## CONTROL Y REGISTRO DE CAMBIO DEL DOCUMENTO

|  |  |
| --- | --- |
| CONTROL | |
| Proyecto | NOMBRE DEL PROYECTO |
| Denominación | Diseño del Sistema de Información NOMBRE DEL PROYECTO |
| Fecha | día de mes de año |
| Edición | VV |
| Grupo | Identificación del grupo |
| Autores | Nombre de los Alumnos |

|  |  |  |
| --- | --- | --- |
| REGISTRO DE CAMBIOS | | |
| VERSIÓN | DESCRIPCIÓN DEL CAMBIO | FECHA DEL CAMBIO |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## ÍNDICE

[CONTROL Y REGISTRO DE CAMBIO DEL DOCUMENTO 1](#_Toc523133479)

[ÍNDICE 1](#_Toc523133480)

[1. DEFINICIÓN DEL SISTEMA 2](#_Toc523133481)

[1.1. Arquitectura del Sistema 2](#_Toc523133482)

[2. MODELO DE DATOS 4](#_Toc523133483)

[2.1. Modelo Entidad-Relación 4](#_Toc523133484)

[2.2. Modelo físico de datos 5](#_Toc523133485)

[2.3. Acceso a los Datos 11](#_Toc523133486)

[3. DISEÑO DE CASOS DE USO 12](#_Toc523133487)

[3.1. Diagrama de casos de uso 12](#_Toc523133488)

[3.2. Diagramas de secuencia del sistema 13](#_Toc523133489)

[4. DISEÑO DE PAQUETES 19](#_Toc523133490)

[4.1. DIAGRAMA DE PAQUETES 19](#_Toc523133491)

[4.2. DEFINICIÓN DE PAQUETES 21](#_Toc523133492)

[5. DISEÑO DE CLASES 23](#_Toc523133493)

[5.1. MODELO DE CLASES 23](#_Toc523133494)

[5.2. DEFINICIÓN DE CLASES 23](#_Toc523133495)

[6. PRUEBAS 37](#_Toc523133496)

[6.1. PRUEBAS UNITARIAS 37](#_Toc523133497)

[6.2. PRUEBAS DE INTEGRACIÓN 39](#_Toc523133498)

[6.3. PRUEBAS DE IMPLANTACIÓN 40](#_Toc523133499)

[7. REQUISITOS DE IMPLANTACIÓN 40](#_Toc523133500)

# DEFINICIÓN DEL SISTEMA

### Arquitectura del Sistema

A continuación, se muestra la arquitectura técnica de la aplicación a desarrollar:



Se trata de una aplicación web tradicional. Los usuarios interactúan con ella a través del navegador, lo que permite que la aplicación sea usada desde cualquier sistema operativo. El navegador web se comunica con el sistema mediante peticiones HTTP, y es necesario que este disponga de un intérprete de JavaScript. La integración MVC en el servidor web se lleva a cabo mediante Spring Boot, y la aplicación se aloja en un servidor Apache Tomcat. Para la persistencia, se emplea un servidor de base de datos MySQL, empleando el mapeado proporcionado por la herramienta Hibernate, así como un patrón DAO, para la integración entre esta y la aplicación Java.

En este caso, el servidor web y el de base de datos están en el mismo equipo, que puede ser un equipo físico o virtual según el tipo de hosting. También puede desplegarse la aplicación en un equipo Windows o MacOS tradicional mediante la herramienta XAMPP, si se desea persistencia permanente, o empleando la base de datos H2 que viene incluida en Spring para una persistencia temporal. De este modo se ofrece al cliente la posibilidad de trabajar con la aplicación de manera remota o en su equipo local, ya sea de manera permanente, o portátil, sin necesidad de instalación.

Finalmente, se aplicará una estructura en 3 capas, que será la capa de Presentación, capa de Negocio y capa de Datos. Esta estructura y las tecnologías que se utilizaran quedan representadas a continuación:



# MODELO DE DATOS

### Modelo Entidad-Relación

La persistencia de la aplicación se gestiona mediante el siguiente modelo Entidad-Relación. Mediante esta persistencia se pretende guardar toda la información posible del flujo de las partidas:



### Modelo físico de datos

El modelo Entidad-Relación anterior se ha implementado mediante el siguiente modelo relacional, que representa cómo será implementada la base de datos:



Las tablas identificadas se describen formalmente a continuación:

| Nombre | | | User | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Usuario que utiliza el sistema para crear o participar en partidas | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único del usuario |
| nickname | | | String | Sí | Apodo del usuario |
| name | | | String | Sí | Nombre real del usuario |
| avatar | | | String | No | Ruta a la imagen del usuario |
| password | | | String | Sí | Contraseña del usuario |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| user\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| - | - | | | | - |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| nickname\_pk | nickname | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | country | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | País con el que un jugador participa en la partida | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único del país |
| name | | | String | Sí | Nombre del país |
| player | | | Long | Sí | Jugador que lo controla |
| game | | | Long | Sí | Partida en la que se utilizó |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| country\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| player\_country\_fk | user | | | | id |
| game\_country\_fk | game | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | game | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Partida jugada o en curso | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único de la partida |
| name | | | String | Sí | Nombre de la partida |
| scenario | | | String | Sí | Escenario histórico |
| start\_date | | | Date | Sí | Fecha de inicio |
| end\_date | | | Date | No | Fecha de cierre |
| master | | | Long | Sí | Director de la partida |
| active\_turn | | | Long | No | Turno activo de la partida |
| scenario | | | Long | Sí | Escenario de la partida |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| game\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| master\_game\_fk | user | | | | id |
| turn\_game\_fk | turn | | | | id |
| scenario\_game\_fk | scenario | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| end\_date\_rest | end\_date | | | | No puede ser anterior a start\_date |

| Nombre | | | turn | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Turno en los que se divide una partida | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único del turno |
| subscenario | | | String | Sí | Subescenario o año del turno |
| game | | | Long | Sí | Partida a la que pertenece |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| turn\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| game\_turn\_fk | game | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | involvement | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Participación de un país en una coalición | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único de la participación |
| commitment\_percent | | | Float | Sí | Porcentaje de recursos dedicados |
| country | | | Long | Sí | País que participa |
| coalition | | | Long | Sí | Coalición en la que se participa |
| won\_the\_roll | | | Boolean | No | Si el país ganó la tirada |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| involvement\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| country\_involvement\_fk | country | | | | id |
| coalition\_involvement\_fk | coalition | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | war | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Enfrentamiento de una partida entre dos coaliciones | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único del enfrentamiento |
| name | | | String | Sí | Nombre del enfrentamiento |
| turn | | | Long | Sí | Turno al que pertenece |
| status | | | String | No | Estado del enfrentamiento |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| war\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| turn\_war\_fk | turn | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | coalition | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Coalición que participa en un enfrentamiento | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único de la coalición |
| name | | | String | Sí | Nombre de la coalición |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| coalition\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| - | - | | | | - |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | roll | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Tirada que determina el resultado de un enfrentamiento | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único de la tirada |
| attacker | | | Long | Sí | Coalición atacante |
| defender | | | Long | Sí | Coalición defensora |
| attacker\_score | | | Double | Sí | Puntuación del atacante |
| attacker\_roll | | | Integer | No | Tirada de dado del atacante |
| defender\_score | | | Double | Sí | Puntuación del defensor |
| defender\_roll | | | Integer | No | Tirada de dado del defensor |
| war | | | Long | Sí | Enfrentamiento al que pertenece |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| roll\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| war\_roll\_fk | war | | | | id |
| attacker\_war\_fk | coalition | | | | id |
| defender\_war\_fk | coalition | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

| Nombre | | | scenario | | |
| --- | --- | --- | --- | --- | --- |
| Descripción | | | Escenario histórico donde se desarrolla la partida | | |
| Atributos | | | | | |
| Campo | | | Tipo | Obligatorio | Descripción |
| id | | | Long | Sí | Id único del escenario |
| name | | | String | Sí | Nombre del escenario |
| description | | | String | No | Descripción del escenario |
| data\_file | | | String | Sí | Ruta al fichero de datos del escenario |
| creator | | | Long | Sí | Usuario creador del escenario |
| Clave primaria | | | | | |
| Nombre | | Columna | | | |
| scenario\_pk | | id | | | |
| Claves ajenas | | | | | |
| Nombre | Destino | | | | Columna |
| creator\_scenario\_fk | user | | | | id |
| Claves únicas | | | | | |
| Nombre | Columnas | | | | |
| - | - | | | | |
| Restricciones | | | | | |
| Nombre | Columnas | | | | Restricción |
| - | - | | | | - |

### Acceso a los Datos

En la aplicación el acceso a los datos se realiza siempre desde las clases del modelo, es decir, del patrón MVC, a través de Hibernate y haciendo uso de JDBC y ORM, que se encuentran en la capa de datos de la arquitectura realizada en el apartado 1.1.

# DISEÑO DE CASOS DE USO

### Diagrama de casos de uso



### Diagramas de secuencia del sistema

A continuación, se detalla el flujo en información entre los actores y el sistema en los casos de uso. Para no dificultar la comprensión del diagrama con demasiados flujos de datos, no se han tenido en cuenta los flujos alternativos; por lo que atributos, métodos o llamadas a objetos usados para validar las entradas, o que sean causados por entradas no válidas, no se han representado en los diagramas.

La clase HttpSession representa la clase proporcionada por Apache a través de Spring para guardar datos en la sesión del usuario. Las flechas de retorno, que devuelven la información al usuario, representan los datos que se asignan a la clase Model de Spring, que permite mostrar la información del controlador a través de las *templates* de Thymeleaf.

* CU01 – Login:



* CU02 – Registro:



* CU03 – Consultar perfil:



* CU04 – Modificar perfil:



* CU05 – Ver Partidas:



* CU06 – Ver Partida: El contenido del método *getChartData()* se ha dejado como abstracto para no dificultar la comprensión del procedimiento, que ya de por sí es complejo, y se representará en el gráfico del CU07 – Ver Estadísticas.



* CU07 – Ver Estadísticas: Se representa aquí el contenido del método *getChartData()*, que se dejó como abstracto en el diagrama anterior



* CU08 – Crear Escenario



* CU09 – Crear Partida



* CU10 – Añadir Escenario



* CU11 – Añadir Jugador



* CU12 – Cerrar Partida



* CU13 – Configurar Coaliciones: La llamada a *mapCountries()* se realiza de forma asíncrona desde el navegador del cliente automáticamente después de cargar la página.



* CU14 – Resolver Tirada



* CU15 – Terminar Enfrentamiento



* CU16 – Avanzar turno



* CU17 – Actualizar Partida: Se repite el mismo procedimiento que en el CU06 – Ver Partida, pero solo se actualiza la sección de la página que contiene la lista de enfrentamientos. Este proceso se lleva a cabo mediante una llamada asíncrona.



* CU18 – Logout:



# DISEÑO DE PAQUETES

### DIAGRAMA DE PAQUETES

La aplicación se ha dividido en paquetes siguiendo el patrón MVC. Los paquetes de los niveles inferiores comunican la información hacia arriba mediante POJOs, y los de los superiores mediante llamadas a métodos. Además, existen algunos paquetes con clases que cuentan con métodos y atributos constantes y estáticos que se pueden llamar desde cualquier punto de la aplicación.



### DEFINICIÓN DE PAQUETES

|  |  |
| --- | --- |
| **PAQ01** | **src.main.resources/templates** |
| Contenido | Contiene los templates de Thymeleaf que implementan la capa de vista de la aplicación. No contiene código Java. |
| Rol MVC | Vista |
| Comentarios | También contiene el código Javascript y CSS pertinente |

|  |  |
| --- | --- |
| **PAQ02** | **src.main.resources/templates/fragments** |
| Contenido | Contiene los fragmentos de templates de Thymeleaf que se repiten en todas las vistas de la aplicación, como los encabezados y los pies de página |
| Rol MVC | Vista |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ03** | **src.main.java** |
| Contenido | Paquete principal de Maven. Contiene todo el código Java de la aplicación. |
| Rol MVC | - |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ04** | **es.upo.tfg.rol** |
| Contenido | Paquete principal del proyecto Java. Contiene la clase principal de la aplicación que contiene el método *main()*, además de una clase con las constantes. |
| Rol MVC | - |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ05** | **exceptions** |
| Contenido | Contiene las excepciones de la aplicación |
| Rol MVC | - |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ06** | **controller** |
| Contenido | Contiene la capa de controlador de la aplicación. Esta capa se divide en dos partes, capa controladora y capa de servicio, según la estructura de Spring. |
| Rol MVC | Controlador |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ07** | **controllers** |
| Contenido | Contiene las clases con los métodos que el usuario activa mediante los templates de Thymeleaf y devuelven los resultados a través de estos templates. También se encargan de validar estas entradas. |
| Rol MVC | Controlador |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ08** | **validators** |
| Contenido | Contiene validadores para ciertos formularios más complejos |
| Rol MVC | Controlador |
| Comentarios | No ha sido necesario crear una clase validadora para cada formulario, solo para aquellos más complejos y restrictivos. |

|  |  |
| --- | --- |
| **PAQ09** | **service** |
| Contenido | Contiene la lógica de negocio de la aplicación |
| Rol MVC | Controlador |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ10** | **model** |
| Contenido | Contiene la capa del modelo de la aplicación |
| Rol MVC | Modelo |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ11** | **pojos** |
| Contenido | Contiene las clases POJO de la aplicación, mapeadas a través de Hibernate para ser representadas en la base de datos. |
| Rol MVC | Modelo |
| Comentarios | Estas clases solo se usan para representar y transmitir información, pero no incluyen ninguna lógica |

|  |  |
| --- | --- |
| **PAQ12** | **comparators** |
| Contenido | Contiene las clases comparadoras de algunos POJOs |
| Rol MVC | Modelo |
| Comentarios | - |

|  |  |
| --- | --- |
| **PAQ13** | **dao** |
| Contenido | Contiene las interfaces que se encargan del acceso a datos mediante la base de datos. Heredando de la interfaz CrudRepository proporcionada por Spring Boot y JPA, no es necesario implementar estar interfaces. |
| Rol MVC | Modelo |
| Comentarios | - |

# DISEÑO DE CLASES

### MODELO DE CLASES

Diagrama de clases de diseño, con las clases que formen parte del subsistema, con las clases ya refinadas. Identificar clases abstractas, herencias, asociaciones, etc.

### DEFINICIÓN DE CLASES

Si la descripción de un atributo coincide con la de una clase, se ha optado por detallarla en la tabla de la clase correspondiente. La funcionalidad soportada por dichas clases es la siguiente:

#### PAQ-07: controllers

La descripción de las clases de este paquete es la siguiente:

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-01** | **UserController** | | |
| **Descripción** | Recoge, valida y procesa las entradas relacionadas con la entidad Usuario que el usuario envía a través de Thymeleaf | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| uServ | UserService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Index | Redirige a la página de inicio | |
| Register | Redirige a la página de registro | |
| Profile | Redirige a la página del perfil | |
| Logout | Cierra la sesión del usuario | |
| submitLogin | Realiza el login mediante usuario y contraseña | |
| submitRegister | Valida y realiza el registro mediante los datos del formulario | |
| submitUpdate | Valida y realiza la modificación de perfil con los datos del formulario | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-02** | **WarController** | | |
| **Descripción** | Recoge, valida y procesa las entradas relacionadas con la entidad Enfrentamiento que el usuario envía a través de Thymeleaf | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| cServ | CountryService | - |
| wServ | WarService | - |
| tServ | TurnService | - |
| rServ | RollService | - |
| scServ | ScenarioService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| createWar | Inicia el proceso para crear un enfrentamiento y dirige a la página de enfrentamientos | |
| endWar | Cierra un enfrentamiento en curso | |
| createRoll | Inicia el proceso para realizar una tirada, y para crear un enfrentamiento si es la primera tirada de este | |
|  | mapCountry | Inicia el proceso para devolver un Map con todos los países de una partida | |
|  | mapRules | Inicia el proceso para devolver un Map con los reglas de la partida | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-03** | **GameController** | | |
| **Descripción** | Recoge, valida y procesa las entradas relacionadas con la entidad Partida que el usuario envía a través de Thymeleaf | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| cServ | CountryService | - |
| gServ | GameService | - |
| uServ | UserService | - |
| tServ | TurnService | - |
| wServ | WarService | - |
| rServ | RollService | - |
| scServ | ScenarioService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| createGame | Comienza el proceso para crear una partida, redirigiendo a la página con tal propósito | |
| openGame | Devuelve una partida y toda la información relevante relacionada con esta | |
| landing | Redirige a la página principal del usuario, devolviendo sus partidas pertinentes | |
| removeScenario | Interrumpe el proceso de creación de partida una vez se ha elegido un escenario | |
| addScenario | Selecciona un escenario e inicia el proceso de generación de turnos, guardando el progreso en la sesión. | |
| removePlayer | Quita a un jugador del proceso de creación de partida | |
| createCountry | Añade un jugador al proceso de creación de partidas, creando un país a medida y guardando el progreso en la sesión | |
| creategameSubmit | Finaliza el proceso de creación de partidas y persiste los cambios | |
| closeGame | Cierra una partida en curso | |
| endTurn | Avanza el turno de una partida | |
| downloadFile | Proporciona el fichero de un país para descargar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-04** | **ScenarioController** | | |
| **Descripción** | Recoge, valida y procesa las entradas relacionadas con la entidad Escenario que el usuario envía a través de Thymeleaf | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| scServ | scenarioService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| scenarioPRG | Redirige a una página de confirmación tras haber creado un escenario, mostrando el resultado dela operación | |
| downloadFile | Proporciona el fichero de un escenario para descargar | |
| createScenario | Redirige a la página para crear un escenario | |
| createScenarioSubmit | Valida las entradas de creación de escenario e inicia dicho proceso | |
| mapScenario | Devuelve un Map con las reglas del escenario | |
| downloadTemplate | Proporciona una plantilla para crear escenarios | |
| findMatches | Busca coincidencias con la descripción de un escenario | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-05** | **ErrorControllerImpl** | | |
| **Descripción** | Implementa el ErrorControler de Spring para manipular errores | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| handleError | Redirige a la página de error | |
| getErrorPath | Devuelve la página de error | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-06** | **Access** | | |
| **Descripción** | Controla el acceso a la aplicación | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| reject | Método estático que devuelve a la página de error en caso de que alguien intente acceder sin estar logueado | |
| **Comentarios** | - | | |

#### PAQ-08: validators

#### PAQ-09: service

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-07** | **CountryService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad País | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| assembleCountry | Genera un país a partir de un fichero | |
| validateCountryFile | Valida un fichero de un país | |
| saveCountry | Persiste un país | |
| findCountries | Busca todos los países de una partida | |
| mapCountry | Mapea un país desde su fichero | |
| demapCountry | Escribe el mapa de un país en un fichero | |
| findById | Busca un país por id. | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-08** | **CountryServiceImpl** | | |
| **Descripción** | Implementa la interfaz CountryService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| countryRep | CountryRepository | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-09** | **GameService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Partida | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| saveGame | Persiste una partida | |
| createGame | Crea y persiste una partida | |
| findOpenGames | Busca partidas abiertas de un usuario | |
| findById | Busca una partida por id | |
| closeGame | Cierra una partida | |
| findClosedGames | Busca partidas cerradas de un usuario | |
| saveTurn | Persiste un turno | |
| nextTurn | Avanza el turno de una partida | |
| getChartDate | Obtiene y organiza los datos para los gráficos | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-10** | **GameServiceImpl** | | |
| **Descripción** | Implementa la interfaz GameService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| countryRep | CountryRepository | - |
| turnRep | TurnRepository | - |
| invRep | InvolvementRepository | - |
| countryService | CountryService | - |
| warService | WarService | - |
| rollService | RollService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| restoreCountries | Restaura los países de una guerra | |
| resetCountry | Resetea el fichero de un país | |
| countriesInCoalition | Devuelve los países de una coalición | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-11** | **RollService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Tirada | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByWar | Devuelve las tiradas de una guerra | |
| getWinner | Devuelve el vencedor de una tirada | |
| findByGame | Devuelve las tiradas de una partida | |
| findRollsFromCountry | Devuelve las tiradas de un país en una partida | |
| findWonRollsFromCountry | Devuelve las tiradas en las que un país ha vencido | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-12** | **RollServiceImpl** | | |
| **Descripción** | Implementa la interfaz RollService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| rollRepository | RollRepository | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-13** | **ScenarioService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Escenario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| createScenario | Crea y persiste un escenario | |
| mapScenario | Mapea y devuelve un escenario según su fichero | |
| findAllScenarios | Devuelve todos los escenarios | |
| findById | Devuelve un escenario según su id | |
| mapRules | Devuelve las reglas según el escenario | |
| validateScenarioFile | Valida el fichero de un escenario | |
| save | Persiste un escenario | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-14** | **ScenarioServiceImpl** | | |
| **Descripción** | Implementa la interfaz ScenarioService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| scenarioRepository | ScenarioRepository | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| findMatches | Devuelve si el escenario contiene las palabras clave proporcionadas | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-13** | **TurnService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Turno | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| generateTurns | Genera los turnos a partir de un escenario | |
| findByGame | Devuelve los turnos de una partida | |
| findTurnFromGame | Devuelve, de entre los turnos de una partida, el que coincide en nombre con el parámetro | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-14** | **TurnServiceImpl** | | |
| **Descripción** | Implementa la interfaz TurnService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| turnRep | TurnRepository | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-15** | **UserService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Usuario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findAllUsers | Devuelve todos los usuarios | |
| findById | Devuelve el usuario según el id | |
| findByLogin | Devuelve el usuario cuyo apodo y contraseña coincidan con los proporcionados | |
| findByNickname | Devuelve el usuario según el apodo | |
| saveUser | Persiste un usuario | |
| register | Da de alta un usuario y lo devuelve | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-16** | **UserServiceImpl** | | |
| **Descripción** | Implementa la interfaz UserService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| userRep | UserRepository | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-17** | **WarService** | | |
| **Descripción** | Interfaz que define la lógica de negocio relativa a la entidad Enfrentamiento | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| createWar | Crea un enfrentamiento | |
| endWar | Cierra un enfrentamiento | |
| findByGame | Devuelve una lista de enfrentamientos de una partida | |
| findById | Busca un escenario por id | |
| findOpenWar | Devuelve el enfrentamiento abierto de una partida, si lo hay | |
| findByTurn | Devuelve los enfrentamientos de un turno | |
| roll | Ejecuta una tirada y devuelve el resultado | |
| copyMap | Realiza una copia completa de un mapa | |
| findWinner | Devuelve la coalición ganadora de una tirada | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-18** | **WarServiceImpl** | | |
| **Descripción** | Implementa la interfaz WarService | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| warRep | WarRepository | - |
| rollRep | RollRepository | - |
| turnRep | TurnRepository | - |
| invRep | InvolvementRepository | - |
| coalRep | CoalitionRepository | - |
| countryService | CountryService | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| applyCostOfWar | Aplica penalizaciones a los países que han participado en una tirada | |
| arrageInvolvements | Reparte las intervenciones de los países | |
| damageCountry | Resta recursos a un país | |
| mergeCountryPartition | Suma las particiones de un mismo país | |
| findCountryByName | Busca un país por nombre en una partida | |
| Mismos que la Interfaz | | |
| **Comentarios** | - | | |

#### PAQ-11: pojos

A continuación, se definen las clases POJO. Todas las clases implementan la interfaz Serializable. La descripción de los atributos que se compartan de la entidad se ha obviado.

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-19** | **Coalition** | | |
| **Descripción** | Entidad Coalición | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| name | String | - |
| countryRep | CountryRepository | - |
| involvements | List<Involvement> | Lista de intervenciones de esta coalición |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-20** | **Country** | | |
| **Descripción** | Entidad País | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| name | String | - |
| data | String | - |
| player | User | - |
| game | Game | - |
| Atributes | Map<> | Mapeado de los atributos de un país según estén contenidos en su fichero |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-21** | **Game** | | |
| **Descripción** | Entidad Partida | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| name | String | - |
| scenario | Scenario | - |
| startDate | Date | - |
| endDate | Date | - |
| master | User | - |
| activeTurn | Turn | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-22** | **Involvement** | | |
| **Descripción** | Entidad Intervención | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| country | Country | - |
| coalition | Coalition | - |
| involvementPercent | Double | - |
| wonTheRoll | Boolean | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-23** | **Roll** | | |
| **Descripción** | Entidad Tirada | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| attackerScore | Double | - |
| attackerRoll | Integer | - |
| defenderScore | Double | - |
| defenderRoll | Integer | - |
| attacker | Coalition | - |
| defender | Coalition | - |
| war | War |  |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-24** | **Scenario** | | |
| **Descripción** | Entidad Escenario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| name | String | - |
| description | String | - |
| data | String | - |
| atributes | Map<> | Mapeado del fichero en forma de mapa. Este atributo es transitorio |
| autor | User | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-25** | **Turn** | | |
| **Descripción** | Entidad Turno | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| subscenario | String | - |
| game | Game | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-26** | **User** | | |
| **Descripción** | Entidad Usuario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| avatar | String | - |
| name | String | - |
| nickname | String | - |
| password | String | - |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-27** | **War** | | |
| **Descripción** | Entidad Enfrentamiento | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| id | Long | - |
| name | String | - |
| status | String | - |
| turn | Turn | - |
| rolls | List<Roll> | Listado de tiradas de una partida |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters de todos los atributos | |
| Equals & hashCode | Métodos que determinan la igualdad de dos objetos | |
| toString | Representa al objeto como String, solo usado para depurar | |
| **Comentarios** | - | | |

#### PAQ-12: comparators

Todos implementan la interfaz Comparator proporcionada por Java:

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-28** | **GameByDateComparator** | | |
| **Descripción** | Ordena las partidas por su fecha de inicio | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| compare | Compara dos partidas y devuelve el orden en función de cual es más antigua | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-29** | **TurnByIdComparator** | | |
| **Descripción** | Ordena los turnos por su id | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| compare | Compara dos turnos y devuelve el orden en función de su id | |
| **Comentarios** | - | | |

#### PAQ-13: dao

Todas las interfaces repositorio heredan de CrudRepository, proporcionada por Spring. Esta interfaz ya contiene métodos genéricos, como *save()* y *findById()*. Además, gracias a este framework no es necesario su implementación, basta con proporcionar la consulta como una anotación.

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-30** | **CoalitionRepository** | | |
| **Descripción** | Acceso a datos para la entidad Coalición | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-31** | **CountryRepository** | | |
| **Descripción** | Acceso a datos para la entidad País | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByGame | Devuelve los países de una partida | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-32** | **GameRepository** | | |
| **Descripción** | Acceso a datos para la entidad Partida | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByMaster | Devuelve las partidas creadas por un usuario | |
| findOpenByMaster | Devuelve las partidas creadas y todavía abiertas de un usuario | |
| finnClosedByMaster | Devuelve las partidas creadas y cerradas de un usuario | |
| findByPlayer | Devuelve las partidas jugadas por un usuario | |
| findOpenByPlayer | Devuelve las partidas jugadas y todavía abiertas de un usuario | |
| findClosedByPlayer | Devuelve las partidas jugadas y cerradas de un usuario | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-33** | **InvolvementRepository** | | |
| **Descripción** | Acceso a datos para la entidad Intervención | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByCoalition | Devuelve las intervenciones de una coalición | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-34** | **RollRepository** | | |
| **Descripción** | Acceso a datos para la entidad Tirada | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByGame | Devuelve las tiradas de una partida | |
| findByGameAndCountry | Devuelve las tiradas de una partida en las que un país ha participado | |
| findWonRollsByGameAndCountry | Devuelve las tiradas de una partida en las que un país ha ganado | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-35** | **ScenarioRepository** | | |
| **Descripción** | Acceso a datos para la entidad Escenario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-36** | **TurnRepository** | | |
| **Descripción** | Acceso a datos para la entidad País | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByGame | Devuelve los turnos de una partida | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-37** | **UserRepository** | | |
| **Descripción** | Acceso a datos para la entidad Usuario | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByNickname | Devuelve al usuario con ese apodo | |
| findByNicknameAndPassword | Devuelve al usuario con esos apodo y contraseña | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-38** | **WarRepository** | | |
| **Descripción** | Acceso a datos para la entidad Enfrentamiento | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| findByTurn | Devuelve los enfrentamientos de un turno | |
| findByGame | Devuelve todos los enfrentamientos de una partida | |
| **Comentarios** | - | | |

#### Resto de clases

A continuación, se definen las clases de utilidad, que ayudan a la validación y homogenización de la aplicación, pero no participan en el proceso de negocio.

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-39** | **NotAurthorized** | | |
| **Descripción** | Excepción que se lanza cuando se intenta acceder a alguna funcionalidad sin estar logueado | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-40** | **Validator** | | |
| **Descripción** | Interfaz que valida los formularios | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| validate | Valida el formulario asegurándose de que todos los atributos de error son *null* | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-41** | **GameValidator** | | |
| **Descripción** | Implementa la interfaz Validator. Sirve para validar el proceso de creación de partida | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| Atributos para cada error posible | String | Si se detecta un error, se guarda un mensaje de error específico |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters para cada mensaje de error | |
| Métodos de la interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-42** | **RollValidator** | | |
| **Descripción** | Implementa la interfaz Validator. Sirve para validar el proceso de creación de una tirada | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| Atributos para cada error posible | String | Si se detecta un error, se guarda un mensaje de error específico |
| **Operaciones** | **Nombre** | **Descripción** | |
| Getters & setters | Getters y setters para cada mensaje de error | |
| Métodos de la interfaz | | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-43** | **Rules** | | |
| **Descripción** | Guarda todas las constantes de la aplicación. En general hay dos tipos, las que reflejan las reglas de la partida, y las que intervienen en el parseo, escritura y almacenamiento de ficheros | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| Constantes estáticas | String, Double e Integer | Todas las constantes de la aplicación |
| **Operaciones** | **Nombre** | **Descripción** | |
| getRules | Devuelve un mapa con los nombres de las reglas de la partida y sus valores | |
| **Comentarios** | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **CL-44** | **RolAplication** | | |
| **Descripción** | Clase que pone la aplicación en marcha. | | |
| **Atributos** | **Nombre** | **Tipo** | **Descripción** |
| **Operaciones** | **Nombre** | **Descripción** | |
| main | Método que arranca la aplicación Spring Boot | |
| **Comentarios** | - | | |

# PRUEBAS

### PRUEBAS UNITARIAS

Las pruebas unitarias se realizarán usando las herramientas proporcionadas por Spring Boot Test y jUnit. El proceso se construirá para que, de manera automática, se lance una vez y se ejecuten todas las pruebas

#### Alcance de las pruebas unitarias

Se testearán los métodos incluidos en la capa de servicio, ya que es la que aloja la lógica de negocio de la aplicación. Para ello, se han definido tres categorías en las que clasificar los métodos de las clases de esta capa, en función de su complejidad, para aplicar el tipo de testeo que mejor se ajuste:

|  |  |  |
| --- | --- | --- |
| **Categoría de complejidad** | **Tipo de testeo** | **Justificación** |
| Alta | Manual | El método realiza varias escrituras en la base de datos y en ficheros. Un testeo manual, usando una herramienta de depuración, permite ejecutar paso a paso el procedimiento y comprobar los valores de las variables que se están escribiendo en tiempo real, lo que permite una granularidad más fina en el control que una prueba de caja negra. |
| Media | Unitario | Métodos que aplican alguna transformación además de acceder a los datos. Con pruebas unitarias se puede comprobar rápidamente que el valor devuelto o escrito es el correcto. |
| Trivial | Ninguno | Métodos que solo escriben o leen la capa de datos. Como estos accesos a datos no se han implementado, sino que se han usado los proporcionados por Spring, no es necesario su testeo. |

#### Definición de las pruebas unitarias

Como la validación de las entradas se realizan en la capa Controladora, la mayoría de los métodos de la capa de servicio no devuelven fallos. La validación de las entradas erróneas se testeará en las pruebas de integración.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CountryService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-01 | assembleCountry | Unitario | OK | OK |
| PR-02 | mapCountry | Unitario | OK | OK |
| PR-03 | validateCountryFile | Unitario | OK | OK |
| PR-04 | validateCountryFile | Unitario | 2 errores | 2 errores |
| PR-05 | demapCountry | Unitario | OK | OK |
| **GameService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-06 | createGame | Unitario | OK | OK |
| PR-07 | findOpenGames | Manual | OK | OK |
| PR-08 | findClosedGames | Manual | OK | OK |
| PR-09 | nextTurn | Unitario | OK | OK |
| PR-10 | getChartData | Manual | OK | OK |
| **ScenarioService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-11 | createScenario | Unitario | OK | OK |
| PR-12 | mapScenario | Unitario | OK | OK |
| PR-13 | validateScenarioFile | Unitario | OK | OK |
| PR-14 | validateScenarioFile | Unitario | 2 errores | 2 errores |
| **TurnService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-15 | generateTurns | Unitario | OK | OK |
| **UserService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-16 | register | Unitario | OK | OK |
| PR-17 | register | Unitario | Excepción | Excepción |
| PR-18 | findLogin | Unitario | OK | OK |
| PR-19 | findLogin | Unitario | NULL | NULL |
| PR-20 | findByNickname | Unitario | OK | OK |
| PR-21 | findByNickname | Unitario | NULL | NULL |
| **WarService** | | | | |
| **Identificador** | **Método** | **Tipo de testeo** | **Resultado esperado** | **Resultado obtenido** |
| PR-22 | copyMap | Unitario | OK | OK |
| PR-23 | roll | Manual | OK | OK |

### PRUEBAS DE INTEGRACIÓN

Las pruebas de integración se llevarán a cabo en la aplicación directamente, simulando el comportamiento de un usuario. Se probará cada caso de uso, o conjunto de casos de uso en caso de que varios constituyan un único proceso de negocio, introduciendo entradas erróneas y válidas y comprobando los resultados.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Proceso de negocio** | **Casos de uso** | **Resultado esperado** | **Resultado obtenido** |
| PI-01 | Login | CU01 | OK | OK |
| PI-02 | Login | CU01 | Error por credenciales incorrectas | Error por credenciales incorrectas |
| PI-03 | Registro | CU02 | OK | OK |
| PI-04 | Registro | CU02 | Error por usuario ya existente | Error por usuario ya existente |
| PI-05 | Registro | CU02 | Error por entradas no válidas | Error por entradas no válidas |
| PI-06 | Modificar Perfil | CU04 | OK | OK |
| PI-07 | Modificar Perfil | CU04 | Error por entradas no válidas | Error por entradas no válidas |
| PI-08 | Ver partida | CU05, CU06, CU07 | OK | OK |
| PI-09 | Crear escenario | CU08 | OK | OK |
| PI-10 | Crear escenario | CU08 | Errores en fichero | Errores de fichero |
| PI-11 | Crear partida | CU09, CU10, CU11 | OK | OK |
| PI-12 | Crear partida | CU09, CU10, CU11 | Error por escenario no seleccionado | Error por escenario no seleccionado |
| PI-13 | Crear partida | CU09, CU10, CU11 | Error por jugador inexistente | Error por jugador inexistente |
| PI-14 | Crear partida | CU09, CU10, CU11 | Error por fichero de jugador | Error por fichero de jugador |
| PI-15 | Cerrar partida | CU12 | Error por contraseña incorrecta | Error por contraseña incorrecta |
| PI-16 | Cerrar partida | CU12 | OK | OK |
| PI-17 | Configurar coaliciones | CU13 | OK | OK |
| PI-18 | Resolver tirada | CU14 | Error por coalición vacía | Error por coalición vacía |
| PI-19 | Resolver tirada | CU14 | Error por más del 100% de los recursos | Error por más del 100% de los recursos |
| PI-20 | Resolver tirada | CU14 | Error por país no reconocido | Error por país no reconocido |
| PI-21 | Resolver tirada | CU14 | Error por escenario no reconocido | Error por escenario no reconocido |
| PI-22 | Terminar enfrentamiento | CU15 | OK | OK |
| PI-23 | Avanzar turno | CU16 | OK | OK |
| PI-24 | Actualizar partida | CU17 | OK | OK |

### PRUEBAS DE IMPLANTACIÓN

Para llevar a cabo estas pruebas, se repetirán las pruebas de integración en el hosting final.

# REQUISITOS DE IMPLANTACIÓN

Para la implantación del sistema, será necesario un hosting que ofrezca una máquina virtual de Ubuntu 16.04, a la que se pueda acceder remotamente mediante SSH. En esta máquina virtual será necesario instalar Java y MySQL, y crear una base de datos. Una vez hecho esto, al ejecutar el paquete JAR generado por Spring Boot, se crearán de manera automática las tablas necesarias y se pondrá en marcha el servidor Apache Tomcat, que Spring incluye en el paquete JAR al compilarlo.

Este mismo procedimiento se puede llevar a cabo para desplegar el sistema de manera local, creando la base de datos mediante XAMPP, y ejecutando el paquete JAR en un ordenador. Para la versión local y portátil, basta con compilar otro JAR donde se ha indicado a Spring en el fichero Properties que la base de datos a usar es H2, para que Spring la incluya también en el paquete.