Zadania dla kandydatów

Instrukcja

1. Zadania należy wykonać w ciągu 3 dni roboczych.
2. Wyniki zadań należy przesłać na adres praca@ktsoft.pl jako archiwum zip lub linki do github.
3. Każde zadanie powinno być w osobnym folderze. Nazewnictwo folderów i plików powinno nie budzić żadnych wątpliwości co do ich zawartości.

Zadanie Blazor

1. Opis zadania

Stwórz aplikację „BlazorNotes”, w której użytkownik może:

1. Przeglądać listę notatek.
2. Dodawać nową notatkę (tytuł + treść).
3. Edytować istniejącą notatkę.
4. Usuwać notatki.

Dane mają być przechowywane w relacyjnej bazie danych SQL (np. lokalny SQL Server lub SQLite). Aplikacja powinna korzystać z EF Core i DI do zarządzania dostępem do danych.

---

2. Wymagania techniczne

- **Platforma**: .NET 8+
- **Typ projektu**: Blazor Web App
- **Baza danych**: SQL Server / SQLite
- **ORM**: Entity Framework Core
- **DI**: Wstrzykiwanie serwisów za pomocą wbudowanego kontenera
- **UI**:
    - Blazor Components (.razor)
    - Prosty układ i podstawowe style CSS
- **Architektura**:
    - Model–widok–serwis (Data Models, Services, Components)
    - Interfejsy dla serwisów (np. INoteService)
- **Dodatkowo:** walidacja formularzy, prosty test jednostkowy serwisu

---

Zadanie CSS

1. **Layout dashboardu z CSS Grid**
    - Stwórz responsywny układ „dashboardu” z nagłówkiem, bocznym panelem i główną sekcją.
    - W szerokich oknach boczny panel (aside) zajmuje 1/4 szerokości, w wąskich (< 800 px) – „chowa się” pod przyciskiem hamburger.
    - Użyj CSS Grid oraz media queries.
2. **Tryb ciemny z wykorzystaniem CSS custom properties**
    - Zdefiniuj zmienne dla kolorów (tło, tekst, akcenty).
    - Przyciskiem w nagłówku przełączaj klasę dark na elemencie <body>, by zmienić tryb.
    - Upewnij się, że przy ładowaniu strony czy tryb (z localStorage) jest zachowany.
3. **Animacja „karuzeli obrazów”**
    - Mając kontener .carousel z kilkoma <img>, zaimplementuj automatyczną przewijaną animację (infinite loop) – fade-in/fade-out kolejnych obrazów co 4 s.
    - Użyj keyframes i właściwości animation.
4. **Zaawansowane selektory i pseudoelementy**
    - Zbuduj przycisk .btn-outline który:
        - Ma ramkę i przezroczyste tło,
        - Po hoverie tworzy animowaną obwódkę „rosnącą” od środka (użyj ::before lub ::after).
    - Nie wstawiaj dodatkowych elementów w HTML – tylko CSS.
5. **Dostępność i optymalizacja**
    - Dodaj do strony „skip link” (<a href="#main" class="skip-link">) widoczny tylko przy fokusie,
    - Zaimplementuj focus-visible dla formularza w headerze (pole wyszukiwania) – wyraźne obramowanie.

Zadanie SQL

Załóż, że masz bazę firmy z tabelami Departments, Employees, Projects i EmployeeProjects. Skrypty tworzące bazę i dane w załączeniu.

Zadania do wykonania

1. Wyświetl wszystkich pracowników posortowanych rosnąco po nazwisku, a następnie po imieniu.
2. Dla każdego działu policz liczbę pracowników i wypisz nazwy działów, posortowane malejąco po liczbie pracowników.

3. Oblicz sumę wynagrodzeń w każdym dziale, wypisz tylko te działy, gdzie łączna kwota przekracza 150 000, posortuj malejąco.
4. Lista wszystkich projektów wraz z liczbą przypisanych do nich pracowników (wypisz również projekty bez pracowników).
5. Wyciągnij imiona i nazwiska pracowników, których pensja jest wyższa od średniej pensji wszystkich pracowników (użyj podzapytania).
6. Korzystając z CTE, przygotuj zestawienie: EmployeeID, FirstName, LastName, DepartmentName, Salary — najpierw CTE pobierające średnią pensję w każdym dziale, a potem dołącz tabelę pracowników i wyświetl tylko tych, którzy zarabiają powyżej średniej w swoim dziale.
7. (Bonus) *opcjonalnie* — jeżeli zaimplementujesz kolumnę ManagerID, napisz zapytanie z rekurencyjnym CTE, które dla zadanego ManagerID = 1 wypisze całą strukturę podległych pracowników (hierarchię).

Rozwiązanie powinno zawierać treść zapytania SQL oraz zwrócony wynik danych.