

APLICACIÓN WEB ROBUSTIANO GAMES

DESARROLLO DE APLICACIONES WEB 2º

AUTOR: UNAI SUÁREZ CARBALLO



CPIFP LOS ENLACES
TUTOR: LUIS MIGUEL MORILLAS

28 DE MAYO DE 2022



TITULO DOCUMENTO

Índice

INTRODUCCIÓN.....	4
1. Documento de propuesta de proyecto	5
1.1. Título del proyecto	5
1.2. Título corto del proyecto.....	5
1.3. Contexto breve	5
1.4. Objetivo breve	5
2. Documento descripción del proyecto	5
2.1. Contexto del proyecto.....	5
2.1.1. Ámbito y entorno	5
2.1.2. Destinatarios	5
2.2. Objetivo del proyecto	6
2.3. Objective of the project	6
3. Documento de Acuerdo del proyecto	7
3.1. . Requisitos funcionales y no funcionales / Historias de usuario.....	7
3.1.1. Requisitos funcionales.....	8
3.1.2. Requisitos no funcionales	15
3.2. Tareas.....	17
3.3. Metodología a seguir para la realización del proyecto	18
3.4. Planificación temporal de tareas	19
3.5. Presupuesto (gastos, ingresos, beneficios).....	19
3.5.1. Costes fijos.....	19
3.5.2. Costes variables.....	19
3.5.3. Costes Totales.....	19
3.5.4. Presupuesto	19
3.5.5. Contrato	19
4. Documento de análisis y diseño	20
4.1. Modelo de datos. Análisis y diseño de base de datos.....	20
4.1.1. Diagrama E/R.....	21
4.1.2. Esquema Relacional Estándar	22
4.1.3. Grafo relacional.....	23
4.1.4. Normalización	24
4.1.5. Dominios de atributos.....	25
4.1.6. Scripts de la Base de Datos.....	26
4.2. Análisis y diseño del sistema funcional (Diagramas que procedan como el de casos de uso, de clases, de secuencia, de actividad, de estados, de flujo...)	27
4.2.1. Lista de actores.....	27
4.2.2. Caso de uso Usuarios de la aplicación	28
4.2.3. Caso de uso Registro de usuario	29
4.2.4. Caso de uso Mensajes.....	30
4.2.5. Caso de uso compra videojuegos	31
4.2.6. Caso de uso Comentarios.....	32
4.2.7. Caso de uso Valoración.....	33
4.2.8. Caso de uso amigos	34
4.2.9. Diagrama de despliegue Login usuarios.....	35



4.2.10.	Diagrama de despliegue Registro de usuario	36
4.2.11.	Diagrama de despliegue Enviar mensaje	37
4.3.	Análisis y diseño de la interfaz de usuarios. Mockups.....	38
4.4.	Diseño de la arquitectura de la aplicación	45
4.4.1.	Tecnologías/Herramientas usadas y descripción de las mismas.	47
4.4.2.	Arquitectura de componentes de la aplicación	48
5.	Documento de implementación e implantación del sistema	49
5.1.	Pruebas.....	49
5.1.1.	Pruebas unitarias	49
5.1.2.	Pruebas funcionales	49
5.2.	Instalación y configuración.....	52
5.3.	Manual de usuario	53
6.	Documento de cierre	59
6.1.	Resultados Obtenidos	59
6.2.	Diario de bitácora	60
7.	Webgrafía.....	61
8.	Anexos.....	61
	ANEXO I: CONTRATO.....	61
	ANEXO II: SCRIPT BASE DE DATOS.....	64



INTRODUCCIÓN

El proyecto se basará en un programa tipo Steam el cual permita a un usuario crear una cuenta y poder comprar juegos desde la aplicación.

En la base de datos por una parte estaría guardada la información de todos los videojuegos (precio, nombre, descripción y alguna imagen) y por otra parte estaría la información de cada usuario (cuenta de correo, contraseña, nombre, dinero en cuenta y los juegos que tiene agregados a su cuenta).

Cuando un usuario nuevo se creará una nueva cuenta no tendría ningún juego asociado a esta por lo cual para poder tener alguno debería ingresar dinero, para ello se necesitará un código el cual al introducirlo agregue cierta cantidad a la cuenta, con este dinero podrá comprar videojuegos, pero claro está, se le descontará lo gastado.

Hasta que el usuario no compre el videojuego no se le permitirá descargarlo, una vez comprado se guardará en su cuenta para siempre.

En el programa cuando entras por primera vez te pedirá una cuenta de usuario, si no tienes se podrá crear una nueva, una vez creada te llevará directamente a la página de inicio en la cual hay diferentes listas como los videojuegos más jugados, en oferta, próximos lanzamientos, etc.

También habrá un filtro el cual permita ver juegos de acción, aventura, fantasía, etc.

En el menú principal aparte del inicio podrás ir tu cuenta personal en la cual puedes ver los juegos que has comprado y que puedes descargar, junto a esto puedes observar las horas jugadas a cada juego, los trofeos conseguidos, comentarios y ayudas de los demás jugadores.

Por último estará la zona de ajuste de usuario en la cual puedes cambiar la información de la cuenta como el nombre, la contraseña, el correo e incluso eliminar la cuenta para siempre.

Área de desarrolladores: Publicador de juegos donde puedes gestionar ventas del juego (si es de pago), ver cuántas descargas tiene, control de errores, actualizaciones, mods o DLCs, etc.



1. Documento de propuesta de proyecto

1.1. Título del proyecto

Robustiano Games tienda de videojuegos digitales

1.2. Título corto del proyecto

Robustiano Games

1.3. Contexto breve

Robustiano Games es una aplicación web que pretende facilitar la venta y compra de videojuegos de manera online y digital, de manera que sea sencillo comprarlos, almacenarlos en una biblioteca online una vez comprados, así como pretende tener una parte social la cual los usuarios puedan hablar entre sí, publicar sus comentarios y reseñas y valorar los videojuegos.

1.4. Objetivo breve

Llegar a desarrollar las funciones mínimas para que la aplicación web sea válida.

2. Documento descripción del proyecto

2.1. Contexto del proyecto

Este proyecto nace de mi pasión por los videojuegos, busco crear una aplicación web sencilla, familiar y amigable para los usuarios, en la que puedan confiar y estar seguros mientras realizan sus compras o hablan con sus amigos

2.1.1. Ámbito y entorno

La finalidad de este proyecto es conseguir una meta propia, siempre me ha interesado el mundo de los videojuegos y cuando tenía que pensar una idea para un proyecto, se me ocurrió, realizar una tienda online la cual permita a cualquier usuario comprar videojuegos de manera digital e incluso venderlos, con esto también dejo claro que mi programa va dirigido a cualquier usuario.

2.1.2. Destinatarios

Los destinatarios a quienes va dirigido mi proyecto son a cualquier usuario, ya sea un desarrollador de videojuegos o un simple aficionado a los videojuegos.



2.2. Objetivo del proyecto

El objetivo de este proyecto es desarrollar una aplicación web la cual permita a cualquier usuario comprar videojuegos de manera digital, así como también permitirles añadir amigos, seguir a otros jugadores para así poder crear una comunidad en la cual se hable de ciertos temas vinculados a los videojuegos.

Además de que cualquier usuario se puede registrar como desarrollador para así poder publicar sus videojuegos y generar ganancias con estos.

Los usuarios que sean amigos podrán enviarse mensajes de texto entre ellos.

Mi intención final es que los usuarios se sientan cómodos y seguros con la aplicación de manera que poco a poco la comunidad crezca.

2.3. Objective of the project

The objective of this project is to develop a web application which allows any user to buy video games digitally, as well as allowing them to add friends, follow other players in order to create a community in which certain topics related to games are discussed. video game.

In addition to that any user can register as a developer in order to publish their video games and generate profits with them.

Users who are friends will be able to send text messages to each other.

My final intention is that users feel comfortable and safe with the application so that little by little the community grows.



3. Documento de Acuerdo del proyecto

3.1.. Requisitos funcionales y no funcionales / Historias de usuario

En este punto detallaré los requisitos funcionales y no funcionales que tendrá mi aplicación web “Robustiano Games”.

Para el desarrollo de este proyecto voy a clasificar los requisitos de la siguiente manera:

- Los **requisitos funcionales** serán aquellos que determinen el comportamiento de la aplicación.
- Los **requisitos no funcionales** serán aquellos que definan las características de la aplicación, ya se la usabilidad, interfaz, etc.

Usaré el siguiente formato de tabla.

Requisito: RF-X “Nombre”			
Descripción	“breve descripción”		
Tipo	“tipo de requisito”	Prioridad	“prioridad: alta, media o baja”
Versión Alta	10/04/2022	Fecha Alta	10/05/2022

**3.1.1. Requisitos funcionales**

Requisito: RF1 Registro de Usuarios			
Descripción	<p>Permitir a los usuarios registrarse para poder acceder a la aplicación web.</p> <p>Se necesitará rellenar un formulario con los siguientes campos:</p> <p>Nombre: será el nombre de usuario, cada uno tendrá el suyo propio y no se podrán repetir.</p> <p>Email: correo electrónico que junto al nombre servirá para iniciar sesión, también será a donde le lleguen las notificaciones.</p> <p>Contraseña: será necesaria para poder iniciar sesión.</p> <p>Desarrollador: esta opción podrá ser marcada para que el usuario sea considerado desarrollador, esto le permitirá publicar videojuegos desarrollados por él., cuando otro usuario compre uno de sus videojuegos el dinero se le añadirá a su cuenta.</p>		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF2 Identificación			
Descripción	Permitir iniciar sesión mediante un usuario y una contraseña anteriormente registrados.		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF3 Cerrar sesión			
Descripción	Los usuarios podrán cerrar sesión. Una vez cerrada serán redirigidos a la página de login		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**Requisito: RF4 Eliminar cuenta**

Descripción	Los usuarios eliminar su cuenta en caso de que no quieran seguir en la aplicación		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF5 Página de inicio

Descripción	La página dispondrá de 3 partes. <ul style="list-style-type: none">• Cabecera• Menú superior• Contenido central		
Tipo	Funcional	Prioridad	media
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF6 Cabecera

Descripción	Contendrá el logo de la empresa junto a las opciones principales de la pagina, ya pueden ser ir al inicio, iniciar sesión o registrarse, cuando el usuario esté registrado se verán nuevas opciones como ver amigos o biblioteca de videojuegos.		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF7 Menú superior

Descripción	Un menú justo debajo de la cabecera donde estarán las opciones y filtros de búsqueda para el contenido central		
Tipo	Funcional	Prioridad	media
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**Requisito: RF8 Añadir saldo**

Descripción	Una opción para los usuarios registrados la cual les permitirá añadir cierta cantidad de dinero para así poder realizar compras. Este podrá ser añadido con un código que proveerá el administrador.		
Tipo	Funcional	Prioridad	media
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF9 Añadir amigos

Descripción	El usuario puede buscar a otros usuarios y añadirlos a una lista de amigos. Para ello el otro usuario tendrá que aceptar la solicitud de amistad.		
Tipo	Funcional	Prioridad	media
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF10 Enviar mensajes

Descripción	El usuario puede enviar un mensaje a los usuarios añadidos como amigos.		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF11 Leer mensajes

Descripción	El usuario puede leer los mensajes recibidos de otros usuarios.		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**Requisito: RF12 Comprar videojuegos**

Descripción	El usuario puede comprar videojuegos mediante el saldo añadido anteriormente, una vez comprados serán añadidos a su biblioteca.		
Tipo	Funcional	Prioridad	alta
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF13 Publicar videojuegos

Descripción	<p>El usuario registrado como desarrollador podrá publicar videojuegos, para ello deberá rellenar un formulario con la información de este:</p> <p>Nombre: este será único y servirá para poder buscar este.</p> <p>Precio: en un inicio aparecerá como 0 y si el usuario no lo cambia eso significará que el juego saldrá como gratuito.</p> <p>Descripción: aquí deberá poner una breve explicación sobre el videojuego.</p> <p>Categoría: para poder así filtrarlo después será necesario agregarle una o más categorías.</p>		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF14 Ver videojuegos publicados

Descripción	Los desarrolladores podrán ver los videojuegos que han publicado, además de ver los análisis de estos, donde se podrá ver, los jugadores que lo han comprado, el dinero generado, las valoraciones y los comentarios de este.		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**Requisito: RF15 Eliminar videojuegos**

Descripción	Los desarrolladores podrán eliminar los videojuegos que han publicado.		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF16 Subir DLC(Downloadable Content)

Descripción	<p>Los desarrolladores podrán publicar DLC para sus videojuegos, para poder hacerlos públicos, se deberá rellenar un formulario:</p> <p>Videojuego: se deberá elegir al videojuego al que pertenecen.</p> <p>Nombre: cada contenido descargable necesita un nombre para ser reconocido.</p> <p>Descripción: al igual que con el videojuego cada DLC deberá contar con una breve explicación sobre este.</p> <p>Precio: en un inicio aparecerá como 0 y si el usuario no lo cambia eso significará que el juego saldrá como gratuito.</p>		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF17 Publicar Comentarios

Descripción	Los usuarios pueden comentar y dar su opinión en cada videojuego		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**Requisito: RF18 Valorar Videojuegos****Descripción**

Los usuarios pueden valorar con hasta 5 estrellas los videojuegos que compren.

El campo a rellenar es:

Valoración: del 1 al 5 el usuario podrá valorar un videojuego

Tipo

Funcional

Prioridad

baja

Versión Alta

23/05/2022

Fecha Alta

23/05/2022

Requisito: RF19 Publicar Comentarios**Descripción**

Los usuarios pueden comentar y dar su opinión en cada videojuego

El campo a rellenar es:

Comentario: texto del comentario

Tipo

Funcional

Prioridad

baja

Versión Alta

23/05/2022

Fecha Alta

23/05/2022

Requisito: RF20 Publicar Valoración**Descripción**

Los usuarios pueden poner una valoración a un videojuego de hasta 5 puntos

Tipo

Funcional

Prioridad

baja

Versión Alta

23/05/2022

Fecha Alta

23/05/2022

**Requisito: RF21 Ver biblioteca**

Descripción	Los usuarios tendrán una opción para ver su biblioteca en la que estarán todos los videojuegos que han comprado, podrán entrar en cada uno de ellos y realizar distintas acciones, como valorarlo o comentarlo, devolverlo(para ello será necesario que no haya pasado más de una semana desde que se compró).		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RF22 Editar Videojuego

Descripción	El desarrollador podrá editar sus videojuegos, cambiándole diferentes aspectos, ya sea, el nombre, descripción, precio, etc.		
Tipo	Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

**3.1.2. Requisitos no funcionales**

Requisito: RFN1 Compatibilidad con navegadores			
Descripción	La aplicación debe ser compatible con las últimas versiones de los principales navegadores.		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RFN2 Diseño Responsive			
Descripción	La interfaz debe ser capaz de adaptarse a cada dispositivo.		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RFN3 Base de datos relacional			
Descripción	Todo el contenido será guardado en una base de datos relacional llamada robustiano		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RFN4 Requisitos LOPD			
Descripción	La aplicación web tiene que contener una interfaz amigable, intuitiva para el usuario, de manera que sea fácil de manejar.		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022



Requisito: RFN5 Diseño

Descripción	La aplicación web debe contener una interfaz sencilla e intuitiva.		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022

Requisito: RFN6 Rendimiento

Descripción	Cuando se acceda a la base de datos no debe de suponer una carga para la aplicación, por lo que el rendimiento debe de ser óptimo.		
Tipo	No Funcional	Prioridad	baja
Versión Alta	23/05/2022	Fecha Alta	23/05/2022



3.2. Tareas

Ahora que se han identificado todos los requisitos se va a definir las tareas de desarrollo del proyecto

Código tarea	Descripción	Duración (horas)
T1	Captura de requisitos	2
T1.1	Identificación de requisitos de los Usuarios.	2
T2	Realización de casos de uso	3
T3	Diagrama de secuencia iniciar sesión	1
T3.1	Diagrama de secuencia enviar mensaje	1
T3.2	Diagrama de secuencia añadir amigo	1
T3.3	Diagrama de secuencia comprar videojuego	1
T3.4	Diagrama de secuencia registrase	1
T4	Diseño de la base de datos	3
T4.1	Diagrama Entidad Interrelación	1
T4.2	Esquema relacional estándar	1
T4.3	Grafo relacional	0.5
T4.4	Normalización	2
T4.5	Insertar datos en las tablas	2
T5	Definir entorno de hardware y software.	0.50
T6.1	Estudio de las tecnologías a utilizar.	0.50
T7	Creación de las interfaces gráficas	3
T8	Realizar pruebas y corregir errores.	4
T8.1	Realizar pruebas unitarias.	3
T8.2	Realizar pruebas del sistema completo.	4
T8.3	Realizar informe de pruebas y resultados.	4
T9	Realizar documentación. Informe de Instalación.	5
Total horas		50



3.3. Metodología a seguir para la realización del proyecto

Ahora que ya he listado las tareas a realizar durante el desarrollo de la aplicación web, voy a establecer la metodología que aplicaré para la realización de estas.

Metodología agile

Se pretende aplicar una metodología la cual los usuarios siempre tengan una implicación en el proyecto, permitiendo así comprobar y validar los requisitos y el diseño antes de comenzar una nueva fase.

Una vez que los usuarios den el visto bueno, cumpliendo con los requisitos, el proceso se reinicia con un conjunto de funcionalidades nuevas.



Lo primero en realizarse será el diseño de la base de datos, además de la interfaz de usuario de la aplicación, se implementará la función de inicio de sesión y de registro de usuario.

Lo siguiente en realizarse será la implementación del resto de funciones. Se corregirán las interfaces en caso de ser poco amigables.

3.4. Planificación temporal de tareas

3.5. Presupuesto (gastos, ingresos, beneficios)

Gracias a un plan económico y financiero voy a conocer los costes y beneficios de realizar este proyecto, ya que me proporciona la información acerca de cuanto me cuesta, producir, vender y ofrecer mi servicio.

3.5.1. Costes fijos

Costes Fijos	
Suministro Luz y Gas	18,74 €
Internet	10 €
Salario	900 €
Cuotas autónomos	50 €
Impuesto de IRPF e IVA	
Dominio y alojamiento web	4,99 €
TOTAL	983,73 €

Total, costes fijos: 983,73 € * 3 meses = 2951,19 €

3.5.2. Costes variables

Costes Variables	
Material (papel)	2 €
Innovación tecnológica (software informático)	50 €
TOTAL	60 € / servicio

Total, costes variables: 60 € * 3 meses = 180 €

3.5.3. Costes Totales

El capital necesario para llevar a cabo mi proyecto es de:
 $2951,19 + 180 = 3.131,19$

3.5.4. Presupuesto

Realizando una estimación el presupuesto final sería de 3500€

Dentro de este entran distintas cosas:

- El plazo de entrega se estima en un plazo aproximado de 3 meses.
- Se aplicará el IVA vigente en el momento del pago.
- Se realizará un diseño que se adapte a las especificaciones de los usuarios.

3.5.5. Contrato

Ver anexo I: Contrato



4. Documento de análisis y diseño

4.1. Modelo de datos. Análisis y diseño de base de datos

Para la realización de mi aplicación web he optado por utilizar el gestor de base de datos SQLite 3, lo he escogido por sus principales características ya pueden ser:

- La base de datos completa se encuentra en un solo archivo.
- Puede funcionar enteramente en memoria, lo que la hace muy rápida.
- Es totalmente auto contenida (sin dependencias externas).
- Cuenta con librerías de acceso para muchos lenguajes de programación.
- Soporta texto en formato UTF-8 y UTF-16, así como datos numéricos de 64 bits.
- Soporta funciones SQL definidas por el usuario (UDF).
- El código fuente es de dominio público y se encuentra muy bien documentado.



4.1.1. Diagrama E/R

Para representar el modelo conceptual se usará el modelo Entidad/Relación. Este modelo consiste en plasmar el resultado del análisis del problema mediante diagramas entidad-relación

Estos diagramas fueron propuestos por Peter Chen a mediados de los años 70. La notación es muy sencilla para que los usuarios (no técnicos) puedan validar si el modelo propuesto se ajusta a sus necesidades

Entidad: Cualquier tipo de objeto o concepto sobre el que se recoge información. Se representan mediante un cuadrado. Hay dos tipos de entidades:

- Fuertes: existe por méritos propios
- Débiles: su existencia depende de otra entidad fuerte

Relación: Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que exprese la finalidad de la relación.

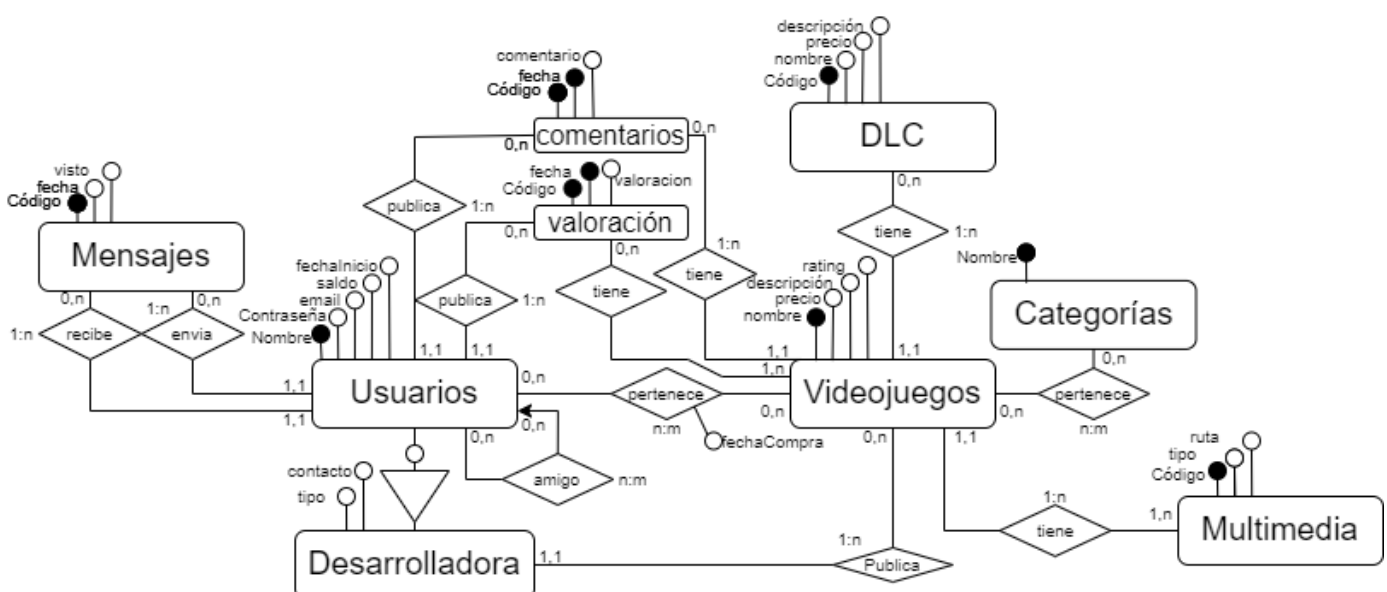
Se representan mediante rombos y su nombre aparece en el interior. Se clasifican según su grado, que es el número de entidades que participan en la relación

- Relaciones binarias
- Relaciones ternarias
- Relaciones unarias o reflexivas
- Relaciones n-arias

Atributos: Los atributos son las características o propiedades que la definen como entidad

- **Atributo clave:** Un atributo subrayado significa que es el atributo clave. Este campo no se puede repetir en ningún registro de la tabla que se creará a partir de la entidad
- **Atributo de relación:** Es aquel que es propio de una relación y que interviene en la relación de dos o más entidades

A continuación muestro el diagrama entidad relación de mi aplicación web





Un usuario puede comprar cero o varios videojuegos (0,n)
Un usuario puede ser desarrollador.
Un usuario puede enviar cero o varios mensajes (0,n)
Un usuario puede recibir cero o varios mensajes (0,n)
Un usuario puede tener cero o varios amigos (0,n)
Un usuario puede publicar cero o varias valoraciones (0,n)
Un usuario puede publicar cero o varios comentarios (0,n)
Un videojuego puede tener cero o varios comentarios (0,n)
Un videojuego puede tener cero o varias valoraciones (0,n)
Un videojuego puede tener cero o varios comentarios (0,n)
Un DLC será de un único videojuego (1,1)
Un videojuego puede pertenecer a cero a varias categorías(0,n)
Un videojuego debe tener uno o varios archivos de multimedia (1,n)
Un desarrollador puede publicar cero o varios videojuegos (0,n)

4.1.2. Esquema Relacional Estándar

Usuarios (nombre, email, contraseña, saldo, fechaInicio)

CP: {nombre}

Desarrolladora (nombre, contacto, tipo)

CP:{nombre}

CF:{nombre} referencia Usuarios

Mensajes (código, visto, fecha, emisor, receptor)

CP: {codigo}

CF:{emisor, receptor} referencia Usuarios

Videojuegos (nombre, desarrolladora, precio, descripción, rating)

CP:{nombre}

CF:{desarrolladora} referencia Desarrolladora

Categorías (nombre)

CP:{nombre}

Multimedia (código, videojuego, tipo, ruta)

CP:{código}

CF:{videojuego} referencia Videojuego

DLC (código, nombre, precio, descripción, videojuego)

CP:{código}

CF:{videojuego} referencia videojuego

Valoración (código, fecha, usuario, videojuego, valoración)

CP:{código, fecha, usuario, videojuego}

CF:{usuario} referencia usuarios

CF:{videojuego} referencia videojuegos



4.1.4. Normalización

Habitualmente, el diseño de una base de datos termina en el paso del modelo entidad-relación al modelo relacional. No obstante, siempre que se diseña un sistema, no solo una base de datos, sino también cualquier tipo de solución informática, se ha de medir la calidad de la misma, y si no cumple determinados criterios de calidad, hay que realizar, de forma iterativa, sucesivos refinamientos en el diseño, para alcanzar la calidad deseada

Uno de los parámetros que mide la calidad de una base de datos es la forma normal en la que se encuentra su diseño

El proceso de obligar a los atributos de un diseño a cumplir ciertas formas normales se llama normalización

1FN—Primera Forma Normal

En esta forma normal se prohíbe que en una tabla haya atributos que puedan tomar más de un valor. Esta forma normal es inherente al modelo relacional puesto que las tablas gestionadas por un SGBD relacional están construidas de esta forma

Además, no se admiten duplicados.

En mi caso ya cumplo esta forma.

2FN—Segunda Forma Normal

FN 2: Un diseño se encuentra en FN2 si está en FN1 y además, cada atributo que no forma parte de la clave tiene dependencia completa de la clave principal

En mi caso ya cumplo esta forma.

3FN—Tercera Forma Normal

Un diseño se encuentra en FN3 si están en FN2 y además no hay ningún atributo no clave que depende de forma transitiva de la clave

En mi caso ya cumplo esta forma.



4.1.5. Dominios de atributos

Cada uno de los atributos de una entidad está asociado a un grupo de valores

Tabla Usuarios					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Nombre	Varchar	20	*		
Email	Varchar	100			
Contraseña	Varchar	50			
Saldo	int				
fechalnicio	date				

Tabla Desarrolladora					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Nombre	Varchar	20	*	* usuarios	
Contacto	Text				
Tipo	Enum				

Tabla Videojuegos					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Nombre	Varchar	100	*		
Desarrolladora	Varchar	20		* desarrolladora	
Descripción	Text				
Precio	int				
Rating	int				

Tabla Amigos					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
usuario	Varchar	20	*	* Usuarios	
Amigo	Varchar	20	*	* Usuarios	

Tabla Mensajes					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Código	Int		*		
Visto	Boolean				
Fecha	Date				
Emisor	Varchar	20		* usuarios	
Receptor	Varchar	20		* usuarios	

Tabla Comentarios					
Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Código	Int		*		
Fecha	Date		*		
Usuario	Varchar	20	*	* usuarios	
Videojuego	Varchar	100	*	* Videojuegos	
Comentario	Text				



Tabla Valoración

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Código	Int		*		
Fecha	Date		*		
Usuario	Varchar	20	*	*usuarios	
Videojuego	Varchar	100	*	*Videojuegos	
Valoracion	int				

Tabla Juegos_usuario

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Usuario	Varchar	20	*	*usuarios	
Videojuego	Varchar	100	*	*Videojuegos	
Fecha	Date		*		

Tabla DLC

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Código	Int		*		
Videojuego	Varchar	100		*Videojuegos	
Nombre	Varchar	50			
Preico	Int				
Descripción	text				

Tabla Multimedia

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Código	Int		*		
Videojuego	Varchar	100		*Videojuegos	
Tipo	Enum				
Ruta	text				

Tabla Categoría_videojuego

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Categoría	Varchar	50	*	*categorías	
Videojuego	Varchar	100	*	*Videojuegos	

Tabla Categorías

Nombre Atributo	Tipo Dato	Longitud	PRIMARY_KEY	FOREIGN_KEY	NULL
Nombre	Varchar	50	*		

4.1.6. Scripts de la Base de Datos.

Ver Anexo II: Scripts Base de Datos



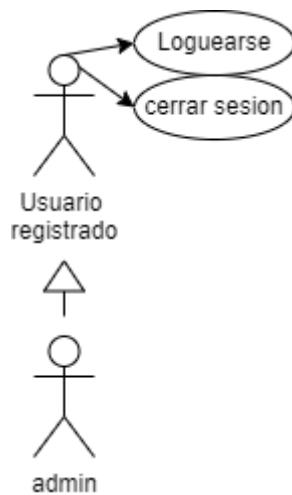
4.2. Análisis y diseño del sistema funcional (Diagramas que procedan como el de casos de uso, de clases, de secuencia, de actividad, de estados, de flujo...)

En este punto el objetivo es obtener los elementos que han sido definidos en la etapa de análisis, para así obtener los diagramas de casos de uso, de secuencia, actividad

4.2.1. Lista de actores

Actor: ACT-001 Admin	
Descripción	Administrador de Robustiano

Actor: ACT-001 Usuario	
Descripción	Usuario registrado en Robustiano



Ahora mostraré los casos de uso, diagramas de actividad y de secuencia realizados para el desarrollo de la aplicación.

Caso de Uso	Descripción
CU-001	Identificación de usuarios
CU-002	Mensajes
CU-003	Añadir amigos
CU-004	Comentarios
CU-005	Valoraciones
CU-006	Publicar videojuego
CU-007	Añadir amigos
CU-008	Publicar DLC
CU-009	Añadir saldo

**4.2.2. Caso de uso Usuarios de la aplicación**

Caso de Uso 001 Identificación de usuarios	
Descripción	Un usuario registrado puede acceder a la página
Actores	ACT-002 Usuario
Secuencia Normal	<ol style="list-style-type: none">1. El usuario accede a la pestaña login2. Introduce nombre/email y contraseña3. El sistema no da error y muestra la página de inicio4. Cerrar sesión
Secuencia(s) Alternativas(s)	
Precondiciones	El usuario debe estar registrado en el sistema
Requisitos relacionados	RF-002 Usuarios registrados RF-003 Identificación



- El usuario puede acceder al login
- El usuario puede iniciar sesión
- El usuario puede ver la página principal
- El usuario puede cerrar sesión

**4.2.3. Caso de uso Registro de usuario**

Caso de Uso 002 Registro de usuario	
Descripción	Un nuevo usuario necesita registrarse
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ -El usuario accede a la página de registro○ Introduce los datos en el formulario○ Le da al botón de registrar○ Le devuelve a la página de inicio
Secuencia(s) Alternativas(s)	<ul style="list-style-type: none">○ Salta un error en los campos erróneos
Precondiciones	
Requisitos relacionados	RF-002 Usuarios



- El usuario se registra
- Es necesario que esté registrado para las principales funciones
- Puede ver los videojuegos sin la posibilidad de compra si no se registra

**4.2.4. Caso de uso Mensajes**

Caso de Uso 002 Mensajes	
Descripción	Un usuario registrado envía un mensaje
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ -El usuario accede hace click en un amigo○ -El usuario escribe el mensaje○ -El usuario envía el mensaje
Secuencia(s) Alternativas(s)	<ul style="list-style-type: none">○ Ver mensaje
Precondiciones	El usuario debe estar registrado en el sistema y la persona quien se lo envía debe ser su amigo
Requisitos relacionados	RF-002 Usuarios registrados RF-003 Identificación



- El Usuario muestra todos los mensajes enviados hasta la fecha.
- El usuario puede enviar o responder a los mensajes.

**4.2.5. Caso de uso compra videojuegos**

Caso de Uso 002 Compra videojuegos	
Descripción	Un usuario registrado puede comprar videojuegos
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ El usuario accede a un videojuego○ El usuario le da a la opción de compra○ El videojuego se queda guardado en la biblioteca del usuario
Secuencia(s) Alternativas(s)	<ul style="list-style-type: none">○ Ver detalles del videojuego
Precondiciones	El usuario debe estar registrado en el sistema y debe tener el saldo suficiente
Requisitos relacionados	RF-002 Usuarios registrados RF-003 Identificación



- -Seleccionar videojuego
- -Comprar
- - Ver información detallada del videojuego

**4.2.6. Caso de uso Comentarios**

Caso de Uso 002 Comentarios	
Descripción	Comentarios de los videojuegos
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ El usuario accede a cualquier videojuego mediante el listado o una búsqueda.○ Añade un comentario.
Secuencia(s) Alternativas(s)	
Precondiciones	El usuario debe estar registrado en el sistema.
Requisitos relacionados	RF-031 Añadir comentario



- -Añadir comentario

**4.2.7. Caso de uso Valoración**

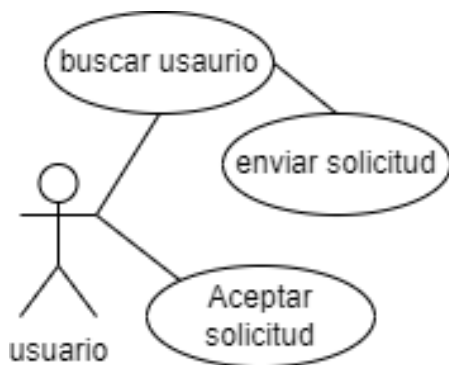
Caso de Uso 002 Valoración	
Descripción	Valoración de los videojuegos
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ El usuario accede a cualquier videojuego mediante el listado o una búsqueda.○ Añade una valoración de hasta 5 puntos.
Secuencia(s) Alternativas(s)	
Precondiciones	El usuario debe estar registrado en el sistema.
Requisitos relacionados	RF-031 Añadir Valoración



- -Añadir Valoración

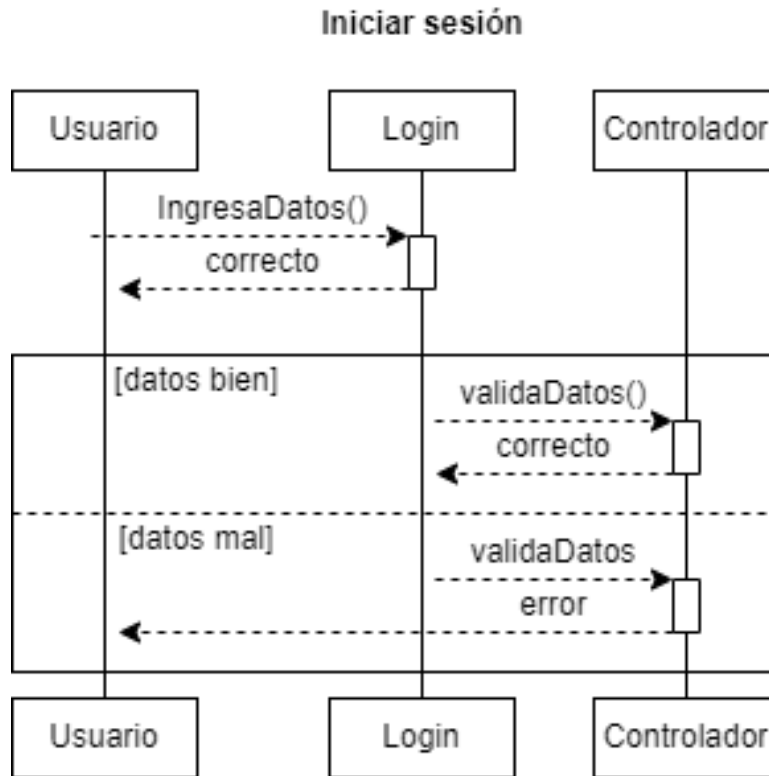
**4.2.8. Caso de uso amigos**

Caso de Uso 002 Amigos	
Descripción	Añadir amigos
Actores	ACT-002 Usuario
Secuencia Normal	<ul style="list-style-type: none">○ El usuario accede a la sección de usuarios donde puede buscar un usuario.○ Envía una solicitud de amistad a ese usuario
Secuencia(s) Alternativas(s)	<ul style="list-style-type: none">○ Acepta la solicitud de amistad
Precondiciones	El usuario debe estar registrado en el sistema.
Requisitos relacionados	RF-031 Añadir Valoración



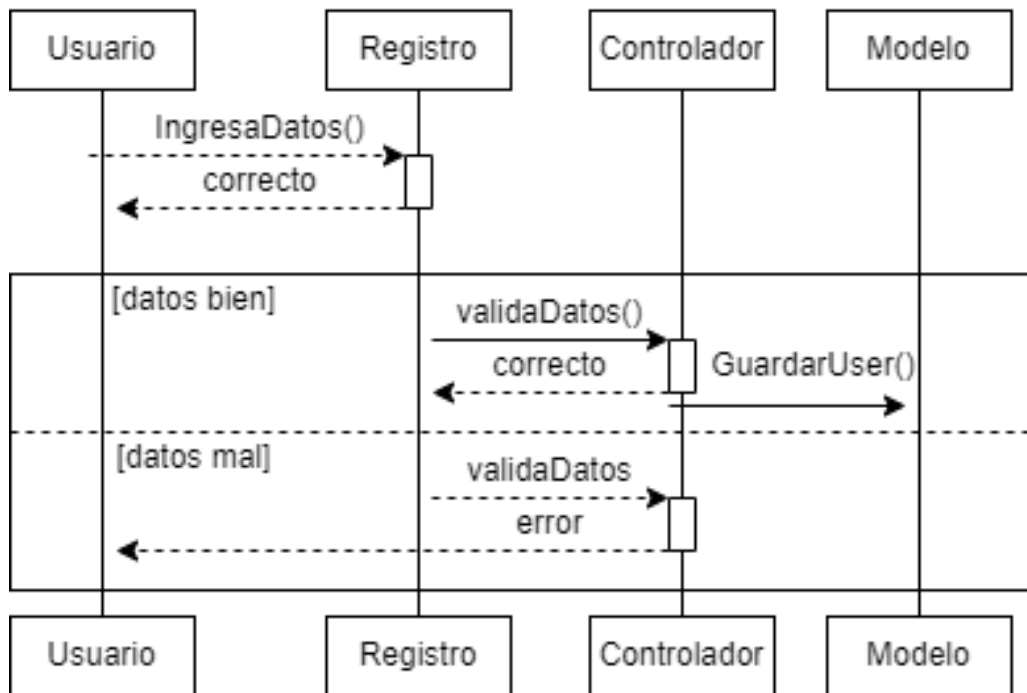
- El usuario puede enviar o recibir solicitudes de amistad
- El usuario puede aceptar o denegar las solicitudes

4.2.9. Diagrama de despliegue Login usuarios

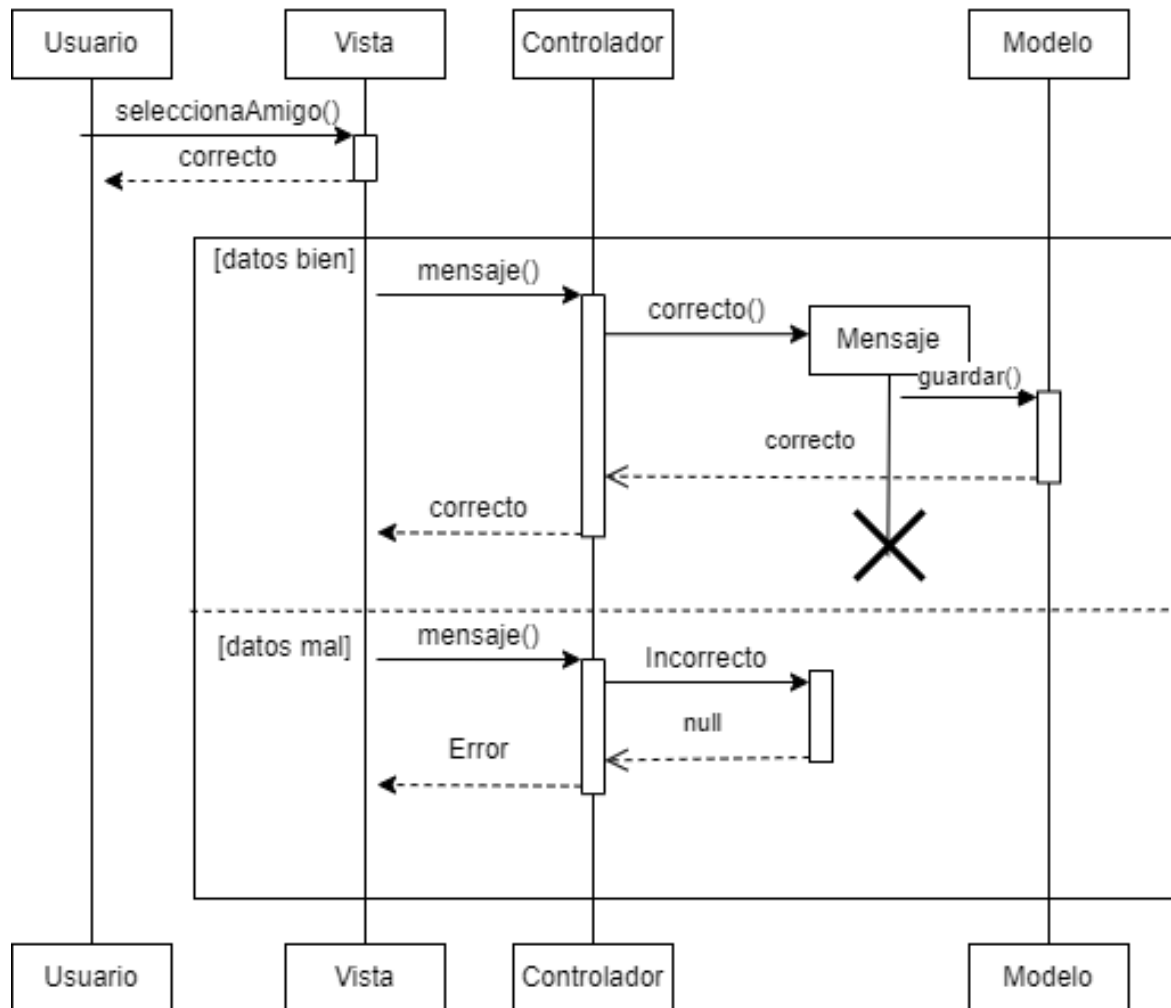


En este diagrama de despliegue se muestra la interacción del usuario con la aplicación web a la hora de iniciar sesión. Para esto el usuario se tendría que haber registrado previamente en la aplicación.

Si el usuario/email y la contraseña son correctos se devolverá a la página de inicio pero esta vez con más opciones disponibles, en caso de ser incorrectas se devolverá un error.

**4.2.10. Diagrama de despliegue Registro de usuario****Registro usuario**

Como se demuestra el diagrama es muy parecido al de inicio de sesión con la diferencia que si los datos son correctos estos se guardarán en la base de datos

4.2.11. Diagrama de despliegue Enviar mensaje**Enviar mensaje**

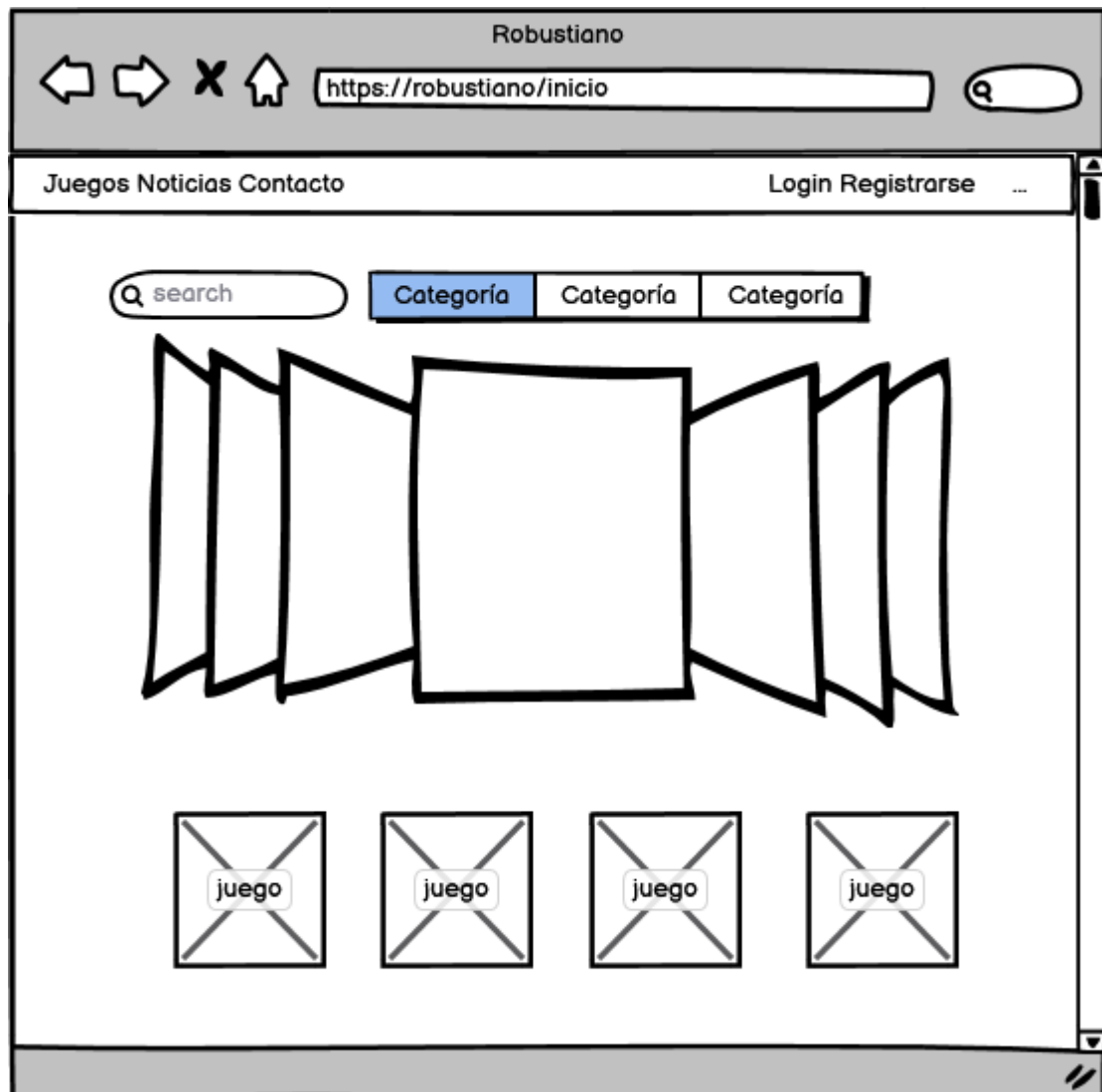
Aquí podemos observar la interacción con el sistema para poder enviar un mensaje.

El usuario una vez registrado accede a sus amigos y escoge a uno para enviar el mensaje, esto le sitúa en el formulario para enviar mensaje. Una vez redactado el mensaje le da a enviar, si está correcto el mensaje será enviado, en caso de no ser así dará error.

4.3. Análisis y diseño de la interfaz de usuarios. Mockups

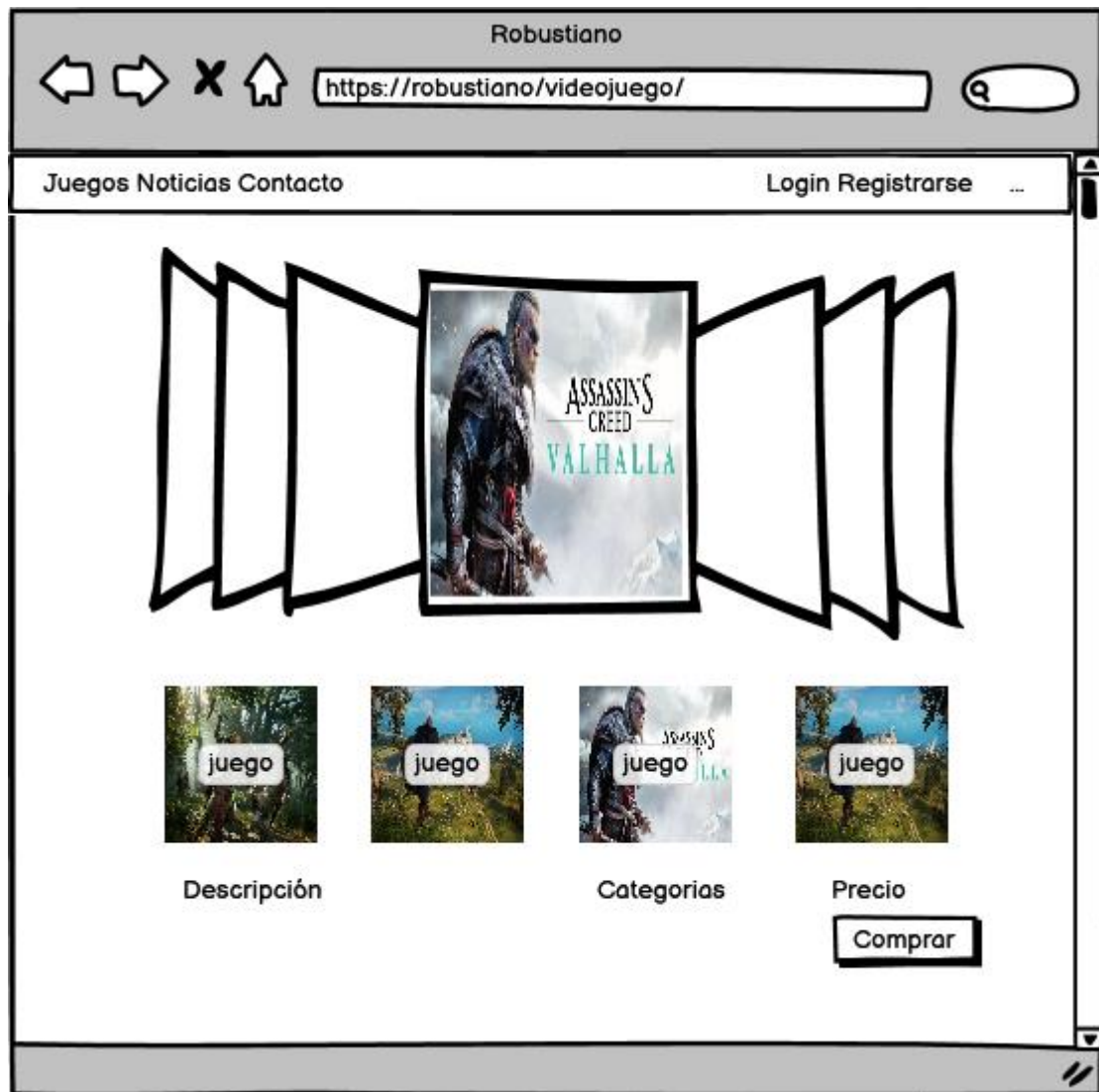
En este punto se analizan las decisiones tomadas anteriormente respecto al diseño que van a tener las interfaces de usuario.

A continuación, presento algunos borradores de las interfaces de usuario, comenzando con la página de inicio.



Página de inicio, donde se muestra un menú superior con las principales páginas. Justo debajo hay un menú de búsqueda y unas opciones de categoría para filtrar los videojuegos.

En medio hay un carrusel donde se muestran los 4 primeros videojuegos, justo debajo están el resto.



Página de videojuego, cuando accedes a cualquier videojuego hay un carrusel con 4 imágenes, justo debajo, las mismas 4 imágenes las cuales al hacer click se muestran arriba.

Debajo está la descripción, las categorías a las que pertenece y el botón de compra junto al precio.



Robustiano

← → × ↶ 🔍 https://robustiano/login

Juegos Noticias Contacto Login Registrarse

Username

Contraseña

Login

[¿No te acuerdas de la contraseña?](#)

Página de login, un formulario sencillo donde el usuario debe ingresar su nombre y contraseña, si no se acuerda de esta hay una opción para recuperarla.



Robustiano


← → × 🏠 🔍

Juegos Noticias Contacto Login Registrarse

Username

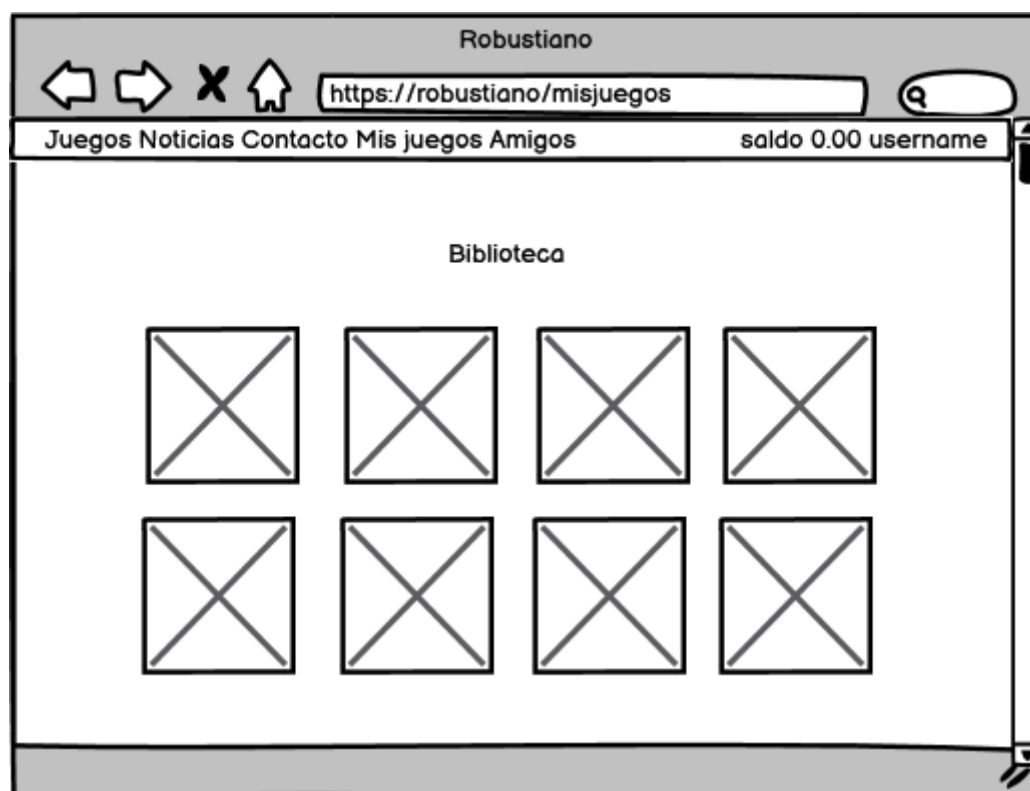
Email

Contraseña

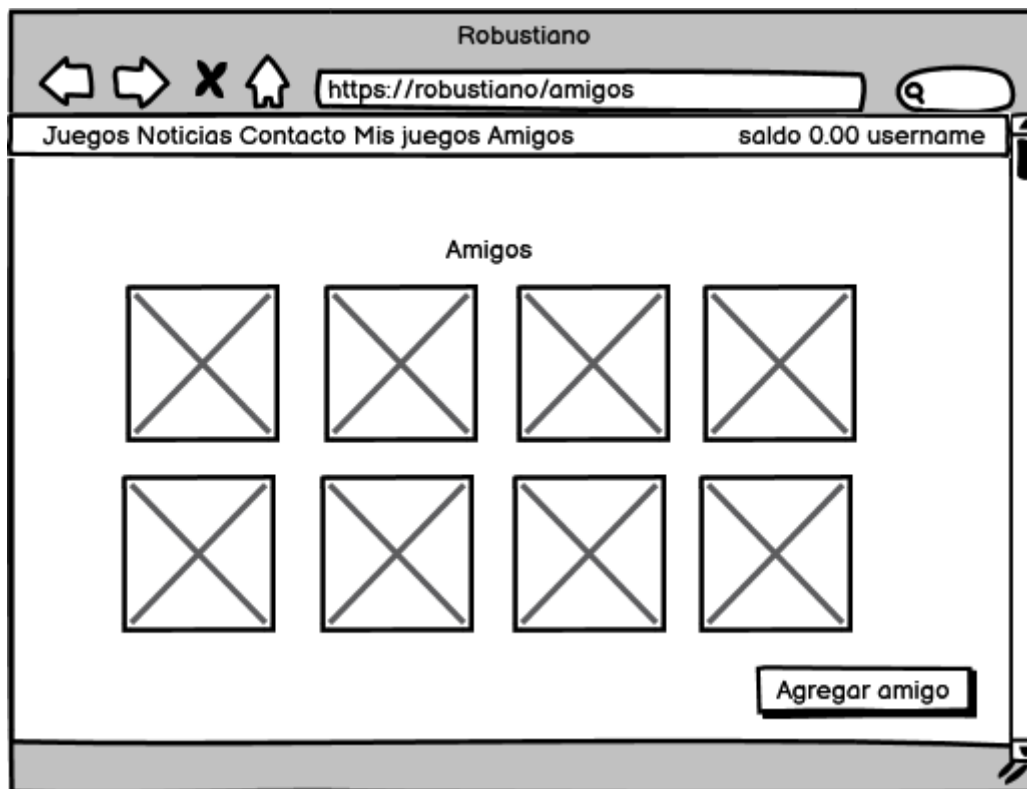
Seleccionar imagen 

Registrarse

Página de registro, para poder acceder a las principales funciones es necesario estar registrado, para ello está este formulario donde una vez completo el usuario podrá iniciar sesión



Página de la biblioteca de juegos, una vez que el usuario haya iniciado sesión puede acceder a esta pestaña donde se muestran los juegos que tenga comprados



Página de los amigos, una vez que el usuario haya iniciado sesión puede acceder a esta pestaña, donde se muestran todos los amigos.

Al lado hay un botón que al presionarlo accede a otra pestaña con todos los usuarios para así poder añadir nuevos amigos.



Robustiano

← → × 🏠 🔍

Juegos Noticias Contacto Mis juegos Amigos saldo 0.00 admin ...

Saldo

Código

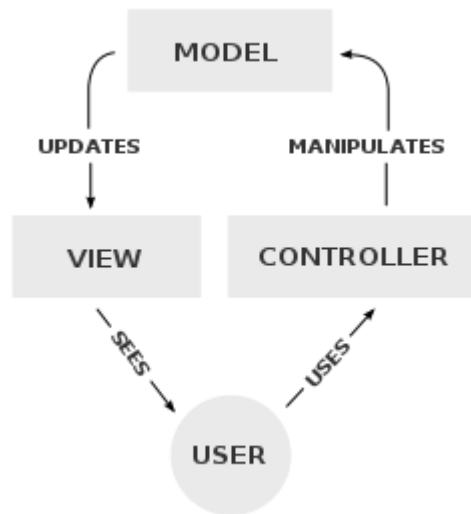
Esta pestaña solo es accesible para el administrador, aquí podrá generar tarjetas para que los usuarios puedan agregar saldo a sus cuentas.

4.4. Diseño de la arquitectura de la aplicación

Para el desarrollo de mi aplicación utilizaré la arquitectura cliente/servidor, un modelo de tres capas que pretende separar la capa de datos y la lógica de negocio de la capa de presentación.

- **Capa de datos:** la base de datos de la aplicación
- **Capa de negocio:** el servidor de la aplicación que controla peticiones y respuestas al usuario.
- **Capa de presentación:** básicamente es lo que ve el usuario.

Voy a utilizar un modelo llamado MVC (modelo, vista, controlador).





Aquí muestro un diagrama de la estructura interna de la aplicación

```
> .vscode
> env
▼ media
  > audios
  > estilos
  > images
▼ tienda
  > __pycache__
  > migrations
  > templates
  🌀 __init__.py
  🌀 admin.py
  🌀 apps.py
  🌀 forms.py
  🌀 models.py
  🌀 tests.py
  🌀 views.py
▼ tiendaVideojuegos
  > __pycache__
  > templates
  🌀 __init__.py
  🌀 asgi.py
  🌀 settings.py
  🌀 urls.py
  🌀 wsgi.py
🔹 .gitignore
≡ db.sqlite3      M
🌀 manage.py
① README.md
≡ requirements.txt
```



4.4.1. Tecnologías/Herramientas usadas y descripción de las mismas.

Lo básico para poder llevar a cabo esto es:

a) Hardware

- **Servidor web:** No ha de ser muy potente, con tener entre 4GB y 8GB de RAM es suficiente, en cuanto al almacenamiento sí que es necesario algo más potente y rápido, estaría bien algo de 10 TB.
- La versión del software y del sistema operativo han de estar actualizadas pero con la posibilidad de cambiar de versiones

b) Software

- **Gestor de base de datos:** como gestor de base de datos SQLite 3 es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña biblioteca escrita en C,
- **Servidor web Apache HTTP Server Project:** es un servidor HTTP de código abierto que implementa el protocolo HTTP. El servidor Apache es desarrollado por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server. Como ventajas es extensible y multi-plataforma.
- **GitHub:** Plataforma para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador.
- **Tecnologías y Lenguajes:** Python como lenguaje de programación, HTML 5 para la maquetación, CSS y Bootstrap 5 para hojas de estilo, javascript para efectos y otras funciones, SQLite 3 para la base de datos.
 - **Lengua de programación Python:** lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código, se utiliza para desarrollar aplicaciones de todo tipo, ejemplos: Instagram, Netflix, Spotify, entre otros. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.
 - **Framework Bootstarp:** biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales.
 - **Sistema operativo:** Debian GNU/Linux es un sistema operativo libre, desarrollado por miles de voluntarios de todo el mundo, que colaboran a través de Internet.

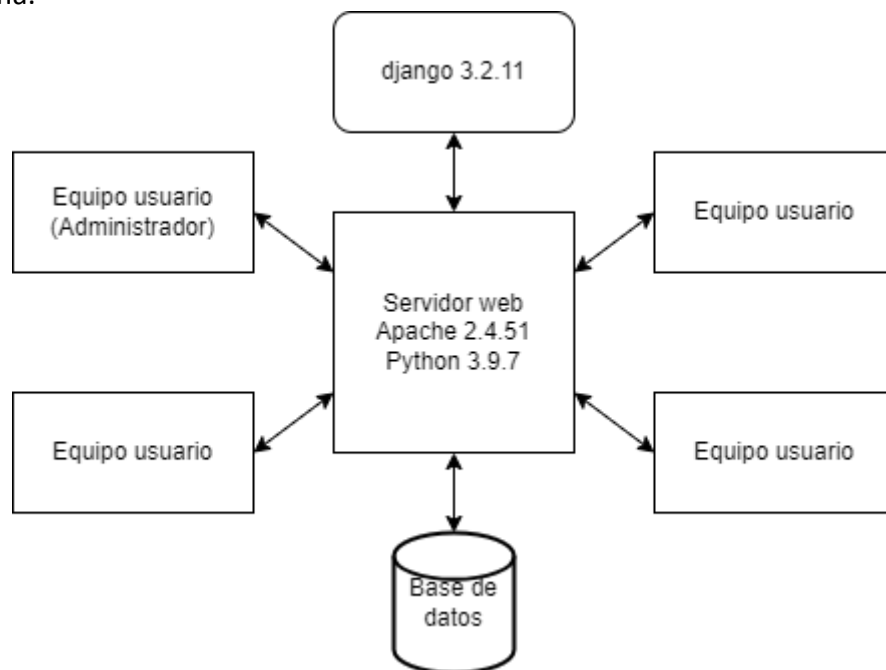
La dedicación de Debian al software libre, su base de voluntarios, su naturaleza no comercial y su modelo de desarrollo abierto la distingue de otras distribuciones del sistema operativo GNU. Todos estos aspectos y más se recogen en el llamado Contrato Social de Debian. Debian se caracteriza por no tener las últimas novedades en GNU/Linux, pero sí tener el sistema operativo más estable posible. Esto se logra por medio de paquetes y librerías antiguas pero con muchos meses de pruebas, asegurando la máxima estabilidad por cada versión que es lanzada por la comunidad de Debian.



- **Django:** framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo–vista–controlador (MVC). Fue desarrollado originalmente para gestionar páginas web orientadas a noticias de la World Company de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005; el framework fue nombrado en alusión al guitarrista de jazz gitano Django Reinhardt.

4.4.2. Arquitectura de componentes de la aplicación

A continuación presento el esquema que representa los diferentes actores y componentes lógicos del sistema.



Los principales componentes son:

- **Equipos de cliente:** Son aquellos equipos desde donde accederán los usuarios a la aplicación web. Habrán dos perfiles (usuario, administrador).
- **Servidor Web Apache + lenguaje de programación Python:** ambos son derivados de los requisitos tecnológicos de la aplicación.
- **Base de dato:** Se encargará de guardar la información.
- **Django:** permitirá procesar algunas solicitudes al servidor sin necesidad de recargar las páginas.



5. Documento de implementación e implantación del sistema

En este punto realizaré las pruebas funcionales para el correcto funcionamiento

5.1. Pruebas

5.1.1. Pruebas unitarias

5.1.2. Pruebas funcionales

Aquí describo el Plan de Pruebas Funcionales que seguirá mi proyecto.

CP_00	
Descripción	
Precondición	
Datos de entrada	
Datos de salida	

CP_001 Requisitos generales	
Descripción	<ul style="list-style-type: none">- Se muestra un mensaje de error si el usuario hace una acción errónea- Se muestra un mensaje correcto en caso de éxito del usuario
Precondición	El usuario debe estar registrado en el sistema
Datos de entrada	Comprar videojuego, añadir saldo, añadir amigo, publicar videojuego
Datos de salida	<ul style="list-style-type: none">a) Si un campo está vacío sale el mensaje de aviso. OKb) Si el usuario ya existe el sistema da un mensaje de error. OKc) Si todo va bien, se mostrara un mensaje. OK

CP_002 Usuarios registrados	
Descripción	Los usuarios registrados pueden acceder mediante el nombre y la contraseña de cuando se registraron.
Precondición	El usuario debe haberse registrado previamente
Datos de entrada	Se inserta el nombre y la contraseña en el login
Datos de salida	<ul style="list-style-type: none">a) Si los datos son correctos se redirige a la página principal. OKb) En caso de ser incorrecto muestra el error. OK

CP_003 Registro de usuarios	
Descripción	Los usuarios se pueden registrar rellenando unos campos
Precondición	
Datos de entrada	Se inserta n los datos requeridos en la página de registro
Datos de salida	<ul style="list-style-type: none">c) Si los datos son correctos se redirige a la página principal. OKd) En caso de ser incorrecto muestra el error. OK

**CP_004 identificación**

Descripción	Si el usuario registrado ha iniciado sesión previamente, el inicio de sesión tiene que ser automático.
Precondición	El usuario debe haberse registrado previamente
Datos de entrada	Se accede a la aplicación.
Datos de salida	El usuario accede automáticamente. OK

CP_005 Cerrar sesión

Descripción	Los usuarios que han iniciado sesión pueden cerrarla mediante un botón. El usuario será redirigido a la página de inicio.
Precondición	El usuario debe haberse registrado previamente
Datos de entrada	Se accede al menú del usuario junto al icono de este y se muestra un desplegable.
Datos de salida	El usuario cierra la sesión y es redirigido a la página login. OK

CP_006 Descripción del Home

Descripción	La pagina principal consta de 3 partes <ul style="list-style-type: none">- Menú superior- Menú inferior- Contenido central
Precondición	
Datos de entrada	El usuario accede a la página principal
Datos de salida	El usuario visualiza todos los contenidos. OK

CP_007 Descripción del menú superior

Descripción	Contiene el logo de la empresa junto a las principales páginas y la opción de registrarse y de iniciar sesión
Precondición	
Datos de entrada	El usuario accede a la página principal
Datos de salida	El usuario visualiza todos los contenidos. OK

CP_008 Descripción del menú inferior

Descripción	Esta justo debajo del menú superior y muestra un buscador y diferentes categorías de videojuegos.
Precondición	
Datos de entrada	El usuario accede a la página principal
Datos de salida	El usuario visualiza todos los contenidos. OK

CP_009 Descripción del contenido central

Descripción	Justo debajo del menú inferior y muestra un carrusel con 4 videojuegos y justo debajo todos los videojuegos disponibles.
Precondición	
Datos de entrada	El usuario accede a la página principal
Datos de salida	El usuario visualiza todos los contenidos. OK

**CP_010 Panel de administrador**

Descripción	Aquí podrá generar códigos con dinero para que así los usuarios puedan agregar saldo a sus cuentas
Precondición	El usuario debe ser administrador
Datos de entrada	La cantidad de saldo y el código que referencia a este.
Datos de salida	Se genera un código con saldo. OK

CP_011 Añadir amigo

Descripción	El usuario puede buscar a otro para añadirlo como amigo
Precondición	El usuario debe estar registrado
Datos de entrada	Selecciona un usuario para agregar.
Datos de salida	a) Le agrega como amigo. OK b) En caso de que ya lo sean el usuario no se muestra. OK

CP_012 Comprar videojuego

Descripción	El usuario puede comprar un videojuego
Precondición	El usuario debe estar registrado y tener el saldo suficiente
Datos de entrada	El usuario escoge un videojuego y le da a la opción de compra
Datos de salida	a) En caso de tener suficiente saldo el juego es añadido a la biblioteca de juegos. OK b) En caso de no tener suficiente saldo dará un error. OK c) En caso de tener ya ese videojuego en la tienda dará un error. OK

CP_013 añadir saldo

Descripción	Si el usuario tiene un código podrá añadir saldo a su cuenta
Precondición	El usuario debe estar registrado y tener un código
Datos de entrada	El usuario entra en la opción añadir saldo y canjea el código
Datos de salida	a) Si el código es válido se le agregará el saldo. OK b) Si el código no existe dará un error. OK c) Si el código ya ha sido canjeado dará un error. OK



5.2. **Instalación y configuración**

a) **Requisitos**

Como se ha especificado anteriormente en las tecnologías usadas los requisitos para llevar a cabo la aplicación son:

- Python 3.9.7
- Django 3.2.11
- Base de datos SQLite 3
- Sistema operativo Windows
- Servidor web Apache

b) **Instalación y configuración**

- Para la instalación se deberá descargar el proyecto y en un entorno python sobre el que pueda funcionar
 - Importamos el fichero requirements.txt
 - Activamos el servicio con el comando "py manage.py runserver", la página estará lista para su uso.
 - Para acceder a la misma simplemente deberá introducir la dirección IP asignada en el navegador.
 - La dirección sería: 127.0.0.1:8000
-



5.3. Manual de usuario

Aquí se pretende enseñar las funcionalidades básicas de la aplicación y su uso

a) Manejo de la aplicación

Inicio de sesión o Login

En este punto es donde el usuario podrá iniciar sesión si se ha registrado previamente.

Pag: <http://127.0.0.1:8000/accounts/login/>

The screenshot shows a dark-themed login page. At the top left is a logo, and next to it are links for 'Juegos', 'Noticias', and 'Contacto'. At the top right are links for 'Login' and 'Register'. The main form area contains a 'Username:' label followed by a text input field, a 'Password:' label followed by a text input field, a 'login' button, and a 'Lost password?' link.

Cuando el usuario complete los campos con los datos correctos y le dé al botón de login se accederá a la página principal, en caso de que los datos no sean correctos se mostrará un error.

Página de registro

En esta página el usuario se puede registrar en caso de que no lo esté

Pag: <http://127.0.0.1:8000/registro/>

The screenshot shows a dark-themed registration page. It contains several form fields: 'Username:', 'Email address:', 'Password:', 'First name:', and 'Last name:', each followed by a text input field. Below these is an 'Imagen:' label followed by a file selection button labeled 'Seleccionar archivo'. At the bottom right is a 'Registrarse' button.

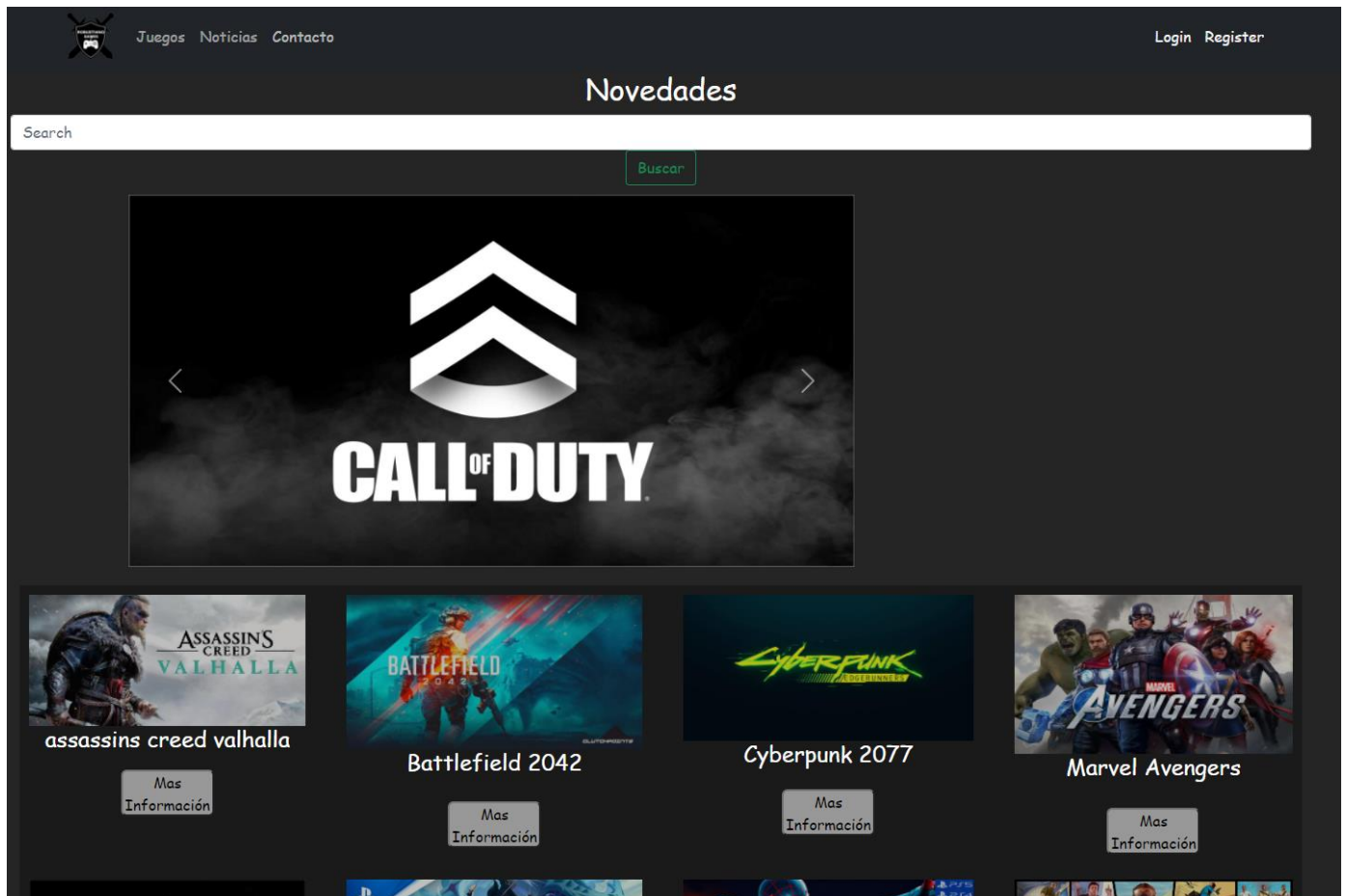


Aquí el usuario debe rellenar los campos, una vez que estén todos, si está todo correcto el usuario se podrá registrar para así poder iniciar sesión.

Página de inicio

Esta es la página de inicio a la que cualquier usuario puede acceder sin necesidad de estar registrado.

Pag: <http://127.0.0.1:8000>

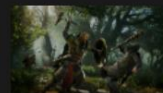


Como se puede observar aquí se muestran todos los videojuegos

Página de compra

Una vez que el usuario haya iniciado sesión podrá comprar videojuegos

Pag: <http://127.0.0.1:8000/videojuego/assassins%20creed%20valhalla>



Información:

Assassin's Creed Valhalla es un videojuego desarrollado por Ubisoft Montreal y publicado por Ubisoft. Es el decimosegundo en importancia y el vigesimosegundo lanzado dentro de la saga de Assassin's Creed, y sucesor al juego del 2018 Assassin's Creed Odyssey.

accion
fantasia
mundo abierto

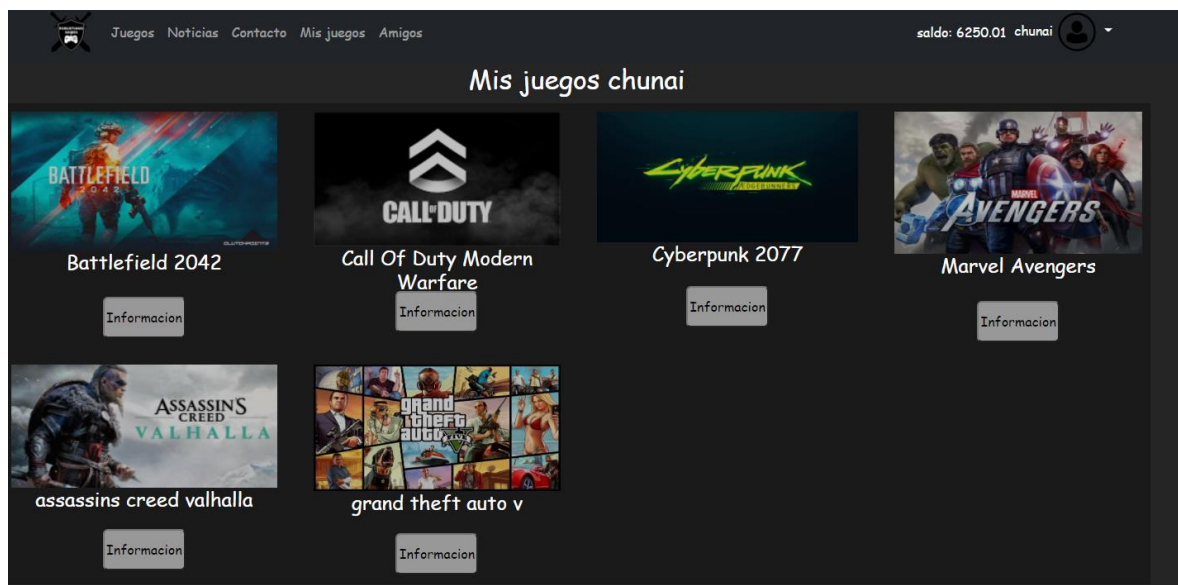
Precio: 69.99

Comprar

Como se puede ver, están las principales imágenes del juego, junto a la descripción, categorías, precio y la opción de compra. En caso de que el usuario tenga suficiente saldo lo podrá comprar.

Biblioteca de juegos

Si el usuario está registrado y ha comprado algún videojuego le aparecerán en su biblioteca
Pag: <http://127.0.0.1:8000/misJuegos/>





Aquí aparecen todos los juegos que ha comprado, en caso de estar vacía no saldría nada

Amigos

Aquí se muestra la lista de amigos agregados

Pag: <http://127.0.0.1:8000/misAmigos/>



Si se le da al botón de agregar amigos se mostrarán todos los usuarios para poder añadirlos

Pag: <http://127.0.0.1:8000/usuarios/>





Añadir saldo

En esta página puede añadir saldo con un código creado por el administrador

Pag: <http://127.0.0.1:8000/añadirSaldo/>

Codigo:

Si el código es correcto se le agregará saldo

b) Administrador

Crear código

Un administrador puede crear un código para que los usuarios puedan añadir saldo a sus cuentas.

Pag: <http://127.0.0.1:8000/crearTarjeta/c>

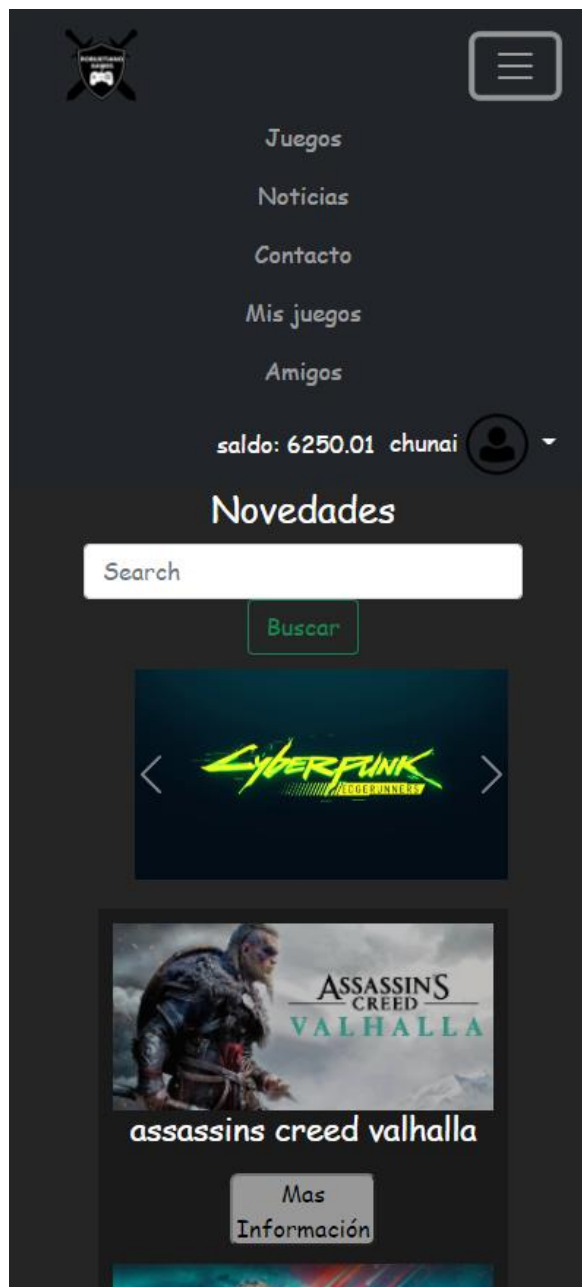
Saldo:

Codigo:

El saldo debe ser 10, 20, 50 o 100 y para el código se puede generar de manera aleatoria



Diseño responsive



Copia de seguridad

La copia de seguridad se realizó copiando los archivos en otro repositorio de github privado.



6. Documento de cierre

6.1. Resultados Obtenidos

Con este Trabajo Fin de Ciclo (TFC) demuestro mis habilidades aprendidas durante estos dos años de formación en el Ciclo Superior en Desarrollo de Aplicaciones Web.

Al terminar este trabajo doy por concluida una meta que tenía desde el día que empecé, desarrollar una tienda de videojuegos, este proyecto se va a quedar para mí, como un objetivo ya cumplido.

Su desarrollo me ha permitido adquirir conocimientos no solo técnicos si no también he aprendido a pensar, a meditar como realizar ciertas funciones, me ha ayudado a conseguir una aplicación amigable con los usuarios.

Con este trabajo me ha quedado claro que incluso aunque las cosas funciones siempre se pueden mejorar y siempre se puede seguir aprendiendo, me ha quedado claro que cada persona piensa de una manera distinta a la hora de realizar ciertas cosas, por eso sé, que después de esto voy a seguir aprendiendo cada día más.

Me he quedado satisfecho sabiendo que he conseguido lo que quería, que he superado mis metas y que sé que voy a seguir mejorando.

**6.2. Diario de bitácora**

Actividad	Fecha de ejecución
Captura de requisitos	1 al 2 de abril
Identificación de requisitos de los Usuarios.	1 al 2 de abril
Realización de casos de uso	2 al 5 de abril
Diagrama de secuencia iniciar sesión	5 al 5 de abril
Diagrama de secuencia enviar mensaje	6 al 7 de abril
Diagrama de secuencia añadir amigo	7 de abril
Diagrama de secuencia comprar videojuego	7 al 8 de abril
Diagrama de secuencia registrase	8 de abril
Diseño de la base de datos	9 al 12 de abril
Diagrama Entidad Interrelación	12 al 14 de abril
Esquema relacional estándar	14 al 15 de abril
Grafo relacional	15 de abril
Normalización	16 al 17 de abril
Insertar datos en las tablas	22 al 24 de abril
Definir entorno de hardware y software.	1 al 3 de mayo
Estudio de las tecnologías a utilizar.	4 de mayo
Creación de las interfaces gráficas	7 al 10 de mayo
Realizar pruebas y corregir errores.	11 al 19 de mayo
Realizar pruebas unitarias.	19 al 23 de mayo
Realizar pruebas del sistema completo.	23 al 25 de mayo
Realizar informe de pruebas y resultados.	27 de mayo
Realizar documentación. Informe de Instalación.	29 de mayo
Captura de requisitos	1 de junio

Repositorio:

<https://github.com/UnaiSuarez/Robustiano-Games>



7. Webgrafía

BOOTSTRAP – FRAMEWORK

<http://getbootstrap.com/>

GITHUB – CONTROL DE VERSIONES

<https://github.com/>

W3SCHOOLS - TUTORIAL

<http://www.w3schools.com/>

8. Anexos

ANEXO I: CONTRATO

CONTRATO DE DISEÑO DE PAGINA WEB

En _____ a ____ de _____ de 20__.

REUNIDOS

De una parte, D./D^a _____ (a partir de ahora “Diseñador”), mayor de edad, Diseñador Web, con domicilio en _____ y D.N.I. Nº _____.

Y de otra, D./D^a _____ (a partir de ahora “Cliente”), mayor de edad, _____, con domicilio en _____ y D.N.I. Nº _____.

EXPONEN

- I. Que el Diseñador es un **autónomo** que ofrece servicios especializados en el diseño y desarrollo de páginas web.
- II. Que ambas partes han convenido la **formalización del presente contrato de diseño de página web** con arreglo a varias cláusulas.

CLÁUSULAS

PRIMERA.- En virtud del presente contrato, el Diseñador se compromete a llevar a cabo, en los términos que en el mismo se establecen, el **diseño de una página web** para el Cliente, quien solicita los servicios de aquel.

SEGUNDA.- El diseño de la página web se llevará a cabo con arreglo a los plazos que se han establecido en



conversaciones previas. Se estima una **duración total del proyecto** de _____, siempre y cuando se entreguen los textos e imágenes en tiempo y forma (de no ser así, la duración total variará).

TERCERA.- El precio del diseño de la página web asciende a _____ (este precio **no incluye IVA**). El pago por parte del Cliente se realizará de la forma que acuerdan a continuación:

- I. En el momento de la firma del presente contrato se abonará un ____% de la cantidad acordada.
- II. La totalidad del precio restante, impuestos incluidos, se abonará, mediante **transferencia**, en el momento de la finalización del proyecto y publicación del sitio web, contra factura emitida.
- III. Un servicio de mantenimiento del sitio se considerará un **nuevo proyecto**, y estará sometido a las cláusulas y condiciones del contrato redactado para el caso.

El abono de cada una de las cantidades se hará efectivo mediante transferencia a la cuenta bancaria número _____, abierta en la entidad _____ a nombre de _____, en las formas acordadas en esta Cláusula.

CUARTA.- El Diseñador desarrollará el proyecto con la **colaboración activa** del Cliente para incorporar, según sus instrucciones, los contenidos del sitio web y facilitarse mutuamente cualquier documentación necesaria, tanto en soporte físico como digital.

El Diseñador se reserva el derecho, previa comunicación y acuerdo mutuo, a **ampliar el plazo de ejecución o modificar la fecha de publicación** por cuestiones técnicas si el Cliente solicitase una modificación esencial del proyecto acordado.

QUINTA.- El Diseñador, una vez termine el trabajo, **cede** al cliente **todos** los derechos de uso, modificación, reproducción total o parcial, traducción, adaptación, arreglo o cualquier otra transformación del diseño de la página web.

El diseñador **no** conservará los **derechos de propiedad intelectual** sobre ningún software, sistema, módulo, herramienta o cualquier otro elemento desarrollado durante el desarrollo del proyecto. La cesión o transferencia de derechos al cliente en las condiciones expresadas en la presente cláusula tendrá lugar a la firma del acta de entrega y recepción definitiva de la página web.

SEXTA.- Una vez entregado el proyecto web por parte del Diseñador, cualquier cambio, ampliación o modificación del proyecto se facturará como un **nuevo proyecto**.

SÉPTIMA.- Las partes se comprometen a mantener estrictamente la **confidencialidad** de la **información** facilitada entre las partes con ocasión del presente contrato de diseño de página web.

OCTAVA.- De acuerdo con sus principios de ética profesional, el Diseñador garantiza **que en ningún momento empleará técnicas ilegales de posicionamiento** (puertas traseras, duplicación de contenidos, granjas de enlaces, cloaking y otras análogas) que puedan suponer algún tipo de penalización para el Cliente por parte de los responsables de servicios de búsqueda por Internet.

NOVENA.- Si el Cliente **no abona el precio** acordado en plazo o no comunica en tiempo y forma los contenidos a publicar, el Diseñador **considerará resuelto el contrato**, sin reintegrar al Cliente cualquier cantidad adelantada y retirando cualquier contenido publicado en el sitio web si el incumplimiento no es subsanado, sin menoscabo de otras acciones legales pertinentes.

DÉCIMA.- Si el Diseñador **no crea la página web** en la forma o plazo estipulado o no comunica en tiempo y forma las razones para el retraso, el Cliente **considerará resuelto el contrato**, sin reintegrar al Diseñador



cualquier cantidad, sin menoscabo de otras acciones legales pertinentes.

DECIMOPRIMERA.- La página web publicada tendrá una garantía indefinida de funcionamiento a partir de la fecha de entrega del proyecto, siempre y cuando **su código se mantenga íntegro y sin modificación alguna**, de acuerdo a la copia del mismo entregada al Cliente al finalizar dicho proyecto, y siempre que se mantenga idéntico el servicio contratado a un tercero en materia de hosting, si es el caso.

DECIMOSEGUNDA.- Se establece un acuerdo de buena voluntad para que, en el caso de discrepancias y dependiendo del momento del proyecto, haya un **consenso** amigable por ambas partes.

DECIMOTERCERA.- El incumplimiento de alguna de las cláusulas será **causa de rescisión** del presente contrato por cualquiera de las partes.

Y en prueba de conformidad, firman y rubrican por duplicado y a un solo efecto en el lugar y fecha.

Leído y conforme,

Leído y conforme,

El Diseñador

El Cliente



ANEXO II: SCRIPT BASE DE DATOS

```
-- phpMyAdmin SQL Dump
-- version 5.1.1
-- https://www.phpmyadmin.net/
--
```

```
-- Servidor: 127.0.0.1:3306
-- Tiempo de generación: 27-05-2022 a las 10:33:54
-- Versión del servidor: 8.0.27
-- Versión de PHP: 7.4.26
```

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";
```

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;
```

```
--
-- Base de datos: `robustiano`
--
```

```
--
-- Estructura de tabla para la tabla `tbls_dlc`
--
```

```
DROP TABLE IF EXISTS `tbls_dlc`;
CREATE TABLE IF NOT EXISTS `tbls_dlc` (
  `codigo` int UNSIGNED NOT NULL,
  `videojuego` int UNSIGNED NOT NULL,
  `nombre` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `precio` int DEFAULT '0',
  `descripcion` text CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  PRIMARY KEY (`codigo`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Estructura de tabla para la tabla `tbl_amigos`
--
```




```
DROP TABLE IF EXISTS `tbl_amigos`;
CREATE TABLE IF NOT EXISTS `tbl_amigos` (
  `usuario` varchar(20) NOT NULL,
  `amigo` varchar(20) NOT NULL,
  PRIMARY KEY (`usuario`,`amigo`),
  KEY `amigo` (`amigo`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Estructura de tabla para la tabla `tbl_categorias`
--
```

```
DROP TABLE IF EXISTS `tbl_categorias`;
CREATE TABLE IF NOT EXISTS `tbl_categorias` (
  `nombre` varchar(50) NOT NULL,
  PRIMARY KEY (`nombre`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Estructura de tabla para la tabla `tbl_categoria_videojuego`
--
```

```
DROP TABLE IF EXISTS `tbl_categoria_videojuego`;
CREATE TABLE IF NOT EXISTS `tbl_categoria_videojuego` (
  `categoria` varchar(50) NOT NULL,
  `videojuego` int UNSIGNED NOT NULL,
  PRIMARY KEY (`categoria`,`videojuego`),
  KEY `videojuego` (`videojuego`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Estructura de tabla para la tabla `tbl_comentarios`
--
```

```
DROP TABLE IF EXISTS `tbl_comentarios`;
CREATE TABLE IF NOT EXISTS `tbl_comentarios` (
  `codigo` int UNSIGNED NOT NULL,
  `fecha` date NOT NULL,
  `usuario` varchar(20) NOT NULL,
  `videojuego` int UNSIGNED NOT NULL,
  `comentario` text,
  PRIMARY KEY (`codigo`,`fecha`,`usuario`,`videojuego`),
  KEY `videojuego` (`videojuego`),
  KEY `usuario` (`usuario`)
```



```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
-----
```

```
--
```

```
-- Estructura de tabla para la tabla `tbl_desarrolladores`
```

```
--
```

```
DROP TABLE IF EXISTS `tbl_desarrolladores`;
```

```
CREATE TABLE IF NOT EXISTS `tbl_desarrolladores` (
```

```
  `nombre` varchar(20) NOT NULL,
```

```
  `tipo` enum('indie','empresa','') CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
```

```
  `contacto` varchar(200) NOT NULL,
```

```
  PRIMARY KEY (`nombre`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
-----
```

```
--
```

```
-- Estructura de tabla para la tabla `tbl_juegos_usuario`
```

```
--
```

```
DROP TABLE IF EXISTS `tbl_juegos_usuario`;
```

```
CREATE TABLE IF NOT EXISTS `tbl_juegos_usuario` (
```

```
  `usuario` varchar(20) NOT NULL,
```

```
  `videojuego` int UNSIGNED NOT NULL,
```

```
  `fechaCompra` date DEFAULT NULL,
```

```
  PRIMARY KEY (`usuario`,`videojuego`),
```

```
  KEY `videojuego` (`videojuego`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
-----
```

```
--
```

```
-- Estructura de tabla para la tabla `tbl_mensajes`
```

```
--
```

```
DROP TABLE IF EXISTS `tbl_mensajes`;
```

```
CREATE TABLE IF NOT EXISTS `tbl_mensajes` (
```

```
  `codigo` int UNSIGNED NOT NULL,
```

```
  `visto` tinyint(1) DEFAULT NULL,
```

```
  `fecha` date DEFAULT NULL,
```

```
  `emisor` varchar(20) NOT NULL,
```

```
  `receptor` varchar(20) NOT NULL,
```

```
  PRIMARY KEY (`codigo`,`emisor`,`receptor`),
```

```
  KEY `emisor` (`emisor`),
```

```
  KEY `receptor` (`receptor`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
-----
```



--

-- Estructura de tabla para la tabla `tbl_multimedia`

--

```
DROP TABLE IF EXISTS `tbl_multimedia`;  
CREATE TABLE IF NOT EXISTS `tbl_multimedia` (  
  `codigo` int UNSIGNED NOT NULL,  
  `videojuego` int UNSIGNED DEFAULT NULL,  
  `tipo` enum('videos','imagenes') DEFAULT NULL,  
  `ruta` text,  
  PRIMARY KEY (`codigo`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

--

-- Estructura de tabla para la tabla `tbl_usuarios`

--

```
DROP TABLE IF EXISTS `tbl_usuarios`;  
CREATE TABLE IF NOT EXISTS `tbl_usuarios` (  
  `nombre` varchar(20) NOT NULL,  
  `contraseña` varchar(50) NOT NULL,  
  `email` varchar(100) NOT NULL,  
  `saldo` int DEFAULT '0',  
  `fecha_inicio` date DEFAULT NULL,  
  PRIMARY KEY (`nombre`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

--

-- Estructura de tabla para la tabla `tbl_valoracion`

--

```
DROP TABLE IF EXISTS `tbl_valoracion`;  
CREATE TABLE IF NOT EXISTS `tbl_valoracion` (  
  `codigo` int UNSIGNED NOT NULL,  
  `fecha` date NOT NULL,  
  `usuario` varchar(20) NOT NULL,  
  `videojuego` int UNSIGNED NOT NULL,  
  `valoracion` int DEFAULT '0',  
  PRIMARY KEY (`codigo`,`fecha`,`usuario`,`videojuego`),  
  KEY `videojuego` (`videojuego`),  
  KEY `usuario` (`usuario`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```



--

-- Estructura de tabla para la tabla `tbl_videojuegos`

--

```
DROP TABLE IF EXISTS `tbl_videojuegos`;
CREATE TABLE IF NOT EXISTS `tbl_videojuegos` (
  `codigo` int UNSIGNED NOT NULL AUTO_INCREMENT,
  `rating` int NOT NULL,
  `nombre` varchar(100) NOT NULL,
  `descripcion` text NOT NULL,
  `precio` int NOT NULL,
  `desarrollador` varchar(20) NOT NULL,
  PRIMARY KEY (`codigo`),
  KEY `desarrollador` (`desarrollador`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

-- Filtros para la tabla `tbls_dlc`

--

```
ALTER TABLE `tbls_dlc`
  ADD CONSTRAINT `tbls_dlc_ibfk_1` FOREIGN KEY (`codigo`) REFERENCES `tbl_videojuegos`
  (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE;
```

--

-- Filtros para la tabla `tbl_amigos`

--

```
ALTER TABLE `tbl_amigos`
  ADD CONSTRAINT `tbl_amigos_ibfk_1` FOREIGN KEY (`usuario`) REFERENCES `tbl_usuarios`
  (`nombre`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `tbl_amigos_ibfk_2` FOREIGN KEY (`amigo`) REFERENCES `tbl_usuarios`
  (`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;
```

--

-- Filtros para la tabla `tbl_categorias`

--

```
ALTER TABLE `tbl_categorias`
  ADD CONSTRAINT `tbl_categorias_ibfk_1` FOREIGN KEY (`nombre`) REFERENCES
  `tbl_categoria_videojuego` (`categoria`) ON DELETE CASCADE ON UPDATE CASCADE;
```

--

-- Filtros para la tabla `tbl_categoria_videojuego`

--

```
ALTER TABLE `tbl_categoria_videojuego`
  ADD CONSTRAINT `tbl_categoria_videojuego_ibfk_1` FOREIGN KEY (`videojuego`) REFERENCES
  `tbl_videojuegos` (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE;
```

--

-- Filtros para la tabla `tbl_comentarios`

--

```
ALTER TABLE `tbl_comentarios`
  ADD CONSTRAINT `tbl_comentarios_ibfk_1` FOREIGN KEY (`videojuego`) REFERENCES
```



```
`tbl_videojuegos` (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `tbl_comentarios_ibfk_2` FOREIGN KEY (`usuario`) REFERENCES
`tbl_usuarios` (`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `tbl_desarrolladores`
--
ALTER TABLE `tbl_desarrolladores`
  ADD CONSTRAINT `tbl_desarrolladores_ibfk_1` FOREIGN KEY (`nombre`) REFERENCES
`tbl_usuarios` (`nombre`);

--
-- Filtros para la tabla `tbl_juegos_usuario`
--
ALTER TABLE `tbl_juegos_usuario`
  ADD CONSTRAINT `tbl_juegos_usuario_ibfk_1` FOREIGN KEY (`videojuego`) REFERENCES
`tbl_videojuegos` (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `tbl_juegos_usuario_ibfk_2` FOREIGN KEY (`usuario`) REFERENCES
`tbl_usuarios` (`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `tbl_mensajes`
--
ALTER TABLE `tbl_mensajes`
  ADD CONSTRAINT `tbl_mensajes_ibfk_1` FOREIGN KEY (`emisor`) REFERENCES `tbl_usuarios`
(`nombre`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `tbl_mensajes_ibfk_2` FOREIGN KEY (`receptor`) REFERENCES `tbl_usuarios`
(`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `tbl_multimedia`
--
ALTER TABLE `tbl_multimedia`
  ADD CONSTRAINT `tbl_multimedia_ibfk_1` FOREIGN KEY (`codigo`) REFERENCES
`tbl_videojuegos` (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `tbl_valoracion`
--
ALTER TABLE `tbl_valoracion`
  ADD CONSTRAINT `tbl_valoracion_ibfk_1` FOREIGN KEY (`videojuego`) REFERENCES
`tbl_videojuegos` (`codigo`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `tbl_valoracion_ibfk_2` FOREIGN KEY (`usuario`) REFERENCES `tbl_usuarios`
(`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `tbl_videojuegos`
--
ALTER TABLE `tbl_videojuegos`
  ADD CONSTRAINT `tbl_videojuegos_ibfk_1` FOREIGN KEY (`desarrollador`) REFERENCES
`tbl_desarrolladores` (`nombre`) ON DELETE CASCADE ON UPDATE CASCADE;
```



COMMIT;

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;  
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;  
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```