informe 3 Arquitectura de Software

Nicolas Cerda

nicolas.cerda1@mail.udp.cl

Rafael Maturana

rafael.maturana@mail.udp.cl

Rayen Millaman

rayen.millaman@mail.udp.cl

Cristopher Kappes

cristopher.osorio_k@mail.udp.cl

Ignacio Marambio

ignacio.marambio_z@mail.udp.cl

1. Descripción

Sistema

El proyecto consiste en el desarrollo de un juego multijugador cooperativo. Este juego provee servicios para completar puzles y laberintos basados en cálculos matemáticos, acertijos, desarrollo de rompecabezas y lógica | código.

El sistema tendrá sus propios gestores de usuarios y sistemas de comunicación entre jugadores que serán manejadas por el mismo. Además, el juego implementará un sistema de progresión que permitirá a los usuarios desbloquear nuevos niveles y desafíos a medida que avanzan. La dificultad de los puzles y laberintos aumentará gradualmente para mantener el interés y la motivación de los jugadores.

Organización

Para este proyecto se trabajará para una desarrolladora de videojuegos de renombre, que encarga la creación del juego.

La compañía sugirió un juego con características educativas, manteniendo un estilo interesante y atractivo para los jugadores.

Según lo solicitado por la desarrolladora, el juego debe tener desafíos que estén orientados principalmente en la lógica, desarrollo de ejercicios matemáticos, acertijos de cultura general y desarrollar habilidades de carácter investigativo y de trabajo en equipo.

Debe tener características que lo hagan inmerso y mantengan la atención y la curiosidad del jugador.

Al estar basado en arquitectura SOA el juego estará compuesto por una serie de servicios autónomos comunicados mediante un bus de servicio.

Área

El área de trabajo es enfocada al desarrollo de juegos, con enfoque en los siguientes componentes:

- Resolución de puzzles, laberintos, acertijos y trivias.
- Colaboración entre jugadores, mediante sistemas de comunicación por voz y mensajería.
- Gestión de usuarios.
- Sistemas de progresión en niveles.
- Herramientas que se podrán desbloquear conforme avanza el juego.
- Foros donde los jugadores podrán compartir progreso y dar ideas para pasar niveles, así como informar de posibles bugs dentro del juego.

2. Objetivos

La historia del juego se centra en una entidad maligna de carácter desconocido, que estaría alojada en alguna parte de la red.

El objetivo del juego es que un grupo de jugadores al estilo de un antiguo RPG con laberintos resuelvan una serie de acertijos, descubran distintos secretos, consigan distintas herramientas mientras exploran y trabajen en equipo en post de destruir esta entidad.

3. Descripción del usuario

En el inicio, existirán 2 tipos de usuario.

Tipo de usuario	Descripción
Usuario	Jugadores regulares.
Administrador	Encargado de la gestión global del juego. Su principal función consiste en la configuración del sistema, estableciendo las reglas y parámetros.

4. Requerimientos

Requerimientos Funcionales:

Funcionalidades que tendrá el sistema:

N	Nombre	Tipo	Descripción	Prioridad
1	Gestor de usuario	Usuario	El usuario podrá iniciar sesión y autenticarse.	Alta
2	Información	Usuario	El usuario podrá consultar los datos de su cuenta.	Media
3	Nuevos Usuarios	Funcional Detallado	El sistema debe permitir que un nuevo usuario pueda crear una cuenta.	Alta
4	Autorización	De sistema	El sistema debe ser capaz de conceder los permisos correspondientes al usuario iniciado.	Media
5	Eliminación	De negocio	El administrador general podrá eliminar la cuenta de un usuario, así como aplicar distintos tipos de	Baja

N	Nombre	Tipo	Descripción	Prioridad
			penalización en caso de de tener una actitud hostil con otros jugadores.	
6	Sistema de progresión	Usuario	El usuario podrá desbloquee niveles al completar desafíos o puzzles previos.	Alta
7	Modo multijugador	Usuario	Los jugadores se podrán unir a partidas cooperativas con otros usuarios en tiempo real.	Alta
8	Panel de control del administrador	De Negocio	El administrador tendrá una vista única sobre los jugadores que le permita gestionar permisos de usuarios, niveles y/o configuraciones varias.	Media
9	Acceso a la base de datos	De Sistema	El sistema debe contener una servicio único al que se le realizaran consultas relacionadas a la base de datos	Media
10	Reintento de instancias.	Usuario	Los usuarios podrán reiniciar las instancias de juego que pierdan mientras cumplan ciertas condiciones.	Baja
11	Salida rápida	De Sistema / De Usuario	El usuario podrá salir rápidamente del juego sin tener mayores pérdidas de progreso, con la posibilidad de volver a la partida nuevamente en caso de haber sido por una falla de conexión.	Media
12	Guardado	De Sistema / De Usuario	El usuario podrá guardar el progreso que haya realizado y podrá eliminar aquellos que no le sean útiles.	Media
13	Guía Cooperativa	De Sistema / De Usuario	Existirá una página donde los jugadores podrán compartir los progresos que hayan tenido, relacionado a un sistema de foro	Baja
14	Generación de Contenido y/o Actividades.	De Sistema	Debe implementar algún sistema de generación de contenido, según lo dispuesto por el administrador.	Media
15	Emparejamiento	De Sistema	El sistema debe permitir emparejar automáticamente a los jugadores disponibles.	Media

Requerimientos No funcionales:

El sistema en su conjunto debe ser capaz cumplir con los siguientes requisitos mínimos no funcionales **(que en principio serán solo atributos de calidad)**

N	Tipo	Descripción	Prioridad
1	Seguridad	Debe ser capaz de validar la autenticación de acceso a la plataforma, implementando control de acceso basado en roles, asegurando que solo los usuarios autorizados tengan acceso a funcionalidades específicas según sus roles.	Alta
2	Verificabilidad	Debe tener un apartado interno donde se grafique los módulos existentes, sus funcionalidades y si este se encuentra operativo, en tiempo real.	Alta
3	Portabilidad	Debe estar implementado en Docker para su optimo transporte y despliegue en diferentes entornos sin cambios significativos	Alta
4	Soportabilidad	Debe mantener un registro de errores centralizado que documente los módulos con fallas, sus dependencias y los servicios impactados, facilitando la identificación de patrones de error y la resolución de problemas.	Media
5	Confiabilidad	Se debe mantener el progreso del juego independiente de algún fallo en el sistema.	Alta

5. Componentes

El sistema incluye los siguientes clientes y servicios:

Clientes

- **Cliente administrador**

Este cliente corresponde al personal encargado de supervisar y administrar el funcionamiento general del juego, así como de gestionar a los usuarios en el sistema.

- **Cliente usuario**

Cliente correspondiente a un usuario regular, que utiliza gran variedad de servicios para acceder al juego preferente juegos, ver su progreso personal y conectarse con otros jugadores.

Servicios

1. **Servicio gestor de usuario**

   Permite iniciar sesión en el sistema y validar las credenciales del usuario.

   RF1, RNF1, RNF3

   - IF IN - Login - Nombre Usuario - Contraseña
   - IF OUT - Login - Respuesta

2. **Servicio de Información del usuario**

   El servicio permite a los usuarios consultar y actualizar sus datos personales.

RF2

- IF IN - infou- IDusuario
- IF OUT -infou - Respuesta (datos personales)

*Requiere autenticación previa mediante el **Gestor de Usuario***

3. **Servicio de Nuevos Usuarios**

   Este servicio permite registrar nuevas cuentas en el sistema. Se conecta al bus y comunica los datos ingresados por el usuario para almacenarlos en la base de datos.

   RF3,RNF3

   - IF IN - regis - Nombre Usuario -Contraseña - Mail
   - IF OUT - regis - Respuesta

   *Conecta con **Gestor de Usuario** para validar los datos*

4. **Servicio de Autorización**

   El propósito de este servicio consiste en conceder los permisos correspondientes al tipo de usuario (administrador o regular). Se activa inmediatamente después de la autenticación, validando el rol y proporcionando acceso a las funcionalidades correspondientes.

   RF4, RNF1

   - IF IN - permi- Nombre Usuario - Contraseña
   - IF OUT - permi- Respuesta

   *Depende de la respuesta del **Gestor de Usuario** para iniciar sesión y autorizar roles.*

5. **Servicio de Eliminación y penalización**

   Este servicio permite que el administrador elimine cuentas de usuarios o aplique penalizaciones por comportamiento inadecuado. Recibe la solicitud del administrador a través del bus, consulta la información del usuario y ejecuta la eliminación o penalización.

   RF5

   - IF IN - elimi - ID usuario
   - IF OUT - elimi- Respuesta

6. **Servicio de Sistema de progresión**

   El propósito de este servicio es gestionar el avance del jugador en el juego. Para ello el servicio recibe el estado actual del jugador y, al cumplir las condiciones requeridas, actualiza su progreso en el **Servicio de Guardado** e invoca al **Servicio de Generación de Contenido** para proporcionar desafíos adicionales.

   RF6

   - IF IN - progr- ID usuario -Nivel actual - Desafíos completados

- IF OUT - progr- Respuesta

7. **Servicio Modo multijugador**

   Este servicio coordina la interacción entre usuarios activos y mantiene la sincronización del estado del juego. Solicita al **Servicio de Emparejamiento** asignar jugadores a partidas y usa el servicio de **Mensajería** para facilitar la comunicación entre los participantes.

   RF7

   - IF IN - multi- ID usuario - ID nivel
   - IF OUT - multi - Respuesta - Estado de la partida

8. **Servicio de Panel de control del administrador**

   Este servicio proporciona al administrador una interfaz para gestionar usuarios, permisos, niveles y configuraciones del sistema. Se conecta al bus para consultar datos en tiempo real y aplicar cambios.

   RF8

   - IF IN - iadmi- ID usuario - Solicitud
   - IF OUT - iadmi- Vista - Respuesta

   *Invoca **Gestor de Usuario**, **Información** y **Eliminación** para mostrar estadísticas y gestionar permisos*

9. **Servicio de Acceso a la base de datos**

   Este servicio le da soporte a los demás servicios permitiendo realizar operaciones de escritura y lectura en la base de datos. El servicio asume que las consultas ya han sido autorizadas, por lo cuál su única función es conectarse a la base de datos, recibir una consulta, procesarla y responder la consulta.

   RF9

   - IF IN - condb - Consulta sql
   - IF OUT - condb- Respuesta

10. **Servicio de Reinicio de instancias.**
    Este servicio interactúa con el
    Servicio de Guardado para recuperar el estado anterior de la instancia de juego. También verifica las condiciones de reintento a través del sistema de reglas predefinidas.

    RF10, RNF5

    - IF IN - reini- ID usuario - ID nivel
    - IF OUT - reini- Respuesta

11. **Servicio de Salida rápida**

    Este servicio permite al usuario salir rápidamente del juego sin perder su progreso. El servicio interactúa con el **Servicio de Guardado** para almacenar el progreso actual antes de que el usuario salga del juego.

    RF11

    - IF IN - salir- ID usuario - ID nivel
    - IF OUT - salir- Respuesta

12. **Servicio de Guardado**

    Este servicio es responsable de guardar el progreso del jugador en el sistema. El servicio interactúa con **Progresión del Jugador**, **Salida Rápida**, y **Reintento de Instancias** para almacenar y recuperar el estado del juego.

    RF12

    - IF IN - savee- ID usuario - ID nivel
    - IF OUT - savee- Respuesta

13. **Servicio de Guía Cooperativa**

    El servicio permite a los jugadores publicar y consultar guías cooperativas a través de una interfaz de comunicación. Los datos se almacenan y se comparten entre los jugadores conectados.

    RF13

    - IF IN - foros- Nombre de usuario
    - IF OUT - foros- Respuesta

14. **Servicio de Generación de Contenido y/o Actividades.**

    El servicio genera contenido entregado el administrador.

    RF14, RNF3

    - IF IN - conte- Type - Parameters
    - IF OUT - conte- OK - Response

15. **Servicio de Emparejamiento**

    El servicio permite la asignación automática de jugadores. Para ello se el servicio recibe las solicitudes de jugadores que desean unirse a partidas multijugador y los empareja con otros jugadores según los parámetros (nivel o tipo de partida). Utiliza el **Modo Multijugador** para gestionar la partida una vez que el emparejamiento está completo.

RF15

- IF IN - busca- Usuarios disponibles - Parámetros de emparejamiento
- IF OUT - busca- Respuesta

6. Modelo de datos

El sistema cuenta con un modelo de datos relacional implementado en SQL, compuesto por cinco entidades principales: **Usuarios**, **Niveles, Progresión, Publicaciones y Trivias**

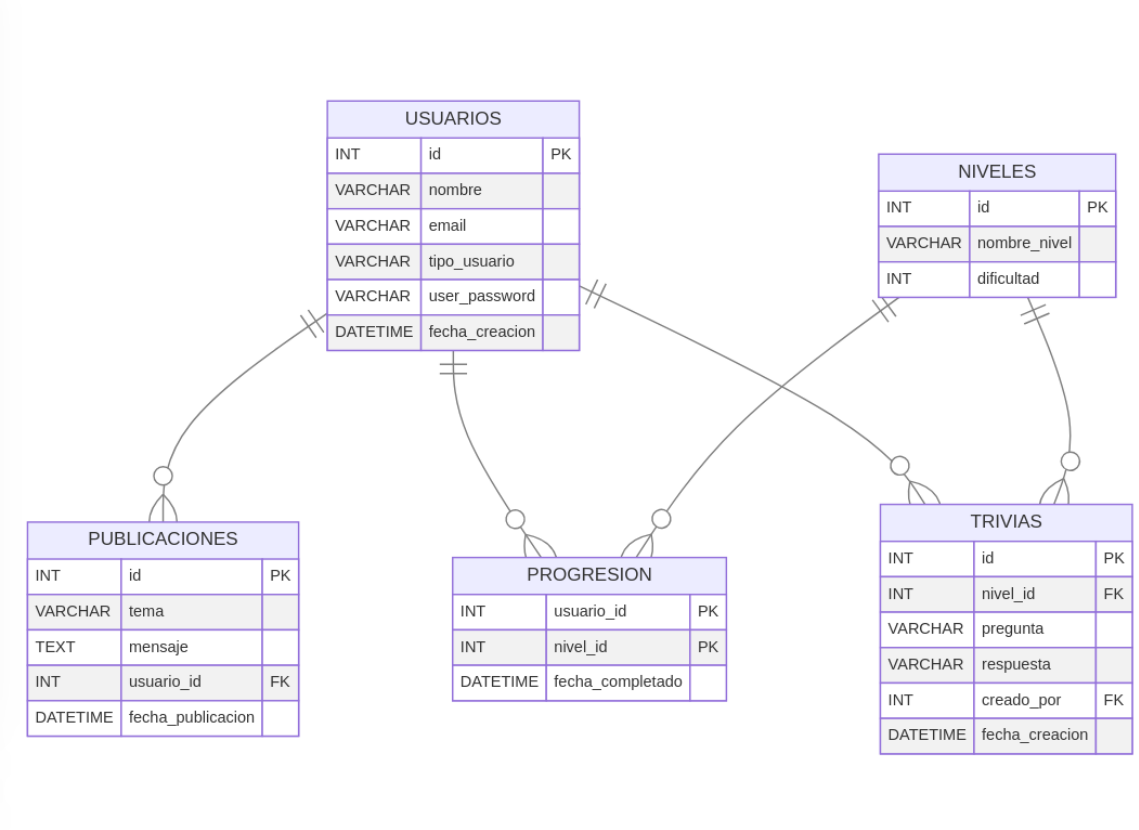
Usuarios almacena la información de los usuarios registrados. Cada usuario es identificado de manera única mediante el campo id como clave primaria. Además, contiene los atributos nombre,email, contraseña, tipo de usuario(admin o regular)y la fecha de creación, que registra la fecha y hora en que el usuario fue creado en el sistema.

Niveles define los diferentes niveles disponibles en el sistema. Cada nivel tiene una clave primaria única representada por el campo id, los atributos nombre del nivel y la dificultad, que indica el nivel de dificultad a través de un número entero.

Progresión actúa como una tabla intermedia que relaciona usuarios con niveles. Específicamente, conecta a los usuarios con los niveles que han completado, proporcionando un registro del progreso de cada usuario. Esta tabla contiene dos claves foráneas (para evitar duplicación): id del usuario, que referencia al usuario asociado, y id del nivel, que referencia al nivel correspondiente. Además, incluye un campo fecha de completado, que registra la fecha y hora en que el usuario completó el nivel.

Publicaciones cada publicación tiene una clave primaria única, id, y contiene un campo tema, que representa la categoría de la publicación, y un campo mensaje, que almacena el contenido del mensaje. La publicación también está asociada a un usuario mediante el campo del id usuaruario como clave foránea, y cuenta con un campo fecha de publicación que registra el momento en que se creó. Las publicaciones están diseñadas para eliminarse automáticamente si el usuario asociado es eliminado.

Trivias está diseñada para almacenar preguntas y respuestas asociadas con los niveles. Esta tabla incluye un id único como clave primaria y un campo del nivel id que referencia al nivel correspondiente en la tabla de **Niveles**. Cada trivia tiene los atributos de pregunta, respuesta y el identificador del administrador que lo creo.



Relaciones:

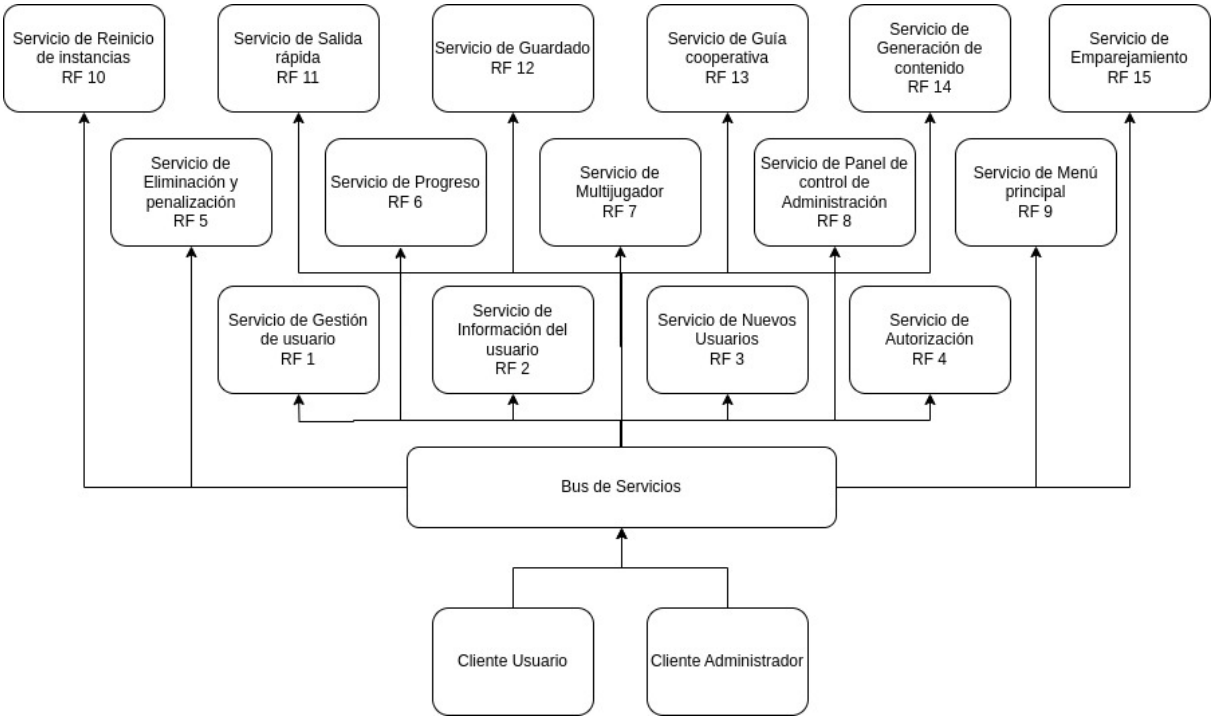
- **Relación entre usuarios y progresión**:
  - Un usuario puede tener múltiples registros en la tabla de progresión, lo que permite rastrear su avance en diferentes niveles.
  - Clave foránea: `usuario_id` en `progresion` referencia a `id` en `usuarios`.
- **Relación entre niveles y progresión**:
  - Un nivel puede estar asociado con múltiples usuarios que lo hayan completado, almacenando dicha información en la tabla de progresión.
  - Clave foránea: `nivel_id` en `progresion` referencia a `id` en `niveles`.
- **Relación entre usuarios y publicaciones**:
  - Un usuario puede generar múltiples publicaciones en la tabla `publicaciones`. Estas publicaciones se eliminan automáticamente si el usuario es eliminado.
  - Clave foránea: `usuario_id` en `publicaciones` referencia a `id` en `usuarios`.
- **Relación entre niveles y trivias**:
  - Un nivel puede tener múltiples preguntas y respuestas asociadas en la tabla `trivias`.
  - Clave foránea: `nivel_id` en `trivias` referencia a `id` en `niveles`.
- **Relación entre usuarios y trivias**:
  - Un administrador puede crear múltiples preguntas y respuestas en la tabla `trivias`. Si un administrador es eliminado, el campo `creado_por` se establece en `NULL`.

- Clave foránea: `creado_por` en `trivias` referencia a `id` en `usuarios`.

- **Relación entre progresión y usuarios**:
  - La tabla progresión conecta a los usuarios con los niveles que han completado.
  - Combinación de las claves foráneas `usuario_id` y `nivel_id` en `progresion` referencia a las tablas `usuarios` y `niveles`.
- **Relación entre progresión y niveles**:
  - La tabla progresión también registra los detalles del progreso de un usuario en un nivel, como la fecha de finalización.
  - Clave foránea: `nivel_id` en `progresion` referencia a `id` en `niveles`.

```sql
CREATE TABLE usuarios (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(25),
    email VARCHAR(25) UNIQUE,
    Tipo_usuario VARCHAR(25),
    user_password VARCHAR(25),
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE niveles (
    id SERIAL PRIMARY KEY,
    nombre_nivel VARCHAR(25),
    dificultad INT
);

CREATE TABLE progresion (
    usuario_id INT REFERENCES usuarios(id),
    nivel_id INT REFERENCES niveles(id),
    fecha_completado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (usuario_id, nivel_id)
);

CREATE TABLE publicaciones (
    id SERIAL PRIMARY KEY,
    tema VARCHAR(10) NOT NULL,
    mensaje TEXT NOT NULL,
    usuario_id INT REFERENCES usuarios(id) ON DELETE CASCADE, -- Si se elimina un usuario, se eliminan sus publicaciones
    fecha_publicacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE trivias (
    id SERIAL PRIMARY KEY,
    nivel_id INT REFERENCES niveles(id) ON DELETE CASCADE, -- Relación con el nivel
    pregunta TEXT NOT NULL, -- La pregunta de la trivia
    respuesta TEXT NOT NULL, -- La respuesta correcta
    creado_por INT REFERENCES usuarios(id) ON DELETE SET NULL, -- Administrador que creó la trivia
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP -- Fecha de creación
);
```

7. Diagrama de arquitectura SOA



Equipo de trabajo

Integrante	Tarea Asignada	Descripción
Nicolás Cerda	Desarrollo del Gestor de Usuarios	Implementar los servicios de autenticación, nuevos usuarios, y autorización (Servicios de usuario).
	Desarrollo del Servicio de Eliminación	Crear el servicio para gestión de penalización y eliminación de cuentas por el administrador.
	Cliente Administrador	Desarrollar la interfaz del cliente administrador, incluyendo gestión de permisos y configuraciones.
Rafael Maturana	Sistema de Progresión	Encargado de diseñar el servicio de progresión y el guardado del progreso del jugador.
	Desarrollo del Bus de Servicios	Implementación del bus de comunicación que conecta a todos los servicios.
	Cliente Usuario	Desarrollar la interfaz para los usuarios regulares, incluyendo visualización de progreso.
Rayen Millaman	Sistema de Información del Usuario	Crear el servicio para consultas y actualizaciones de datos de usuario.
	Panel de Control del Administrador	Desarrollar el servicio del panel administrativo para gestión de jugadores y configuraciones.
	Menú Principal	Implementar el servicio del menú principal para navegación inicial y selección de opciones.
Cristopher Kappes	Servicio de Modo Multijugador	Desarrollar el servicio para gestionar las partidas multijugador, incluyendo comunicación en tiempo real.
	Servicio de Generación de Contenido	Crear el servicio para generar contenido dinámico y desafíos en los niveles.

	Servicio de Guía Cooperativa	Implementar el servicio para compartir guías y progresos entre los jugadores.
Ignacio Marambio	Servicio de Emparejamiento	Desarrollar el servicio de emparejamiento automático en el modo multijugador.
	Servicio de Reinicio de Instancias	Implementar el servicio para permitir el reinicio de niveles bajo ciertas condiciones.
	Servicio de Salida Rápida	Crear el servicio que permite al usuario salir rápidamente sin perder el progreso.

Flujo del jugador

Con el fin de estructurar cada etapa del juego de manera lógica y fluida a través de los niveles, tanto en modo individual como multijugador, se procede a realizar el siguiente flujo.

1. **Inicio del Juego**: Cliente se conecta y muestra el menú principal.
2. **Autenticación**: Validación del usuario a través del servicio de autenticación.
3. **Selección de Nivel**: El jugador elige un nivel o continúa desde el último progreso guardado.
4. **Iniciar Nivel**: El jugador interactúa con el contenedor del nivel seleccionado y envía la respuesta para avanzar.
5. **Progresión**: El bus de servicios permite al jugador avanzar al siguiente nivel tras completar uno.
6. **Modo Multijugador** (opcional): Los jugadores se sincronizan en una partida cooperativa.
7. **Guardar y Cargar Progreso**: El jugador puede guardar y reanudar su progreso.
8. **Salida del Juego**: El jugador sale del juego con su progreso guardado.

Se procede a detallar cada punto.

1. Inicio del Juego - Menú Principal

- **Cliente se conecta al sistema**: El cliente ejecuta el juego desde su contenedor.

  El cliente muestra el menú principal, donde el jugador puede:

  - Iniciar sesión (Servicio de Autenticación).
  - Crear una nueva cuenta (Servicio de Autenticación).
  - Ver los datos de su cuenta (Servicio de Gestión de Usuario).
  - Configurar opciones del juego o personalizar el menú.
  - Elegir entre los modos de juego disponibles (por ejemplo, individual o multijugador).

2. Autenticación del Jugador

- **Iniciar Sesión (RF1)**: El jugador ingresa sus credenciales en el menú principal.
- **Servicio de Autenticación (RF1)**: Las credenciales se envían al servicio de autenticación en el servidor, que valida si el usuario existe y si la contraseña es correcta.

- **Verificación**: Si las credenciales son válidas, el sistema devuelve una confirmación y el jugador puede continuar al siguiente paso. Si no, se le pide que lo intente nuevamente o que recupere su cuenta.

3. Selección de Nivel o Modo de Juego

- **Menú de Selección de Niveles**: Después de iniciar sesión, el jugador ve un menú con los niveles disponibles (individual o multijugador).
- **Progreso Guardado (RF6, RF12)**: Si el jugador ha progresado en niveles anteriores, el sistema muestra desde qué nivel puede continuar.
- **Selección de Nivel o Continuación**: El jugador selecciona el nivel donde desea comenzar o continuar. La elección se envía al bus de servicios.

4. Iniciar Nivel

- **Nivel 1 o Nivel Seleccionado** : El jugador es conectado al contenedor del nivel correspondiente.
- **Servicio de Validación de Nivel**: El contenedor del nivel gestiona el desafío o puzzle y espera la respuesta (por ejemplo, la contraseña).
  - Si el nivel es completado con éxito, el contenedor del nivel devuelve una señal de éxito.
  - Si el jugador falla, el contenedor proporciona retroalimentación (RF11).

5. Progresión a Niveles Superiores

- **Validación de Progreso (RF6)**: Cuando el jugador completa el nivel, el bus de servicios actualiza el estado de su progreso en el servidor.
- **Acceso al Siguiente Nivel**: El jugador recibe acceso al siguiente nivel, y el flujo se repite.
- **Reintento de Niveles (RF10)**: Si el jugador falla repetidamente, puede optar por reiniciar el nivel bajo ciertas condiciones, gestionadas por el servicio del nivel correspondiente.

6. Modo Multijugador

- **Unirse a Partida Multijugador (RF7)**: Si el jugador opta por el modo multijugador, el sistema asigna al jugador a una partida cooperativa.
- **Coordinación de Jugadores**: El bus de servicios gestiona la sincronización entre los jugadores, permitiendo que trabajen juntos para resolver los desafíos del nivel.
- **Progreso Compartido**: El progreso del equipo se registra y se actualiza en el servidor a medida que completan niveles de manera conjunta.

7. Guardar y Cargar Progreso

- **Guardar Progreso (RF12)**: En cualquier momento, el jugador puede pausar el juego y guardar su progreso en el servicio de persistencia.
- **Reanudar Juego**: Si el jugador desea volver al juego, puede cargar su progreso desde el menú principal y continuar desde el último nivel guardado.

8. Salida del Juego

- **Salida Rápida (RF11)**: El jugador puede optar por salir rápidamente del juego a través del menú de pausa. El sistema guarda el progreso automáticamente en puntos críticos para que pueda retomarlo más tarde.
- **Confirmación de Salida**: El jugador recibe una confirmación de que el progreso ha sido guardado y el contenedor del cliente se cierra.

Últimas consideraciones

En principio, todos los servicios mencionados anteriormente estarán almacenados en Docker. Esto con el objetivo de maximizar la velocidad de desarrollo de este proyecto.

Junto con esto también se utilizaran templates externos e interfases como Godot para facilitar la creación del mismo, sin tener que invertir tantos recursos y energía en crear servicios completos desde cero.

Cabe mencionar también que este juego cae dentro de lo que es la categoría independiente, por consiguiente, cosas como el apartado gráfico, niveles de realismo, complejidad de mundo abierto, entre otras características, serán muy inferiores en comparación a lo que sería un juego de categoría A.

Se espera que al final del desarrollo, estas características faltantes sean sopesadas por otras como las que se mencionaron a lo largo de este informe, con perspectiva de que se cumplan las expectativas de un videojuego que pueda ser agradable para los jugadores que disfrutan de este genero.

8. Documentación implementación de servicios

A modo de reutilización de código, se implementó el archivo `comunicacion_bus.py`. El cual proporciona una solución genérica que permite a los servicios interactuar con el bus de mensajes, permitiendo registrarse con su prefijo, recibir mensajes y enviarlos de manera eficiente.

Además para los mensajes de enviado o recibido, la información de los datos se almacenará en formato JSON.

A continuación se detalla cada servicio implementado entre la comunicación sel servicio y el bus, con su link de documentación.

1. **Servicio gestor de usuario**

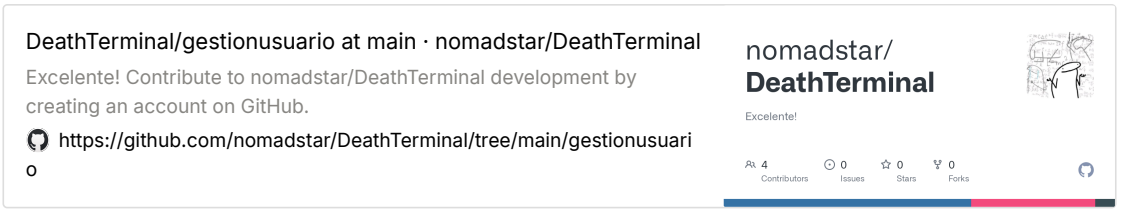
   El siguiente servicio denominado **“login”** permite iniciar sesión en el sistema y validar las credenciales del usuario. Para ello el servicio recibe los parámetros del cliente; nombre y la contraseña que serán validadas en servicio de consultas a la base de datos para poder validar las credenciales.

   RF1, RNF1, RNF3

   - IF IN - login - nombre, contraseña
   - IF OUT - login - Respuesta

   ```
   Iniciando servicio de gestión de usuarios...
   Mensaje registro: 00010sinitlogin
   Mensaje recibido: sinitOKlogin
   Mensaje recibido: 00048login {'nombre': 'holo', 'user_password': '1234'}
   mensaje enviado: 00087condb{"sql": "SELECT * FROM usuarios WHERE nombre = 'holo' AND user_password = '1234'"}
   Mensaje recibido: 00175condb OK {'data': [{'id': 3, 'nombre': 'holo', 'email': 'holo@example.com', 'tipo_usuario': 'usuario', 'user_password': '1234',
   'fecha_creacion': '2024-12-04 20:03:15.006329'}]}
   mensaje enviado: 00042login{"message": "credenciales correctas"}
   ```

   *Requiere del servicio de **consultas a la base de datos***



2. **Servicio de Información del usuario**

   El servicio con prefijo **"infou"** es el encargado de procesar y notificar al usuario; sus datos personales. Recibe a través del servicio de cliente el id asociado a consultar.

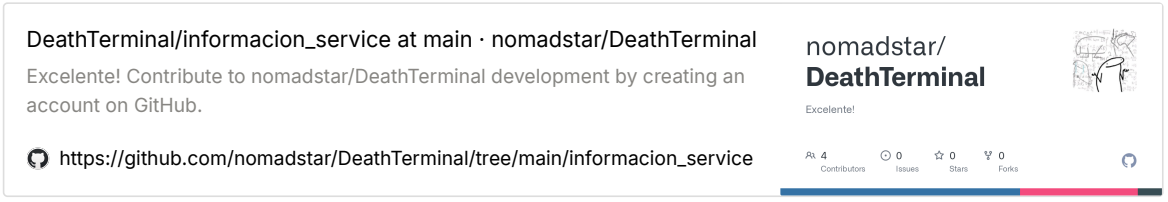
   RF2

   - IF IN -infou- IDusuario
   - IF OUT -infou - Respuesta (datos personales)

```
Iniciando servicio de gestión de información del usuario...
Mensaje registro: 00010sinitinfou
Mensaje recibido: sinitOKinfou
Mensaje recibido: 00014infou {'id': 3}
mensaje enviado: 00436condb{"sql": "SELECT usuarios.id, usuarios.nombre, usuarios.email, usuarios.Tipo_usuario, niveles.nombre_nivel, niveles.dif
d\n               FROM usuarios\n               LEFT JOIN progresion ON usuarios.id = progresion.usuario_id\n               LEFT JOIN niveles
gresion.nivel_id = niveles.id\n               WHERE usuarios.id = '3'\n               ORDER BY progresion.fecha_completado DESC\n
T 1"}
Mensaje recibido: 00142condb OK {'data': [{'id': 3, 'nombre': 'hola', 'email': 'hola@example.com', 'tipo_usuario': 'admin', 'nombre_nivel': None,
cultad': None}]}
mensaje enviado: 00118infou{"id": 3, "nombre": "hola", "email": "hola@example.com", "tipo usuario": null, "nivel": null, "dificultad": null}
```

Ln 31, Col 37 Spaces: 4

*Requiere autenticación previa mediante el **Gestor de Usuario,** mientras que las consultas son realizadas por el **servicio de consultas a la base de datos***



3. **Servicio de Nuevos Usuarios**

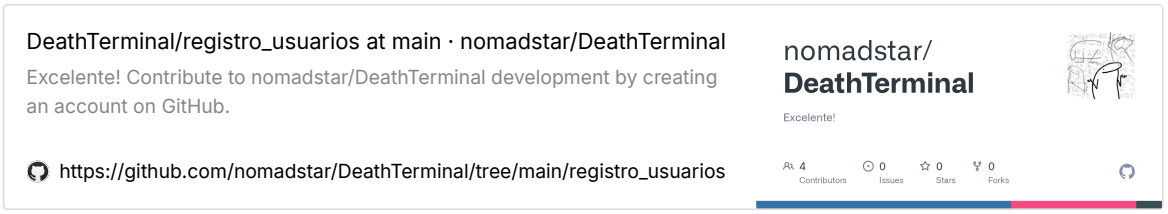
   Este servicio denominado **"regis"** facilita el registro de nuevas cuentas en el sistema mediante la conexión con el servicio acceso a la base de datos. Recibe como parámetros el nombre, contraseña y correo electrónico, y registra al usuario como un perfil estándar, procesando los datos proporcionados a través de la interfaz del cliente.

   RF3,RNF3

   - IF IN - regis - Nombre Usuario ,Contraseña , Mail
   - IF OUT - regis - Respuesta

```
Iniciando servicio de registro de usuarios...
Mensaje registro: 00010sinitregis
Mensaje recibido: sinitOKregis
Mensaje recibido: 00077regis {'nombre': 'juan', 'user_password': '1234', 'email': 'juan@example.com'}
mensaje enviado: 00060condb{"sql": "SELECT * FROM usuarios WHERE nombre = 'juan'"}
Mensaje recibido: 00019condb OK {'data': []}
mensaje enviado: 00136condb{"sql": "INSERT INTO usuarios (nombre, email, user_password, Tipo_usuario) VALUES ('juan', 'juan@example.com', '1234', 'usua
rio')"}
Mensaje recibido: 00040condb OK {'data': {'consulta exitosa': 1}}
mensaje enviado: 00051regis{"message": "Usuario registrado exitosamente"}
```

*Requiere del servicio de **consultas a la base de datos***



4. **Servicio de Autorización**

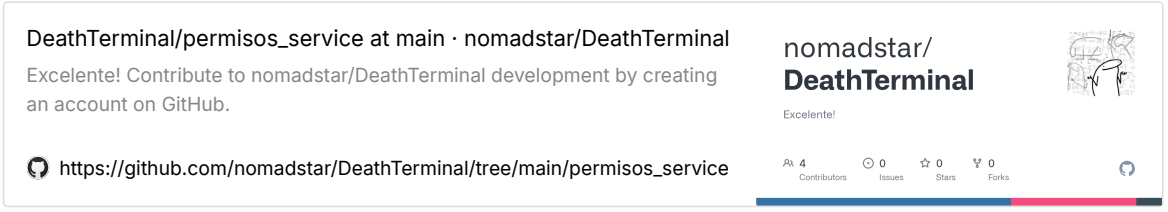
   Servicio denominado **“permi”** revisa si el usuario ingresado es de carácter administrador o regular. Se inicia inmediatamente después de la autenticación, quien valida el rol mediante consulta a la base de datos para proporcionar el acceso correspondiente.

   RF4, RNF1

   - IF IN - permi- ID usuario
   - IF OUT - permi- Respuesta

```
Iniciando servicio de gestión de permisos...
Mensaje registro: 00010sinitpermi
Mensaje recibido: sinitOKpermi
Mensaje recibido: 00014permi {'id': 3}
mensaje enviado: 00062condb{"sql": "SELECT Tipo_usuario FROM usuarios WHERE id = 3"}
Mensaje recibido: 00046condb OK {'data': [{'tipo_usuario': 'usuario'}]}
mensaje enviado: 00038permi{"message": "permisos denegados"}
```

   *Requiere del servicio de **consultas a la base de datos y del gestor de usuario***



5. **Servicio de Eliminación y penalización**

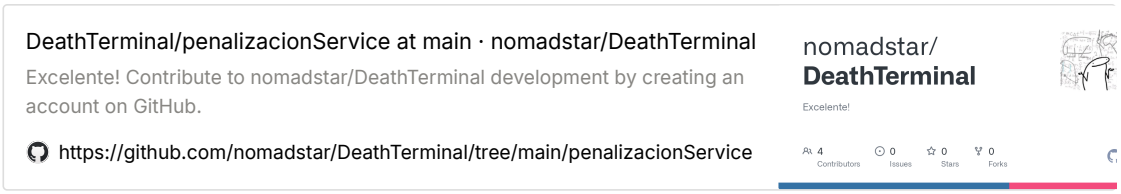
   Este servicio denominado con el prefijo **“elimi”** permite que el administrador elimine cuentas de usuarios. Recibe la solicitud del administrador a través del bus sobre el ID del usuario, consulta la información del usuario con el servicio de base de datos y ejecuta la eliminación o penalización.

   RF5

   - IF IN - elimi - ID usuario
   - IF OUT - elimi- Respuesta

```
Iniciando servicio de penalización...
Mensaje registro: 00010sinitelimi
Mensaje recibido: sinitOKelimi
Mensaje recibido: 00021elimi {'user_id': '2'}
mensaje enviado: 00049condb{"sql": "DELETE FROM usuarios WHERE id = 2"}
Mensaje recibido: 00040condb OK {'data': {'consulta exitosa': 1}}
```

   *Requiere del servicio de **consultas a la base de datos y del servicio de autorización** (solo aplicable para administradores)*



6. **Servicio de Sistema de progresión**

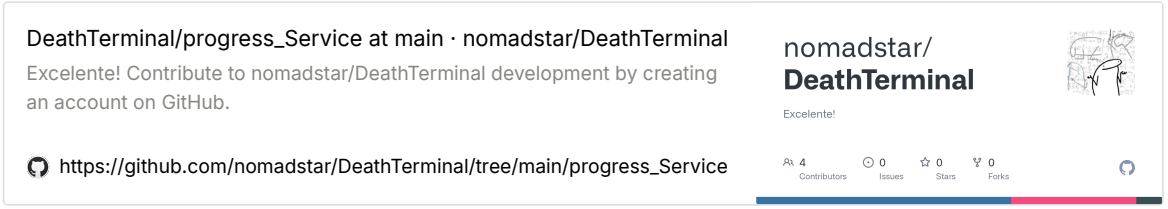
   Servicio denominado **"progr"** su propósito es gestionar el avance del jugador en el juego o entregar el nivel actual en que se encuentra el jugador. Recibe el id del usuario y un identificador para relacionar si el usuario esta pidiendo el nivel actual o desea actualizarlo.

   RF6

   - IF IN - progr- ID usuario -Nivel actual - Desafíos completados
   - IF OUT - progr- Respuesta

```
Iniciando servicio de progreso...
Mensaje registro: 00010sinitprogr
Mensaje recibido: sinitOKprogr
Mensaje recibido: 00027progr {'id': 2, 'opcion': 1}
mensaje enviado: 00310condb{"sql": "SELECT niveles.nombre_nivel, niveles.id\n                    FROM progresion\n                    JOIN niveles ON progresion.nive
l_id = niveles.id\n                    WHERE progresion.usuario_id = '2'\n                ORDER BY progresion.fecha_completado DESC\n                    LIMIT 1
"}
Mensaje recibido: 00055condb OK {'data': [{'nombre_nivel': 'Nivel 1', 'id': 1}]}
mensaje enviado: 00077progr{"message": "Nivel entregado", "nivel_actual": "Nivel 1", "id_nivel": 1}
```

   *Requiere del servicio de **consultas a la base de datos** para la actualización o busqueda del nivel actual.*



7. **Servicio Modo multijugador**

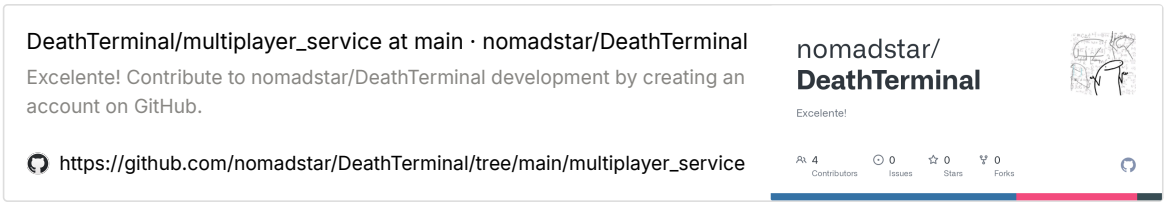
   Este servicio denominado **"multi"** coordina la interacción entre usuarios. Solicita al **Servicio de Emparejamiento** asignar jugadores a partidas, donde en el caso que completen el nivel ambos usuarios deberán pasar al siguiente nivel.

   RF7

   - IF IN - multi- ID usuarios - ID nivel
   - IF OUT - multi - Respuesta - Estado de la partida

```
Iniciando servicio de gestión de multiplayer...
Mensaje registro: 00010sinitmulti
Mensaje recibido: sinitOKmulti
Mensaje recibido: 00057multi {'id_usuario': 3, 'id_encontrado': 2, 'id_nivel': 1}
mensaje enviado: 00193condb{"sql": "UPDATE progresion\n                SET estado = 'finalizado', fecha_completado = CURRENT_TIMESTAMP\n
 WHERE nivel_id = 1\n                AND usuario_id IN (3, 2);"}
Mensaje recibido: 00040condb OK {'data': {'consulta exitosa': 2}}
mensaje enviado: 00078multi{"message": "Modificaci\u00f3n exitosa", "data": {"consulta exitosa": 2}}
```

   *Requiere del servicio de **consultas a la base de datos** para la actualización en ambos usuarios.*



8. **Servicio de Panel de control del administrador**

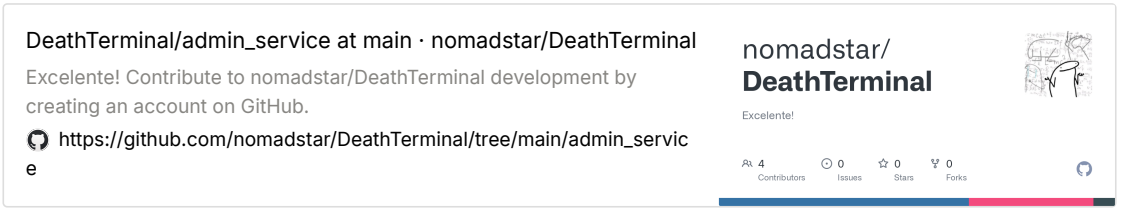
   Este servicio denominado **“iadmi”** recibe consultas a realizar proporcionadas por cliente administrador, es utilizado para revisar los usuarios y los niveles inscritos. Ademas de permitir eliminar o actualizar niveles o usuarios.

   RF8

   - IF IN - iadmi- ID usuario - Solicitud
   - IF OUT - iadmi- Vista - Respuesta

```
Mensaje registro: 00010sinitiadmi
Mensaje recibido: sinitOKiadmi
Mensaje recibido: 00020iadmi {'opcion': '3'}
mensaje enviado: 00037condb{"sql": "SELECT * FROM niveles"}
Mensaje recibido: 00127condb OK {'data': [{'id': 1, 'nombre_nivel': 'Nivel 1', 'dificultad': 1}, {'id': 2, 'nombre_nivel': 'Nivel 2', 'dificultad
}
mensaje enviado: 00157iadmi{"message": "Datos encontrados", "data": [{"id": 1, "nombre_nivel": "Nivel 1", "dificultad": 1}, {"id": 2, "nombre_nive
Nivel 2", "dificultad": 2}]}
Mensaje recibido: 00020iadmi {'opcion': '1'}
mensaje enviado: 00038condb{"sql": "SELECT * FROM usuarios"}
Mensaje recibido: 00497condb OK {'data': [{'id': 1, 'nombre': 'Admin', 'email': 'admin@example.com', 'tipo_usuario': None, 'user_password': 'passw
3', 'fecha_creacion': '2024-12-07 03:29:11.921441'}, {'id': 2, 'nombre': 'Jugador1', 'email': 'jugador1@example.com', 'tipo_usuario': None, 'user_
ord': '12345', 'fecha_creacion': '2024-12-07 03:29:11.921441'}, {'id': 3, 'nombre': 'hola', 'email': 'hola@example.com', 'tipo_usuario': 'admin',
_password': '1234', 'fecha_creacion': '2024-12-07 03:29:11.925329'}]}
mensaje enviado: 00527iadmi{"message": "Datos encontrados", "data": [{"id": 1, "nombre": "Admin", "email": "admin@example.com", "tipo_usuario": nu
user_password": "password123", "fecha_creacion": "2024-12-07 03:29:11.921441"}, {"id": 2, "nombre": "Jugador1", "email": "jugador1@example.com",
usuario": null, "user_password": "12345", "fecha_creacion": "2024-12-07 03:29:11.921441"}, {"id": 3, "nombre": "hola", "email": "hola@example.com
po_usuario": "admin", "user_password": "1234", "fecha_creacion": "2024-12-07 03:29:11.925329"}]}
Mensaje recibido: 00031iadmi {'opcion': '2', 'id': '2'}
mensaje enviado: 00021elimi{"user_id": "2"}
```

   *Servicio que utiliza la comunicación constante con el **servidor de base de datos y el servicio de Eliminación de los usuarios***



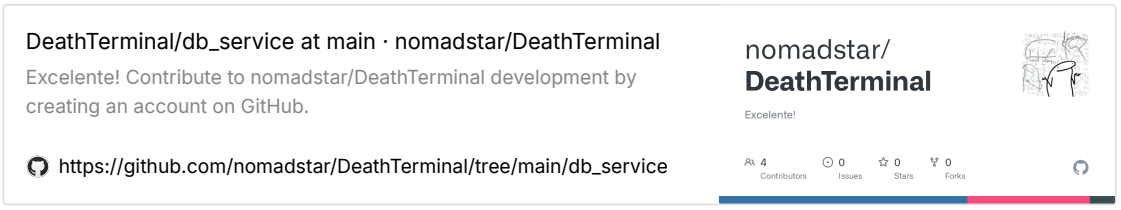
9. **Servicio de Consultas a la base de datos**

   Este servicio denominado **“condb”**, recibe solicitudes para consultar a la base de datos y devuelve las respuestas de tales consultas.

   RF9

   - IF IN - condb- Consultasql
   - IF OUT - condb- Respuesta

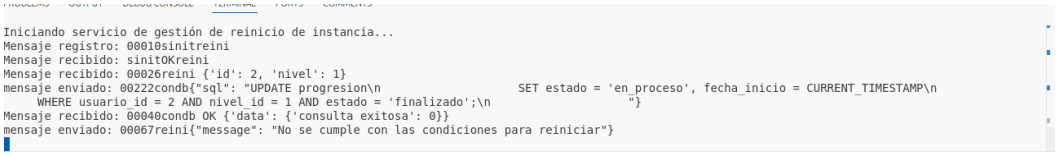
```
Iniciando servicio de gestión de consultas db...
Mensaje registro: 00010sinitcondb
Mensaje recibido: sinitOKcondb
Mensaje recibido: 00063condb {'sql': "SELECT * FROM usuarios WHERE nombre = 'juanito'"}
mensaje enviado: 00017condb{"data": []}
Mensaje recibido: 00142condb {'sql': "INSERT INTO usuarios (nombre, email, user_password, Tipo_usuario) VALUES ('juanito', 'juanito@example.com', '1234
', 'usuario')"}
mensaje enviado: 00038condb{"data": {"consulta exitosa": 1}}
```

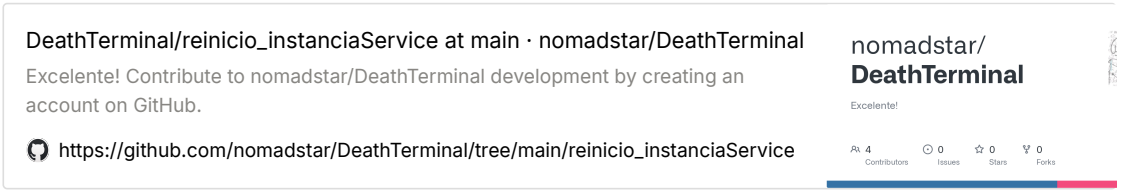


10. **Servicio de Reinicio de instancias.**
    Este servicio denominado
    "reini" que verifica si la partida actual esta en estado "finalizado" para poder reiniciar el nivel y volver a jugar dicho nivel.

    RF10, RNF5

    - IF IN - reini- ID usuario - ID nivel
    - IF OUT - reini- Respuesta



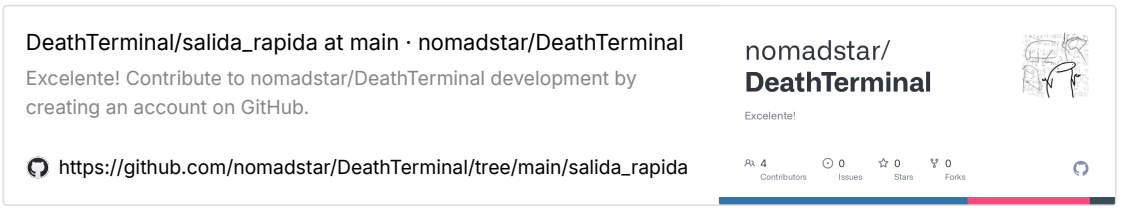


11. **Servicio de Salida rápida**

    Este servicio denominado **"salir"** permite al usuario salir rápidamente del juego sin perder su progreso. El servicio interactúa con el de **Progreso** para identificar el nivel en que se encuentra y el **Servicio de Guardado** para almacenar el progreso actual antes de que el usuario salga del juego.

    RF11

    - IF IN - salir- ID usuario
    - IF OUT - salir- Respuesta



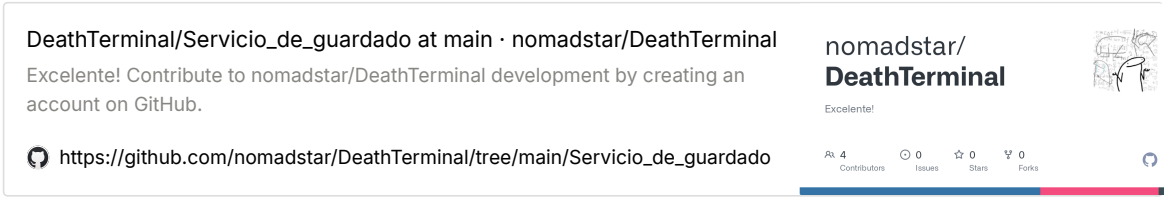
12. **Servicio de Guardado**

    Este servicio denominado **"savee"** es responsable de guardar el progreso del jugador en el sistema. El servicio recibe el mensaje de **Progresión del Jugador** para almacenar el estado del juego de un jugador en particular en la base de datos:

    RF12

    - IF IN - savee- ID usuario - ID nivel
    - IF OUT - savee- Respuesta

```
Executing task: docker logs --tail 1000 -f 661fb98c9850216ee8b8bb6e7ce5e1cad72b7f32241fbb231a9084c0c840b1cc

Iniciando servicio de guardado de instancias...
Mensaje registro: 00010sinitsavee
Mensaje recibido: sinitOKsavee
Mensaje recibido: 00037savee {'id_usuario': 1, 'id_nivel': 1}
mensaje enviado: 00246condb{"sql": "\n        INSERT INTO progresion (usuario_id, nivel_id, fecha_completado)\n        VALUES (1, 1, CURRENT_TIMESTAMP)
\n        ON CONFLICT (usuario_id, nivel_id)\n        DO UPDATE SET fecha_completado = CURRENT_TIMESTAMP;\n        "}
Mensaje recibido: 00040condb OK {'data': {'consulta exitosa': 1}}
mensaje enviado: 00078savee{"message": "Modificaci\u00f3n exitosa", "data": {"consulta exitosa": 1}}
```



    *Requiere del servicio de **consultas a la base de datos** para la actualización del nivel.*

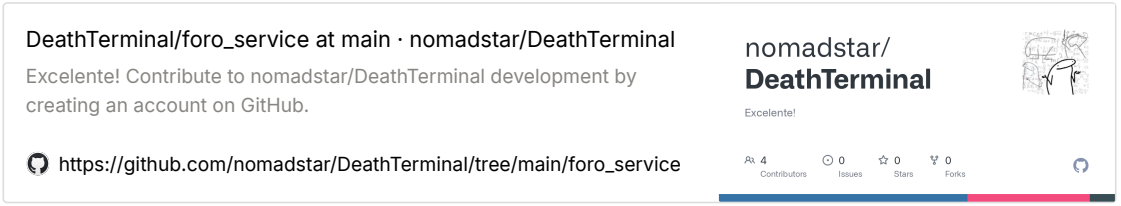
13. **Servicio de Guía Cooperativa**

    El servicio denominado **"foros"**el cual permite a los jugadores publicar guías cooperativas a través de una interfaz de comunicación. Los datos se almacenan y son de libre acceso a los demás jugadores. La entrega de información al servicio depende de las actividades que se quieran realizar tales como acceder a las publicaciones de un usuario en particular, ver todas las publicaciones o realizar una publicación.

    RF13

    - IF IN - foros- Datos
    - IF OUT - foros- Respuesta

```
Iniciando servicio de foro...
Mensaje registro: 00010sinitforos
Mensaje recibido: sinitOKforos
Mensaje recibido: 00027foros {'opcion': 1, 'id': 1}
mensaje enviado: 00064condb{"sql": "SELECT * FROM publicaciones WHERE usuario_id = 1"}
Mensaje recibido: 00019condb OK {'data': []}
mensaje enviado: 00040foros{"message": "No hay publicaciones"}
Mensaje recibido: 00074foros {'opcion': 2, 'id': 1, 'tema': 'salud', 'mensaje': 'hola como estas'}
mensaje enviado: 00108condb{"sql": "INSERT INTO publicaciones (usuario_id, tema, mensaje) VALUES (1, 'salud', 'hola como estas')"}
Mensaje recibido: 00040condb OK {'data': {'consulta exitosa': 1}}
mensaje enviado: 00078foros{"message": "Modificaci\u00f3n exitosa", "data": {"consulta exitosa": 1}}
Mensaje recibido: 00027foros {'opcion': 1, 'id': 1}
mensaje enviado: 00064condb{"sql": "SELECT * FROM publicaciones WHERE usuario_id = 1"}
Mensaje recibido: 00143condb OK {'data': [{'id': 1, 'tema': 'salud', 'mensaje': 'hola como estas', 'usuario_id': 1, 'fecha_publicacion': '2024-12-07 22
:44:52.581335'}]}
mensaje enviado: 00172foros{"message": "Consulta exitosa", "data": [{"id": 1, "tema": "salud", "mensaje": "hola como estas", "usuario_id": 1, "fecha_pu
blicacion": "2024-12-07 22:44:52.581335"}]}
```

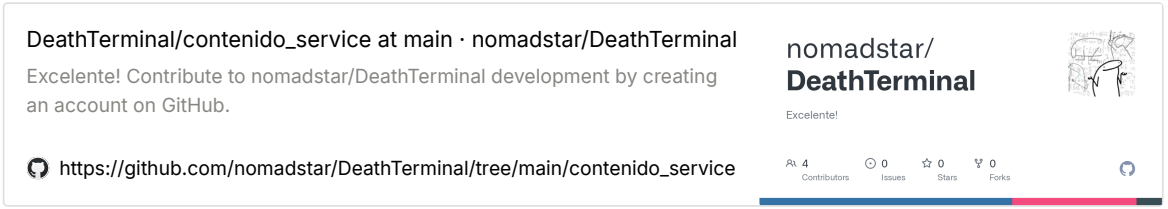


14. **Servicio de Generación de Contenido y/o Actividades.**

    El servicio denominado **"conte"** es utilizado para generar el contenido del nivel, usando los parametros proporcionados por el usuario administrador, en este caso recibo los parametros de pregunta con su respuesta que sera almacenado en la tabla de niveles.

    RF14, RNF3

- IF IN - conte- Parámetros del nivel
- IF OUT - conte- Respuesta

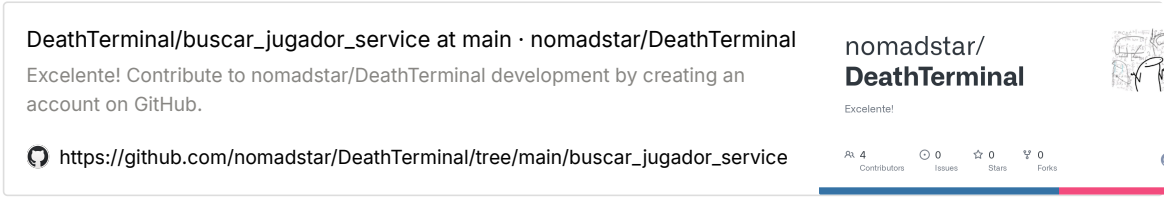


15. **Servicio de Emparejamiento**

    El servicio con el prefijo **"busca"** permite la asignación automática de jugadores. Para ello se el servicio recibe el id del usuario y el nivel en que se encuentra, el servicio buscara un usuario que se encuentre en el mismo nivel para su asignación conjunta.

    RF15

- IF IN - busca- Id usuario - Id nivel
- IF OUT - busca- Respuesta



9. Análisis

El desarrollo bajo la arquitectura SOA permitió un entendimiento más claro sobre la estructura y necesidades del sistema, durante la realización del proyecto facilitó el entendimiento tanto de la

implementación como la adaptabilidad del sistema, asegurando que los servicios y los clientes pudieran interactuar eficientemente a través de una base de datos.

Por su parte, el diseño del sistema, definido por requisitos funcionales y no funcionales, sirvió como una guía esencial para mantener la coherencia y la alineación del desarrollo., poniendo en especial énfasis en la creación de servicios independientes, lo que simplificó significativamente su implementación en el código. Asimismo, se logró una mejor comprensión del flujo de comunicación entre los servidores mediante el uso del bus, asegurando que estos procesos estuvieran alineados con los objetivos planteados y fortaleciendo la estructura general del sistema.

Dificultades encontradas

Una de las principales dificultades encontradas durante el desarrollo fue la comunicación entre los servicios, específicamente en el manejo de solicitudes y respuestas. Entender cómo interactúan los servicios y sus clientes supuso un proceso reiterativo de ensayo y error, donde se realizaron múltiples ajustes a nivel de código para garantizar que los servicios respondieran de manera coherente y cumplieran con sus funciones de transmisión y recepción de información.

Además, se planteó la implementación de una interfaz gráfica con el objetivo de hacer el sistema más dinámico y atractivo. Sin embargo, debido a ciertos inconvenientes técnicos y de compatibilidad, se decidió ajustar el enfoque para minimizar las interferencias en el funcionamiento del sistema principal, priorizando su funcionamiento.

Por otro lado, la decisión de utilizar el formato JSON fue realizada con el fin de simplificar la codificación y decodificación de la información, resolviendo problemas que surgieron durante el desarrollo. Finalmente, el avance del proyecto se llevó a cabo mediante un enfoque iterativo, basado en pruebas pequeñas y progresivas.

10. Anexos

Además, puede encontrar todos los códigos utilizados en este proyecto en el siguiente repositorio de GitHub:

https://github.com/nomadstar/DeathTerminal