Dokumentacja

W dokumentacji, moim zdaniem, jedyne, co trzeba dokładniej opisać, to testowanie programu i kryteria jego uruchomienia, ponieważ wszystkie pozostałe aspekty są realizowane tak samo jak u innych studentów.

Kryteria uruchomienia

* Projekt jest przystosowany do wersji JDK 23. Jeśli macie inną wersję, pamiętajcie, aby zmienić ją w pliku pom.xml.
* Ten projekt korzysta z bazy danych MySQL, więc nie zapomnicie jej utworzyć (kod do jej utworzenia znajduje się w folderze projektu).

Testowanie Postman

1. **GET** /Home

Pobranie strony głównej API.

Wynik: Pomyślne wyświetlenie strony startowej aplikacji.

2. **GET** /Authors

Pobranie listy wszystkich autorów.

Wynik: Wyświetlenie formularza dodawania książki add-book

3. **POST** /Add-author

Dodanie nowego autora do bazy danych.

Wynik: Pomyślne dodanie autora i przekierowanie na stronę z listą autorów.

4. **GET** /Books

Pobranie listy wszystkich książek.

Wynik: Tablica książek z atrybutami.

5. **POST** /Add-book

Dodanie nowej książki do bazy danych.

Wynik: Pomyślne dodanie książki i przekierowanie na stronę z listą książek.

6. **GET** /Book-info

Pobranie szczegółowych informacji o konkretnej książce.

Wynik: Obiekt książki ze wszystkimi jej atrybutami.

7. **GET** /Book-Info\_full

Pobranie pełnych informacji o książce wraz z autorem.

Wynik: zczegółowe informacje o książce i autorze.

8. **GET** /Author-information

Pobranie pełnych informacji o konkretnym autorze.

Wynik: Informacje o autorze, w tym lista jego książek.

9. **GET** /Written-books

Pobranie listy książek napisanych przez konkretnego autora.

Wynik: Tablica książek powiązanych z autorem.

10. **GET** /Author-delete

Usunięcie autora z bazy danych.

Wynik: Pomyślne usunięcie autora i przekierowanie na stronę z listą autorów.

11. **GET** /Book-delete

Usunięcie książki z bazy danych .

Wynik: Pomyślne usunięcie książki i przekierowanie na stronę z listą książek.

12. **GET /**Edit-author

Pobranie formularza do edycji autora.

Wynik: Formularz edycji z bieżącymi danymi autora.

13. **GET /**Edit-book

Pobranie formularza do edycji książki.

Wynik: Formularz edycji z bieżącymi danymi książki.

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

Testowanie Junit

1)Testy dla klasy authorsControlTest

1. testShowAddAuthorForm()

Sprawdzenie, czy metoda showAddAuthorForm dodaje nowego autora do modelu i zwraca poprawny widok formularza dodawania autora.

Wynik: Wyświetlenie formularza dodawania nowego autora add-author

2. testAddAuthor()

Sprawdzenie, czy poprawnie obsługiwane jest dodanie nowego autora.

Wynik: Przekierowanie na stronę z listą autorów po dodaniu.

3. testDeleteAuthor()

Sprawdzenie, czy metoda deleteAuthor usuwa autora po jego ID.

Wynik: Przekierowanie na stronę z listą autorów po usunięciu.

4. testDeleteAuthor\_NonExistingAuthor()

Sprawdzenie, że metoda deleteAuthor nie usuwa autora, jeśli taki autor nie istnieje.

Wynik: Przekierowanie na stronę z listą autorów.

5. testShowEditAuthorForm()

Sprawdzenie, czy formularz edycji dla istniejącego autora jest poprawnie wyświetlany.

Wynik: Wyświetlenie formularza edycji autora.

6. testEditAuthor()

Sprawdzenie, czy poprawnie zapisane są zmiany w edytowanym autorze.

Wynik: Przekierowanie na stronę z listą autorów po edycji.

7. testGetAuthorBooks()

Sprawdzenie, czy poprawna lista książek autora jest przekazywana do modelu.

Wynik: Przekierowanie na stronę z pełnymi informacjami o autorze i jego książkach.

2)Testy dla klasy booksControlTest

1. testGetBooks()

Sprawdzenie, czy metoda getBooks poprawnie przekazuje listę książek do modelu i zwraca odpowiedni widok.

Wynik: Zwrócenie widoku books z dodanymi książkami

2. testShowAddBookForm()

Sprawdzenie, czy formularz dodawania książki jest poprawnie wyświetlany.

Wynik: Wyświetlenie formularza dodawania książki add-book

3. testAddBook()

Sprawdzenie, czy książka została poprawnie dodana do repozytorium.

Wynik: Przekierowanie na stronę z listą książek po dodaniu.

4. testDeleteBook()

Sprawdzenie, czy książka została poprawnie usunięta.

Wynik: Przekierowanie na stronę z listą książek po usunięciu.

5. testShowEditBookForm\_WhenBookExists()

Sprawdzenie, czy formularz edycji dla istniejącego autora jest poprawnie wyświetlany.

Wynik: Wyświetlenie formularza edycji książki edit-book.

6. testShowEditBookForm\_WhenBookNotFound()

Sprawdzenie, czy wyświetlany jest komunikat o błędzie, jeśli książka nie istnieje.

Wynik: Wyświetlenie komunikatu o błędzie i formularza edycji książki.

7. testEditBook()

Sprawdzenie, czy książka została poprawnie edytowana i zapisana.

Wynik: Przekierowanie na stronę z listą książek po edycji.

8. testGetBooksWithoutAuthor()

Sprawdzenie, czy metoda getBooksWithoutAuthor poprawnie zwraca informacje o książce, która nie ma przypisanego autora.

Wynik: Zwrócenie widoku book-info.

9. testGetBooksWithoutAuthor\_WhenBookNotFound()

Sprawdzenie, czy wyświetlany jest komunikat o błędzie, jeśli książka nie istnieje.

Wynik: Wyświetlenie komunikatu o błędzie i przekierowanie na stronę z książkami.

10. testGetAuthorAll()

Sprawdzenie, czy poprawnie są wyświetlane informacje o autorze powiązanym z książką.

Wynik: Zwrócenie widoku z pełnymi informacjami o książce i autorze full\_info\_book..

11. testGetAuthorAll\_WhenBookNotFound()

Sprawdzenie, czy użytkownik jest przekierowywany, gdy książka nie zostaje znaleziona

Wynik: Przekierowanie na stronę z książkami.