POLITECHNIKA GDAŃSKA

Platformy Technologiczne

Laboratorium 5

Testy Jednostkowe

Studenci w ramach zajęć laboratoryjnych zapoznają się z podstawowymi mechanizmami pisania testów jednostkowych za pomocą bibliotek JUnit oraz Mockito.

Zadanie powinno być realizowane jako projekt oparty na narzędziu Apache Maven skonfigurowanym tak aby wykorzystywał platformę Java w wersji 11 lub wyższej. Należy zwrócić uwagę na poprawną identyfikację projektu oraz pakiet wykorzystane w aplikacji.

W ramach zadania należy wykorzystać bibliotekę JUnit oraz Mockito. Do przygotowania asercji zaleca się wykorzystanie biblioteki AssertJ lub Hamcrest.

Należy zaimplementować aplikację pozwalającą na zarządzanie zbiorem encji przechowywanych w pamięci. Tradycyjnie kod powinien zostać rozdzielony na trzy komponenty: repozytorium, serwis i kontroler. Na potrzeby laboratorium zostaną wykorzystane dwa komponenty: repozytorium i kontroler.

Repozytorium należy zaimplementować w taki sposób, że zawiera kolekcję obiektów, którymi zarządza oraz udostępnia metody pozwalające na zapisanie nowego, usunięcie i wyszukiwanie. Kolekcja powinna być przechowywana w pamięci aplikacji. Nie ma potrzeby korzystania tutaj z mechanizmów JPA czy JDBC. Usunięcie i wyszukanie realizowane jest na podstawie przyjętego klucza głównego. Definicję klas podaje prowadzący na zajęciach (przykład na końcu instrukcji).

Specyfikacja repozytorium:

- próba usuniecia nieistniejącego obiektu powoduje IllegalArgumentException,
- próba pobrania nieistniejącego obiektu zwraca pusty obiekt Optional,
- próba pobrania istniejącego obiektu zwraca obiekt Optional z zawartością,
- próba zapisania obiektu, którego klucz główny już znajduje się w repozytorium powoduje IllegalArgumentException.

Kontroler powinien korzystać z repozytorium dostarczonego przez wstrzykiwanie zależności. W przeciwieństwie do repozytorium, metody kontrolera przyjmują i zwracają obiekty String tłumaczone z/na obiekty encyjne.

Specyfikacja kontrolera:

- próba usunięcia nieistniejącego obiektu powoduje zwrócenie obiektu String o wartości done.
- próba usunięcia nieistniejącego obiektu powoduje zwrócenie obiektu String o wartości not found,
- próba pobrania nieistniejącego obiektu powoduje zwrócenie obiektu String o wartości not found,



Platformy Technologiczne

- próba pobrania istniejącego obiektu zwraca obiekt String reprezentujący znaleziony obiekt encyjny,
- próba zapisania nowego obiektu skutkuje wywołaniem metody z serwisu z poprawnym parametrem i zwróceniem obiektu String o wartości done,
- próba zapisania nowego obiektu, którego klucz główny znajduje się już w repozytorium powoduje zwrócenie obiektu String o wartości bad request.

Należy zrealizować następujące zadania:

- Implementacja klasy encyjnej, dto, repozytorium i kontrolera. Przy implementacji kontrolera należy zwrócić uwagę na odpowiednie odwzorowanie zwracanych wartości lub rzucanych wyjątków przez repozytorium. (1 pkt)
- 2. Przetestowanie działa repozytorium. Należy przetestować przynajmniej wszystkie przypadki wymienione wyżej jako specyfikacja repozytorium. (1 pkt)
- 3. Przetestowanie działa kontrolera. Należy przetestować przynajmniej wszystkie przypadki wymienione wyżej jako specyfikacja kontrolera. Testując obiekt kontrolera należy wykorzystać atrapę obiektu repozytorium. (3 pkt)



Platformy Technologiczne

```
Przykład:
public class Mage {
    private String name;
    private int level;
}
public class MageRepository {
    private Collection<Mage> collection;
    public Optional<Mage> find(String name) {
    }
    public void delete(String name) {
        //...
    public void save(Mage mage) {
        //...
    }
}
public class MageController {
    private MageRepository repository;
    public MageController(MageRepository repository) {
        //...
    }
    public String find(String name) {
        //...
    }
    public String delete(String name) {
        //...
    public String save(String name, String level) {
        //...
    }
}
```