

Wymagania na projekt

1. Przekazać należy archiwum z kompletną solucją dla JetBrains Rider zawierającą pełne kody źródłowe obu projektów wraz z (opcjonalnie) zależnościami pakietów NuGet oraz bibliotek zewnętrznych umożliwiając ocenę tak statyczną (code review) jak i dynamiczną (kompilacja oraz uruchomienie).
2. Target framework: net6.0; konsola / terminal jako interfejs użytkownika (input / output); nie należy zakładać wykorzystania konkretnego systemu operacyjnego / platformy do uruchomienia rozwiązania (np. MacOS).
3. W ramach opisanej w sprawozdaniu demonstracyjnej ścieżki uruchomienia rozwiązania musi nastąpić przynajmniej jednokrotna, obustronna (duplex) komunikacja międzyprocesowa pomiędzy projektami (tj. z $A \rightarrow B$ i z $B \rightarrow A$). Jeżeli komunikacja wyzwalana jest manualnie, tj. odbiorca komunikatu wymaga ręcznego działania w celu pobrania danych, maksymalna możliwa ocena za takie rozwiązanie wynosi 4,5.
4. Dwie płaszczyzny oceny:
 - projekt solucji, sprawozdanie i działanie rozwiązania jako całość;
 - kod źródłowy obu projektów z wykorzystaniem Scorecard oceniany osobno.

Scoreboard

Ocena db.

- Dwustronna komunikacja międzyprocesowa

Programowanie obiektowe

- Kontrakty: klasy abstrakcyjne, interfejsy
- Polimorfizm: uogólnianie typów, przesłanianie metod
- Typy wyliczeniowe
- Zaprojektowanie typów generycznych, kowariancja i kontrawariancja

Metadane

- Atrybuty: własne lub wykorzystanie istniejących

Programowanie funkcyjne

- Delegaty: metody anonimowe, wyrażenia lambda
- Kolekcje danych i Language Integrated Query

Programowanie asynchroniczne

- Metody async, synchronizacja await lub Task API

+ 1/2 do oceny

Przynajmniej 3 punkty z poniższej listy:

- Wykorzystanie wyrażeń regularnych
- Implementacja i prawidłowe wykorzystanie interfejsu IDisposable
- Extension methods dla typów z Base Class Library
- Użycie CancellationToken
- Wykorzystanie synchronization primitives

+ 1/2 do oceny

- Komunikacja międzyprocesowa jest w pełni automatyczna, tj. nie wymaga ręcznego wyzwalania pobierania komunikatów po stronie odbiorcy.
-

– 1/2 do oceny

- Nie uwalnianie zasobów z interfejsem IDisposable, jeżeli jest on dostępny w wykorzystywanych obiektach.

Wymagania dla sprawozdania

1. Ogólny opis zaimplementowanego rozwiązania / use case.
 2. (O ile występują) specyficzne wymagania odnośnie konfiguracji środowiska w celu uruchomienia i dynamicznej oceny rozwiązania.
 3. Opis demonstracyjnej ścieżki uruchomienia rozwiązania.
 4. Dla każdego z projektów osobno: wskazanie w kodzie źródłowym wraz z krótkim uzasadnieniem dla zastosowania mechanizmu każdego pokrytego ze Scorecard punktu.
-

ROZWIĄZANIE

Ogólny opis zaimplementowanego rozwiązania

Rozwiązanie symuluje działania dziennika elektronicznego. Występują w nim dwa projekty: Klient oraz Serwer. Procesy komunikują się ze sobą poprzez potok nazwany (named pipe). Proces klienta jest interfejsem użytkownika. Instrukcje wprowadzane przez użytkownika są przesyłane do procesu serwera. Ten, realizuje je poprzez operacje na bazie danych i wynik tych działań zwraca. Klient przekazuje je użytkownikowi.

W celu zbudowania poprawnie działającego projektu została zbudowana relacyjna baza danych. Jej tabele są przechowywane w formie plików JSON. Przykładowa baza danych jest tworzona poprzez użycie skryptu.

System umożliwia zalogowanie się na jedno z czterech typów konta: ucznia, rodzica, nauczyciela oraz admina. Konto ucznia umożliwia wyświetlenie przedmiotów oraz ocen, do których jest on przypisany. Konto rodzica umożliwia identyczne działanie – wyświetlenie ocen dziecka. W ramach uproszczenia, jeden rodzic posiada w systemie jedno dziecko. Konto nauczyciela umożliwia przeglądanie uczniów oraz ich ocen w ramach klasy oraz przedmiotu, do którego jest przypisany ten nauczyciel. Może on im również dodawać oraz usuwać oceny. Konto administratora umożliwia dodawanie uczniów oraz usuwanie uczniów z klas. Dodatkowo, w procesie klienta wyświetlany jest czas zalogowania użytkownika.

Dane do logowania można znaleźć w pliku accounts.json. Dane do logowania dla admina to login: A436235, password: admin.

Specyficzne wymagania odnośnie konfiguracji

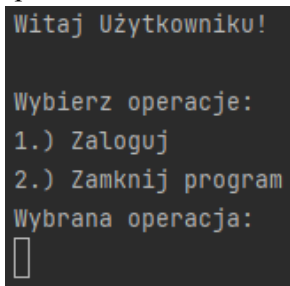
Rozwiązanie wykorzystuje jeden NuGet. Jest nim Newtonsoft.Json.

W feedbacku, wspomniał Pan, aby skorzystać z Build Eventsów i kopiować bazę danych do katalogu z zbudowanym rozwiązaniem. Udało nam się to zrobić, jednak nie wiedzieliśmy, co zrobić, aby po zakończeniu działania programu ta skopiowana baza danych została przeniesiona do katalogu z oryginalną bazą, tak aby nadpisać stare dane nowymi (np. w przypadku, gdy zostanie dodany nowy uczeń lub usunięta jakaś ocena), dlatego zostaliśmy przy starym rozwiązaniu.

Demonstracyjna ścieżka uruchomienia

Dla konta typu uczeń:

1. Ekran startowy po uruchomieniu obu procesów:



```
Witaj Użytkowniku!  
  
Wybierz operacje:  
1.) Zaloguj  
2.) Zamknij program  
Wybrana operacja:  
█
```

2. Wybranie i zalogowanie się przykładowym kontem:

```
{
  "login": "S534888",
  "password": "bm04xcoag3",
  "subaccount": "S3"
},
```

```
Wybierz operacje:
1.) Zaloguj
2.) Zamknij program
Wybrana operacja:
1

Podaj login:
S534888

Podaj haslo:
bm04xcoag3
```

3. Okno zalogowanego ucznia:

```
Wybierz polecenie:
1.) Wyświetl oceny ucznia
2.) Wyloguj
1
```

4. Wybranie opcji nr 1, tj. wyświetlenie ocen:

```
Uczen: Szarlina Brzozowski

Biology: 3 4 6 5 2 2
Chemistry: 5 2 4 6 2
Computer science: 4 6 5 4 2
English: 4 4 6 4
Geography: 3 4 2 1 5
History: 5 1 2
Mathematics: 3 6 2 1
PE: 3 3 3 4
Physics: 4 1 5 6 5 5 3
Polish: 1 1 6 6 5 2 3 3 6 1

Weisnij dowolny przycisk, aby wrocic do menu.
```

Dla konta typu nauczyciel:

1. Tak jak w kroku wcześniejszym:

```
{
  "login": "T418250",
  "password": "yccy1rcfps",
  "subaccount": "T72"
},
```

```
Witaj Użytkowniku!

Wybierz operacje:
1.) Zaloguj
2.) Zamknij program
Wybrana operacja:
1

Podaj login:
T418250

Podaj hasło:
yccy1rcfps
```

2. Okno zalogowanego nauczyciela:

```
Uzytkownik: Ronald Tomczyk

Wybierz polecenie:
1.) Wyświetl oceny klasy
2.) Wyloguj
1
```

3. Po wybraniu opcji 1 ukazują nam się klasy, w których uczy nauczyciel:

```
Prowadzone klasy:
1.) 2c
2.) 8d
Wprowadz numer klasy (inny numer spowoduje powrot do menu)
2
```

4. W klasie 8d prowadzi tylko przedmiot Fizyka:

```
Prowadzone przedmioty:
1.) Physics
Wprowadz numer przedmiotu (inny numer spowoduje powrot do menu)
1
```

5. Lista uczniów i ich ocen z tego przedmiotu:

```
8d - Physics:
1. Baldwin Rogala 1 2 1
2. Tomisława Tkacz 6 3 6 2 6 2 4
3. Huberta Skowron 4 3 3 2 2 4 1 4
4. Danuta Cebula 1 1
5. Abdon Markowski 5 1
6. Pabian Rak 3 6 5 2 2 5
7. Danko Lewicki 1
8. Zaira Kulesza 2 3 5 3 2 1 5 6 1 3
9. Jutta Sikora 6 3 3
10. Rufus Olejnik 6 1 2
11. Idosława Cebula 2 2 3 5 6 5 4 1 3
12. Karima Czajka 3 4 2 4
13. Siemowit Gajda 6 4 4
14. Teofil Konieczna 2 3 1
15. Pakosław Radomski 1 3 3 6 1 2 5
16. Alfons Kula 2 2 1 3 4 4
17. Hildegarda Szymczak 1 5 6 5 2 4 5 5 3 2
18. Maciej Grochowski 4 6
Wprowadz numer ucznia (inny numer spowoduje powrot do menu)
6
```

6. Po wybraniu ucznia Pabian Rak wybieram opcję pierwszą – dodaj ocenę:

```
Oceny ucznia: Pabian Rak
1. 3
2. 6
3. 5
4. 2
5. 2
6. 5
Co chcesz zrobic
1. Dodaj ocene
2. Usun ocene
3. Wroc do menu
1
```

```
Wpisz ocene z zakresu 1-6 aby ja dodac (wybor innej liczby spowoduje powrot do menu):
5
```

7. Po dodaniu oceny 5 uczniowi Pabian Rak powrót do menu:

```
Uzytkownik: Ronald Tomczyk

Wybierz polecenie:
1.) Wyświetl oceny klasy
2.) Wyloguj

```

8. Stan klasy po dodaniu oceny (nowa ocena przy pozycji Pabian Rak):

```
8d - Physics:
1. Baldwin Rogala 1 2 1
2. Tomisława Tkacz 6 3 6 2 6 2 4
3. Huberta Skowron 4 3 3 2 2 4 1 4
4. Danuta Cebula 1 1
5. Abdon Markowski 5 1
6. Pabian Rak 3 6 5 2 2 5 5
7. Danko Lewicki 1
8. Zaira Kulesza 2 3 5 3 2 1 5 6 1 3
9. Jutta Sikora 6 3 3
10. Rufus Olejnik 6 1 2
11. Idosława Cebula 2 2 3 5 6 5 4 1 3
12. Karima Czajka 3 4 2 4
13. Siemowit Gajda 6 4 4
14. Teofil Konieczna 2 3 1
15. Pakosław Radomski 1 3 3 6 1 2 5
16. Alfons Kula 2 2 1 3 4 4
17. Hildegarda Szymczak 1 5 6 5 2 4 5 5 3 2
18. Maciej Grochowski 4 6
Wprowadz numer ucznia (inny numer spowoduje powrot do menu)
█
```

Scoreboard dla procesu Servera

	Podpunkt	Uzasadnienie
Ocena db.	Dwustronna komunikacja międzyprocesowa	Server -> Program.cs Proces serwera przekazuje informacje za pomocą dwuleksowego potoku nazwanego w trybie Byte.
	Kontrakty: klasy abstrakcyjne, interfejsy	-
	Typy wyliczeniowe	Server -> RequestService -> RequestType.cs Typ wyliczeniowy RequestType służy do identyfikowania jakiego typu zapytanie będzie obsługiwane.
	Zaprojektowanie typów generycznych, kowariancja i kontrawariancja	Server -> DataBaseClient.cs -> GetAll<T> Metoda GetAll zwraca z bazy wszystkie elementy, które są typu T. Server -> DataBaseClient.cs -> GetNameAndSurname Uzyskany obiekt – ucznia, nauczyciela, rodzica lub admina – konwertowany (kowariancja) jest na typ ogólniejszy – Person. Nie ma konfliktu zwracanego typu.
	Atrybuty: własne lub wykorzystanie istniejących	Server -> Users.cs

		Przy klasach w tym pliku wykorzystane są atrybuty, które wskazują przy serializacji i deserializacji pliku JSON, do których pól obiektu ma odnosi się dana właściwość pliku JSON.
	Delegaty: metody anonimowe, wyrażenia lambda	Server -> DataBaseClient.cs Do wykonywania zapytań klienta wykorzystywany jest LINQu. Przy stosowaniu go często pojawia się wyrażenie lambda.
	Kolekcje danych i Language Integrated Query	Server -> DataBaseClient.cs Do wykonywania zapytań klienta wykorzystywane są kolekcje danych (List<>, Tuple<>) oraz LINQu.
	Metody async, synchronizacja await lub Task API	Server -> DataBaseClient.cs -> GetStudentsAndTheirGradesFromSpecyfifSubject W metodzie tej dla konkretnego przedmiotu są wyłuskiwane oceny wszystkich uczniów z konkretnej klasy. Dzieje się to przy użyciu Tasków. Kolejne Taski otrzymują zadanie pobrania ocen dla pojedynczego studenta. Następnie następuje oczekiwanie na zakończenie pracy każdego z Tasków.
+ 1/2 do oceny	Wykorzystanie wyrażeń regularnych	-
	Implementacja i prawidłowe wykorzystanie interfejsu IDisposable	-
	Extension methods dla typów z Base Class Library	-
	Użycie CancellationToken	-
	Wykorzystanie synchronization primitives	-
+ 1/2 do	Komunikacja międzyprocesowa jest w pełni automatyczna, tj. nie wymaga ręczne wyzwalania pobierania komunikatów po stronie odbiorcy	Server -> Program.cs Server oczekuje w pętli na kolejne zapytania ze strony klienta. Po dostaniu go, realizuje je i odsyła.
- 1/2 do	Nie uwalnianie zasobów z interfejsem IDisposable, jeżeli jest on dostępny w wykorzystywanych obiektach	-

Scoreboard dla procesu Klienta

	Podpunkt	Uzasadnienie
Oce	Dwustronna komunikacja międzyprocesowa	Client -> Program.cs

		Proces serwera przekazuje informacje za pomocą dwukrotnego potoku nazwanego w trybie Byte.
	Kontrakty: klasy abstrakcyjne, interfejsy	Client -> Sessions -> Session.cs Klasa Session jest klasą abstrakcyjną. Dziedziczą po niej konkretne rodzaje sesji.
	Typy wyliczeniowe	Client -> RequestService -> RequestType.cs Typ wyliczeniowy RequestType służy do tego, aby serwer, po odebraniu zapytania od klienta wiedział jakiego typu jest to zapytanie.
	Zaprojektowanie typów generycznych, kowariancja i kontrawariancja	-
	Atrybuty: własne lub wykorzystanie istniejących	Client -> RequestService -> ResultTypes.cs Atrybuty są tu wykorzystane, aby wskazać że obiekty zaimplementowanych tam klas są serializowalne.
	Delegaty: metody anonimowe, wyrażenia lambda	Client -> Sessions -> TeacherSession.cs -> ChooseStudent Aby zapis był krótszy użyto na kolekcji <i>studentGrades</i> metody <i>ForEach</i> , która na kolekcji wykonuje jakąś akcję. W tym przypadku jest to drukowanie ocen.
	Kolekcje danych i Language Integrated Query	-
	Metody async, synchronizacja await lub Task API	Client -> Sessions -> Session.cs Task aktualizujący czas zalogowania użytkownika, a po wylogowaniu się wypisujący czas sesji na konsolę.
+ 1/2 do oceny	Wykorzystanie wyrażeń regularnych	Client -> Program.cs Wyrażenia regularne są tu użyte w celu sprawdzenia poprawności wprowadzanego przez użytkownika loginu. Dzięki temu unikamy dodatkowego przeszukiwania bazy, bo już na poziomie procesu klienta weryfikujemy poprawność pierwszego elementu z pary <login, hasło>.
	Implementacja i prawidłowe wykorzystanie interfejsu IDisposable	-
	Extension methods dla typów z Base Class Library	Client -> Sessions -> Session.cs W pliku Session.cs oprócz klasy Session znajduje się klasa statyczna SessionExtensions. Zawiera ona jedną metodę <i>Number</i> , która konwertuje podany jako argument string na liczbę.

	Użycie CancellationToken	Client -> Sessions -> Session.cs CancellationToken jest tu użyty do przerywania aktualizacji czasu zalogowania użytkownika w systemie.
	Wykorzystanie synchronization primitives	-
$+ \frac{1}{2} \text{ do}$	Komunikacja międzyprocesowa jest w pełni automatyczna, tj. nie wymaga ręczne wyzwalania pobierania komunikatów po stronie odbiorcy	Client -> Program.cs Client wysyła zapytania, po czym oczekuje na odpowiedź.
$- \frac{1}{2} \text{ do}$	Nie uwalnianie zasobów z interfejsem IDisposable, jeżeli jest on dostępny w wykorzystywanych obiektach	-