CRUD Demo - Wytyczne

> **Wytyczne dla praktykantów realizujących projekt w Departamencie Systemów Informatycznych**

Wstęp

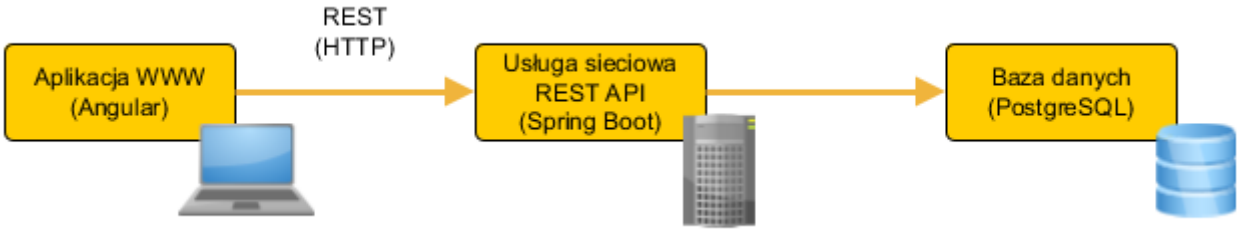
Celem zadania jest wykonanie aplikacji demonstracyjnej z wykorzystaniem następujących technologii:

- **Frontend:**
  - Framework: **Angular**
  - GUI: **Angular Material** lub **ng-bootstrap** (ewentualnie inny uzgodniony z opiekunem)
  - Język programowania: **TypeScript**
  - Standardy: HTML5, CSS3
- **Backend:**
  - Framework: **Spring Boot**
  - Język programowania: Java
- **Baza danych**:
  - **PostgreSQL**

Temat aplikacji (funkcje biznesowe) jest dowolny - zachęcamy do wyboru własnego tematu zgodnego z zainteresowaniami. Warunkiem jest to aby aplikacja realizowała funkcje **CRUD** na obiektach biznesowych przechowywanych w bazie danych Przykładowo, jeśli aplikacja ma służyć do zarządzania zbiorem książek to powinna umożliwiać:

- dodawanie do bazy danych Autora i Tytułu książki
- przypisywanie nowych tytułów do istniejących autorów
- edytowanie atrybutów książki (tytuł, gatunek, rok wydania, itp...)
- usuwanie omyłkowo dodanych książek lub autorów
- wyszukiwanie autorów i tytułów według zadanych kryteriów

Aplikacja ma być zbudowana w architekturze trójwarstwowej, zgodnie z poniższym schematem.



Wymagania

Poniżej zdefiniowano minimalne wymagania dla aplikacji. Jak już wspomniano wcześniej temat aplikacji nie jest narzucony i powinien być wybrany przez praktykanta w uzgodnieniu z opiekunem. Na etapie analizy wymagań należy uzupełnić tę specyfikację o wymagania biznesowe dostosowane do konkretnej dziedziny, którą będzie obsługiwała aplikacja.

Wymagania funkcjonalne

1. Aplikacja prezentuje listę elementów pobranych z bazy danych (typ elementu w zależności od wybranego tematu: np. lista książek, lista kontrahentów, lista samochodów) w formie stronicowanej
2. Aplikacja umożliwia wyszukiwanie elementów na podstawie zadanych kryteriów
3. Aplikacja umożliwia sortowanie listy elementów na podstawie wybranego atrybutu
4. Aplikacja umożliwia dodawanie i edytowanie elementu (dowolna rola)
5. Aplikacja umożliwia usuwanie elementu (tylko *Administrator*)
6. Podczas dodawania lub edytowania elementu należy weryfikować poprawność danych. Walidacja ma być realizowana zarówno po stronie GUI jak i po stronie usługi sieciowej.

Wymagania bezpieczeństwa

7. Logowanie do aplikacji odbywa się przez wpisanie nazwy użytkownika i hasła.
8. Każdy użytkownik ma przypisaną jedną z ról: *Administrator*, *Pracownik*
9. Rola *Administrator* ma dostęp do wszystkich funkcji aplikacji. Rola *Pracownik* ma dostęp tylko do wybranych funkcji.
10. Autoryzacja (weryfikowanie uprawnień) realizowana jest dwuetapowo:
    - po stronie GUI - ukrywanie lub dezaktywowanie poszczególnych przycisków, elementów menu, itp.
    - po stronie usługi sieciowej - ograniczenie dostępu do określonych funkcji REST API w zależności od roli
11. Aplikacja posiada moduł zarządzania użytkownikami dostępny dla roli *Administrator*. Moduł umożliwia:
    - dodanie nowego użytkownika (imię, nazwisko, login, hasło, rola)
    - edycja konta użytkownika (w tym zmiana hasła)
    - dezaktywowanie konta użytkownika

Zadania do realizacji

W ramach prac należy przejść kolejno przez typowe etapy realizacji projektu informatycznego, tj:

1. Analiza wymagań
2. Projekt techniczny
3. Przygotowanie środowiska developerskiego
4. Implementacja
5. Testy i stabilizacja
6. Dokumentacja

Szczegóły dotyczące poszczególnych etapów opisano w poniższych podrozdziałach.

Analiza wymagań

W ramach analizy wymagań należy:

- zapoznać się ze **specyfikacją wymagań**
- przygotować listę pytań do specyfikacji
- przedyskutować pytania i wątpliwości z opiekunem
- notować wszelkie ustalenia - będą one uzupełnieniem specyfikacji wymagań

Celem tego etapu jest pełne zrozumienie wymagań w zakresie wykonywanego systemu tak, aby możliwe było opracowanie projektu technicznego.

Projekt techniczny

Należy opracować w zwięzłej formie projekt techniczny rozwiązania, który powinien zawierać:

- koncepcję realizacji - kilka zdań opisujący co zostanie zrealizowane
- lista i krótki opis funkcji, które będą zrealizowane a aplikacji
- szkice najważniejszych ekranów aplikacji (mogą być odręczne rysunki na kartce)
- model danych przechowywanych w bazie danych (tabele, pola, relacje)

Przygotowanie środowiska

W tym etapie należy zainstalować na stacji roboczej i skonfigurować wszystkie komponenty wymagane do implementacji i testowania rozwiązania:

- Silnik bazy danych **PostgreSQL** i narządzie do zarządzania bazą i wykonywania zapytań (np. **pgAdmin**)
- Java (JDK)

- IDE (Java i TypeScript) - wybór według własny preferencji, np:
  - **IntelliJ IDEA**
  - **Eclipse**
  - **Visual Studio Code**
- Narzędzie do kontroli wersji oprogramowania: **git**
- Środowisko dla **Angular**
- Narzędzia do budowania projektów Java: **Maven** lub **Gradle**

> ! Wymienione wyżej komponenty mogą wymagać zainstalowania dodatkowych bibliotek lub programów. Są one szczegółowo opisane w instrukcjach instalowania tych komponentów.

Implementacja

Ten etap obejmuje pisanie kodu aplikacji WWW i usługi sieciowej oraz wymaganych skryptów SQL. Jeśli programista nie ma doświadczenia w technologiach, które mają być wykorzystane przy implementacji (w szczególności Angular i Spring) to etap ten należy rozpocząć od nauki i ćwiczeń praktycznych. Celem nauki ma być przyswojenie danej technologii w stopniu wymaganym do rozpoczęcia pracy nad projektem.

Do nauki można wykorzystać m.in. następujące materiały:

- Angular
  - **Getting Started**
  - **Tutorial**
- Spring Boot
  - **Spring Guides**
- HTML + CSS
  - **HTML5 Tutorial**
  - **CSS Tutorial**

W trakcie implementacji należy konsekwentnie stosować wskazówki dotyczące stylu kodowania:

- **Angular Style Guide**
- **Google Java Style Guide**

Ważnym elementem prac implementacyjnych jest komentowanie kodu źródłowego. Należy stosować komentarze opisujące co realizują poszczególne bloki kodu źródłowego. Komentarze powinny w zwięzły sposób wyjaśniać krok realizowanego algorytmu, spodziewany rezultat i sposób implementacji. Należy też zamieszczać wszelkie uwagi, które nie wynikają w sposób oczywisty z samego kodu źródłowego.

Przykład złego komentarza

```js
// przypisz 5 do zmiennej maxItems
maxItems = 5;
```

Przykład dobrego komentarza

```js
// maksymalna liczba elementów listy - wartość wynika z wymagań zamawiającego
maxItems = 5;
```

Należy także dokumentować klasy i metody przy użyciu standardów:

- **JSDoc** - dla kodu TypeScript (Angular)
- **Javadoc** - dla kodu Java

Testy i stabilizacja

W ramach tego etapu należy

- przygotować i uzgodnić z opiekunem scenariusze testowe (lista przypadków testowych w zwięzłej formie)
- prowadzić testy zgodnie ze scenariuszami i ewentualnie uzupełniać scenariusze o nowe przypadki testowe
- poprawiać błędy wykryte w czasie testów

Należy rozważyć automatyzację testów, np. przy użyciu narzędzi dołączonych do Angular - **Angular Testing**.

Dokumentacja

Należy opracować i przedstawić opiekunowi do weryfikacji następujące typy dokumentacji:

- **Projekt techniczny**
- Scenariusze testów
- Instrukcję wdrożeniową opisującą
  - kompilowanie kodu źródłowego
  - instalowanie poszczególnych komponentów
  - dostępne parametry i opcje konfiguracyjne