Projekt zaliczeniowy

Programowanie obiektowe
Rok akademicki 2025/2026

Autorzy:
Kamil Celadyn
Oskar Fryc
Mykhailo Bondar

System Ewidencji Pracowników i Obliczania Wynagrodzeń

Projekt jest aplikacją typu Desktop (WPF) służącą do obsługi działu kadr w małym przedsiębiorstwie. System umożliwia ewidencjonowanie pracowników różnych typów (etatowych oraz zleceniobiorców), automatyczne obliczanie ich wynagrodzeń zgodnie z typem umowy oraz trwały zapis danych w formacie JSON. Głównym celem projektowym było praktyczne zastosowanie paradygmatów programowania obiektowego.

Podział ról

Kamil Celadyn **(Backend Developer):**

- Stworzenie abstrakcyjnej logiki biznesowej (`Pracownik`, `SystemKadrowy`).
- Implementacja mechanizmów dziedziczenia i polimorfizmu (`PracownikEtatowy`, `Zleceniobiorca`).
- Obsługa kolekcji danych i serializacja JSON.

Oskar Fryc **(Frontend Developer):**

- Projekt i implementacja interfejsu graficznego (WPF - `MainWindow.xaml`).
- Logika interakcji użytkownika (`MainWindow.xaml.cs`).
- Obsługa zdarzeń przycisków i wyświetlanie danych w liście.

Mykhailo Bondar **(Quality Assurance & Documentation):**

- Implementacja obsługi błędów i własnych wyjątków (`KadryException`).
- Stworzenie interfejsów dodatkowych (`IBonusowalny`).
- Przygotowanie Diagramu UML oraz pełnej dokumentacji technicznej.

Opis klas

1. **Klasa `Pracownik` (Abstrakcyjna)**
   - **Rola:** Fundament całego systemu. Definiuje wspólne cechy wszystkich pracowników (imię, nazwisko, PESEL).
   - **Uzasadnienie:** Jako klasa abstrakcyjna wymusza na klasach pochodnych implementację metody `ObliczPensje()`. Implementuje interfejsy `IComparable` (sortowanie) i `IEquatable` (unikalność).
   - **Modyfikatory:** Właściwość `Pesel` posiada publiczny getter i setter z logiką walidacji, co zapewnia hermetyzację (enkapsulację) danych – niepoprawny PESEL nie zostanie przypisany.
2. **Klasa `PracownikEtatowy`**
   - **Rola:** Reprezentuje pracownika na umowie o pracę.
   - **Uzasadnienie:** Rozszerza klasę bazową o pole `PensjaZasadnicza`. Implementuje interfejs `IBonusowalny`, co pozwala na przyznawanie premii, czego nie mają zleceniobiorcy.
3. **Klasa `Zleceniobiorca`**
   - **Rola:** Reprezentuje pracownika na umowie cywilnoprawnej.
   - **Uzasadnienie:** Oblicza wypłatę w inny sposób (stawka * godziny). Wprowadza dodatkowe walidacje w konstruktorze (stawka nie może być ujemna).
4. **Klasa `SystemKadrowy`**
   - **Rola:** Kontroler (Logic Layer). Zarządza kolekcją `List<Pracownik>`.
   - **Uzasadnienie:** Oddziela logikę przechowywania danych od interfejsu graficznego. Odpowiada za dodawanie (z sprawdzaniem duplikatów), sortowanie oraz serializację danych.
5. **Klasa `MainWindow` (GUI)**
   - **Rola:** Warstwa prezentacji (WPF).
   - **Uzasadnienie:** Pozwala użytkownikowi na interakcję z systemem bez konieczności edycji kodu. Obsługuje błędy wyrzucane przez niższe warstwy (np. `KadryException`) i wyświetla je w czytelnych oknach dialogowych.

Opis funkcjonalności

- **Dodawanie pracowników:** Użytkownik wypełnia formularz. System automatycznie rozpoznaje typ umowy i dostosowuje pola (ukrywa pole "Godziny" dla etatu). Walidacja blokuje dodanie pracownika ze złym numerem PESEL. Klasa odpowiedzialna: `MainWindow` -> `SystemKadrowy`.

- **Obliczanie wypłat:** Lista pracowników automatycznie wyświetla obliczoną pensję netto/brutto wykorzystując polimorficzną metodę `ObliczPensje()`.
- **Trwałość danych:** Przycisk "Zapisz" serializuje listę obiektów do pliku JSON, co pozwala na odtworzenie stanu aplikacji po ponownym uruchomieniu. Klasa odpowiedzialna: `SystemKadrowy` (metody `System.Text.Json`).
- **Sortowanie:** Możliwość posortowania listy pracowników alfabetycznie po nazwisku. Klasa odpowiedzialna: `Pracownik` (metoda `CompareTo`).

Diagram klas