PWr Spotkanie 11
Aplikacje webowe na platformie .NET
Laboratorium – **Lista 11**

Wstęp

Pamiętanie danych we właściwościach statycznych najczęściej nie jest dobrym pomysłem. W liście zadań nr 8 (która skupiała się na regułach routingu) gra tylko wygląda na poprawnie działającą. Warto ją przetestować uruchamiając co najmniej dwie różne przeglądarki internetowe (symulując działanie dwóch użytkowników aplikacji). Szybko się okaże, że losowanie wykonane w jednej przeglądarce ma wpływ na grę w drugiej przeglądarce. Powodem jest właśnie użycie elementów statycznych w kontrolerze. Aby każdy użytkownik Waszej aplikacji grał w niezależną grę, należy użyć **zmiennych sesyjnych**.

Lista zadań

Zadania 3 i 4 wykonać w wybranej architekturze: MVC lub strony Razor-owe. W zadaniu nie są wprost przedstawione klasy, które należy stworzyć do obsługi koszyka towarów, jednak powinny się one pojawić w kodzie.

.

1. Zmodyfikować grę-zgadywankę z listy 8 tak, aby każdy użytkownik grał niezależnie od innego. Użyj **zmiennych sesyjnych**. Podpowiedź: użycie zmiennych sesyjnych w getterach i setterach dla wcześniejszych pól/właściwości statycznych pozwala uniknąć modyfikacji pozostałego kodu tej gry.
2. Wykonać listę zadań numer 10 przy użyciu **stron Razora** (zamiast użycia wzorca MVC).
3. Do rozwiązania zadania z listy 10 dla `ShopController`-a dopisać pamiętanie koszyka zakupów **w ciasteczkach** (dla **każdego artykułu** stworzyć **oddzielne** ciasteczko). Strona pokazująca towary (lista 10 zadanie 6) przy każdym towarze powinna mieć przycisk/link, aby dodać towar do koszyka. Jeśli użytkownik wybierze kilka razy ten sam towar będzie on dodany w kilku sztukach (czyli pamiętać należy w ciasteczkach identyfikator towaru (w kluczu) i jego liczność w koszyku (w wartości), np. klucz `article`**15**, wartość **2**). Każdy użytkownik (np. można przetestować poprzez użycie dwóch różnych przeglądarek) ma mieć swój własny, niezależny koszyk. Zawartość koszyka powinna być pamiętana przez tydzień, nawet jeśli użytkownik zamknie przeglądarkę.
4. W ramach sklepu powinien istnieć przycisk/link do koszyka w menu. Za jego pomocą można oglądnąć zawartość koszyka na nowej stronie (widok podobny jak dla widoku sklepu, ale z licznościami towarów w koszyku). Na tej nowej stronie można również zwiększyć, zmniejszyć liczbę sztuk danego towaru, ewentualnie całkiem go usunąć z koszyka. Strona, po każdej takiej zmianie, **na bieżąco** przelicza jaki jest **całkowity koszt** towarów w koszyku. Jeśli nie ma nic w koszyku, strona powinna w widoczny sposób poinformować o tym użytkownika.

   W widoku koszyka dodanie/usunięcie sztuki towaru powoduje zapytanie GET do odpowiedniej akcji kontrolera, który zmodyfikuje zawartość koszyka i pokaże jego nową zawartość.

Pytanie: Jak operacje na koszyku mają się do operacji usuwania towaru/kategorii, którą **inny** użytkownik Waszej aplikacji może wykonać **w międzyczasie**?

Data I: Spotkanie 12 (max 100 punktów)
Data II: Spotkanie 13 (max 80 punktów)
Data III: Spotkanie 14 (max 50 punktów)