# IES NERVIÓN

# PLATAFORMA DE DISTRIBUCIÓN DE VIDEOJUEGOS

## Curso 2018-2019

# Diseño del Sistema de Información

# ÍNDICE

1. Diseño de la Arquitectura del Sistema
   1. Descripción general del entorno tecnológico del Sistema
   2. Catálogo de requisitos de Diseño
2. Modelo de clases de diseño
3. Modelo físico de datos
4. Plan de migración y carga inicial de datos
5. Plan de pruebas
6. Catálogo de requisitos de implantación

# Diseño de la Arquitectura del Sistema

## Descripción general de entorno tecnológico del Sistema

El Sistema guardará los datos que los usuarios introduzcan en la aplicación en una base de datos. Para acceder a estos datos, la aplicación de escritorio se conectará con un servidor usando SignalR para permanecer en contacto con el Sistema en tiempo real y durante todo el tiempo que la tenga abierta.

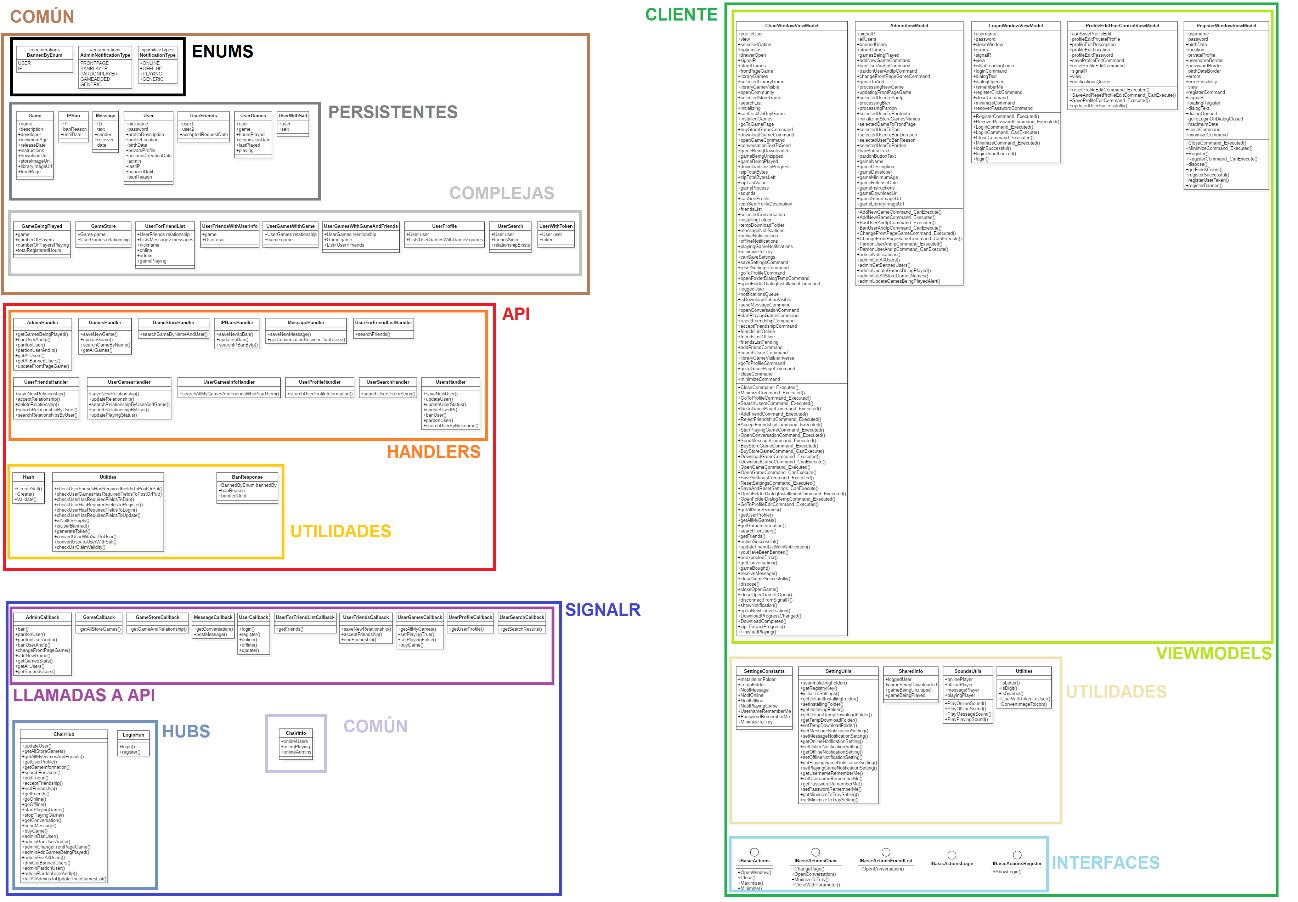
El servidor de SignalR se encargará de proporcionarle los datos a los clientes que hagan las peticiones. Para obtener los datos, SignalR hará uso de una API que servirá como intermediaria entre la base de datos y el servidor.

Para almacenar los juegos, se plantean varias posibilidades, todas de ellas gratuitas, y la más conveniente sería Google Drive. Sin embargo, gracias a la ayuda de un

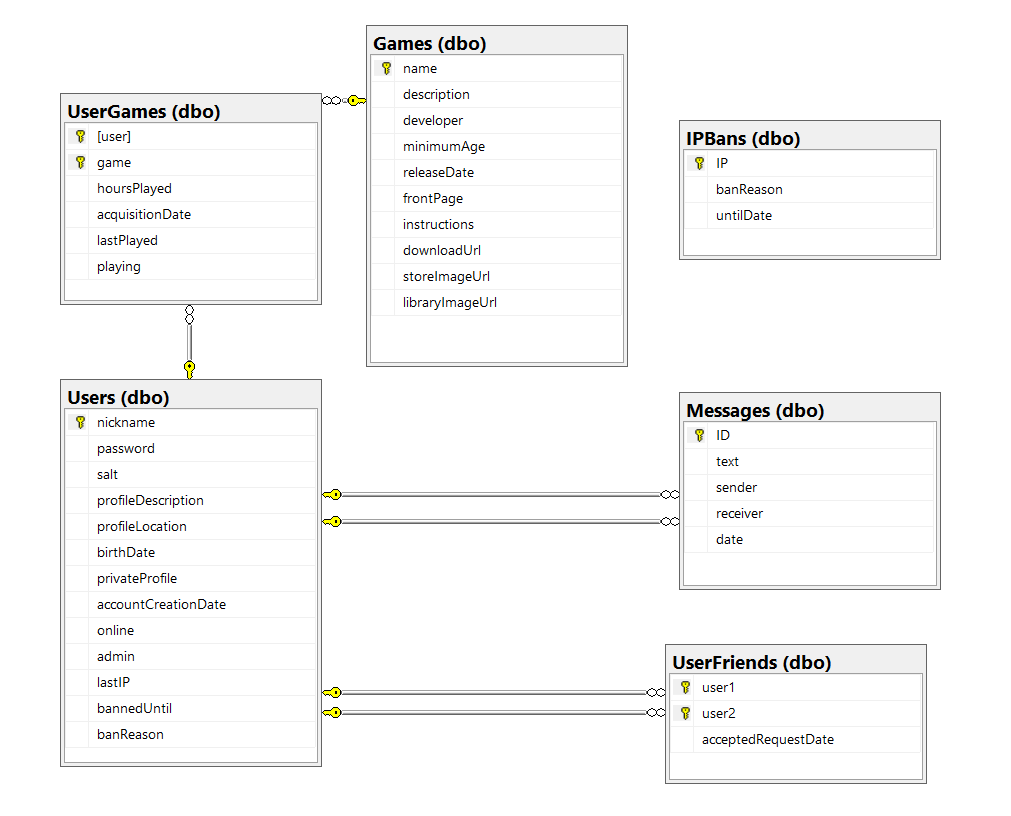
## Catálogo de requisitos de Diseño

|  |  |  |
| --- | --- | --- |
| **Código** | **Requisito** | **Descripción** |
| RQD1 | Lenguaje C# | Se usará C# para la programación del cliente y del servidor SignalR |
| RQD2 | Lenguaje SQL | Se usará SQL (MySQL) para la base de datos. |
| RQD3 | Lenguaje XAML | Se usará XAML en WPF para la creación de la UI. |

# Modelo de clases de diseño



# Modelo físico de datos



La aplicación usa una base de datos alojada en Azure y cuenta con 6 tablas:

* **Users**: Tabla que contiene la información de todos los usuarios
  + **Nickname**: Nombre del usuario
  + **Password**: Contraseña hasheada del usuario
  + **Salt**: Sal usada para hashear la contraseña
  + **ProfileDescription**: Descripción del usuario en su perfil
  + **ProfileLocation**: Localización del usuario en su perfil
  + **BirthDate**: Fecha de nacimiento
  + **PrivateProfile**: Si su perfil es privado o no
  + **AccountCreationDate**: Fecha en la que creó la cuenta
  + **Online**: Si está conectado o no
  + **Admin**: Si es un administrador o no
  + **LastIP**: Última IP con la que se conectó a la aplicación
  + **BannedUntil**: Fecha hasta la que se encuentra baneado
  + **BanReason**: Razón por la que se encuentra baneado
* **Messages**: Contiene todos los mensajes que se han enviado los usuarios
  + **ID**: Id única del mensaje
  + **Text**: Texto del mensaje
  + **Sender**: Usuario que envió el mensaje
  + **Receiver**: Usuario que recibe el mensaje
  + **Date**: Fecha en la que se envió el mensaje
* **UserFriends**: Tabla con las relaciones entre usuarios (amigos)
  + **User1**: El usuario que envió la petición de amistad
  + **User2**: El usuario que recibió la petición de amistad
  + **AcceptedRequestDate**: Fecha en la que se aceptó la solicitud
* **UserGames**: Tabla con las relaciones entre usuarios y los juegos (adquisiciones de juegos)
  + **User**: Usuario relacionado
  + **Game**: Juego relacionado
  + **HoursPlayed**: Horas que ha jugado el juego
  + **AcquisitionDate**: Fecha en la que adquirió el juego
  + **LastPlayed**: Fecha en la que jugó por última vez
  + **Playing**: Si el usuario está jugando ahora mismo a ese juego o no
* **Games**: Tabla con la información de los juegos en la tienda
  + **Name**: Nombre del juego
  + **Description**: Descripción del juego
  + **Developer**: Desarrollador del juego
  + **MinimumAge**: Edad mínima para jugarlo
  + **ReleaseDate**: Fecha en la que salió el juego
  + **FrontPage**: Si el juego está colocado como juego principal o no
  + **Instructions**: Instrucciones para correr el juego
  + **DownloadURL**: Link de descarga del juego
  + **StoreImageURL**: Link con el banner del juego
  + **LibraryImageURL**: Link con la imagen cuadrada del juego
* **IPBans**: Tabla con los baneos por IP
  + **IP**: IP Baneada
  + **BanReason**: Razón por la que la IP está baneada
  + **UntilDate**: Fecha hasta la que se encuentra baneada

# Plan de migración y carga inicial de datos

La aplicación no necesita de datos externos de ningún tipo salvo los videojuegos que se vayan añadiendo de forma progresiva a la plataforma, ya que esa es su finalidad. Al principio de la aplicación no habrá ningún videojuego ni ningún usuario, y a medida que los desarrolladores vayan pidiendo subir sus videojuegos y los usuarios se vayan registrando para jugarlos, la aplicación crecerá.

Por supuesto, para realizar las pruebas se necesitaran de datos ficticios y cuentas de usuario y administradores de prueba, pero al desplegar la aplicación para su uso real, no habrá datos iniciales.

# Plan de pruebas

Las pruebas se realizarán a medida que se vayan programando los diferentes módulos que conforman la aplicación. Una vez pasen todas las pruebas correctamente, se implantarán junto al resto de módulos, y una vez implantados se realizarán más pruebas en conjunto para comprobar que todo funciona de acuerdo a como debería.

|  |  |  |
| --- | --- | --- |
| **Código** | **Descripción de prueba** | **Tipo** |
| PR1 | La pantalla de login debe permitir viajar entre el registro y la aplicación principal | I |
| PR2 | Los botones custom de minimizar, cerrar y arrastrar funcionan correctamente | F |
| PR3 | Prueba de validación de datos válidos e inválidos introducidos en el login | F |
| PR4 | Prueba de validación de datos válidos e inválidos introducidos en el registro | F |
| PR5 | El registro crea una cuenta si no existe ese usuario | F |
| PR6 | El registro avisa al usuario de que la cuenta existe si el usuario existe | F |
| PR7 | El login avisa al usuario si su IP o su usuario se encuentra baneado (prohibido el acceso) | F |
| PR8 | El registro avisa al usuario si su IP está baneada | F |
| PR9 | Recuperar la contraseña realiza su función principal | F |
| PR10 | Si se deja marcado remember me y se vuelve a entrar a la aplicación, no hace falta volver a introducir los datos | F |
| PR11 | El perfil de un usuario queda escondido si lo tiene marcado como privado | F |
| PR12 | La imagen del usuario aparece correctamente en función de si es admin o no | I |
| PR13 | El perfil muestra los juegos que ha jugado el usuario recientemente | F |
| PR14 | Si se compra un juego en la tienda, queda asociado en la cuenta y aparece en la librería | F |
| PR15 | Si se pulsa en descargar el juego, éste se descargar y se instala en la ruta de descarga | F |
| PR16 | Si se pulsa en jugar, éste se abre | F |
| PR17 | Si se cierra el juego, se actualizan las horas que ha jugado | F |
| PR18 | Cuando un amigo abre un juego, éste cambia su estado en la lista de amigos | F |
| PR19 | Cuando un amigo cierra la aplicación somos avisados con notificación y sonido | F |
| PR20 | Cuando un amigo entra en la aplicación somos avisados con notificación y sonido | F |
| PR21 | Cuando un amigo nos envía un mensaje somos avisado con la pantalla de la conversación por delante y un sonido | F |
| PR22 | Si se pulsa en el menú hamburguesa, se despliega el menú | I |
| PR23 | Se puede navegar entre las opciones del menú | I |
| PR23 | Al buscar un usuario en Community, salen los resultados de los usuarios con lo que se ha buscado | F |
| PR24 | Al pulsar el botón de añadir amigo, el otro usuario es avisado y puede aceptarlo en su lista de amigos | F |
| PR25 | Al pulsar el botón de lista de amigos, ésta se abre con nuestros amigos | F |
| PR26 | Al introducir un juego nuevo en el apartado de Admin, se introduce en la tienda | F |
| PR27 | Al banear un usuario con o sin su IP desde el panel de Admins, el usuario queda baneado | F |
| PR28 | Al perdonar un usuario con o sin su IP desde el panel de Admins, el usuario queda perdonado | F |
| PR29 | Al cambiar el juego de la pantalla principal en el panel de Admin, cambia en la tienda | F |
| PR30 | Los datos de los juegos se actualizan en tiempo real en el panel de Admin | F |