RPG Data API

Use Case Description: Allows a user to update a weapon's attributes stored in a database. The attributes represent the weapon's generic statistics that can be utilized by any role-playing game application.

Use Case Name: Update Weapon

Actors:

- User

Triggers:

- The user wants to update a weapon for their role-playing game

Preconditions:

- The user knows which weapon and attribute values should be updated

Post Conditions:

- A role-playing game weapon is updated

Normal Flow:

1. The user executes an HTTP PUT call using available HTTP client such as curl or Postman and sends the attribute information in a JSON request body and the UUID for the weapon.
2. The RPG Data API receives the PUT request sent by the user, and the WeaponsController object executes the "put" method.
3. The JSON request body sent by the user is transformed into a Weapon object, and the object's contents are validated.
4. If the Weapon object has invalid data, an error response is returned to the user.
5. If the data is valid, the "updateWeapon" method is called on the WeaponsService interface which executes the method implemented on the WeaponsServiceImpl object.
6. The "updateWeapon" method gets the weapons UUID identifier. If the identifier can not be found an exception is thrown and an error message is displayed to the user.

7. If the weapon does not exist, the method returns null.
8. If the weapon does exist and is not null, the weapon and the associated attributes are updated.
9. The updated weapon object makes a “save” method call on the WeaponsRepository interface.
10. The WeaponsRepository interface executes the “save” method via the Spring Data JPA framework. When the method call fails an exception is thrown and an error message is displayed.
11. The updated weapon is returned to the WeaponsController object and the WeaponsController object creates and returns a ResponseEntity object storing the updated weapon attributes and an HTTP response code 200 OK to the end user.