Lab05 - Spring Boot

Zadania

1. Zaimplementuj kontroler restowy, obsługujący przykładowe żądania oraz dodaj kilka własnych :

Lp.	Typ metody	Endpoint	Opis
1	POST	/api/horse	dodaje konia do stadniny
2	DELETE	/api/horse/:id	usuwa konia ze stadniny (listy)
3	GET	/api/horse/rating/:id	zwraca średnią ocenę konia
4	POST	/api/ horse /rating	dodaje ocenę dla konia
5	GET	/api/ stable	zwraca wszystkie stadniny
6	GET	/api/ stable /:id	zwraca wszystkie konie w podanej stadninie
7	GET	/api/ stable /:id/csv	zwraca wszystkie konie w formie pliku CSV
8	POST	/api/ stable	dodaje nową stadninę
9	DELETE	/api/ stable /:id	usuwa stadninę
10	GET	/api/ stable /:id/fill	zwraca zapełnienie stadniny

2. Aplikacja powinna przechowywać dane w bazie danych (wykonane ćwiczenie lab03) lub w StableManager (wykonane ćwiczenie lab02).
3. Napisz **testy jednostkowe** dla wskazanych w zadaniu 1 endpointów.
4. Przetestuj działanie np. wykorzystując aplikację Postman.
5. Wykorzystaj Mavena do dołączania zależności oraz uruchamiania i testowania projektu.
6. Pamiętaj o poniższych uwagach!

Uwagi

1. W kontrolerze powinna znajdować się obsługa przychodzącego obiektu **request** oraz zwrócenie obiektu **response**. Wszystkie pozostałe operacje, tj. połączenie z bazą danych, generacja CSV, wyliczanie wartości, powinno odbywać się w **innej**, **dedykowanej** do tego celu, **klasie**.
2. Jeżeli obiekt o podanym id nie istnieje, należy zwrócić **status 404**.
3. Żadna metoda GET nie może modyfikować stanu obiektów.
4. Przechwytywanie wyjątków jest obowiązkowe. Należy bowiem unikać zwracaniu **error 500**.
5. Do testów można wykorzystać aplikacje [**Postman**](https://www.postman.com/) lub [Insomnia](https://insomnia.rest/).
6. Przesłanie nieprawidłowych parametrów danego routingu powinny powodować zwrócenie odpowiedniego **statusu HTTP** oraz **informacji o błędzie**.
7. Obsługa **wyjątków** powinna być sensowna - wykorzystaj odpowiednie kody HTTP.

Materiały

1. [Dobry start w IntelliJ] (https://medium.com/@ahmetkapusuz/spring-boot-hello-world-application-with-intellij-idea-1524c68ddaae)
2. [Start z Mavenem] (https://spring.io/guides/gs/spring-boot/)
   Zachęcam do poprawnego korzystania z Mavena. Jest mi łatwiej uruchamiać projekt.
3. [Spring Boot tutorials] (https://mkyong.com/tutorials/spring-boot-tutorials/)
4. [Baeldung Spring Tutorials] (https://www.baeldung.com/spring-tutorial)
5. [Dokumentacja Spring Boot] (https://docs.spring.io/spring-boot/docs/current/reference/html/index.html)

Przykładowe pytania teoretyczne lab 9

1. Komunikacja HTTP: podstawy, budowa zapytania (request) oraz odpowiedzi (response), statusy (200, 201, 400, 402, 404, 500) i ich kategorie.
2. Spring Boot vs Spring.
3. Co może zwracać kontroler w Spring Boot? Co oznacza skrót POJO?
4. Adnotacje: @SpringBootApplication, @RestController, @RequestMapping, @Autowired.
5. ***Ważne!*** Czym jest Dependency Injection i jaką pełni rolę w Springu?
6. Tomcat, Jetty, Glassfish, Undertow - czym są te aplikacje, do czego służą i w jaki sposób są wykorzystywane w Spring Boot.
7. Jak wygląda obsługa wyjątków w Spring Boot?