Projekt roczny — implementacja

Czas: 2 × 45 min | praca indywidualna | Wykład 1.30

Cel: Zrealizujesz aplikację WPF zgodnie ze specyfikacją z zadania 32, pracując małymi krokami z historią w Git, i przeprowadzisz przegląd kodu w parach.

Po zajęciach potrafisz:

- implementuje aplikację zgodnie z wymaganiami
- rozdziela logikę od interfejsu
- dokumentuje postęp commitami z opisowymi komunikatami
- przeprowadza i przyjmuje przegląd kodu

Sprzęt i materiały

- komputer z Windows 11 i dostępem do internetu
- pendrive lub konto w repozytorium do zapisania projektu

Oprogramowanie

- **Visual Studio Community 2026 (obciążenie „Programowanie aplikacji klasycznych platformy .NET")** — darmowe dla uczniów i nauczycieli; w pracowni zainstalowane visualstudio.microsoft.com/pl/downloads
- **.NET 10 SDK (LTS)** — instalowany razem z Visual Studio; sprawdzenie: `dotnet --version` dotnet.microsoft.com/pl-pl/download/dotnet/10.0

Przygotowanie przed zajęciami

- Zatwierdzona specyfikacja z zadania 32 i repozytorium projektu.

Bezpieczeństwo

- Zapisuj projekt w swoim folderze (np. `C:\Users\<login>\source\repos`), a na koniec zajęć skopiuj go na pendrive lub wyślij do repozytorium.
- Nie uruchamiaj programów i pakietów z nieznanych źródeł; pakiety NuGet instaluj tylko te, które wskazuje zadanie.

Przebieg zadania

1 Klasy logiki

Zacznij od części bez interfejsu.

1. Zaimplementuj klasy logiki z projektu; walidacja danych w klasach (wyjątki lub komunikaty).
2. Zaimplementuj zapis i odczyt danych (JSON lub CSV) w osobnej klasie.
3. Sprawdź logikę krótkim programem konsolowym lub pierwszymi testami. Commit.

2 Okno główne

Interfejs według makiety.

1. Zbuduj układ okna kontenerami (bez sztywnych Margin).
2. Podłącz zdarzenia — code-behind tylko przekazuje dane do logiki i wyświetla wynik.
3. Dodaj style w App.xaml (zadanie 29).
4. Commit po każdym działającym wymaganiu, z numerem wymagania w opisie (np. „W3: filtr po gatunku”).

3 Kolejne okna i pliki

Pozostałe wymagania.

1. Okno dialogowe (np. dodawanie lub edycja elementu) otwierane ShowDialog z przekazywaniem danych.
2. Wczytywanie danych przy starcie i zapis przy zamykaniu lub na żądanie; obsługa błędów plików.
3. Uzupełnij tabelę realizacji wymagań.

4 Przegląd kodu w parach

Ostatnie 20 minut.

1. Zamień się z kolegą/koleżanką: przejrzyjcie nawzajem kod według listy: nazwy, podział logika/okno, obsługa błędów, powtórzenia kodu, komentarze.
2. Zapisz 3 uwagi otrzymane i 3 przekazane.
3. Popraw co najmniej jedną uwagę i zrób commit „Poprawki po przeglądzie”.

W sprawozdaniu umieść

- ☐ tabela realizacji wymagań
- ☐ historia commitów (`git log --oneline`)
- ☐ uwagi z przeglądu kodu i wprowadzone poprawki
- ☐ zrzuty działających okien

Pytania kontrolne

1. Które wymaganie zmieniłeś względem specyfikacji i dlaczego?
2. Która uwaga z przeglądu kodu była najbardziej przydatna?
3. Jak sprawdzasz, czy code-behind nie zawiera logiki, która powinna być w klasach?

Kryteria oceny

Kryterium	Punkty
Realizacja wymagań (wg tabeli)	8
Podział logika / interfejs	4
Historia commitów	3
Przegląd kodu i poprawki	3
Pytania	2
Razem	**20**

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).