



Age of STEMpires



- FASE INICIAL

Consideraciones IMPORTANTES:

1. El proyecto se entregará vía GitHub: <https://classroom.github.com/a/6CevaKa1>
2. La gestión de versiones es **obligatoria para que el proyecto sea evaluado**:
 - a. Debemos ver varios commits para comprobar la evolución del proyecto.
 - b. Proyectos realizados por arte de magia, con un único commit final, varios commits con el mismo contenido, pocos commits o commits en un espacio muy breve de tiempo, se entenderán como no realizados por el alumno.
 - c. Si no se cumple este requisito, la calificación del proyecto es de cero (0) puntos.
3. La defensa del proyecto es obligatoria. Si no se realiza la defensa, la calificación del proyecto es de cero (0) puntos.

Otras consideraciones:

1. La aplicación se deberá diseñar e implementar respetando el patrón de diseño **Modelo-Vista-Controlador**.
2. Existirá un **repositorio compartido** entre las asignaturas de **Bases de Datos** y **Programación**, el resto de asignaturas presentarán sus repositorios independientes.
3. Para diferenciar los commits referentes a cada una de las asignaturas (Bases de Datos o Programación) se escribirá **BBDD PROG** al principio de la descripción de todos los commits de la parte de programación.



Alternativas para realizar los proyectos:

- Si se quiere realizar solo la práctica de **programación**: El uso de la aplicación se realizará a través de PostMan. La base de datos se sustituirá por estructuras de datos creadas y gestionadas directamente desde el Modelo.
- Si se quiere realizar solo la práctica de **bases de datos**: El uso de la aplicación se realizará a través de PostMan. El controlador no deberá realizar ningún tipo de validación.
- Si se quiere realizar solo la práctica de **lenguaje de marcas**: En caso de no desarrollar la API del juego, se podrá usar la suministrada por los profesores.
- Si se quiere realizar solo la práctica de **programación y bases de datos**: El uso de la aplicación se realizará a través de PostMan y no hará falta realizar el portal web.
- Si se quiere realizar solo la práctica de **programación y lenguaje de marcas**: La base de datos se sustituirá por estructuras de datos creadas y gestionadas directamente desde el Modelo.
- Si se quiere realizar solo la práctica de **bases de datos y lenguaje de marcas**: El uso de la aplicación se realizará a través de PostMan. El controlador no deberá realizar ningún tipo de validación y el portal web se deberá conectar a la API de los profesores.


Requisitos Funcionales iniciales (Realiza al menos un commit para cada RF):

-  **Crear monarca** : (**POST** /monarca).
-  **Asesinar monarca** : (**DELETE** /monarca).
-  **Listar todos los monarcas** : (**GET** /monarca).
-  **Obtener monarca por ID** : (**GET** /monarca/{id}).
-  **Actualizar monarca** : (**PUT** /monarca).

Crear monarca (POST /monarca):

- Este requisito funcional implica la capacidad de **crear** un **nuevo monarca** en el sistema. Se realiza mediante una solicitud HTTP **POST** al endpoint /monarca.
- Para crear un monarca, el cliente debe **enviar** los **datos necesarios** en el cuerpo de la solicitud:
 - Nombre.
 - Dinastía.
 - Escudo.
 - Descripción.
 - Nivel de estrategia.
 - Nivel de diplomacia.
 - **El resto** de valores **se generan automáticamente**.
- El servidor procesa la solicitud:
 - **Si** el **monarca** recibido **es correcto**, **crea el nuevo** monarca en la base de datos (el Modelo no debe recibir jamás monarcas incorrectos) **y devuelve** una respuesta indicando el éxito de la operación (**ID del nuevo monarca**).
 - **Si** el **monarca** recibido **es incorrecto** (no cumple alguna de las validaciones), el servidor **devuelve** una respuesta indicando el fracaso de la operación (**un cero 0**).

Asesinar monarca (DELETE /monarca):

- Este requisito funcional implica la capacidad de **asesinar a un monarca**  existente en el sistema, **marcándolo como muerto**. Se realiza mediante una solicitud HTTP **DELETE** al endpoint /monarca, junto con el monarca a asesinar **en el cuerpo de la petición**.
- El servidor procesa la solicitud:
 - **Si** el **monarca** recibido **existe**, marca al monarca **especificado como muerto** en la base de datos y devuelve una respuesta indicando el éxito de la operación (true).

- **En caso contrario**, devuelve una respuesta indicando el **fracaso** de la operación (false).

Listar todos los monarcas (GET /monarca):

- Este requisito funcional implica la capacidad de obtener una lista 📄 de todos los monarcas registrados en el sistema. Se realiza mediante una solicitud HTTP **GET** al endpoint /monarca.
- El servidor procesa la solicitud, **recupera la lista de monarcas de la base de datos** y la devuelve como parte del cuerpo de la respuesta.

Obtener monarca por ID (GET /monarca/{id}):

- Este requisito funcional implica la capacidad de **obtener información detallada** 📄 **sobre un monarca** específico en el sistema, **identificado** por su **ID**. Se realiza mediante una solicitud HTTP **GET** al endpoint /monarca/{id}, donde {id} es el ID único del monarca.
- El servidor procesa la solicitud, busca el monarca correspondiente en la base de datos utilizando el ID proporcionado y devuelve los detalles del monarca como parte del cuerpo de la respuesta. **Si el ID no es válido, devuelve un monarca nulo.**

Actualizar monarca (PUT /monarca):

- Este requisito funcional implica la capacidad de **actualizar la información de un monarca existente** en el sistema. Se realiza mediante una solicitud HTTP **PUT** al endpoint /monarca, junto con el monarca a modificar **en el cuerpo de la petición.**
- Sin embargo, solo se permite cambiar:
 - el nivel de estrategia,
 - el nivel diplomático
 - o el nivel de experiencia del monarca.
- El servidor procesa la solicitud:
 - **Si el monarca a actualizar existe, actualiza los campos** especificados del monarca en la base de datos y devuelve una respuesta indicando el éxito de la operación (true).
 - **En caso contrario, devuelve** una respuesta indicando el fracaso de la operación (false).

Clase MONARCA:

- **ID del monarca:** Es un número entero **generado automáticamente por la base de datos** para identificar de manera única a cada monarca.
- **Nombre del monarca** 👑: Debe ser uno de los **nombres** propios **asociados a su dinastía**. Por ejemplo, si pertenece a la dinastía de Tudor, su nombre debe ser Henry, Elizabeth, Edward, etc.
- **Ordinal del nombre del monarca** : Se trata de un **número romano** que **se calcula** en función de la cantidad de **monarcas con el mismo nombre de su misma dinastía** que haya en la base de datos. Por ejemplo, si hay dos monarcas llamados Henry en la misma dinastía, el segundo se representaría como Henry II. Si no hay ningún Henry en su dinastía, se representaría como Henry I.
- **Dinastía del monarca** 🏰: Se selecciona **de una lista** de dinastías proporcionada. Esto identifica la línea familiar a la que pertenece el monarca, como Tudor, Habsburgo, Valois, etc.
- **Escudo del monarca** 🛡️: Es la **URL** que apunta **al escudo** asociado al monarca. Este escudo **representa visualmente al monarca**.
- **Descripción del monarca** 📖: Incluye **detalles físicos y de personalidad** del monarca. Esto puede abarcar desde su apariencia física hasta sus características de liderazgo y estilo de gobierno.
- **Nivel de estrategia del monarca** 📊 : Representa la **habilidad estratégica** del monarca y se valora en una escala **del 1 al 10**, donde 1 es el nivel más bajo y 10 el más alto. Este nivel se puede ajustar dentro del rango **inicialmente** establecido **entre 1 y 5 puntos**, teniendo **inicialmente** un total de **5 puntos a repartir entre** el nivel de **estrategia y el nivel diplomático** del monarca.
- **Nivel diplomático del monarca** 📈: Indica la **habilidad diplomática** del monarca y también se valora en una escala **del 1 al 10**. Al igual que el nivel de estrategia, este nivel se puede ajustar dentro del rango **inicialmente** establecido **entre 1 y 5 puntos**, teniendo **inicialmente** un total de **5 puntos a repartir entre** el nivel de **estrategia y el nivel diplomático** del monarca.
- **Nivel de experiencia del monarca (edad)** 🧒: Corresponde a la **edad** del monarca. **Al comienzo** de su reinado se establece **aleatoriamente entre los 14 y los 35 años**.
- **Vivo o muerto** 💀: Indica el estado actual del monarca, es decir, si está vivo o muerto. Esto proporciona información sobre la situación actual de cada monarca en el sistema.

Lista de dinastías y nombres propios de cada una:

1. **Dinastía de Alfonso (Portugal):** Afonso, Leonor, Pedro, Isabel, João, Beatriz, Catarina, Manuel, Antonia y Maria.



2. **Dinastía de Anjou (Francia):** Louis, Marguerite, Charles, Catherine, Philippe, Marie, René, Isabelle, Henri y Jeanne.
3. **Dinastía de Aragón (España):** Pedro, Isabel, Jaime, Leonor, Juan, Maria, Alfonso, Beatriz, Martin y Juana.
4. **Dinastía de Borgoña (Francia):** Philippe, Marguerite, Charles, Catherine, Louis, Isabelle, René, Marie, Henri y Jeanne.
5. **Dinastía de Braganza (Portugal):** João, Maria, Pedro, Isabel, Afonso, Beatriz, Catarina, Manuel, Antonia y Leonor.
6. **Dinastía de Castilla (España):** Fernando, Isabel, Diego, Juana, García, Beatriz, Juan, Teresa, Rodrigo y Leonor.
7. **Dinastía de Habsburgo (Sacro Imperio Romano Germánico):** Maximilian, Maria, Charles, Isabella, Philip, Margaret, Ferdinand, Joanna, Albert y Catherine.
8. **Dinastía de Lancaster (Inglaterra):** Henry, Margaret, Edward, Anne, Richard, Elizabeth, John, Catherine, Thomas y Mary.
9. **Dinastía de Normandía (Normandía):** William, Matilda, Robert, Adela, Richard, Emma, Henry, Constance, Geoffrey y Joan.
10. **Dinastía de Tudor (Inglaterra):** Henry, Elizabeth, Edward, Mary, Catherine, Anne, Arthur, Margaret, Jane y Thomas.
11. **Dinastía de Valois (Francia):** Louis, Catherine, Charles, Marie, Philippe, Isabelle, Henri, Jeanne, Marguerite y René.
12. **Dinastía de Wittelsbach (Alemania):** Louis, Isabella, Otto, Elisabeth, Rudolf, Maria, Albert, Margaret, William, Agnes.
13. **Dinastía de York (Inglaterra):** Edward, Margaret, Richard, Anne, John, Elizabeth, Thomas, Catherine, Henry y Mary.



Requisitos Funcionales primer incremento (**Realiza al menos un commit para cada RF**):

- Crear reino : (**POST** /reino).
- Eliminar reino: (**DELETE** /reino).
- Listar todos los reinos: (**GET** /reino).
- Obtener reino por ID: (**GET** /reino/{id}).
- Actualizar reino: (**PUT** /reino).

Crear reino (**POST** /reino):

- Este requisito funcional implica la capacidad de **crear** un **nuevo reino** en el sistema. Se realiza mediante una solicitud HTTP **POST** al endpoint /reino.
- Para crear un reino, el cliente debe **enviar** los **datos necesarios** en el cuerpo de la solicitud:
 - Nombre.
 - Descripción.
 - Año.
 - Capital.
 - Monarca.
 - Superficie.
 - Población.
 - Bandera.
 - **El resto** de valores **se generan automáticamente**.
- El servidor procesa la solicitud:
 - **Si** el **reino** recibido **es correcto**, **crea el nuevo** reino en la base de datos (el Modelo no debe recibir jamás reinos incorrectos) **y devuelve** una respuesta indicando el éxito de la operación (**ID del nuevo reino**).
 - **Si** el **reino** recibido **es incorrecto** (no cumple alguna de las validaciones), el servidor **devuelve** una respuesta indicando el fracaso de la operación (**un cero 0**).

Eliminar reino (**DELETE** /reino):

- Este requisito funcional implica la capacidad de **eliminar un reino** existente en el sistema. Se realiza mediante una solicitud HTTP **DELETE** al endpoint /reino, junto con el reino a eliminar **en el cuerpo de la petición**.
- El servidor procesa la solicitud:
 - **Si** el **reino** recibido **existe**, elimina el reino en la base de datos y devuelve una respuesta indicando el éxito de la operación (true).



PROYECTO TERCER TRIMESTRE

Programación y Bases de Datos

- **En caso contrario**, devuelve una respuesta indicando el **fracaso** de la operación (false).

Listar todos los reinos (**GET** /reino):

- Este requisito funcional implica la capacidad de obtener una lista 📄 de todos los reinos registrados en el sistema. Se realiza mediante una solicitud HTTP **GET** al endpoint /reino.
- El servidor procesa la solicitud, **recupera la lista de reinos de la base de datos** y la devuelve como parte del cuerpo de la respuesta.

Obtener reino por ID (**GET** /reino/{id}):

- Este requisito funcional implica la capacidad de **obtener información detallada** 📄 **sobre un reino** específico en el sistema, **identificado** por su **ID**. Se realiza mediante una solicitud HTTP **GET** al endpoint /reino/{id}, donde {id} es el ID único del reino.
- El servidor procesa la solicitud, busca el reino correspondiente en la base de datos utilizando el ID proporcionado y devuelve los detalles del reino como parte del cuerpo de la respuesta. **Si el ID no es válido, devuelve un reino nulo.**

Actualizar reino (**PUT** /reino):

- Este requisito funcional implica la capacidad de **actualizar la información de un reino existente** en el sistema. Se realiza mediante una solicitud HTTP **PUT** al endpoint /reino, junto con el reino a modificar **en el cuerpo de la petición**.
- Sin embargo, solo se permite cambiar:
 - la capital,
 - la descripción,
 - el monarca,
 - la bandera del reino.
- El servidor procesa la solicitud:
 - **Si el reino** a actualizar **existe, actualiza los campos** especificados del reino en la base de datos y devuelve una respuesta indicando el éxito de la operación (true).
 - **En caso contrario, devuelve** una respuesta indicando el fracaso de la operación (false).



Clase REINO:

- **Nombre del Reino:** Debe tener un máximo de 20 caracteres y no puede contener números. Además, la primera letra de cada palabra debe ser en mayúscula.
- **Descripción del Reino:** Una breve descripción que brinde información relevante sobre el reino.
- **Año de fundación:** Debe ser un año comprendido entre 400 y 1400, representando el período de establecimiento del reino.
- **Capital del Reino:** Se limita a 20 caracteres y no puede contener números. Al igual que el nombre del reino, la primera letra de cada palabra debe ser en mayúscula.
- **Monarca del Reino:** Debe haber un monarca designado para el reino. Para más detalles sobre el monarca, consulte la clase Monarca.
- **Superficie:** Se especifica en kilómetros cuadrados y representa el área total del territorio del reino.
- **Población del Reino:** Debe tener al menos una persona, que es el monarca. La población máxima depende de la superficie del reino y no puede exceder una densidad de población de 10 habitantes por kilómetro cuadrado.
- **Lista de Antiguos Monarcas del Reino:** Se presenta en orden cronológico, desde el monarca más antiguo hasta el más reciente.
- **Bandera del Reino:** Se proporciona una URL que dirige a la bandera del reino.
- **Cantidad de Piedra del Reino:** Se genera un valor aleatorio entre 1 y un valor calculado utilizando la fórmula:
$$piedra\ máxima = \left(\frac{superficie\ del\ reino}{superficie\ del\ reino\ más\ grande} \times 500 \right) + 500$$
- **Cantidad de Madera del Reino:** Se genera un valor aleatorio entre 1 y un valor calculado utilizando la fórmula:
$$madera\ máxima = \left(\frac{superficie\ del\ reino}{superficie\ del\ reino\ más\ grande} \times 500 \right) + 500$$
- **Cantidad de Oro del Reino:** Se genera un valor aleatorio entre 1 y un valor calculado utilizando la fórmula:
$$cantidad\ de\ oro\ máximo = \left(\frac{población\ del\ reino}{población\ del\ reino\ más\ grande} \times 500 \right) + 500$$

Requisitos Funcionales segundo incremento (Realiza al menos un commit para cada RF):

- **Atacar reino** 🗡️:
(**GET** /reino/atacar/{idReinoAtacante}/{idReinoAtacado}).
- **Pasar turno** : (**GET** /reino/pasar).

Paso de turno (**GET** /reino/pasar):

A todos los reinos que queden en pie...

- Se les suma un **tiempo de vida** de entre 1 y 10 años a sus **monarcas** ⌚, esté valor será obtenido de forma aleatoria.
- La probabilidad de **morir** 💀 de un monarca depende de dos valores:
 - Uno de los valores es un número **aleatorio** 🎲 comprendido entre el 0 y el 100.
 - El otro valor es un número **aleatorio** 🎲 comprendido entre 0 y la edad del monarca ⌚.
 - **Si el segundo valor es mayor que el primero, el monarca muere** 💀.
- Cuando un monarca muere 💀:
 - Se debe generar un **nuevo monarca** que pertenecerá a la **misma dinastía**.
 - Se le asignará un **nombre aleatorio** de entre la lista de nombres proporcionados para dicha dinastía.
 - Este **nuevo monarca** será ahora el monarca del reino 👑.
- La cantidad de **piedra** 🪨 que tiene un reino **aumenta** un valor aleatorio 🎲 entre el 1 y el valor obtenido a partir de la siguiente fórmula:

$$\text{cantidad de piedra} = \left(\frac{\text{superficie del reino}}{\text{superficie del reino más grande}} \times 500 \right) + 500$$

- La cantidad de **madera** 🌲🪚 que tiene un reino **aumenta** un valor aleatorio 🎲 entre el 1 y el valor obtenido a partir de la siguiente fórmula:

$$\text{cantidad de madera} = \left(\frac{\text{superficie del reino}}{\text{superficie del reino más grande}} \times 500 \right) + 500$$

- La cantidad de **oro** 💰 que tiene un reino **aumenta** un valor aleatorio 🎲 entre el 1 y el valor obtenido a partir de la siguiente fórmula:

$$\text{cantidad de oro} = \left(\frac{\text{población del reino}}{\text{población del reino más poblado}} \times 500 \right) + 500$$

- La cantidad de **población** 🌍 que tiene un reino **aumenta** un valor aleatorio 🎲 entre el 1 y el valor obtenido a partir de la siguiente fórmula:

$$\text{cantidad de población} = \left(\frac{\text{población del reino}}{\text{población del reino más poblado}} \times 5000 \right) + 5000$$

Guerra - Atacar reino 🗡️**(GET /reino/atacar/{idReinoAtacante}{idReinoAtacado}):**

- Hay dos reinos, el **atacante** 🗡️ (primero en ser seleccionado) y el **atacado** 🛡️ (segundo en ser seleccionado).
- El método devolverá el **ID del reino** que resulte **vencedor** 🏆.
- Se debe comprobar que los reinos indicados existen. En caso contrario, devuelve 0.
- Para cada reino, se calculará una **puntuación** de la siguiente forma:
 - **Puntuación por población** 🌍: Se obtiene un valor entre el 1 y el 10 a partir de la siguiente fórmula:

$$\text{puntuación por población} = \frac{\text{población del reino} - \text{población mínima}}{\text{población máxima} - \text{población mínima}} \times 10$$
 Siendo la población máxima y la población mínima la población del reino con más población y la población del reino con menos población respectivamente.
 - **Puntuación por territorio** 🗺️: Se obtiene un valor entre el 1 y el 10 a partir de la siguiente fórmula:

$$\text{puntuación por territorio} = \frac{\text{territorio del reino} - \text{territorio mínimo}}{\text{territorio máximo} - \text{territorio mínimo}} \times 10$$
 Siendo el territorio máximo y el territorio mínimo el territorio del reino con más territorio y el territorio del reino con menos territorio respectivamente.
 - **Puntuación por estrategia** ♟️: Será el valor del nivel estratégico del monarca.
 - **Puntuación por experiencia** ⌚:
 - Si la edad del monarca es mayor o igual a 80 (👴), la puntuación por experiencia es 0.
 - Si la edad del monarca es menor que 80 (👦), la puntuación por experiencia es un valor entre el 1 y el 10 a partir de la siguiente fórmula:

$$\text{puntuación por experiencia} = -\frac{1}{160} \times (\text{edad del monarca} - 40)^2 + 10$$
 - **Componente aleatorio** 🎲: Un valor aleatorio entre 0 y 30.
- **Gana el reino que consiga más puntos** 💪.

Cuando la guerra termina...

- El reino perdedor se **elimina** ❌.
- El nivel de **estrategia** del monarca ganador sube un punto 📈 (hasta un máximo de 10).
- El reino ganador **actualiza** su **población** 🌍, que pasará a ser un 70% de la suma de las poblaciones de los dos reinos, e incorpora los **territorios** 🗺️ (completos) y **recursos** (todos) del perdedor a los suyos.
- El monarca del reino **perdedor muere** 💀.