Programowanie Aplikacji Użytkowych: lab09 - Spring Boot

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)

Zadania

1. Zaimplementuj kontroler restowy, obsługujący następujące żądania:

Lp.	Typ metody	Endpoint	Opis
1	POST	/api/product	dodaje pojazd do salonu
2	DELETE	/api/product/:id	usuwa pojazd do salonu
3	GET	/api/product/:id/rating	zwraca średnią ocenę pojazdu/salonu
4	GET	/api/product/csv	zwraca wszystkie pojazdy w formie pliku CSV
5	GET	/api/fulfillment	zwraca wszystkie salony
6	POST	/api/fulfillment	dodaje nowy salon
7	DELETE	/api/fulfillment/:id	usuwa salon
8	GET	/api/fulfillment/:id/products	zwraca wszystkie pojazdy w salonie
9	GET	/api/fulfillment/:id/fill	zwraca zapełnienie procentowe salonu
10	POST	/api/rating	dodaje ocenę dla pojazdu/salonu

2. Aplikacja powinna przechowywać dane w bazie danych (wykonane ćwiczenie lab08) lub w CarShowroomContainer (wykonane ćwiczenie lab05).
3. Napisz testy jednostkowe dla wskazanych w zadaniu 1 endpointów (1 pkt).
4. Wykorzystaj Mavena do dołączania zależności oraz uruchamiania i testowania projektu.

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.

Przykładowe pytania teoretyczne

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?