
2- Ver reseñas
3- Volver
```

Descargar imagen

```
1- Descargar imagen
2- Ver reseñas
3- Volver
1

--- Descargando imagen ---
Imagen descargada
```



Ver reseñas

```
--- Mostrando reseñas ---
Servidor:

Reseñas de Warframe

Juego: Warframe Usuario: Law Puntuacion: 10 Texto: Ta bueno

Juego: Warframe Usuario: Law1 Puntuacion: 10 Texto: Ta bueno
```

Publicar reseña

```
--- Calificar Juego ---
```

```
--- Listar Juegos ---
```

```
Servidor:
```

```
1 - W
```

```
2 - Ark
```

```
3 - Warframe
```

```
Nombre del juego a calificar: Warframe
```

```
Calificación del juego: 10
```

```
Reseña del juego: Ta bueno
```

```
Servidor:
```

```
Reseña enviada.
```