











**Proyecto Final**

Jiayi Chen  
DAM2B

# DESASTRE HUMANO

or HUMAN DISASTER

## INDEX

- 01**  Antecedentes y objetivos del proyecto
- 02**  Arquitectura del sistema
- 03**  Mecánica de juego y tecnología principales
- 04**  Diseño de aplicaciones comunitarias
- 05**  Implementación de doble backend
- 06**  Diseño de bases de datos
- 07**  Pruebas
- 08**  Resumen y perspectivas futuras

## Antecedentes y objetivos del proyecto

De "Crear juegos" a "Construir un ecosistema"

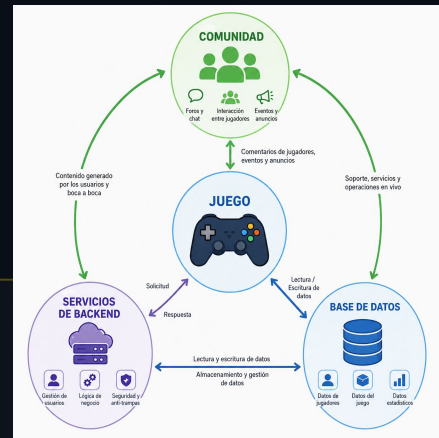
### Antecedentes

- > Proyectos anteriores
- > Ecosistema
- > Mercado



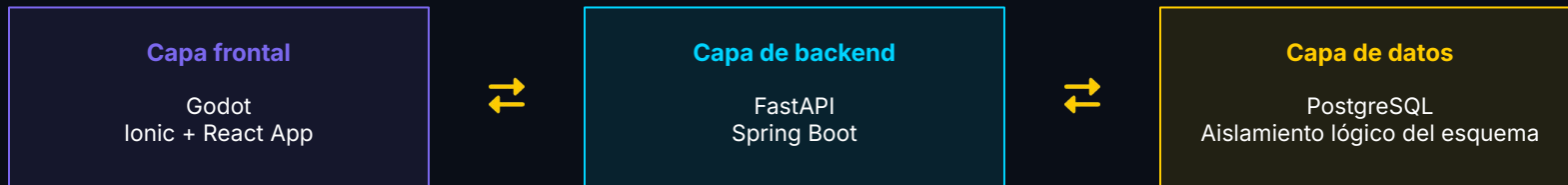
### Objetivos del proyecto

- ✓ Juegos con múltiples mecánicas y guardado en la nube.
- ✓ La aplicación StarShip DH Community (Ionic + React) es compatible con Android.
- ✓ API de guardado de juegos FastAPI + API de la comunidad Spring Boot.
- ✓ Una instancia unificada de PostgreSQL, dividida en dominios de negocio mediante esquemas.



## Arquitectura general del sistema

Desacoplamiento de doble backend + PostgreSQL compartido

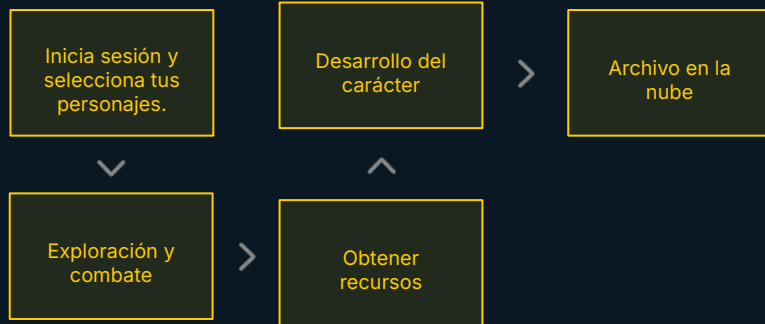


# DESASTRE HUMANO · Jugabilidad principal

Una mezcla de diferentes mecanismos y sistemas

## 🔄 Ciclo de juego principal

Sistema de crecimiento de ciclo cerrado impulsado por los recursos:



**Basado en datos:** Las estadísticas, los datos de armas y los recursos de habilidades están alineados con el JSON del sistema de backend.

**Login**

Enter your username to log in

Enter your password to log in

Remember Me

LOGIN

Don't have an account yet? [Register](#) [Forget the password](#)

**Tables (36)**

- > achievements
- > character\_achievements
- > character\_currencies
- > character\_gene\_modules
- > character\_gene\_slots
- > character\_genes

**FishMan**  
Berserk Mutant  
Level 1 / 30  
10 / 225

Max Health 100 / 100

Defense 25

Attack Bonus 300

Evasion 5.0%

More Attribute

# DESASTRE HUMANO · Jugabilidad principal

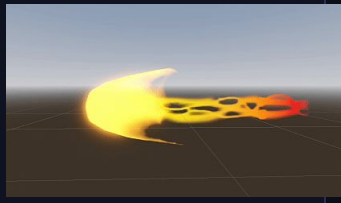
Una mezcla de diferentes mecanismos y sistemas

## ⌚ Sistema de desarrollo de personajes

↑ **Nivel de experiencia:** Gana experiencia a través de batallas para mejorar tus atributos básicos.

⚡ **Avance de sincronización:** Consume recursos específicos para superar el límite de velocidad de sincronización y desbloquear el límite superior.

🧠 **Módulos genéticos:** Fragmentos genéticos conectables que proporcionan bonificaciones de habilidad activas/pasivas.



## 🛡️ Implementación del sistema de combate

🎯 Sistema de armas

🤖 IA enemiga

📊 Información sobre impactos

✍️ Sistema de habilidades

📄 Aspectos técnicos destacados:

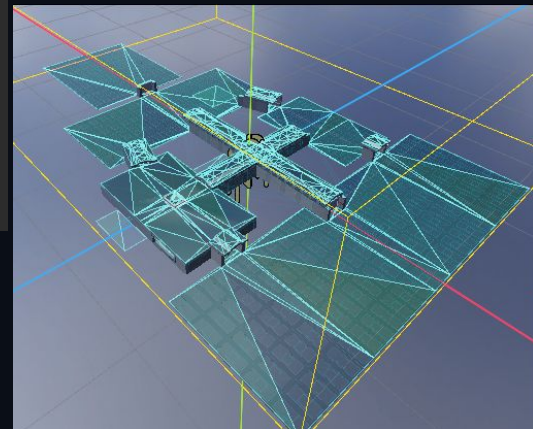
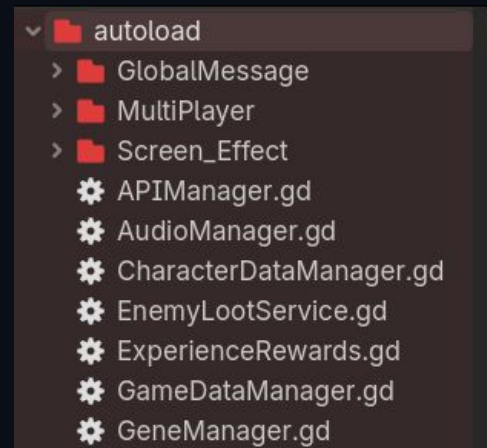
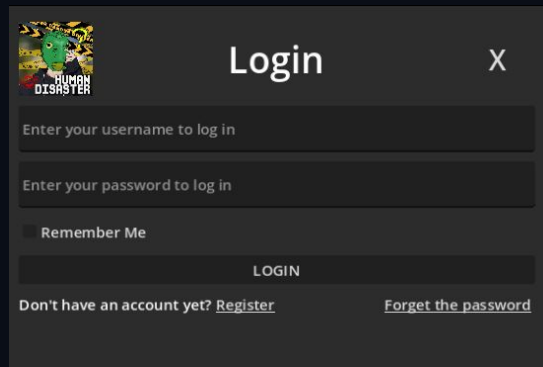
**Sistema unificado de gestión de daños y atributos:**

- **AttackData** como único transportista de daños.
- **Stats.take\_damage()** es el único punto de entrada para calcular la salud y los atributos.



## GODOT 4.6 · Implementación técnica

Diseño de arquitectura de capa de servicios y basado en recursos



# STARSHIP DH

La plataforma social del juego permite a los usuarios comunicarse.

Ionic

React

Capacitor

TypeScript

Tailwind CSS

## 👤 Funciones sociales básicas

Publicar contenido, interactuar con los usuarios, recibir notificaciones en tiempo real y gestionar perfiles de usuario.

## 🔄 Sincronización de datos

Sincroniza los datos del jugador con el backend de Spring mediante JWT.

## 🛡️ Sistema de certificación unificado

Comparte tu cuenta de auth.users con el juego para lograr una transición fluida entre "jugar" e "interactuar con la comunidad".

### Aspectos destacados:

"Rompe las barreras de los videojuegos y habilita la interacción social y la sincronización de datos en cualquier momento y lugar."

🔒 **Lectura anónima:** detalles de la publicación, lista de comentarios

🔑 **Iniciar sesión:** Publicar, comentar, editar perfil, etc.

Siguiendo **Recomendado** Eventos

🔍 Buscar contenidos de interés... 3 resultados

**Anuncio** El jugador Baixuan obtuvo el personaje Catwoman.

**Fijado** El mantenimiento comenzará mañana a las 12:00.

**Fijado** Eventos | Un nuevo evento está a punto de comenzar.

**S** **SomeOne**  
Hace 1 h

**¡Qué escena tan genial!**

¡Acabo de descubrir un laboratorio oculto en el sector 7! Las paredes estaban cubiertas de datos cifrados y había terminales aún encendidos. Parece que alguien...



👍 30 💬 3 👁 2000 🗣 12

**D** **DH Oficial - ES**

🏠 INICIO 🧰 HERRAMIENTAS ➕ NOTIFICACIONES 👤 YO

# FASTAPI · Backend del juego

Interfaz de datos estáticos y guardado de partidas de alto rendimiento

## Interfaz de autenticación

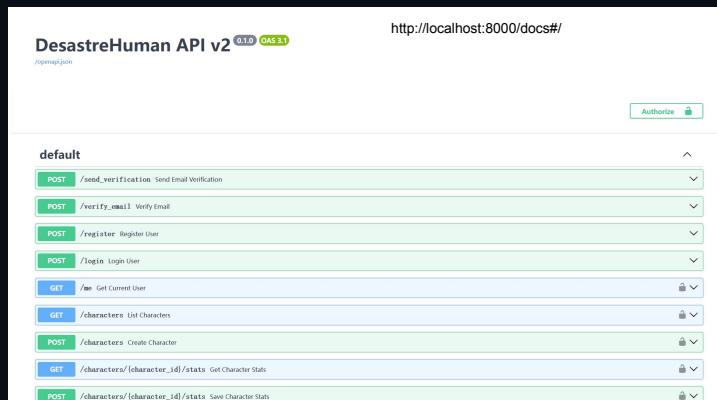
```
POST /register
POST /login
GET /me (JWT Bearer)
```

## Guardado del personaje

```
GET /characters/{id}/stats
PUT /characters/{id}/inventory
PUT /characters/{id}/scene_state
```

## Datos estáticos

```
GET /game-data/items
GET /game-data/weapons
GET /game-data/skills
```



DesastreHuman API v2 0.1.0 0MS 1.1 <http://localhost:8000/docs/#/>

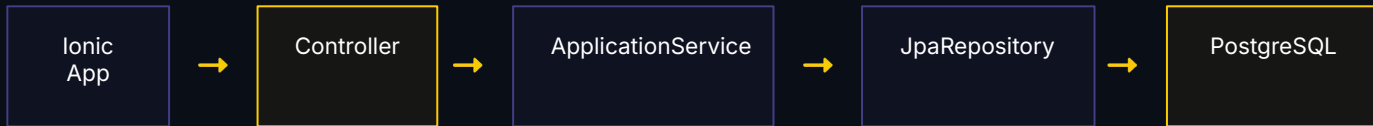
[Authorize](#)

default ^

- POST /send\_verification Send Email Verification
- POST /verify\_email Verify Email
- POST /register Register User
- POST /login Login User
- GET /me Get Current User
- GET /characters List Characters
- POST /characters Create Character
- GET /characters/{character\_id}/stats Get Character Stats
- POST /characters/{character\_id}/stats Save Character Stats

# SPRING BOOT · Backend del APP

Lógica empresarial comunitaria y control de seguridad



- application
  - dto
  - service
- infrastructure
  - auth**
  - config
- persistence
  - entity
  - mapper
  - repository
- security
- interfaces.controller

```
code: 0
data: { hasMore: false, items: (5)[...], nextCursor: null }
message: "success"
```

# POSTGRESQL

Separación de responsabilidades y aislamiento lógico mediante schemas

## auth

Cuenta de usuario

## game

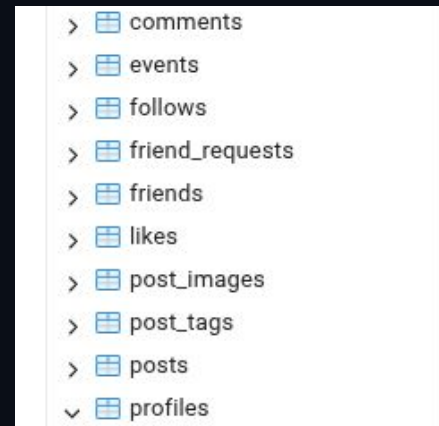
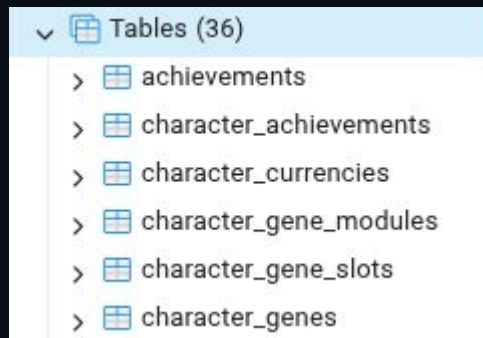
Archivos de guardado de personajes,  
plantillas estáticas

## community

Publicaciones, comentarios, etc.

## chat

Notificaciones, Reservas de sesión



# PRUEBAS

Pruebas automatizadas y garantía de calidad

## Pruebas de API de backend

- ✓ pytest
- ✓ MockMvc
- ✓ JWT Validation

## Pruebas de lógica del cliente

- ✓ api\_test.tscn
- ✓ run\_unit\_tests.gd
- ✓ Manual Scenarios

## Verificación de la integración del sistema

- ✓ Sync Logic  
Verificar la coherencia del archivo
- ✓ Cross-App Flow  
Intercambio de datos en tiempo real
- ✓ Crash Recovery  
Simule una interrupción anormal para verificar la capacidad de recuperación de datos del sistema.

# Resumen y perspectivas futuras

## Conclusión y trabajo futuro

### Resume

- ✓ Se construyó un sistema completo.
- ✓ Coherencia de datos en múltiples dispositivos
- ✓ Desarrollo full-stack individual

### Perspectivas futuras

- + Presentamos la tecnología WebSocket.
- + Ampliar el contenido del juego.
- + Optimiza el rendimiento móvil y mejora la funcionalidad.



Thanks for listening  
**Gracias por escucharnos!**

